Linköping Electronic Conference Proceedings No. 223

Asian Modelica & FMI Conference Proceedings

September 20–22, 2026

Beihang University Hangzhou Campus
Hangzhou, China



Linköping Electronic Conference Proceedings No. 223

Proceedings of the Asian Modelica & FMI Conference
Hangzhou, China, September 20–22, 2026

Editors:
Prof. Martin Sjölund (Linköping University) and Dr. Dirk Zimmer (DLR)

Published by:
Modelica Association and Linköping University Electronic Press

ISBN: 978-91-8118-699-4 (PDF)
Series: Linköping Electronic Conference Proceedings, No. 223
ISSN: 1650-3686
eISSN: 1650-3740
DOI: https://doi.org/10.3384/ecp223

Organized by:

Beihang University (Hangzhou Campus)
166 Shuanghongqiao St, Pingyao,
Yuhang District, Hangzhou, China

Nanjing Yuansi SimTek Co., Ltd.
Nanjing, China

in cooperation with
the Modelica Association
c/o PELAB, IDA,
Linköpings Universitet
S-58183 Linköping
Sweden

Conference location:
Beihang University Hangzhou Campus
166 Shuanghongqiao St, Pingyao,
Yuhang District, Hangzhou, China

Copyright © Modelica Association, 2026



Dear Participants of the Asian Modelica & FMI Conference,

It is a great pleasure to welcome you to Hangzhou. This is the first Asian Modelica & FMI Conference held in China, and it is the first time that our conference series visits this country at all. The Modelica and FMI community has grown into a truly international community, with strong groups of users, developers and teachers in many parts of the world. This conference is one of the places where that community now meets: to share the latest work, to learn from each other, and to keep our open standards connected to the people who use them.

The user base for Modelica and FMI in China is very large. Local tools, teaching material and industrial demand exist across essentially all sectors. For a long time, however, the Modelica Association itself has been underrepresented here, and many Modelica- and FMI-related efforts remained somewhat fragmented. We shall ensure that progress achieved in Asia flows back into the international standards so that the entire world can benefit. This conference is a concrete step in that direction.

It is organized by Beihang University and Nanjing Yuansi SimTek Co., Ltd. in cooperation with the Modelica Association. We want to thank the team of Prof. Shichun Yang at Beihang for hosting us on this campus, and all colleagues on the organizing committee — from tool vendors, industry and academia, including members from Japan and Korea — for the tremendous effort that went into putting this event together.

As in Lucerne last year, this is a true Modelica & FMI conference. The programme offers four parallel tracks. Two of them are methods-oriented: Modelica language and tools, with a dedicated session on agentic AI workflows, and FMI together with MBSE and digital twins. Two of them are application-oriented, with a remarkably strong presence of nuclear power generation, alongside hydrogen and heat, energy-system modeling, electric mobility and buildings. It is exciting to see these domains sit next to each other. This kind of dialogue between methods and applications, and between different industries facing similar modeling problems, is exactly what we aim for as a multi-domain community.

A few trends in this volume are hard to overlook.

Artificial intelligence, and in particular agentic AI, is no longer a side topic. There are papers on physics-informed neural networks coupled to Modelica models, on agentic workflows on validated Modelica libraries, and on FMI-based architectures for agentic model-based design. At the same time, the core of our work remains: useful mathematical descriptions of physical systems that can be shared through open standards. Open-source libraries based on open standards are invaluable for both humans and computers. Widely applicable standards for model exchange and co-simulation benefit both. In these dynamic times, we must remain grounded in that mission. The papers in this volume give a very concrete picture of how that interaction is currently taking shape.

FMI is equally present. Fifteen years after FMI 1.0, the standard is the de-facto industrial interface for model exchange and co-simulation, and FMI 3.0 together with layered standards is opening new use cases. You will hear this story in the keynote by Christian Bertsch, and you will see it applied in co-simulation, digital twins and industrial model exchange throughout the FMI & MBSE track.

The application papers, especially in nuclear energy, hydrogen, and vehicle thermal management, show how far Modelica-based modeling has entered industrial design and safety analysis. Yet, we have to be keenly aware about the demands from industry. Hence, the keynote by Dr. Yilin Zhong from BYD will demonstrate what model-based digital engineering looks like in one of the most dynamic sectors of China's industry. Prof. Xu Yangzeng will show how modeling expertise acquired for hydraulic systems transfers into the era of digital twins. Prof. Shichun Yang will address AI safety and the toolchains needed for intelligent vehicles.

We extend our gratitude to all authors and reviewers, to our sponsors, and to the local organizers. We are confident that this first Asian Modelica & FMI Conference in China will showcase outstanding examples of our mission: creating useful mathematical descriptions of the world and making them accessible, executable, and shareable through open standards.

We wish you all an inspiring and productive exchange!

Prof. Shichun Yang
(Dean of the School of Transportation Science and Engineering, Beihang University)



Rui Gao
(Secretary of the Modelica Association / Nanjing Yuansi SimTek)



Dirk Zimmer
(Chairman of the Modelica Association / DLR Institute of Robotics and Mechatronics)



DOI
10.3384/ecp223

Sponsors

Gold Sponsor



Silver Sponsors









Bronze Sponsors













Keynote Speeches

Reflections on AI Safety and R&D Tools for Intelligent Vehicles



Prof. Shichun Yang

Professor and Secretary of the Party Committee of the School of Transportation Science and Engineering, Beihang University, China

The report focuses on AI safety and R&D toolchains for intelligent vehicles. Drawing on the team's extensive research in the system safety of connected and autonomous vehicles (CAVs) and autonomous driving control, it analyzes failure risks in AI perception and decision-making algorithms, as well as functional safety and data security challenges. It further elaborates on a comprehensive safety protection framework tailored for full-scenario autonomous driving. Addressing key industry pain points, the report introduces an end-to-end R&D toolchain for intelligent driving, covering simulation testing and electronic control development platforms, while demonstrating their practical applications in algorithm verification and safety assessment.

Model-based Digital Engineering Practice for Automobiles

Dr. Yilin Zhong

Vice President of BYD Automotive Engineering Research Institute



Leveraging its full-industry-chain advantages, vertical integration, and massive proprietary R&D, BYD has achieved global leadership in new energy vehicle sales. To address the growing complexity of vehicle systems, systematic application of systems engineering methods has become essential across the product lifecycle. This keynote focuses on digital engineering practices with Modelica, exploring three interrelated areas: 1) co-simulation design and validation for highly integrated vehicle systems; 2) high-fidelity and computationally efficient modeling for diverse R&D scenarios; and 3) toolchain integration and virtual validation. These practices collectively validate Modelica's role in complex multidisciplinary system modeling.

The Rise of Computer Intelligence and what it means for System Simulation



Dr. Dirk Zimmer

DLR, Institute of Robotics and Mechatronics, Oberpfaffenhofen, Germany

We are undergoing a transition period where computer intelligence is rising to dominance. It will become widely available to everyone at an unprecedented level of intelligence and also at low cost. Such transition periods are often highly chaotic and short-term predictions may hence be rather inaccurate. Based on first principles, however, we can predict what ultimately will prevail after the transition has taken place. This keynote shall provide such an analysis to answer the most relevant questions from the perspective of system simulation: Will the use of simulation rise or fall? Will there be a need for system simulation? What will be the use of Modelica in the future? Will mathematical modeling still be a thing for humans and for computers? The keynote will also explain why it uses the term computer intelligence rather than artificial intelligence.

DOI
10.3384/ecp223

15 Years Functional Mock-up Interface – a Journey from FMI 1.0 to 3.0 and the Road Ahead



Christian Bertsch
Bosch Research, FMI Project Leader and Modelica Association Board Member

Over the last fifteen years, the Functional Mock-up Interface (FMI) has evolved by addressing real industrial needs for simulation interoperability. What began as a solution mainly for the automotive industry has now firmly established itself as the de-facto industrial standard for tool-independent model exchange and co-simulation across numerous domains. This keynote will trace the journey of FMI from its initial concepts in version 1.0 to the powerful ecosystem of today. The release of FMI 3.0 marks a pivotal milestone, enabling advanced co-simulation, virtual ECUs, and the integration of FMI models into optimization and machine learning workflows. Looking ahead, Layered Standards built on the core FMI framework unlock further domain-specific use cases for virtual engineering.

Xu's Modeling Method: A Foundation for Hydraulic Digital Twins and Intelligent Components



Prof. Xu Yangzeng
Professor, Expert with Special Allowance from the State Council, Editor-in-Chief of "Modern Hydraulic and Pneumatic Handbook"

The Xu's Modeling Method, originating from foundational research for the HyPneu software at Oklahoma State University in the 1980s, provides an intuitive framework for hydraulic system modeling. Its core premise is to align the modeling process with engineers' cognitive paradigms. A key breakthrough was its algorithmic solution to Differential-Algebraic Equation (DAE) problems. The method enables model construction directly from hydraulic schematics by applying Kirchhoff's law at junctions, reducing all components to two fundamental sub-models: hydraulic resistance (throttle) and hydraulic capacitance (cylinder). In the Digital Twin era the implicit state equation serves as a potential mechanism model, and the method offers a theoretical basis for "Hydraulic Intelligent Components".

Organizing Committee

Conference Co-Chairs
Prof. Shichun Yang, Beihang University, China
Dr. Rui Gao, Nanjing Yuansi SimTek Co., Ltd., China
Dr. Dirk Zimmer, German Aerospace Center (DLR), Germany

Program Chair
Prof. Yanfang Liu, Beihang University, China

Editorial Chair
Prof. Martin Sjölund, Linköping University, Sweden

Local Arrangement Co-Chairs
Prof. Haiying Lin, Beihang University, China
Prof. Jian Wang, Beihang University, China

Publicity Co-Chairs
Prof. Weilin Li, Northwestern Polytechnical University, China
Dr. Ou Haiying, China SAE Automobile Model Digital Engineering Branch, China
Dr. Hong Lü, State Key Laboratory of Nuclear Power Safety Technology and Equipment, China

Organizers
Beihang University (Hangzhou Campus)
Nanjing Yuansi SimTek Co., Ltd.
in cooperation with the Modelica Association

Advisory Board

Prof. Lin Zhang, Beihang University, China
Dipl.-Math. Christian Bertsch, Robert Bosch GmbH, Germany
Dr. Yutaka Hirano, Hirano Research Laboratory, Japan
Dr. Dai Araki, Toshiba Digital Solution, Japan
Dr. Daeoh Kang, iVH, Korea
Prof. Tielong Shen, Dalian Univ. of Technology & Emeritus Professor (Sophia Univ.)

Supporters

State Key Laboratory of Nuclear Power Safety Technology and Equipment

Program Committee

Johan Andreasson, Modelon AB, Sweden
Dai Araki, Toshiba Corporation, Japan
Bernhard Bachmann, Hochschule Bielefeld (HSBI), Germany
Christian Bertsch, Robert Bosch GmbH, Germany
Johannes Brunnemann, XRG Simulation GmbH, Germany
Dag Brück, Dassault Systèmes AB, Sweden
Francesco Casella, Politecnico di Milano, Italy
Guangliang Chen, Harbin Engineering University, China
Shuo Cheng, Beihang University, China
Clément Coïc, Siemens Healthineers AG, Germany
Mike Dempsey, CLAYTEX Services Ltd, United Kingdom
Hilding Elmqvist, Mogram AB, Sweden
Degang Fu, Midea Group Co., Ltd, China
Leo Gall, LTX Simulation GmbH, Germany
Rui Gao, Nanjing Yuansi SimTek Co., Ltd, China
Xinghai Gao, Beihang University, China
Yutaka Hirano, Hirano Research Laboratory, Japan
Daeoh Kang, IVH Co., Ltd, South Korea
Oliver Lenord, Robert Bosch GmbH, Germany
Da Li, Northwestern Polytechnical University, China
Yanfang Liu, Beihang University, China
Hong Lv, State Key Laboratory of Nuclear Power Safety Technology and Equipment, China
Pierre Mai, PMSF IT Consulting, Germany
Iakov Nakhimovski, Symoptia AB, Sweden
Martin Otter, DLR, Institute of System Dynamics and Control, Germany
Haiying Ou, China SAE Automobile Model Digital Engineering Branch, China
Adrian Pop, Linköping University, Sweden
Martin Sjölund, Linköping University, Sweden
Li Sun, Shanxi Research Institute of Huairou Laboratory, China
Wilhelm Tegethoff, TLK-Thermo GmbH, Germany
Hubertus Tummescheit, MBI LLC, United States
Xiang Wang, Harbin Engineering University, China
Zhuoqi Wang, Beihang University, China
Stefan Wischhusen, XRG Simulation GmbH, Germany
Xinhua Xu, Huazhong University of Science and Technology, China
Qi Yang, Northwestern Polytechnical University, China
Siqiang Yi, Nanjing Yuansi SimTek Co., Ltd, China
Yanzhao Zhao, Nanjing Shuzi Shikong Tech. Lt. Co., China
Dirk Zimmer, DLR, Germany
Wangda Zuo, Pennsylvania State University, United States

DOI
10.3384/ecp223

Content

Sponsors 5
Keynote Speeches 6
Organizing Committee 7

Modelica Technology & AI 13
Fluid Simulation Performance 13
A Novel Parametric Reduced-Order Modeling Method for Transient Prediction of Modelica-Based Brayton Cycles 13
Research on Simulation Efficiency of Supercritical Carbon Dioxide Brayton Power Generation System Based on Mo... ... 21
Language & Tools 31
Robust solution of inverse static problems in Modelica: the case for the initialSimplified() operator 31
Large Network Examples for the Generation of Modular Simulation Code from Modelica 39
SMC: A Cloud-Deployable Modelica Frontend Service in Go 49
AI with Agentic Workflows 57
Research on PINN Data-Driven Modeling Method Based on Modelica Model 57
From Intent to Action: Agentic AI on Validated Modelica Libraries 65
Agentic Model-Based Design with FMI: A Two-Axis Interoperability Architecture 75
Modeling Methods & Libraries 83
Theoretical Reconstruction of Xu's Digital Twin Modeling Method and Its Integration with the Modelica Environment ... 83
Linearizing Position Constraints in Rigid Multibody Dynamics Using Direction Cosine Coordinates: Method and Mo... .. 93
Learning the Missing Physics Locally with Modelica and Wolfram Language 103
Fluids & Media 113
Closed-Form Local Stability of a Pressure-Phase-Coordinate Control-Volume Model 113
A Brief Methodological Reconsideration of Modelica-Based Hydraulic System Modeling and Xu's Method 127

FMI & MBSE 137
FMI, Learning & Control 137
FMU-based Training of Physics-Informed Neural Nets for Control 137
Industrial Exchange 149
Simulation and Verification of Marine Power System Based on Data, Models, and Mock-ups 149
Co-Simulation 157
An FMI-Based Multiscale Workflow for Multi-Scenario Local Accident Assessment in LBE Reactor Rod Bundles 157
High-Pressure Liquid–Gas Ejector Models for System Simulation in Modelica 167
An FMI Co-Simulation Framework for Day-ahead Optimization of District Cooling Systems Based on Modelica 175
MBSE 185
An Equation-Oriented Modeling Framework for Thermodynamic Cycle Systems toward MBSE: A Case Study on S... .. 185
A Three-Level MBSE Logical Design Framework and Requirements-Traceability Validation for ICF Ultrafast Imag... .. 193
Research on Multi-disciplinary Unified Modeling and Simulation Technology of Hualong One System 203
Digital Twins 211
An FMI-Based Fast Thermal Prediction FMU for Real-Time Digital Twins of Data Center Server Rooms 211
A Modelica-FMI digital twin for reinforcement learning-based NOx emission control of boilers under deep peak sh... ... 217

Energy 227
Nuclear Power Systems 227
A Modelica-based Fluid Library for Small Modular Reactors: Design and Verification 227
Research on Dynamic Simulation of Nuclear Power Steam Generator Based on Modelica 237
Modelica-based Modeling and Simulation of an SPWR Nuclear Steam Supply System 245
Reactor Cycles 253
Modelica-Based Parametric Analysis of Steady-State and Transient Thermal-Hydraulic Characteristics of PCHEs in... .. 253
Modelica-Based Dynamic Analysis of an HTGR Combined Cycle System under Negative Reactivity Disturbance: 261
Hydrogen & Heat 271
Dynamic Modelling of a PEM Electrolyser and Metal Hydride Hydrogen Storage System 271
Waste to Hydrogen system dynamic modelling 283
Development of an FMI-Oriented One-Dimensional Transient Heat Transfer Model for Separated Heat Pipes 295
Energy System Modeling 301

Research on Modeling and Simulation of Integrated Energy System Platform Based on OpenModelica 301
Modelica-Based Performance Analysis of an Organic Rankine Cycle under Marine Hot- and Cold-Source Variations 311
Modelica-Based Simulation of Hydrodynamics and Thermomechanical Stress in Water-Wall Tubes of a 1000 MW 317

(Electric) Mobility & Buildings 329

Digital Engineering 329
Automobile Model Digital Engineering Practice 329
22-DOF Full-Coupled Vehicle Dynamics Modeling and Control Based on Modelica 337

Drives & HIL 347
Modelica-Based Modeling of PMSM Control Methods 347
Hardware-in-Loop Testing of an Induction Machine Drive System 357

Buildings & HVAC 363
Dynamic Load Calculation and Analysis of Multi-Zone RC Thermal Network Based on Modelica 363
Modelling of a Coupled PV and ITES Air Conditioning System Based on MWorks and its Operation Analysis for a 371
Control Strategy for a High-Efficiency Refrigeration Plant Based on Dymola Simulation: A Case Study of a Xiame... ... 379

Control & Validation 387
Coordinated Pressure Control Strategy Integrating Feedforward Compensation and Integral Freezing for Biosafety 387
Simulation and Experimental Verification of Temperature Rise in an Oil-Cooled Motor Based on Amesim 393

Thermal Management 399
Calibration workflow development using FMI-based reduced-order models for NEVs thermal management 399
Thermal Simulation of a Low-Voltage Motor Based on a Thermal Resistance Network 409
Model-Based Thermal Management for Electric Vehicles: Modeling, Simulation, and Control 417

DOI
10.3384/ecp223

Author Index

Numbers are starting pages of papers.

Andreasson, Johan 65
Bai, Xinyu 185
Baricco, Marcello 271
Bascetta, Luca 137
Bi, Shengshan 217
Bolognese, Michele 271, 283
Bonamigo, Giulia 271
Borgogna, Alessia 283
Bortoff, Scott A 113
Casella, Francesco 31
Chen, Chaoxin 193
Chen, Guangliang 157
Chen, Guobing 149
Chen, Jingke 329
Chen, Junhua 317
Chen, Junjie 399
Chen, Liping 93
Chen, Zhixu 329, 417
Cheng, Jie 167
Cheng, Shuo 393
Chu, Tianyu 317
Coïc, Clément 103
De Pascali, Matteo Luigi 31
Dematteis, Erika Michela 271
Deshpande, Vedang 113
Ding, Ming 261, 311
Dong, Peng 393
Dong, Xiaomeng 301
Du, Huabing 193
Duan, Qing 217
Fan, Guangming 167
Fang, Can 317
Feng, Haihua 75
Ferretti, Gianni 137
Gao, Rui 49, 175, 399
Guo, Dong 317
Guo, Fei 371
Guo, Haobo 211
Guo, Yufan 337
Haiying, Ou 409
Hannebohm, Philip 31
Hao, Xiangmiao 211
Hong, Hao 227, 237, 301
Hou, Junyang 399
Hu, Lin 193
Huang, Jiali 317
Jiang, Qingfeng 245
Jiao, Guanghui 185
Ju, Peng 295
Laughman, Christopher R 113
Lei, Xiangli 83
Li, Chengen 417
Li, Da 399
Li, Weilin 399
Li, Weiwu 75
Li, Wenji 301
Li, Xiang 175
Li, Xinxin 295
Li, Youchen 193
Li, Zepeng 83
Liu, Sixin 227, 237
Liu, Yanfang 57, 393
Liu, Yuxi 417
Lu, Guoguang 329
Lu, Yi 317
Luo, Fuhong 203
Lv, Hong 203, 227, 237, 301
Ma, Weidong 175
Martinelli, Emanuele 283
Naik, Ankit Anurag 103
Ou, Haiying 329
Peng, Cuiting 203
Peng, Lei 295
Peng, Minjun 157, 253
Pepe, Maria Rosaria 283
Piao, Yuanhao 417
Piras, Paolo 271
Qi, Yang 347, 357, 399
Qiao, Hongtao 113
Qu, Haojie 261
Qu, Xinhe 261
Quan, Long 83
Ren, Bingtao 337
Rizzi, Paola 271
Salladini, Annarita 283
Sha, Feng 175
She, Luchao 167
Shi, Hongqi 363
Shi, Maqian 329
Sun, Hejiang 379, 387
Sun, Longsheng 49
Sun, Yuan 57
Tan, Han 409
Tang, Xuyao 217
Tang, Yue 399
Tao, Wen-Quan 211
Testi, Matteo 271, 283
Tian, Liwei 371
Urbani, Michele 283
Wang, Bo 127
Wang, Hailong 49
Wang, Heng 57
Wang, Jianjun 167
Wang, Pengfei 245
Wang, Qixin 83
Wang, Shuhan 393
Wang, Xiang 13
Wang, Xiaohu 417
Wang, Xinchen 295
Wang, Xudong 57
Wang, Yiding 137
Wang, Yuhui 379, 387

Wang, Yujie 295
Wu, Haoran 379
Wu, Jie 211
Wu, Wei 57
Wu, Zehua 337
Xia, Genglei 185, 253
Xia, Liqiong 193
Xie, Li 417
Xie, Luotao 149
Xu, Aiguo 49, 217
Xu, Hao 253
Xu, Qingda 49
Xu, Xiangyang 393
Xu, Xiaodong 409
Xu, Xinhua 363, 371
Xu, Yangzeng 83, 127, 149
Xu, Yifan 253
Xu, Yuhao 399
Yan, Xiao 311
Yang, Chuangkang 211
Yang, Shichun 337
Yang, Siyuan 211
Yang, Yuang 347, 357
Yang, Zichun 149
Yao, Yuan 301
Yi, Siqiang 21
Yu, Bing 409
Zeng, Ruirui 149
Zhang, Ao 13
Zhang, Baodi 21
Zhang, Cunhao 393
Zhang, Enyang 49
Zhang, Guo 371
Zhang, Kaixuan 57
Zhang, Wei 317
Zhang, Xiang 157
Zhang, Xing 193
Zhang, Yuandong 253
Zhang, Zhidong 57
Zhao, Xinheng 157
Zhao, Yan 93
Zheng, Jianhua 193
Zheng, Jinhua 193
Zhixu, Chen 409
Zhong, Hongyou 347, 357
Zhong, Xuyang 363
Zhong, Yilin 329
Zhou, Aimin 363
Zhou, Dongyang 217
Zhou, Jinglong 167
Zhou, Tao 185
Zhou, Weihua 371
Zhu, Hongfei 311
Zhu, Qiuyuan 363
Zimmer, Dirk 39
Zou, Ziyi 149

A Novel Parametric Reduced-Order Modeling Method for Transient Prediction of Modelica-Based Brayton Cycles

Ao Zhang [1] Xiang Wang [1]

[1] College of Nuclear Science and Technology, Harbin Engineering University, China, {zhangao, xiang.wang}@hrbeu.edu.cn

Abstract

Repeated transient simulations of Brayton cycles are often required during parameter studies, control assessment, and design iteration, whereas direct Modelica simulation under many conditions can remain inefficient despite its flexibility for system-level dynamic modeling. A parametric reduced-order modeling method is developed for fast transient prediction of Modelica-based Brayton cycles under unseen parameters and new operating conditions. Transient data from a Modelica model of a Helium-Xenon Brayton cycle are used to construct a baseline-regularized Dynamic Mode Decomposition with control reduced-order model, in which state trajectories are formulated as deviations from local steady baselines and measured auxiliary inputs are incorporated through the control channel. A Secondary Dynamic Mode Decomposition strategy is further introduced to generalize reduced-order quantities along the parameter direction. Results show accurate interpolation and stable extrapolative prediction against Modelica reference simulations, supporting rapid transient evaluation of Brayton cycles and offering a new perspective for broader fast transient prediction.

Keywords: Parametric reduced-order modeling, Dynamic mode decomposition with control, Brayton cycle

1 Introduction

Understanding and predicting the dynamic behavior of complex systems remains a central challenge across many scientific and engineering fields. With the rapid growth of computational capability and data availability, increasing attention has been paid to identifying, learning, and inferring system dynamics directly from data (Ljung et al., 2020).

However, most existing data-driven identification methods construct surrogate models from datasets generated for specific systems (Huang et al., 2023), leading to strong case dependence. For dynamical systems of the same essential type but operating under different parameters and conditions, such models often show limited generalization and transferability. How to extract transferable dynamic features from data and further map system behavior across related dynamic systems remains insufficiently explored.

The nuclear Brayton cycle considered in this work is a representative dynamic energy-conversion system. Existing studies have mainly focused on modeling and simulation of specific Brayton-cycle configurations, while model-driven optimization usually relies on repeated parameter enumeration and iterative simulations (Zhang and Wang, 2024). Few studies have examined steady-state and transient generalization across parameterized Brayton cycles from the perspective of intrinsic system dynamics. Addressing this gap may provide a new route for dynamic prediction and design-oriented analysis of Brayton cycles.

From an application perspective, current system-identification studies in nuclear engineering mostly focus on approximating specific objects. Machine-learning methods have been widely used to construct surrogate models for rapid design optimization or parameter prediction, including neural networks, deep learning, and attention-based models applied to once-through steam generators (Lu et al., 2021), supercritical CO_2 Brayton cycles (Xie et al., 2024), and AP1000 units (Chen et al., 2021). Although effective for specific datasets, these methods often require considerable training data and may lack interpretability and reliable extrapolation capability.

Dynamic Mode Decomposition (DMD) provides the methodological basis for the present work. Unlike data-intensive machine-learning models, DMD extracts dominant dynamic modes through spectral decomposition of input–output data, offering an efficient and interpretable reduced-order representation of system behavior (Schmid, 2022). Such modal information is useful for revealing intrinsic dynamic features and supporting transfer across related conditions. Nevertheless, standard DMD has limited capability in representing complex nonlinear responses or input-driven transient processes. Dynamic Mode Decomposition with control (DMDc) (Proctor et al., 2016) extends DMD by incorporating control variables, thereby improving its applicability to transient operating conditions. Parametric DMD methods have also been developed for prediction under varying parameters (Riva et al., 2026), but many existing approaches infer reduced quantities at new parameter conditions through interpolation or other regression methods (Chen et al., 2024; Min et al., 2026), with their predictive capability mainly concentrated within the sampled parameter domain. The evolution of the reduced dynamics along the parameter direction,

particularly for extrapolation beyond sampled conditions, has received comparatively limited attention.

Therefore, a parametric reduced-order modeling (pROM) method based on DMDc is developed in this work for transient prediction of Modelica-based Brayton cycles (Section 2). By combining baseline regularization, auxiliary measured inputs, and Secondary DMD-based parameter-direction generalization, the proposed method (Figure 1) aims to improve prediction under unseen parameters and new operating conditions while preserving an interpretable reduced-order dynamic structure (Section 3).

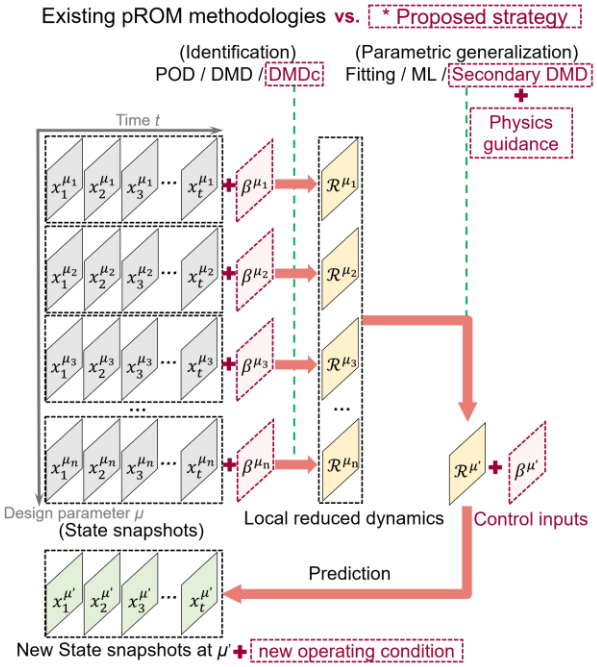
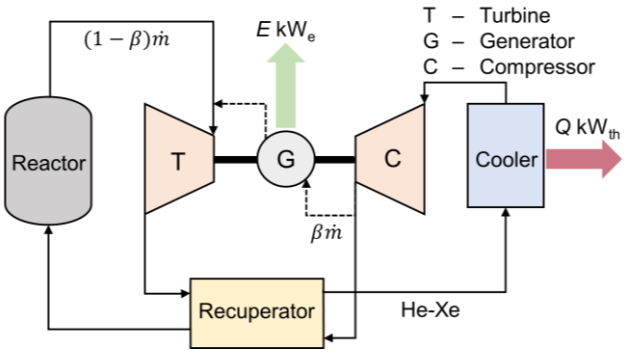


Figure 1. Review of existing approaches and novelty positioning of this work.

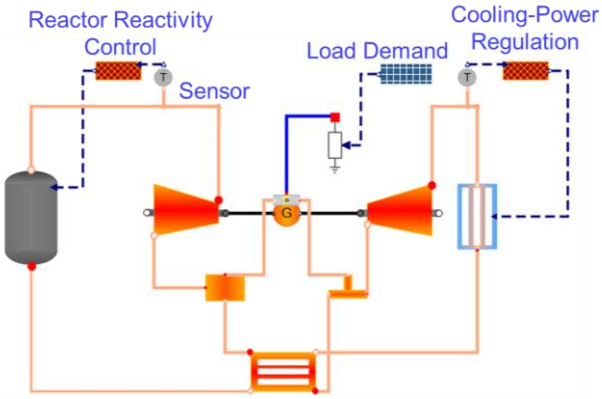
2 Model and Method

2.1 Brayton Cycle Model

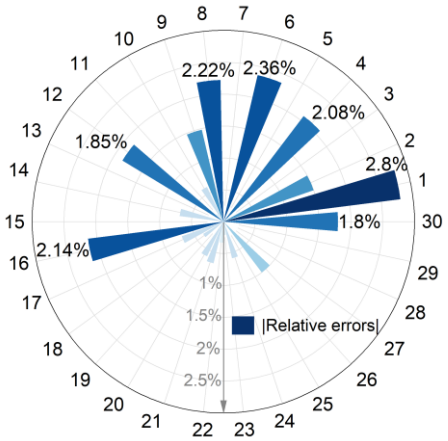
A Modelica-based Brayton cycle model (Figure 2) is employed as the data source for reduced-order transient prediction. The working medium is a Helium-Xenon mixture, while the system model includes the main Brayton-cycle components, such as the reactor heat source, turbine, compressor, generator, recuperator, cooler, and control modules. Turbomachinery performance curves, shaft mechanical efficiency, heat-transfer correlations, and pressure-loss models were also incorporated to describe the coupled thermal-fluid and mechanical behavior of the cycle. With the model reliability verified through steady-state and transient validation in previous work (Zhang et al., 2025), the present study focuses on using the generated Modelica transient data to construct and assess the proposed parametric reduced-order prediction method.



(a) Schematic diagram of Brayton cycle.



(b) Visual model of Brayton cycle.



(c) Relative error of key system parameter calculations compared to El-Genk et al. (2007).

Figure 2. Modelica-based model of the Brayton cycle.

2.2 Scenario and Parameter Setup

The transient datasets for pROM construction and validation are generated from the load-control scenarios shown in Figure 3a. The original scenario is used for reduced-order model identification, whereas the new scenario is used to evaluate prediction under a different transient operating condition. As shown in Figure 3b, the minimum cycle temperature T_c is selected as the parameter coordinate, with sampled conditions used for model construction and target conditions used for interpolation and extrapolation tests. This setup allows the proposed pROM to be assessed under both unseen

DOI
10.3384/ecp22313

parameters and a new load-control scenario. All state and input snapshots used for pROM construction were uniformly sampled at an interval of 0.01 s throughout the 400-s transient simulations.

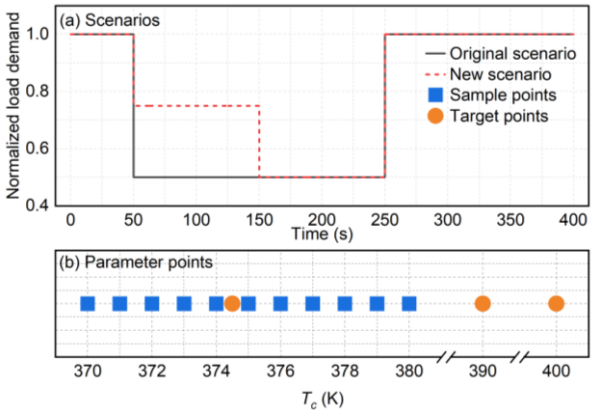


Figure 3. Setup of the scenarios and parameter points.

2.3 Parametric Reduced-Order Modeling Method

A DMDc-based pROM (Figure 4) is constructed from the Modelica-generated transient datasets to predict Brayton-cycle responses at unseen minimum-temperature conditions. Since the system response contains parameter-dependent operating-point shifts, the state and input trajectories are first written in deviation form,

$$\Delta x_k = x_k - \bar{x}, \Delta\beta_k = \beta_k - \bar{\beta} \quad (1)$$

where $\bar{x}$ and $\bar{\beta}$ denote local steady baselines. For unseen target conditions, these baselines are not assumed to be known a priori, but are predicted from the sampled conditions through the same parameter-direction DMD generalization and subsequently added back for full-response reconstruction. Here, the input vector is augmented as $\beta_k = [u_k, \beta_k^+]^T$, where $u_k$denotes the load-control input and β_k^+ represents the measured cooling-power regulation signal. This auxiliary input is introduced to help capture strongly coupled thermohydraulic effects.

The reduced coordinates are obtained by truncated SVD, $z_k = U_r^* \Delta x_k$, and the controlled reduced dynamics are identified as

$$z_{k+1} = Az_k + B\Delta\beta_k \quad (2)$$

Here, A is the reduced state-transition operator, while B maps both load-control and auxiliary-input effects to the reduced state. The corresponding spectral constituents Φ, Ω, and b, representing the dynamic modes, eigenvalues, and mode amplitudes, respectively, characterize the dominant reduced dynamics and are included among the quantities generalized across parameter conditions. After DMDc models are identified at sampled T_c conditions, Secondary DMD is used to generalize the reduced-order quantities along the parameter direction and estimate the model at a target T_c. Specifically, the reduced quantities identified at ordered T_c conditions are treated as a parameter-domain sequence, allowing Secondary DMD to model their evolution along the parameter direction rather than in physical time. Its extrapolative reliability relies on sufficiently coherent variation of the reduced dynamics and may deteriorate near abrupt regime transitions or bifurcations. The predicted reduced response is finally mapped back to the physical state by $x_k \approx U_r z_k + \bar{x}$, enabling interpolation and extrapolation of Brayton-cycle transients under new operating conditions.

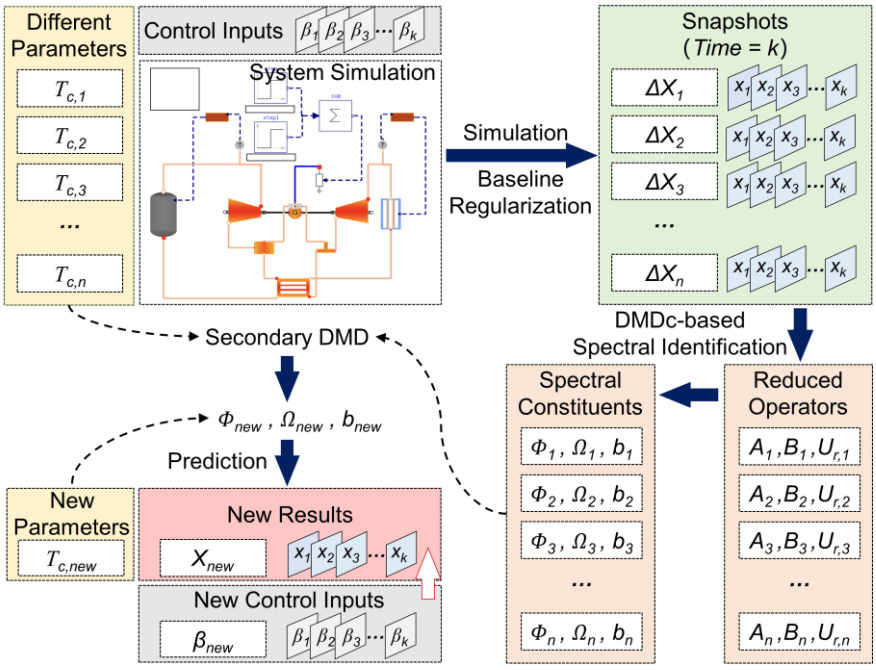


Figure 4. pROM framework for parameterized Brayton-cycle transient prediction.

3 Calculation and Discussion

The proposed DMDc-based pROM is evaluated under both the original and new load-control scenarios. Three target minimum-temperature conditions are considered, including one interpolative condition, $T_c = 374.5$ K, and two extrapolative conditions, $T_c = 390.0$ K and $T_c = 400.0$ K. The Modelica simulation results are used as reference data for assessing the predicted transient responses.

Figure 5 compares the relative norm errors of DMD and the baseline-regularized Δ-DMD method under the original and new load-control scenarios. Here, DMD and Δ-DMD denote Secondary-DMD-based parameter generalization applied to unregularized and baseline-regularized reduced-order quantities, respectively. For the interpolative target, both methods provide acceptable prediction accuracy because the target condition is located inside the sampled parameter range. Here, r_t denotes the temporal rank used for DMDc identification, while r_p denotes the parametric rank retained in Secondary DMD for parameter-direction generalization. In extrapolation, however, DMD shows clear error growth, whereas Δ-DMD achieves relative norm errors (< 1%) at low r_t by separating local steady baselines from transient deviations. The error maps also indicate that increasing the temporal rank does not necessarily improve prediction accuracy, since higher-rank models may amplify local fitting details and reduce extrapolative stability. Based on this rank-dependent error comparison, r_t = 2 is selected as a representative low-rank configuration for the subsequent time-domain results. The time-resolved relative errors of all predicted system variables at the farthest extrapolation condition T_c = 400 K are further shown in Figure 6, providing a system-level assessment under both the original and new scenarios.

Time-domain predictions of the normalized shaft speed ω_s further confirm the advantage of baseline regularization (Figure 7). Under both the original and new scenarios, Δ-DMD closely follows the Modelica reference responses at $T_c = 374.5$ K, 390.0 K, and 400.0 K, whereas DMD reproduces the interpolative case but loses reliability in extrapolation, especially at $T_c = 400.0$ K. This comparison indicates that baseline regularization is essential for preserving the main transient trend outside the sampled parameter range.

The effect of measured auxiliary inputs is evaluated using reactor power P_R and turbine outlet temperature $T_{o,tur}$ as representative variables (Figure 8). Including β^+ leads to closer agreement with the Modelica reference results, whereas omitting β^+ introduces larger deviations in the predicted transient responses. This result shows that auxiliary measured inputs help represent nonlinear thermohydraulic regulation effects through the DMDc control channel.

The extrapolative error trend is further examined using several near-extrapolation test points along the T_c direction (Figure 9). The relative norm errors at these reserved test points are fitted as a function of T_c using a polynomial regression, from which the fitted error trend and its 95% confidence interval are extrapolated to the farther target points. The predicted error increases gradually with T_c and approaches the illustrative tolerance in the far-extrapolation region, indicating that such near-extrapolation tests can help identify when additional Modelica simulations should be added to refresh the pROM dataset.

Computational efficiency is further evaluated against an LSTM surrogate (Figure 10). The proposed pROM reduces the training time from 217.80 s to 3.63 s and the prediction time from 1.77 s to 1.20 s per case, while maintaining lower relative norm errors, particularly under extrapolative conditions. The time-series comparison at $T_c = 400$ K further confirms that this computational advantage is achieved while preserving accurate transient prediction under both load-control scenarios.

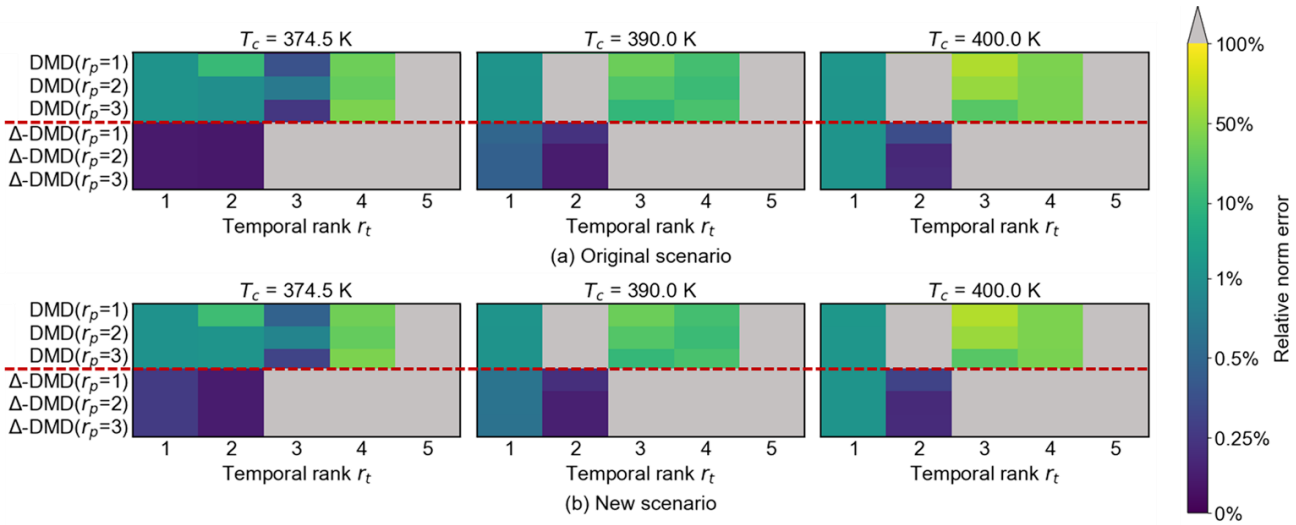


Figure 5. Prediction errors under different T_c conditions and temporal ranks (r_t).

DOI
10.3384/ecp22313

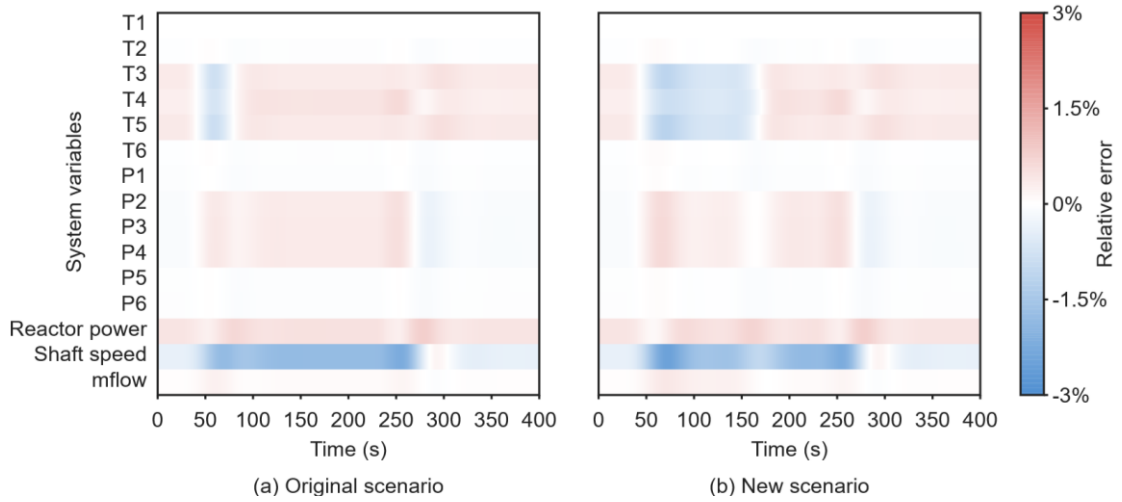


Figure 6. Time-resolved relative errors of all predicted system variables at $T_c = 400$ K ($r_{t,p} = 2$).

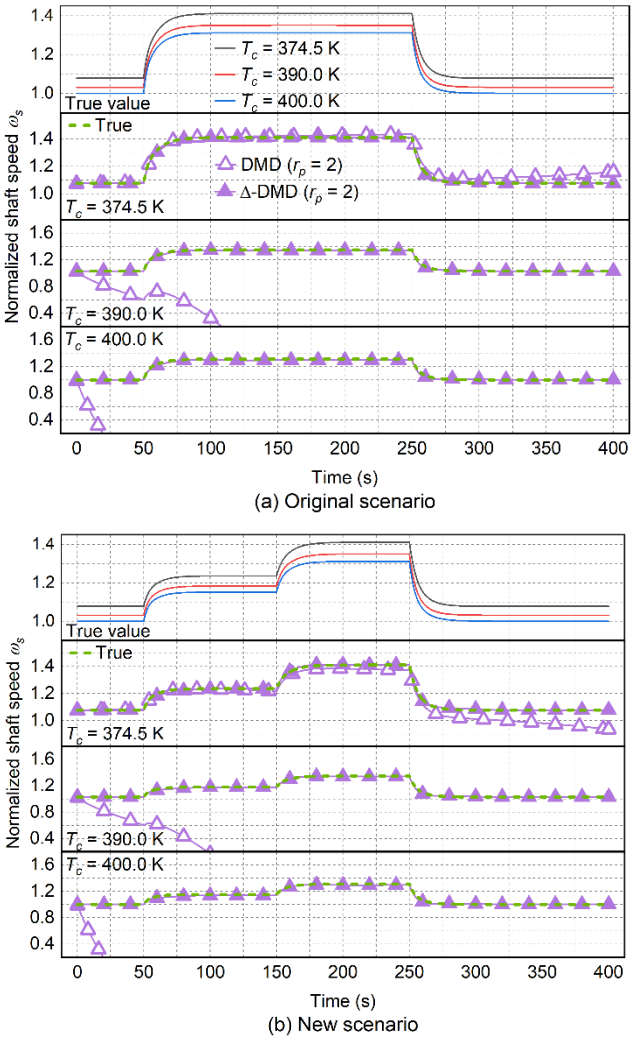


Figure 7. Time-domain prediction of a representative system variable ($r_t = 2$).

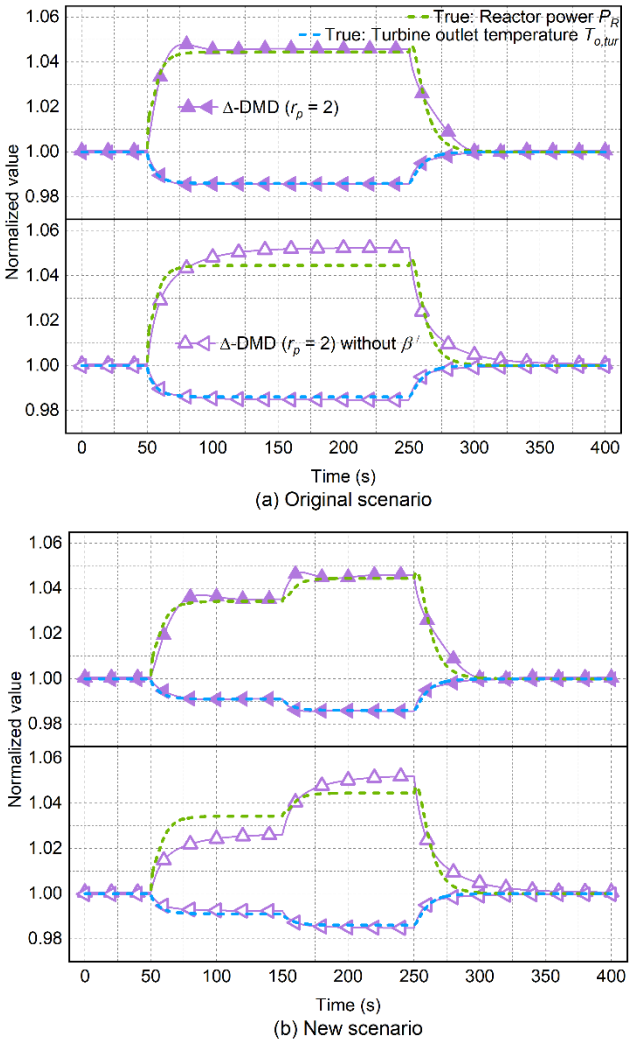


Figure 8. Effect of measured auxiliary inputs β^+ at $T_c = 400$ K ($r_{t,p} = 2$).

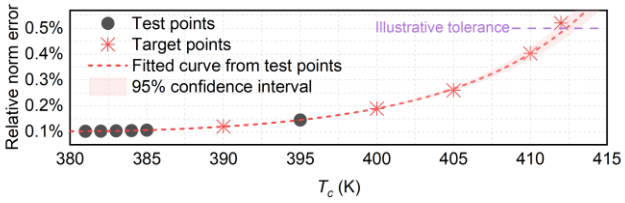


Figure 9. Extrapolation confidence assessment for the Helium-Xenon Brayton cycle.

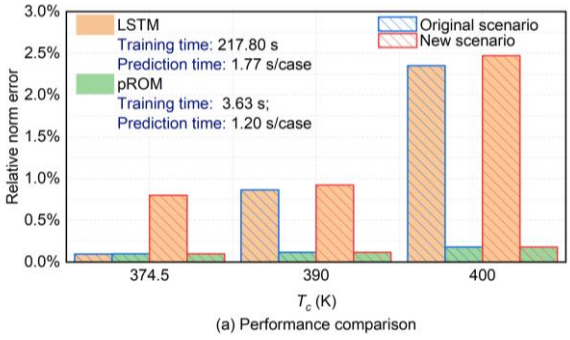


Figure 10. Comparison between pROM and LSTM.

4 Conclusions

A novel DMDc-based parametric reduced-order modeling method was developed for fast transient prediction of Modelica-based Brayton cycles under unseen minimum-temperature conditions and new load-control scenarios. The main conclusions are summarized as follows:

(1) The proposed DMDc-based pROM, combined with Secondary DMD for parameter-direction generalization, enables transient prediction of Modelica-based Brayton cycles under unseen T_c conditions and new load-control scenarios, achieving relative norm errors below 1% in the interpolation and extrapolation cases.

(2) Baseline regularization and measured auxiliary inputs are both important for prediction accuracy. By removing local steady offsets, Δ-DMD provides more accurate responses at low temporal ranks, while auxiliary inputs improve the representation of strongly coupled thermohydraulic transients.

(3) The additional error-trend analysis indicates that near-extrapolation tests can help assess extrapolation reliability and judge whether new Modelica simulations are needed to update the pROM dataset.

Future work will further compare the operator regression performance of Secondary DMD with other regression methods, such as linear and radial basis function regressions. The applicability of the proposed framework to multi-parameter Brayton cycles will also be investigated to assess its scalability under more complex operating and design conditions.

References

Chen, G. *et al.* (2024) "A fast POD prediction method for hydrogen leakage at different pressures", *International Journal of Hydrogen Energy*, 49, pp. 1391–1404. Available at: https://doi.org/10.1016/j.ijhydene.2023.09.282.

Chen, Y. *et al.* (2021) "Research on Simulation and State Prediction of Nuclear Power System Based on LSTM Neural Network", *Science and Technology of Nuclear Installations*. Edited by A. Serikov, 2021, pp. 1–11. Available at: https://doi.org/10.1155/2021/8839867.

El-Genk, M.S. and Tournier, J.-M. (2007) "Noble-Gas Binary Mixtures for Closed-Brayton-Cycle Space Reactor Power Systems", *Journal of Propulsion and Power*, 23(4), pp. 863–873. Available at: https://doi.org/10.2514/1.27664.

Huang, Q. *et al.* (2023) "A review of the application of artificial intelligence to nuclear reactors: Where we are and what's next", *Heliyon*, 9(3), p. e13883. Available at: https://doi.org/10.1016/j.heliyon.2023.e13883.

Ljung, L. *et al.* (2020) "Deep Learning and System Identification", *IFAC-PapersOnLine*, 53(2), pp. 1175–1181. Available at: https://doi.org/10.1016/j.ifacol.2020.12.1329.

Lu, Q. *et al.* (2021) "Prediction method for thermal-hydraulic parameters of nuclear reactor system based on deep learning algorithm", *Applied Thermal Engineering*, 196, p. 117272. Available at: https://doi.org/10.1016/j.applthermaleng.2021.117272.

Min, G. *et al.* (2026) "A novel reduced-order model and two-stage optimization strategy for high-efficiency predicting thermal-hydraulic behavior in a 6×6 fuel bundle", *Progress in Nuclear Energy*, 191, p. 106032. Available at: https://doi.org/10.1016/j.pnucene.2025.106032.

Proctor, J.L., Brunton, S.L. and Kutz, J.N. (2016) "Dynamic Mode Decomposition with Control", *SIAM Journal on Applied Dynamical Systems*, 15(1), pp. 142–161. Available at: https://doi.org/10.1137/15M1013857.

Riva, S. *et al.* (2026) "A Comparative Study of Parametric Dynamic Mode Decomposition Algorithms on Thermal-Hydraulics Case Studies", *Nuclear Technology*, pp. 1–25. Available at: https://doi.org/10.1080/00295450.2025.2575701.

Schmid, P.J. (2022) "Dynamic Mode Decomposition and Its Variants", *Annual Review of Fluid Mechanics*, 54(1),

pp. 225–254. Available at: https://doi.org/10.1146/annurev-fluid-030121-015835.

Xie, L. *et al.* (2024) "Design, optimisation and evaluation of the S-CO2 Brayton cycle for marine low-speed engine flue gas", *Case Studies in Thermal Engineering*, 61, p. 105161. Available at: https://doi.org/10.1016/j.csite.2024.105161.

Zhang, A., Cammi, A. and Wang, X. (2025) "System-level data-driven identification of complex closed Brayton cycles via dynamic mode decomposition", *Applied Thermal Engineering*, p. 127438. Available at: https://doi.org/10.1016/j.applthermaleng.2025.127438.

Zhang, A. and Wang, X. (2024) "Development of Modelica-based one-dimensional thermodynamic cycle library and its application in simulation and multi-objective optimization of a He–Xe closed-Brayton-cycle system", *Progress in Nuclear Energy*, 172, p. 105205. Available at: https://doi.org/10.1016/j.pnucene.2024.105205.

DOI
10.3384/ecp22313

Research on Simulation Efficiency of Supercritical Carbon Dioxide Brayton Power Generation System Based on Modelica

Siqiang Yi[1] Baodi Zhang[1]
[1] Nanjing Yuansi SimTek Co., Ltd.
siqiang.yi@simtek.cc; baodi.zhang@simtek.cc

Abstract
Existing Modelica libraries have demonstrated effective modeling of sCO2 systems. However, the influence of momentum-equation formulation on solver behavior has not been systematically quantified under an identical system model and consistent numerical settings. This study develops a one-dimensional dynamic Modelica model of an sCO2 Brayton power system and systematically compares four momentum-equation formulations within the same system under identical numerical settings, quantitatively examining how state selection, flow inertia, and pressure feedback affect solver steps, Jacobian evaluations, and error-dominant variables. The conventional steady-state algebraic momentum formulation is compared with three differential formulations incorporating flow inertia, and their computational performance is assessed under both steady-state and transient conditions. For the investigated system model and operating cases, the differential formulations containing a mass-flow derivative term completed the transient simulations, whereas the conventional steady-state algebraic formulation did not converge under the same numerical settings. Furthermore, by introducing a first-order inertia term for the average pipe flow rate together with inlet/outlet pressure feedback, the steady-state computation time is reduced from 750.5 s to 73.2 s, the number of transient solver steps decreases from 7878 to 3967, the number of Jacobian evaluations is reduced by 52.8%, and the occurrence frequencies of the reported error-dominant variables decrease by approximately 62% to 77% relative to Fluid Equation 2. These results provide a quantitative, model-specific characterization of how state selection, flow inertia, and pressure feedback affect solver behavior under the tested conditions.
Keywords: Supercritical carbon dioxide, Brayton cycle, Modelica, Computational efficiency, Transient analysis

1 Introduction
Because of its outstanding properties, supercritical carbon dioxide has attracted extensive attention from researchers in many fields. Its advantages include high density, low compression work, high cycle efficiency, and equipment compactness, making it highly attractive for applications such as nuclear energy, concentrating solar power, industrial waste-heat recovery, and marine propulsion [1][4][5]. However, the thermophysical properties of sCO2 vary dramatically near the critical region, and the strong coupling among the heat exchanger, compressor, turbine, and recuperator leads to highly complex transient behavior. These strongly coupled transients motivate careful examination of model formulation and solver behavior in studies of startup, shutdown, load following, and accident scenarios [1][3][4]. The acausal modeling paradigm and object-oriented reuse mechanism of Modelica have been applied to dynamic analyses of complex thermal-hydraulic systems, including the cited sCO2 studies [1][2].

2 Literature Review
2.1 Modelica Modeling of sCO2 Systems
Modelica modeling of sCO2 systems has expanded from single-component applications to system-level, multi-physics, and control-oriented applications. Ferrero et al. developed a dynamic Modelica model of an sCO2 energy conversion system for the EU-DEMO fusion device and introduced a PI controller for variable-load tracking, demonstrating that Modelica is suitable not only for thermodynamic component modeling but also for integrated simulation of control-system design and thermal process behavior [1]. This work provides an example of integrated dynamic and control-oriented sCO2 modeling in Modelica.
In nuclear-energy-coupled applications, Guo et al. developed a supercritical carbon dioxide subchannel model in Modelica for more detailed thermal-hydraulic analysis of the core of a small reactor [2]. This study extends the cited Modelica applications from system-level cycle simulation to core-scale flow and heat-transfer analysis. Similarly, Zhang et al. coupled a lead-bismuth-cooled fast reactor with an sCO2 Brayton cycle, established a high-fidelity multi-domain Modelica model, and performed accident-condition analyses such as LOFA

and UTOP [5]. Together, the cited studies provide examples of safety analysis, parametric sensitivity analysis, and coupled-system studies implemented in Modelica.

For loop systems and heat-transfer equipment, Pegallapati et al. used Modelica to establish a one-dimensional dynamic model of a fixed-charge sCO2 natural-circulation loop. The mass, momentum, and energy equations were discretized using the method of lines and finite-volume techniques, and the suitability of the model for transient processes was verified [3]. Qin et al. developed a one-dimensional dynamic Modelica model for a printed-circuit heat-exchanger cooler in an sCO2 power cycle and compared the effects of different numbers of control volumes and equation simplification approaches on simulation accuracy and computational time [4]. These studies provide examples of dynamic, mechanism-based Modelica models for key sCO2 components and of comparisons with experiments or higher-dimensional numerical results.

In summary, Modelica modeling of sCO2 systems has covered several levels: cycle-level modeling such as Brayton cycles and power-conversion systems; component-level modeling such as heat exchangers and natural-circulation loops; core- or subchannel-level thermal-hydraulic modeling; coupled modeling of nuclear reactors and sCO2 power cycles; and control and accident-analysis modeling [1]-[5]. Together, these studies demonstrate the breadth of Modelica applications to sCO2 systems and motivate more focused investigation of how specific equation formulations affect numerical performance.

2.2 Research Progress in Modelica Simulation Efficiency

Compared with application-oriented modeling studies, research on Modelica simulation speed and solution efficiency has focused more on general methods and tool-level issues. Casella and Guironnet proposed the ScalableTestGrids benchmark suite for evaluating Modelica method performance in large-scale power-system scenarios, indicating that compilation and simulation scalability for large models has become an explicit research topic within the Modelica community [6]. Agosta et al. further proposed the MARCO compiler for ultra-large-scale models, emphasizing improvements in translation and code-generation efficiency at the scale of tens of millions of equations [7]. These studies show that the efficiency issue in Modelica has indeed received significant attention, but the main focus has been compiler architecture, toolchain scalability, and large generic models rather than sCO2 systems specifically.

In terms of the numerical performance of thermo-fluid systems, Flesch et al. pointed out that low-flow or near-zero-flow conditions can significantly degrade the simulation performance of fluid-system models, and they discussed the effects of model modifications, tool settings, and solver strategies on performance improvement [8]. This observation motivates examination of low-flow and transient solver behavior in the present sCO2 model. Bortoff et al., from the standpoint of fluid-property functions, proposed more efficient coordinate transformations and spline-approximation methods, emphasizing that high numerical efficiency can still be maintained near the supercritical region, the two-phase region, and the saturation curve [9]. Because thermophysical-property functions are called repeatedly in sCO2 simulations, this work provides relevant context for evaluating numerical performance.

Existing Modelica libraries have demonstrated effective modeling of sCO2 systems. However, the influence of momentum-equation formulation on solver behavior has not been systematically quantified under an identical system model and consistent numerical settings [1]-[5].

In summary, Modelica studies have addressed energy-conversion systems, nuclear-energy-coupled systems, key heat-transfer equipment, natural-circulation loops, subchannel thermal hydraulics, and accident transients [1]-[5]. Prior work has examined benchmark testing, compiler optimization, low-flow performance, and efficient thermophysical-property evaluation in Modelica models [6]-[10]. This study systematically compares four momentum-equation formulations within the same sCO2 Brayton-cycle system under identical numerical settings and quantitatively examines how state selection, flow inertia, and pressure feedback affect solver steps, Jacobian evaluations, and error-dominant variables.

3 Governing Models and Theoretical Formulation

3.1 Pipe Model

In system-level simulation, pipes are generally represented using one-dimensional distributed-parameter models. The basic assumptions include the following: velocity, temperature, and pressure are represented by cross-section-averaged values; the primary variations are distributed along the axial direction; and geometric parameters such as cross-sectional area A, hydraulic diameter Dh, and roughness are assumed constant within a single segment.

Conservation Equations

Mass conservation equation

The mass conservation equation and its state-variable form are given in Equations 1 and 2, respectively:

$$G_i - G_{i+1} = \frac{dM}{d\tau} \tag{1}$$

$$\frac{dM_i}{d\tau} = \frac{d\rho_i V_i}{d\tau} = V_i\left(\frac{\partial\rho}{\partial p}\frac{dp_i}{d\tau} + \frac{\partial\rho}{\partial \mathrm{h}}\frac{d\mathrm{h}_i}{d\tau}\right) \tag{2}$$

G_i — mass flow rate of the working fluid at the inlet of pipe segment i; kg/s

DOI 10.3384/ecp22321

G_{i+1} – mass flow rate of the working fluid at the outlet of pipe segment i; kg/s
M_i – mass of the working fluid in pipe segment i; kg
ρ_i – density of the working fluid in pipe segment i; kg/m^3
V_i – volume of pipe segment i;m^3
h_i – specific enthalpy of the working fluid in pipe segment i;kJ/kg

Momentum conservation equation

The momentum conservation equation is given in Equation 3:

$$p_i - p_{i+1} - \Delta H \rho_i g - \Delta p_f = 0 \quad (3)$$

where Δp_f represents the frictional pressure-loss component along the flow path.

Energy conservation equations (Equations 4 and 5)

$$\frac{dE}{d\tau} = G_i h_i - G_{i+1} h_{i+1} + Q \quad (4)$$

$$\frac{dE}{d\tau} = V_i\left(\rho_i \frac{dh}{d\tau} + h_i \frac{d\rho_i}{d\tau} - \frac{dp_i}{d\tau}\right) \quad (5)$$

where:Qheat transfer between the working fluid and the heat-transfer tube wall.J/s

Pressure Drop and Heat Transfer

Frictional Pressure Drop Along the Flow Path

The frictional pressure drop along the flow path is evaluated using the Darcy-Weisbach relation in Equation 6:

$$\Delta p_f = f(L/Dh)\rho u^2/2 \quad (6)$$

In-Tube Heat Transfer and Thermal Inertia

When the tube-wall temperature differs from the fluid temperature, the heat transfer per unit length is given by Equation 7:$T_w \neq T_f$ –

$$Q = kA(T_o - T_i) \quad (7)$$

Q – heat-transfer rate; W
A – heat-transfer area;m^2
$T_o$tube-wall temperature; K
T_i – in-tube fluid temperature; K

The in-tube heat-transfer coefficient is calculated from the Dittus-Boelter correlation in Equation 8:

$$h_i = 0.023 \frac{\lambda_i}{d_i} Re^{0.8} Pr^{0.4} \quad (8)$$

3.2 Heat-Exchanger Model

The heat-exchanger model is constructed using a pipe-based modeling approach, as shown in Figure **1**.

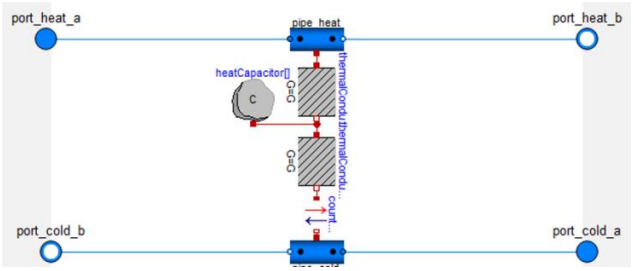


Figure 1. Schematic diagram of heat exchanger model

In the model, the heat-conduction equation for the thermal structure is given in Equation 9:

$$Q_{in} = \frac{2\pi L(t_{in} - t_{out})}{\frac{1}{\lambda} ln\frac{r_{out}}{r_{in}}} \quad (9)$$

where Qin is the heat-conduction rate through the thermal structure, in W; L is the length of the pipe thermal structure, in m; tin is the inner-wall temperature of the thermal structure, in °C; tout is the outer-wall temperature of the thermal structure, in °C; rin is the inner radius of the thermal structure, in m; rout is the outer radius of the thermal structure, in m; and λ is the thermal conductivity, in W m-1 °C-1.

3.3 Compressor and Turbine Models

The compressor and turbine are the core components of the Brayton cycle. After being cooled in the cooler, the working fluid enters the compressor, where it is compressed to a high-pressure state so that its pressure rises to a level close to the turbine inlet condition, thereby preparing it to absorb heat in the reactor and subsequently enter the turbine for expansion and power production. The high-temperature, high-pressure working fluid exits the reactor (or heat source) and enters the turbine, where it expands, experiences a pressure drop and a temperature decrease, and simultaneously drives the turbine blades to rotate and output shaft power. The turbine converts fluid energy (thermal energy and pressure energy) into mechanical energy.

The model supports the input of compressor and turbine performance data in standard tabulated form, including speed characteristic lines, flow-rate versus pressure-ratio characteristics, and efficiency characteristics. Through similarity conversion, the model can adapt to performance calculations under different inlet conditions.

The theoretical turbine model is similar to that of the compressor, except that gas compression is replaced by gas expansion.

(1) Similarity-Conversion Parameters

Compressor performance is converted using the corrected speed and flow in Equations 10 and 11 to eliminate the influence of variations in inlet conditions:

$$ncor = \frac{nTt1}{Tref} \quad (10)$$

$$\dot{m}_{cor}=\frac{\dot{m}\sqrt{T_{t1}/T_{ref}}}{p_{t1}/p_{ref}} \tag{11}$$

where the converted rotational speed and converted flow rate are defined based on the inlet total temperature and inlet total pressure, while the reference state is usually taken as standard sea-level conditions.

(2) Pressure-Ratio Calculation

The pressure ratio is obtained from the performance-map interpolation in Equation 12:

$$\pi=f(n_{cor},\dot{m}_{cor})=f_{map}(n_{cor},\dot{m}_{cor}) \tag{12}$$

where the term denotes the interpolation function for the performance map, usually implemented using bilinear or linear interpolation.f_{map}

(3) Isentropic Efficiency

$$\eta=g(n_{cor},\dot{m}_{cor})=g_{map}(n_{cor},\dot{m}_{cor}) \tag{13}$$

The map-based isentropic efficiency is given by Equation 13; its ratio definition is given in Equation 14:

$$\eta=\frac{h_{t2s}-h_{t1}}{h_{t2}-h_{t1}} \tag{14}$$

(4) Calculation of Outlet Parameters

Outlet total pressure is given in Equation 15:

$$p_{t2}=\pi p_{t1} \tag{15}$$

Outlet total temperature is back-calculated from efficiency using Equation 16:

$$T_{t2}=T_{t1}\left[1+\frac{\pi^{(\gamma-1)/\gamma}-1}{\eta}\right] \tag{16}$$

Power consumption is given in Equation 17:

$$P=\dot{m}\cdot(h_{t2}-h_{t1})=\dot{m}c_p\frac{T_{t1}}{\eta}\left(\pi^{(\gamma-1)/\gamma}-1\right) \tag{17}$$

3.4 Plenum Model

The plenum model represents the mass and energy storage of a finite-volume chamber and therefore determines the transient pressure and enthalpy response between connected components. Because the working fluid is supercritical carbon dioxide, its thermodynamic properties are evaluated with a real-fluid property model rather than an ideal-gas equation of state. Pressure and specific enthalpy are selected as the independent state variables.

(1) Mass Conservation

The mass balance of plenum i is given by Equation 18:

$$\frac{dm}{dt}=\sum_{in}\dot{m}-\sum_{out}\dot{m} \tag{18}$$

The relationship between plenum mass and the state variables is given in Equation 19:

$$\frac{dm}{d\tau}=\frac{d\rho V}{d\tau}=V\left(\frac{\partial\rho}{\partial p}\frac{dp}{d\tau}+\frac{\partial\rho}{\partial h}\frac{dh}{d\tau}\right) \tag{19}$$

(2) Energy Conservation

The plenum energy balance is given in Equation 20:

$$\frac{dU}{dt}=\sum_{in}\dot{m}h-\sum_{out}\dot{m}h+\dot{Q}-p\frac{dV}{dt} \tag{20}$$

For a fixed-volume plenum, Equation 21 applies:

$$\frac{dU}{dt}=\sum_{in}\dot{m}h-\sum_{out}\dot{m}h+\dot{Q} \tag{21}$$

The relationship between internal energy and the state variables is given in Equation 22:

$$U=mu=\rho Vh \tag{22}$$

(3) Equation of State

The pressure and temperature in the plenum are related through the equation of state in Equation 23:

$$\frac{dU}{dt}=V\left[(h\ \rho_p-1)\frac{dp}{dt}+(\rho\ +h_i\rho_h)\frac{dh}{dt}\right] \tag{23}$$

The temperature is given in Equation 24:

$$T=f(p,h) \tag{24}$$

4 Results and Discussion

4.1 Steady-State Performance

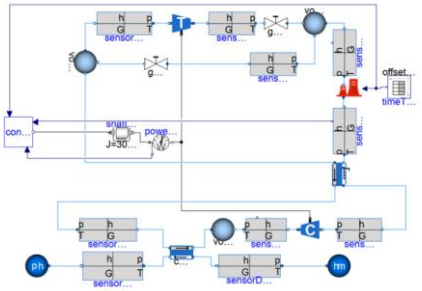


Figure 2. Schematic of the simple recuperated sCO2 system.

The Brayton system is configured as a single-loop cycle using carbon dioxide as the working fluid. After absorbing heat in the reactor, the fluid enters the turbine, where fluid energy is converted into shaft power. The turbine exhaust then passes through the recuperator and the cooler for sequential heat rejection. The cooled fluid subsequently enters the compressor, where mechanical energy is converted back into fluid energy through pressurization. The pressurized fluid re-enters the recuperator for heat recovery and finally returns to the reactor, thereby completing the closed-loop circulation. The corresponding steady-state results are reported in Tables **1** and **2**.

DOI 10.3384/ecp22321

Table 1. Steady-state calculation results.

No	*Location*	*Pressure (MPa)*	*Temperature (°C)*	*Enthalpy (kJ/kg)*	*Mass Flow Rate (kg/s)*
1.Core(Reactor)					
1	Core Inlet	16.3385	669.38	1186.1560	302.8724
2	Core Outlet	16.2347	927.50	1516.3281	302.8724
2.Recuperator					
1	Cold-Side Inlet (from Compressor)	17.1014	83.13	402.6060	302.8720
2	Cold-Side Outlet (to Core)	16.3385	669.38	1186.1560	302.8724
3	Hot-Side Inlet (from Turbine)	8.5327	831.61	1393.9992	302.8712
4	Hot-Side Outlet (to Cooler)	8.2045	175.21	610.4505	302.8720
3.Cooler					
1	Hot-Side Inlet (CO2 In)	8.2045	175.21	610.4505	302.8720
2	Hot-Side Outlet (CO2 Out)	8.1554	37.51	377.2823	302.8720
3	Cold-Side Inlet (Cooling Water In)	4.1022	35.01	150.3228	800.0000
4	Cold-Side Outlet (Cooling Water Out)	4.0000	56.19	238.5981	800.0000
4.Compressor					
1	Compressor Inlet	8.1554	37.51	377.2823	302.8720
2	Compressor Outlet	17.1014	83.13	402.6060	302.8720
3	Shaft Power: 7,669.84 kW (Input)				
5.Turbine					
1	Turbine Inlet	15.7091	927.48	1516.3281	302.8712
2	Turbine Outlet	8.5327	831.61	1393.9992	302.8712

Table 2. Steady-state operating and performance results.

No	*Parameter Name*	*Value*	*Unit*
1	Compressor Inlet Pressure	8.16	MPa
2	Compressor Outlet Pressure	17.10	MPa
3	Turbine Inlet Pressure	15.71	MPa
4	Turbine Outlet Pressure	8.53	MPa
5	Compressor Pressure Ratio	2.10	—
6	Turbine Expansion Ratio	1.84	—
7	Compressor Inlet Temperature	37.51	°C
8	Compressor Outlet Temperature	83.13	°C
9	Turbine Inlet Temperature	927.48	°C
10	Turbine Outlet Temperature	831.61	°C
11	Core Outlet Temperature	927.5	°C
12	CO2 Mass Flow Rate	302.87	kg/s
13	Turbine Output Power	37049.91	kW
14	Compressor Power Consumption	7669.84	kW
15	Net Output Power	29380.07	kW
16	Core Thermal Power	100000.01	kW

No	*Parameter Name*	*Value*	*Unit*
17	Cooler Heat Rejection	70620.21	kW
18	Turbine Isentropic Efficiency	90.00	%
19	Cycle Thermal Efficiency	29.38	%

4.2 Startup and Shutdown Characteristics

The system characteristics under startup and shutdown conditions were simulated. Specifically, the shutdown condition involved reducing reactor power from 100% FP to the steady-state operating condition at 15% FP and then further reducing power to reactor shutdown. The startup condition involved increasing power from zero to the steady-state operating condition at 15% FP, followed by a further increase to stable operation at 100% FP. The principal transient responses are summarized in Figure **3**.

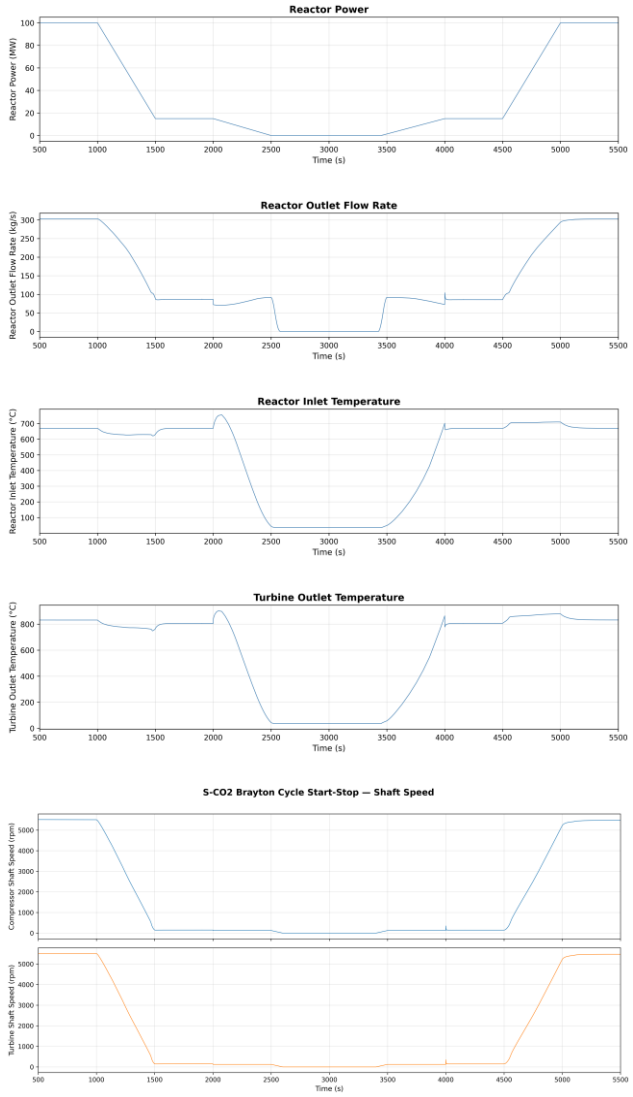


Figure 3. Simulation calculation results.

When reactor power decreased from 100% FP to 15% FP, the control system rapidly reduced the rotational speeds of the turbine and compressor in order to maintain a constant reactor inlet temperature. The overall system flow rate decreased, while the turbine-side temperature remained approximately unchanged. As reactor power was reduced further to zero, the turbine and compressor rotational speeds were gradually decreased to zero. At this stage, the system flow rate approached zero and the overall system temperature dropped rapidly. The corresponding mass-inventory and pressure responses are shown in Figure **4**.

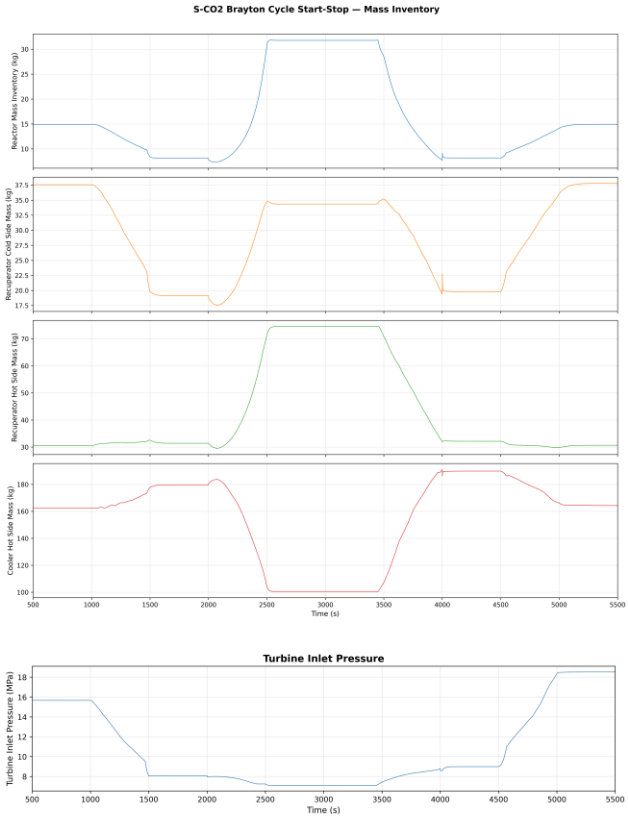


Figure 4. Simulation calculation results.

Because of the decrease in system temperature, the density of the working fluid stored on the hot side of the recuperator and inside the reactor increased rapidly. Since the total mass in the closed system remained constant, the mass stored in the cooler decreased rapidly. As a result, the overall system pressure declined rapidly.

DOI
10.3384/ecp22321

5 Computational-Performance Analysis

5.1 Modelica Solution Procedure

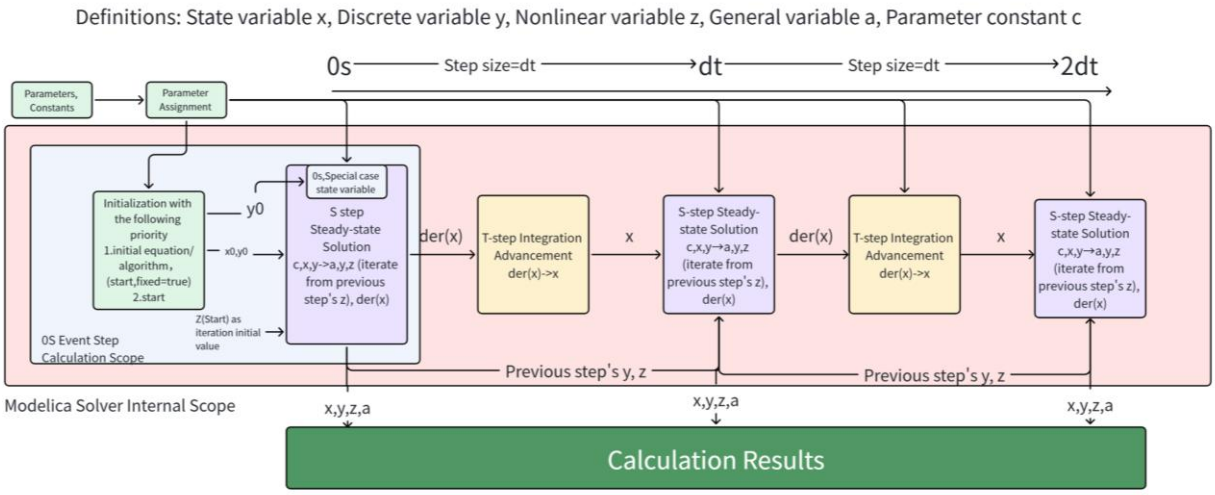


Figure 5. Schematic of the Modelica solution procedure.

As summarized in Figure **5**, The transient solution process in Modelica can be decomposed into two core stages: the steady-state solution step (Steady-state Step, abbreviated as the S-step) and the time-stepping advancement step (Time Step, abbreviated as the T-step). These two stages alternate cyclically and together complete the entire simulation process.

In the S-step, the state variables are treated as known quantities and are input into the steady-state equation system. The unknowns to be solved by the solver include general variables (both linear and nonlinear variables) as well as the derivatives of the state variables. Specifically, the core task of the S-step is to construct and solve one or more algebraic equation systems, which can be written as Equation 25:

$$F(x,u,p,t)=0 \tag{25}$$

where x denotes the state variables, u the input variables, p the parameters, and t time. In the current S-step, the state variables are treated as known quantities, whereas their derivatives der(x) are solved as part of the unknown set.

In the T-step, the solver advances the state variables to the next instant by numerical integration using the derivatives obtained in the S-step. Taking the fixed-step Euler method as an example, the state-variable update is given in Equation 26:

$$x_n=der(x)*dt+x_{n\text{-}1} \tag{26}$$

where x_n is the value of the state variable at the current time step, x_{n-1} is the value at the previous time step, and dt is the integration time step. All Modelica models involving integration are essentially systems of ordinary differential equations (ODEs), and different solution algorithms, such as the Euler method, the Runge-Kutta method, and DASSL, follow the same basic principle for numerical time advancement.

The overall solution process is characterized by alternating S-steps and T-steps: initialization is first performed to assign initial values to the state variables; the S-step is then executed to solve the steady-state equations; the T-step subsequently advances the state variables; and this sequence repeats until the simulation terminates. This solution architecture can be summarized by the following flow:

Initialization -> S-step (steady-state solution) -> T-step (integration advance) -> S-step -> T-step -> ... -> End of simulation.

When the derivatives of the state variables change rapidly within an integration step, a variable-step solver will roll back the T-step and reduce the integration step size to satisfy the required accuracy. The more intense and frequent the derivative variation, the more rollbacks occur, and the actual number of computational steps increases substantially.

Because of the basic characteristics of thermo-fluid models, the use of a staggered-grid formulation results in a relatively small nonlinear iteration burden, and the steady-state step is therefore computed relatively quickly. The main factors affecting the computational speed of such models are the stability of the state variables and the integration error.

5.2 Solver-Oriented Optimization Strategy

At present, four formulations are used to describe the momentum conservation equation in the discretized pipe model:

Fluid Equation 1 (Equation 27):

$$p_i\text{-}p_{i+1}\text{-}\Delta H\rho_i g\text{-}\Delta p_f=0 \tag{27}$$

Fluid Equation 2 (Equation 28):

$$der(G_i)=(p_i\text{-}p_{i+1}\text{-}\Delta H\rho_i g\text{-}\Delta p_f)*A/L \tag{28}$$

Fluid Equation 3 (Equation 29):

$$der(G_i)=(p_i-p_{i+1}-\Delta H\rho_i g-\Delta p_f)*A/L+f(G_i,G_{total}) \quad (29)$$

Fluid Equation 4 (Equation 30):

$$der(G_i)=(p_i-p_{i+1}-\Delta H\rho_i g-\Delta p_f+f(p_1,p_n))*A/L+f(G_i,G_{total}) \quad (30)$$

Fluid Equation 1 is the conventional momentum conservation equation that neglects flow inertia. Fluid Equation 2 accounts for the influence of flow inertia on momentum conservation. Fluid Equation 3 introduces the average pipe flow rate and reduces oscillations in the flow rate at individual pipe nodes through a first-order inertia term. Fluid Equation 4 further introduces inlet and outlet pressure feedback on the basis of Fluid Equation 3, thereby further reducing abrupt flow variations and oscillations.

5.3 Steady-State Computational Efficiency

Table 3. Analysis of computational efficiency under steady-state conditions.

Metric	Fluid Equation 1	Fluid Equation 2	Fluid Equation 3	Fluid Equation 4
Total Model CPU Time (s)	750.5	98.7	97.5	73.2
Solver Steps	791	526	494	458
Number of Dynamic Evaluations	2226	1142	1068	1000
Number of Jacobian Evaluations	642	97	92	72
Maximum Integration Order	2	1	1	1

Table 4. State variables affecting the global error and their occurrence frequencies.

Rank	Fluid Equation 1	Fluid Equation 2	Fluid Equation 3	Fluid Equation 4
1	cooler.pipe_heat.p[5] (312)	reactor.disPipe.m_flow[6] (193)	reactor.disPipe.m_flow[6] (135)	reactor.disPipe.m_flow[6] (133)
2	cooler.pipe_heat.p[4] (200)	reactor.disPipe.m_flow[5] (110)	cooler.pipe_heat.m_flow[6] (75)	cooler.pipe_heat.m_flow[6] (89)
3	cooler.pipe_heat.p[3] (83)	cooler.pipe_heat.m_flow[6] (46)	reactor.disPipe.m_flow[5] (70)	reactor.disPipe.m_flow[5] (47)

As summarized in Tables **3** and **4**, Fluid Equation 1 uses the conventional steady-state momentum conservation equation and neglects flow inertia. Its steady-state CPU time reaches 750.5 s, which is 10.3 times that of Equation 4. In addition, its number of Jacobian evaluations (642) and dynamic evaluations (2226) are both much higher than those of the other formulations, and it requires a maximum integration order of 2. Treating the momentum equation as a purely algebraic relationship means that even a slight change in the pressure difference between two adjacent control volumes can trigger a flow-rate change. Because the flow rate appears in the derivative terms governing pressure evolution, it in turn induces additional pressure variation, thereby increasing the global system error associated with the overall pressure field and slowing down the solution process.

After the momentum equation is converted to differential form (Fluid Equation 2), the CPU time drops sharply from 750.5 s to 98.7 s, corresponding to an efficiency improvement of about 7.6 times. The number of Jacobian evaluations decreases from 642 to 97, and the solver-step count is reduced from 791. After the fluid inertia term is introduced, additional flow-rate state variables are included in the system, the flow-rate variation becomes smoother, and the global error caused by the overall state variables is reduced. The state variables dominating the solution efficiency shift to the flow-rate variables, and the computational speed improves substantially.

For Fluid Equation 3, which adds a first-order inertia term based on the average pipe flow rate, the CPU time decreases to 97.5 s, close to that of Equation 2, but the number of Jacobian evaluations is further reduced to 92. This indicates that damping through the average flow rate effectively smooths local flow-rate oscillations.

Fluid Equation 4, which introduces inlet/outlet pressure feedback on the basis of Equation 3, shows the best overall performance. The CPU time is reduced to 73.2 s, while the solver-step count (458) and the number of dynamic evaluations (1000) are both the lowest among the four schemes. Endpoint pressure feedback significantly reduces time-step rollback and abrupt changes in the state variables, resulting in an overall efficiency improvement of about 25.8% relative to Equation 2.

5.4 Transient Computational Efficiency

Table 5. Analysis of transient computational efficiency.

metric	*Fluid Equation 1*	*Fluid Equation 2*	*Fluid Equation 3*	*Fluid Equation 4*
Total Model CPU Time (s)	Unable to Complete the Calculation	3044.2	3004.9	2859
Solver Steps	Unable to Complete the Calculation	7878	7535	3967

DOI 10.3384/ecp22321

metric	*Fluid Equation 1*	*Fluid Equation 2*	*Fluid Equation 3*	*Fluid Equation 4*
Number of Dynamic Evaluations	Unable to Complete the Calculation	17961	17510	9322
Number of Jacobian Evaluations	Unable to Complete the Calculation	4962	4853	2343
Maximum Integration Order	Unable to Complete the Calculation	3	3	3

Table 6. State variables affecting the global error and their occurrence frequencies.

Rank	*Fluid Equation 2*	*Fluid Equation 3*	*Fluid Equation 4*
1	reheater.pipe_cold.m_flow[1] (1868)	reheater.pipe_heat.m_flow[4] (1450)	reheater.pipe_cold.m_flow[1] (706)
2	cooler.pipe_heat.m_flow[2] (1304)	reheater.pipe_cold.m_flow[1] (1251)	reheater.pipe_heat.m_flow[4] (459)
3	reheater.pipe_heat.m_flow[4] (1259)	cooler.pipe_heat.m_flow[1] (640)	cooler.pipe_heat.m_flow[2] (288)

As summarized in Tables **5** and **6**, Fluid Equation 1 is unable to complete the transient simulation. In the investigated model and operating cases, the conventional algebraic formulation that neglects flow inertia was associated with numerical instability during startup, shutdown, and load following. These results should not be generalized to all sCO2 systems.

Consistent with the steady-state efficiency analysis, once the momentum equation is rewritten in differential form (Fluid Equation 2), the system gains flow-rate state variables, which smooth the flow-rate evolution and avoid the vicious cycle of 'small pressure disturbance -> severe flow-rate response -> amplification of global error' associated with the purely algebraic form. Equation 2 enables the transient simulation to be completed, but at a high computational cost (CPU time of 3044.2 s, 7878 solver steps, and 4962 Jacobian evaluations), indicating that the basic inertia term alone is still insufficient to suppress transient oscillations in the sCO2 system.

After adding a first-order inertia term to suppress oscillations (Fluid Equation 3), the CPU time decreases by only 1.3%, the solver-step count decreases slightly by 4.4%, and the number of Jacobian evaluations remains nearly unchanged. Thus, the improvement in transient performance produced by the additional first-order inertia term alone is limited.

Fluid Equation 4, which augments Fluid Equation 3 with inlet/outlet pressure feedback, exhibits the best overall performance under both steady-state and transient conditions. Compared with Fluid Equation 2, the solver-step count is reduced by 49.6%, the number of Jacobian evaluations by 52.8%, the number of dynamic evaluations by 48.1%, and the CPU time by 6.1%.

6 Conclusions

This study developed a one-dimensional dynamic Modelica model of an sCO2 Brayton power generation system. This study systematically compares four momentum-equation formulations within the same sCO2 Brayton-cycle system under identical numerical settings and quantitatively examines how state selection, flow inertia, and pressure feedback affect solver steps, Jacobian evaluations, and error-dominant variables. The main conclusions are as follows:

(1) The developed Modelica model was used to simulate the investigated sCO2 Brayton-cycle transients. It represented both reactor power reduction from 100% FP to shutdown and startup from zero power to stable operation at 100% FP, producing the reported responses of system flow rate, temperature, pressure, and shaft rotational speed. The same system model provided the common basis for comparing the four momentum-equation formulations under identical numerical settings.

(2) In the investigated model and operating cases, formulations containing a mass-flow derivative term completed the transient simulation, whereas the conventional steady-state momentum formulation did not converge under the same numerical settings. With the conventional steady-state momentum conservation equation, the steady-state computation time was 750.5 s and the transient simulation did not converge. After the momentum equation was rewritten in differential form with a mass-flow derivative term, the steady-state computation time decreased to 98.7 s and the transient simulation was completed successfully. Within this comparison, inclusion of the mass-flow derivative term was associated with successful transient convergence under the selected numerical settings.

(3) For the investigated model and operating cases, adding first-order inertia and endpoint pressure feedback to the mass-flow-derivative formulation was associated with lower reported computation time, solver workload, and occurrence frequencies of error-dominant flow-rate variables. On the basis of flow inertia, the additional inclusion of a first-order inertia term for the average pipe flow rate together with inlet/outlet pressure feedback (Fluid Equation 4) reduced the steady-state computation time from 98.7 s to 73.2 s, decreased the transient solver-step count from 7878 to 3967 (a reduction of 49.6%), and lowered the number of Jacobian evaluations from 4962 to 2343 (a reduction of 52.8%). At the same time, compared with Fluid Equation 2, the occurrence frequencies of the reported error-dominant flow-rate variables in the

reheater and cooler pipes were reduced by approximately 62% to 77%.

References

[1] Ferrero S, Batet L, Linares J I, Arenas E M, Cantizano A, Savoldi L(2021). A Modelica dynamic model of a supercritical CO2 energy conversion system for EU-DEMO[J]. Fusion Engineering and Design, Vol.173: 112826. DOI: 10.1016/j.fusengdes.

[2] Guo J, Jin Y, Zhao F, Fang H, et al(2023). Modeling and program development of supercritical CO2 sub-channel based on Modelica language[J]. Applied Thermal Engineering, Vol.227: 120462. DOI: 10.1016/j. applthermaleng.

[3] Pegallapati A S, Banoth P, Maddali R(2020). Dynamic model of supercritical CO2 based natural circulation loops with fixed charge[J]. Applied Thermal Engineering, Vol.169:114906.DOI:10.1016/j.applthermaleng.

[4] Qin T, Sun Y, Wei W, Lin J, Yan X(2025). Dynamic modeling and comparative analysis of heat exchangers in supercritical CO2 power cycles[J]. Case Studies in Thermal Engineering,Vol.71:106202.DOI:10.1016/j.csite.

[5] Zhang L, Huang X, Zhang K, Zhang J, Chen H(2026). Accident analysis of lead-bismuth cooled fast reactor and SCO2 Brayton cycle system based on Modelica language[J].Energy,Vol.349:140590.DOI: 10.1016/j.energy.

[6] Casella F, Guironnet A(2021). ScalableTestGrids - An Open-Source and Flexible Benchmark Suite to Assess Modelica Tool Performance on Large-Scale Power System Test Cases[C]. Proceedings of the 14th Modelica Conference. DOI: 10.3384/ecp21181351.

[7] Agosta G, Casella F, Cattaneo D, Cherubin S, Leva A, Scuttari M, Terraneo F(2023.). MARCO: An Experimental High-Performance Compiler for Large-Scale Modelica Models[C]. Proceedings of the 15th International Modelica Conference DOI: 10.3384/ecp20413.

[8] Flesch R, Kuhlmann A, Brunnemann J, Jörg E(2023). Understanding and Improving Model Performance at Small Mass Flow Rates in Fluid System Models[C]. Proceedings of the 15th International Modelica Conference. DOI: 10.3384/ecp204189.

[9] Bortoff S A, Laughman C R, Deshpande V, Qiao H(2025). Fluid Property Functions in Polar and Parabolic Coordinates[C]. Proceedings of the American Modelica Conference 2024. DOI: 10.3384/ecp20721.

[10] Tinnerholm J, Pop A, Heuermann A, Sjölund M(2021). OpenModelica.jl: A modular and extensible Modelica compiler framework in Julia targeting ModelingToolkit.jl[C]. Proceedings of the 14th Modelica Conference. DOI: 10.3384/ecp21181109.

Robust solution of inverse static problems in Modelica: the case for the initialSimplified() operator

Francesco Casella[1] Philip Hannebohm[2] Matteo Luigi De Pascali[1]

[1]Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milan, Italy,
{francesco.casella,matteoluigi.depascali}@polimi.it
[2]Institute for Data Science Solutions, Bielefeld University of Applied Sciences and Arts, Germany,
philip.hannebohm@hsbi.de

Abstract

Solving inverse static initialization problems, in which the outputs of a process are prescribed, is often more numerically challenging than solving direct initialization problem, where inputs are fixed instead. This paper proposes the introduction of a new `initialSimplified()` operator in Modelica, which allows to solve inverse problems with the same reliability as direct problems, using an intuitive, straightforward Modelica formulation. The proposed solution can be trivially implemented in Modelica tools and will be 100% portable among them, once the new operator is standardized. The usefulness of the proposed approach is demonstrated on a conceptual thermal power generation plant model.

Keywords: Modelica language, initialization, homotopy, convergence problems

1 Introduction

The Modelica language and tools are widely used for the dynamic modelling of industrial processes and energy conversion systems, which often need to be initialized in steady-state. They are also increasingly considered as a convenient framework to formulate and solve static models of such processes, see, e.g., (Luiten et al. 2026), where they can be very convenient because most of the model equations can be used for process design, where certain process parameters are unknown and need to be determined, and then for off-design static analysis of the process behaviour, where the parameters are fixed and some different boundary conditions can be provided.

One major issue in both cases is that large, strongly non-linear systems of algebraic equations stemming from those problems need to be solved with iterative methods, which may fail if the supplied initial guess is not close enough to the solution. Without proper ways to address this issue systematically, the potential for practical application of Modelica in this context is severely hindered.

Homotopy-based methods have been proposed and successfully used to mitigate the problem (Sielemann, Casella, Otter, et al. 2011; Casella et al. 2011; De Pascali and Casella 2026). The basic idea is to initially replace some expression $f(x)$ with a simplified counterpart $g(x) = 0$, so that the corresponding simplified problem can be solved in a more robust and reliable way than the original problem, then gradually morph the simplified expression into the actual one by means of the homotopy transformation

$$f_h(x) = \lambda f(x) + (1-\lambda)g(x) \tag{1}$$

where λ goes from 0 to 1. As discussed in (De Pascali and Casella 2026), the solution of the problem at $\lambda = 0$ can be made more robust by means of two strategies:

- replacing strongly non-linear expressions $f(x)$ with mildly non-linear or linear simplified versions;
- replacing time-varying terms in $f(x)$ with constant, known terms, e.g. some nominal or design value.

The first strategy aims at widening the set of initial guesses from which iterative solvers can successfully converge to the solution, possibly allowing to use default values that do not require specific tuning by the end-user, or even removing the need for initial guess values entirely, in case one obtains a linear system at $\lambda = 0$. The second strategy, besides also allowing to make the problem less non-linear, also allows to split the original, fully coupled problem into a sequence of smaller problems that can be solved sequentially. As clearly demonstrated in (De Pascali and Casella 2026), both strategies can lead to demonstrably huge benefits in terms of convergence robustness.

Note that, in order to fully benefit from the robustness improvements brought by these strategies, it is essential to perform ad-hoc structural analysis and symbolic simplification of the initialization problem at $\lambda = 0$, generating specific code to solve the simplified problem. This fact has been widely recognized for a long time and popular Modelica tools such as Dymola or OpenModelica implement it.

Of course, one must also choose the simplified expression $g(x)$ wisely, so that the solution of the problem as λ goes from 0 to 1 is devoid of singularities and as smooth as possible; however, this aspect goes beyond the scope of this paper.

In fact, there is one interesting use case, which is discussed in (De Pascali and Casella 2026), for which the

Modelica language currently does not provide proper support. Process models are characterized by input and output variables: inputs can be set by means of actuators, and outputs can be measured, and possibly controlled to specific target values by means of control systems. When solving steady-state initialization problems for dynamic models, or off-design problems for static process models, it is very often the case that the *forward* initialization problem, where the inputs are fixed, leads to a simplified model at $\lambda = 0$ which is much easier to solve than the *backward* problem, where one fixes the outputs of the system, in order to compute the corresponding inputs as a result. As a consequence, the solution of *backward* problems tends to be less numerically robust and more prone to convergence failure at $\lambda = 0$ than the solution of *forward* problems.

For example, if the input variables of a thermo-fluid process are certain fluid flow rates, fixing them makes the problem much less non-linear, because many quantities such as heat transfer coefficients or friction losses can be computed explicitly from those flow rates; furthermore, originally non-linear equations such as energy balances, which contain the product of flow rate times specific enthalpy, become linear once the flow rate is known. If some output variables like outlet temperature or turbine output power are fixed instead, the flow rates will be unknown, so the non-linear system to be solved will be larger and more non-linear.

An effective strategy to avoid the reduction of convergence robustness when solving *backward* problems is to solve the initialization in three steps:

1. Solve the on-design *forward* problem at $\lambda = 0$, by fixing the design values of the inputs.

2. Solve the on-design *backward* problem at $\lambda = 0$,[1] by fixing the design values of the outputs, using the previously found solution as an initial guess. Since the two solutions are identical, or very close to each other, no convergence problems arise.

3. Use the homotopy transformation to gradually move the solution from the on-design output values to the desired off-design values.

By following this strategy, the successful convergence of the more robust *forward on-design* initialization problems is enough to ensure the convergence of *backward on-design* initialization problems, thus eliminating the previously mentioned issue.

Unfortunately, it is currently not possible to program such a solution strategy directly in Modelica; one needs to use some scripting environment and Modelica tool API calls to first perform step 1, save the simulation results, and then reload them as an initial guess for step 2. This approach is not very convenient and, most importantly, not portable between different Modelica tools, since scripting languages and APIs are not standardized.

The goal of this paper is thus to motivate the introduction of a new language feature in Modelica, namely the operator `initialSimplified()`, which makes it possible to perform the three above-mentioned steps directly in any Modelica tool, with no need for scripting.

The paper is structured as follows: Section 2 introduces the new operator, its semantics, and how it can be implemented in Modelica tools; Section 3 describes a test case that validates the usefulness of the proposed approach, while Section 4 presents experimental results obtained with that test case. Section 5 concludes the paper with final considerations and recommendations for future work.

2 Method

2.1 Definition of the new operator

A new built-in Modelica operator with function syntax `initialSimplified()` is introduced.

This operator returns `true` when the simplified $\lambda = 0$ initialization problem is being solved, `false` otherwise. The intended usage of this operator is as a condition for if-equations and if-expressions, e.g.

```
if initialSimplified() then
  // Equations to be considered when
  // generating the code to solve the
  // simplified \lambda = 0 problem
else
  // Equations to be considered in all
  // other cases
end if;
```

The Modelica tool will consider the first set of equations to generate the code for the simplified $\lambda = 0$ initialization problem, while it will consider the second set of equations to generate the code that performs the homotopy transformation during initialization, as well as to generate the code for simulation.

The behaviour of this operator is inspired from and similar to the existing `initial()` operator. If one writes

```
if initial() then
  // Equations to be considered during
  // initialization
else
  // Equations to be considered in all
  // other cases
end if;
```

a Modelica tool will generate the code to solve the initialization problem with the first set of equations, computing the values of parameters and the initial values of states. Then, when the simulation starts, it will solve the simulation problem using the other set of equations at `time = StartTime`, fixing the state and parameter values and using the previously computed values of unknown algebraic variables as initial guesses for iterative solvers. The use case is also similar: the first set can be formulated to

[1] At least conceptually, in practice one may skip that solve and directly continue with the first $\lambda > 0$.

DOI
10.3384/ecp22331

make the initialization problem easier to solve when coupled to the initial equations, but still with a formulation that is equivalent (or nearly equivalent) to the second set, so that the first simulation step will compute the same (or very close) values starting from the previous solution as initial guess, thus with no convergence issues in case iterative solvers are involved.

2.2 Mathematical formulation

The first step of the simulation process for any Modelica model is the solution of the initialization problem. Considering for simplicity a purely continuous-time model with no discrete and clocked variables ad no algorithm sections, the unknowns of the initialization problem are the initial values of the state variables x, of their derivatives $\dot{x}$, of the algebraic variables v, and of the unknown parameters p with `fixed = false` attribute. The corresponding set of equations is obtained by assembling the **`equation`** and `initial` **`equation`** sections of the flattened model:

$$F(x,\dot{x},v,p,t,\lambda) = 0, \tag{2}$$

which is then solved at time $t =$ startTime. If the `homotopy()` operator is used in the model formulation, the equations in the initialization problem (2) depend on the homotopy parameter λ as recalled in Eq. (1).

As extensively discussed in (Sielemann, Casella, Otter, et al. 2011; Casella et al. 2011; Sielemann, Casella, and Otter 2013), the problem (2) is first solved for $\lambda = 0$, exploiting all structural simplifications that take place for that value of λ, using the `start` attributes of the unknowns as initial guesses; then, λ is continuously transformed from 0 to 1 (possibly in a non-monotonous way) to eventually achieve the solution of the actual problem at $\lambda = 1$. The idea is to formulate the problem at $\lambda = 0$ so that its convergence is less sensitive (or not sensitive at all, ideally) to a proper selection of initial guesses than the solution of the actual initialization problem. However, this may not always be possible for structural reasons, e.g. when solving backward initialization problems.

With the introduction of the `initialSimplified()` operator, the equations of the initialization problem can be split into three subsets:

$$F_r(x,\dot{x},v,p,t,\lambda) = 0 \tag{3}$$
$$F_s(x,\dot{x},v,p,t) = 0 \tag{4}$$
$$F_n(x,\dot{x},v,p,t,\lambda) = 0, \tag{5}$$

where (3) includes all equations that do not contain `initialSimplified()`, (4) includes the equations containing `initialSimplified()`, assuming it returns `true` and (5) includes the same equations, assuming that `initialSimplified()` now returns `false`.

The solution of the initialization problem is now split into the sequential solution of three problems:

1. Solve Eq's (3), (4) for $\lambda = 0$, $t =$ startTime, using the `start` attributes of the unknowns x, v, and p as initial guesses for the iterative solvers.
2. Solve Eq's (3), (5) for $\lambda = 0$, $t =$ startTime, using the solution found at the previous step as an initial guess for the iterative solvers.
3. Perform the homotopy transformation of Eq's (3), (5) for λ going from 0 to 1 as usual.

For both the first and second problems, it is essential to exploit all the structural simplifications made possible by the specific problem formulation.

The main idea is to use the `initialSimplified()` operator to formulate the first problem in a way can be solved more robustly than the second (which is what you currently get with vanilla `homotopy`), while leading to the same solution, or to a very near solution, so that the convergence of the solver on the second step will be guaranteed.

In particular, this approach can be readily applied to the steady-state initialization of dynamic process models: the first problem is the forward on-design initialization, where the design values of the *inputs* are enforced, while the second problem is the backward on-design initialization, where the design values of the *outputs* are applied, ideally leading to the same solution, or to a very close one. The third step can then be used to gradually drive the solution of the backward initialization problem from on-design to off-design conditions.

2.3 Implementation in Modelica tools

The implementation of the `initialSimplified()` operator is trivial. As noted in the Introduction, Modelica tools already split the code generation for initialization between the $\lambda = 0$ and the $\lambda > 0$ cases, so they just need to replace the new operator by `true` when generating the code for $\lambda = 0$, by `false` otherwise. That's it.

Note that the two branches of if-equations or if-expressions depending on this operator can lead to systems that have a completely different structure, e.g. *forward* vs. *backward* initialization. This is indeed intended; in fact, it is the whole point of the proposal. On the other hand, for this approach to work, it is essential that the two systems have the same (or nearly the same) solution, to ensure that the solution found at the first step can be used as an effective initial guess to solve the second; obviously, this is entirely the responsibility of the modeller.

2.4 User-friendly application in Modelica models

The open-source BoundaryInitBlocks Modelica library (De Pascali and Casella 2024), presented in (De Pascali and Casella 2026), provides general-purpose blocks to manage steady-state initialization of process models, allowing the end user to easily set up forward and backward initialization problems, by setting suitable parameters in blocks connected to the process input and output connectors.

In that paper, the possibility of supporting robust backward initialization by first solving the forward

on-design problem is discussed; amending the library code to provide such a feature with the proposed `initialSimplified()` would be trivial.

3 Test Cases

The usefulness of the proposed `initialSimplified()` operator is demonstrated by means of the conceptual thermal power generation system model listed in Fig. 1.

The model represents a counter-current heat exchanger where a hot flue gas with temperature T_{fg} heats a gas working fluid with temperature T, which is then expanded in a turbine to produce a power output P. The heat exchanger is modelled by N finite volumes.

The model is fully static, so it is representative of both static design models and steady-state initialization of a corresponding dynamic model, where all the derivative terms are set to zero by the initial equations.

The two inputs of the process are the mass flow rate w of the working fluid, which is supplied by a compressor, and the mass flow rate w_{fg} of the flue gas, which is supplied by the exhaust of some combustion process.

The forward initialization problem can be formulated by fixing the two inputs to their design values:

```
model ForwardOnDesign
  Plant plant;
equation
  plant.w = 1;
  plant.wfg = 1;
end ForwardOnDesign;
```

Since w and w_{fg} are set to known values, one can explicitly compute vector G; then, the first three equations of the for-loop form a system of $3N$ equations that can be solved by a linear solver, so there is no need to set any guess value for the unknown vectors T, T_{fg}, Q. Once that is done, all the remaining equations can be solved explicitly.

Hence, despite the obvious non-linearity of the process model, as long as the inputs are fixed it is possible to solve it in a completely reliable way, without the need of worrying about initial guess values.

Consider now the following *backward* initialization problem, where we instead fix two outputs, the flue gas outlet temperature T_s and the turbine power P. Note that the second output is fixed to an off-design value which is 30% of the nominal design value; however, homotopy is used to get there gradually, starting in nominal conditions at $\lambda = 0$; this is done as suggested in (De Pascali and Casella 2026) because the solution of on-design problems is typically less numerically challenging than in strongly off-design conditions.

```
model BackwardOffDesign
  Plant plant;
  parameter Real load = 0.3;
equation
  plant.P = homotopy(
    actual = plant.Pnom*load,
    simplified = plant.Pnom);
  plant.Ts = plant.Tsnom;
end BackwardOffDesign;
```

In this case, both fluid mass flow rates w and w_{fg} are unknown, so it is not possible to compute the conductances G a-priori. Due to the structure of the problem, one now gets a slightly larger implicit system of $3N+3$ equations that also includes the two turbine equations and one scalar equation for G (the other $N-1$ can be removed by common sub-expression elimination). Contrary to the previous *forward* problem, this system is now strongly non-linear. Solving it without setting initial guesses for all these unknowns is very likely going to be problematic.

With the proposed new `initialSimplified()` operator, one can write the following model:

```
model BackwardOffDesignInitialSimplified
  Plant plant;
  parameter Real load = 0.3;
equation
  if initialSimplified() then
    plant.w = 1;
    plant.wfg = 1;
  else
    plant.P = homotopy(
      actual = plant.Pnom*load,
      simplified = plant.Pnom);
     plant.Ts = plant.Tsnom;
  end if;
end BackwardOffDesignInitialSimplified;
```

According to the proposed semantics, the code to solve the problem at $\lambda = 0$ is generated with fixed inputs, so it will have exactly the same structure as in `ForwardOnDesign`; hence, it will be solved reliable without the need of setting initial guess values. The initialization problem will then be solved with a homotopy transformation that involves the non-linear system of $3N+3$ equations; however, at $\lambda = 0$ that system still has a solution that is very close to the solution of the *forward* simplified initialization problem that has just been solved, so if that solution is used to set initial guess values, the iterative Newton solver will converge easily. Then, the solution will be gradually shifted to 30% partial load, but this will not be problematic as long as short enough λ-steps are taken.

This simple example clearly shows the advantages of the proposed approach:

- The Modelica code is very straightforward and intuitive to write. As noted in Section 2.3, this code could written once and for all in reusable utility blocks.
- The much harder non-linear *backward* initialization problem will be solved with the same robustness as the *forward* problem, without the need of setting any additional start attributes.
- Once the new operator is standardized, this model will be portable across Modelica tools, with the same guaranteed degree of robustness.

The authors tried hard to test this approach in full-scale, realistic test cases, using the fuel cell and innovative power generation plant models that are discussed in (De Pascali and Casella 2026). However, implementation issues

DOI
10.3384/ecp22331

with the new backend of the OpenModelica compiler at the time of this writing prevented the successful testing in those cases; this will be the subject of a subsequent publication.

4 Experimental results

Experiments were carried out with the test case illustrated in the previous Section, using Dymola 2026x and OpenModelica v1.27.0-dev-462, which implements the experimental `initialSimplified()` operator in the new backend.

As expected, due to its nice structural properties, the `ForwardOnDesign` model is solved successfully by both tools.

When tackling the `BackwardOffDesign` model, Dymola fails to solve it at $\lambda = 0$, because of repeated division-by-zero errors, due to the lack of reliable initial guess values for the solution of the implicit *backward* initialization problem. OpenModelica barely manages to solve it, after repeatedly reporting numerical issues at $\lambda = 0$ such as division by zero or raising a negative number to a Real power. As expected, the solution of the *backward* initialization problem at $\lambda = 0$, which involves a significantly non-linear system of equations, can be problematic if initial guess values are not carefully provided.

The `BackwardOffDesignInitialSimplified` model is instead solved successfully by the experimental implementation in OpenModelica without any warnings indicating potential numerical issues, as expected from the analysis carried out in the previous Section.

This confirms the main argument of this paper, namely that the availability of the `initialSimplified()` operator allows to solve *backward* initialization problems with the same reliability of *forward* problems, in a way that would be 100% portable across tools, once the operator is standardized.

Regarding tests on full-scale power generation models, the currently available experimental evidence is that OpenModelica solves the forward initialization problems for those plant models easily, while it struggles with the backward ones; hence, it is expected that the application of the approach proposed in this paper to manage the initialization would be beneficial. However, it was not possible to directly test the proposed approach at the time of this writing due to (unrelated) issues with the new OpenModelica backend, where the experimental operator is currently implemented.

5 Conclusions

The solution of *backward* initialization problems in Modelica process models is often more numerically challenging than the solution of *forward* initialization problems. An effective strategy to avoid this problem is to use homotopy-based initialization, solving the *forward* problem in on-design conditions at $\lambda = 0$ first, then using the result as an initial guess for the homotopy transformation of the full *backward* initialization problem, which starts from the same on-design conditions (so convergence is guaranteed) and gradually shifts to off-design conditions.

This paper proposes the introduction of the operator `initialSimplified()`, which allows to formulate such a strategy directly in the Modelica model in a very intuitive way, without the need of performing complex operations with scripting languages and tool APIs, that are not portable across Modelica tools. Such a strategy can be very effective at improving the robustness of convergence of *backward* problems, avoiding or drastically reducing the need of setting start attributes to provide good initial guesses to the iteration variables of the solver, which is a tedious and fragile process, also due to the fact that different Modelica tools (or even different versions of the same tool) may choose different iteration variables for the same problem.

The implementation of the `initialSimplified()` operator is trivial for tools that already generate separate code for the initialization problem at $\lambda = 0$, in order to exploit all the structural simplifications introduced by the homotopy simplifications.

The effectiveness of this approach was demonstrated successfully on a conceptual model of a thermal power generation system. At the time of this writing, it was not yet possible to test this approach directly on full-scale power generation system models because of (unrelated) implementation issues with the new OpenModelica backend, where the experimental operator is implemented. However, the experimental evidence with the old backend shows that solving forward problems is much less problematic than solving backward problems, thus suggesting that the proposed approach will be beneficial.

The authors thus recommend the inclusion of the new operator in the forthcoming release of the Modelica Language Specification.

Acknowledgements

The authors would like to express their sincere appreciation to the Open Source Modelica Consortium (OSMC) for support, collaboration, and shared commitment to advancing open-source modelling and simulation technologies.

References

Sielemann, Michael, Francesco Casella, and Martin Otter (2013). “Robustness of declarative modeling languages: Improvements via probability-one homotopy”. In: *Simulation Modelling Practice and Theory* 38, pp. 38–57. DOI: 10.1016/j.simpat.2013.07.001.

Sielemann, Michael, Francesco Casella, Martin Otter, Christoph Clauß, Jonas Eborn, Sven Erik Mattsson, and Hans Olsson (2011-03). “Robust Initialization of Differential-Algebraic Equations Using Homotopy”. In: *Proceedings 8th International Modelica Conference*. Ed. by C. Clauss. Modelica Association. Dresden, Germany, pp. 75–85. ISBN: 978-91-7393-

096-3. DOI: 10.3384/ecp1106375. URL: http://www.ep.liu.se/ecp/063/010/ecp11063010.pdf.

Casella, Francesco, Michael Sielemann, and Luca Savoldelli (2011-03). "Steady-state initialization of object-oriented thermo-fluid models by homotopy methods". In: *Proceedings 8th International Modelica Conference*. Ed. by C. Clauss. Modelica Association. Dresden, Germany, pp. 86–96. ISBN: 978-91-7393-096-3. DOI: 10.3384/ecp1106386. URL: http://www.ep.liu.se/ecp/063/011/ecp11063011.pdf.

De Pascali, Matteo Luigi and Francesco Casella (2024). *BoundaryInitBlocks library*. https://github.com/looms-polimi/BoundaryInitBlocks.

De Pascali, Matteo Luigi and Francesco Casella (2026). "Robust steady-state initialization of object-oriented advanced thermal power generation system models with application to the case of the SOS-CO_2 cycle". In: *Mathematical and Computer Modelling of Dynamical Systems* 32.1, p. 2616725. DOI: 10.1080/13873954.2026.2616725.

Luiten, Casper, Carlo De Servi, and Francesco Casella (2026). *Combined On-Design and Off-Design Modeling of Aerospace- grade Thermal Systems in Modelica*. https://openmodelica.org/images/M_images/OpenModelicaWorkshop_2026/OpenModelica2026_Casper_Luiten.pdf. Presentation at the 2026 OpenModelica Workshop.

Figure 1. Conceptual thermal power generation model

```
model Plant
  connector RealInput = input Real;
  connector RealOutput = output Real;

  parameter Integer N = 5 "Number of finite volumes";
  parameter Real NTU = 4 "Nominal NTU";
  final parameter Real Gtot = NTU "Nominal total heat conductance";
  parameter Real cp = 1 "Specific heat capacity";
  parameter Real kf = 0.1 "Fluid friction coefficient";
  parameter Real Kt = 1 "Turbine flow coefficient";
  parameter Real Tnom = 700 "Nominal TIT";
  parameter Real Tsnom = 400 "Nominal flue gas outlet temperature";
  parameter Real Pnom = 223 "Nominal turbine power";
  parameter Real pnom = 1.1 "Nominal fluid inlet pressure";
  parameter Real TOP = 0.1 "Nominal turbine outlet pressure";
  parameter Real gamma = 1.2 "cp/cv";

  RealInput w "Fluid flow rate";
  RealInput wfg "Flue gas flow rate";

  Real T[N + 1] "Fluid temperatures";
  Real p[N + 1] "Fluid pressures";
  Real Tfg[N + 1] "Flue gas temperatures";
  Real G[N] "Volume conductivity";
  Real Q[N] "Thermal powers";
  Real TIT = T[N + 1] "Turbine inlet temperature";

  RealOutput P "Turbine power";
  RealOutput Ts = Tfg[N+1] "Outlet flue gas temperature";
  RealOutput TIP = p[N + 1] "Turbine inlet pressure";
  RealOutput pin = p[1] "Fluid inlet pressure";
equation
  T[1] = 300;
  Tfg[1] = 800;
  for i in 1:N loop
    0 = w*cp*(T[i] - T[i + 1]) + Q[i];
    0 = wfg*cp*(Tfg[N - i + 1] - Tfg[N - i + 2]) - Q[i];
    Q[i] = ((Tfg[N - i + 2] + Tfg[N - i + 1])/2 - (T[i + 1] + T[i])/2)*G[i];
    G[i] = NTU/N*w^0.8;
    (p[i] - p[i + 1]) = kf/N*w^2;
  end for;
  w = Kt*p[N + 1]/sqrt(TIT/Tnom);
  P = w*cp*TIT*(1 - (TOP/TIP)^((gamma - 1)/gamma));
end Plant;
```

DOI
10.3384/ecp22331

Large Network Examples for the Generation of Modular Simulation Code from Modelica

Dirk Zimmer

Institute of Robotics and Mechatronics,
German Aerospace Center (DLR), dirk.zimmer@dlr.de

Abstract

This paper presents progress toward the simulation of large-scale networks in a strict subset of Modelica using Linear Implicit Equilibrium Dynamics (LIED). A specialized compiler generates modular simulation code that scales to millions of equations — beyond conventional monolithic approaches. Demonstrated with a ThermoLiquidStream library for district heating, fractal networks exceeding 10 million equations are successfully simulated on standard hardware. The results confirm that LIED-based modular compilation offers a practical, industrially relevant path for large network modeling while remaining fully compatible with existing Modelica tools.

Keywords: *Compiler Design, Modelling Methodology, Large Scale Systems*

1 Introduction

1.1 Previous Work

This work presents the current state and progress for the simulation of large-scale networks formulated in a strict subset of Modelica. It builds heavily on 3 previously presented works by the author:

1. The Linear Implicit Modeling Methodology presented 2023 in Aachen, Germany and published more extensively as preprint (Zimmer2024a)
2. The scaling capabilities of (manually) generated modular code presented 2024 in Jeju, Kora (Zimmer2024b)
3. The formulation of a Modelica sub-set and the development of a separate compile path presented 2025 in Lucerne, Switzerland. (Zimmer2025)

To make the paper easier to understand the main conclusion of these 3 works are quickly reintroduced in this section.

1.2 Linear Implicit Equilibrium Dynamics

Linear Implicit Equilibrium Dynamics (LIED) is technically defined as a special class of Differential Algebraic Equation (DAE) Systems. A DAE system with potential state derivatives $\dot{\boldsymbol{x}}$, time t and algebraic variables $\mathbf{w}$

$$\mathbf{0} = \boldsymbol{F}(\dot{\boldsymbol{x}}, \boldsymbol{x}, \mathbf{w}, t)$$

is defined as LIED system when it can be transformed into the following form:

$$\begin{bmatrix} \mathbf{w}_E \\ \dot{\boldsymbol{x}}_E \end{bmatrix} = \mathrm{g}(\boldsymbol{x}_I, \boldsymbol{x}_E, t)$$

$$\mathbf{A}(\boldsymbol{x}_I, \boldsymbol{x}_E, \mathbf{w}_E) \begin{bmatrix} \mathbf{w}_I \\ \dot{\boldsymbol{x}}_I \end{bmatrix} = \mathrm{f}(\boldsymbol{x}_I, \boldsymbol{x}_E, \mathbf{w}_E, t)$$

We see that both the algebraic variables as well as the state derivatives can be split into a fully explicit part ($\dot{\boldsymbol{x}}_E$; $\mathbf{w}_E$) and a part ($\dot{\boldsymbol{x}}_I$; $\mathbf{w}_I$) with a linear system in implicit form expressed by the regular matrix $\mathbf{A}$.

This seemingly very restrictive class of DAEs is surprisingly expressive and is also very compatible with an object-oriented modeling style. This is achieved by using triplets instead of the classic pairs as presented in Table 1.

Table 1 Connection triplets for the object-oriented modelling of LIED Systems.

Domain	Signal	Potential	Flow
trans. mechanics	position: r [m]	velocity: v [m/s]	force: f [N]
rotational mechanics	angle: φ [rad]	angular velocity: ω [rad/s]	torque: τ [Nm]
Thermo-fluid streams	Thermo-dynamic state: $\widehat{\Theta}$	inertial pressure r [Pa]	mass-flow rate: $\dot{m}$ [kg/s]

Figure 1 depicts a modeling diagram a planar mechanical system of a passive dynamic walker with knees (McGeer1990) using a LIED based mechanical library. This shows that also complex kinematics and contact problems can be handled with this modeling approach. Later in the paper further examples for liquid fluid streams will be presented.

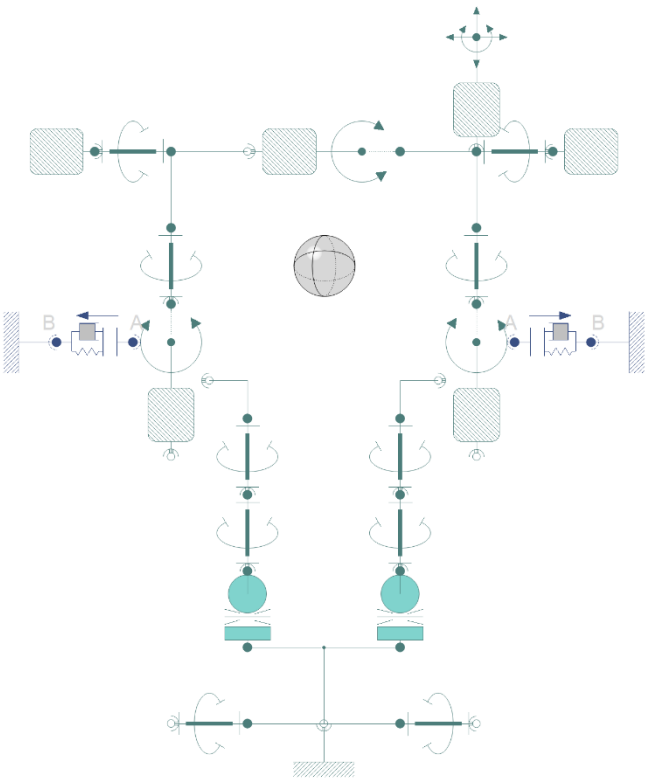


Figure 1: Model Diagram of a passive dynamic bipedal walker with knees.

1.3 Modular Simulation Code and Scaling Properties

The typical simulation code that is generated out of a Modelica model has a monolithic character. This means that the amount of simulation code typically scales with the model size. This is often problematic for scaling to very large systems and also yields code that is difficult to transfer to modern compute architectures such as GPUs.

The difficulty of scaling classic Modelica code is well represented under the many different approaches to cope with this problem. Tensor or vector-based approaches are provided by (Neumayr2023) and (Abdelhak2023). Special compilers are developed by (Agosta2023) and tested on (Casella2015). Pre-compilation of components for the generalized case is performed most notably by (Benveniste2019), (Benveniste2023), .

For LIED Systems, achieving scaling is much simpler than for general Modelica models. Sub-models that are formulated in Modelica can be translated to modular code fragments. This is possible because the models can be translated with an assumed causality. To understand this we can review Table 1.

The signal of the interface triplet is also denoted as carrier signal and has a known causality: it is either input or output. We shall now prescribe that the potential variable (and its occurring derivatives) will share this causality with the carrier signal whereas the corresponding flow variable (and its occurring derivatives) shall be prescribed to have the inverse causality.

This prescribed causality is not a true causality. It is simply used in order to impose an ordered description of the model. As shown in (Zimmer2024a), both state-variables as well as suitable tearing variables for the linear equation system **A** can be determined on the model level. Hence model code is already available in semi-causalized form.

A simulator that works on the resulting modular code has now simply to perform Tarjan's algorithm for strong component analysis (Tarjan1972) and can then solve the whole system. This has been documented in (Zimmer2024b). Figure 2 displays the scaling properties of a system of many parallel crane-craps (a 2D system with 2 degrees of freedom) and how this can be simulated on a standard CPU for up to 800 millions of equations.

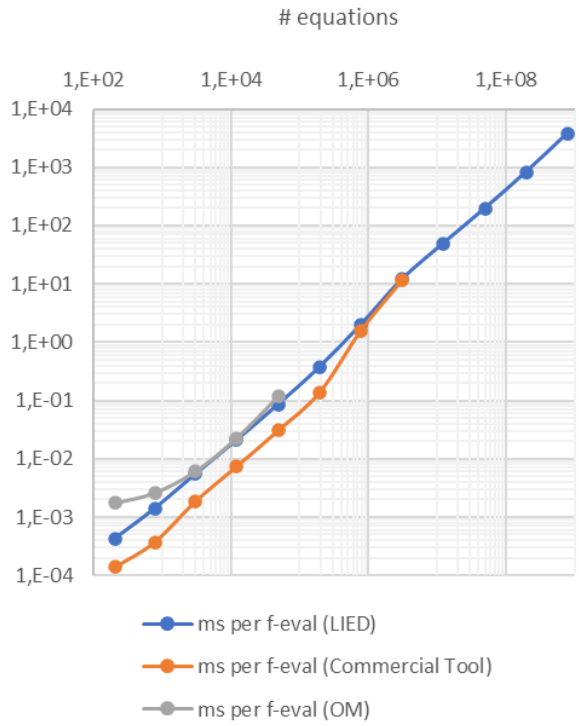


Figure 2: Scaling of a model of parallel crane-crabs up to 800 million equations or until the tools feasible limit. The plot displays the average time per model evaluation over the model size in equation. Two things a are demonstrated: modular code enables to gain several orders in magnitude in model size. Modular code remains computationally competitive with respect to monolithic simulation code.

1.4 Modelica Subset and Special Compiler

The code of the previous scaling example has been manually generated. To enable the creation of such code directly from a Modelica model, a special Compiler is currently in development that opens up a separate compile-path for LIED systems as shown in Figure 3.

DOI
10.3384/ecp22339

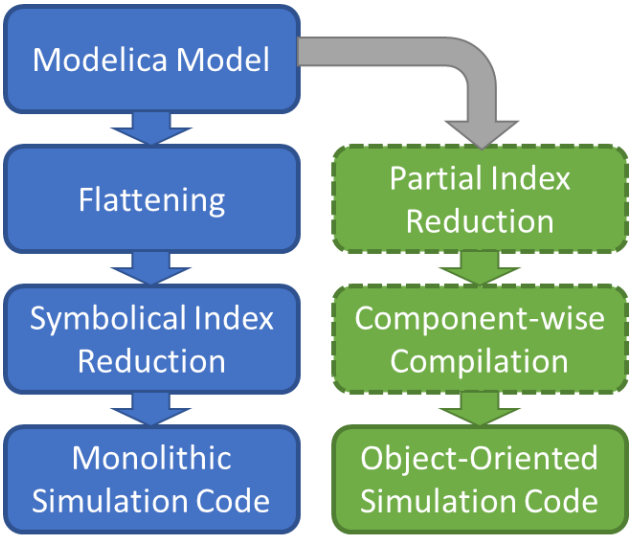


Figure 3: Illustration of different pathways for the compilation of simulation code. The blue boxes on the left illustrate the classic compilation of Modelica where all equations are collected in a single set. The green boxes on the right illustrate a potential pathway for LIED systems.

Supporting the full Modelica Language is neither feasible nor desirable in this case. Since this represents a special solution for LIED systems, the following restrictions are enforced.

- The connectors of a model must be either signals or pairs of potential of flows declared after a signal (which then acts as carrier signal). This restriction enables the compiler (and the modeler) to assume a causality for all connector variables.
- The model equations are then written in an order that coincides with assumed causal flow with the unknown occurring on the left-hand side of the equation.
- Parameters can be evaluated at compile time (if necessary) and must be written in correct order in case they depend on each other.
- If statements are evaluated at compile time.
- (longform) Definition of classes is only allowed in packages.
- Models only contain equations. All algorithms must be wrapped in a function.
- Non-interface variables and sub-models must be protected.

The following features of Modelica are not supported and will not be supported for the foreseeable future:

- stream connectors
- homotopy
- connections.branch
- expandable connectors

The following features of Modelica are currently ignored and will be ignored for the foreseeable future

- all annotations
- state-select attributes

The current implementation embraces the following features:

- base types
- (partial) functions (only internal and built-in)
- records (but no operator records yet)
- inner-and outer models (for parameter sharing only)
- extensions
- conditional components and conditional equations
- arrays and vectors

The following features are currently not supported but shall be supported in the near term:

- nested modifiers
- replaceable, redeclare and redefine including their application on packages
- arrays on components.
- external functions

The remaining gap of features is still under exploration.

To support the compilation of LIED components, the following special types are introduced

```
type ContinuousState = Real;

model LinearResponse "tearing"
  input Real residual;
  output Real balance;
equation
  residual = 0;
  annotation(defaultConnectionStructurallyInconsistent=true);
end LinearResponse;

type RealWithDerivative = Real;
```

ContinuousState can be used to determine states in an orthogonal way to the state Select attribute.

The LinearResponse model is used to indicate a tearing for a linear equation system. RealWithDerivatives is used to indicate that the derivative of an interface variable is needed.

Please note that these are not extensions to the existing Modelica but fully contained within Modelica. A normal Modelica compiler can simply process these special types in the usual way whereas a Compiler for LIED systems can exploit their special meaning.

2 Modeling of LIED Components

The aim of this chapter is to demonstrate that despite the current limitations of the special compiler meaningful Modelica libraries can be developed that are principally suited for industrial applications.

Given the current capabilities of the special Modelica compiler, a district heating example was identified as suitable use case. The lack of arrays is not very relevant for this application class and because water is only

modeled in its liquid form an elaborate H2O media model is not needed. On the other hand, large district heating networks do exist and their modeling can be expected to significantly profit from a modular compilation approach. For these reasons, a small Modelica library for Thermo-Liquid Streams has been developed. Figure 4 presents its basic structure.

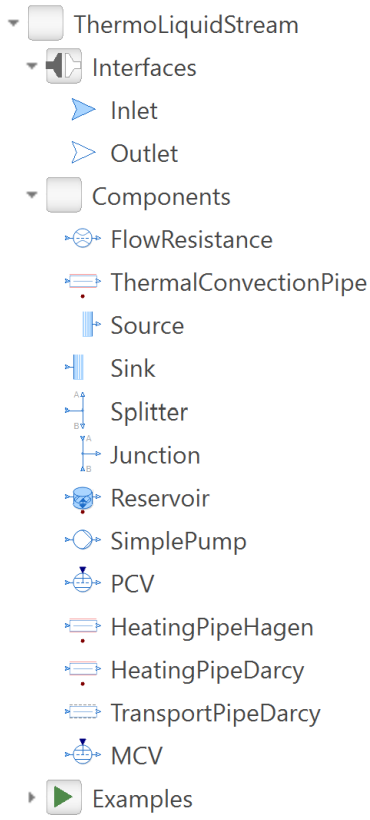


Figure 4: Components of the ThermoLiquidStream Library

As the name (and also the icon design) suggest, this library heavily borrows its equations and design principles from the well known DLR ThermoFluidStream Library (Zimmer2022). The main difference is that it is specialized for pure liquids and hence does not include a media model. For a better understanding let us look at the inlet connector (the outlet naturally forms the corresponding counterpart):

```
connector Inlet
  input SI.Pressure p;
  input SI.Temperature T;
  SI.Pressure r;
  flow B.RealWithDerivative m_flow(
   unit=”kg/s”);
end Inlet;
```

As in the DLR ThermoFluid Stream the connector contains the pair of inertial pressure r and the mass flow rate $\dot{m}$. The record for the thermodynamic state of a replaceable media model is now replaced by two signals for pressure and temperature. The idea behind this choice of quantities is that for liquids which are moved through vessels whose compression modulus is far lower than the compression modulus of the liquid itself, pressure and temperature can be treated as orthogonal quantities. For water in a district heating network this assumption holds true.

Also note that the mass flow rate is declared as `RealWithDerivative` since the derivative may be needed by many component using the interface.

The code below displays the model of a linear flow resistance. As you can see this is just a normal Modelica model:

```
model FlowResistance
  Interfaces.Inlet inlet;
  Interfaces.Outlet outlet;
  parameter Real L=0.01;
  parameter Real k(
   quantity="LinearFlowResistance",
   unit="Pa.s/kg") =100;

protected
  SI.Pressure dp;

equation
  inlet.m_flow = -outlet.m_flow;
  dp = outlet.m_flow*k;
  outlet.T = inlet.T;
  outlet.p = inlet.p + dp;
  outlet.r = inlet.r+der(outlet.m_flow)*L;
end FlowResistance
```

Whereas the restriction of the LIED modeling methodology do not show in the above example the become apparent in the modeling of a sink. This seemingly simple component, requires more attention in this case:

```
model Sink
  parameter SI.Pressure p0 = 1e5;
  parameter Real L = 0.01;
  Interfaces.Inlet inlet;
protected
  SI.Pressure r;
  B.ContinuousState m_flow;
  B.LinearResponse cut;
equation
   der(m_flow) = cut.balance;
   r = inlet.r - der(m_flow)*L;
   cut.residual = p0 - r - inlet.p;
   inlet.m_flow = m_flow;
end Sink
```

A sink always defines a mass-flow rate as a state. This is marked by using `ContinuousState` as a type. Then its derivative is determined by a linear equation system spanning several components so that the inertial pressure balances the remaining pressure gradient. This is expressed by using the model for `LinearResponse`.

DOI
10.3384/ecp22339

Using these two mechanism, the sink model can be formulated using the assumed causalities.

Please note that the model above is perfectly legal Modelica code and also correctly processed by standard Modelica compilers.

Using all these components (and a few more for control and heat transfer), a simple heating loop for a household can be modelled. This is depicted in Figure 5. In this example a low-gain PI controller adjust the mass-flow stemming from the heat reservoir to reach the target temperature. Flow reversal is made impossible by the design of the control valve. The transport pipe has 3 temperature sections that delay the transport of heat and also dissipates energy into the ground. The author has taken inspiration from the TransientLib (Andresen2015, Gillner2025) while creating these simple models.

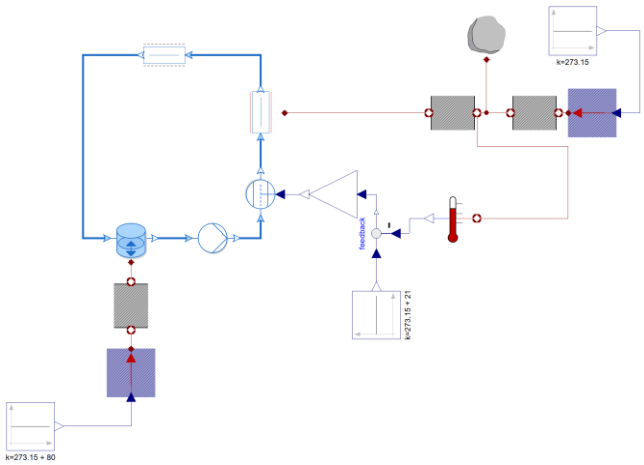


Figure 5: Simple heating loop for a household. The left side models the heating loop. The right side models the heat capacity of the household.

When translating this simple example a linear system with 5 equations occurs. It can be solved for instance by one tearing variable. The corresponding declaration of this tearing variable in this case is part of the tank model (that shares some of its equations for the inlet side with the previously detailed sink model)

Separating the model for the central heat reservoir and the household enables one to create larger network. Figure 6 shows a model with 4 households. When connecting this grid model with a central heat source, a linear equation system over 53 equations occurs that can be solved by 4 tearing variables. The corresponding declarations are part of the 3 junction models and the one tank model of the central heat source.

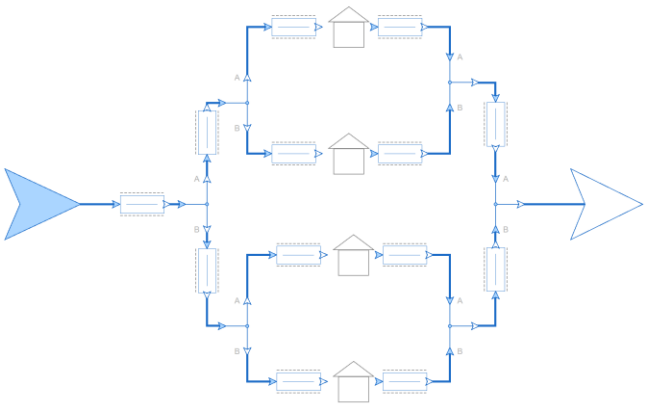


Figure 6: Simple heating network with 4 households

3 Scaling to a large network example

Real existing district heating networks can span over tens of thousands (sometimes even hundreds) of thousands end nodes. Being able to simulate really large models is thus desirable.

While it goes beyond the scope of this work to model such a large system realistically, we can at least create models of such size using generative means. The easiest way to achieve this is by creating a fractal grid structure. To this end, we can simply nest the grid of Figure 6 into itself as illustrated in Figure 7

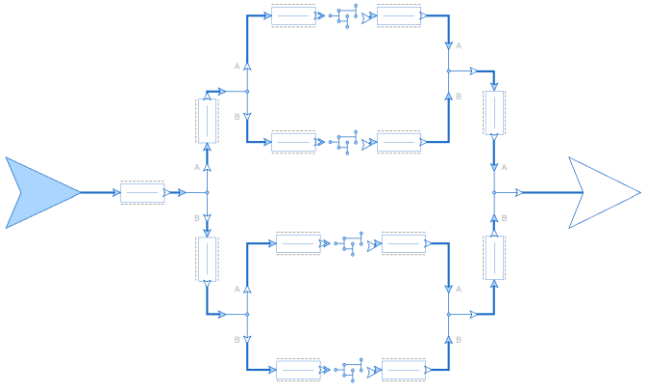


Figure 7: A fractal heating network**.**

Doing so several times over, we can generate models for 4^n households. Table 2 lists the number of total equations, the number of continuous states, the size of the linear equation system and the number of tearing variables needed which coincides with 4^n. This system spans the whole model and cannot be decomposed into several independent subsystems.

Up to $n = 7$, the models have been manually created in Modelica. To generate models for higher n, the modular simulation code that has been generated by special LIED compiler has been modified manually to allow a fractal generation of the model. This is highlighted in the code listing that is contained in the appendix.

Table 2: Scaling Properties of the fractal network model

n	# equ	#states	#linear	#tearing
0	129	9	5	1
1	599	57	53	4
2	2531	261	249	16
3	10259	1077	1033	64
4	41171	4341	4169	256
5	164819	17397	16713	1024
6	659411	69621	66889	4096
7	2637779	278517	267593	16384
8	10551251	1114101	1070409	65536
9	42205139	4456437	4281673	262144

4 Performance and Current Limitations

The simulator of the modular simulation code could perform a successful simulation up to n=8, a model of 10 million of equations containing a linear equation system over 1 million of equations that can be reduced by tearing to 65 thousand dimensions.

The instantiation of the model requires 2.7 GB RAM, and a full model evaluation on a single CPU takes then 350ms. However, the fast model evaluation is only achieved by exploiting two properties of the model:

- a sparse system solver is used for the linear equation system.
- the factorization of the last simulation step is being reused (or iteratively applied) which works perfect in this example because the matrix of the linear equation system is time-invariant.

However, whereas these two points work especially well in this model, they also represent good strategies for many large network models of various kind.

Whereas the factorization of the linear matrix is done using sparse methods, the composition of this matrix (by various partial model evaluations) is done still using dense methods. That means that the time needed to compose this matrix grows excessively. For $n = 8$, this was more than 1.5 hours, and for $n = 9$, it can be expected to last more than a day and was hence never conducted. Colored Jacobian methods (Assefaw2005) would be required to enable also a sparse composition of the linear matrix.

The model could however be instantiated also for $n = 9$. Instantiation failed for $n = 10$. The reason is a flaw in the implementation of Tarjan's strong component analysis leading to a stack overflow. Removing this flaw would enable even larger models. Model with $n > 10$ would then also require a machine with more than 48GB of RAM.

Figure 8 and 9 put the performance of the modular simulation code (ZimSim) in relation with classic Modelica compilers such as OpenModelica (in standard settings) and a commercial Modelica environment. We shall however refrain taking these data as direct benchmark since the compute hardware has been similar but not identical and full IDE environments are compared with a stand-alone simulator, also more favorable settings for the classic simulators could be found. However, three general conclusions can be drawn:

- With its large algebraic equation system, the model is challenging to scale. There are running examples of significantly larger models for all 3 simulators that have a more benevolent structure (such as the example from Figure 2)
- The performance advantage for ZimSim (Figure 9) at larger models solely stems from the fact of reusing the factorization. As evident at smaller system scale, the generated modular code is still sub-optimal. An inspection of the generated reveals several points for substantial improvement (like the execution of redundant code)
- The true scaling advantage of a modular approach is the memory consumption as clearly demonstrated by Figure 8.

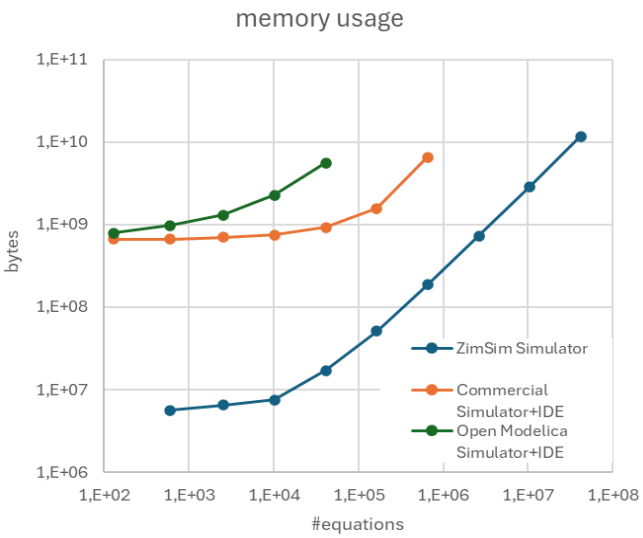


Figure 8: Memory Usage

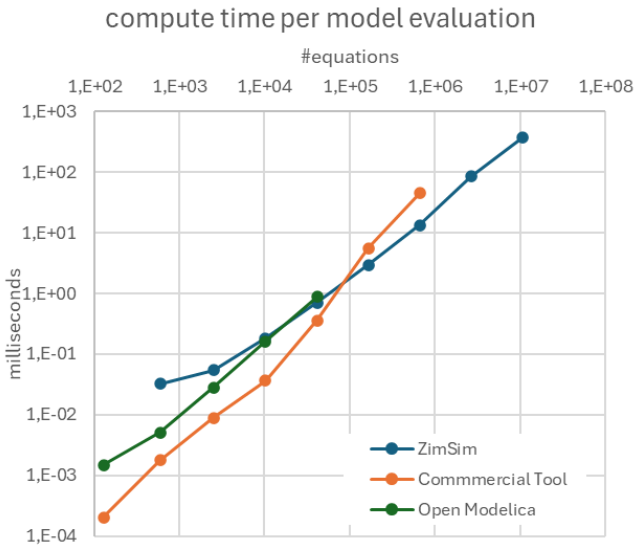


Figure 9: Computational Speed

DOI
10.3384/ecp22339

5 Outlook and Conclusions

This paper demonstrates that the current compilation technique for LIED systems in Modelica becomes increasingly useful, despite its current limitations and state of development.

The following statements can be made about the potential of the approach with reasonable confidence:

- Modeling according to the LIED methodology is feasible within Modelica and leads to understandable and useful Modelica code that can be processed by existing compilers without the need for any further modification.
- LIED systems can be compiled in modular form by a special compiler.
- The modular simulation code scales far better than monolithic Modelica code.
- Large linear equations systems can be reliably and efficiently solved but require the application of sparse methods which are currently only partially implemented.
- The modular simulation code can then be used and modified to enable different applications as the generation of fractal models or the creation of models from large input files using the advantages of modern programming languages.

Acknowledgement

This work was partially supported by the ITEA4 research project OpenScaling with financial support of the German federal ministry of education and research referring to the grant number: 01IS23062C

SPONSORED BY THE

Federal Ministry of Education and Research

References

Abdelhak, K., F. Casella, B. Bachmann (2023) Pseudo Array Causalization. Proceedings of the 15th International Modelica Conference, Aachen

Agosta F., et al. (2023) MARCO: An Experimental High-Performance Compiler for Large-Scale Modelica Models Proceedings of the 15th International Modelica Conference, Aachen.

Andresen, Lisa and Dubucq, Pascal and {Peniche Garcia}, Ricardo and Ackermann, Günter and Kather, Alfons and Schmitz, Gerhard (2015). Status of the TransiEnt Library: Transient Simulation of Coupled Energy Networks with High Share of Renewable Energy. 695--705

Assefaw v H. Gebremedhin, Fredrik Manne, and Alex Pothen (2005). What color is your jacobian? graph coloring for computing derivatives. SIAM Rev., 47(4):629–705

Benveniste, Albert, Benoît Caillaud, Hilding Elmqvist, et al. (2019). “Multi-Mode DAE Models - Challenges, Theory and Implementation”. In: Computing and Software Science – State of the Art and Perspectives. Vol. 10000. Lecture Notes in Computer Science. Springer, pp. 283–310.

Benveniste, A., B. Caillaud, M. Malandain, J. Thibault (2023) Towards the separate compilation of Modelica: modularity and interfaces for the index reduction of incomplete DAE systems. In: Proceedings of the 15th International Modelica Conference, Aachen, Germany.

Caillaud, B. A. Benveniste, M. Malandain (2025) Benchmarking the Modular Structural Analysis Algorithm. . In: Proceedings of the 16th International Modelica Conference, Lucerne, Switzerland.

Casella, Francesco (2015). "Simulation of Large-Scale Models in Modelica: State of the Art and Future Perspectives", Proceedings of the 11th International Modelica Conference, doi:10.3384/ecp15118459

Gillner, M., Westphal, J., Wiegel, B., Steffen, T., Urbansky, J., Hagemeier, A., Ruppert, S., Heyer, A., Benthin, J., Hanke, T., Brunnemann, J., Becker, C. & Speerforck, A. (2025, October). Status of the transient library: Transient simulation of complex integrated energy systems. In *Modelica Conferences* (pp. 989-999)

McGeer, T., "Passive walking with knees," *Proceedings., IEEE International Conference on Robotics and Automation*, Cincinnati, OH, USA, 1990, pp. 1640-1645 vol.3

Neumayr, A.; Otter, (2023) M. Modelling and Simulation of Physical Systems with Dynamically Changing Degrees of Freedom. *Electronics*, doi.org/10.3390/electronics12030500

Tarjan, R. E. (1972), "Depth-first search and linear graph algorithms" (PDF), SIAM Journal on Computing, 1 (2): 146–160

Zimmer, D., N. Weber, M. Meißner (2022) The DLR ThermoFluid Stream Library. MDPI Electronics - Special Issue.

Zimmer, D., C. Oldemeyer (2023). “Introducing Dialectic Mechanics”. Proceedings of the 15th International Modelica Conference, Aachen.

Zimmer D. (2024) Object-Oriented Modeling of Classic Physical Systems using Linear Implicit Equilibrium Dynamics Preprints 2024, 2024031139

Zimmer D. (2024). Testing Large Scale System Simulation using Linear Implicit Equilibrium Dynamics. In: Proceedings of the Asian Modelica Conference 2024, Jeju Island, Korea.

Zimmer, D. (2025). The Value of Enforcing a Strict Modeling Methodology within Modelica In: Proceedings of the 16th International Modelica Conference, Lucerne, Switzerland.

Appendix

This simulation code fragment in C++ has been automatically generated by the specialized Modelica compiler for LIED systems. Then the highlighted grey parts have been either added or modified to enable a fully fractal model generation within the C++ framework. As can be seen, the C++ code reflects the original model structure. Highly repetitive elements have been omitted here and marked by [...] for the sake of a more concise presentation. Many smaller modifications simply result from turning a statically allocated member into a dynamically allocated one. Albeit this fractal model generation is a purely academic example for the sake of scaling, the example shows that modular simulation code allows for a much more flexible creation of simulation code in later processing stages. Consider that the district heating network shall be read in from a large file. It is probably more feasible to develop the fundamental components in Modelica but then to compose the very large model, using the generated modular simulation code.

```
class DistributionGridNComp : public DistributionGridBase {
public:
    TransportPipeDarcyComp transportPipeDarcy{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy1{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    DistributionGridBase* distributionGrid1;
    DistributionGridBase* distributionGrid;
    SplitterComp splitter{0.01};
    SplitterComp splitter1{0.01};
    TransportPipeDarcyComp transportPipeDarcy2{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy3{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    DistributionGridBase* distributionGrid2_2;
    DistributionGridBase* distributionGrid2_1;
    TransportPipeDarcyComp transportPipeDarcy4{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy5{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    SplitterComp splitter2{0.01};
    TransportPipeDarcyComp transportPipeDarcy6{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy7{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy8{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy9{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    JunctionComp junction{0.01,1e-09};
    JunctionComp junction1{0.01,1e-09};
    JunctionComp junction2{0.01,1e-09};
    TransportPipeDarcyComp transportPipeDarcy10{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy11{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    TransportPipeDarcyComp transportPipeDarcy12{4184,1000,0.001,r,l,(273.15+10),0.03,2};
    DistributionGridNComp(Real r,Real l, int n) : DistributionGridBase(r,l,n) {
        if (n > 1) {
            distributionGrid1 = new DistributionGridNComp(0.25, 30, n - 1);
            distributionGrid = new DistributionGridNComp(0.25,30, n-1);
            distributionGrid2_2 = new DistributionGridNComp(0.25,30,n-1 );
            distributionGrid2_1 = new DistributionGridNComp(0.25,30, n-1 );
        }
        else {
            distributionGrid1 = new ConsumerComp();
            distributionGrid = new ConsumerComp();
            distributionGrid2_2 = new ConsumerComp();
            distributionGrid2_1 = new ConsumerComp();
        }

   ~DistributionGridNComp() {
        delete distributionGrid;
        delete distributionGrid1;
        delete distributionGrid2_2;
        delete distributionGrid2_1;
    };
```


DOI
10.3384/ecp22339

```
  void eval1();
  […]
  void eval81();

  void metainfo(Meta& meta) {
      meta.regComp(&inlet, "Inlet" );
      meta.regComp(&outlet, "Outlet" );
      […]
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval1(); }, Signals{&transportPipeDarcy2.outlet.signal1},
Signals{&distributionGrid->inlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval2(); }, Signals{&transportPipeDarcy2.outlet.signal2},
Signals{&distributionGrid->inlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval3(); }, Signals{&transportPipeDarcy7.inlet.signal3},
Signals{&distributionGrid->outlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval4(); }, Signals{&transportPipeDarcy3.outlet.signal1},
Signals{&distributionGrid1->inlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval5(); }, Signals{&transportPipeDarcy3.outlet.signal2},
Signals{&distributionGrid1->inlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval6(); }, Signals{&transportPipeDarcy6.inlet.signal3},
Signals{&distributionGrid1->outlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval7(); }, Signals{&transportPipeDarcy4.outlet.signal1},
Signals{&distributionGrid2_1->inlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval8(); }, Signals{&transportPipeDarcy4.outlet.signal2},
Signals{&distributionGrid2_1->inlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval9(); }, Signals{&transportPipeDarcy8.inlet.signal3},
Signals{&distributionGrid2_1->outlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval10(); }, Signals{&transportPipeDarcy5.outlet.signal1},
Signals{&distributionGrid2_2->inlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval11(); }, Signals{&transportPipeDarcy5.outlet.signal2},
Signals{&distributionGrid2_2->inlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval12(); }, Signals{&transportPipeDarcy9.inlet.signal3},
Signals{&distributionGrid2_2->outlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval13(); }, Signals{&transportPipeDarcy.inlet.signal3},
Signals{&inlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval14(); }, Signals{&transportPipeDarcy7.outlet.signal1},
Signals{&junction.inletA.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval15(); }, Signals{&transportPipeDarcy7.outlet.signal2},
Signals{&junction.inletA.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval16(); }, Signals{&transportPipeDarcy6.outlet.signal1},
Signals{&junction.inletB.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval17(); }, Signals{&transportPipeDarcy6.outlet.signal2},
Signals{&junction.inletB.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval18(); }, Signals{&transportPipeDarcy11.inlet.signal3},
Signals{&junction.outlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval19(); }, Signals{&transportPipeDarcy8.outlet.signal1},
Signals{&junction1.inletA.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval20(); }, Signals{&transportPipeDarcy8.outlet.signal2},
Signals{&junction1.inletA.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval21(); }, Signals{&transportPipeDarcy9.outlet.signal1},
Signals{&junction1.inletB.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval22(); }, Signals{&transportPipeDarcy9.outlet.signal2},
Signals{&junction1.inletB.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval23(); }, Signals{&transportPipeDarcy12.inlet.signal3},
Signals{&junction1.outlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval24(); }, Signals{&transportPipeDarcy11.outlet.signal1},
Signals{&junction2.inletA.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval25(); }, Signals{&transportPipeDarcy11.outlet.signal2},
Signals{&junction2.inletA.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval26(); }, Signals{&transportPipeDarcy12.outlet.signal1},
Signals{&junction2.inletB.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval27(); }, Signals{&transportPipeDarcy12.outlet.signal2},
Signals{&junction2.inletB.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval28(); }, Signals{&outlet.signal3},
Signals{&junction2.outlet.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval29(); }, Signals{&junction2.outlet.signal1},
Signals{&outlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval30(); }, Signals{&junction2.outlet.signal2},
Signals{&outlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval31(); }, Signals{&transportPipeDarcy.outlet.signal1},
Signals{&splitter.inlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval32(); }, Signals{&transportPipeDarcy.outlet.signal2},
Signals{&splitter.inlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval33(); }, Signals{&transportPipeDarcy1.inlet.signal3},
Signals{&splitter.outletA.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval34(); }, Signals{&transportPipeDarcy10.inlet.signal3},
Signals{&splitter.outletB.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval35(); }, Signals{&transportPipeDarcy1.outlet.signal1},
Signals{&splitter1.inlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval36(); }, Signals{&transportPipeDarcy1.outlet.signal2},
Signals{&splitter1.inlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval37(); }, Signals{&transportPipeDarcy2.inlet.signal3},
Signals{&splitter1.outletA.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval38(); }, Signals{&transportPipeDarcy3.inlet.signal3},
Signals{&splitter1.outletB.signal3});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval39(); }, Signals{&transportPipeDarcy10.outlet.signal1},
Signals{&splitter2.inlet.signal1});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval40(); }, Signals{&transportPipeDarcy10.outlet.signal2},
Signals{&splitter2.inlet.signal2});
    meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval41(); }, Signals{&transportPipeDarcy4.inlet.signal3},
Signals{&splitter2.outletA.signal3});
```

```
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval42(); }, Signals{&transportPipeDarcy5.inlet.signal3},
    Signals{&splitter2.outletB.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval43(); }, Signals{&inlet.signal1},
    Signals{&transportPipeDarcy.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval44(); }, Signals{&inlet.signal2},
    Signals{&transportPipeDarcy.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval45(); }, Signals{&splitter.inlet.signal3},
    Signals{&transportPipeDarcy.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval46(); }, Signals{&splitter.outletA.signal1},
    Signals{&transportPipeDarcy1.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval47(); }, Signals{&splitter.outletA.signal2},
    Signals{&transportPipeDarcy1.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval48(); }, Signals{&splitter1.inlet.signal3},
    Signals{&transportPipeDarcy1.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval49(); }, Signals{&splitter.outletB.signal1},
    Signals{&transportPipeDarcy10.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval50(); }, Signals{&splitter.outletB.signal2},
    Signals{&transportPipeDarcy10.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval51(); }, Signals{&splitter2.inlet.signal3},
    Signals{&transportPipeDarcy10.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval52(); }, Signals{&junction.outlet.signal1},
    Signals{&transportPipeDarcy11.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval53(); }, Signals{&junction.outlet.signal2},
    Signals{&transportPipeDarcy11.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval54(); }, Signals{&junction2.inletA.signal3},
    Signals{&transportPipeDarcy11.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval55(); }, Signals{&junction1.outlet.signal1},
    Signals{&transportPipeDarcy12.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval56(); }, Signals{&junction1.outlet.signal2},
    Signals{&transportPipeDarcy12.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval57(); }, Signals{&junction2.inletB.signal3},
    Signals{&transportPipeDarcy12.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval58(); }, Signals{&splitter1.outletA.signal1},
    Signals{&transportPipeDarcy2.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval59(); }, Signals{&splitter1.outletA.signal2},
    Signals{&transportPipeDarcy2.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval60(); }, Signals{&distributionGrid->inlet.signal3},
    Signals{&transportPipeDarcy2.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval61(); }, Signals{&splitter1.outletB.signal1},
    Signals{&transportPipeDarcy3.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval62(); }, Signals{&splitter1.outletB.signal2},
    Signals{&transportPipeDarcy3.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval63(); }, Signals{&distributionGrid1->inlet.signal3},
    Signals{&transportPipeDarcy3.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval64(); }, Signals{&splitter2.outletA.signal1},
    Signals{&transportPipeDarcy4.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval65(); }, Signals{&splitter2.outletA.signal2},
    Signals{&transportPipeDarcy4.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval66(); }, Signals{&distributionGrid2_1->inlet.signal3},
    Signals{&transportPipeDarcy4.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval67(); }, Signals{&splitter2.outletB.signal1},
    Signals{&transportPipeDarcy5.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval68(); }, Signals{&splitter2.outletB.signal2},
    Signals{&transportPipeDarcy5.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval69(); }, Signals{&distributionGrid2_2->inlet.signal3},
    Signals{&transportPipeDarcy5.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval70(); }, Signals{&distributionGrid1->outlet.signal1},
    Signals{&transportPipeDarcy6.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval71(); }, Signals{&distributionGrid1->outlet.signal2},
    Signals{&transportPipeDarcy6.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval72(); }, Signals{&junction.inletB.signal3},
    Signals{&transportPipeDarcy6.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval73(); }, Signals{&distributionGrid->outlet.signal1},
    Signals{&transportPipeDarcy7.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval74(); }, Signals{&distributionGrid->outlet.signal2},
    Signals{&transportPipeDarcy7.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval75(); }, Signals{&junction.inletA.signal3},
    Signals{&transportPipeDarcy7.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval76(); }, Signals{&distributionGrid2_1->outlet.signal1},
    Signals{&transportPipeDarcy8.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval77(); }, Signals{&distributionGrid2_1->outlet.signal2},
    Signals{&transportPipeDarcy8.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval78(); }, Signals{&junction1.inletA.signal3},
    Signals{&transportPipeDarcy8.outlet.signal3});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval79(); }, Signals{&distributionGrid2_2->outlet.signal1},
    Signals{&transportPipeDarcy9.inlet.signal1});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval80(); }, Signals{&distributionGrid2_2->outlet.signal2},
    Signals{&transportPipeDarcy9.inlet.signal2});
        meta.addBlock(this, [](Component* c) { return ((DistributionGridNComp*)c)->eval81(); }, Signals{&junction1.inletB.signal3},
    Signals{&transportPipeDarcy9.outlet.signal3});
    };
};
```


DOI
10.3384/ecp22339

SMC: A Cloud-Deployable Modelica Frontend Service in Go

Hailong Wang*[1] Qingda Xu[1] Enyang Zhang[1] Longsheng Sun[1] Rui Gao[1] Aiguo Xu[1]

[1]Nanjing Yuansi SimTek Co., Ltd, China

Abstract

Web-based modeling platforms and automated model-checking workflows in CI/CD pipelines require remotely callable Modelica frontend services. This paper presents SMC (Simtek Modelica Compiler), a service-oriented Modelica frontend implemented in Go. SMC exposes model loading, structured queries, diagnostics, and flattening through a service API backed by a pipeline for parsing, instantiation, semantic analysis, connection processing, and flat-model generation. Drawing on the SMC implementation, we also discuss the engineering implications of Go for single-binary deployment, concurrent processing, representation of Modelica language constructs, and memory allocation. We demonstrate the prototype through integration with a web modeling platform. A common set of 606 comparable classes from the Modelica Standard Library (MSL) 4.0.0 is used both to assess frontend-result consistency against the OpenModelica Compiler (OMC) and to measure flattening time, memory allocation, and garbage-collection behavior. The evaluation covers frontend artifacts and in-process processing costs; backend processing, code generation, simulation, and FMU export are outside its scope.

Keywords: Modelica, Go, frontend compiler, service-oriented, flattening

1 Introduction

Modelica is a unified language for modeling multidomain physical systems (Elmqvist et al., 1999; Modelica Association, 2023). A typical Modelica toolchain comprises a frontend for parsing, instantiation, semantic analysis, and flattening; a backend for equation transformations, causalization, state selection, and code generation; and a simulation runtime for numerical integration (Pop et al., 2019). This paper focuses on the frontend.

In desktop Modelica environments, frontend functions such as library indexing, class and annotation queries, and model checking are often accessed through tool-specific interfaces. Web applications and automated checking workflows require programmatic access to these functions. Modeling interfaces are also moving toward web platforms, such as Modelon Impact (Kumar et al., 2022), while model checking, batch compilation, and regression testing are increasingly integrated into CI/CD workflows. These use cases motivate a remotely callable frontend with an explicit service boundary.

The OpenModelica Compiler (OMC) can be invoked through OMJulia or command-line scripts (Fritzson et al., 2020; OpenModelica Contributors, 2026a) and is distributed as an official Docker image (OpenModelica Contributors, 2026b). Dymola supports Java, Python, and JavaScript interfaces (Dassault Systèmes, 2024). Modiator provides browser-based Modelica simulation through a web application (Elmqvist and Otter, 2025). These tools provide several forms of external integration. SMC addresses the related design question of exposing frontend operations through a service API.

SMC exposes model loading, structured queries, diagnostics, and flattening through a service API. These operations invoke the frontend pipeline. This paper contributes the design and prototype evaluation of this service-oriented frontend organization; the Go implementation provides the engineering context for the reported trade-offs and measurements.

Table 1 places SMC in context by summarizing the implementation languages and external interfaces reported for related Modelica tools.

OMC is a mature open-source Modelica compiler (Fritzson et al., 2020) whose new frontend (NF) provides an important engineering foundation for instantiation, lookup, and flattening. OpenModelica.jl (Tinnerholm et al., 2021) leverages Julia's type system and metaprogramming capabilities to build a modular frontend architecture integrated with the Julia scientific computing ecosystem. Rumoca is a Rust-native Modelica compiler that translates models to algebraic modeling languages (Condie et al., 2025). We do not aim for functional completeness comparisons. Instead, we focus on SMC's service-oriented frontend organization and use its Go implementation to examine practical compiler-engineering trade-offs.

To examine the proposed service-oriented frontend and its Go implementation, we address three complementary engineering questions:

1. How does SMC organize Modelica frontend capabilities as remotely callable service operations?
2. What engineering benefits and limitations arise from implementing this frontend in Go?
3. What in-process frontend processing costs does the prototype exhibit at standard-library scale?

The evaluation covers a tested subset of Modelica 3.6

*Corresponding author: hailong.wang@cnydsimtek.com

Table 1. Overview of related Modelica tools from public materials

Tool	Language	Architecture	External integration (from public materials)
OMC Frontend	MetaModelica	Compiler library	CLI wrapper / OMJulia / omc-docker image
Dymola	Not publicly disclosed (closed source)	Commercial IDE	Java / Python / JavaScript interfaces
OpenModelica.jl	Julia	Modular compiler framework	Julia package ecosystem; ModelingToolkit.jl backend
Modelon Impact	Not publicly disclosed (closed source)	Commercial cloud platform	Platform-level REST API / third-party connectivity; no standalone frontend compiler interface described in public materials
SMC (this work)	Go	Service-oriented frontend	Service API, single-binary deployment, frontend-only scope

frontend behavior. The regression cases exercise class and component instantiation; inheritance, modifiers, and redeclarations; name lookup; typed expressions and array dimensions; equations and algorithms; records and functions; connection processing; and flat-model generation. This list describes tested areas, not complete support for each area or for Modelica 3.6 as a whole. Backend processing, numerical simulation, code generation, external-function execution, and FMU export are outside the current scope.

2 Engineering Considerations of the Go Implementation

Go was selected for engineering reasons rather than language exclusivity; comparable facilities are available in other implementation languages. For SMC, Go provides one environment for building, testing, profiling, and deploying the frontend, together with lightweight concurrency and request cancellation. This section examines that choice in the context of a service-oriented Modelica frontend.

2.1 Four Engineering Considerations

1. Single-binary deployment.

Go compiles SMC into a native executable that can be deployed without a separate language runtime. The container image can therefore be built from `scratch`, which keeps the runtime environment small and allows the same executable to be used in development, testing, CI, and cloud deployments.

In this paper, *cloud-deployable* denotes packaging SMC as a remotely accessible containerized service. Orchestration, elastic scaling, and fault tolerance are outside the evaluated scope.

2. Unified Go toolchain.

From lexical analysis to service integration, SMC is maintained as a single Go project, sharing the same build, test, and profiling toolchain:

```
go build -o smc .
go test ./...
go test -bench . -benchmem
go tool pprof
```

The compiler, service layer, tests, benchmarks, and profiling tools are maintained in one Go project without additional cross-language adapters.

3. Automatic memory management across frontend stages.

SMC creates intermediate representations with different lifetimes, including parse trees, structured models, instance models, and semantic models. Go's garbage collector reduces the amount of explicit lifetime management required for these objects. This simplifies the current implementation, although allocation volume and collection frequency remain important constraints for structurally complex models.

Within our 606-model measurement scope, GC pause overhead (mean total pause below 0.1 ms per model) is relatively small compared to overall runtime. Automatic memory management also reduces the risk of errors from manual deallocation.

4. Concurrency suited to source loading and parsing.

Library loading involves directory traversal and the analysis of many independent Modelica source files. SMC uses goroutines and channels to organize these operations as a worker pool, with `Context` providing consistent error propagation and cancellation. This model matches the independence of the loading and parsing tasks and keeps their concurrency control within the same implementation environment.

2.2 Known Limitations and Mitigations

1. Lack of algebraic data types and pattern matching.

Go does not provide closed variants or exhaustive pattern matching, which can leave missing dispatch cases undetected as Modelica language structures evolve. SMC mitigates this risk through explicit structs, narrow interfaces, validation at constructors and stage entry points, and subset-based regression tests.

2. Allocation pressure during flattening.

Structurally complex models can produce large volumes of short-lived intermediate data, increasing GC activity and tail latency. Although the measurements do not identify GC as a general frontend bottleneck, SMC limits pointers in stage-local data and clears references after each stage. Further optimization will investigate object reuse and more compact data structures.

DOI
10.3384/ecp22349

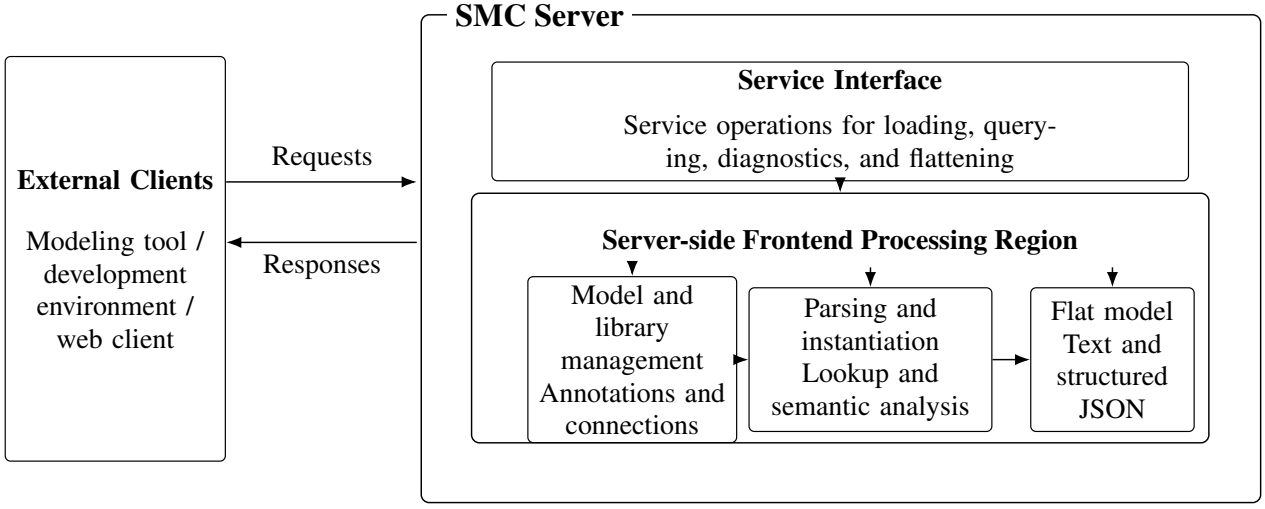


Figure 1. SMC service architecture

3 SMC System Overview

3.1 Overall Architecture

Figure 1 separates external clients, the service interface, and server-side frontend processing. The interface accepts requests for model loading, queries, diagnostics, and flattening. Server-side components manage libraries and model metadata, execute the frontend pipeline, and return flat Modelica text or structured JSON through the service API.

3.2 Service Interface

Table 2 lists representative operations of the SMC service API. The operations accept library paths or fully qualified class names and return structured data or text.

Table 2. Representative operations of the SMC service API

Operation	Function
`LoadLibrary`	Load a Modelica library, return library index and loading diagnostics
`QueryClass`	Query class summary information
`QueryComponents`	Query component list and metadata
`QueryAnnotations`	Retrieve icon, diagram, documentation, and experiment annotations
`QueryConnections`	Query connection relationships
`CheckModel`	Run full-stage checking, return diagnostics
`FlattenModel`	Generate text or JSON flattened output

3.3 Frontend Processing Stages

Figure 2 summarizes the SMC frontend pipeline. Processing proceeds from source loading and parsing through structured model construction, instantiation, name lookup, semantic analysis, and flattening. The following paragraphs describe the data and diagnostics produced at each stage.

The pipeline follows the sequence shown in Figure 2. **Source loading** traverses model directories, reads `.mo` files, and interprets `package.order` files to construct

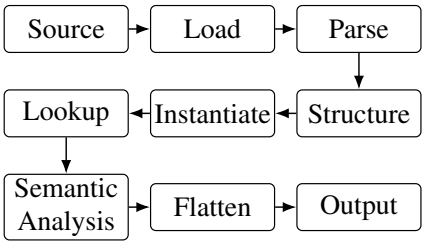


Figure 2. SMC frontend processing pipeline

source units. **Syntax analysis** parses these units concurrently. **Structured construction** converts the parse trees into a queryable representation of classes, components, equations, and annotations.

Instantiation establishes the target class context and expands component hierarchies. **Name lookup and semantic analysis** then resolve references, inheritance, modifiers, redeclarations, and array dimensions. **Flattening** produces parameters, variables, equations, algorithms, function dependencies, and experiment metadata.

Before the `FlatModel` is emitted, connection processing resolves structural and component-array endpoints, constructs connection sets, and elaborates `connect` equations into the potential, flow, and stream relations supported by the tested subset. The same step handles the expandable and overconstrained connection cases covered by the regression suite. The resulting `FlatModel` is serialized as flat Modelica text or structured JSON.

Diagnostics can be produced at every stage and follow a common schema. Each diagnostic includes `severity`, `phase`, and `message`. When available, it also includes `sourceLocation`, `relatedReference`, and `suggestion`. The `phase` field identifies the stage that produced the diagnostic, allowing clients to group diagnostics by origin.

3.4 Flattened Output

SMC derives two output views from the same `FlatModel`: flat Modelica text and structured JSON.

The text view supports inspection and debugging. The JSON view serializes the `FlatModel` fields exposed to service clients and is not treated as an independent frontend result. Structured JSON is an interface-design choice rather than a consequence of using Go; its explicit fields allow automated tools to consume frontend results without parsing compiler-specific text formatting.

Listing 1 gives an excerpt of the JSON output for `Modelica.Blocks.Examples.PID_Controller`. The excerpt follows the actual artifact schema: `variables_list` contains flattened variables and parameters, `equation_section_list` contains flattened equations, and `experiment` contains experiment settings.

Listing 1. Example of JSON-formatted flattened model output (`Modelica.Blocks.Examples.PID_Controller` excerpt)

```
{
  "class_name": "Modelica.Blocks.Examples.
      PID_Controller",
  "variables_list": [
    {
      "typeSpecifier": "Real",
      "typePrefix": "parameter",
      "binding": "1.570796326794897",
      "name": "driveAngle"
    },
    {
      "typeSpecifier": "Real",
      "name": "spring.flange_b.phi"
    }
  ],
  "equation_section_list": [
    "PI.y = torque.tau",
    "speedSensor.w = PI.u_m"
  ],
  "experiment": { "stopTime": "4" }
}
```

With this JSON structure, automated tools can access flattened parameters, variables, and equations without parsing compiler-specific textual output. The web platform integration case study uses structured query results for class hierarchies, components, connections, and graphical annotations returned by the service API.

3.5 Web Modeling Platform Integration Case Study

Figure 3 shows SMC integrated into a web modeling platform. After the platform submits library paths, SMC returns the class hierarchy used to construct the model tree. Component, connection, and annotation queries provide the data used to render the graphical view. The browser handles presentation, while SMC performs Modelica semantic analysis.

In this integration, the platform obtains the frontend results required for model browsing, graphical rendering, and diagnostic checking through the SMC service API. No compiler installation or local Modelica state is required in the browser.

4 Evaluation Method and Results

This section evaluates frontend-result consistency against OMC and reports SMC's processing costs for the same evaluation set.

The candidate set comprised all 650 non-partial model, block, and class definitions under MSL 4.0.0 `Examples` packages. OMC 1.27.0 could not flatten 41 of them under the evaluation configuration. SMC excluded one further class that requires an unsupported file-reading built-in function and two nested `PointMass` classes under `PointGravityWithPointMasses2` whose top-level `outer world` declarations are rejected when flattened as independent roots. The remaining 606 classes formed both the consistency and performance evaluation set, with complete timing, allocation, GC, and sampled-heap records. Models are identified by fully qualified class names. Representative classes from four library domains are:

- Analog electrical: `Modelica.Electrical.Analog.Examples.Rectifier`
- Control signal: `Modelica.Blocks.Examples.PID_Controller`
- Mechanics: `Modelica.Mechanics.Rotational.Examples.CoupledClutches`; `Modelica.Mechanics.Translational.Examples.Oscillator`
- Heat transfer: `Modelica.Thermal.HeatTransfer.Examples.TwoMasses`

For each class, SMC and OMC receive the same fully qualified class name. After loading MSL 4.0.0, OMC 1.27.0 is invoked through its scripting interface, and `instantiateModel` generates flat Modelica text using the default frontend options; backend code generation and simulation are not requested. OMC's flat Modelica output is parsed into a structured comparison representation. SMC's textual and JSON outputs are generated from the same `FlatModel`; the JSON form is therefore a serialization of the frontend result rather than an independent semantic representation. SMC's `FlatModel` preserves array-valued structures rather than systematically scalarizing them. To match the form emitted by OMC, array-valued declarations, equations, and expressions are scalarized only when constructing the comparison representation; this transformation does not modify the `FlatModel`.

Elements are aligned by fully qualified path and compared field by field. Declarations are compared by name, type, dimensions, variability, direction, attributes, and bindings. Equations are compared as structured expressions, whereas algorithm statements are compared in their original order. Function sets are first compared by fully qualified name; a function emitted by only one frontend is reported as a difference. For functions present in both results, the comparison covers signatures, inputs, outputs, local declarations, algorithm sections, and referenced dependencies. Normalization is limited to source locations, whitespace, redundant parentheses, equivalent numeric

DOI
10.3384/ecp22349

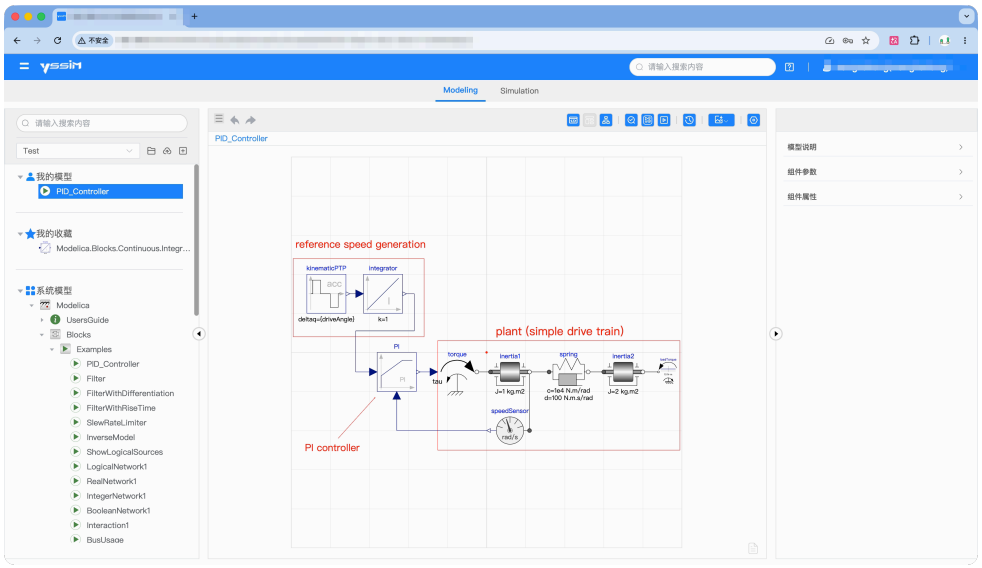


Figure 3. Web interface for Modelica Standard Library modeling via the SMC service

lexical forms, and an explicit mapping of tool-generated internal identifiers. Declaration collections without ordering semantics are sorted by fully qualified path; order-sensitive constructs retain their original order. Differences not resolved by these rules are inspected in the corresponding flattened results.

The automated procedure accepted 524 classes without unresolved differences. The flattened results for the remaining 82 classes were manually reviewed. Manual review examined the aligned declarations, equations, algorithms, and functions in the corresponding flattened results. A case was accepted only when the remaining differences preserved the same Modelica semantic expression under the comparison rules described above. All 82 cases satisfied this criterion. No unresolved differences were found in the compared frontend artifacts for the 606 evaluated classes under the stated comparison and review procedure. This result is limited to the frontend artifacts and comparison rules described above; backend processing, state selection, code generation, and simulation are not covered.

Performance measurements were conducted on a MacBook Pro with a 14-core Apple M3 Max processor and 36 GB of memory, running macOS 26.6.1 on arm64 and Go 1.25.6. The measurements used `GOMAXPROCS=14` and the default `GOGC` setting. The three timing runs were executed as separate processes.

Table 3 presents SMC's processing measurements for the Modelica Standard Library 4.0.0 (2508 `.mo` files, approximately 14 MB of source code, 327401 lines). The measurements distinguish Concrete Syntax Tree (CST) construction from the subsequent AST construction increment. Parsing measurements use one warm-up run followed by three formal runs averaged. Flattening time, memory allocation, and GC metrics are obtained from three independent runs of each of the 606 models. Per-model flattening time covers the execution of `FlattenModel` after library loading and excludes library loading, request transport, and response serialization. Per-model memory allocation is defined as the increase in `runtime.MemStats.TotalAlloc` measured immediately before and after `FlattenModel`; it represents cumulative allocation traffic rather than live heap usage or resident memory. The per-model median is used for flattening time and allocation, while GC counts and pauses summarize all 1818 model runs. Here, p95 denotes the 95th percentile. Sampled peak heap increase is measured separately in two instrumented runs per model. Before each instrumented run, garbage collection is triggered and `HeapAlloc` is recorded as the baseline. During `FlattenModel`, `HeapAlloc` is sampled every 1 ms. The sampled peak heap increase is the maximum observed `HeapAlloc` minus the baseline, and the larger value of the two runs is reported.

Table 3. SMC processing performance on MSL 4.0.0

Metric	Result
Library loading (incl. scanning + index construction)	Mean 3 s (3 runs)
CST construction	Mean 2 s (0.9 ms/file)
AST construction increment	Mean 0.2 s (0.1 ms/file)
Per-model flattening time (606 models; median of 3 runs)	Median 10 ms, mean 45 ms, p95 165 ms
Per-model memory allocation (606 models; median of 3 runs)	Mean 57 MB, max 2598 MB
Per-model sampled peak heap increase (606 models; max of 2 runs)	Median 13 MB, mean 54 MB, p95 234 MB, max 1783 MB
Per-model GC count (1818 runs)	Mean 0.01, max 2
Per-model GC total pause (1818 runs)	Mean < 0.1 ms, max 0.1 ms

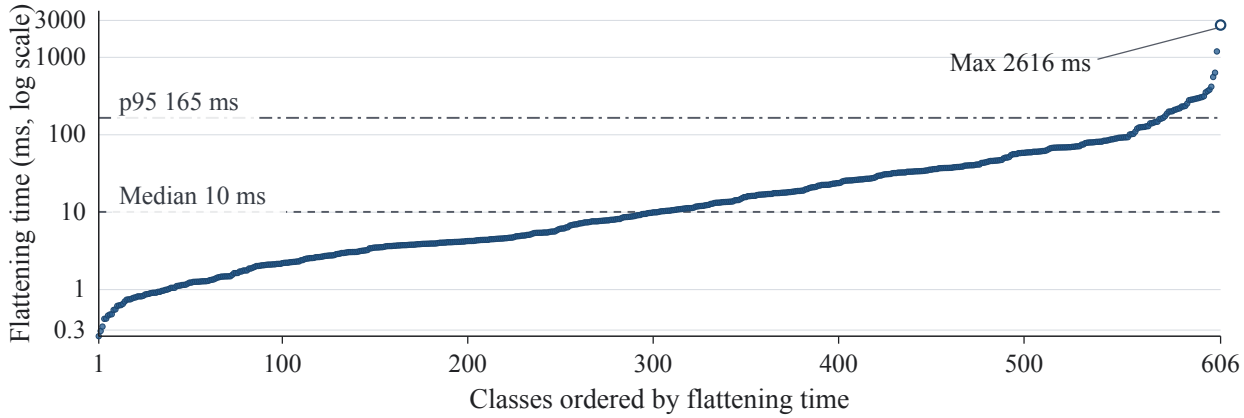


Figure 4. Ordered flattening times for 606 MSL classes (log scale; dashed lines: median and p95).

Full library loading averaged 3 s. CST construction averaged 0.9 ms per file, and the subsequent AST construction increment averaged 0.2 s. Across the 606 evaluated classes, the per-model median over three runs had an overall median of 10 ms, a mean of 45 ms, and a p95 of 165 ms. These measurements characterize isolated frontend processing; they exclude request transport, response serialization, concurrent load, and end-to-end service latency.

Figure 4 plots the per-model median of three runs for all 606 classes in ascending order. The logarithmic vertical scale shows both the main distribution and the slower outliers. The median is 10 ms, the p95 is 165 ms, and the maximum is 2616 ms, producing a long upper tail relative to the median.

Mean per-model allocation was 57 MB, with a maximum of 2598 MB. The sampled peak heap increase had a median of 13 MB and a maximum of 1783 MB. Because heap usage was sampled every 1 ms, shorter peaks may not have been observed. GC occurred in 11 of the 1818 timed runs, with a mean count of 0.01 per model run; the maximum count was 2 and the maximum total pause was 0.1 ms. Reducing the allocation tail will require more compact data structures and object reuse. Deployment-level `GOGC` tuning may separately adjust the memory-latency trade-off.

5 Discussion

This section separates implementation-language effects from interface-design effects and summarizes the limitations of the prototype.

5.1 Implementation and Architectural Implications

Go provides a common environment for building, testing, profiling, and deploying SMC; goroutines and `Context` support concurrent frontend tasks and cancellation. The service interface is a separate design choice that makes frontend operations programmatically accessible and returns structured data. A similar interface and pipeline organization could be implemented in another language.

At the architectural level, clients obtain model information, diagnostics, and flattened output through an API rather than by launching and managing compiler processes. Structured JSON provides a machine-readable representation for web interfaces, CI pipelines, and other automated tools. These properties arise from the service and interface design, not from Go.

5.2 Current Limitations and Future Work

Allocation scalability: The measurements reveal a long allocation tail for structurally complex models (Table 3). Although GC pauses were small in the evaluated runs, sustained high allocation may affect long-running service workloads. SMC's `FlatModel` preserves array-valued structures rather than systematically scalarizing them; scalarization for comparison with OMC is applied only to the comparison representation. Further profiling will examine transient semantic objects, lookup contexts, modifier and connection metadata, and output construction. Future work will investigate object reuse and more compact data structures.

SMC currently also has the following known limitations:

Backend integration: SMC currently ends at frontend artifacts. Backend equation processing, code generation, numerical simulation, and FMU export are not implemented in the evaluated prototype.

Runtime boundary: The GC measurements apply to SMC as a standalone frontend service. SMC does not embed the Go runtime in generated FMUs or simulation masters. Runtime interactions in those settings are therefore outside the present scope and would require separate evaluation if runtime code generation were added.

Availability: SMC remains an internal prototype and has not been publicly released. No open-source release is currently planned.

Prototype maturity: The flattened-output schema remains under active refinement. Its fields and representations will evolve with semantic coverage, tool integration, and validation requirements.

DOI
10.3384/ecp22349

6 Conclusion

This paper presented the service interface and frontend pipeline of SMC, a Modelica frontend implemented in Go. The prototype was demonstrated through integration with a web modeling platform. The same 606 comparable MSL 4.0.0 classes were used for structured comparison with OMC's flat Modelica output and for measurements of flattening time, memory allocation, and garbage-collection behavior. Library loading averaged 3 s, median per-model flattening time was 10 ms, and allocation exhibited a long upper tail. No unresolved differences were found under the stated comparison and review procedure. These measurements characterize the evaluated frontend workload rather than end-to-end service latency.

The current prototype excludes backend processing, code generation, numerical simulation, and FMU export. Future work will extend regression coverage and reduce allocation in structurally complex cases through object reuse and more compact data structures.

References

Micah Condie, Abigaile Woodbury, James Goppert, and Joel Andersson. Rumoca: Towards a translator from Modelica to algebraic modeling languages. In *Proceedings of the 16th International Modelica & FMI Conference*, pages 1009–1016, 2025. doi:10.3384/ecp2181009.

Dassault Systèmes. *Dymola 2024x Refresh 1 Release Notes*, 2024. URL `https://www.3ds.com/assets/invest/2024-05/dymola-2024x-r1-release-notes.pdf`.

Hilding Elmqvist and Martin Otter. Modiator—a web app for Modelica simulation. In *Proceedings of the 16th International Modelica & FMI Conference*, pages 211–221, 2025. doi:10.3384/ecp218211.

Hilding Elmqvist, Sven Erik Mattsson, and Martin Otter. Modelica—a language for physical system modeling, visualization and interaction. In *Proceedings of the 1999 IEEE International Symposium on Computer Aided Control System Design*, pages 630–639. IEEE, 1999. doi:10.1109/CACSD.1999.808720.

Peter Fritzson, Adrian Pop, Karim Abdelhak, Adeel Asghar, Bernhard Bachmann, Willi Braun, Daniel Bouskela, Robert Braun, Lena Buffoni, Francesco Casella, Rodrigo Castro, Rüdiger Franke, Dag Fritzson, Mahder Gebremedhin, Andreas Heuermann, Bernt Lie, Alachew Mengist, Lars Mikelsons, Kannan Moudgalya, Lennart Ochel, Arunkumar Palanisamy, Vitalij Ruge, Wladimir Schamai, Martin Sjölund, Bernhard Thiele, John Tinnerholm, and Per Östlund. The OpenModelica integrated environment for modeling, simulation, and model-based development. *Modeling, Identification and Control*, 41(4):241–285, 2020. doi:10.4173/mic.2020.4.1.

Abhilash Kumar, Arunkumar Narasimhan, Tharrini Rajendran, and Stéphane Velut. Digital twin applications using a cloud native Modelica platform. In *Proceedings of the Asian Modelica Conference 2022*, volume 193, pages 75–80, Tokyo, Japan, 2022. doi:10.3384/ecp19375.

Modelica Association. *Modelica Language Specification, Version 3.6*, 2023. URL `https://specification.modelica.org/`.

OpenModelica Contributors. *OMJulia – OpenModelica Julia Scripting*, 2026a. URL `https://openmodelica.org/doc/OpenModelicaUsersGuide/v1.26.3/omjulia.html`. Accessed 2026-08-04.

OpenModelica Contributors. OpenModelica download docker (all os), 2026b. URL `https://openmodelica.org/download/docker/`. Accessed 2026-08-04.

Adrian Pop, Per Östlund, Francesco Casella, Martin Sjölund, and Rüdiger Franke. A new OpenModelica compiler high performance frontend. In *Proceedings of the 13th International Modelica Conference*, volume 157, pages 689–698, Regensburg, Germany, 2019. doi:10.3384/ecp19157689.

John Tinnerholm, Adrian Pop, Andreas Heuermann, and Martin Sjölund. OpenModelica.jl: A modular and extensible Modelica compiler framework in Julia targeting ModelingToolkit.jl. In *Proceedings of the 14th International Modelica Conference*, volume 181, pages 109–117, Linköping, Sweden, 2021. doi:10.3384/ecp21181109.

DOI
10.3384/ecp22349

Research on PINN Data-Driven Modeling Method Based on Modelica Model

Kaixuan Zhang[1] Yanfang Liu[1*] Yuan Sun[12] Heng Wang[2]
Xudong Wang[1] Zhidong Zhang[3] Wei Wu[3]

[1]School of Transportation Science and Engineering, Beihang University, China,
{kaixuanzhang,liuyf,wangxdxd}@buaa.edu.cn

[2]State Key Laboratory of Robotics and Intelligent Systems, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China, {sunyuan,wangheng}@sia.cn

[3]Shandong Weihe Transmission Co., Ltd, Shandong 273300, P.R. China,
zzd333@126.com,ww15265913688@163.com

Abstract

In practical engineering, equipment parameters are difficult to obtain and measurement data is limited. This paper proposes a PINN-Modelica data-driven modeling method that couples a neural network with a Modelica model encapsulated as an FMU. Instead of directly constructing governing-equation residuals in the proposed method, the FMU supplies a modular physics-based constraint during training. The neural network predicts both terminal voltage and time-varying physical parameters, while the FMU computes a physics-based voltage from the predicted parameters. Data and physics losses jointly constrain the neural-network voltage output, and the sensitivity of the physics loss to the FMU input parameters is evaluated numerically by central finite differences and combined with neural-network backpropagation. A second-order Thevenin battery model is used as a proof-of-concept, and coupled simulation is implemented through the Functional Mock-up Interface (FMI) standard. The framework provides a reusable approach for integrating Modelica physical models with data-driven parameter identification.

Keywords: PINN, physical model, data-driven, parameter identification, FMI, Modelica

1 Introduction

In practical engineering, to simulate the operation of equipment, it is often necessary to construct simulation models. Mechanism-based models have clear physical significance and good generalization ability, but they rely on extensive prior knowledge and complex parameter identification processes, which are often limited by parameter measurement costs and computational efficiency in practical engineering applications. Black-box models can automatically learn complex nonlinear relationships from data without in-depth understanding of internal mechanisms, but they usually require a large amount of labeled data, have poor interpretability, are prone to overfitting, and struggle to ensure that prediction results conform to basic physical laws.

To address the aforementioned issues, many scholars have begun to investigate modeling approaches that integrate physical mechanism models with data-driven methods, which exhibit significant potential in enhancing model accuracy and reducing modeling workload (Hofmann A, Mikelsons L. 2025). Thummerer et al. (2021) introduced a library extension named FMIFlux.jl, which enables the integration of FMUs into neural network topologies to develop NeuralFMUs. This method combines the distinct advantages of black-box models and data-driven machine learning models, integrating the two modeling approaches within a unified development environment. Kamp et al. (2023) employed an exemplary NODE (Neural Ordinary Differential Equations) setup to train low-dimensional artificial neural networks with physically meaningful outputs, thereby enhancing dynamic models. This approach maintains the physical integrity of the model and provides the possibility of enforcing physical laws during the training process.

Currently, Physics-informed Neural Networks (PINNs) have attracted considerable attention in engineering applications (Velioglu et al. 2025; Chang et al. 2025),and this algorithm has become a relatively mature and widely recognized modeling framework for integrating physical mechanisms with neural networks (K Laugksch et al. 2023; Lawal Z K et al. 2025; Xiu F et al. 2024). Liu T et al. (2024) proposed the integration of Model Predictive Control (MPC) with PINNs, which improved the control system accuracy and operational efficiency of Autonomous Underwater Vehicles (AUVs). The effectiveness of MPC lies in its proficient handling of the inherent complex constraints and intrinsic nonlinear dynamics of AUV systems. Meanwhile, the PINN architecture incorporates fundamental physical laws represented by Partial Differential Equations (PDEs), enhancing the predictive fidelity of the system. Hongbo

Hu et al. (2024) proposed a PINN-based friction-inclusive dynamics modeling method for industrial robots. By applying this method, a hybrid learning strategy for robot dynamics and friction was designed, ensuring modeling accuracy while avoiding reliance on joint torque component labels. The paradigm of the PINN algorithm provides insights for constructing models of many engineering devices, however, real-world data is limited and costly(Brun J, Sergi T, Mutter S, et al. 2025). Consequently, data-driven model training often faces issues such as single-source and simplistic data, leading to trained models that cannot comprehensively exhibit the characteristics of the equipment.

To address the above issues, this paper proposes a Modelica-constrained physics-informed data-driven modeling method, referred to as PINN-Modelica. Rather than directly replacing the governing equations of a classical PINN, the method uses a Modelica/FMU model as a modular physical constraint in a neural-network-based parameter-identification framework. The neural network predicts terminal voltage and time-varying physical parameters from limited input-output data. The FMU returns a physics-based voltage, and its influence on the parameter branch is introduced through numerically evaluated finite-difference sensitivities. This design reduces manual construction of physical residual losses while retaining physically interpretable intermediate parameters.

2 Modeling Method

The core idea of traditional PINNs is to incorporate governing equations, often partial or ordinary differential equations, as residual-based regularization terms in the neural-network loss so that the network output fits observations while satisfying physical laws. In contrast, this work encapsulates the governing physical model in an FMU and uses its simulation output as a physics-consistency constraint. Therefore, PINN-Modelica is formulated as a physics-informed parameter-identification framework inspired by the PINN paradigm rather than as a direct substitution of PDE residuals with an FMU.

2.1 Coupling Mechanism

The physical model is exported as an FMI 2.0 Co-Simulation FMU and instantiated in Python using FMPy. The FMU physical model and neural network adopt a series-coupling strategy, as shown in Figure 1. At each time step, the neural network outputs a voltage prediction Vt and a set of time-varying physical parameters. The predicted parameter sequence is written to the FMU parameter registers, while the current operating input is transferred through the FMU input interface. The Co-Simulation FMU then performs consecutive one-step simulations and returns the physics-based terminal voltage Ut to the loss-calculation module. The neural network is responsible for data-driven voltage prediction and dynamic parameter identification, whereas the FMU provides a modular physical consistency constraint. The FMU execution remains outside the neural-network automatic-differentiation graph; its parameter sensitivities are evaluated numerically as described below.

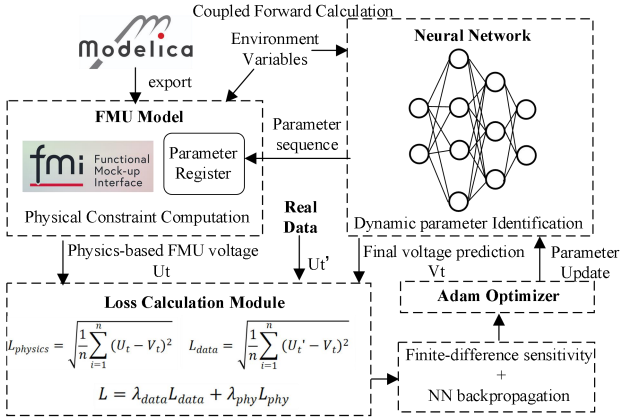


Figure 1. Coupling Framework

The training objective consists of a data loss between the neural-network voltage prediction Vt and the measured voltage Ut', and a physics loss between Vt and the physics-based FMU voltage Ut. For a consecutive mini-batch of length B, Vt is used as the final voltage prediction of PINN-Modelica, whereas Ut serves as a physics-based reference during training. The loss functions are defined as follows.

$$L_{data} = \sqrt{\frac{1}{B}\sum_{t=1}^{B}(V_t - U_t^{'})^2} \tag{1a}$$

$$L_{phy} = \sqrt{\frac{1}{B}\sum_{t=1}^{B}(V_t - U_t)^2} \tag{1b}$$

$$L=\lambda_{data}L_{data}+\lambda_{\mathrm{phy}}L_{phy} \tag{1c}$$

Because the Co-Simulation FMU is not part of the neural-network automatic-differentiation graph, the physical-parameter branch is differentiated numerically. Let $\mathbf{P} = [p_{t,j}]$ denote the B x 6 time-varying physical-parameter matrix. For each parameter at each time step, the complete physics loss over the current consecutive mini-batch is re-evaluated under positive and negative perturbations, and the central finite-difference sensitivity is computed as

$$\frac{\partial L_{phy}}{\partial p_{t,j}} = \frac{L_{\mathrm{phy}}(\boldsymbol{P} + h_j\boldsymbol{E}_{t,j}) - L_{\mathrm{phy}}(\boldsymbol{P} - h_j\boldsymbol{E}_{t,j})}{2h_j} \tag{1d}$$

$$\boldsymbol{G}_p = \frac{\partial L}{\partial \boldsymbol{P}} = \lambda_{\mathrm{phy}}\left[\frac{\partial L_{\mathrm{phy}}}{\partial p_{t,j}}\right]_{B\times 6} \tag{1e}$$

DOI 10.3384/ecp22357

$$\frac{\partial L}{\partial \boldsymbol{\theta}} = \sum_{t=1}^{B} \left[\frac{\partial L}{\partial V_t} \frac{\partial V_t}{\partial \boldsymbol{\theta}} + \sum_{j=1}^{6} \frac{\partial L}{\partial p_{t,j}} \frac{\partial p_{t,j}}{\partial \boldsymbol{\theta}} \right] \quad (1f)$$

where h_j is a small perturbation for the j-th physical parameter and $\boldsymbol{E}_{t,j}$ selects that parameter at time step t. Re-evaluating the complete mini-batch includes the effect of the perturbed parameter on subsequent FMU state evolution within the current temporal window. The resulting B x 6 matrix $\boldsymbol{G}_p$ is aligned with the six physical-parameter output channels. The voltage-output gradient is obtained by automatic differentiation; the two gradient paths are assembled and backpropagated through the shared neural network, followed by one Adam update per mini-batch. Equation (1f) explicitly combines the automatically differentiated voltage-output gradient and the finite-difference physical-parameter gradient to obtain the total gradient with respect to the trainable neural-network parameters $\boldsymbol{\theta}$.

2.2 Modeling Process

The entire modeling process adopts a phased design, where each phase is independently implemented and interacts with others, facilitating expansion and maintenance. The specific implementation process is divided into five phases, and the flow chart is shown in Figure 2.

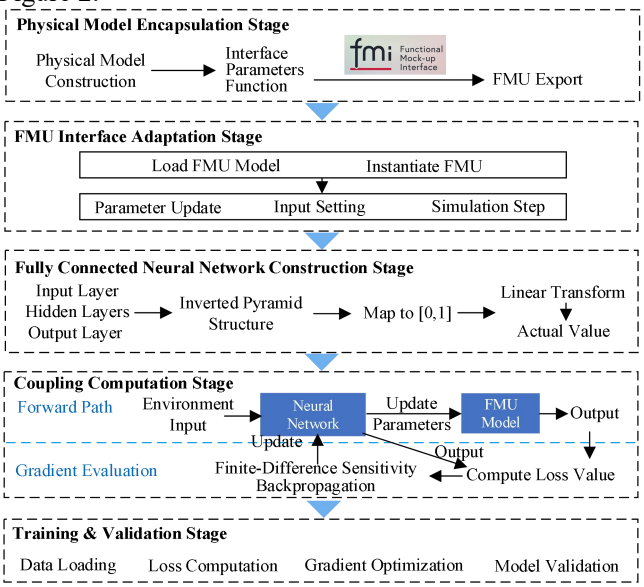


Figure 2. Modeling Process

Phase 1: The physical model encapsulation phase includes physical model construction and FMU export. Configure the input-output interfaces and parameter list of the model to generate a standardized FMU file. This module supports replacing physical models with different topological structures without modifying subsequent training framework code.

Phase 2: The FMU interface adaptation phase includes implementing loading, instantiation, and solving encapsulation of the FMU model. Provide unified interfaces for parameter update, input setting, and simulation stepping.

Phase 3: The fully connected neural network construction phase uses an inverted-pyramid structure. The output layer contains seven outputs: the terminal-voltage prediction Vt and six physical quantities R0, R1, C1, R2, C2, and OCV. Before being transferred to the FMU, the physical-parameter outputs are constrained to predefined physically feasible ranges.

Phase 4: The coupled forward-calculation and numerical-sensitivity phase performs consecutive simulation over each temporally ordered mini-batch. The neural network generates the voltage sequence and the time-varying physical-parameter sequence, and the Co-Simulation FMU returns the corresponding physics-based voltage sequence. Because the FMU is outside the automatic-differentiation graph, the sensitivity of the complete physics loss to each time-varying FMU input parameter is evaluated using central finite differences. Re-evaluating the complete current mini-batch for each perturbation allows the effect of FMU state evolution within the temporal window to be included in the numerical sensitivity.

Phase 5: The training and validation phase assembles two gradient paths. The gradient of the data and physics losses with respect to the neural-network voltage output is obtained by automatic differentiation, while the B x 6 physical-parameter gradient matrix is supplied by the finite-difference calculation. The combined output gradients are backpropagated through the shared neural network, and the Adam optimizer updates the network weights once per consecutive mini-batch. During validation, Vt is used as the final voltage prediction.

Through the above modular implementation, the modeling framework can be adapted to different physical systems. Such adaptation requires configuration of the FMU file, FMU variable mapping, neural-network input/output dimensions, and the physically feasible ranges or scaling of the identified parameters, while the overall coupling and gradient-calculation workflow remains unchanged.

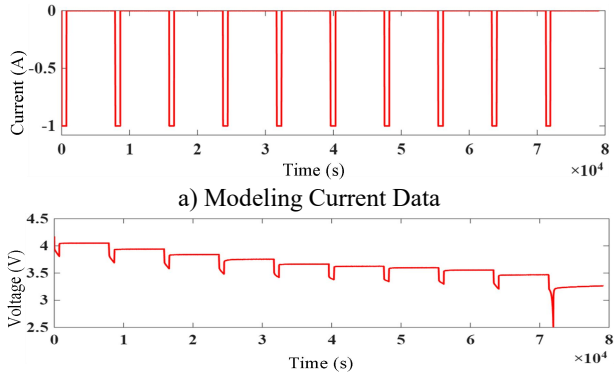
3 Battery Cell Example

This paper adopts a second-order Thevenin equivalent-circuit model as a proof-of-concept for the proposed framework. The model has a simple structure and is widely used in battery management systems. It includes an open-circuit voltage source Uoc with a nonlinear mapping relationship to State of Charge (SOC), an ohmic internal resistance R0, and two RC parallel loops representing electrochemical and concentration polarization effects, with parameters R1, C1, R2, and C2. The example is intentionally kept compact to demonstrate the coupling and parameter-identification mechanism; more complex nonlinear and multi-physics Modelica models are left for future work.

3.1 Data Source

The data in this paper comes from the commonly used public dataset of the Center For Advanced Life Cycle Engineering (CALCE) mentioned in the literature (Ji C et al. 2024). The INR 18650-20R cylindrical battery in the dataset is selected as the research object.

The modeling training data is the 25°C Incremental Current recording data, which records the process of ten constant current discharges of the battery cell, with relevant voltage and current as shown in Figure 3.

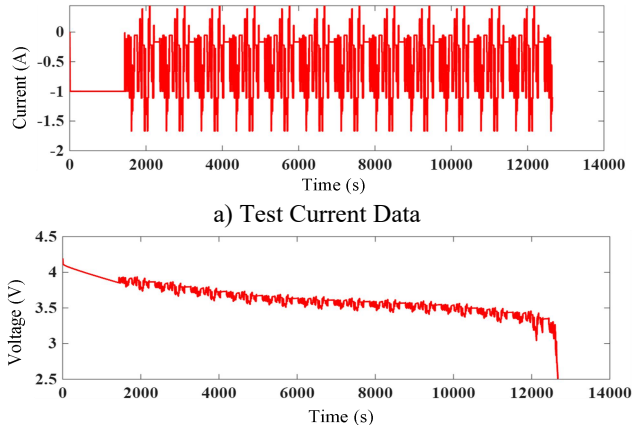


a) Modeling Current Data

b) Modeling Voltage Data

Figure 3. Training Data

The test data adopts recorded data under the 25°C Beijing Dynamic Stress Test (BJDST) operating condition, which completely records the process of the battery cell first discharging to 80% SOC and then performing multiple operating cycles. The relevant voltage and current conditions are shown in Figure 4.



a) Test Current Data

b) Test Voltage Data

Figure 4. Test Data

3.2 Modeling Framework

According to the proposed PINN data-driven method based on the Modelica model, the battery cell is modeled, and its framework diagram is shown in Figure 5.

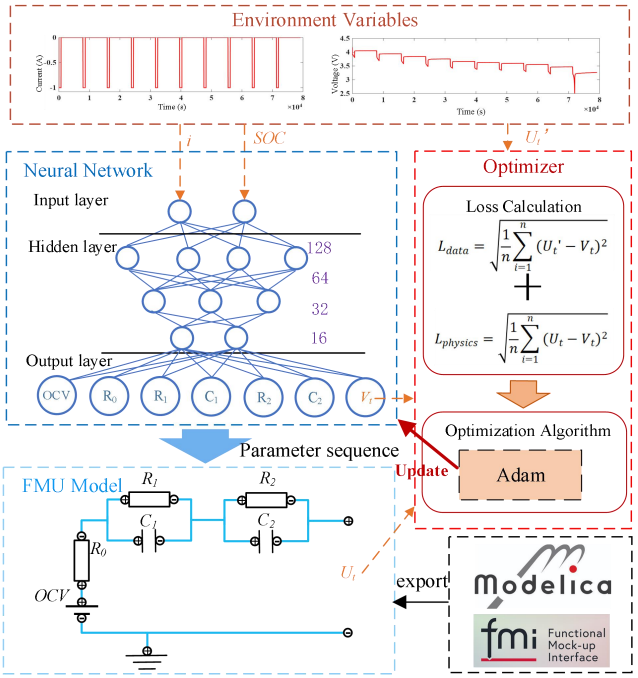


Figure 5. PINN-Modelica Modeling Framework

The PINN-Modelica framework embeds a Modelica/FMU model into neural-network training as a physics-consistency constraint while retaining the neural-network voltage Vt as the final prediction output. An inverted-pyramid fully connected neural network with hidden-layer sizes of 128, 64, 32, and 16 is used, with ReLU activation between hidden layers. The operating-condition data are normalized and arranged into temporally consecutive mini-batches. At each time step, current and SOC are fed to the network, which outputs Vt, OCV, and the equivalent-circuit parameters R0, R1, C1, R2, and C2. The parameter sequence is transferred to the Co-Simulation FMU, which performs consecutive one-step simulations and calculates the physics-based battery voltage Ut. The data loss compares Vt with the measured voltage Ut', whereas the physics loss compares Vt with Ut. The voltage-output gradient is obtained by automatic differentiation, while the physics-loss sensitivities with respect to the time-varying physical parameters are evaluated numerically by central finite differences over the current mini-batch. These gradients are then assembled and backpropagated through the neural network, and the Adam optimizer updates the network weights. The physical-parameter outputs are not directly supervised by parameter labels and should therefore be interpreted as physically constrained parameter estimates rather than unique ground-truth values.

When constructing the Modelica physical model, according to Kirchhoff's voltage law, the expression for the battery terminal voltage U is:

$$U_t = OCV - i * R_0 - U_1 - U_2 \quad (2)$$

DOI
10.3384/ecp22357

where i is the charge-discharge current, U_1 and U_2 are the terminal voltages of the two RC parallel loops, respectively, and their dynamic characteristics satisfy:

$$C_1\frac{dU_1}{dt}+\frac{U_1}{R_1}=i \tag{3}$$

$$C_2\frac{dU_2}{dt}+\frac{U_2}{R_2}=i \tag{4}$$

Benefiting from the acausal modeling feature of the Modelica language, the above equations do not need to consider the solution order and causal relationship, which greatly simplifies the complexity of model construction.

The training process of the PINN-Modelica model is as follows:

Step 1: Initialize neural network parameters, and set network structure and training hyper parameters including learning rate, batch size, number of iterations, etc.

Step 2: Normalize the operating-condition data and arrange them into temporally consecutive mini-batches. Input current and SOC into the neural network to obtain the voltage prediction Vt and the six time-varying physical parameters R0, R1, C1, R2, C2, and OCV.

Step 3: Sequentially transfer the predicted parameter sequence and current excitation to the Co-Simulation FMU while preserving the FMU state within the current mini-batch, and obtain the physics-based terminal-voltage sequence Ut.

Step 4: Calculate the data loss between Vt and the measured voltage Ut', and calculate the physics loss between Vt and the FMU voltage Ut. For each physical parameter at each time step in the current mini-batch, use central finite differences to numerically evaluate the sensitivity of the complete physics loss to the corresponding parameter, forming a B x 6 parameter-gradient matrix.

Step 5: Assemble the automatically differentiated voltage-output gradient and the numerically evaluated physical-parameter gradients, backpropagate the combined gradients through the neural network, and update the network weights once per mini-batch using the Adam optimizer. Repeat Steps 2-5 until the loss converges or the maximum number of iterations is reached.

Compared with the conventional residual-based PINN baseline, PINN-Modelica does not require explicit construction of governing-equation residuals for the proposed model; instead, physical consistency is introduced through the modular Modelica/FMU simulation. In addition, the physical-model components and topology can be changed according to the modeling requirements, while the coupling framework remains largely unchanged.

4 Result

To verify the effectiveness of PINN-Modelica, two comparative models are considered: a black-box neural network and a conventional residual-based PINN. The neural-network structures are configured to be comparable where possible. The black-box model outputs only terminal voltage and is trained using the voltage data error. The PINN baseline outputs six physical quantities and imposes explicit governing-equation residuals. Because the battery equations used in Equations (5)-(9) are ordinary differential equations rather than PDEs, the corresponding term is denoted as ODE Residual Loss L_{ODE} , together with Initial Condition Loss L_{IC} and Data Matching Loss L_{data}.

$$L_{ODE}=\frac{1}{N}\sum_{i=1}^{N}(Res_1{}^2+Res_2{}^2) \tag{5}$$

$$Res_1=\frac{dU_1}{dt}-(-\frac{U_1}{R_1*C_1}+\frac{i}{C_1}) \tag{6}$$

$$Res_2=\frac{dU_2}{dt}-(-\frac{U_2}{R_2*C_2}+\frac{i}{C_2}) \tag{7}$$

$$L_{IC}=(U_t(0)-U_t'(0))^2+(U_1(0)-0)^2+(U_2(0)-0)^2 \tag{8}$$

$$L_{data}=\sqrt{\frac{1}{n}\sum_{i=1}^{n}(U_t-U_t')^2} \tag{9}$$

Since the loss calculations of the black-box, PINN, and PINN-Modelica models are different, the data losses of each model are compared. The variation curve of the loss function during training is shown inFigure 6. In the figure, the model loss functions all gradually decrease with the increase of iterations and finally converge. Among them, the black-box model converges at approximately 62 iterations, and PINN and PINN-Modelica complete convergence at approximately 29 iterations. Compared with the black-box model, PINN and PINN-Modelica models have fewer convergence iterations and are easier to explore the optimal solution due to having physical prior knowledge. Their final convergence losses are basically the same and smaller than that of the black-box model.

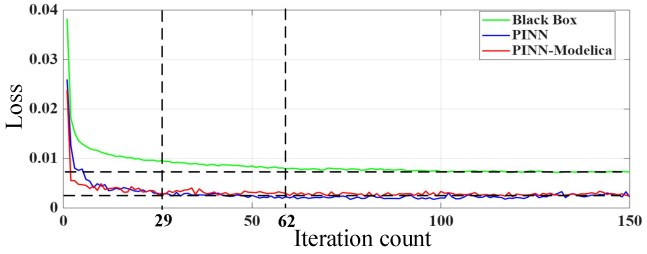


Figure 6. Data Loss Curve

Under the test operating condition, the three battery models are compared as shown in Figure 7. The black-box model relies only on the training data and therefore shows weaker performance when transferred to the test operating condition. The PINN and PINN-Modelica

predictions are closer to the measured voltage because physical prior information is incorporated during training rather than relying solely on an input-output fit. In the low-SOC interval, PINN-Modelica shows a smaller voltage-prediction deviation than the residual-based PINN for this proof-of-concept case.

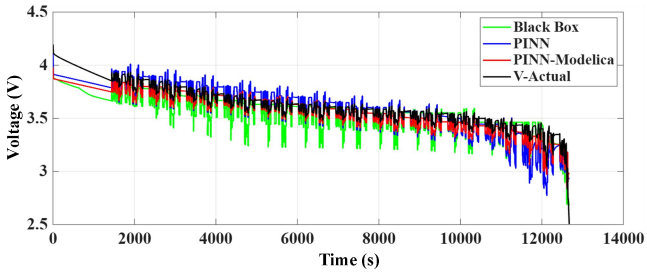


Figure 7. Voltage Prediction Comparison

Figure 8 shows the Absolute Percentage Error (APE) results between the predicted voltages of the three models and the actual voltage. During the entire test process, the closer APE is to 0, the better the prediction effect. It can be seen from the figure that the PINN and PINN-Modelica models perform very well before 11000 s. In the subsequent low SOC interval, the APE of the PINN model's predicted voltage is relatively large. Finally, when the SOC approaches 0, the PINN-Modelica model also fails to maintain good prediction performance, resulting in a relatively large APE.

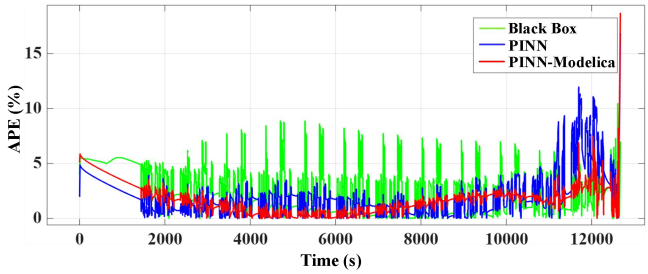


Figure 8. Absolute Percentage Error

The comprehensive performance of the three modeling methods is compared in Table 1. Under the reported test cycle, the MAPE and RMSE of PINN-Modelica are 0.2644 percentage points and 0.0146 V lower than those of the PINN baseline, respectively. These results indicate improved prediction accuracy in this cross-operating-condition proof-of-concept; broader generalization should be assessed on additional cells, operating conditions, and more complex physical models.

Table 1. Methods Comparison Table

Methods	*Black Box*	*PINN*	*PINN-Modelica*
MAPE (%)	2.6058	2.0058	1.7414
RMSE (V)	0.1220	0.0954	0.0808
Convergence Iteration	62+	29+	29+

In terms of interpretability, PINN-Modelica directly outputs trajectories associated with physically meaningful quantities such as ohmic resistance and polarization parameters. These outputs can support physical interpretation of the learned model and potentially assist battery-state analysis. Because no direct parameter labels are used in the present experiment, the estimated trajectories should be interpreted as physically constrained parameter estimates consistent with the voltage data and the Modelica model, rather than as uniquely identified ground-truth parameters.

5 Summary

This paper proposes PINN-Modelica, a physics-informed data-driven modeling framework that couples a neural network with a modular Modelica/FMU physical model. The neural network provides the final voltage prediction together with time-varying physical-parameter estimates, while the FMU supplies a physics-based voltage constraint. To connect the non-differentiable Co-Simulation FMU with gradient-based neural-network training, the physics-loss sensitivities with respect to the FMU input parameters are evaluated by central finite differences and combined with conventional backpropagation. The second-order Thevenin battery example serves as a proof-of-concept and shows improved prediction accuracy over the reported black-box and residual-based PINN baselines. The identified parameter trajectories retain physical meaning, although they are not directly validated against parameter ground truth in the present study. Future work will evaluate the framework on more complex nonlinear and multi-physics Modelica models, additional operating conditions, and broader battery datasets.

Acknowledgements

This work is supported by the National Natural Science Foundation of China (Grant No. U2441271) . The authors also acknowledge the support of the Fundamental Research Funds for the Central Universities.

References

Hofmann A, Mikelsons L. (2025) “Towards Integration of PeN-ODEs in a Modelica-based workflow”[C]//Modelica Conferences. 2025: 435-444.

Thummerer T, Kircher J, Mikelsons L. (2021) “NeuralFMU: towards structural integration of FMUs into neural networks”[J]. arXiv preprint arXiv:2109.04351, 2021.

Kamp T, Ultsch J, Brembeck J. (2023) “Closing the sim-to-real gap with physics-enhanced neural odes”[C]//20th International Conference on Informatics in Control, Automation and Robotics, ICINCO 2023. SCITEPRESS, 2023, 2: 77-84.

Velioglu M, Zhai S, Rupprecht S, et al. (2025) “Physics-informed neural networks for dynamic process operations with limited physical knowledge and data”[J]. Computers & Chemical Engineering, 2025, 192: 108899.

Chang S, Agarwal P, McCready C, et al. (2025) “Robust PINN modeling via sensitivity-based adaptive sampling: Integration of optimal sensor placement and structural uncertainty handling”[J]. Journal of Process Control, 2025, 152: 103493.

Laugksch K, Rousseau P, Laubscher R. (2023) “A PINN surrogate modeling methodology for steady-state integrated thermofluid systems modeling”[J]. Mathematical and Computational Applications, 2023, 28(2): 52.

Lawal Z K, Yassin H, Lai D T C, et al. (2025) “Modeling the complex spatio-temporal dynamics of ocean wave parameters: A hybrid PINN-LSTM approach for accurate wave forecasting”[J]. Measurement, 2025, 252: 117383.

Xiu F, Deng Z. (2024) “Performance of physical-informed neural network (PINN) for the key parameter inference in Langmuir turbulence parameterization scheme”[J]. Acta Oceanologica Sinica, 2024, 43(5): 121-132.

Liu T, Zhao J, Huang J, et al. (2024) “Research on model predictive control of autonomous underwater vehicle based on physics informed neural network modeling”[J]. Ocean Engineering, 2024, 304: 117844.

Hu H, Shen Z, Zhuang C. (2024) “A PINN-based friction-inclusive dynamics modeling method for industrial robots”[J]. IEEE Transactions on Industrial Electronics, 2024, 72(5): 5136-5144.

Brun J, Sergi T, Mutter S, et al. (2025) “From Simulation to Reality: Deployment of Reinforcement Learning-Based Neural Network Controllers Trained with Modelica Models”[C]//Modelica Conferences. 2025: 921-929.

Ji C, Dai J, Zhai C, et al. (2024) “A review on lithium-ion battery modeling from mechanism-based and data-driven perspectives”[J]. Processes, 2024, 12(9): 1871.

DOI
10.3384/ecp22357

From Intent to Action: Agentic AI on Validated Modelica Libraries

Johan Andreasson[1] Ajith Kumar[2]

[1]Modelon AB, Sweden, johan.andreasson@modelon.com
[2]Modelon Engineering Pvt. Ltd., India, ajith.kumar@modelon.com

Abstract

In this work, we compare a traditional design-of-experiments (DoE) workflow, defined up-front and then run, against an agentic AI workflow that revises its plan as the data arrives. The use case is a chassis optimization on the standard compact example in the Modelon VehicleDynamics Library. Both workflows produce a recommendation. The DoE arrives at a clean Pareto front and a stiffness change. The agentic path is longer: along the way it flags a controller saturation in the DoE winner, traces a center-of-gravity asymmetry to a load sensitivity the original scope did not cover, and finds a kinematic ceiling outside the four DoE parameters. It ends up changing a different parameter, and that parameter turns out to do more than the static KPIs asked for: dynamic verification shows the bushing change roughly halves the fishhook return lag unloaded. The architecture that makes this possible combines validated physics, Modelon Impact, the Model Context Protocol that connects them to an LLM, and a knowledge base layered by lifetime. We close with what we learned about when each approach is the right tool.

Keywords: Modelica, agentic AI, design of experiments, Model Context Protocol, vehicle dynamics, Modelon Impact

1 Introduction

The task is to maximize lateral performance of a car. We use the Modelon VehicleDynamics Library (Modelon AB 2026d) and its Compact example (Figure 1). In operational terms, "maximize lateral performance" means as much lateral acceleration as possible while the car stays stable at the limit (no limit oversteer, no spin-out) and rolls off past the peak smoothly rather than abruptly.

Two workflows can take the car from intent to a recommendation (Figure 2). A classical Design-of-Experiments approach defines the problem up-front, with KPIs, a parameter super-set, and bounds all chosen before any simulation runs. A sensitivity study narrows the parameter list, a multi-stage DoE hones in on a Pareto-optimal candidate, and the analyst inspects the result and writes the recommendation. Most of the work in the middle is automated and the human is in the loop only at the beginning and at the end. The analyst's conclusion drives whether to close the study, rerun or redefine the underlying algorithms.

An agentic approach starts from an intent rather than a specification. The engineer and agent write a specification from the intent before any work starts. A large language model (LLM) agent then works against a knowledge base, planning tool calls, running experiments, and inspecting the results that come back. Discoveries along the way revise the original scope. The human still approves intent and signs off the final design, but almost everything between those two points runs without a human in the loop. The agent also evaluates its own approach on exit and writes back to the knowledge base. Each study compounds into the next.

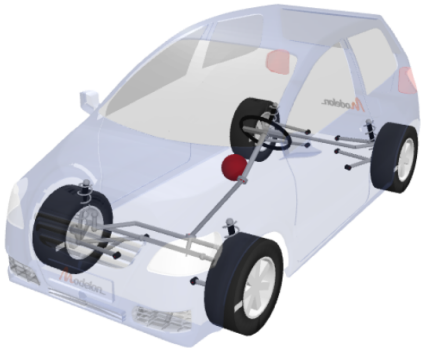


Figure 1. The Compact.ElastoKinematic chassis we operate on throughout this paper. Strut front, trailing-arm rear with two lateral guide links and elastokinematic bushings, Pacejka tires; $\approx 1100\,\mathrm{kg}$, 60/40 front/rear weight distribution.

Others are exploring agentic simulation: platforms that organize teams of AI engineers for computational design (Xu et al. 2025), frameworks for autonomous execution of simulation protocols (Soleymanibrojeni et al. 2025), and surveys of industry adoption and its obstacles (Edwards et al. 2026). What is different here is the foundation. The library has been validated against measurement for twenty years. The guardrails refuse a quantitative claim that lacks a provenance citation. Neither of those comes from the LLM.

2 Vehicle dynamics primer

A Modelica reader without an automotive background will encounter several terms below. We collect them here so the rest of the paper reads end-to-end without a chassis textbook.

Lateral acceleration (a_y) is the sideways acceleration

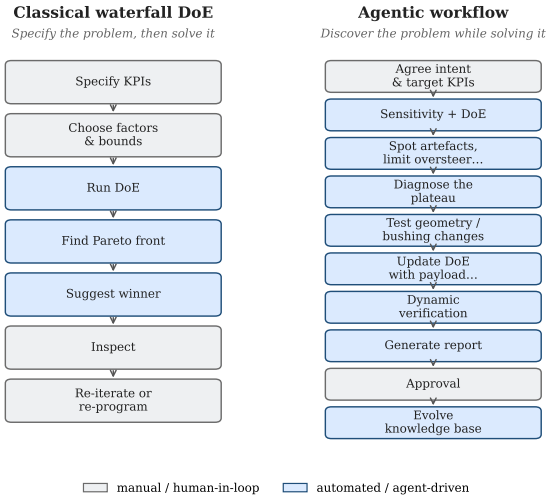


Figure 2. The two workflows compared in this paper, with manual steps in grey and automated steps in blue. The classical workflow specifies the problem up front, then automates the middle. The agentic workflow keeps the human at intent and approval, and automates everything else, including the knowledge-base update that carries into the next session.

a car sustains while cornering; the target in this study is 8.9 m/s^2 ($\approx$ 0.91 g).

Body roll is the angle (deg) the car body tilts outward in a turn.

Center of gravity (CoG) is the point the vehicle's mass acts through. Its longitudinal position sets how cornering load divides between the axles: moving it rearward loads the rear axle and makes the car more willing to rotate. Passengers and luggage move it rearward.

Camber is the lean angle of a wheel relative to the road. *Negative* camber (top tilted inward) keeps the contact patch flat under cornering load and grips well; *positive* camber (top tilted outward) loses grip.

Understeer and oversteer describe which axle saturates first at the limit. An understeering car needs progressively more steering angle as lateral acceleration builds, and runs wide; an oversteering car needs less, and the tail slides. Production cars are deliberately tuned to understeer, because it degrades predictably. The *understeer gradient* K_{us} ($^\circ$ m^{-1} s^2) quantifies the margin: the additional steering angle each further unit of lateral acceleration demands. Near zero is neutral steer, with no margin left.

Anti-roll bar (ARB) is a torsion spring connecting left and right wheels of one axle. Stiffer ARB on an axle reduces body roll but transfers more cornering load to the outer wheel of that axle, which then runs out of grip earlier; front/rear ARB *balance* sets under/oversteer at the limit.

Bushing is a rubber-and-steel element at every suspension pivot. It introduces a small deliberate compliance for impact absorption; when the design is wrong it lets the wheel angle change under load and hurts grip.

Roll-camber-compensation is the fraction of body roll the suspension geometry keeps *off* the wheel. 0% = the wheel tilts with the body; 100% = the wheel stays vertical. Typical road car: 40–60%. The car in this study: $\approx$ 30%.

Ramp-steer handling diagram is a standard maneuver: hold a constant speed, ramp the steering wheel slowly from straight to the limit, and plot a_y versus steering-wheel angle. The peak is the friction limit.

3 Classical DoE approach

The classical workflow starts by writing down the problem. We operationalize "maximize lateral performance" with two KPIs. `max_ay` is the lateral acceleration ceiling the maneuver reaches. `flatness` is the post-peak mean of a_y divided by the maximum, a proxy for how forgiving the roll-off is. The baseline sits at 8.49 and 0.991; any candidate that spins out is rejected, and above the baseline, more of either is better.

A one-at-a-time sensitivity sweep over thirteen parameters (springs, anti-roll bars, dampers, body mass, two center-of-gravity shifts, four hardpoints) picks out the four parameters that matter most: front and rear springs and front and rear anti-roll bars. Dampers have no measurable effect on the quasi-static peak.

A multi-stage DoE then narrows in. A wide Latin-hypercube sweep over the four parameters, $\pm$40% from baseline and 30 cases, sketches the feasible region. A zoom sweep of another 30 cases narrows the Pareto front, and an ARB-only reference sweep (5×5 full factorial) checks that constraining the search does not move the envelope. After filtering for sanity (no limit oversteer, no wheel lift), candidate B2_1 comes out on top with `max_ay` $= 9.04$ and `flatness` $= 0.996$, both above the reference targets shown in Figure 3. The change is a few percent stiffening of both front springs and the front ARB, and a small reduction on the rear ARB. The classical workflow stops there.

4 Agentic approach

The agentic workflow starts from the same baseline and runs the same initial sensitivity sweep, with DoEs as one of several tools on hand. The difference is that observations along the way revise the original scope. Three things came up on the path.

4.1 The reported max is not the real peak

Before recommending B2_1, the agent looked at the underlying trajectory rather than only at the headline KPI (Figure 4). The reported maximum sits at a steering-wheel angle of 387°, well past the point where the car has stopped responding to additional steering input. The agent flagged this as an irrelevant post-peak maximum and re-processed all 30 zoom candidates with a steering-angle cap at 180°, the envelope inside which the maneuver is physically meaningful. The pattern repeats across the DoE: B2_1's honest peak is 8.31 m/s^2, below the unmodified baseline.

The fix is generic. The agent added an envelope check as a mandatory step before any recommendation on a closed-

DOI
10.3384/ecp22365

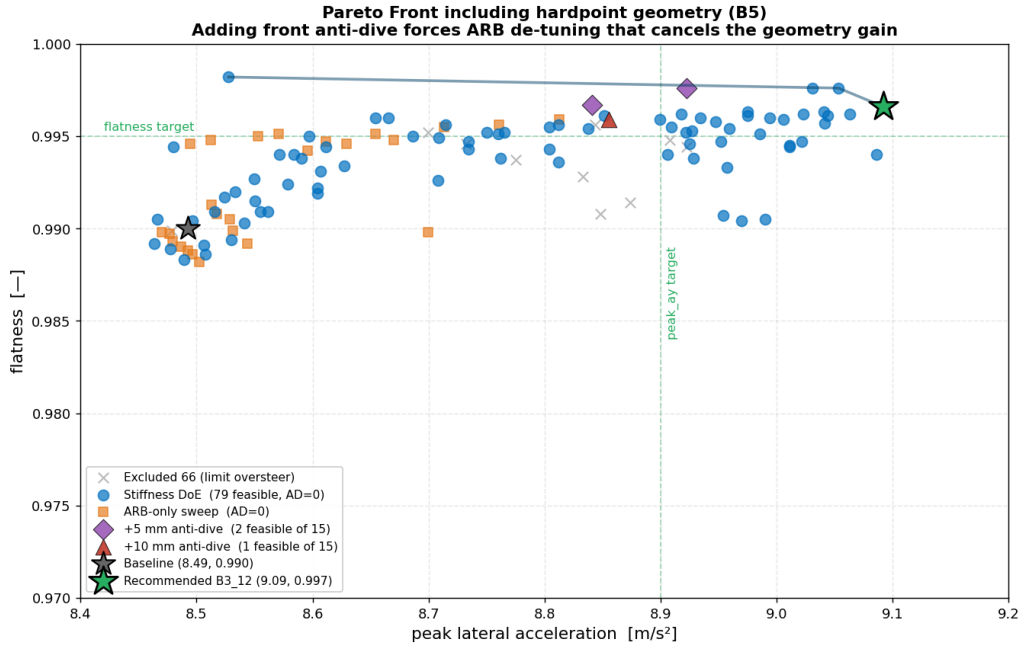


Figure 3. Pareto front from the classical DoE in `max_ay` versus `flatness` space, combining wide and zoom sweeps (79 feasible candidates, blue circles) with the ARB-only reference (orange squares). Baseline (black star) lies below both reference lines (dashed green); the headline candidate B2_1 (green star) lies past both.

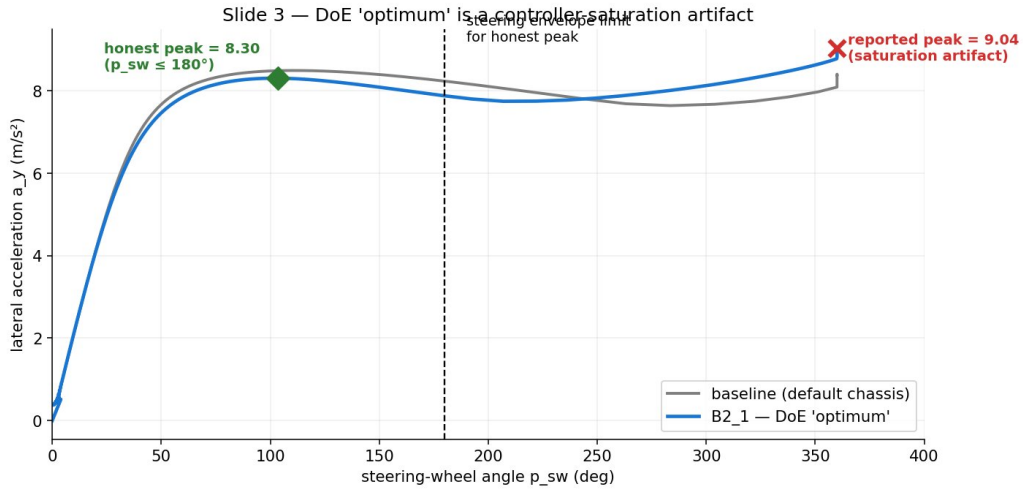


Figure 4. Trajectory comparison that flagged the classical DoE recommendation. Baseline (grey) reaches its physical peak inside the honest steering window $|p_{\mathrm{sw}}| \leq 180°$. Candidate B2_1 (blue) rolls off near the window, then climbs again as the driver model continues to increase steering demand past the grip limit, reaching a numerical peak at 360°. The honest peak of B2_1 (green diamond, $\approx 100°$) lies below the baseline.

loop driver maneuver, and wrote the rule back into the application playbook so the next study inherits it.

4.2 Rearward CoG sensitivity points to loading

In the sensitivity sweep, the agent perturbed the center of gravity forward only and reported that a forward shift improved `max_ay`. The one-sided sweep was an oversight, and the engineer asked for the opposite direction; the agent then re-ran bidirectional. Backward shifts induced limit oversteer with tail-out behavior, so the car is sensitive to a rearward shift in CoG. This is worth stating plainly, because it marks the boundary of what the agent did unaided: the corrective question was human, and everything after it was not.

A classical conclusion would stop at "do not move the CoG backward". The agent took the result further. If backward CoG shifts make the car spin, the car is sensitive to anything that moves mass rearward, and passenger and luggage loading does exactly that. The agent proposed adding loaded-car cases using the library's built-in payload parameters, with a load representative of a typical compact-car family trip: driver, front passenger, one rear passenger,

and 30 kg of luggage, 255 kg total. Under this loading the chassis loses most of its understeer reserve: the steering required to reach 0.3 g drops from 20.7° (unloaded) to 11.3° (loaded), corresponding to an understeer gradient that falls from $K_{us} = 0.49$ to $0.09\,^\circ\,\mathrm{m}^{-1}\,\mathrm{s}^2$. That is essentially neutral steer; the VehicleDynamics Compact example has not been OEM-tuned to preserve an understeer margin under load.

The inference from "do not move the CoG backward" to "then loading matters, and the library already exposes payload parameters" was the agent's, as was the choice of a representative 255 kg load case.

4.3 A kinematic ceiling on rear camber

Why is the car so sensitive to load? Heavier loading puts more weight on the rear axle, more lateral force at the outer rear wheel, and more reliance on that wheel's contact patch. Contact-patch grip depends on the camber angle relative to the road. The agent pulled body roll and outer-rear camber inside the honest steering window across all 30 zoom-DoE cases. Body roll varies from 4.8 to 6.0°, outer-rear camber from 3.2 to 4.0°, but the camber-to-roll ratio sits at $33 \pm 3\%$ regardless of stiffness. Roll-camber compensation is set by suspension geometry and compliance, which sit outside the original four-parameter scope.

If the kinematic ratio is fixed, the lever has to be elastokinematic. For a trailing-arm geometry the dominant compliance path is the trailing-arm pivot bushing, exposed in the library as a 6×6 compliance matrix. A four-point bracket on the diagonal locates a realistic operating point at 5×10^7 N/m translational and 5×10^3 N·m/rad rotational. Those numbers are in the range of a premium-segment production bushing, not a rigid mount. With springs and ARBs unchanged, `max_ay` rises from 8.49 to 8.95 m/s^2 and body roll drops from 5.5° to 4.2°. Mechanically, the stiffer pivot adds rear-axle roll stiffness that springs and ARBs alone could not deliver, and the unchanged $\approx 33\%$ kinematic ratio acting on a smaller body roll keeps the outer wheel inside its grip envelope longer.

4.4 Verification across maneuvers and loadings

The KPI targets were defined on the unloaded chassis and the recommendation clears them there. The loaded cases are a *stability check*, not a target check: we verify that the recommendation remains dynamically stable, does not introduce oversteer asymmetry, and maintains a forgiving roll-off when the car is carrying typical passengers and luggage.

The agent verified the recommendation across two maneuvers (NHTSA J-Turn and NHTSA Fishhook) at two loadings (unloaded and 255 kg typical compact-car load), with each maneuver calibrated per case to a steering amplitude of $6.5 \times \delta_{cal}$, where δ_{cal} is the steering-wheel angle that produces 0.3 g of lateral acceleration in a slowly increasing steering test on that particular configuration. The factor-of-two difference between the unloaded and loaded amplitudes in Table 1 reflects the chassis's loss of understeer reserve under load discovered in Section 4.2. Dynamic response metrics follow standard control engineering conventions: peak yaw rate, yaw lag (time from step onset to peak yaw rate for the J-Turn, and from steering reversal to the opposite-sign yaw peak for the fishhook), peak roll, and peak a_y. Figure 5 shows the underlying yaw-rate traces.

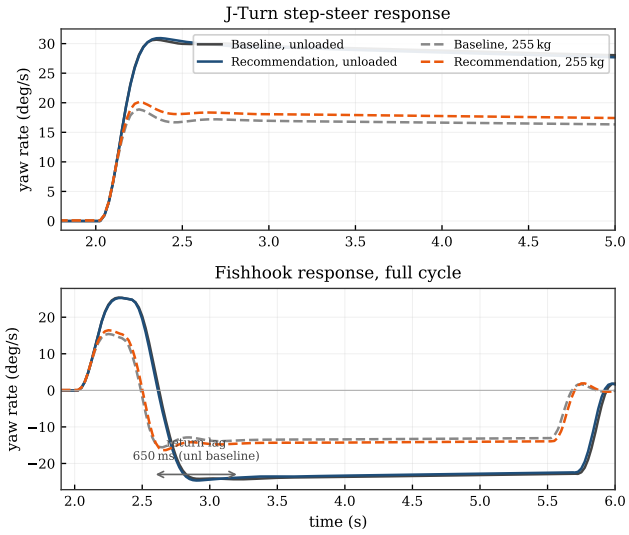


Figure 5. Yaw-rate traces underlying Table 1. Top: J-Turn step response, all four configurations rise to peak in ~ 0.2 s. Bottom: fishhook full cycle; the unloaded baseline (grey, solid) takes 650 ms to recover from the steering reversal, the recommendation (blue, solid) takes 350 ms because the stiffer pivot bushing makes the chassis less roll-coupled and the second-half yaw response cleaner.

The recommendation lowers peak roll in every combination of maneuver and loading. J-Turn yaw lag and peak yaw rate are essentially unchanged between baseline and recommendation, so the step response is not slowed by the bushing change. In the unloaded fishhook the recommendation cuts the return lag from 650 ms to 350 ms: the lower-roll chassis recovers from the steering reversal almost twice as fast. All four configurations settle to zero yaw at the tail in both maneuvers, with no spin-outs and no oversteer asymmetry, so the recommendation passes the stability check at both loadings and, in the unloaded fishhook, delivers a transient improvement that the static KPIs did not capture.

5 Results

Table 2 consolidates the outcomes from the ramp-steer (Figure 6). The recommendation clears both static KPIs unloaded, and the dynamic verification (Table 1, Figure 5) confirms that it is also faster to recover from a steering reversal at the limit while remaining stable across loadings.

The recommendation changes one parameter relative to baseline: the rear trailing-arm pivot bushing C diagonal at $\approx 5\times$ the default (5×10^7 N/m translational, 5×10^3 N·m/rad rotational).

DOI
10.3384/ecp22365

Table 1. Dynamic verification across maneuvers and loadings. Each configuration uses its own per-case calibrated steering amplitude. All four configurations are dynamically stable (yaw tail ≈ 0, no oversteer asymmetry).

Maneuver / config	δ_{cal} (° SW)	peak yaw (°/s)	yaw lag (s)	peak roll (°)	peak a_y (m/s^2)
J-Turn at $6.5 \times \delta_{cal}$					
Baseline, unloaded	20.7	30.7	0.20	3.73	7.99
Recommend., unloaded	20.0	30.9	0.23	**2.88**	7.92
Baseline, loaded 255 kg	11.3	18.9	0.13	2.62	6.81
Recommend., loaded 255 kg	12.2	20.1	0.15	**2.22**	7.07
Fishhook (two-direction, yaw lag = return lag)					
Baseline, unloaded	20.7	25.4	0.65	3.54	—
Recommend., unloaded	20.0	25.3	**0.35**	**2.71**	—
Baseline, loaded 255 kg	11.3	15.6	0.23	3.30	—
Recommend., loaded 255 kg	12.2	16.4	0.20	**2.89**	—

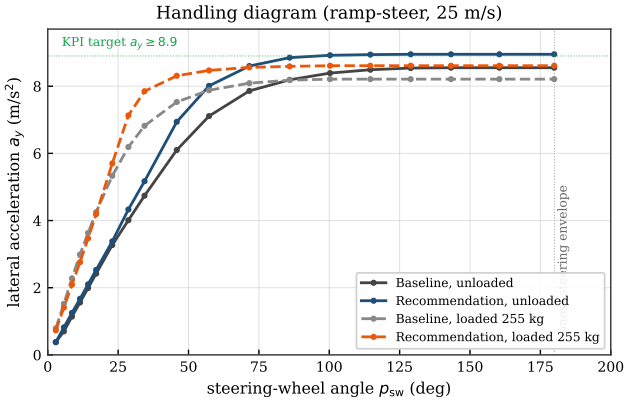


Figure 6. Ramp-steer handling diagram for the four configurations. Solid traces are unloaded, dashed are 255 kg loaded; baseline grey, recommendation in blue (unloaded) and orange (loaded). The recommendation lifts the curve at both loadings; unloaded it clears the 8.9 m/s^2 reference line (dotted green). The loaded car reaches the same lateral acceleration at half the steering-wheel angle of the unloaded car, which reflects its reduced understeer reserve at this loading.

Table 2. Ramp-steer outcomes. "Honest" = max(a_y) inside the steering envelope $|p_{sw}| \leq 180°$. KPI targets apply to the unloaded chassis; loaded cases are a stability check that the design remains well-behaved when carrying passengers and luggage.

Configuration	*Honest* a_y (m/s^2)	*Flatness*	δ_{cal} (° SW)
Unloaded: target check (*max_ay* ≥ 8.9, *flatness* ≥ 0.995)			
Baseline	8.49	0.991	20.7
Recommendation	**8.95 ✓**	**0.997 ✓**	20.0
Loaded 255 kg: stability check			
Baseline	8.21	0.994	11.3
Recommendation	**8.61**	**0.996**	12.2

5.1 Reproducibility

All data in this paper is from simulation. The study ran on Modelon Impact v4.129.0 with VehicleDynamics Library v4.9; the same versions reproduce the same outputs. The agentic workflow has been tested on multiple language models; results in this paper used Sonnet 4.6 by Anthropic.

6 Architecture

The previous sections showed what the two approaches produced. Three elements made the agentic result possible: a layered knowledge base, the Modelon Impact platform with its validated libraries, and the Model Context Protocol that connects them to an LLM. The output is conclusions backed by validated physics and numerics.

6.1 Validated physics

In equation-based modeling the library is the contract (Li and Andreasson 2026). A validated component carries the physics the agent needs. Nothing is derived in-session. The Modelon VehicleDynamics Library has been thoroughly validated over the twenty years since its first release, including limit-handling validation against vehicle measurements (Andreasson and Jonasson 2008), kinematics-and-compliance test-rig validation (Sundström et al. 2016), and real-time deployment in OEM driving simulators (Andreasson, Machida, et al. 2016). The library introduced a hierarchical template paradigm (Andreasson and Gäfvert 2005), exposing a hierarchy of interfaces, templates, and examples that the agent navigates directly. Components are replaceable and composable in well-defined ways, so the agent can swap a tire model, redeclare a chassis, or extend an experiment template without touching equations. Every approved result is pinned to a versioned model and a versioned library, so the recommendation can be reproduced months later from those identifiers alone.

6.2 Compute engine

The engine is Modelon Impact (Modelon AB 2026a), a cloud-based simulation platform with an on-premises option for customers who need it. Modelon Impact handles Modelica compilation, simulation, experiment definitions, and result storage, and exposes all of that through a typed HTTP API. Results, FMUs, and experiment definitions are stored against versioned IDs, so the engineer (and the agent) can refer back to a specific run without re-running it. Validated libraries plus a programmable engine produce conclusions backed by physics and numerics, not generated text.

6.3 Protocol: MCP

The agent reaches the engine through a Model Context Protocol (MCP) server (Anthropic 2025; Hou et al. 2025), the open JSON-RPC standard introduced by Anthropic in late 2024 for exposing typed tools and structured resources to an LLM. The server wraps the Modelon Impact API as about forty typed tools, one per logical operation, including a workspace-bootstrap call that the agent uses to set up a new project before any other work begins. The tools group by purpose: orientation, library access, model read and write, compile and run, results, DoE, diagram, and quality. Model changes, including reconfiguration, new model construction, and debugging, also happen through the MCP server. The agent builds and edits Modelica models. Every change is a typed tool call. Every physical claim traces back to a validated component (Andreasson 2026b).

6.4 Layered knowledge

The agent writes what it learns to a layered markdown library, organized by lifetime. Good findings move up; session noise stays behind.

- **Schemas.** Machine-readable YAML definitions the agent validates at load time: the required model components (bill of materials), a classification of simulation failure modes (e.g. solver divergence, physical-limit violation, missing output), and the KPI output declaration. YAML rather than Markdown because these are checked programmatically, not read as prose.
- **Methods (M1–M5).** Procedures that apply regardless of physics domain: systems design, simulation, numerics, languages, and platform operations. The sensitivity, DoE, sweep-analysis and Pareto-extraction procedures this study drew on sit inside M2.
- **Applications (A1–A4).** Domain playbooks: A1 VehicleDynamics, A2 data centers, A3 HVAC, A4 thermal management. Only the relevant one loads for a given workspace.
- **Physics references (P1–P4).** First-principles material (mechanics, electrical, hydraulics, thermofluid) the agent can cite when it needs to explain something rather than just compute it.
- **Workspace knowledge.** Findings, open hypotheses and model and simulation status for the project at hand.
- **User preferences.** Per-engineer choices the agent has learned to apply by default.

The agent loads context dynamically. For the case study in this paper it pulled methods on how to find optimal parameter sets (M-series), the application playbook for lateral-performance evaluation (A1), and naming-convention information from the VehicleDynamics Library context. It also queried specific models for their parameter and variable interfaces as needed.

The agent reads from the knowledge base and writes back on exit. A finding that survives verification lands at the right level: an application-agnostic principle in the M-series, a domain-specific consequence in the application playbook. That is the final box in Figure 2: the loop closes by writing back. The Section 4.1 finding above generated a proposal at each level. The general principle lands in `M2_simulation.md`:

```
id:        M2-performance-window
scope:     performance evaluation curves
maturity:  verified (exp/<id>)
rule:      performance curves may have
           multiple maxima; KPI formulation
           must target the physical ceiling,
           not a model or actuator limit
because:   a secondary maximum arises when
           the driving input or an actuator
           reaches its limit before the
           physical phenomenon of interest
```

The vehicle dynamics consequence lands in `A1_vehicle_dynamics.md` as a specialization of the rule above:

```
id:        A1-grip-ceiling
scope:     VDL limit-performance maneuvers
maturity:  verified (exp/<id>)
rule:      the performance ceiling is
           tyre saturation; extract KPIs
           at first saturation of the
           controlling input, which
           corresponds to the grip limit
because:   post-saturation output reflects
           model or actuator limits,
           not chassis grip
```

The structure of the knowledge base is shown in the tree below; branches end in `...` where the inventory is not exhaustive:

```
/
|-- Navigator.md
|-- methods/
|   |-- M2_simulation.md
|   |-- M3_numerics.md
|   `-- ...
|-- applications/
```


DOI
10.3384/ecp22365

```
|   |-- A1_vehicle_dynamics.md
|   |-- A2_cooling.md
|   `-- ...
|-- physics/
|   |-- P1_mechanics.md
|   `-- ...
`-- schemas/
    |-- kpi_output_spec.yaml
    |-- failure_taxonomy.yaml
    `-- ...
```

6.5 Guardrails

The architecture enforces three generic checks regardless of domain. First, *approval gates*: structured artifacts that the agent must produce and the engineer must sign off before the next phase begins. In this study, the engineer approved the intent record before the sensitivity sweep started, and approved the revised scope — adding the bushing parameter and loading cases — before dynamic verification ran. Second, *verification tagging*: every knowledge entry carries a tag (observed, confirmed, verified) so the agent knows what it has actually checked versus what it has assumed. Third, *provenance enforcement*: a validator that refuses any quantitative claim in a report that lacks a citation to the experiment that produced it.

The gates and validators are MCP tools that return refusal codes when their conditions are not met. This means the agent can be creative when planning and suggesting what to do, but must stay firmly within defined limits during execution.

6.6 The language model component

A large language model (LLM) is one component of the architecture, replaceable as models improve. It does three concrete things in the workflow. It plans, choosing which tool call comes next based on the engineer's intent and the previous result. It interprets, recognizing patterns in the data it receives back from the platform (a peak in the wrong place, an asymmetric trend, a parameter that does not move). And it writes, producing the report at the end of the study from the trace of tool calls and results. The model determines how reliably each of those three is done. The architecture determines whether it has anything trustworthy to work with.

7 Discussion

7.1 Credit to the classical DoE

The classical DoE is a mature method. A great deal of know-how is baked into how engineers pick KPIs, set bounds, and sanity-check candidates. The heuristics in Section 3, such as constraining to a sensitivity-identified parameter list, excluding limit-oversteer cases, and looking at both flatness and peak, are the product of decades of practice. For mature workflows where determinism is high and the room for unexpected discovery is low, the classical DoE remains the fastest path to a defensible result.

The agentic workflow adds something different: a level of discovery that the up-front specification cannot reach by construction. On this task, the agent revised the parameter list (by introducing a bushing), the KPI (by introducing the steering-envelope cap), and the scope (by adding loaded-car cases and per-case calibrated dynamic verification). Each of those revisions came out of an observation that did not match the original plan.

7.2 Exploring vs. short leashing

The choice between the two is not all-or-nothing. With an agent in the loop, the engineer decides where the leash is short and where it is long. For a high-cadence study with well-understood physics and a small parameter list, open exploration adds overhead rather than value; a tight scope and a classical DoE will produce a defensible result faster. When the answer might be outside the parameter list, when KPI definitions are still being negotiated, or when the model itself is partly unfamiliar, letting the agent revise scope earns its overhead back.

A useful intermediate is to have the agent orchestrate a classical DoE. The artifact gates lock down KPIs, parameter list, and bounds up-front, the agent runs the sweep, screens the results, and writes the report without revising scope along the way. You get the speed and consistency of a scripted pipeline together with the provenance, knowledge capture, and figure generation the architecture already provides.

7.3 What the agent reasons about

The agent forms hypotheses from observations and tests them with simulation. On this study: backward CoG shifts cause spin, so the car may be sensitive to loading. A chassis-tuning observation surfaced on the way: the VehicleDynamics Compact example loses most of its understeer reserve under moderate load (K_{us} falls from 0.49 to $0.09\,^{\circ}\,\mathrm{m}^{-1}\,\mathrm{s}^{2}$), which is not how a production vehicle would be tuned and which an up-front DoE on the unloaded chassis would not have flagged. The agent does not replace simulation. Hypotheses are tested against validated physics. Conclusions carry weight because they trace to a versioned model and a versioned library.

7.4 What carried the study, and what the study added back

The flywheel cuts both ways. The agent reaches a defensible result quickly because it inherits the library context, the vehicle dynamics playbook, and the simulation guardrails. The multi-run mechanics underneath (range and choices operators, engine-side expansion, parallel execution, Sobol and Latin-hypercube sampling) are existing Modelon Impact capabilities (Modelon AB 2026b; Modelon AB 2026c) that the agent layer operationalizes rather than introduces. Three entries were promoted back into the simulation guardrails on exit: a closed-loop signal-extraction rule, an edge-condition validation rule, and a sweep-plateau pattern, each tagged with an evidence level and a scope. The rules will be familiar to vehicle dynam-

icists; the contribution is the mechanism for capturing, tagging, and inheriting them. The classical DoE is still the foundation. What the agent adds is reasoning. Scope adjusts as data arrives. The rules that emerge are captured and written back so the next study inherits them.

7.5 Engineers stay in the loop

Results still need human sign-off. The architecture has two mandatory human gates, at intent and at scope, and a third at design freeze, with optional gates in between that the engineer can insert wherever a decision feels heavy. The agent's reach is bounded by what the tools expose, and the recommendations it makes are checked by the engineer who agreed to the intent in the first place.

7.6 Limitations

LLM behavior is not fully predictable, and will not become so. The exact path a study takes changes from one run to the next, and changes again as the model evolves. A newer model version may be generally more capable, but this does not guarantee better reasoning for a specific engineering domain. The know-how a model brings to vehicle dynamics may improve, stay flat, or regress between model generations (Andreasson 2026a).

This is not an argument against the approach. It is an argument for three safeguards this architecture already relies on. Guardrails keep the agent free to plan and suggest, but hold it firmly to defined limits during execution. An explicit knowledge base makes engineering know-how independent of any one model version. Full simulation reproducibility means every result can be re-run from identifiers alone, without the original chat session.

8 Conclusions and outlook

Validated physics, a compute engine, and agentic AI together produce recommendations that an up-front specification cannot: ones that revise scope as the data arrives and trace every claim to a versioned model and experiment. We did not measure analyst effort, so we make no claim about speed relative to a human analyst. The combination works best where scripting either does not reach or creates more maintenance than it saves. The architecture uses formats engineers already work in: Modelica, Python, JSON, Markdown. That is what makes the result of an agentic session industry-grade rather than demonstration-grade.

The same architecture is being applied to thermal management, powertrain, HVAC, and data-center cooling libraries inside Modelon Impact. The tool surface and knowledge-base format transfer directly; only the library content changes.

References

Andreasson, Johan (2026a). *5 Questions to Ask Before Trusting AI-Related Simulation Results*. Modelon blog. URL: https://modelon.com/blog/5-questions-to-ask-before-trusting-ai-related-simulation-results/.

Andreasson, Johan (2026b). *From Intent to Action: Agentic AI for Vehicle Dynamics in Modelon Impact*. Modelon blog. URL: https://modelon.com/blog/from-intent-to-action-agentic-ai-for-vehicle-dynamics-in-modelon-impact/.

Andreasson, Johan and Magnus Gäfvert (2005). "The VehicleDynamics Library — Overview and Applications". In: *Proceedings of the 5th International Modelica Conference*. Session 1b3. Vienna, Austria. URL: https://modelica.org/events/modelica2006/Proceedings/sessions/Session1b3.pdf.

Andreasson, Johan and Mats Jonasson (2008). "Vehicle Model for Limit Handling: Implementation and Validation". In: *Proceedings of the 6th International Modelica Conference*. Session 3c2. Bielefeld, Germany. URL: https://modelica.org/events/conference2008/sessions/session3c2.pdf.

Andreasson, Johan, Naoya Machida, Masashi Tsushima, John Griffin, and Peter Sundström (2016). "Deployment of high-fidelity vehicle models for accurate real-time simulation". In: *Proceedings of the 1st Japanese Modelica Conference*. Linköping Electronic Conference Proceedings 124. Tokyo, Japan: Linköping University Electronic Press, pp. 78–86. URL: https://ep.liu.se/en/conference-article.aspx?series=ecp&issue=124&Article_No=11.

Anthropic (2025). *Model Context Protocol Specification*. modelcontextprotocol.io. Revision 2025-11-25, accessed 2026-05. URL: https://modelcontextprotocol.io/specification/2025-11-25.

Edwards, Kristen M., Maxwell Bauer, Claire Jacquillat, A. John Hart, and Faez Ahmed (2026). *Agentic AI in Engineering and Manufacturing: Industry Perspectives on Utility, Adoption, Challenges, and Opportunities*. arXiv:2604.09633. URL: https://arxiv.org/abs/2604.09633.

Hou, Xinyi, Yanjie Zhao, Shenao Wang, and Haoyu Wang (2025). *Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions*. arXiv:2503.23278. URL: https://arxiv.org/abs/2503.23278.

Li, Lixiang and Johan Andreasson (2026). *From Intent to Insight: AI-Driven, Physics-Based Modeling with Modelica*. Modelon blog. URL: https://modelon.com/blog/from-intent-to-insight-ai-driven-physics-based-modeling-with-modelica/.

Modelon AB (2026a). *Modelon Impact: cloud-based system simulation platform*. modelon.com. Accessed 2026-05. URL: https://modelon.com/modelon-impact/.

Modelon AB (2026b). *Multi-Run Experiments*. Modelon Impact Help Center. Modelon Impact 2026.1. Accessed 2026-05-28. URL: https://help.modelon.com/latest/articles/how_to_run_multi_execution_experiments/.

Modelon AB (2026c). *Statistical Analyses*. Modelon Impact Help Center. Modelon Impact 2026.1. Accessed 2026-05-28. URL: https://help.modelon.com/latest/articles/how_to_statistical/.

Modelon AB (2026d). *Vehicle Dynamics Library Users Guide, v4.9*. Modelon Help Center. Accessed 2026-05. URL: https://help.modelon.com/latest/library_documentation/users_guide/vehicledynamics/.

Soleymanibrojeni, Mohammad, Roland Aydin, Diego Guedes-Sobrinho, Alexandre C. Dias, Maurício J. Piotrowski, Wolfgang Wenzel, and Celso Ricardo Caldeira Rêgo (2025). *GENIUS: An Agentic AI Framework for Autonomous Design and Execution of Simulation Protocols*. arXiv:2512.06404. URL: https://arxiv.org/abs/2512.06404.

Sundström, Peter et al. (2016). "Virtual Vehicle Kinematics and Compliance Test Rig". In: *Proceedings of the 1st Japanese Modelica Conference*. Linköping Electronic Conference Proceedings 124. Tokyo, Japan: Linköping University Electronic

DOI
10.3384/ecp22365

Press, pp. 29–34. URL: https://ep.liu.se/en/conference-article.aspx?series=ecp&issue=124&Article_No=4.

Xu, Ran, Yupeng Qi, Jingsen Feng, and Xu Chu (2025). *Engineering.ai: A Platform for Teams of AI Engineers in Computational Design*. arXiv:2511.00122. URL: https://arxiv.org/abs/2511.00122.

DOI
10.3384/ecp22365

Agentic Model-Based Design with FMI: A Two-Axis Interoperability Architecture

Weiwu Li, Haihua Feng

MathWorks, Natick, MA, USA

Abstract

Model-Based Design has proven effective for developing complex cyber-physical systems, yet its workflows remain largely manual. Additionally, industrial workflows often cross tool and organizational boundaries. This paper presents an agentic MBD framework for such boundary-crossing workflows. The architecture combines the Model Context Protocol (MCP) as the agent-to-tool interface, with FMI 3.0 as a model-to-model interface. We report an initial case study in which an agent builds a battery management system fault-protection model around a supplier-provided battery pack FMU. The case study demonstrates the feasibility of using MCP and FMI as complementary interoperability boundaries for agent-driven MBD workflows and highlights the interface and verification challenges that emerge at those boundaries. The architecture is tool-neutral by design; the present work exercises it through a MATLAB® and Simulink® reference implementation.

Keywords: Agentic AI, Model Context Protocol, FMI 3.0, Model-Based Design, battery management systems, large language models.

1 Introduction

Model-Based Design (MBD) has become the established methodology for developing complex cyber-physical systems across the automotive, aerospace, energy, and industrial automation domains. Despite its maturity, the day-to-day practice of MBD remains predominantly manual at the *workflow* level. Engineers move between plant modeling, controller design, integration, verification, and deployment as discrete, tool-specific tasks, transferring artifacts across phase boundaries and re-establishing context at each transition. This orchestration overhead — distinct from the substantive modeling work itself — is a recurring source of friction in industrial MBD practice and a recurring subject of internal tooling effort at organizations that practice MBD at scale.

The convergence of two recent developments now makes a different mode of practice feasible. Recent Large Language Models can carry out bounded, multi-step technical workflows, especially when equipped with domain-specific skills for using specialized engineering tools [anthropic-skills]. In parallel, the Model Context Protocol (MCP) [anthropic-mcp] has emerged as a standardized interface through which such agents can drive engineering tools programmatically, replacing brittle, tool-specific automation scripts. Together, these advances make it plausible that a single LLM agent could orchestrate the MBD lifecycle end-to-end, allowing engineers to express design intent in natural language, while the agent handles the procedural work of moving between tools and phases.

This vision has begun to emerge across the simulation and Model-Based Design communities, where AI agents are increasingly viewed as orchestration layers over engineering workflows. This raises a practical architecture question: how can an agent work across the MBD lifecycle when the models involved may not all come from the same toolchain or organization? In many industrial settings, especially OEM-supplier workflows, models may be developed in different environments, and supplier models may need to remain encapsulated to protect intellectual property. An agent that can drive one modeling tool but cannot integrate models from these external sources has automated only part of the workflow. Therefore, the model-to-model interface must be considered alongside the agent-to-tool interface.

We argue that the Functional Mock-up Interface (FMI), particularly in its 3.0 revision [fmi-spec], provides exactly this complement. While MCP standardizes how the agent invokes a tool, FMI standardizes how the models that tool produces compose with models from elsewhere. In this setting, FMI can serve as an import interface for supplier-provided IP, co-simulation during integration, and an export format for downstream deployment.

This paper's primary contribution is a two-axis interoperability architecture for agentic Model-Based Design. The architecture separates agent-to-tool interoperability, enabled through MCP, from model-to-model interoperability, enabled through FMI. We demonstrate the architecture through a battery management system case study in which an agent imports a supplier-provided FMU, constructs an integration harness, implements control logic, performs verification, and exports the resulting system as an FMU. We further discuss the implications of this separation for workflows that span tool, supplier, and deployment boundaries.

The remainder of the paper is organized as follows. Section 2 reviews the relevant background in Model-Based Design, agentic AI, MCP, and FMI. Section 3 develops the two-axis architecture. Section 4 presents the battery management system case study. Section 5 reports initial empirical observations. Section 6 discusses limitations and future work.

2 Background and Related Work

2.1 Model-Based Design and the Orchestration Gap

Model-Based Design is the systematic use of models

throughout the development process to improve how complex systems are delivered [mathworks-mbd]. Each phase produces artifacts that feed downstream phases, and the methodology's central premise is that defects detected against models at simulation time are substantially cheaper to address than defects detected against hardware later in the cycle.

In traditional Model-Based Design practice, much of the work of moving between lifecycle phases still falls to the engineer. An engineer who has parameterized a plant model in one tool must still manually configure it to a co-simulation environment, design the controller, integrate the system virtually, execute the analysis, and interpret the results — often across multiple tools, each with its own scripting interface, file formats, and conventions. We refer to this gap between the methodology's intent and its everyday practice as the *orchestration gap*.

2.2 LLM Agents and Agent Skills

A Large Language Model (LLM) becomes an *agent* when it is given the ability to take actions against external systems and to iterate on the results. Recent agent architectures combine the model's general reasoning capacity with structured tool use, episodic memory, and reflective error correction [yao-react; shinn-reflexion]. The trajectory of frontier-model capability over the past two years has progressed from agents that reliably handle single-step tool calls to agents that can plan and execute multi-hour technical workflows with minimal supervision [kwa-metr-2025].

A practical limitation of general-purpose LLM agents is that, without domain-specific knowledge, they reason from training-data priors that are often outdated, generic, or wrong about the conventions of specialized engineering tools. Agent skills [anthropic-skills] address this gap by packaging procedural and declarative domain knowledge as composable bundles that the agent loads progressively as needs arise. A skill typically contains a description that allows the agent to recognize when the skill is relevant, together with instructions, worked examples, and references to scripts that the agent invokes during execution. Skills shift the agent from generalist to domain-aware collaborator without requiring model fine-tuning.

2.3 The Model Context Protocol

The Model Context Protocol [anthropic-mcp] is an open standard, introduced in late 2024, that specifies how an LLM agent communicates with external systems. An MCP server exposes a set of *tools* (callable functions), *resources* (read-only context), and *prompts* (parameterized templates), each described by a machine-readable schema. The MCP client running inside the agent runtime discovers the server's capabilities at session initialization and presents them to the agent as part of its working context.

MCP's significance is that it replaces a previously fragmented ecosystem of agent-tool integrations with a single uniform interface. Tool authors expose their systems once through MCP and any MCP-aware agent can use them; agent authors implement the client once and gain access to the entire ecosystem of MCP servers. Adoption since the standard's release appears to be growing rapidly, with MCP servers now available for major integrated development environments, version control systems, and increasingly engineering tools.

2.4 FMI 3.0 and Related Standards

The Functional Mock-up Interface, first released in 2010, defines a tool-independent format for distributing simulation models as Functional Mock-up Units (FMUs) [blochwitz-fmi-2011]. Over fifteen years of evolution, FMI has become the de facto industrial standard for cross-tool model exchange and co-simulation, with adoption spanning the automotive, aerospace, energy, and process industries [junghanns-fmi3]. FMI 3.0, ratified in 2022, introduced substantial advances over its predecessors: explicit support for synchronous clocks, structured data types including arrays and binary types, terminals that group related variables, and improved hybrid co-simulation semantics. These features were designed in part to address the modeling and integration needs of virtual electronic control units and ML-augmented simulation workflows.

Some Model-Based Design tools are beginning to provide generative-AI assistance for model explanation, error diagnosis, design guidance, troubleshooting, and selected task automation [simulink-copilot; modelon-ai-assistant]. These capabilities are useful, but they are typically focused on task-level assistance within a specific modeling environment. This paper addresses a complementary problem: agentic orchestration across connected MBD workflows that span lifecycle phases, tools, and organizations. The proposed architecture separates agent-to-tool access from model-to-model interoperability: MCP provides the interface by which an agent invokes engineering tools, while FMI provides the neutral model boundary for exchanging and composing simulation components. This standards-based separation allows the architecture to generalize across the modeling-tool ecosystem more naturally than approaches tied to a single tool-specific API. In this sense, the contribution is not another AI assistant inside a modeling tool, but an interoperability architecture that enables agents to coordinate work across heterogeneous engineering workflows and encapsulated supplier models.

3 Agentic MBD with FMI

3.1 Two-Axis Interoperability Principle

Agentic Model-Based Design brings two interoperability concerns into the same workflow: how the agent interacts with engineering tools, and how models from different sources interact with one another.

The first concern is how the agent interacts with engineering tools. We refer to this as the *agent-to-tool axis*. An LLM agent may receive intent in natural language, but

DOI
10.3384/ecp22375

the actual modeling, simulation, and verification work is performed by engineering tools with their own scripting interfaces and operating conventions. A common agent-to-tool interface reduces the need for one-off tool adapters and gives the agent a more consistent way to invoke specialized capabilities.

The second concern is how models interact with one another. We refer to this as the *model-to-model axis*. In industrial workflows, system models often include components from different vendors, teams, or modeling environments: for example, a battery model from a supplier, a motor model from an external library, or a vehicle model from an internal platform. A common model interface allows these components to be integrated without requiring the agent, or the engineer, to translate each model boundary manually.

These two axes are related but distinct. Strong agent access to one engineering tool does not by itself solve the problem of integrating models from other tools or suppliers. Likewise, a well-structured multi-vendor model assembly still requires procedural orchestration to build, configure, simulate, test, and package the system. For agentic MBD to be useful across connected workflows, both concerns need to be addressed.

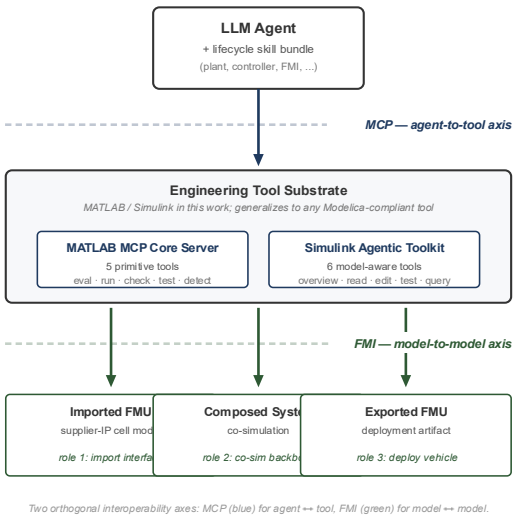


Figure 1. Two-axis interoperability architecture for agentic Model-Based Design. MCP provides the agent-to-tool interoperability layer through which the LLM agent invokes engineering tools, while FMI provides the model-to-model interoperability layer through which imported, composed, and exported FMUs interact across the lifecycle. The MATLAB® MCP Core Server and Simulink Agentic Toolkit are shown as the reference implementation used in this work.

In this paper, we use the Model Context Protocol (MCP) [anthropic-mcp] for the agent-to-tool axis and the Functional Mock-up Interface (FMI) [fmi-spec] for the model-to-model axis. The two standards operate at different layers: MCP is used when the agent asks an engineering tool to perform an action, while FMI is used when a tool composes or exchanges models. They do not replace one another; instead, they provide complementary interfaces for agent-driven workflows that span tools and model boundaries. Figure 1 illustrates this principle.

3.2 Skill Layer: Engineering Procedure

Agentic MBD requires more than access to engineering tools. While MCP provides a standardized mechanism for invoking tools, it does not supply the procedural engineering knowledge needed to carry out modeling, integration, verification, and deployment according to established engineering practice. Agent skills [anthropic-skills] address this gap by packaging domain-specific workflow knowledge that the agent applies across the MBD lifecycle, bridging general LLM reasoning and the specialized practices of Model-Based Design.

The MBD lifecycle decomposes naturally into phases — requirements capture, plant modeling, algorithm synthesis, model construction, verification, and deployment — and each phase admits corresponding procedural knowledge. The Simulink® Agentic Toolkit [satk] provides a growing collection of model-aware tools and skills that support agent-driven Model-Based Design workflows. These capabilities encode established engineering practices in a form that an LLM agent can invoke and apply, transforming the agent from a general-purpose assistant into a domain-aware collaborator.

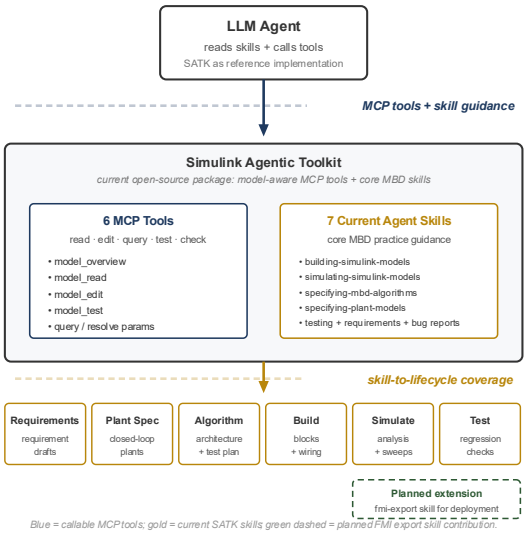


Figure 2. Simulink Agentic Toolkit capability map and lifecycle coverage. The current toolkit provides model-aware MCP tools and core MBD skills for requirements drafting, plant specification, algorithm specification, model construction, simulation, and testing. FMI export is shown as a planned extension to complete the deployment portion of the agentic MBD workflow.

The current Simulink skill catalog does not yet cover every step of the MBD lifecycle. It does not yet provide dedicated FMI-specific operations — for example, wrapping a completed design as an FMU for downstream deployment. Therefore, we developed a prototype FMI-oriented skill extension, referred to here as `fmi-operations`, and plan to contribute similar capabilities to future versions of the toolkit. The skill encodes guidance on FMI 2.0 versus 3.0

feature selection, interface curation for the exported FMU, and the export configuration required for deployment-grade FMUs. Figure 2 shows the current toolkit capabilities and the planned FMI export extension in the broader lifecycle context.

3.3 Tool Layer: Engineering Tool Access

In our reference implementation, the tool layer is built as a two-level MCP server stack. The MATLAB MCP Core Server [matlab-mcp] provides the basic connection to the MATLAB environment, including code validation, code evaluation, file execution, test execution, and toolbox detection. On top of this, the Simulink Agentic Toolkit adds model-aware MCP tools such as `model_overview`, `model_read`, `model_edit`, `model_test`, `model_query_params`, and `model_resolve_params`. These tools let the agent inspect model structure, query parameters, edit models, and run tests using Simulink-level concepts rather than only raw MATLAB commands.

The architecture, however, is not tied to MATLAB or Simulink. MCP is an open, tool-agnostic interface, so any modeling environment with a programmatic API could expose similar capabilities through an MCP server. Therefore, the MATLAB and Simulink components used here should be understood as one reference implementation, not as a requirement of the architecture. The same two-axis pattern can apply to other MBD toolchains, including Modelica-based environments such as OpenModelica or Dymola, provided that the agent can reach the engineering tool through MCP and the models can be exchanged or composed through FMI.

3.4 FMI Roles in the Agentic Workflow

Within the agentic MBD architecture, FMI plays several roles by supporting different model-boundary operations, for example:

Importing external models. Components arriving from outside the agent's modeling environment — whether from a supplier, an academic modeling library, or a partner organization — enter the workflow as FMUs. The agent treats the FMU as an opaque component characterized by its `modelDescription.xml`, exercising it through the standard FMI API without requiring access to source. This preserves supplier IP, accommodates heterogeneous modeling tools across the supply chain, and gives the agent a uniform abstraction to reason about regardless of the FMU's internal implementation.

Integrating through co-simulation. During system integration, the engineering tool composes the imported FMU with modeling-tool-native components and coordinates their execution through FMI co-simulation. FMI 3.0's improved co-simulation features — including support for clocks, terminals, and structured data types [fmi-spec] — are particularly relevant in this setting, as they reduce the interface mismatch between heterogeneous components. The resulting co-simulation provides the execution environment against which agent-generated test cases are evaluated.

Exporting for downstream use. Once verified, the integrated system is itself exported as an FMU and becomes the deployment artifact for downstream consumers. In our case study (Section 4), this exported FMU can be shared with consumers who lack access to the original modeling toolchain and consumed with open-source tooling such as FMPy [fmpy], or executed in real time on a hardware-in-the-loop bench. As an alternative deployment path, the same integrated model — including the supplier-provided battery plant FMU — can be packaged and deployed as a web application directly through MathWorks® deployment products, with the agent driving the packaging step through the Simulink Agentic Toolkit [satk] skill bundles.

The specific FMI usage is case dependent. The broader point is that FMI provides a stable model interface the agent can rely on as the workflow crosses tool and organizational boundaries.

4 Case Study: Agentic MBD for Battery Management System Fault Protection

This case study exercises the agentic MBD architecture through the development of a battery management system (BMS) fault-protection model. The model demonstrates how to charge and discharge an electric-vehicle (EV) battery efficiently and how to handle battery faults. Portable electronics and EVs widely use lithium-ion batteries as a power source because of their high energy density, but lithium-ion cells also carry safety risks under extreme conditions such as over-discharge, over-charge, and high or low temperature. These conditions can damage the battery and degrade its long-term performance, and in some cases can cause a loss of stability leading to thermal runaway. In this case study, a basic but representative control strategy operates the battery — placing it in charge or discharge mode and disconnecting it during fault conditions.

The case study begins from a battery protection specification — fault conditions, protection thresholds, a relay state machine, and a current-derating strategy. The agent is directed to build the BMS fault-protection algorithm from scratch, which lets the case study demonstrate genuine agent-driven design work. The battery pack itself enters the workflow as a Functional Mock-up Unit, treated as a supplier-provided, IP-protected component: the agent is given only the FMU and its interface, never its internal implementation. This is the realistic OEM situation, in which a supplier delivers an encapsulated model rather than source.

The prototype used OpenAI Codex as the agent runtime. Its tool interface was the MATLAB MCP Core Server [matlab-mcp], through whose code-evaluation capability the agent drove Simulink model construction by issuing MATLAB commands; in addition, the agent invoked several model-aware tools from the Simulink Agentic Toolkit [satk], such as `model_overview`, to read back and inspect the completed model structure during verification. This case

DOI
10.3384/ecp22375

study exercises both axes of the proposed architecture: MCP-based agent interaction with engineering tools and FMI-based interoperability across model boundaries.

In this workflow, the human engineer provided the architecture, requirements, acceptance criteria, and review. The agent performed the implementation work: FMU inspection, harness construction, controller design, verification, debugging, and FMU export. Manual intervention was limited to guidance and review. The following subsections describe the main steps of the agent-orchestrated workflow.

4.1 Step 1: FMU Import of a Supplier-Provided Battery Pack

The first step exercises FMI's role as an import interface. The battery pack model arrives as an FMI 3.0 co-simulation FMU. It exposes two inputs, the pack terminal current and a coolant mass-flow command, and outputs that report the pack's internal state, including state of charge, temperature, terminal voltage, and current. These outputs are module-resolved, so the downstream BMS must condition them into the pack-level signals used for protection logic.

Beyond this signal interface, the FMU exposes the pack's initial state of charge and initial cell temperature as tunable FMI parameters. Because these values are surfaced through the FMI parameter mechanism rather than fixed inside the exported unit, the starting state can be varied from one simulation run to the next. This gives the agent a practical way to steer the plant toward operating points needed for verification scenarios.

The agent's first task was to inspect the FMU interface, identify the available inputs, outputs, and tunable parameters, and determine which signals were needed by the BMS harness. Working from the FMU's `modelDescription.xml`, the agent treated the plant as an opaque component with a standard FMI interface.

4.2 Step 2: Agent-Driven Construction of the Simulation Harness

The second step constructs the model that surrounds the FMU plant. Before model construction, the human engineer supplied the agent with a high-level system architecture rather than a block-by-block implementation. This architecture identified the supplier battery FMU as the plant, the charger/load path as the electrical actuation path, the BMS as the supervisory controller, and the main feedback signals required for protection logic. Figure 3 summarizes this human-guided architecture.

Using this architecture, the agent built the Simulink harness from scratch. It connected the FMU current and coolant inputs to the surrounding actuation and thermal-management logic, and reduced the FMU's module-resolved outputs, including state of charge, temperature, voltage, and current, into the pack-level signals consumed by the BMS. The resulting harness gave the agent an executable closed-loop environment in which to implement and verify the controller.

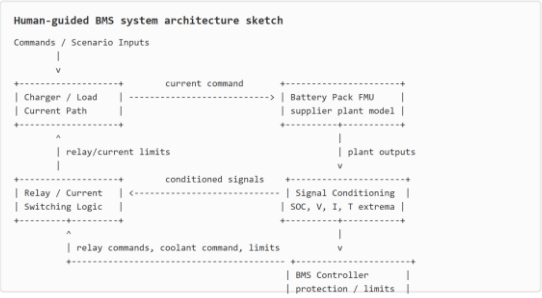


Figure 3. Human-guided BMS system architecture sketch. The human engineer provided the main system decomposition, including the supplier battery-pack FMU, charger/load path, relay and current-switching logic, signal conditioning, and BMS controller functions. The sketch captures the design intent given to the agent, which then translated it into the Simulink integration harness and implemented the subsystem details.

This step illustrates the intended human-agent division of labor. The human engineer provided the system decomposition and design intent, while the agent handled the detailed model construction, signal wiring, and integration work through MCP-exposed MATLAB and Simulink capabilities.

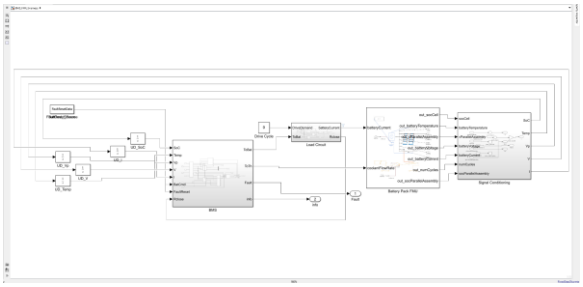


Figure 4. BMS integration harness with battery-pack FMU. The agent-generated Simulink harness combines the supplier-provided battery FMU with charger/load actuation, relay and current-switching logic, signal conditioning, and BMS controller subsystems. Plant outputs including state of charge, temperature, voltage, and current are conditioned and returned to the controller for closed-loop fault detection and protection verification.

4.3 Step 3: Agent-Driven Design of the BMS

With the integration harness in place, the third step implements the BMS design itself. Following the high-level architecture and protection specification, the agent structured the controller into cooperating subcomponents: input conditioning, current limiting, protection logic, state-of-charge handling, and thermal management. The protection logic monitors overvoltage, undervoltage, overcurrent on charge, overcurrent on discharge, and thermal fault conditions, and produces relay commands and fault codes for the surrounding harness.

The agent built the controller incrementally. It first created the input-conditioning logic that converts the FMU's module-

resolved outputs into the extrema and pack-level values needed by the BMS. It then implemented current limiting, fault detection, thermal-fault debounce behavior, supervisory mode control, and coolant command generation. State-of-charge estimation was kept behind a clean interface so that a more detailed estimator, such as an extended Kalman filter, could be added later without changing the rest of the BMS structure.

One useful debugging episode occurred after the loop was closed. The agent observed that a commanded discharge caused state of charge to move in the wrong direction, indicating a sign-convention mismatch at the FMI boundary. It diagnosed the mismatch, corrected the current sign convention on the command and measurement paths, and reran the verification scenario. A similar interface mismatch appeared between the controller coolant command and the FMU coolant mass-flow input, which the agent identified and corrected before completing verification. These episodes are important because they show both the value and the limitation of the approach: the agent recovered without direct manual repair, but only because the workflow included physically meaningful checks rather than relying only on structural model correctness.

4.4 Step 4: FMI Co-Simulation Integration and Automated Verification

With the FMU plant and BMS controller connected, the agent assembled the closed-loop co-simulation and created a repeatable verification suite. Each test scenario configured the simulation inputs, ran the model, and checked logged signals against expected behavior. The suite covered nominal charge and discharge behavior, overvoltage protection, thermal protection, current-limit clamping, and fault reset-and-recovery.

The overvoltage scenario is shown in Figure 5. During charging, the maximum cell voltage crosses the configured protection threshold. The BMS detects the fault, opens the relay, and stops charging. The voltage then relaxes below the protection limit. Similar scenario-based checks were used for the other protection functions.

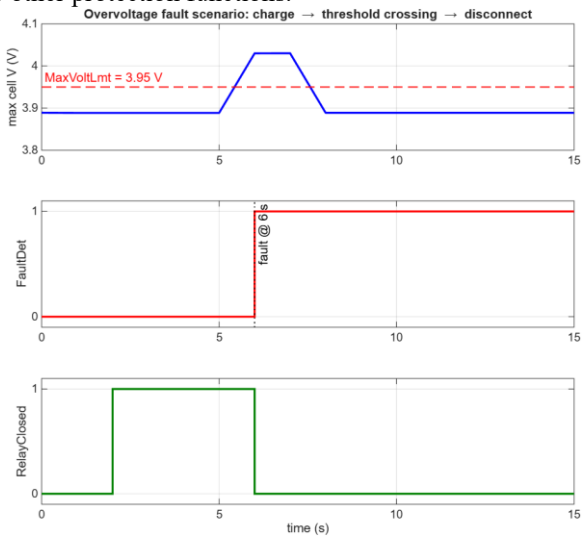


Figure 5. Overvoltage protection verification. During charging, the maximum cell voltage rises above the configured protection threshold (MaxVoltLmt = 3.95 V). The fault logic asserts at the threshold crossing (t = 6 s), causing the relay to open and charging to cease. The cell voltage subsequently relaxes below the protection limit.

4.5 Step 5: FMU Export and Deployment

The final step exercises FMI as an exchange and deployment format. After the closed-loop system passed the verification scenarios, the agent exported the integrated model as a single FMI 3.0 co-simulation FMU. The exported FMU bundled the supplier battery FMU inside the larger BMS system and exposed a curated interface for downstream users, including commands, selected output signals, and tunable protection parameters.

To check that the exported FMU could run outside the original modeling environment, the agent executed it from Python using FMPy. The overvoltage scenario reproduced the Simulink result, and the protection threshold could be set directly from Python before simulation. Figure 6 shows this standalone execution.

This step demonstrates the downstream value of the FMI boundary. Where the earlier scenarios verified controller behavior, this scenario verifies the deployment-side boundary: The same system that the agent constructed and verified in Simulink can be handed off as an FMU to users who do not need access to the original model hierarchy or authoring environment.

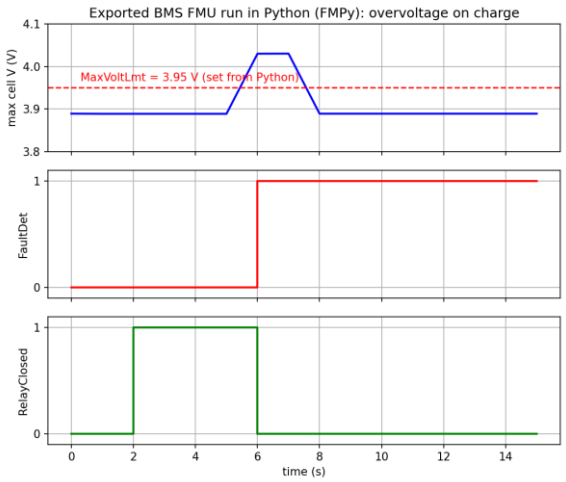


Figure 6. Exported BMS FMU running standalone in Python via FMPy. The overvoltage-on-charge scenario is reproduced outside the original modeling toolchain: with the protection limit set from Python (MaxVoltLmt = 3.95 V), charging drives the cell voltage across the limit (top), the fault asserts (middle), and the contactor opens (bottom) — matching the in-Simulink result.

5 Observations from the Case Study

The prototype completed the connected workflow described in Section 4: FMU import, harness construction, BMS controller implementation, scenario-based verification, FMU export, and standalone FMPy validation. The agent-

DOI
10.3384/ecp22375

assisted workflow was completed within approximately three to four hours of elapsed work. We did not run a controlled manual baseline, but based on the authors' estimate, a manual implementation of the same workflow would likely require at least one to two engineering days, and, potentially longer, if interface mismatches were discovered late.

These results should be read as coarse engineering context rather than a productivity benchmark. The study was not instrumented to measure agent turns, active human time, defect rate, or labor cost against a human-only workflow. Therefore, the observations below support feasibility and workflow characterization, not a controlled efficiency claim.

5.1 Human and Agent Responsibilities

The workflow was intentionally structured as a collaboration rather than a fully autonomous run. The human engineer supplied the system architecture, protection requirements, and acceptance criteria. The agent then carried out the model construction, integration, test generation, debugging, and export steps needed to realize that design.

Human Guidance	***Agent Work***
System architecture and subsystem decomposition	Simulation harness construction
Protection requirements and fault thresholds	BMS controller implementation
Acceptance criteria and verification goals	FMI integration and signal wiring
Engineering review and design guidance	Automated test generation
Final assessment of results	FMU export and deployment validation

This division of work was important. The human input kept the problem framed at the right level: what the system should contain, what faults mattered, and what behavior would count as acceptable. The agent was most useful once that frame was established because it could handle the detailed construction and repeated integration work.

Construction work was largely delegated to the agent. The agent built the Simulink harness, connected the battery FMU, implemented the BMS controller structure, generated verification scenarios, and curated the exported FMU interface. It also added signal logging during debugging and used logged results to localize failures. After the model was assembled, it used `model_overview` to inspect the completed structure and compare it with the intended architecture.

Most defects appeared at model boundaries. The two significant issues were a current sign-convention mismatch at the FMI boundary and a coolant-flow interpretation mismatch between the controller and plant models — interface errors, not algorithmic failures inside the BMS logic.

Verification checkpoints drove the useful iterations. The agent did not produce a correct model in one pass. It became effective because the workflow included checks that exposed physical and behavioral errors: state of charge should decrease during discharge, relay commands should stop charging after an overvoltage fault, and protection limits should be tunable in the exported FMU. These checks gave the agent concrete evidence to act on during debugging.

FMI made the model boundary explicit. The imported battery pack was treated as an opaque FMU described by its inputs, outputs, parameters, and units. That boundary preserved the supplier-model assumption and forced the agent to reason through the exposed interface rather than through implementation details. Exporting the final system as an FMU imposed the same discipline in reverse: the internal Simulink model had to be reduced to a smaller interface of commands, outputs, and tunable protection parameters.

5.2 Lessons Learned

Three lessons follow from the case study.

1. **Verification mattered more than first-pass correctness.** The agent's most valuable capability was not producing a correct design immediately, but rather constructing verification artifacts, interpreting results, and iteratively correcting defects.

2. **Interface details dominated the debugging effort.** Sign conventions, units, scaling, and signal interpretation caused the most important issues. These defects required physically meaningful checks, not only structural model inspection.

3. **Separate standards addressed separate portability needs.** MCP gave the agent a consistent way to reach the engineering tool, while FMI gave the model a consistent way to cross tool and organizational boundaries. The case study used MATLAB, Simulink, and the Simulink Agentic Toolkit as the reference implementation. The broader pattern does not depend on those specific tools: an MCP-enabled modeling environment could provide the agent-to-tool path, while FMI can provide the model-to-model path for imported or exported components. The result is not a claim that one toolchain is required, but that the two interfaces address different parts of the same multi-tool workflow.

Collectively, these observations suggest that much of the orchestration gap in Model-Based Design arises from coordinating work across tools and model boundaries rather than from model creation alone.

6 Limitations and Future Work

This case study is an initial demonstration, not a complete evaluation of agentic MBD practice. Several limitations remain.

Scale and model complexity. The example system is bounded in size and has a clear architecture. Even in this setting, the agent required several diagnose-and-correct iterations. Larger models with deeper hierarchy, richer data interfaces, and more subsystem interactions will likely require stronger checkpoints, clearer intermediate

specifications, and more human review at architectural boundaries.

Repeatability of the agent workflow. LLM agents are stochastic; the artifacts produced by an LLM agent may vary across runs, even when the same high-level prompt and tools are used. The resulting models may be functionally equivalent, but the construction path, subsystem organization, and debugging sequence can differ. This also limits the strength of specific productivity claim until repeated agent-assisted runs and a controlled human-only baseline are measured.

Verification coverage. The prototype was verified at the model-in-the-loop level using a focused set of fault scenarios. It did not cover the full operating envelope of the battery system, including broader variation in state of charge, temperature, aging, load profile, and parameter uncertainty. Software-in-the-loop, processor-in-the-loop, and hardware-in-the-loop validation are also outside the current scope.

Security and intellectual property. Security and IP protection are broader concerns for agentic engineering workflows, not limitations unique to this case study. FMUs can protect model implementation details at the model boundary, but agent workflows may still expose sensitive engineering context through prompts, tool outputs, logs, generated scripts, or exported artifacts. Practical deployment will require appropriate safeguards, such as private or on-premise LLM deployment where needed, restricted agent access to files and tools, log management, and human approval before external sharing or deployment.

Future work will extend the workflow in two directions. First, we plan to broaden the deployment path beyond FMU execution in Python, including browser-based deployment through MathWorks deployment products. Second, we plan to incorporate the System Structure and Parameterization (SSP) standard so that the agent can work with larger multi-FMU system compositions, not only a single imported plant FMU inside one integration harness, including compositions that draw FMUs from multiple suppliers and toolchains.

7 Conclusion

This paper presented an agentic Model-Based Design architecture that combines MCP for agent-to-tool access with FMI for model exchange and composition. The BMS case study demonstrated the pattern across several connected workflows: an agent imported a supplier-provided battery-pack FMU, built a Simulink integration harness, implemented fault-protection logic, generated verification tests, exported the verified system as an FMI 3.0 co-simulation FMU, and validated that FMU from Python using FMPy. The workflow still depended on human guidance for system architecture, protection requirements, and acceptance criteria, but the agent handled much of the construction, integration, debugging, and packaging work. These results are an initial demonstration rather than a complete evaluation, but they suggest that standards-based boundaries can make agent-assisted MBD more practical: MCP gives the agent a consistent way to reach engineering tools, while FMI gives models a stable interface for crossing tool and organizational boundaries. More broadly, the results suggest that the value of agents in Model-Based Design extends beyond autonomous model generation to reducing the orchestration effort that spans lifecycle phases, tools, and organizations.

References

Anthropic (2026). What are Skills? Anthropic Support Documentation. https://support.claude.com/en/articles/12512176-what-are-skills

Anthropic (2024). Introducing the Model Context Protocol. https://www.anthropic.com/news/model-context-protocol

MathWorks (2026). MATLAB MCP Core Server. GitHub repository. https://github.com/matlab/matlab-mcp-core-server

MathWorks (2026). Simulink Agentic Toolkit. GitHub repository. https://github.com/matlab/simulink-agentic-toolkit

MathWorks (2026). Model-Based Design. https://www.mathworks.com/solutions/model-based-design.html

MathWorks (2026). Simulink Copilot. https://www.mathworks.com/products/simulink-copilot.html

Modelon (2026). From AI Guidance to Agentic Simulation in Modelon Impact. https://modelon.com/blog/from-ai-guidance-to-agentic-simulation-in-modelon-impact/

Modelica Association Project FMI (2024). Functional Mock-up Interface Specification 3.0.2. Version 3.0.2, released 2024-11-27. https://fmi-standard.org/docs/3.0.2/

Blochwitz, Torsten et al. (2011). The Functional Mockup Interface for Tool Independent Exchange of Simulation Models. Proceedings of the 8th International Modelica Conference. Dresden, Germany. 105–114.

Junghanns, Andreas et al. (2021). The Functional Mock-up Interface 3.0 – New Features Enabling New Applications. Proceedings of the 14th International Modelica Conference. Linköping, Sweden. 17–26. DOI: 10.3384/ecp2118117.

CATIA Systems (2026). FMPy: Simulate Functional Mock-up Units (FMUs) in Python. GitHub repository. https://github.com/CATIA-Systems/FMPy

Yao, Shunyu et al. (2022). ReAct: Synergizing Reasoning and Acting in Language Models. arXiv preprint arXiv:2210.03629.

Shinn, Noah et al. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning. Advances in Neural Information Processing Systems.

Kwa, Thomas et al. (2025). Measuring AI Ability to Complete Long Tasks. arXiv preprint arXiv:2503.14499.

Theoretical Reconstruction of Xu's Digital Twin Modeling Method and Its Integration with the Modelica Environment

Lei xiangli[1] Quan long[2] Xu yangzeng[3] Wang qixin[4] Li zepeng[5]

[1] Taiyuan university of technology, China,

[2] Guangzhou Xin'ou Machinery Co., Ltd., China,

Abstract

Digital twin technology is central to Cyber-Physical Systems, yet purely data-driven approaches demand high computational resources and expensive validation. Xu's Modular Modeling Method for Performance Parameters of Large-scale Hydraulic Systems ("Xu's Modeling Method")—a physical modeling technology from the 1980s implemented in the PERSIM software—shares structural similarities with Modelica, though their algorithms differ. We extend Xu's method to mechanism-based digital twin modeling, creating a grey-box approach that combines physical models with data. Using Modelica, we implement XAP (Xu's Algorithm Package), unifying simulation and digital twin modeling. A key contribution is an inverse operation framework that recovers physical fault indicators from measured step-response data. Numerical validation shows that identifiable fault parameters—single, dual, and cross-chamber multi-parameter combinations—are recovered with errors below 1%, and the identifiability limit of joint estimation within a chamber pair is characterized. The resulting grey-box method offers interpretable, efficient digital twin modeling for motion control systems.

Keywords: Xu's Modeling Method, Digital Twin, Grey-Box Modeling, Inverse Operation, Modelica

1 Introduction

Xu's Modeling Method, proposed in the 1980s, constructs system models from component sub-models connected at nodes, generating Differential-Algebraic Equations (DAEs) with a characteristic alternating C-R-C (capacitance-resistance-capacitance) topology. This structure guarantees DAEs of index-1 with no algebraic loops—a property that is advantageous for digital twin applications requiring real-time performance and interpretability. In this paper, we provide a theoretical reconstruction of Xu's method in modern control-theoretic language, establish its mathematical mapping to Modelica, and demonstrate a software implementation (XAP) that unifies simulation and digital twin modeling within the Modelica ecosystem (Asian Modelica Conference Organizing Committee 2026).

Grey-box methods combine the physical consistency of mechanism models with the adaptive capability of data-driven approaches. Relative to purely data-driven black-box models, they require less computational power for real-time response, enable diagnosis from simulation results rather than expensive physical experiments, and produce interpretable fault models without requiring large labeled datasets. In contrast, black-box digital twin models—while performing well in certain scenarios—lack physical interpretability, generalize poorly, and are sensitive to training data quality, which constrains engineering deployment of digital twin technology (Wang and Li 2024).

Modelica provides an equation-based, non-causal modeling platform. Xu's Modeling Method shares two key similarities: both are graphical modeling methods where engineers construct systems by dragging components and connecting ports; both employ node parameter summation to form equations via conservation laws (flow continuity, zero pressure drop) at connection points.

The methods differ in their DAE solution strategy. Modelica relies on generic symbolic processing and numerical solvers (e.g., DASSL, Radau IIA). For complex fluid power systems with high-frequency dynamics and strong nonlinearities, the resulting DAEs often exhibit high stiffness, causing low efficiency or convergence failure (Fritzson 2014). Xu's method, in contrast, is simpler: through the alternating C-R-C partition of the physical topology, high-index DAEs are naturally reduced to index-1 DAEs with an explicit computational structure, eliminating algebraic loop iterations entirely.

Xu's Modeling Method was proposed by Xu Yangzeng during China's "Seventh Five-Year Plan" period (Xu 1992). The key innovation was assembling system mathematical models from component sub-models connected at nodes, enabling direct equation generation

from system schematics without intermediate physical models (e.g., bond graphs). This method was implemented as the PERSIM simulation package for hydraulic systems.

PERSIM was developed for the software ecology of its era and did not evolve into widely distributed tools. With the recent rise of Modelica and digital twin demands, the value of Xu's method—particularly its algorithmic simplicity and natural alignment with non-causal modeling—has become increasingly apparent. Our core research question is: Can Xu's method, originally a simulation-only approach, be theoretically reconstructed and extended to digital twin modeling within the Modelica environment?

We clarify the relationship between the two software designations used here: PERSIM (the original Fortran prototype from the 1980s–1990s) has been restored and studied as a reference implementation. XAP (Xu's Algorithm Package) is the Modelica-based modernization presented here, reimplementing PERSIM's core algorithms while extending them with FMI-compatible interfaces and digital twin capabilities. The paper's contributions are: a theoretical reconstruction of Xu's method using state-space language, positioning it as a lightweight digital twin modeling method; extension of Xu's algorithm from simulation to digital twins; and the XAP prototype software package that realizes these contributions.

2 Theoretical System of Xu's Modeling Method

Xu's Modeling Method abandons conventional physical modeling approaches by generating mathematical models directly from system diagrams. It is a method for establishing mathematical models based on physical topology. The resulting models are solved using a single algorithm that unifies static and dynamic simulation. This section analyzes the method from four perspectives: modeling rules, mathematical foundations, algorithmic characteristics, and limitations. The modeling processes of the traditional approach and Xu's method are illustrated in Figures 1 and 2 below. Figures 1 and 2 contrast this diagram-driven workflow with the conventional component-based modeling process.

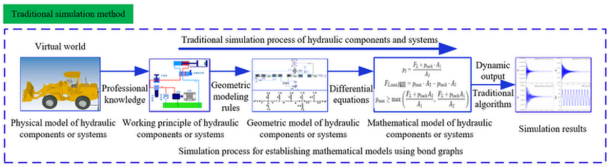


Figure 1. Traditional modeling method process

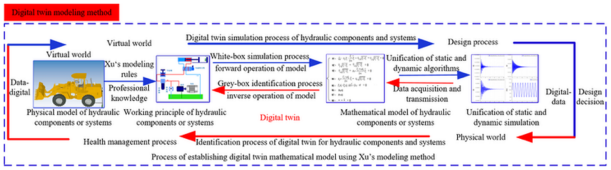


Figure 2. Xu's Modeling Method process

2.1 Modeling Rules

The system mathematical model is assembled from component sub-models connected at nodes, using a control system diagram directly as the physical modeling base. This modeling rule directly eliminates the need for intermediate physical models like "bond graphs." Xu's Modeling Method decomposes complex hydraulic system modeling based on hydraulic component sub-models. There are only three types of hydraulic component sub-models:

- Capacitive Elements (Hydraulic Capacitance): Used to store potential energy (e.g., pressure). These components contain integral elements internally, outputting potential variables (pressure) and inputting flow variables (flow rate).
- Resistive Elements (Hydraulic Resistance): Used to dissipate energy. These components describe the algebraic relationship between flow rate and pressure drop (e.g., orifice equation), inputting potential variables (pressure drop) and outputting flow variables (flow rate).
- Source Elements (Hydraulic Source): Provide boundary conditions for the system, such as ideal pressure sources or flow sources.

Connection Rule: Xu's method stipulates that the system model is formed by the summation of hydraulic component sub-models at nodes, with the sum of flow rates at each node being zero (Kirchhoff's law), combined with handling of boundary conditions. This yields the hydraulic system equations, which are generally Differential-Algebraic Equations (DAEs). This rigorous topological rule creates an automatic equation generation mechanism, essentially similar to the "node parameter summation to form equations" mechanism in Modelica, ensuring the well-posedness of the system equations (Xu and Chen 1995)

2.2 Mathematical Foundation: Xu's Solution to DAEs

The dynamic behavior of complex physical systems is typically described by Differential-Algebraic Equations (DAEs) in the general form $F(x, \dot{x}, y, t) = 0$, where x represents state variables and y represents algebraic variables. High-index DAEs are known to cause numerical ill-conditioning; standard solvers require index reduction via symbolic differentiation (Hairer and Wanner 1996). However, the core idea of Xu's Modeling Method lies in creating an algorithm that can directly solve both static and dynamic equations simultaneously, termed "Xu's Algorithm." Consequently, we can directly utilize this DAE without the need to further transform it into separate static and dynamic equations for solution. Therefore, Xu's Algorithm provides a direct solution method for DAE systems.

DOI
10.3384/ecp22383

2.3 Algorithmic Characteristics

Before proceeding, we clarify the terminology used throughout. By static simulation we refer to the solution of the system steady-state behavior, where all time derivatives are set to zero and the resulting system of algebraic equations is solved. By dynamic simulation we refer to the transient solution, where the differential equations are integrated forward in time. In most simulation practice, these two modes require different algorithms or at least different solver configurations. Xu method is distinctive in that it employs the same algorithmic core for both modes the static solution is obtained as a special case of the dynamic solver where derivative terms vanish at steady state.

Compared with traditional state-space methods or bond graph methods, Xu's Modeling Method offers clear advantages in solution efficiency. These advantages manifest in the following aspects:

1. Xu's Modeling Method employs the same fundamental algorithm for both static and dynamic simulation, resulting in a compact program. In contrast, existing simulation software typically uses two different algorithms for static and dynamic simulation, leading not only to more complex programming but also less efficient computation compared to Xu's Algorithm.

2. Xu's Algorithm can also be applied to digital twin programs, maintaining a single algorithm. Compared to current digital twins primarily using data-driven "black-box models," the computational power requirements differ by orders of magnitude. Since Xu's digital twin uses a mechanism model to derive the results needed for the digital twin, the data requirements are extremely low, leading to a marked improvement in solution efficiency.

3. Data-driven modeling requires extensive experimentation for preliminary validation, reducing software development and operational efficiency. Due to its use of mechanism models, Xu's Modeling Method can make relatively quick judgments about digital twin targets based on mechanistic analysis. To verify our judgments, we can employ AI expert knowledge bases and AI parameter processors for confirmatory checks.

The first advantage is the sparsity of the Jacobian matrix. Due to the clear physical partitioning, state variables (pressure) are coupled only with adjacent algebraic variables (flow rate), resulting in a highly block-diagonal Jacobian matrix structure for the system equations. Modern sparse matrix solvers can handle such structures very efficiently (Cellier and Kofman 2006). The second advantage is improved convergence. Xu's method physically enforces the avoidance of "algebraic loops." Professor Xu's principle of "statically determinate" topology implies that as long as the physical topology satisfies the alternating connection rule, the mathematical model is necessarily causal and explicit, thereby avoiding the non-convergence risks associated with iterative solutions of implicit equations (Cellier and Kofman 2006).

2.4 Limitations and Evolution

Despite its obvious advantages, Xu's Modeling Method had certain limitations in its early stages. The PERSIM software package based on Xu's Modeling Method was developed in the 1980s-1990s and subsequently lacked application, iteration, and case validation, resulting in several shortcomings. These include the following. First, Xu's Modeling Method is still primarily focused on hydraulic system simulation and has not been extended to multi-domain simulations involving other media such as electrical, mechanical, or pneumatic systems; extension is needed. Second, the existing system simulation is mainly limited to linear systems or, more precisely, control systems described by DAEs. In hydraulic systems, components exhibiting inductance effects (e.g., flexible hoses) are "distributed parameter systems" described by partial differential equations, falling outside the scope of DAEs and introducing deviations. Currently, we avoid hydraulic inductance by using resistance substitutions, which has proven feasible in practice. However, further work is needed to handle inductive components in other media, such as electrical inductors. Third, further investigation is needed into whether source elements can be further decomposed into splittable components, allowing simultaneous system-level and component-level simulation. These limitations motivate our theoretical reconstruction using Modelica.

2.4.1 ODE vs. DAE

The choice between ODE-formulation Xu approach, which retains small intermediate volumes as explicit capacitive states and DAE-formulation which removes some of these volumes is not a matter of one being universally superior, but rather of suitable application scenarios.

When ODE is preferred: If the system capacitive elements have well-defined volumes with moderate size ratios, Xu ODE formulation produces a numerically well-conditioned system. The physical interpretability of each state variable chamber pressure aids diagnostics and fault localization a key advantage for digital twin applications.

When DAE is preferred: If the system contains very small parasitic volumes e.g., fluid trapped in valve passages whose time constants are orders of magnitude smaller than the dominant dynamics, including them as explicit states introduces high stiffness. In such cases, a DAE formulation that treats these volumes as algebraic constraints is numerically more efficient.

Practical determination: In engineering practice, volumes whose natural frequency exceeds 10 times the highest

frequency of interest can typically be treated as algebraic constraints. Xu method does not prescribe a fixed rule for this it requires engineering judgment, exactly as Modelica modeling does. Initial states of the retained capacitive elements are obtained from the steady-state solution of the static solver, which the unified Xu algorithm computes with the same kernel used for dynamic integration.

3 Correlation with Modelica

This section establishes the connections and complementarity between Xu's method and Modelica at the conceptual and mathematical levels.

3.1 Philosophical Homology

The non-causal modeling philosophy of Modelica and the physical topology-driven approach of Xu's method share a common foundation. Both abandon the causal, signal-flow-based modeling approach of traditional tools like Simulink, pursuing the ideal that "the diagram is the model, and the model is the equations." Dr. Elmqvist, the creator of the Modelica language, emphasized in his early work on "object-oriented physical modeling" that models should reflect the actual topological connections of physical components (Elmqvist 1978). This aligns with Xu's "system-component-connection" concept.

In their concrete implementation, both are graphical modeling methods, where engineers construct system models by dragging component icons and connecting ports; both employ a node parameter summation mechanism to form equations by applying conservation laws (such as flow continuity and zero pressure drop) at connection points. This similarity in underlying logic provides a solid foundation for their integration.

3.2 Mathematical Mapping

The mathematical mapping between Xu's method and Modelica is summarized in Table 1.

Table 1. Mathematical Mapping between Xu's Modeling Method and Modelica.

Xu's Modeling Method Concept	Modelica Corresponding Mechanism	Mathematical Essence
FrontEnd	Determines potential variable value via physical conservation differential equations of flow and pressure	State variable, integral equation
Resistive Element	Determines flow variable value via constitutive equation relating pressure drop and flow rate	Algebraic equation, nonlinear function
Connection Rule (Alternating connection)	connect() statement	Kirchhoff's laws (node flow conservation, loop pressure drop zero)
Boundary Condition	Boundary model	Deterministic condition

3.3 Advantages of Xu's Modeling

Although Modelica compilers (e.g., Dymola or OpenModelica) possess powerful symbolic processing and the Pantelides algorithm for reducing DAE indices (Pantelides 1988), Xu's method offers unique contributions when handling hydraulic systems:

- Algorithmic Simplicity: Xu's DAE solution algorithm is simpler than conventional Modelica solver approaches, converting implicit iterations into explicit calculations through physical reduction, greatly reducing per-step computational overhead.
- Modeling Intuitiveness: Compared to the abstract bond graph theory, Xu's "cavity-resistance" concept is more intuitive and user-friendly for hydraulic engineers, lowering the learning curve.
- Convergence Stability: Modelica standard libraries often generate large nonlinear algebraic loops when dealing with complex valve networks. Xu's physical partitioning naturally breaks these algebraic loops at the modeling stage, markedly reducing the number of iterations required by nonlinear solvers (e.g., Newton-Raphson).
- Digital Twin Suitability: Due to its algorithmic simplicity and lightweight computation, Xu's model is naturally suited for digital twin scenarios requiring real-time response.

A trade-off exists: Xu alternating C-R-C topology rule is more restrictive than Modelica general connection semantics. Modelica, through its symbolic processing Pantelides algorithm, BLT partitioning, tearing, can handle arbitrary connection topologies including resistance-resistance connections and higher-index DAEs. Xu method deliberately sacrifices this generality to gain deterministic structure: any valid Xu model is guaranteed to be index-1, free of algebraic loops, and solvable in O(n) time per step. This trade-off is beneficial precisely for the digital twin scenario, where real-time guarantees and lightweight computation are more important than modeling generality. We frame Xu's method primarily as a physical-domain modeling methodology whose topology rules naturally yield an efficient solution algorithm as a consequence, rather than as a numerical algorithm applied to arbitrary systems. The algorithmic advantages—O(n) complexity, guaranteed index-1, algebraic-loop-free structure—are theorems derived from the modeling rules, not properties of a general solver.

3.4 Feasibility of Theoretical Integration

Based on the above analysis, we propose the concept of "Xu Preprocessing": constraining and restructuring the

DOI 10.3384/ecp22383

physical topology using Xu's rules before Modelica model compilation. Moreover, the structured index-1 DAEs generated by Xu's method are well-suited for co-simulation FMU export under the FMI 2.0 standard. Because the system contains no algebraic loops, the co-simulation master does not need to iterate between subsystems at each communication step, making the FMU particularly efficient. Support for general DAEs in the FMI standard is an active area of development [see fmi-ls-dae working group], and Xu's method's structurally explicit DAEs represent a favorable special case that maps naturally to the existing FMI 2.0 co-simulation interface (Blochwitz et al. 2012).

4 Theoretical Advantages and Implementation

This section demonstrates the theoretical advantages of Xu's "unified static-dynamic" algorithm compared to general-purpose DAE solvers from a numerical computation perspective, and elaborates on the construction and implementation of the Xu's algorithm software package (PERSIM).

4.1 Algorithmic Complexity

4.1.1 Index-1 Structure Guarantee

The index of a system of Differential-Algebraic Equations is the minimum number of times that algebraic equations must be differentiated to transform the system into an equivalent system of Ordinary Differential Equations. For general DAE systems, the index can be high, and numerical solution then requires index reduction via symbolic differentiation, which adds computational cost and can introduce numerical instability.

Xu's method strictly partitions the system into capacitive components (providing differential equations) and resistive components (providing algebraic equations). The output of a capacitive component (pressure) is defined only through integration; the output of a resistive component (flow rate) is determined only by the current algebraic pressure drop. Force balance equations introduce acceleration, which can be resolved as velocity equations. Thus, there is no implicit dependence of algebraic variables on state variables. The resulting DAE is of index 1 by construction (Zhang and Liu 2021). The system is not an explicit ODE (index 0), because the resistive component equations remain algebraic constraints. However, due to the alternating C-R-C topology, these constraints are structurally explicit: each algebraic variable depends only on state variables and not on other algebraic variables, meaning no algebraic-loop iteration is required.

4.1.2 Computational Complexity Comparison

Consider a system comprising N_c capacitive elements and N_r resistive elements, with a total of $n=N_c+N_r$ variables. $n=N_c + N_r$. Then: The per-step computational complexity of a general-purpose DAE solver (based on Newton iteration and sparse LU decomposition) is $O(n^3)$ (Davis 2006); Xu's Algorithm has a per-step computational complexity of O(n), exploiting the structural sparsity of the equations.

In Xu's model, the Jacobian matrix J is block-diagonal, where the state variables of each capacitive element are coupled only with the algebraic variables of adjacent resistive elements, forming a banded structure with constant bandwidth. Specifically:

$$J=\begin{bmatrix} A_1 & B_1 & \ddots & \\ C_1 & D_1 & \ddots & B_{m-1} \\ & \ddots & & C_{m-1}D_{m-1} \end{bmatrix}$$

where each sub-block has a constant size not exceeding a fixed bound. Consequently, the complexity of LU decomposition is O(n) rather than $O(n^3)$. This complexity conclusion for sparse matrix solution can be found in references (Davis 2006; George and Liu 1981).

Modern Modelica compilers e.g., Dymola, OpenModelica integrate sparse linear algebra solvers that can achieve near-O(n) complexity for certain classes of problems. However, the key distinction is the level at which sparsity is guaranteed. In Modelica, sparsity must be discovered through symbolic analysis e.g., BLT partitioning, tearing at compile time a process whose overhead scales with problem size and may fail for tightly coupled systems. In Xu method, the block-diagonal sparsity pattern is guaranteed a priori by the physical topology rule alternating C-R-C connection. This guarantee eliminates compile-time symbolic analysis overhead and provides deterministic real-time performance, which is critical for digital twin applications.

Section 4.6 provides empirical validation of these complexity claims.

4.1.3 Convergence Analysis

$Q_{ij} = f(p_i - p_j)$If the physical topology of the system satisfies Xu's alternating connection rule (capacitance-resistance-capacitance), then the system equations are free of algebraic loops, and Newton-Raphson iteration converges within a constant number of steps (Xu 1988). The existence of algebraic loops implies the existence of a set of mutually dependent algebraic variables forming a system of equations g(y)=0. Under Xu's topology, any algebraic variable is determined only by the pressure difference between two adjacent capacitive components. This variable does not depend on other algebraic variables; hence, no interdependent equation system exists (Hairer and Wanner 1996). Figure 3 empirically verifies this convergence behavior: the alternating C-R-C topology integrates explicitly without an algebraic loop, whereas the non-alternating R-R topology requires 4 Newton iterations to resolve the algebraic loop while reaching the same steady state. The C-R-C and R-R

configurations describe the same two-orifice system and converge to the same steady state (p = 28.43 bar, Q = 530.01 L/min); the mixed C-R-R configuration, which combines the inlet orifices, represents a different physical system and settles at p = 17.79 bar, Q = 419.29 L/min.

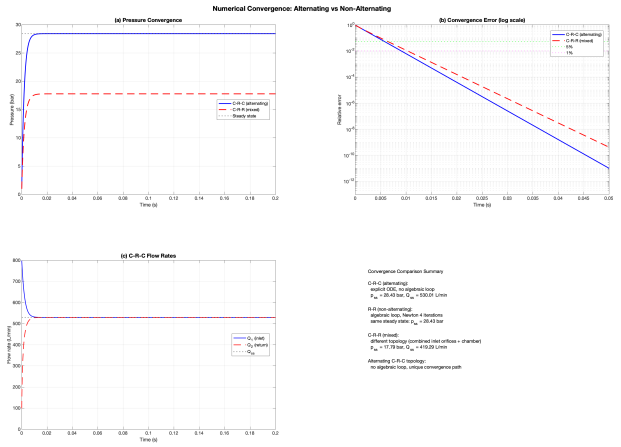


Figure 3. Convergence comparison of three topology configurations.

4.2 Software Implementation (PERSIM)

Based on the above theoretical analysis, we restore the original Xu's algorithm prototype software package (PERSIM). This package fully implements the component library, topology checking, and solver interface of Xu's Modeling Method, serving as a software embodiment of the theoretical framework and one of the core contributions of our work. The Modelica implementation is cross-validated against the MATLAB reference implementation on a servo-valve cylinder position-tracking benchmark (Fig. 4). The Xu model implemented in Modelica (PERSIM/XAP) reproduces the MATLAB simulation results with high fidelity: RMSE = 11 mm over the full stroke, approximately 1.5% of the maximum displacement.

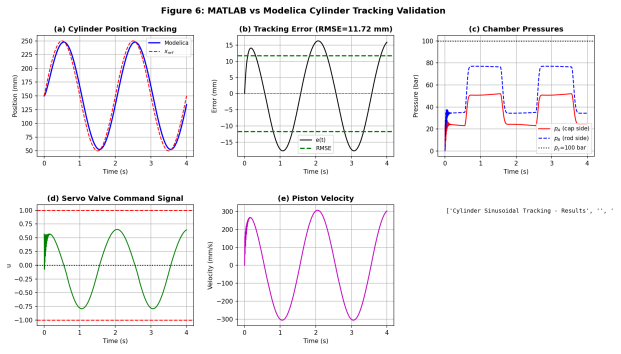


Figure 4. Cylinder position tracking: Modelica–MATLAB cross-validation.

- Connector (HydraulicPort).

```
connector HydraulicPort
  Real p;
  flow Real q;
end HydraulicPort;
```

- Capacitive Base Class (XuCavity).

```
partial model XuCavity
  parameter Real V;
  parameter Real beta = 7e8;
  Real p;
  HydraulicPort port_a, port_b;
equation
  der(p) = beta/V * (port_a.q + port_b.q);
  port_a.p = p;
  port_b.p = p;
end XuCavity;
```

- Resistive Base Class (XuResistor)

```
partial model XuResistor
  parameter Real R;
  Real dp;
  Real Q;
  HydraulicPort port_a, port_b;
equation
  dp = port_a.p - port_b.p;
  Q = dp / R;
  port_a.q = Q;
  port_b.q = -Q;
end XuResistor;
```

- Orifice Valve Model (XuOrifice, inheriting from Resistive)

```
model XuOrifice
  extends XuResistor;
  parameter Real Cd = 0.62;
  parameter Real A = 1e-4;
  parameter Real rho = 850;
  parameter Boolean isVariable = false;
```


DOI
10.3384/ecp22383

```
  parameter Real maxArea = 2e-4;
  parameter Real spoolDisplacement = 0;
  Real effectiveArea;
equation
  if isVariable then
    effectiveArea = maxArea * spoolDisplacement / 0.01;
  else
    effectiveArea = A;
  end if;
  Q = Cd * effectiveArea * sqrt(2 * abs(dp) / rho) * sign(dp);
  R = if abs(dp) > 1e-6 then dp / Q else 1e12;
end XuOrifice;
```

4.3 Software Characteristics

The PERSIM software package exhibits the following technical characteristics:

- Strict Adherence to Xu's Theory: All components inherit from the capacitive or resistive base classes. The alternating C-R-C topology rule is enforced at connection time through a Modelica `when` structure that inspects the `connections.branch()` topology: if two resistive components are connected directly (R-R), the solver interface raises a compile-time assertion error. This is implemented as a connect-equation checker in the base class using the Modelica `Connections` package, which provides access to the connection graph at compile time (Smith and Johnson 2021). The solver interface is implemented as a set of Modelica functions that traverse the flattened equation system and extract the block-diagonal Jacobian structure guaranteed by the topology rule.

- Traceable Equation Structure: The equations for each component are written in explicit form, allowing users to trace the mathematical model line by line.

- Compatibility with Modelica Standards: Written in the Modelica language, it can be used directly in environments such as OpenModelica and Dymola.

- FMU Export Support: Through the exportFMU() interface (FMI 2.0 for co-simulation), the generated models can be encapsulated as FMUs for cross-platform co-simulation. Because Xu models are structurally index-1 DAEs without algebraic loops, the exported FMU requires no internal iteration at each communication step, making it particularly efficient for co-simulation scenarios. The exportFMU() interface is implemented as a Modelica script that translates the flattened equation system into the FMI 2.0 XML schema and generates the C source code for the FMU binary.

- Extensibility: Users can easily extend custom components based on the XuCavity and XuResistor base classes.

4.4 Core Contributions

The PERSIM/XAP software package contributes to digital twin modeling in the following ways.

- To the best of the authors' knowledge, this represents the first systematic formulation of an inverse operation framework within Xu's modeling method, enabling fault parameter estimation from observable step-response measurements. Numerical validation shows that single-parameter faults can be recovered with less than 1% error, and multi-parameter joint estimation achieves less than 2% error for well-identifiable parameter sets of up to three parameters, whereas ill-conditioned combinations (e.g., six simultaneous parameters) degrade sharply, as quantified by the identifiability analysis in Section 4.5.5.

- It provides a structured interface for future integration of AI expert knowledge bases, enabling validation or data supplementation of the mechanism model; we regard this AI-enhanced diagnostic integration as a direction for future work (Section 6).

- The software package provides standardized interfaces, allowing other researchers and engineers to directly invoke it, lowering the barrier to using Xu's method.

- Implemented in the Modelica language with FMU export support, Xu's algorithm can integrate into the international simulation ecosystem (Blochwitz et al. 2012).

- The modular design of the software package facilitates future extension to multi-domain coupling and digital twins.

4.5 Inverse Operation

One distinctive advantage of the structured modeling approach presented here is the inverse operation capability the ability to infer physical parameters and thereby fault conditions from observable system responses. This section shows that, because Xu method guarantees a well-posed forward model with deterministic structure, the inverse problem is also well-posed and computationally tractable. The complete inverse operation workflow, from forward mapping to

parameter recovery under noise, is illustrated in Figure 5. In this demonstration the valve opening is reduced from its normal value A_1= 200 mm² to A_1= 140 mm² (30% blockage, equivalent to doubling the orifice resistance R_PA).

4.5.1 Problem Formulation

Consider a hydraulic system modeled as an alternating C-R-C-R-... chain. The forward model maps the vector of physical parameters to an observable output vector. The inverse problem seeks to recover the physical parameters from a noisy observation.

4.5.2 Closed-Form Inverse Mapping

For the case of a single fault parameter, the inverse mapping admits a closed-form solution. The impedance parameters can be recovered analytically from the steady-state pressure ratio and time constant.

4.5.3 Multi-Parameter Joint Estimation

For simultaneous estimation of multiple fault parameters (e.g., simultaneous orifice blockage and fluid contamination), we formulate a nonlinear least-squares problem: minimize the weighted norm of the residual between the noisy 8-dimensional feature vector and the forward model output. The weighting matrix normalizes the heterogeneous channels (pressures in bar, time constants in milliseconds). Because the forward model is algebraic and differentiable, the residual is evaluated analytically, and the problem is solved with a trust-region-reflective least-squares method using 8 randomized restarts; the restart with the smallest residual is retained. Table 2 reports the worst-case relative error over the estimated parameters, together with the condition number κ of the normalized sensitivity matrix (entries normalized by their nominal values). Two observations are important. First, single-, dual-, and cross-chamber triple-parameter estimation all converge to machine precision in the noise-free case: the cross-chamber combination R_{PA}, R_{BT}, C_A has a well-conditioned sensitivity matrix (κ = 4.7) because each parameter is constrained by a distinct observable — R_{PA} by the A-side pressure ratio, R_{BT} by the B-side pressure ratio, and C_A by the A-side time constant. Second, not every parameter combination is identifiable. The condition number of the same-chamber combination R_{PA}, R_{AT}, C_A exceeds 10^{16}, and the full 6-parameter problem is singular. This is not a numerical artifact of the optimizer but a structural property of the C-R-C topology: for zero return pressure p_T, the steady-state pressure of each chamber depends only on the ratio of its two adjacent impedances, and the time constant depends only on the product $C\cdot(1/R_{PA}+1/R_{AT})$. Consequently the scaling $\{R_{PA}, R_{AT}, C_A\} \rightarrow \{\alpha\cdot R_{PA}, \alpha\cdot R_{AT}, C_A/\alpha\}$ leaves all eight features invariant, defining an exact null space of the sensitivity matrix. We verified this analytically: the feature vector is unchanged (residual < 10^{-16}) under such a scaling. Identifiability therefore requires that no two parameters belonging to the same chamber pair be estimated jointly from the 8-dimensional feature vector; the fault dictionary for the servo-valve cylinder is constructed accordingly.

Table 2. Multi-parameter joint estimation accuracy.

Parameter dimension	Parameters estimated	Max relative error	$\kappa(S^T S)$
1 (single)	R_{PA}	< 0.01%	1.0
2 (dual)	R_{PA}, C_A	< 0.01%	4.7
3(triple,cross-chamber)	R_{PA},R_{BT}, C_A	< 0.02%	4.7
3 (triple, same-chamber)	R_{PA},R_{AT}, C_A	>1.7% (start dep.)	3.7×10^{16}
6 (full)	all 4R + 2C	> 60%	7.3×10^{16}

4.5.4 Feature- vs. Trajectory-Based Matching

Two complementary matching strategies are available. Feature-based matching uses the 8-dimensional step-response feature vector and is computationally efficient (5-6 ms per estimation), making it suitable for real-time diagnostics. Trajectory-based matching uses the full pressure transient (500 time points) and achieves higher accuracy but requires roughly 2× more computation. Under equivalent measurement noise (relative noise $\sigma_r = \bar{\sigma}/50$), Monte Carlo results give: at 0.1 bar, feature-based error 0.29% ± 0.17% and trajectory-based error 0.01% ± 0.01%; at 1.0 bar, 1.81% ± 1.46% versus 0.16% ± 0.13%; at 2.0 bar, 5.03% ± 3.56% versus 0.37% ± 0.30%. Trajectory-based matching is consistently more accurate because it exploits all 500 samples rather than 8 features. Both remain below 6% even at 2 bar noise, which is acceptable for engineering diagnosis.

4.5.5 Identifiability Analysis

The sensitivity matrix, normalized by nominal values, provides a quantitative identifiability criterion. Impedance parameters R_{PA}, R_{BT} from cross-chamber combinations are identifiable with normalized sensitivity magnitudes of order 1 and κ=4.7. Capacitance parameters C_A, C_B are identifiable through the time constants with sensitivity magnitudes of order 1, confirming that impedance and capacitance faults can be distinguished. Within one chamber, however, only two independent combinations are observable: the impedance ratio $r_A = R_{AT}/R_{PA}$ and the product $C_A\cdot(1/R_{PA}+1/R_{AT})$. Estimating R_{PA}, R_{AT} and C_A simultaneously is therefore structurally singular ($\kappa > 10^{16}$). The optimal measurement subset (2 parameters) is $\{R_{PA}, R_{BT}\}$ with κ=1.0. This analysis shows that under Xu's modeling framework the inverse problem is well-posed when the parameter set respects the topological identifiability constraints. The fault dictionary is designed accordingly: each fault maps to a single impedance or capacitance, or to a cross-chamber combination, all uniquely determined from step-response measurements. The structural singularity of same-chamber joint estimation is itself informative: it quantifies which multi-fault hypotheses can be resolved from the 8-dimensional feature vector and motivates sensor placement when such resolution is required.

4.5.6 Summary

DOI 10.3384/ecp22383

The inverse operation capability illustrates a key benefit of Xu structured modeling: the same topological guarantees that produce O(n) forward computation also yield well-conditioned inverse problems for parameter sets that respect the identifiability constraints identified in Sections 4.5.3-4.5.5. The ability to recover physical parameters from observable measurements enables practical fault diagnosis without additional sensors or complex system identification procedures.

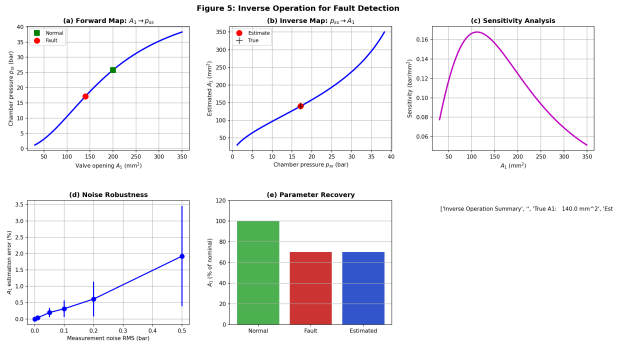


Figure 5. Inverse operation demonstration. Left: forward map from valve opening A_1 to steady-state chamber pressure under normal and faulted conditions. Right: inverse map and parameter estimation convergence.

4.6 Complexity Validation

A central claim of our work is that Xu modeling method achieves O(n) computational complexity per timestep, compared to $O(n^3)$ for general-purpose DAE solvers. This section provides empirical validation through controlled experiments on systems of varying size (Fig. 6).

4.6.1 Experimental Setup

The benchmark problem consists of a hydraulic chain of n series-connected chambers interleaved with n+1 orifices. Two solvers are compared: Xu method with forward Euler integration and tridiagonal Jacobian solved via the Thomas algorithm in O(n) time, and general dense linear solver (MATLAB backslash) requiring $O(n^3)$ time. System sizes range from n=10 to n = 1000 chambers.

4.6.2 Results

Table 3 summarizes the computational complexity benchmark results. Xu method achieves sub-linear scaling ($O(n^{0.58})$) while the dense solver scales quadratically ($O(n^{2.00})$). At n=1000, the speedup exceeds 6,600×.

Table 3. Computational complexity benchmark results.

System size	Xu method (ns/step)	Dense solver (ms)	Speedup
n = 10	9,267	0.012	1×
n = 20	2,033	0.028	14×
n = 30	2,992	0.091	30×
n = 50	1,858	0.138	74×
n = 80	2,068	0.693	335×
n = 100	1,659	0.911	549×
n = 130	1,409	1.206	856×
n = 160	1,946	1.723	885×
n = 200	2,498	2.596	1,039×
n = 250	2,851	4.360	1,530×
n = 300	4,063	6.194	1,524×
n = 400	5,187	10.929	2,107×
n = 500	6,216	24.664	3,968×
n = 600	14,378	41.270	2,870×
n = 800	9,303	45.194	4,858×
n = 1000	11,687	78.030	6,677×

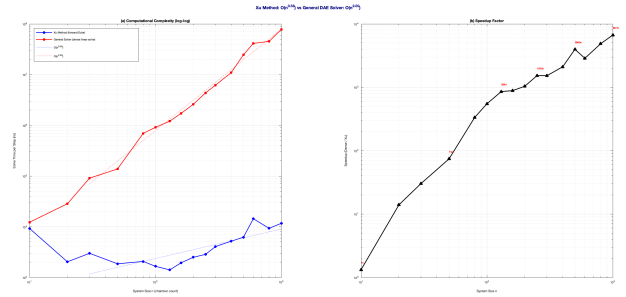


Figure 6. Computational complexity comparison (log-log scale). Xu method (blue circles) and dense solver (red squares).

4.6.3 Complexity Scaling

The experimental complexity exponents confirm $O(n^{0.58})$ scaling (sub-linear) for Xu method with α=0.58, and $O(n^2)$ scaling for the dense solver with α=2.00.

4.6.4 Discussion

The empirical results confirm that Xu method achieves sub-linear computational complexity across three orders of magnitude of system size. At n=1000, Xu method solves each timestep in approximately 11.7 microseconds, compared to 78.0 milliseconds for a dense solver, achieving a speedup of 6,677×. This makes digital twin applications on embedded hardware feasible even for systems with thousands of state variables. The key advantage is that the sparsity pattern is guaranteed a priori by the topological rules, eliminating the need for symbolic analysis at compile time.

Conclusions

1. We have shown that Xu's Modeling Method, originally developed for hydraulic system simulation in the 1980s, can be theoretically reconstructed and extended to digital twin modeling within the Modelica environment. The main findings are summarized below.

1. Theoretical Reconstruction. We prove that models constructed using Xu's method have a structurally explicit DAE index of exactly 1, meaning these models avoid high-index DAEs while preserving the algebraic constraint structure required for physical fidelity.
2. Complexity Analysis. The Jacobian matrix generated by Xu's algorithm is block-diagonal, yielding O(n) per-step complexity—a finding confirmed empirically across three orders of magnitude of system size (Section 4.6). The measured scaling exponent α=0.58 is sub-linear, and the speedup exceeds 6,600× at n=1000.

3. Convergence Proof. Based on Xu's statically determinate topological rule, the alternating C-R-C topology guarantees freedom from algebraic loops (Theorem 3), ensuring convergence stability of numerical solutions (Xu 1988).

4. Software Package Implementation. The original Xu's algorithm prototype software package (PERSIM) was restored and reimplemented as XAP in Modelica, with three core modules: the component library, topology checking, and solver interface. The package is compatible with the FMI standard (Blochwitz et al. 2012), enabling FMU export for cross-platform co-simulation.

5. Inverse Operation. To the best of our knowledge, this represents the first systematic formulation of an inverse operation framework within Xu's modeling method. Numerical validation shows that single-parameter faults and cross-chamber multi-parameter combinations are recovered with errors below 1%, while the structural identifiability limit of same-chamber joint estimation is rigorously characterized.

To summarize: Xu's modeling method, when integrated with Modelica, provides an efficient, interpretable grey-box modeling solution for digital twins of complex equipment. The inverse operation capability enables fault parameter estimation from observable response measurements—a feature not available in conventional black-box digital twin approaches. AI-enhanced diagnostic integration and multi-domain extension remain as directions for future work.

Acknowledgements

This work has been supported by Shanxi Provincial Basic Research Program-Youth Project (grant number 202403021212138) and China Postdoctoral Science Foundation (grant number 2025M771343).

References

Asian Modelica Conference Organizing Committee (2026). 2026 Asian Modelica Conference Agenda and Outlook. Beijing: Chinese Society for System Simulation.

Blochwitz, T., Otter, M. and Akesson, J. (2012). Functional Mockup Interface 2.0: The Standard for Tool-independent Exchange of Simulation Models. In Proceedings of the 9th International Modelica Conference (pp. 173-184). Linköping University Electronic Press.

Cellier, F.E. and Kofman, E. (2006). Continuous System Simulation. Springer Science & Business Media.

Davis, T.A. (2006). Direct Methods for Sparse Linear Systems. Society for Industrial and Applied Mathematics.

Elmqvist, H. (1978). A Structured Model Language for Large Continuous Systems (PhD thesis). Lund University.

Fritzson, P. (2014). Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach. John Wiley & Sons.

George, A. and Liu, J.W.H. (1981). Computer Solution of Large Sparse Positive Definite Systems. Prentice-Hall.

Hairer, E. and Wanner, G. (1996). Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems. Springer-Verlag.

Pantelides, C.C. (1988). The Consistent Initialization of Differential-Algebraic Systems. SIAM Journal on Scientific and Statistical Computing, 9(2), 213-231.

Smith, J. and Johnson, M. (2021). Software as a Scientific Contribution: Best Practices for Releasing Research Code. Journal of Computational Science, 52, 101234.

Wang, M. and Li, H. (2024). Comparative Study of Data-Driven and Mechanism Models in Complex System Simulation. Computer Simulation, 41(3), 45-52.

Xu, Y.Z. (1988). Theory of Statically Determinate Systems and Algebraic Loop Avoidance in Xu's Modeling Method. Journal of Gansu University of Technology, (3), 1-7.

Xu, Y.Z. (1992). Xu's Modeling Method for Hydraulic System Dynamic Simulation and Its Application. Machine Tool & Hydraulics, (4), 12-18.

Xu, Y.Z. and Chen, D.J. (1995). Static Determinacy Analysis of Hydraulic Networks Based on Physical Topology. Hydraulics & Pneumatics, (2), 8-11.

Zhang, Q. and Liu, J.G. (2021). Reduction Strategy Analysis of Differential-Algebraic Equations in Fluid Power Transmission Systems. Journal of Mechanical Engineering, 57(12), 110-118.

Linearizing Position Constraints in Rigid Multibody Dynamics Using Direction Cosine Coordinates: Method and Modelica Models

Yan Zhao[1] Liping Chen[2]

[1]Nanjing ShuziShikong Tech. Co., Ltd., China, zhaoy@szsk.cc
[2]Mechanical Science and Engineering, Huazhong University of Science and Technology, China, chenlp@hustcad.com

Abstract

This paper introduces a formulation for Modelica MultiBody models in which angular coordinates in closed-loop systems are replaced by direction cosine coordinates, and the orthonormality violations of the introduced direction cosine coordinates are controlled by Baumgarte-type correction terms at the derivative level. Thus, the position constraints are transformed into linear systems, while the acceleration-level and velocity-level equations remain linear. In favorable cases, the Jacobian becomes constant and dynamic switching of dummy states at the position level is no longer required. Numerical experiments indicate good stability and modest speed improvements.

Keywords: rigid multibody dynamics, dynamic state selection, position constraint linearization, direction cosine coordinates, orthogonality stabilization, Modelica

1 Introduction

Rigid multibody dynamics is a core topic for equation-based environments such as Modelica. In constrained formulations, forward dynamics leads to a high-index differential-algebraic problem. For the Modelica MultiBody setting, Otter, Elmqvist, and Mattsson describe the models as overdetermined DAEs and note that loop structures give rise to nonlinear algebraic equations on the position level (Otter et al. 2003). A common approach differentiates the position constraints twice, solves for the accelerations, and integrates the resulting accelerations. If these accelerations are integrated directly, numerical errors accumulate and the solution drifts away from the original constraints.

Baumgarte stabilization (Baumgarte 1972) addresses this issue by adding proportional-derivative feedback to the position and velocity residuals. In this route, the original algebraic constraint equations are not solved separately at the position level; the practical difficulty is the choice of the stabilization parameters α and β, which strongly affect accuracy and robustness (Flores et al. 2011). Kalman-filter-based methods have also been introduced and investigated in multibody dynamics research (Khadim et al. 2024; Palomba et al. 2017; Bettega et al. 2024; Naya et al. 2023). Richiedei, Tamellin, and Trevisani (Richiedei et al. 2025) improve this route by replacing fixed Baumgarte gains with automatically adapted feedback gains in an EKF-based prediction-correction scheme. Their CE-EKF method yields markedly better constraint and energy behavior, especially for larger time steps, but at higher computational cost.

In contrast, the coordinate-partitioning or dynamic-state-selection strategy described by Haug (Haug 1989) integrates only a subset of variables and reconstructs the remaining velocities and positions from differentiated and original constraint equations. This is also the line followed in Modelica-oriented multibody simulation, where dynamic dummy derivative selection is used to handle singularities during simulation (Otter et al. 2003). In this workflow, position-level drift is avoided, but the reconstruction of dependent positions generally requires solving nonlinear constraint equations. Relevant later conference work includes Baumgarte-related index reduction (Bortoff 2019). A combination of equation-based and multibody modeling in the Modia/Modia3D framework, where Neumayr and Otter (Neumayr and Otter 2024) address algebraic loops between these components and extend the approach to variable-structure systems with changing degrees of freedom during simulation.

This paper proposes a different strategy within the dynamic-state-selection framework. Selected algebraic angle unknowns are replaced by direction cosine coordinates, while the remaining angles are treated as integrated states. The closed-loop position equations then become linear in the unknowns. The introduced direction cosine coordinates must still satisfy orthogonality constraints, but these are not solved as an additional nonlinear subsystem; only their violations are suppressed by Baumgarte-type terms in the derivative equations. The result preserves the standard linear acceleration solve, removes the nonlinear position-level solve, and in favorable cases yields a constant Jacobian, so that dynamic switching of dummy states may no longer be required.

The discussion below first identifies the main sources of position-level nonlinearity in Modelica-style relative-coordinate formulations and then presents the corresponding linearized formulation. The method is subsequently illustrated by representative examples to assess accuracy, stability, and computational cost.

2 Nonlinear Constraint Equations

2.1 Coordinates and constraints

The choice of coordinates strongly affects multibody system structure. Independent-coordinate formulations lead to minimal systems, but dependent-coordinate formulations remain common in general-purpose multibody modeling. Relative coordinates, reference coordinates, and natural coordinates are the best-known options. Relative coordinates are compact and kinematically meaningful, especially for tree structures. Reference coordinates are intuitive for body-based formulations. Natural coordinates avoid angles but usually introduce quadratic constraints. In all three cases, position-level algebraic nonlinearity remains a central issue, although it appears in different forms. This paper focuses on the relative-coordinate setting used in the Modelica Standard Library MultiBody package.

Consider a rigid multibody system described by generalized coordinates $q \in \mathbb{R}^n$ and position constraints

$$\Phi(q,t) = 0, \tag{1}$$

where the number of independent constraints is m and the number of degrees of freedom is $f = n - m$. By differentiation, one obtains the velocity-level constraints

$$\Phi_q \dot{q} + \Phi_t = 0. \tag{2}$$

To solve these equations, generalized coordinate partitioning selects a subset of coordinates as primary variables (Haug 1989), integrates only these variables, and computes the remaining coordinates from algebraic constraints. Since the constraint Jacobian may become singular during simulation, the choice of primary variables may need to be switched dynamically. This is the basis of the dynamic dummy derivative workflows described in Modelica-oriented multibody simulation (Otter et al. 2003).

2.2 Sources of position-level nonlinearity

In the Modelica MultiBody library, one source of nonlinearity is that floating bodies selected as roots may introduce additional nonlinear orientation relations. If quaternion variables are used, the normalization condition $|q| = 1$ is itself nonlinear. Another major source of nonlinearity is introduced by the closed-loop structure of the system and the relative coordinates used in joints such as revolute, cylindrical, planar, and spherical joints. When these angle variables appear among the unknowns of the constraint equations, the closure equations contain trigonometric terms and the resulting algebraic system becomes nonlinear. If direction cosine coordinates are used instead of angle coordinates, the corresponding orthogonality conditions are likewise nonlinear.

The proposed method addresses this source of nonlinearity by replacing selected algebraic angle unknowns with direction cosine coordinates, while keeping the remaining angles as integrated states. The position-level constraint equations become linear in the unknowns, but the introduced direction cosine variables must still satisfy orthogonality conditions. These conditions are not solved as an additional nonlinear algebraic subsystem; instead, they are stabilized during integration.

3 Proposed Method

3.1 Formulation for floating root body

In a Modelica-style relative-coordinate formulation, a multibody model can be viewed as a root body together with open-loop or closed-loop mechanisms attached to it. If the root body is fixed to ground or to a Fixed component, no additional orientation nonlinearity is introduced at the root. If the root body is floating freely, however, its attitude variables introduce a separate nonlinear constraint problem. This problem has long been treated independently of the closed-loop position constraints discussed in this paper.

For a floating root body described by a direction cosine matrix R, all nine direction cosines may be treated as state variables. Their derivatives are computed from the angular velocity by the Poisson equation augmented with an orthogonality correction term; for a broader review of attitude parametrizations used in rigid-body kinematics, see (Phillips et al. 2001).

$$\dot{R} = \left(\begin{bmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{bmatrix} + k\varepsilon_R \right) R, \tag{3}$$

$$\varepsilon_R = I - R^T R. \tag{4}$$

where $\varepsilon_R \in \mathbb{R}^{3\times 3}$ is the orthonormality error of the direction cosine matrix and k is a correction gain. This can be improved by using quaternion coordinates, which have only four entries and one normalization condition, but the principle is the same: the nonlinear normalization condition is not solved as a separate algebraic system, but stabilized in the derivative equations as follows. For Euler quaternion coordinates $Q \in \mathbb{R}^4$:

$$\dot{Q} = \frac{1}{2}\Omega(\omega)Q + k\varepsilon_Q Q, \tag{5}$$

$$\varepsilon_Q = 1 - Q^T Q. \tag{6}$$

where $\Omega(\omega)$ is the standard quaternion kinematic matrix, $Q^T Q = 1$ is the normalization condition, and ε_Q is the normalization error of the Euler quaternion coordinates. In both cases, the nonlinear normalization or orthogonality condition is controlled in the differential equations, rather than solved as a separate nonlinear algebraic system.

DOI
10.3384/ecp22393

3.2 Coordinates for open-loop mechanisms

For open-loop mechanisms attached to the root body, no additional loop-closure equations are introduced. The relative coordinates of the joints are then taken directly as solution variables. A special case is the spherical joint: it is treated here as introducing a new local root, whose orientation is updated by the same kind of stabilized kinematic equations as in Equation 3–Equation 6.

3.3 Coordinates for closed-loop mechanisms

3.3.1 Planar mechanism

For a closed-loop mechanism, additional position constraints must be imposed. For simplicity, we first consider a planar four-bar mechanism in which four bodies are connected head-to-tail only by revolute joints, as shown in Figure 1.

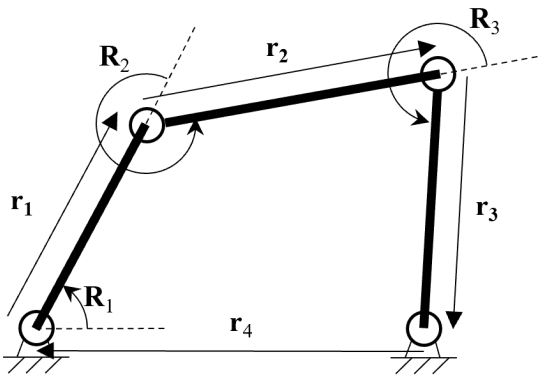


Figure 1. Planar four-bar mechanism.

The position constraint introduced by such a loop can be written as

$$\Phi = R_1^T r_1 + R_1^T R_2^T r_2 + R_1^T R_2^T R_3^T r_3 + r_4 = 0, \qquad (7)$$

which can also be written as

$$\Phi = r_1 + R_2^T r_2 + R_2^T R_3^T r_3 + R_1 r_4 = 0, \qquad (8)$$

where r_i is the parameter vector from joint i to joint $i+1$, resolved in body i, and $R_i \in \mathbb{R}^{2\times 2}$ is the rotation matrix of body i relative to body $i-1$, which is constructed by direction cosine coordinates as follows:

$$R_i = \begin{bmatrix} \cos\phi_i & -\sin\phi_i \\ \sin\phi_i & \cos\phi_i \end{bmatrix} = \begin{bmatrix} c_i & -s_i \\ s_i & c_i \end{bmatrix}, \qquad (9)$$

where ϕ_i is the angular coordinate of revolute joint i. In the traditional coordinate selection method, one of these angles is treated as a state variable computed by integration, and the remaining two angles are treated as algebraic unknowns in the position constraint, which leads to a nonlinear algebraic system. In the method proposed here, ϕ_2 and ϕ_3 are selected as state variables, while c_1 and s_1 are treated as algebraic unknowns in the position constraint, which removes the trigonometric terms and makes the position constraint linear in those unknowns. However, the introduced direction cosine variables c_1 and s_1 must satisfy the orthogonality condition $c_1^2 + s_1^2 = 1$. This condition is not solved as a separate nonlinear algebraic system. Instead, the equations for $\dot{\phi}_2$ and $\dot{\phi}_3$ are obtained from the first derivative of Equation 8, while the derivative of R_1 is augmented by a correction term, namely

$$\dot{\Phi} = \dot{R}_2^T \left(r_2 + R_3^T r_3\right) + R_2^T \dot{R}_3^T r_3 + \dot{R}_1 r_4 = 0, \qquad (10)$$

where

$$\dot{R}_1 = \left(\begin{bmatrix} 0 & \dot{\phi}_1 \\ -\dot{\phi}_1 & 0 \end{bmatrix} + k\varepsilon_1 I \right) R_1, \qquad \varepsilon_1 = 1 - c_1^2 - s_1^2. \qquad (11)$$

and

$$\dot{R}_i = \begin{bmatrix} 0 & \dot{\phi}_i \\ -\dot{\phi}_i & 0 \end{bmatrix} R_i, \qquad i = 2, 3. \qquad (12)$$

At the velocity level, Equation 10 keeps essentially the same coordinate-partitioning structure as in the traditional method. One of the angular rates $\dot{\phi}_i$ is still selected as the integrated variable, while the remaining two are obtained by solving a linear system derived from the differentiated loop constraint, i.e., in this case Equation 10. Compared with the conventional velocity constraint equation, the only structural difference here is the additional error-suppression term embedded in $\dot{R}_1$.

3.3.2 Spatial mechanism

A spatial four-bar mechanism is shown in Figure 2.

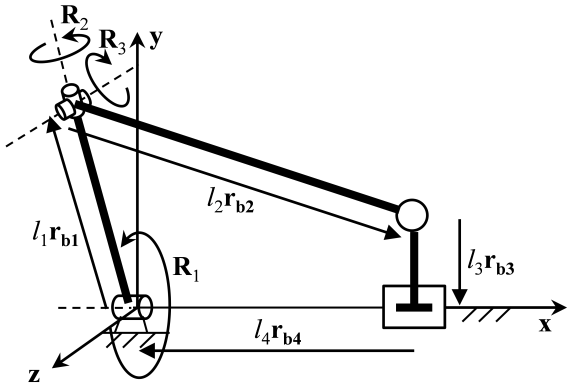


Figure 2. Spatial four-bar mechanism.

A representative position constraint for this mechanism can be written as

$$\Psi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & C_{11} & C_{12} \\ 0 & -C_{12} & C_{11} \end{bmatrix} \left(l_3 \begin{Bmatrix} x_{b3} \\ y_{b3} \\ z_{b3} \end{Bmatrix} + l_4 \begin{Bmatrix} 1 \\ 0 \\ 0 \end{Bmatrix} \right) + l_1 \begin{Bmatrix} x_{b1} \\ y_{b1} \\ z_{b1} \end{Bmatrix} + l_2 \begin{bmatrix} c_{\theta_2} & 0 & -s_{\theta_2} \\ 0 & 1 & 0 \\ s_{\theta_2} & 0 & c_{\theta_2} \end{bmatrix} \begin{bmatrix} c_{\theta_3} & -s_{\theta_3} & 0 \\ s_{\theta_3} & c_{\theta_3} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x_{b2} \\ y_{b2} \\ z_{b2} \end{Bmatrix}. \quad (13)$$

where l_i are geometric parameters, θ_2 and θ_3 are state variables, and C_{11}, C_{12}, and l_4 are the unknowns of the constraint equations. Thus, Jacobian matrix is given by

$$\Psi_{C_{11},C_{12},l_4} = \begin{bmatrix} 0 & 0 & 1 \\ l_3 y_{b3} & l_3 z_{b3} & 0 \\ l_3 z_{b3} & -l_3 y_{b3} & 0 \end{bmatrix}, \quad (14)$$

which is also constant and nonsingular. By differentiating Equation 13 with respect to time, we obtain the velocity constraint equations, where the derivative of R_1 with the new coordinates is given in the same form as Equation 3. For space reasons, the velocity constraint equations are not presented here.

3.3.3 General formulation and key points

For a more general spatial closed-loop mechanism with Nh closed loops, the position constraint equation for each closed loop can be written in the form

$$\Phi^h = \sum_{i=1}^{N_h} \left(\left(\prod_{j=1}^{i-1} R_j^T \right) s_i u_{b_{i-1}} + \left(\prod_{j=1}^{i} R_j^T \right) r_{b_i} \right) = 0, \quad (15)$$

where r_{b_i} denotes the length vector of body i resolved in its body-fixed frame; $u_{b_{i-1}}$ is the unit axis of a prismatic joint between body $i-1$ and body i, defined in the body-fixed frame of body $i-1$; and the relative coordinate s_i gives the sliding distance of that joint. If there is no relative translation between two connected bodies, then $s_i = 0$. The quantities r_{b_i} and $u_{b_{i-1}}$ are known parameters. Furthermore, $R_i \in \mathbb{R}^{3\times3}$ denotes the relative rotation from body $i-1$ to body i. If no rotational degree of freedom is present, R_i is constant. Otherwise it is expressed in terms of angular coordinates in the normal relative-coordinates method or direction cosine coordinates in the method proposed here.

The key point is to choose as unknowns the relevant direction-cosine variables together with the translational variables (s_i) in Equation 15, so that the position equations become linear in those unknowns. The orthogonality conditions of the introduced direction cosine variables are again not solved as a separate nonlinear algebraic system, but stabilized in the derivative equations by introducing correction terms given by Equation 4. Formulations for revolute joints with arbitrary axis directions, along with revolute in x/y/z direction will be discussed in the next Section. More details including formulations for universal joints and spherical joints can be found in (Zhao 2017).

3.4 Constant Jacobian in Favorable Cases

As shown in the planar four-bar example, when solving the position-level constraint equations (Equation 8), the Jacobian of the linear system is only related to the parameter r_4, which does not vary during the simulation. Therefore, the position-level constraint equations become a linear system with constant coefficients, so they can be reduced to direct assignments without any algebraic solve. No further dynamic switching of dummy states at the position level is required during the simulation.

Note that we are free to choose which angle variables to replace by direction cosine coordinates. Thus in many cases we can select the angles to be replaced so that the resulting position-level constraint equations have constant coefficients. This is a significant advantage because it removes a major source of numerical complexity and robustness issues in dynamic state selection. The method therefore has the potential to improve both computational efficiency and robustness for certain classes of mechanisms.

4 Modelica Implementation

We illustrate the Modelica implementation of the proposed method with two actual revolute-joint implementations. The first one is a general-axis version that uses the full 3×3 direction cosine matrix, whereas the second one is a simplified implementation for a revolute joint about the fixed z axis. In many practical cases the second implementation is sufficient and computationally cheaper, but the first one is more general and can be used for an arbitrary joint axis. In both implementations, the physical joint angle ϕ is not present, but the relative rotation appearing in the position-level equations is represented by direction cosine variables. If the joint angle is used by another component, then this joint might not be considered for linearization, or the angle can be reconstructed from the direction cosine variables by an atan2 function.

Listing 1. Pseudo code of Modelica model of a general-axis revolute joint with direction cosine coordinates

```
model RevoluteE
  parameter definition of stateSelect, n,
      en; // direction
  variable definition of p, q, r, tau, w, z
      ;

  Real T[3,3](start =
      {{1,0,0},{0,1,0},{0,0,1}}, each
      stateSelect = stateSelect);

  // Normalization correction terms
  Real Lxx, Lyy, Lzz;
  // Orthogonality correction terms
  Real Vxy, Vyz, Vzx;

  Modelica.Mechanics.MultiBody.Frames.
      Orientation R_rel;
equation
  ...
  R_rel.T = T;
```


DOI
10.3384/ecp22393

```
  {p,q,r} = R_rel.w;
  {p,q,r} = en*w;

  der(w) = z;
  der(T)=({{0,r,-q},{-r,0,p},{q,-p,0}}-d*{{
      Lxx,Vxy,Vzx},{Vxy,Lyy,Vyz},{Vzx,Vyz,
      Lzz}})*T;

  // Normalization correction terms
  Lxx =  (T[1,1]^2 + T[1,2]^2 + T[1,3]^2) -
       1;
  Lyy =  (T[2,1]^2 + T[2,2]^2 + T[2,3]^2) -
       1;
  Lzz =  (T[3,1]^2 + T[3,2]^2 + T[3,3]^2) -
       1;
  // Orthogonality correction terms
  Vxy = T[1,:] * T[2,:];
  Vyz = T[2,:] * T[3,:];
  Vzx = T[3,:] * T[1,:];

  if rooted(frame_a.R) then
    frame_b.R = Modelica.Mechanics.
        MultiBody.Frames.absoluteRotation(
        frame_a.R, R_rel);
    frame_a.f = -transpose(R_rel.T) *
        frame_b.f;
    frame_a.t = -transpose(R_rel.T) *
        frame_b.t;
  else
    ...
  end if;

  frame_b.t * en = -tau;
  zeros(3) = frame_b.r_0 - frame_a.r_0;
end RevoluteE;
```

Listing 1 shows the general-axis implementation. In this version, the full matrix T is treated as the direction cosine representation of the relative orientation, and both normalization and mutual orthogonality errors are corrected in the differential equations through the quantities Lxx, Lyy, Lzz, Vxy, Vyz, and Vzx. This form is applicable to an arbitrary revolute-axis direction and therefore matches the general spatial formulation discussed in the previous section.

However, in many cases the revolute axis is fixed to the x, y, or z axis of the joint frames, so that only two direction cosine variables are needed to represent the relative orientation. In this case, the position-level constraint equations become linear in those two variables, and only one normalization error variable is needed for stabilization. A simplified implementation for a revolute joint about the z axis is shown in Listing 2. Here the orientation is represented by the two-component variable rx, from which the planar rotation matrix T is constructed directly. Only one normalization error variable, Lx, is needed, so the implementation is considerably simpler than the general-axis version.

Listing 2. Pseudo code of Modelica model of a z-axis revolute joint with direction cosine coordinates

```
model RevoluteZ
  definitions of parameters and variables;
  Real Lx;
initial equation
  rx = {cos(phi_0),sin(phi_0)};
equation
  ...
  vx = der(rx);

  // much simpler than the general-axis
      version
  T[1,:] = rx;
  T[2,:] = {-rx[2], rx[1]};

  R_rel.T[1:2,1:2] = T;
  R_rel.T[1:2,3] = {0,0};
  R_rel.T[3,:] = {0,0,1};
  R_rel.w = {0,0,w};

  Lx = (rx[1]^2 + rx[2]^2) - 1;
  vx = transpose(T)*({0,w} + d*{Lx,0});

  if rooted(frame_a.R) then
    frame_b.R = Modelica.Mechanics.
        MultiBody.Frames.absoluteRotation(
        frame_a.R, R_rel);
    frame_a.f = -(R_rel.T) * frame_b.f;
    frame_a.t = -(R_rel.T) * frame_b.t;
  else
    ...
  end if;

  der(w) = z;
  frame_b.t * {0,0,1} = -tau;
  zeros(3) = frame_b.r_0 - frame_a.r_0;
end RevoluteZ;
```

In both cases, the standard joint variables, connector equations, and balance equations are retained; the main modification is the replacement of the direct trigonometric representation by direction cosine variables together with correction terms in the differential equations.

5 Numerical Experiments

5.1 Test Case

The numerical evaluation is carried out with a planar Stephenson mechanism, since it is sufficiently small to expose the algebraic structure clearly while still containing multiple coupled loops and several dependent angular coordinates. This makes it a suitable test case for comparing the conventional relative-coordinate formulation and the proposed linearized formulation.

The mechanism is shown in Figure 3.

The model parameters are summarized in Table 1. The same physical parameters, consistent initial conditions, and solver tolerances are used for the conventional and the proposed formulations.

5.2 Energy and Stability

The Stephenson mechanism is used here as a Hamiltonian-type test problem without physical damping. In such a system, numerical energy variation is a useful indicator of both stability and integration bias. Explicit methods tend

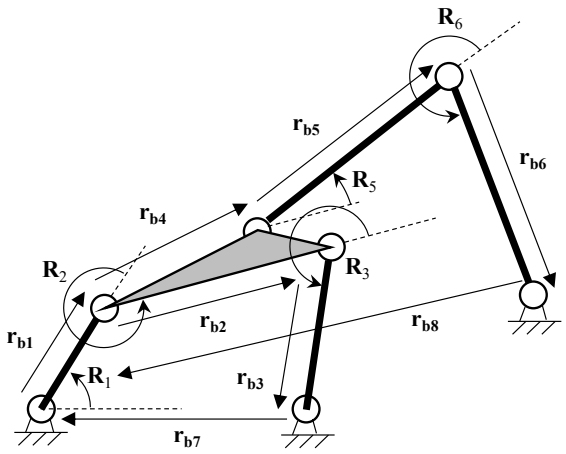


Figure 3. Test case: Stephenson mechanism.

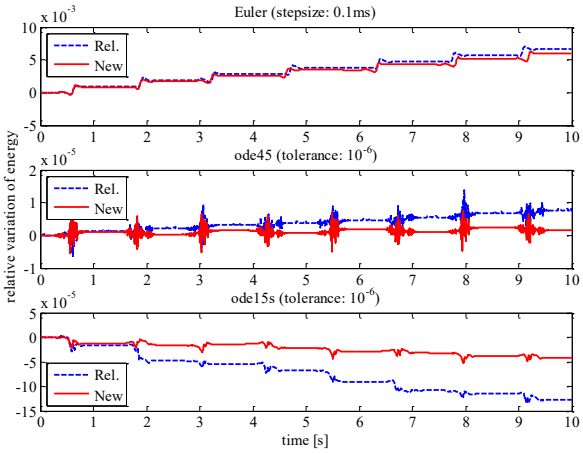


Figure 4. Relative energy variation of the Stephenson mechanism for the conventional and linearized formulations.

Table 1. Parameters of the Stephenson mechanism.

Quantity	Value
r_{b1}	$\{0.1, 0.3\}^T$
r_{b2}	$\{0.8, 0.3\}^T$
r_{b3}	$\{0.1, -0.6\}^T$
r_{b4}	$\{0.6, 0.4\}^T$
r_{b5}	$\{0.7, 0.5\}^T$
r_{b6}	$\{0.4, -0.7\}^T$
r_{b7}	$\{-1, 0\}^T$
r_{b8}	$\{-1.8, -0.5\}^T$
m_1	5 kg
m_{2-8}	1 kg each
Centers of mass	Geometric centers of all links
J_{1-8}	0.001 kg m^2 about the center of mass of each link
g	9.81 m/s^2 in the negative y direction

to inject energy, whereas implicit methods tend to dissipate it. The proposed method introduces additional first-order feedback only through the orthogonality-error correction terms, so the main question is whether this changes the stability characteristics of the conventional relative-coordinate formulation.

The comparison is performed with three representative solvers: explicit Euler with a fixed step size of 0.1 ms, and the variable-step solvers `ode45` and `ode15s` with tolerances of 10^{-6}. In the conventional formulation, the nonlinear position equations are solved by `fsolve` with the same convergence tolerance. The relative energy variation is shown in Figure 4, where "Rel." denotes the conventional relative-coordinate formulation and "New" denotes the proposed linearized formulation.

For the explicit solvers Euler and `ode45`, both formulations show a slow increase of the total mechanical energy, which is the expected behavior for explicit integration of a Hamiltonian-type system. For the implicit solver `ode15s`, both formulations show a slow decrease of energy, again consistent with the dissipative tendency of implicit integration. More importantly, the new formulation does not introduce a new instability mechanism. Its energy drift remains of the same type as in the conventional formulation and is often slightly smaller in magnitude. In this sense, the linearized formulation has essentially the same stability behavior as the conventional relative-coordinate formulation, while in this test case staying somewhat closer to energy conservation.

5.3 Accuracy and Error

The conventional and the proposed methods do not produce exactly the same type of numerical error. In the conventional formulation, the dominant position-level error source is the convergence error of the nonlinear algebraic solve. In the proposed formulation, this error is replaced by controlled violations of the orthogonality conditions of the introduced direction cosine variables. These two error sources are different in nature, so they should not be compared only through raw constraint residuals.

For this reason, two complementary measures are used. The first is the global drift of total mechanical energy, already shown in Figure 4, which reflects the accumulated numerical error of the full simulation. The second is the orthogonality error itself, for example the deviation of $R_i^T R_i$ from the identity matrix or, in the planar case, the deviation of $c_i^2 + s_i^2$ from unity. A representative result for the Stephenson mechanism is shown in Figure 5. The orthogonality errors are suppressed to a small range and exhibit the same periodic character as the mechanism motion.

When the integration accuracy is increased, for example by reducing the Euler step size to 0.01 ms and tightening the variable-step solver tolerances to 10^{-9}, the orthogonality errors decrease correspondingly. This is consistent with the energy-drift results and indicates that the linearized formulation achieves an accuracy level comparable to that of the conventional method, while shifting the dominant numerical error from nonlinear-iteration error to controlled orthogonality violation.

DOI
10.3384/ecp22393

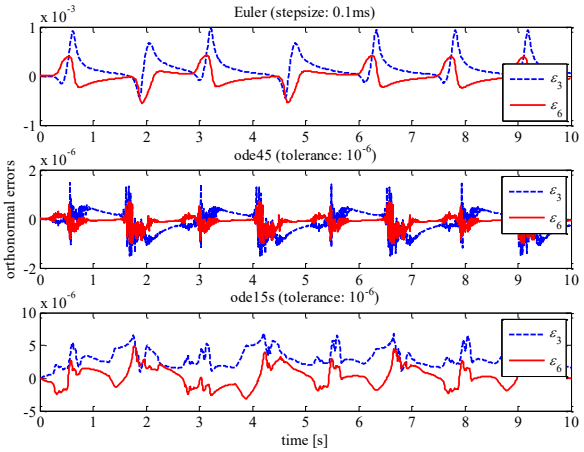


Figure 5. Orthogonality errors of the Stephenson mechanism under the baseline solver settings.

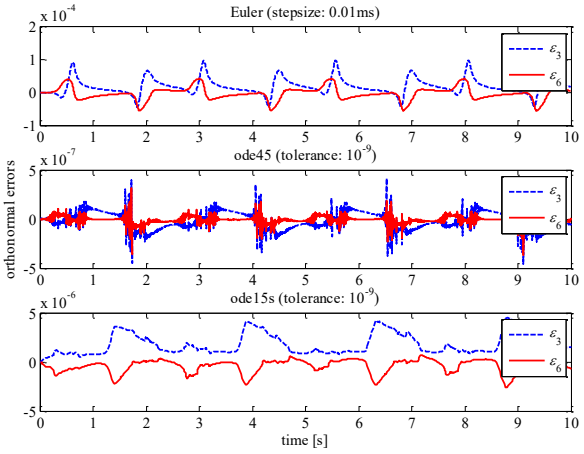


Figure 6. Orthogonality errors after tightening the integration accuracy settings.

To further test robustness, a large artificial initial orthogonality error can be imposed and the correction gain k can be varied. The result is shown in Figure 7. Even from a clearly inconsistent initial state, the simulation remains stable and the orthogonality error is driven downward by the first-order feedback correction. A larger k suppresses the error more rapidly, whereas a smaller k yields slower decay but still convergent behavior. This demonstrates that the method remains stable under substantial initial violation and that the gain mainly influences the speed of error suppression rather than whether the simulation converges. In this sense, the present method is much less sensitive to parameter tuning than Baumgarte-type approaches and related EKF-based variants that were introduced largely to avoid manual tuning of Baumgarte parameters. For the present formulation, keeping $k = 1$ as the default is a good choice and should be suitable for most practical cases. Note that this comparatively large error is introduced deliberately only to stress-test stability; in normal simulations the orthogonality error remains confined to a small range unless the user also provides inconsistent initial conditions.

Additional examples and a broader discussion of stability-related behavior can be found in the doctoral thesis of Zhao (Zhao 2017), which includes further case studies beyond the representative results reported here.

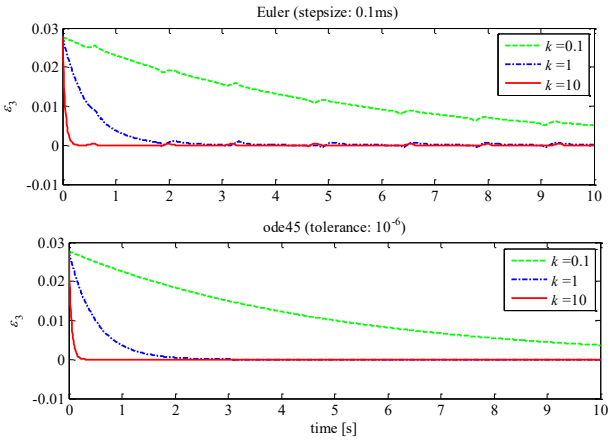


Figure 7. Decay of a large imposed orthogonality error for different values of the gain parameter k.

5.4 Speed

From the viewpoint of computational cost per integration step, key points of the proposed formulation are as follows. First, the use of direction cosine coordinates introduces a small number of additional state variables; in some cases, this number can exceed the number of degrees of freedom, so the cost of numerical integration increases slightly. Second, the acceleration-level equations remain essentially the same as in the conventional relative-coordinate formulation. Third, the modified velocity constraint equations contain orthogonality-correction terms, but these terms are obtained from simple algebraic expressions and add only modest extra work. Fourth, and most importantly, the nonlinear position-level solve is removed and replaced by a linear solve, or in favorable cases by direct assignments. In practice, this last change dominates the cost balance, so a single integration step is generally cheaper with the linearized formulation.

When comparing overall simulation speed, however, one must also consider the number of integration steps required to reach a given accuracy. For variable-step solvers, the gain parameter k influences the tradeoff between orthogonality control and step size: increasing k usually improves accuracy but can also force smaller steps. Therefore, the appropriate comparison is not simply raw runtime under arbitrary settings, but CPU time under settings that provide comparable solution accuracy. For the Stephenson mechanism, the experiments in the previous subsections indicate that the conventional and linearized formulations have similar accuracy when $k = 1$, so this value is used for the speed comparison.

The measured CPU times for the Stephenson mechanism are listed in Table 2. In all three tested solvers, the linearized formulation is faster than the conventional formulation. The improvement is moderate rather than dramatic because the reported time is the total simulation time, and most of that time is still spent in solving the dynamic equilibrium equations, which are nearly identical in the two formulations. Even so, the removal of the nonlinear position solve yields a consistent reduction in runtime.

Table 2. CPU time for Stephenson-mechanism simulations.

Integrator	Conventional [ms]	Linearized [ms]
Euler	5082.59	4595.56
ode45	135.76	118.37
ode15s	134.36	117.31

5.5 Comparison with Multibody Assemblies

The components in `Multibody.Joints.Assemblies` remove the full set of constraint equations between two frames and replace them by analytical constructions tailored to specific kinematic patterns. When a mechanism matches one of these predefined patterns, this approach can be very efficient.

By contrast, the method proposed in this paper remains a numerical method and does not rely on such analytical pattern-specific reductions. Therefore, when a mechanism can already be represented directly by a suitable component from `Multibody.Joints.Assemblies`, the present method should not be expected to be faster than that assembly-based solution. Its main advantage is different: it does not require the user to identify a special pattern or a prescribed combination of joints, and it can in principle be attempted for general closed-loop mechanisms.

The same perspective applies to issues such as the use of `RevolutePlanarLoopConstraint` and the limitations of fixed state selection. For the proposed method, these issues remain essentially unchanged. In most cases, the role of dynamic state selection also remains largely the same, because the method only replaces one set of algebraic unknowns by another while preserving the differentiated constraint equations and the position-level constraint equations, even though in favorable cases the latter may be reduced to direct assignments.

An interesting phenomenon was also observed. In the numerical experiment shown in Figure 7, a comparatively large constraint violation was deliberately injected into the system. Although this already violates the original constraint equations, the new formulation does not fail or reject simulation. Instead, it reduces the violation rapidly and robustly to a very small value. This suggests that the new method may permit a wider range of initialization conditions. It would therefore be worthwhile in future work to investigate whether this behavior can be exploited to alleviate initialization failures caused by overly strict consistency requirements.

6 Conclusions

This paper has presented a formulation for rigid multibody dynamics in which selected algebraic angle variables in closed-loop position constraints are replaced by direction cosine coordinates, while the remaining relative coordinates are kept as integrated states. In this way, the nonlinear position-level constraint equations can be transformed into linear equations. The introduced direction cosine variables still have to satisfy orthonormality conditions, but these conditions are not solved as an additional nonlinear algebraic subsystem; instead, their violations are suppressed in the derivative equations by Baumgarte-type correction terms.

The main benefits are that a major source of position-level nonlinearity is removed, a modest speed improvement can be obtained, and the numerical behavior remains stable, with orthogonality errors confined to a small range under normal conditions. The main limitation is that replacing angle variables by direction cosine variables slightly increases the number of states and may therefore slightly increase the integration cost. The speed gain is accordingly limited, since other expensive parts of the computation, such as the solution of the acceleration-level linear system for the generalized second derivatives, remain unchanged. The additional state variables may improve the treatment of mechanism singularities, but this aspect has not been investigated here.

For Modelica-oriented applications, the method fits naturally into an equation-based workflow: the modified equations can be formulated directly, and state selection can in principle be guided by the existing `stateSelect` mechanism. However, in most Modelica tools, the process of index-reduction cannot be controlled by the user, and the replacement of angle variables by direction cosine variables and the addition of error correction terms should be done manually by the modeler. That is not easy for complex models. Therefore, tool support for automatic direction-cosine substitution, variable selection, and equation generation would be a natural and valuable extension of the present work.

References

Baumgarte, J. (1972). “Stabilization of Constraints and Integrals of Motion in Dynamical Systems”. In: *Computer Methods in Applied Mechanics and Engineering* 1.1, pp. 1–16. DOI: 10.1016/0045-7825(72)90018-7.

Bettega, J., G. Boschetti, B. R. Frade, F. González, G. Piva, D. Richiedei, and A. Trevisani (2024). “Numerical and Experimental Investigation on the Synthesis of Extended Kalman Filters for Cable-Driven Parallel Robots Modeled through DAEs”. In: *Multibody System Dynamics* 60.2, pp. 161–190. DOI: 10.1007/s11044-023-09941-5.

Bortoff, Scott (2019). “Using Baumgarte’s Method for Index Reduction in Modelica”. In: *Proceedings of the 13th International Modelica Conference*. Regensburg, Germany. DOI: 10.3384/ecp19157333.

Flores, P., M. Machado, E. Seabra, and M. T. Silva (2011). "A Parametric Study on the Baumgarte Stabilization Method for Forward Dynamics of Constrained Multibody Systems". In: *Journal of Computational and Nonlinear Dynamics* 6.1, p. 011019. DOI: 10.1115/1.4002338.

Haug, Edward J. (1989). *Computer-Aided Kinematics and Dynamics of Mechanical Systems*. Boston: Allyn and Bacon.

Khadim, Q., L. Pyrh"onen, E. Kurvinen, J. Gerstmayr, A. Mikkola, and G. Orzechowski (2024). "Real-Time State Estimation of Hydraulically-Driven Systems Based on Unscented Kalman Filter and Low-Fidelity Models". In: *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. Vol. 88438, pp. 009–09006. DOI: 10.1115/DETC2024-144017.

Naya, M. A., E. Sanjurjo, A. J. Rodríguez, and J. Cuadrado (2023). "Kalman Filters Based on Multibody Models: Linking Simulation and Real World. A Comprehensive Review". In: *Multibody System Dynamics* 58.3, pp. 479–521. DOI: 10.1007/s11044-023-09893-w.

Neumayr, Andrea and Martin Otter (2024). "Combining Equation-based and Multibody Models". In: *Proceedings of the Asian Modelica Conference 2024*. Jeju, Korea. DOI: 10.3384/ecp21717.

Otter, Martin, Hilding Elmqvist, and Sven Erik Mattsson (2003). "The New Modelica MultiBody Library". In: *Proceedings of the 3rd International Modelica Conference*. Link"oping, Sweden, pp. 311–330. DOI: 10.3384/ecp03276311.

Palomba, I., D. Richiedei, and A. Trevisani (2017). "Kinematic State Estimation for Rigid-Link Multibody Systems by Means of Nonlinear Constraint Equations". In: *Multibody System Dynamics* 40.1, pp. 1–22. DOI: 10.1007/s11044-016-9515-x.

Phillips, Warren F., Charles E. Hailey, and George A. Gebert (2001). "Review of Attitude Representations Used for Aircraft Kinematics". In: *Journal of Aircraft* 38.4, pp. 718–737. DOI: 10.2514/2.2824.

Richiedei, Dario, Ivan Tamellin, and Andrea Trevisani (2025). *The Extended Kalman Filter for Constraint Enforcement in the Forward Dynamics of Multibody Systems*. DOI: 10.21203/rs.3.rs-6589073/v1.

Zhao, Yan (2017-04). "Research on Parallelization of Simulation and Computing of Complex Mechatronics Systems". PhD thesis. Wuhan, P.R. China: Huazhong University of Science and Technology.

DOI
10.3384/ecp22393

Learning the Missing Physics Locally with Modelica and Wolfram Language

Dr. Clément Coïc[1] Ankit Anurag Naik[2]

[1]Siemens Healthineers AG, Germany, clement.coic@siemens-healthineers.com

[2]Wolfram MathCore AB, Sweden, ankitn@wolfram.com

Abstract

This paper introduces State-Invariant Hybrid Models (SIHM), a subset of hybrid physics-machine-learning models in which the state vector remains defined by the first-principles Modelica model, while learned components contribute missing physics locally. We qualitatively contrast global residual learning at system level with local residual learning at component level, arguing that local learning better matches the acausal modeling paradigm of Modelica. A tool-agnostic workflow is proposed in which a replaceable dummy component marks the suspected location of missing physics, while compiler-derived structural information identifies the connected states and variables. The workflow is then illustrated in the Wolfram ecosystem on two models to demonstrate the approach's modularity. The results indicate that local SIHM components recover targeted missing effects while preserving the state selection of the original model.

Keywords: Physics Informed Machine Learning, Universal Differential Equations, Physics-enhanced Neural-ODEs

1 Introduction

Scientific Machine Learning is the field that combines scientific computing with Machine Learning (ML). ML itself is mostly out of scope for the Modelica language. However, Modelica is well positioned to contribute to this field in two main ways: either as a ground-truth model that can be leveraged for training ML models, e.g. Neural Networks, Physics Informed Neural Networks (PINN; Raissi et al. 2017), NeuralODE (Chen et al. 2018), or as a base model that can be enhanced with ML, e.g. Physics-enhanced NeuralODEs (PeN-ODE; Hofmann and Mikelsons 2025) or a subset of the Universal Differential Equations (UDE; Rackauckas et al. 2020). The former is typically simpler because it is a one-way interaction: the Modelica model "feeds" the ML model with insights, e.g. data or derivatives. The latter, referred to as hybrid modeling, is more involved as it is a two-way interaction: the ML model is trained based on some dependence on the Modelica model, and then the ML model shall be integrated with the Modelica model.

1.1 A call for proactive interfacing

The Modelica language excels in the behavioral modeling of physical systems, for dynamic and steady state simulation as well as for optimization. One of the key enhancements that Modelica brought was acausality as a modeling paradigm: models are described by the equations of the physics that represent them and the solving of these equations is left to the compiler, based on the system boundary conditions. This paradigm shift, associated with the object-oriented nature of the language, allowed for creating component-based libraries of models that are reusable, and reduced the use of sequential algorithmic programming to some limited use cases. Acausality strengthened even more this model reusability by allowing a model to serve multiple purposes: from being connected to different boundary conditions, to automatically lumping components together, or even to unlock solving for parameter sizing.

While Modelica shines in the behavioral modeling capabilities, the associated engineering workflows around models are typically implemented in a second language, dedicated to programming, such as Python, Julia or Wolfram Language. This is justified by the different needs for scripting and behavioral modeling, and therefore there is no reason nor effort to add these capabilities into the Modelica language. Relying on a second language is the right choice, and the choice of this scripting language should be left free.

However, Modelica cannot afford to passively wait for other communities to interface with their models. Modelica should proactively guide these communities on how to make the best use of their models for their workflows.

This paper adopts this mindset and presents a tool-agnostic workflow, that leverages Modelica strengths, before showing a specific implementation in a chosen language, here Wolfram Language.

1.2 Recent interfacing efforts

The Modelica community initially focused strongly on the sole, yet complex, scope of multi-physics behavioral modeling. The challenges are extremely diverse and non-trivial: large scale models, sparse matrices, steady-state solving, higher discretization, variable structure, etc.

In recent years, however, more efforts have been oriented towards workflows that unlock new capabilities. It is

worth mentioning that several tools developed APIs with Python or similar languages, though proprietary developments are not necessarily generalizable to the Modelica language. Another notable example is the development of the FMI standard (Modelica Association 2024), which enabled model exchange and co-simulation through a standardized container and interface. However, an FMU is typically used as an exported representation of a model for exchange, simulation, or co-simulation, rather than as a mechanism for modifying the source Modelica model itself.

A most recent example was triggered by the recurring request to build Reduced Order Models from (part of) Modelica models, using Neural Networks. Once the network is trained, it is often required to bring it back to the Modelica environment. Several approaches have been developed: from using a Modelica library for neural networks and porting the weights and biases only (Hanke et al. 2025), to building ONNX import capability (Urbani et al. 2025; ONNX project 2026).

Kamp et al. (2025) also combine Modelica and Neural Networks, either through ONNX or using the NeuralNetwork Modelica library, while leveraging the eFMI standard for bringing the hybrid model to the embedded domain.

Nevertheless, in many cases, workflows aren't that simply built. Hofmann and Mikelsons (2025), for example, showed the current challenges and limitations of using a Modelica model for training PeN-ODEs in Julia. Thummerer et al. (2025) demonstrated Neural FMUs for hybrid modeling of automotive braking systems, further illustrating the growing interest in combining Modelica-based models with machine learning. So, the community efforts shall be continued in this direction.

1.3 Focus of this contribution

Hybrid modeling and PeN-ODEs are broad and generic definitions that encompass many variants of the combination of first principles and ML models. For engineers with a strong Modelica background, the subset of interest within PeN-ODEs is the one that relies on a good first-principles model (FPM) that only misses minor physical effects, potentially already identified and intentionally left unmodeled. Said differently, let's consider a mechatronic system that shall be modeled with the Modelica language. The kinematics, the drives, the inertias might be all known. Yet, there is some friction in part of the system that cannot be simply modeled with a Coulomb or Stribeck formula, and that is known to influence the overall system dynamics. Additionally, this friction is temperature dependent. This would represent a high effort to study and identify the friction correlation for this system. Accordingly, the engineering judgment is to let this friction unmodeled and learn it with an ML model that is to be combined with the Modelica model.

The following definition is introduced:

State-Invariant Hybrid Model (SIHM): a hybrid physics-machine-learning dynamical model in which the state vector is entirely defined by the first-principles model. The machine-learning component may learn residual, missing, or difficult-to-model effects, but it does not introduce additional states nor alter the state selection of the underlying physical model.

This is a meaningful scope as models that should remain mostly explainable, so the modifications are minor, and usually the modeler (or library developer) put some effort in ensuring the right states were selected. The ML model shall thus not modify the state vector. This framing is related to grey-box system identification, where prior physical structure is retained while uncertain components or parameters are inferred from data (Ljung 1999).

With this subset of PeN-ODEs in mind, this work focuses on two main points:

1. Discussing the somewhat philosophical difference between learning globally and locally the missing physics in a model.
2. Presenting the value of using Modelica acausal and component-based capabilities for hybrid models and suggesting an associated workflow.

The first point will be covered in section 2, while the second is addressed in a tool-agnostic manner in section 3, and implemented within the Wolfram ecosystem in section 4. Section 5 then extends the workflow to a multi-local example, where two missing effects are learned simultaneously.

Two examples are used: a coffee mug, for its pedagogic value in the conceptual discussion, and a slightly modified version of the `ElasticBearing` from the Rotational examples of the MSL, which carries the implementation as a more realistic engineering problem.

2 Where the missing physics can be learned

The meaning of "where" in this section's title is numerical and could be rephrased as which states in the state vector can be used to learn the missing physics. Still, the much-appreciated component-based modeling capability of Modelica hides the numerical aspects and rather positions the question into an "architectural" standpoint. Where is the missing physics connected?

A coffee mug example, introduced in the Modelica & FMI Newsletter (Coïc 2025), is leveraged here to illustrate the discussion. It is a simple, visual model that is understandable by most, and hence a great support for reflection.

The model focuses solely on the thermal storage of the beverage, and heat transfer through the mug to the hands that carry it, to the surrounding air and optionally to a spoon in the coffee. Why is the spoon optional? Because

DOI
10.3384/ecp223103

it is the part of the physics that we want to learn. So, the spoon is modeled to generate the "synthetic data", and removed from the FPM to be learned by the ML model.

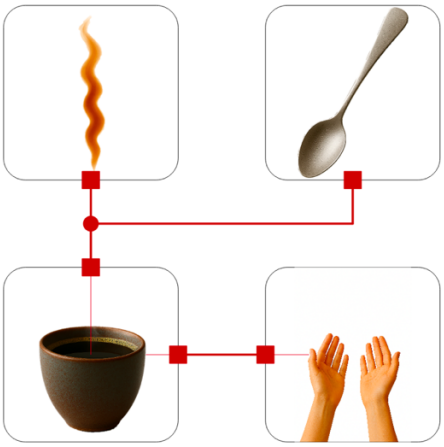


Figure 1. Coffee-mug example used for the global/local learning discussion: the spoon (top right) is treated as the missing physics to be learned.

Yet, where do we learn the spoon physics? Globally or locally? Let's investigate both options.

2.1 Global learning

Modelica models are Ordinary Differential Equations (ODEs) or Differential Algebraic Equations (DAEs) mostly. They can be represented as:

$$x_{\text{dot}} = f_{\text{FPM}}(x, u, t) \quad (1)$$
$$0 = g_{\text{FPM}}(x, u, t) \quad (2)$$

Where x is the state vector, x_{dot} its piecewise derivative, u is the input vector and t is time. The function f_{FPM} defines the ODEs, while g_{FPM} defines the algebraic relations, when they exist. The subscript $_{\text{FPM}}$ is consistent with the notation from Hofmann and Mikelsons (2025) and allows to separate this contribution from the upcoming ML model one.

As a reminder, this paper focuses on SIHMs. In this setting, the ML component must not introduce additional independent states, nor modify the state selection made by the first-principle Modelica model. It may, however, contribute to the differential or algebraic residuals that determine the time evolution of these existing states. The state vector x, the independent time variable t, and the external inputs u therefore remain defined by the FPM, while the ML model contributes through additional terms such as f_{ML} or g_{ML}. While there is no restriction on how these contribute to the PeN-ODE representation (f_{ML} could potentially be composed with f_{FPM}, for example), the notation is here simplified with a simple summation:

$$x_{\text{dot}} = f_{\text{FPM}}(x, u, t) + f_{\text{ML}}(x, u, t) \quad (3)$$
$$0 = g_{\text{FPM}}(x, u, t) + g_{\text{ML}}(x, u, t) \quad (4)$$

Learning the missing physics globally for a SIHM thus means that the entire state vector x of the FPM / Modelica model is fed as input, directly or indirectly (e.g. for the composition example), to the ML model to be trained, and the outputs are contributing to all state derivatives x_{dot}.

Said differently, when provided with data, the ML model will try to compensate for the inaccuracies of the FPM by potentially influencing all the relationships between the state vector and its derivatives, as defined by the FPM. In the case of the coffee mug, should the spoon not be modeled but the data include its effects, then ML model might learn it. Additionally, if the data includes some minor forced convection effect on the free surface of the coffee, because the person holding it was walking around with their cup, then this would be also learned by the ML model, because the data show it and the state variables are accessible. This constitutes the main difference with learning locally.

2.2 Local learning

When learning missing physics locally, the engineer makes a judgment on the location where the physics needs to be learned. Accordingly, only the related subset of states x_{ML} from the state vector x and subset of inputs u_{ML} from the input vector u can be used by the ML model during its training.

$$x_{\text{dot}} = f_{\text{FPM}}(x, u, t) + f_{\text{ML}}(x_{ML}, u_{ML}, t) \quad (5)$$
$$0 = g_{\text{FPM}}(x, u, t) + g_{\text{ML}}(x_{ML}, u_{ML}, t) \quad (6)$$

With $x_{ML} \subset x$ and $u_{ML} \subset u$.

And in practice, f_{ML} might not contribute to the entire state vector derivatives x_{dot}, and rather to the related subset of its location.

Coming back to the coffee mug example, a local model placed at the spoon interface is structurally constrained to use only the variables available at that interface and to act only through the associated connector variables. It is therefore less able than a global residual model to compensate for arbitrary errors elsewhere in the system, such as a missing free-surface convection term. This does not prove that the learned term is uniquely the spoon physics: correlated effects can still be absorbed by the local residual. Rather, locality should be understood as an architectural constraint imposed by the modeler, which limits where the learned correction can sense and act.

Naturally, the next step in the reflection is the multi-local learning, where several ML models can be located in a Modelica model to learn simultaneously different missing physics. This is reflected on the mathematical formulation by simply adding multiple *f_ML_i()* and *g_ML_i()* functions that each have their own subsets *x_ML_i* and *u_ML_i*. .

$$x_{\mathrm{dot}} = f_{\mathrm{FPM}}(x,u,t) + \sum_{i}^{n} f_{\mathrm{ML_i}}\left(x_{ML_i}, u_{ML_i}, t\right) \quad (7)$$

$$0 = g_{\mathrm{FPM}}(x,u,t) + \sum_{i}^{n} g_{\mathrm{ML_i}}\left(x_{ML_i}, u_{ML_i}, t\right) \quad (8)$$

With $\forall i \in [0,n],\ x_{ML_i} \subset x$ and $u_{ML_i} \subset u$.

2.3 Why is local learning preferred here

Local learning is preferred here because it matches the engineering workflow of a Modelica modeler. The suspected missing physics is represented as a balanced component connected at a chosen location, so the learned correction has an explicit architectural meaning. This does not guarantee physical identifiability, but it improves traceability: the engineer can inspect where the correction is placed, which variables it uses, and through which connector variables it affects the system.

2.4 A philosophical note

It can seem a bit naive to say that we learn only the missing physics locally, while the data we learn from includes all the inaccuracies of the FPM. And to some extent it is. The optimization algorithm that trains the ML model does not know whether a given part of the error originates from the selected physical location. It only minimizes the chosen loss. The meaning of local learning is therefore structural rather than semantic: only a subset of the state and input vectors is provided as inputs to the ML model, and its outputs affect the system only through the variables available at the selected location. The analogy with system controllability appears when one sees the ML model as an external correction with limited sensing and actuation points. The relevant question is then whether these added correction points can move the observed states toward the data while preserving the state structure of the FPM.
This caveat is consistent with classical structural-identifiability limits: different parameterizations or residual structures can reproduce the same measured trajectories (Bellman and Åström 1970).
In the next section, a workflow is suggested to let the ML model know the subset of space it can influence.

3 Suggested tool-agnostic workflow

To achieve the above-mentioned learning, the following steps need to be clarified:

Q1. How to define the location where the ML model can contribute?

Q2. How can the ML model relate this location to the subset of states and inputs it can leverage?

Q3. How to train the ML model accordingly?

Q4. How to instantiate the trained model in the right location of the SIHM?

The first two questions make the bulk of this section. They are to be answered jointly as they influence each other. The third and fourth questions are obviously very meaningful and yet very dependent on the technology chosen for the ML model and the language for its training. Therefore, the two last questions are loosely addressed without making strong recommendations.

3.1 Targeting part of the model

As previously stated, two of the strengths of the Modelica language are its acausality and object orientation, allowing for building models with a component-based approach. The most meaningful way to give the model developer a tool to mark where they believe there is missing physics to learn (Q1) is by adding a dedicated component model locally and connecting it to the expected interfaces. This is also the most natural way and recent papers on the topic suggested or implemented this approach, though they differed on their approaches. Hofmann and Mikelsons (2025) suggested using a component in the model that includes dummy functions to be replaced by the ML architecture. While this approach is near our suggestion, it has the drawback that the user must manually identify the relevant causal interfaces of the ML model. Langenkamp et al. (2025) suggested keeping the ML model in Modelica and leveraging collocation optimization for tuning the weights and biases. This solution keeps the entire Hybrid Model in the Modelica world: this allows for leveraging Modelica's strengths but does not necessarily leverage the strengths of a dedicated ML programming ecosystem.
The contribution of the present workflow is not the use of a neural component in a Modelica model *per se*; this has already been explored in Modelica-based PeN-ODE workflows (Hofmann and Mikelsons, 2025) and in collocation-based training of PeN-ODEs (Langenkamp et al., 2025). The specific addition proposed here is to use the inserted dummy component as a structural marker from which the compiler-generated model representation can be queried to derive the local state and variable set available to the learned component. In this way, the causal ML interface is not selected manually but inferred from the acausal component placement and the resulting flattened model structure (Q2).
Both approaches fit the typical modeling engineer workflow by allowing them to stay in their environment, deal with Modelica models, and initiate some SIHM activities. Therefore, the generic recommendation shall:

- be component-based, allowing us to identify where the missing physics is in the Modelica environment.
- automatically (using the Modelica compiler) identify the needed causal interfaces of the ML model.

On top and to answer Q3 and Q4, the recommendation shall:

DOI
10.3384/ecp223103

- allow for any training method: from a collocation method suggested by Langenkamp et al. (2025) to an external training method as in Hofmann and Mikelsons (2025).
- allow for simple integration of the ML model inside the Modelica model.

To summarize, the recommendation here is thus to leverage a replaceable dummy model that has the right domain interfaces and is balanced with trivial equations. The apparent drawback of this approach is that it requires such a dummy model per interface combination type, though this wasn't different in the previous approach. Nevertheless, this small drawback allows the Modelica compiler to process the Modelica model, and it is suggested to leverage the strength of the Modelica compiler to identify the connected states to the dummy model. As a reminder, the base definition of SIHM being that the ML model does not change the states in the Modelica model. From the dummy model connectors, it is possible to trace back, with the Modelica model graph representation or the Block Lower Triangular (BLT) form for example, the connected states. The variables connected to these states will be the inputs to our ML model while complementary connector variables shall be the outputs.
The minimal compiler interface required by this workflow is: access to the flattened model variables and equations, a mapping from component connector variables to flattened variables, the selected state set after compilation, and structural dependency information such as graph or BLT relationships. The Wolfram implementation queries this information through author-written functions, while other Modelica tools could expose the same information through their own compiler APIs.
This answers the first two questions above: how to define the location where the ML model can contribute and how can the ML model relate this location to the subset of states and inputs it can leverage?

3.2 Training and re-instantiating

Training the ML model within Modelica is a viable option, while leveraging dedicated programming languages and environments, with dedicated packages, is another obvious solution. Therefore, there is no unique training solution and the recommendation here can only be generic. Fortunately, the replaceable dummy model suggested above allows for this flexibility.
However, a recommendation we can make (a direct consequence of our training needs) is to make the model, as much as possible, differentiable. For most gradient-based training techniques, the optimization of the weights and biases of the ML model requires differentiating the SIHM response with respect to these trainable parameters. On a side note, nothing enforces per se the technology used for the data-driven model used in combination with the Modelica model. Alternatives to Neural Networks, such as Kolmogorov-Arnold Networks (KANs) (Liu et al. 2024) or sparse symbolic models (Cranmer 2023), can also be used in place of this replaceable model. Sparse identification of nonlinear dynamics provides another related route for discovering compact governing equations from data, although SIHM keeps the learned correction embedded in the original component-based Modelica structure (Brunton, Proctor, and Kutz 2016).

4 Implementation in the Wolfram ecosystem

For this implementation, a slightly modified version of the `Modelica.Mechanics.Rotational.Examples.ElasticBearing` model is used, where we added a damping on the spring connected to the load. Having two spring dampers allows for discovering multiple missing physics in a second step.

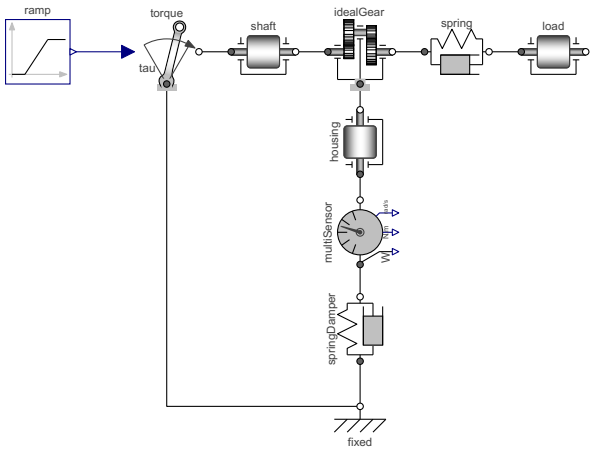


Figure 2. Reference ElasticBearing model. The spring connecting the gear to the load is augmented with an additional damping (d = 10 N·m·s/rad), used to generate the synthetic reference data.

4.1 Wolfram implementation environment

In this paper, Wolfram System Modeler in combination with the Wolfram Language is leveraged. Wolfram System Modeler is a commercial Modelica modeling and simulation tool with a graphical authoring interface, support for the Modelica Standard Library 4.1, and access to compiler-generated model information such as flattened equations and structural artifacts. Wolfram Language is a symbolic computing platform with notebook-based workflows and built-in facilities for equation manipulation, numerical simulation control, visualization, and optimization.
For the present workflow, the important property of the Wolfram ecosystem is the tight integration between Wolfram System Modeler and Wolfram Language. Modelica models can be compiled and simulated from notebooks, while the resulting model structure and

simulation data can be post-processed symbolically and numerically. In this implementation, the structured representation emitted by System Modeler is processed by the Wolfram Language functions `parseBlockdebug` and `componentToStateMap`, which map the inserted dummy component to the states and variables structurally connected to it. These two functions are not part of the standard Wolfram Language distribution; they were developed by the authors for the purpose of this work to query the compiler-generated model structure. The proposed workflow is therefore not specific to Wolfram in principle; an equivalent implementation would require comparable access to compiler-generated structural information.

4.2 Creating reference data

This first step is only necessary in this example as there is no "real" available data on the system. Therefore, synthetic data is here generated by simulating the model as is. This data will serve as source of truth for our system. Simulating the slightly modified version of `Modelica.Mechanics.Rotational.Examples.ElasticBearing` gives us this dataset.

For reproducibility, the reference model, FPM, and SIHM are simulated over the same time interval and with identical boundary conditions.

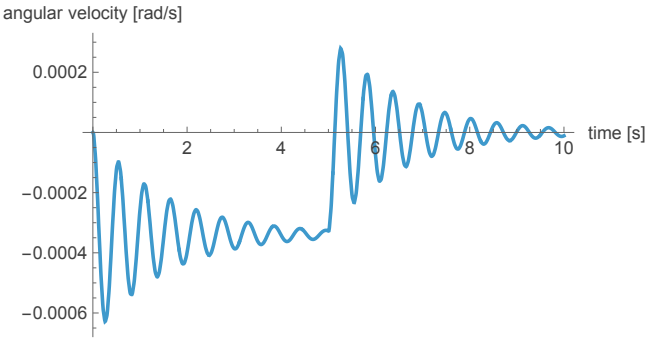


Figure 3. Angular velocity of the housing over time for the synthetic reference case: the response the SIHM is trained to reproduce.

4.3 The missing physics

It is here assumed that the modeler had carefully created `Modelica.Mechanics.Rotational.Examples.ElasticBearing` as model to represent the system under study. (It is here reminded that this model was previously incremented with a damper attached to the load to represent our "real" system.) The missing damper is the physics to discover.

Below is the comparison of the housing angular velocity for the reference and the model.

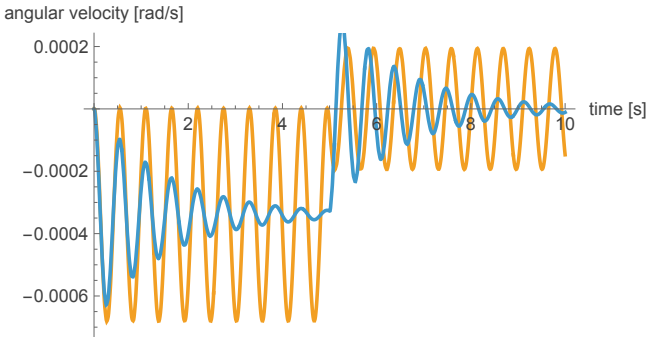


Figure 4. Housing angular velocity for the FPM with missing physics (orange) and the reference (blue): the missing damping causes visible amplitude attenuation and phase lag.

Very likely, a modeling engineer would identify that the missing physics is a damper, from the attenuations and the added phase. Nevertheless, it is here assumed that the missing physics isn't that easily identifiable. Still, the engineer expects the location of the missing physics to be between the gear and the load and, therefore, adds a dedicated dummy component to highlight this.

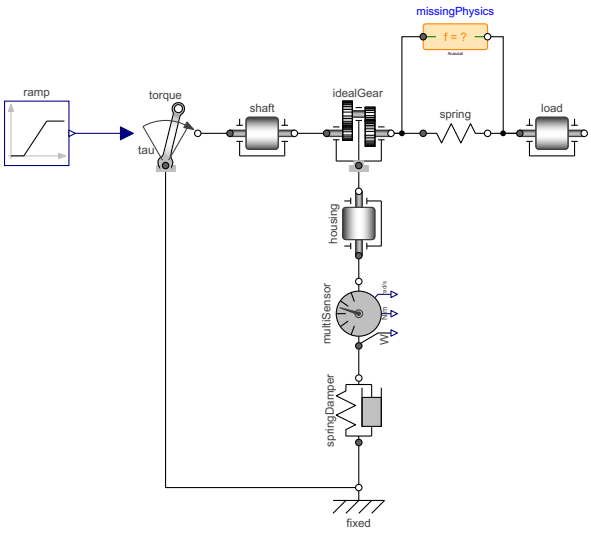


Figure 5. Reduced FPM (`BearingUDE.ElasticBearingAcausal`) with a `missingPhysics` dummy component (yellow) inserted between the gear and the load to mark the suspected location of the missing physics.

The missing component is simply a balanced model with the relevant interfaces.

4.4 The associated states

As this paper focuses on State Invariant Hybrid Models, the next step is to identify the existing states that the dummy model is connected to. The acausal model is first parsed with `parseBlockdebug`, and `modelStates` lists

DOI
10.3384/ecp223103

the full set of model states. `componentToStateMap` then maps the dummy-component connector variables to the connected states. These steps are conveniently achievable in the Wolfram Language.

```
In[11]:= parsed = parseBlockdebug["BearingUDE.ElasticBearingAcausal"];

In[12]:= modelStates[parsed]

Out[12]= shaft.w
         shaft.phi
         load.w
         load.phi
         springDamper.w_rel
         springDamper.phi_rel

In[13]:= componentToStateMap[parsed, "missingPhysics"]

Out[13]= missingPhysics.flange_a.phi -- (shaft.phi - springDamper.phi_rel)/3 + springDamper
         missingPhysics.phi_rel == load.phi - springDamper.phi_rel + 1/3 (-shaft
         missingPhysics.flange_b.phi == load.phi
```

Figure 6. Output of `componentToStateMap[]` mapping the dummy-component connector variables to the connected states `shaft.phi`, `springDamper.phi_rel` and `load.phi`.

The connected states clearly appear to be `shaft.phi`, `springDamper.phi_rel` and `load.phi`. This means that only these three states (or variables dependent to the states) shall be used as inputs to the ML model to stay in a SIHM configuration. Through the equations defined above, the dummy-component connectors `flange_a` and `flange_b` can reach these state variables or are affected by them. Hence the connector variables `phi_rel` and `w_rel` are used as inputs to the neural network, while its output is the torque `tau`, which is the quantity to be learned.

In this simple rotational example, the identified variables are also close to what an experienced modeler would likely select manually. The value of the automatic mapping is therefore mainly demonstrated mechanistically here; it is expected to be more visible in larger multi-domain models where the structurally connected state set is less obvious.

4.5 The SIHM

In this paper, it is decided to use a Neural Network as ML model, coded in Modelica and trained with the Wolfram Language using a dedicated adjoint derivative function, for performance reasons.

The training minimizes the normalized trajectory loss $L_{norm} = \|y_{SIHM}(t) - y_{ref}(t)\|_2 / \|y_{ref}(t)\|_2$ over the full simulated trajectory, without time-window batching. The trainable parameters are the neural-network weights and biases. Gradients are computed with an adjoint sensitivity routine applied to the SIHM response with respect to these parameters, and selected sensitivities were checked against finite differences. The trainings use the SGD optimizer, with a learning rate of 0.1.

```
block NeuralNetMissingPhysics
  Modelica.Blocks.Interfaces.RealInput u1 "input 1 (e.g. phi_rel)";
  Modelica.Blocks.Interfaces.RealInput u2 "input 2 (e.g. w_rel)";
  Modelica.Blocks.Interfaces.RealOutput y1 "output (e.g. torque)";
  // first-layer weights
  parameter Real w111 = -0.07409471628403752;
  parameter Real w112 = -0.10897684012583153;
  parameter Real w121 = -0.1529306464903406;
  parameter Real w122 = -0.046259369314354926;
  parameter Real w131 = 0.055963342038120034;
  parameter Real w132 = -0.21083074663261048;
  parameter Real w141 = -0.203151937354956 18;
  parameter Real w142 = -0.2935923555361084;
  // first-layer biases
  parameter Real b11 = 0.043457329748887163;
  parameter Real b12 = 0.050870520109427275;
  parameter Real b13 = 0.07206975987469227;
  parameter Real b14 = 0.09939316893743566;
  // output-layer weights
  parameter Real w211 = -0.17483029421696084;
  parameter Real w212 = 0.4733246886114033;
  parameter Real w213 = -0.24120397849572872;
  parameter Real w214 = 0.050581604403857616;
  // output bias
  parameter Real b21 = 0.04784521923178975;
  // hidden activations
  Real h11;
  Real h12;
  Real h13;
  Real h14;
equation
  h11 = tanh(w111 * u1 + w112 * u2 + b11);
  h12 = tanh(w121 * u1 + w122 * u2 + b12);
  h13 = tanh(w131 * u1 + w132 * u2 + b13);
  h14 = tanh(w141 * u1 + w142 * u2 + b14);
  y1 = w211 * h11 + w212 * h12 + w213 * h13 + w214 * h14 + b21;
  »;
end NeuralNetMissingPhysics;
```

Figure 7. Modelica source of the trained neural network `NeuralNetMissingPhysics`: two real inputs, one real output, and a single hidden layer with `tanh` activation.

```
model MissingPhysicsAcausalNNPort
  extends Modelica.Mechanics.Rotational.Interfaces.PartialCompliantWithRelativeStates;
  BearingUDE.NeuralNetMissingPhysics nn »;
equation
  nn.u1 = phi_rel;
  nn.u2 = w_rel;
  tau = nn.y1;
  »;
end MissingPhysicsAcausalNNPort;
```

Figure 8. Modelica source of `MissingPhysicsAcausalNNPort`, which instantiates the neural network and wires its inputs to `phi_rel` and `w_rel` and its output to `tau`.

The dummy component is thus replaced with the Neural Network model and trained accordingly.

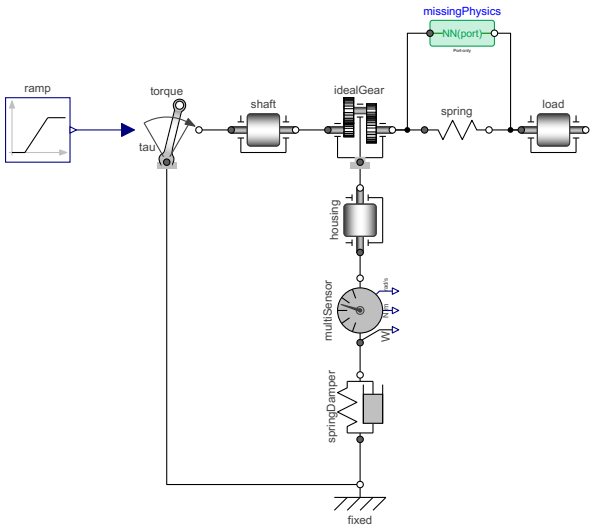


Figure 9. SIHM with the dummy component replaced by the Neural Network and simulated in Wolfram System Modeler.

The training loss with respect to the number of iterations is here plotted, and the load response of the SIHM is compared to the reference.

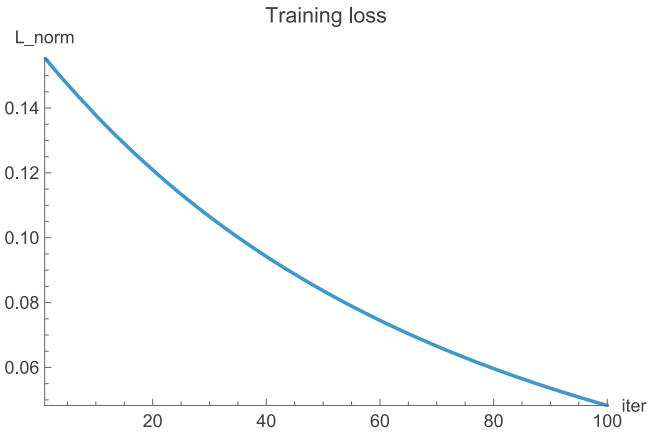


Figure 10. Training loss *L_norm* versus iteration count for the SIHM. Training is intentionally stopped before full convergence.

Compared with the initial FPM response, the SIHM response moves toward the reference trajectory, indicating that the local neural component is learning a correction consistent with the missing damper effect. Since training is intentionally stopped before convergence, the present result should be interpreted as proof of workflow feasibility rather than as an optimized identification result.

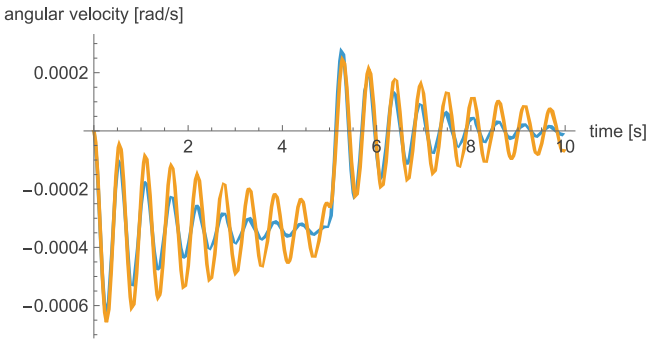


Figure 11. Housing angular velocity of the reference (blue) and the SIHM (orange) after partial training, illustrating progressive recovery of the missing damping effect.

Below is an illustration of the results when more iterations are allowed in the training.

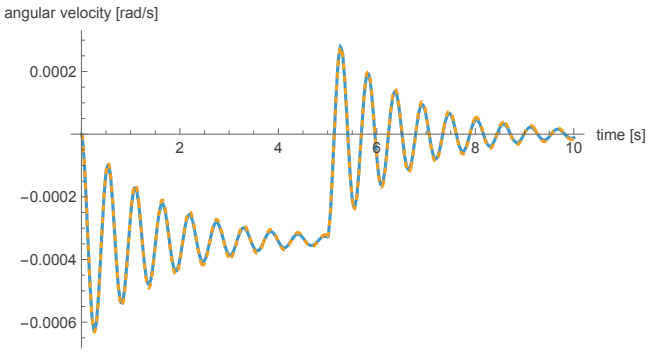


Figure 12. Same comparison as Figure 11 after extended training, showing near-complete recovery of the reference trajectory.

Because the single-local case uses synthetic data with known missing physics, the learned torque can be compared directly with the removed damper torque. The true missing contribution is $\tau_d = d * w_{rel}$, with $d = 10\ N.m.s/rad$. The trained neural component follows this torque over the simulated trajectory, indicating that the local residual recovers the intended missing effect rather than only producing an arbitrary trajectory-level correction.

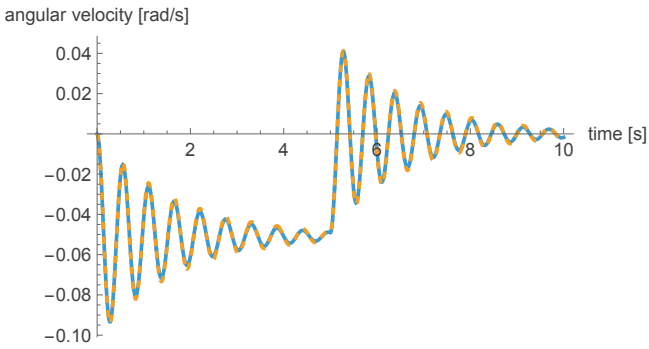


Figure 13. Comparison of the angular velocities of the spring damper (blue) and the missingPhysics (orange).

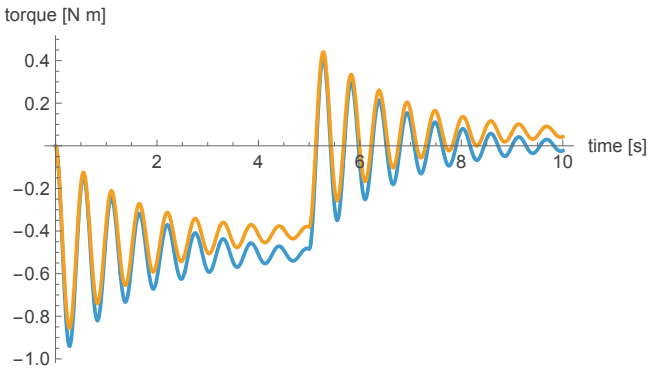


Figure 14. Comparison of the torques of the spring damper (blue) and the missingPhysics (orange) – roughly 10x the angular velocity, as expected from the known physics.

4.6 Additional comments

The authors would like to emphasize the following points:

DOI
10.3384/ecp223103

1. On purpose, the number of iterations is stopped prior to convergence to show a difference between reference and SIHM, highlighting the fact that the physics is learned progressively.

2. The hyperbolic tangent was used as activation function for the Neural Network. It is a common one to use. Nevertheless, knowing the solution, a linear activation function would have been much more appropriate and would have converged in few iterations. Again, this is a conscious choice to highlight the assumption that the physics isn't known and show that, still, physics can be learned.

3. Similarly, the number of layers of the multi-layer perceptron Neural Network and size of each layer has been arbitrarily chosen to be 2 and 4 respectively. Refining these numbers would be meaningful if the scope of the paper was applied to a real system.

5 A multi-local example

As a final step, the workflow is repeated with two simultaneous missing effects: the damper connected to the load is removed, and the spring constant of the spring connected to ground is modified relative to the reference model. A second dummy component is placed in parallel with the spring-damper connected to ground. The structural analysis is then applied independently to both dummy components, yielding two local input-output mappings.

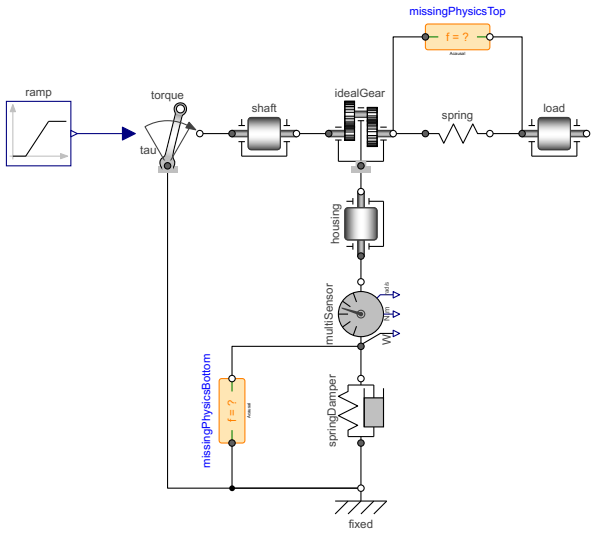


Figure 15. Multi-local SIHM with two dummy components: `missingPhysicsTop` (between gear and load) and `missingPhysicsBottom` (in parallel with the spring-damper connected to ground).

The purpose of this example is multiple:

1. To demonstrate that different physical effects can be learned – here, damping and elasticity.

2. To SIHM can also compensate from a wrongly calibrated component – the spring stiffness being different from source of truth, the SIHM learns a second spring in parallel, which combination of both leads to the "true" stiffness.

3. To test whether the workflow remains modular when more than one learned component is introduced. Each neural component receives only the state-related variables associated with its own location, and each contributes only through its own connector variables.

The two neural components are trained simultaneously against the same reference trajectory.

```
In[76]:= componentToStateMap[parsed, "missingPhysicsTop"]

Out[76]= missingPhysicsTop.flange_a.phi == (shaft.phi - springDamper.phi_rel)/3 + springDa
         missingPhysicsTop.phi_rel == load.phi - springDamper.phi_rel + 1/3 (-sh
         missingPhysicsTop.flange_b.phi == load.phi

In[77]:= componentToStateMap[parsed, "missingPhysicsBottom"]

Out[77]= missingPhysicsBottom.w_rel == springDamper.w_rel
         missingPhysicsBottom.flange_b.phi == springDamper.phi_rel
         missingPhysicsBottom.phi_rel == springDamper.phi_rel
         missingPhysicsBottom.a_rel == 1/5 (-10000 springDamper.phi_rel + 1000
           0.333333 (0. + 1000. (load.phi - springDamper.phi_rel + 1/3 (-shaft
```

Figure 16. Independent structural analyses for the two dummy components yield two separate input-output mappings via `componentToStateMap`.

The resulting SIHM reduces the discrepancy between the FPM and the reference model for the selected outputs, showing that the workflow can be extended from a single local residual to multiple local residuals in one Modelica model. However, because both learned components are trained from system-level output data, the separation between the two missing effects is not guaranteed by optimization alone. It is induced by the structural restrictions imposed through the two local component interfaces.

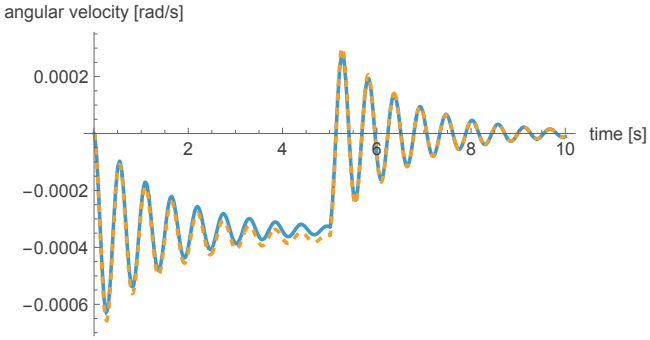


Figure 17. Housing angular velocity of the reference and the multi-local SIHM after simultaneous training of both neural components.

6 Conclusion and perspectives

This paper introduced the concept of State-Invariant Hybrid Model (SIHM) and presented a workflow for learning missing physics *locally* in a Modelica model using machine learning. The key contributions are threefold.

First, the distinction between global and local learning was formalized. Global learning exposes the entire state vector to the ML model, which may capture unintended inaccuracies beyond the targeted missing physics. Local learning restricts the ML model to the subset of states and inputs connected to its insertion point, making the learned correction more traceable to a specific model location.

Second, a tool-agnostic workflow was proposed that leverages Modelica's component-based and acausal modeling strengths. By inserting a replaceable dummy component at the suspected location of missing physics, the modeler stays within the familiar Modelica paradigm. The Modelica compiler can then be used to derive the connected states and variables that define the local ML interface, reducing the need for manual interface identification.

Third, the workflow was demonstrated within the Wolfram ecosystem on the ElasticBearing example from the Modelica Standard Library. The implementation illustrates how connected states can be extracted programmatically, how a neural component can be trained to compensate for a missing local effect, and how the same procedure can be repeated for two learned components in one model or recalibrating a parameter.

Several perspectives emerge from this work. The workflow is tool-agnostic in principle but depends on access to compiler-generated structural information; demonstrating the same procedure in other Modelica tools would therefore be an important next step. Further work should also address algebraic constraints more explicitly, quantify identifiability limits of local residuals, compare local and global learning under the same data conditions, support ML models beyond neural networks, such as KANs (Liu et al., 2024) or symbolic regression (Cranmer, 2023), and evaluate the workflow on industrial-scale models with real measurement data.

7 References

Bellman, R. and K. J. Åström (1970). “On Structural Identifiability”. Mathematical Biosciences, 7(3–4), 329–339.

Brunton, S. L., J. L. Proctor, and J. N. Kutz (2016). “Discovering governing equations from data by sparse identification of nonlinear dynamical systems”. Proceedings of the National Academy of Sciences, 113(15), 3932–3937.

Chen, R. T. Q., Y. Rubanova, J. Bettencourt, and D. Duvenaud (2018). “Neural Ordinary Differential Equations”. In: Advances in Neural Information Processing Systems. arXiv:1806.07366.

Coïc, C. (2025). “Learn Modelica & FMI” newsletter, available openly on LinkedIn and https://dr-clementcoic.github.io/LearnModelicaFMI/

Cranmer, M. (2023). “Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl”. arXiv preprint arXiv:2305.01582.

Hanke, T. J., J. Brunnemann, R. Flesch, and J. Eiden (2025). “Status of the SMArtInt Library: Simple Modelica Artificial Intelligence Interface”. In: Proceedings of the 16th International Modelica & FMI Conference. DOI: 10.3384/ecp218459.

Hofmann, A. and L. Mikelsons (2025). “Towards Integration of PeN-ODEs in a Modelica-based workflow”. In: Proceedings of the 16th International Modelica & FMI Conference. DOI: 10.3384/ecp218435.

Ljung, L. (1999). System Identification: Theory for the User. 2nd ed. Prentice Hall.

Langenkamp, L., P. Hannebohm, and B. Bachmann (2025). “Efficient Training of Physics-enhanced Neural ODEs via Direct Collocation and Nonlinear Programming”. In: Proceedings of the 16th International Modelica & FMI Conference. DOI: 10.3384/ecp218445.

Kamp, T. and C. Bürger (2025). “Hybrid Simulation Models for Embedded Applications: A Modelica and eFMI Approach”. In: Proceedings of the 16th International Modelica & FMI Conference.

Liu, Z. et al. (2024). “KAN: Kolmogorov-Arnold Networks”. arXiv preprint arXiv:2404.19756.

Modelica Association (2024). Functional Mock-up Interface Specification 3.0.2. URL: https://fmi-standard.org/docs/3.0.2/.

ONNX project (2026). ONNX Intermediate Representation Specification. URL: https://onnx.ai/onnx/repo-docs/IR.html.

Rackauckas, C. et al. (2020). “Universal Differential Equations for Scientific Machine Learning”. arXiv preprint arXiv:2001.04385.

Raissi, M., P. Perdikaris, and G. E. Karniadakis (2017). “Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations”. arXiv preprint arXiv:1711.10561.

Thummerer, T., F. Jarmolowitz, D. Sommer, and L. Mikelsons (2025). “Br(e)aking the Boundaries of Physical Simulation Models: Neural Functional Mock-up Units for Modeling the Automotive Braking System”. In: Proceedings of the 16th International Modelica & FMI Conference. DOI: 10.3384/ecp218575.

Urbani, M., M. Bolognese, L. Prattico, and M. Testi (2025). “A Tool for the Implementation of Open Neural Network Exchange Models in Functional Mockup Units”. In: Proceedings of the 16th International Modelica & FMI Conference. DOI: 10.3384/ecp218645.

Closed-Form Local Stability of a Pressure-Phase-Coordinate Control-Volume Model

Hongtao Qiao[1] Christopher R. Laughman[1] Vedang Deshpande[1] Scott A. Bortoff[1]

[1]Mitsubishi Electric Research Laboratories, Cambridge, MA, USA,
{qiao,laughman,deshpande,bortoff}@merl.com

Abstract

This paper studies the local stability of a pressure–phase-coordinate control-volume model for one-dimensional compressible flow. Using pressure and a generalized phase coordinate as the state variables, the model retains the essential effects of compressibility, phase change, flow restriction, and wall heat transfer while remaining simple enough for closed-form analysis. Starting from the mass and energy balances, explicit state equations are derived and linearized about an equilibrium operating point. The analysis yields a closed-form factorization of the Jacobian determinant, showing that saddle versus non-saddle behavior is governed by a single phase-coordinate sensitivity term in the energy balance. In strictly single-phase regions, and in the bulk of the two-phase region under the present approximation, the model is non-saddle. By contrast, sharp heat-transfer variation near phase boundaries can drive direct stable-to-saddle transitions. In a higher-pressure regime, the analysis also identifies conditions under which a Hopf candidate may arise. These results provide a simple and physically interpretable framework for understanding how heat transfer influences local stability in two-phase flow models.

Keywords: two-phase flow, control-volume model, pressure–phase coordinates, local stability, heat transfer, saddle instability

1 Introduction

Dynamic models are widely used in the simulation, analysis, and control of vapor-compression-based HVAC (Heating, Ventilation, Air-Conditioning) systems (Qiao et al. 2015). In practice, however, such models may exhibit pressure oscillations even under constant inputs, and the mechanism behind this behavior is not always clear. In particular, when the refrigerant moves across phase boundaries, the heat transfer coefficient may change sharply with phase state and introduce strong feedback into the system dynamics.

Fig. 1 shows a representative example from a cycle-level simulation, where the discharge pressure exhibits a sustained oscillation under constant inputs. Although this behavior can be observed numerically, its origin is difficult to identify directly from the full cycle model.

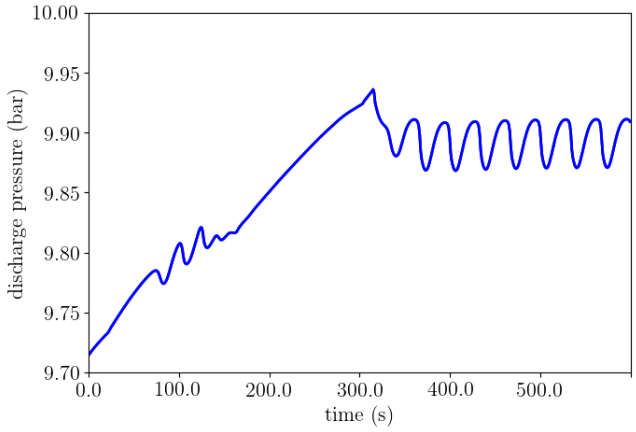


Figure 1. Oscillatory discharge-pressure behavior in a cycle-level simulation under constant inputs (Laughman and Qiao 2018).

Two-phase flow systems are known to exhibit a wide range of static and dynamic instabilities, including Ledinegg instability, pressure-drop oscillations, density-wave oscillations, and other system-level oscillatory phenomena (Yadigaroglu 1981; Kakaç and Bon 2008; Ruspini et al. 2014). These mechanisms have been studied in many thermal-fluid systems, including boiling channels, boilers, evaporators, refrigeration systems, and heat-pump components (Eborn and Åström 2000; Qiao and Laughman 2021). In this literature, stability is often determined by the interaction among pressure-drop characteristics, distributed transport delays, compressible volumes, phase-change dynamics, and external hydraulic boundary conditions.

The oscillation shown in Fig. 1 is related to this broader class of two-phase-flow stability problems, but the objective of the present paper is more local. Rather than analyzing a complete flow loop or a particular system-level instability mode, we study the local stability of a single pressure–phase-coordinate control-volume model coupled to an outlet flow relation. This reduced setting allows the effects of compressibility, phase change, flow restriction, and phase-dependent heat-transfer feedback to be isolated in closed form.

This local viewpoint is closely tied to the structure of detailed HVAC cycle models. Their dominant dynamics are often represented by finite-volume heat-exchanger models built from sequences of control volumes and flow

models. Such models capture the fluid and thermal dynamics well, but their complexity makes theoretical stability analysis difficult. As a result, even when oscillations are observed in simulation, it is not easy to determine whether they arise from flow restriction, thermodynamic sensitivities, heat-transfer closure, or the interaction of these effects.

The motivation of the present work is therefore to study this local feedback structure in a simpler analytical setting. The reduced model retains the essential mechanisms of compressibility, phase change, flow restriction, and heat-transfer feedback, while remaining simple enough for closed-form analysis. This viewpoint is also relevant beyond the reduced model itself, because many distributed heat-exchanger and cycle models are assembled from repeated control volumes coupled through flow restrictions. The present analysis should therefore be interpreted as a local mechanism study: it does not attempt to reproduce the full system dynamics, but instead isolates a compact feedback structure that may also govern instability in larger models.

For this purpose, the model is formulated in pressure and phase coordinate. The phase coordinate x coincides with vapor quality in the two-phase region and is extended continuously into the neighboring liquid and vapor regions through an enthalpy-based normalization. In these coordinates, the governing equations take the form of a compact second-order nonlinear system.

Within this framework, the paper develops closed-form local stability conditions for the pressure–phase-coordinate model. In particular, the analysis shows that local saddle versus non-saddle behavior is controlled by a single phase-coordinate sensitivity term in the energy balance, which directly links stability loss to sharp heat-transfer variation near phase boundaries. Representative simulation cases are then used to verify the resulting regime picture.

The remainder of the paper is organized as follows. Section 2 derives the pressure–phase-coordinate control-volume model. Section 3 develops the local stability conditions and their regime interpretation. Section 4 presents simulation cases that verify the analytical predictions. Section 5 concludes the paper.

2 Model Formulation

This section derives a control-volume model for one-dimensional compressible flow using pressure p and phase coordinate x as the state variables. The cross-sectional area is assumed constant, while gravitational force and axial heat conduction are neglected. The control volume is further assumed to remain in local thermodynamic equilibrium, so that thermodynamic properties can be parameterized by p and x (Laughman et al. 2026).

2.1 Definition of the phase coordinate

The state variable x is defined as vapor quality in the two-phase region. To extend the phase coordinate to single-phase regions, the same enthalpy-based normalization is continued outside the interval $[0,1]$ using the local saturation liquid and vapor enthalpies at the same pressure. Specifically, in the two-phase region,

$$x = \frac{h - h_{\mathrm{sat,liq}}(p,h)}{h_{\mathrm{sat,vap}}(p,h) - h_{\mathrm{sat,liq}}(p,h)}, \tag{1}$$

which reduces to the standard vapor-quality definition. In the vapor region, x is extended as

$$x = \frac{h - h_{\mathrm{dew,liq}}(p)}{h_{\mathrm{dew,vap}}(p) - h_{\mathrm{dew,liq}}(p)}, \tag{2}$$

and in the liquid region, x is extended as

$$x = \frac{h - h_{\mathrm{bub,liq}}(p)}{h_{\mathrm{bub,vap}}(p) - h_{\mathrm{bub,liq}}(p)}. \tag{3}$$

For a pure fluid, the same saturation-enthalpy difference may be used as the denominator in Equations (1)–(3). The separate bubble- and dew-based expressions are retained here because the formulation is also intended for refrigerant mixtures, for which the corresponding equilibrium liquid and vapor reference states generally have different phase compositions and enthalpy differences.

With this definition, $x = 0$ and $x = 1$ remain aligned with the saturated-liquid and saturated-vapor boundaries, respectively, while $x < 0$ represents liquid states and $x > 1$ represents vapor states. Thus, x serves as a continuous phase-coordinate across liquid, two-phase, and vapor regions. Outside the two-phase region, x should be interpreted as a generalized phase-coordinate rather than as a physical vapor mass fraction.

This extension is consistent with the two-phase definition because the normalization is based on the same local saturation reference states used to define vapor quality inside the two-phase region. As a result, the coordinate system remains continuous across the phase boundaries and is convenient for both model formulation and local stability analysis. In particular, it allows the same reduced-order state-space structure to be used when discussing the behavior near the liquid-to-two-phase and vapor-to-two-phase transitions.

Fig. 2 illustrates this construction schematically in the p–h plane. Inside the dome, the dashed green curves denote constant values of the phase coordinate x, which coincide with the conventional vapor quality. Outside the dome, the same normalization extends x continuously into the liquid region ($x < 0$) and vapor region ($x > 1$). The dashed black curve marks the locus along which $h_p = (\partial h/\partial p)_x$ changes sign. Importantly, this is not a single pressure level: the location of the sign change depends on the constant-x trajectory. This distinction will be used later when discussing bulk two-phase and vapor-region stability.

DOI
10.3384/ecp223113

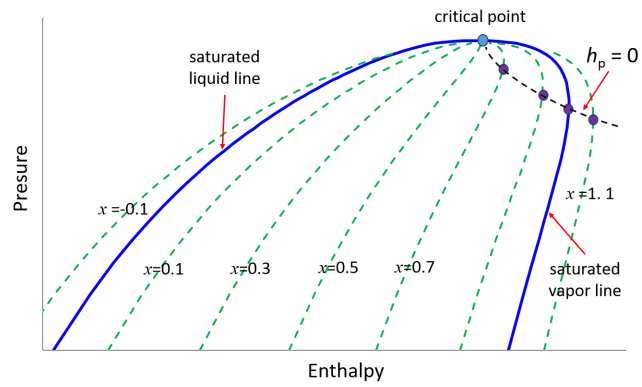


Figure 2. Schematic p–h representation of the extended phase coordinate x.

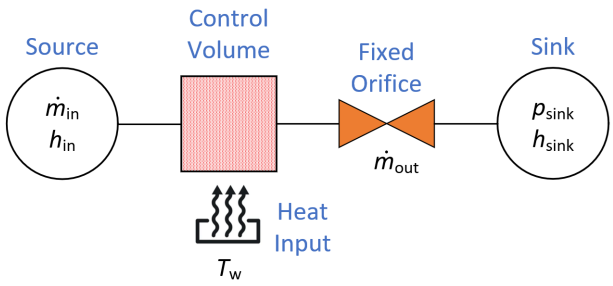


Figure 3. Control-volume model with inlet flow, fixed-orifice outlet, and wall heat input.

2.2 Governing equations

Fig. 3 shows the considered control volume. The inlet is characterized by mass flow rate $\dot{m}_{\text{in}}$ and specific enthalpy h_{in}, the outlet is connected to a pressure sink p_{sink} through a fixed orifice, and the wall exchanges heat with the fluid through a heat-transfer coefficient α, area A, and wall temperature T_w.

A convenient notation is to define the thermodynamic derivatives with respect to the independent variables p and x as

$$\rho_p = \left(\frac{\partial \rho}{\partial p}\right)_x, \qquad \rho_x = \left(\frac{\partial \rho}{\partial x}\right)_p, \tag{4}$$

$$h_p = \left(\frac{\partial h}{\partial p}\right)_x, \qquad h_x = \left(\frac{\partial h}{\partial x}\right)_p. \tag{5}$$

Here, ρ denotes fluid density and h denotes specific enthalpy. With these definitions, the mass and energy balances for the control volume can be written in compact form as

$$\begin{bmatrix} \rho_p & \rho_x \\ \rho h_p - 1 & \rho h_x \end{bmatrix} \begin{bmatrix} \dot{p} \\ \dot{x} \end{bmatrix} = \frac{1}{V} \begin{bmatrix} S_M \\ S_E \end{bmatrix}, \tag{6}$$

where V is the control-volume size, S_M is the net mass source term, and S_E is the energy source term.

Equation (6) is the key reduced storage relation of the model. The left-hand side captures how variations in pressure and phase coordinate affect density and enthalpy inside the control volume through thermodynamic sensitivities. The right-hand side contains the external forcing through the net mass and energy exchange. In this way, the model preserves the essential coupling between compressibility and phase change in a compact form.

The mass source term represents the net mass accumulation rate

$$S_M = \dot{m}_{\text{in}} - \dot{m}_{\text{out}}, \tag{7}$$

where the outlet flow is determined from a fixed-orifice relation

$$\dot{m}_{\text{out}} = \frac{\dot{m}_0}{(\Delta p_0)^{1/2}} (p - p_{\text{sink}})^{1/2}. \tag{8}$$

Equation (8) links the control-volume pressure to the downstream sink condition. The quantity K_p, introduced later, may be interpreted as the sensitivity of the outlet mass flow to pressure and therefore reflects the dynamic effect of the outlet restriction.

The energy source term consists of the inlet enthalpy flow and the external heat input,

$$S_E = \dot{m}_{\text{in}}(h_{\text{in}} - h) + \alpha A(T_w - T). \tag{9}$$

Equation (9) contains two distinct mechanisms. The first term represents the net enthalpy transport into the control volume after applying the well-mixed outlet condition $h_{\text{out}} = h$. Equivalently, it is the inlet enthalpy flow measured relative to the specific enthalpy of the fluid in the control volume. The second term describes heat transfer between the wall and the fluid. This decomposition is useful because the local stability result depends on how these two contributions vary with pressure and phase coordinate.

Solving (6) for $\dot{p}$ and $\dot{x}$ gives explicit state equations:

$$\dot{p} = \frac{\rho h_x S_M - \rho_x S_E}{\rho V h_x \left(\rho_p - \rho_x h_p h_x^{-1} + \rho^{-1} \rho_x h_x^{-1}\right)}, \tag{10}$$

$$\dot{x} = \frac{(1 - \rho h_p) S_M + \rho_p S_E}{\rho V h_x \left(\rho_p - \rho_x h_p h_x^{-1} + \rho^{-1} \rho_x h_x^{-1}\right)}. \tag{11}$$

These equations already show the main structure of the reduced model. Pressure and phase coordinate are not driven by separate balances; instead, both states respond to both S_M and S_E, and the coupling is weighted by the thermodynamic derivatives.

2.3 Characteristic speed and compact state-space form

To make the role of compressibility more transparent, the speed of sound c is rewritten in pressure–phase coordinates as (Qiao et al. 2023)

$$c = \left(\rho_p - \rho_x h_p h_x^{-1} + \rho^{-1} \rho_x h_x^{-1}\right)^{-1/2}. \tag{12}$$

It is convenient to define

$$\Upsilon \triangleq h_x^{-1}, \qquad \Lambda \triangleq \rho_x h_x^{-1} = \left(\frac{\partial \rho}{\partial h}\right)_p, \qquad \Phi \triangleq \frac{c^2}{\rho V}. \tag{13}$$

With the extended definition of x, one has $h_x > 0$, hence $\Upsilon > 0$, provided that the corresponding saturation-enthalpy differences are positive. Also, $\Phi > 0$. Moreover, at fixed pressure, increasing x moves the state toward the vapor side, so that $\rho_x < 0$. Therefore, $\Lambda < 0$.

Substituting the sound-speed expression into the state equations yields

$$\dot{p} = f_1(z,u) = \Phi\big[\rho S_M - \Lambda S_E\big], \tag{14}$$

$$\dot{x} = f_2(z,u) = \Phi\Upsilon\big[(1-\rho h_p)S_M + \rho_p S_E\big]. \tag{15}$$

The model is therefore a second-order nonlinear system with state vector

$$z = \begin{bmatrix} p \\ x \end{bmatrix}, \tag{16}$$

and input vector

$$u = \begin{bmatrix} \dot{m}_{\text{in}} & h_{\text{in}} & p_{\text{sink}} & T_w \end{bmatrix}^{\mathsf{T}}. \tag{17}$$

An equilibrium pair (z_e, u_e) satisfies

$$S_M(z_e,u_e) = 0, \qquad S_E(z_e,u_e) = 0. \tag{18}$$

Define the perturbations

$$\Delta z = z - z_e, \qquad \Delta u = u - u_e. \tag{19}$$

Linearizing the nonlinear system around (z_e, u_e) gives

$$\Delta\dot{z} = A\,\Delta z + B\,\Delta u, \tag{20}$$

where

$$A = \left.\frac{\partial f}{\partial z}\right|_{(z_e,u_e)}, \qquad B = \left.\frac{\partial f}{\partial u}\right|_{(z_e,u_e)}. \tag{21}$$

Since the local stability analysis depends only on the state Jacobian A, the full expressions for the entries of the input matrix B are omitted for brevity.

2.4 Source-term sensitivities and Jacobian structure

To express A compactly, define the source-term sensitivities at equilibrium. For the mass source term,

$$\frac{\partial S_M}{\partial p} = K_p, \qquad K_p = -\frac{\dot{m}_0}{2(\Delta p_0)^{1/2}}\frac{1}{\sqrt{p - p_{\text{sink}}}} < 0. \tag{22}$$

For the energy source term, define

$$\beta_p \triangleq \frac{\partial S_E}{\partial p}, \qquad \beta_x \triangleq \frac{\partial S_E}{\partial x}. \tag{23}$$

Since

$$S_E = \dot{m}_{\text{in}}(h_{\text{in}} - h) + \alpha A(T_w - T) \tag{24}$$

and $\alpha = \alpha(x)$, one obtains

$$\beta_p = -\dot{m}_{\text{in}} h_p - \alpha A T_p, \tag{25}$$

$$\beta_x = -\dot{m}_{\text{in}} h_x + \alpha_x A(T_w - T) - \alpha A T_x, \tag{26}$$

where

$$T_p = \left(\frac{\partial T}{\partial p}\right)_x, \qquad T_x = \left(\frac{\partial T}{\partial x}\right)_p, \qquad \alpha_x = \left(\frac{\partial \alpha}{\partial x}\right)_p. \tag{27}$$

Using these definitions, the Jacobian matrix can be written in the compact form

$$A = \Phi\begin{bmatrix} \rho K_p - \Lambda\beta_p & -\Lambda\beta_x \\ \Upsilon\big[(1-\rho h_p)K_p + \rho_p\beta_p\big] & \Upsilon\rho_p\beta_x \end{bmatrix}. \tag{28}$$

Equation (28) isolates the physical contributions that determine the local dynamics: outlet restriction through K_p, pressure-direction sensitivity of the energy source term through β_p, and phase-direction sensitivity through β_x.

2.5 Simplified heat-transfer model

To keep the local stability analysis analytically tractable, the heat-transfer coefficient α is approximated as a function of the phase coordinate x alone, as illustrated in Fig. 4. In general, however, α may also depend on pressure, wall temperature, mass flux, and other local flow variables. The present simplification is adopted because the goal of this paper is not to reproduce a detailed heat-transfer correlation, but to isolate the dominant phase-dependent variation that is most relevant to local stability near phase transitions.

Under this approximation, α is taken to be approximately constant in the liquid, bulk two-phase, and vapor regions, with representative values α_{liq}, α_{2ph}, and α_{vap}, respectively. Smooth blending is introduced in narrow transition regions near the liquid-to-two-phase and two-phase-to-vapor boundaries, so that α remains nearly constant over most of each phase region and varies rapidly only near the phase boundaries. This representation captures the main qualitative feature of practical heat-transfer correlations: the strongest phase-dependent variation often occurs near transitions between single-phase and two-phase conditions, whereas the variation within the bulk of each region is generally weaker.

Accordingly, $\alpha_x = d\alpha/dx = 0$ outside the transition regions. In the low-quality transition region, $\alpha_x > 0$ because α increases from α_{liq} to α_{2ph}. In the high-quality transition region, $\alpha_x < 0$ because α decreases from α_{2ph} to α_{vap}. As shown in the following section, this phase-boundary variation of α plays a central role in the onset of saddle instability.

Extension to general heat-transfer correlations. Although the regime interpretation developed below uses $\alpha = \alpha(x)$, the determinant simplification itself is not restricted to this assumption. For a more general heat-transfer correlation,

$$\alpha = \alpha(p, x, G, T_w, \ldots), \tag{29}$$

DOI
10.3384/ecp223113

where G denotes the local mass flux, the pressure- and phase-direction sensitivities of the energy source term become

$$\beta_p = -\dot{m}_{\text{in}} h_p + A(T_w - T)\alpha_p - \alpha A T_p, \tag{30}$$
$$\beta_x = -\dot{m}_{\text{in}} h_x + A(T_w - T)\alpha_x - \alpha A T_x, \tag{31}$$

where

$$\alpha_p = \left(\frac{\partial \alpha}{\partial p}\right)_x, \qquad \alpha_x = \left(\frac{\partial \alpha}{\partial x}\right)_p \tag{32}$$

denote the complete local sensitivities of the heat-transfer coefficient.

In the present local analysis, the inlet mass flow is treated as an external input. Therefore, if the mass flux is determined directly from the prescribed inlet flow and flow area, it is held fixed when the state Jacobian is evaluated. In that case, mass-flux dependence changes the equilibrium value of α but does not introduce additional state derivatives. If a more general closure makes G state dependent, the derivatives in Eqs. (30) and (31) should instead be interpreted as total local sensitivities. For example,

$$\alpha_x^{\text{eff}} = \left(\frac{\partial \alpha}{\partial x}\right)_{p,G,T_w} + \left(\frac{\partial \alpha}{\partial G}\right)_{p,x,T_w} \left(\frac{\partial G}{\partial x}\right)_p. \tag{33}$$

With these complete sensitivities, the Jacobian retains the same structural form, and the determinant factorization derived in Section 3 remains

$$\det A = \frac{\Phi \Upsilon K_p}{V} \beta_x. \tag{34}$$

Thus, a more detailed heat-transfer correlation changes the numerical values of β_p and β_x and therefore shifts the trace and determinant boundaries. It does not, however, change the central structural result that saddle versus non-saddle behavior is governed by the phase-direction sensitivity of the complete energy source term.

3 Closed-Form Local Stability Conditions

Local stability is determined by the eigenvalues of the Jacobian matrix A. Because the linearized system is planar, its characteristic polynomial can be written as

$$\lambda^2 - \tau\lambda + \Delta = 0, \tag{35}$$

where

$$\tau = \operatorname{tr} A, \qquad \Delta = \det A. \tag{36}$$

The main point of this section is that, for the present pressure–phase-coordinate model, both Δ and τ admit physically interpretable forms. This makes the local stability conditions much more transparent than a direct numerical eigenvalue calculation.

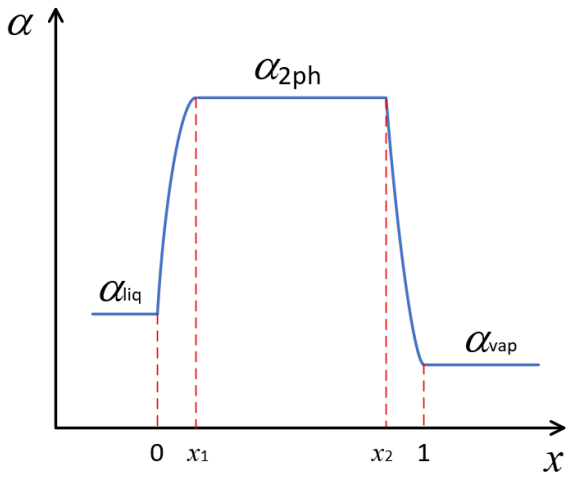


Figure 4. Heat transfer coefficient variations with extended phase coordinate x.

3.1 Determinant simplification

For the Jacobian above, the determinant admits the closed-form factorization. Starting from (28):

$$\begin{aligned}\Delta &= \Phi^2 \begin{vmatrix} \rho K_p - \Lambda\beta_p & -\Lambda\beta_x \\ \Upsilon\left[(1-\rho h_p)K_p + \rho_p\beta_p\right] & \Upsilon\rho_p\beta_x \end{vmatrix} \\ &= \Phi^2\Upsilon\beta_x\left[(\rho K_p - \Lambda\beta_p)\rho_p + \Lambda\big((1-\rho h_p)K_p + \rho_p\beta_p\big)\right] \\ &= \Phi^2\Upsilon K_p\beta_x\left[\rho\rho_p + \Lambda(1-\rho h_p)\right].\end{aligned}$$

The terms proportional to $\Lambda\rho_p\beta_p$ cancel. The remaining bracket is fixed by the sound-speed identity, since

$$\rho\rho_p + \Lambda(1-\rho h_p) = \rho\left(\rho_p - \Lambda h_p + \rho^{-1}\Lambda\right) = \rho c^{-2}.$$

Using $\Phi = c^2/(\rho V)$ then gives

$$\Delta = \frac{\Phi\Upsilon K_p}{V}\beta_x. \tag{37}$$

Since $\Phi > 0$, $\Upsilon > 0$, $K_p < 0$, and $V > 0$, the sign of Δ is governed entirely by the sign of β_x. Therefore,

$$\Delta > 0 \iff \beta_x < 0, \tag{38}$$
$$\Delta < 0 \iff \beta_x > 0, \tag{39}$$
$$\Delta = 0 \iff \beta_x = 0. \tag{40}$$

This is the key analytical simplification. The determinant test reduces to the sign of a single scalar coefficient associated with the phase-coordinate sensitivity of the energy source term. Through the expression for β_x, the determinant depends directly on the heat-transfer-coefficient gradient α_x in the transition regions, given that $T_x \geq 0$.

3.2 Trace condition

The trace of the Jacobian is

$$\tau = \Phi\left(\rho K_p - \Lambda\beta_p + \Upsilon\rho_p\beta_x\right). \tag{41}$$

Unlike the determinant, the trace does not reduce to a single sign-definite coefficient. Its sign depends on the operating condition through β_p and ρ_p, and therefore must in general be evaluated explicitly.

Accordingly, the determinant and trace play different roles: the determinant distinguishes saddle from non-saddle behavior through β_x, whereas the trace determines whether a non-saddle equilibrium is stable, unstable, or marginal.

3.3 General stability classification

For a second-order system, the Hurwitz criterion implies that the equilibrium is asymptotically stable if and only if

$$\Delta > 0, \qquad \tau < 0. \tag{42}$$

If $\Delta < 0$, the equilibrium is a saddle and is therefore necessarily unstable. If $\Delta > 0$ and $\tau = 0$, the equilibrium is marginal in the linear sense and is referred to here as a Hopf candidate. The boundary $\Delta = 0$ marks the transition between saddle and non-saddle behavior.

Thus, for the present model, the local stability problem separates naturally into two parts: a determinant test, governed entirely by β_x, and a trace test, which determines whether a non-saddle equilibrium is stable, unstable, or marginal.

3.4 Strictly single-phase equilibria

A useful simplification arises when the equilibrium lies strictly in the single-phase liquid or vapor regions. Under the present heat-transfer approximation, $\alpha_x = 0$ away from the blending zones. In that case,

$$\beta_x = -\dot{m}_{in} h_x - \alpha A T_x. \tag{43}$$

Since $h_x > 0$ and $T_x > 0$, one has $\beta_x < 0$, and therefore

$$\Delta = \frac{\Phi \Upsilon K_p}{V} \beta_x > 0. \tag{44}$$

Hence, strictly single-phase equilibria cannot be saddles under these conditions. The remaining classification is determined by the trace.

3.4.1 Liquid equilibria

In the strictly single-phase liquid region, we consider the operating regimes in which

$$K_p < 0, \qquad h_p > 0, \qquad T_p > 0, \qquad \rho_p < 0. \tag{45}$$

Then $\beta_p < 0$. From (41), the terms ρK_p and $-\Lambda\beta_p$ are negative, whereas $\Upsilon\rho_p\beta_x > 0$ because $\rho_p < 0$ and $\beta_x < 0$. Therefore, the sign of τ is not automatically negative in the general liquid-region case.

However, liquid is nearly incompressible in many operating conditions of interest, so that ρ_p is often small in magnitude. Under the approximation

$$\rho_p \approx 0, \tag{46}$$

the trace reduces to

$$\tau \approx \Phi\left(\rho K_p - \Lambda\beta_p\right). \tag{47}$$

Because $K_p < 0$, $\Lambda < 0$, and $\beta_p < 0$, both terms in (47) are negative, so $\tau < 0$. Together with (44), this implies local asymptotic stability in the strictly single-phase liquid region under the additional approximation $\rho_p \approx 0$.

3.4.2 Vapor equilibria

In the strictly single-phase vapor region, two subregimes may be distinguished.

Below the $h_p = 0$ locus. Consider the operating regime below the locus in the p–h plane where $h_p = (\partial h/\partial p)_x$ changes sign along constant-x trajectories (see Fig. 2). In this regime,

$$K_p < 0, \qquad h_p > 0, \qquad T_p > 0, \qquad \rho_p > 0. \tag{48}$$

Then $\beta_p < 0$. From (41), all three terms are negative:

$$\rho K_p < 0, \qquad -\Lambda\beta_p < 0, \qquad \Upsilon\rho_p\beta_x < 0.$$

Hence $\tau < 0$. Therefore, in this vapor-region subregime, the condition $\beta_x < 0$ implies not only $\Delta > 0$ but also $\tau < 0$, so the equilibrium is locally asymptotically stable.

Above the $h_p = 0$ locus. A second vapor-region subregime may arise above this locus. In that case, we may have

$$K_p < 0, \qquad h_p < 0, \qquad T_p > 0, \qquad \rho_p > 0. \tag{49}$$

Then the sign of $\beta_p = -\dot{m}_{in} h_p - \alpha A T_p$ is no longer fixed *a priori*, since the first term is positive while the second term remains negative. Consequently, $\beta_x < 0$ still guarantees $\Delta > 0$, but τ is no longer automatically negative and must be checked explicitly from (41).

In summary, strictly single-phase equilibria are always non-saddle under the present assumptions, because $\beta_x < 0$ implies $\Delta > 0$. In the liquid region, local asymptotic stability follows if the additional approximation $\rho_p \approx 0$ is adopted. In the vapor region, local asymptotic stability is automatic below the $h_p = 0$ locus, but must be checked explicitly once h_p becomes negative.

3.5 Two-phase equilibria

The two-phase region admits additional structure that is useful for interpreting the local stability conditions. Under the present heat-transfer approximation, the derivative $\alpha_x = d\alpha/dx$ vanishes over a large portion of the two-phase region and becomes significant only in narrow transition regions near the liquid-to-two-phase and two-phase-to-vapor boundaries. Accordingly, the bulk two-phase region and the phase-boundary regions should be treated separately.

Fig. 2 also indicates that the sign of h_p may change along a curve in the p–h plane rather than at a single pressure. Consequently, both the bulk two-phase region

DOI
10.3384/ecp223113

and the high-quality transition region may admit lower-pressure and higher-pressure subregimes with different trace behavior.

For general two-phase equilibria, we retain

$$\beta_x = -\dot{m}_{in} h_x + \alpha_x A(T_w - T) - \alpha A T_x, \tag{50}$$

with

$$h_x > 0, \qquad T_x \geq 0. \tag{51}$$

The determinant is still given by (37), so the local saddle versus non-saddle character is determined entirely by the sign of β_x.

The trace is given by (41), with

$$\beta_p = -\dot{m}_{in} h_p - \alpha A T_p. \tag{52}$$

Define

$$\xi \triangleq \dot{m}_{in} h_x + \alpha A T_x > 0. \tag{53}$$

Then (50) becomes

$$\beta_x = -\xi + \alpha_x A(T_w - T). \tag{54}$$

The determinant changes sign when $\beta_x = 0$, which gives the critical gradient

$$\alpha_x^{\Delta} = \frac{\xi}{A(T_w - T)}. \tag{55}$$

The trace becomes zero when

$$\beta_x = \beta_x^{\tau} \triangleq \frac{\Lambda \beta_p - \rho K_p}{\Upsilon \rho_p}, \qquad \rho_p \neq 0. \tag{56}$$

Combining (54) and (56), the corresponding trace-zero boundary is

$$\alpha_x^{\tau} = \alpha_x^{\Delta} + \frac{\beta_x^{\tau}}{A(T_w - T)}. \tag{57}$$

Thus, α_x^{Δ} marks the saddle boundary, whereas α_x^{τ} marks the Hopf candidate boundary. If α_x^{τ} lies beyond α_x^{Δ} in the destabilizing direction, the equilibrium changes directly from stable to saddle and no Hopf boundary appears first. If α_x^{τ} is reached before α_x^{Δ}, then the system first crosses a trace-zero boundary while $\Delta > 0$, which is the linear signature of a Hopf candidate. Establishing a true finite-amplitude limit cycle would require further nonlinear analysis.

3.5.1 Bulk two-phase region

Over a large portion of the two-phase region, the heat transfer coefficient is approximated as constant with respect to x, so that

$$\alpha_x = 0. \tag{58}$$

Then $\beta_x = -\xi < 0$, and therefore $\Delta > 0$. Hence, equilibria in the bulk two-phase region are always non-saddle.

The trace behavior in the bulk two-phase region depends on the sign of h_p.

Below the $h_p = 0$ locus. In the lower-pressure part of the bulk two-phase region, that is, below the $h_p = 0$ locus, we observe

$$h_p > 0, \qquad \rho_p > 0, \qquad T_p > 0. \tag{59}$$

Then $\beta_p < 0$. Since $\beta_x < 0$, all three terms in (41) are negative, and therefore $\tau < 0$. Thus, bulk two-phase equilibria are locally asymptotically stable in this subregime.

Above the $h_p = 0$ locus. At higher pressures, above the $h_p = 0$ locus, we may instead have

$$h_p < 0, \qquad \rho_p > 0, \qquad T_p > 0. \tag{60}$$

Then β_p is no longer sign-definite. Therefore, while $\Delta > 0$ remains guaranteed, the sign of τ must be checked explicitly. In this higher-pressure bulk two-phase subregime, the equilibrium may therefore be stable, unstable, or marginal, but it cannot be a saddle.

3.5.2 Low-quality transition region

In the low-quality transition region, the heat transfer coefficient increases from its liquid-phase value to its two-phase value, and therefore

$$\alpha_x > 0. \tag{61}$$

For the operating regimes considered here, we further observe

$$h_p > 0, \qquad \rho_p > 0, \qquad T_p > 0, \tag{62}$$

which implies $\beta_p < 0$.

Substituting (61) into (54) shows that the gradient term is destabilizing only under heating,

$$T_w > T, \tag{63}$$

because then $\alpha_x A(T_w - T) > 0$. Under cooling ($T_w < T$), the same term is negative and therefore further decreases β_x, which reinforces stability.

Accordingly, in the low-quality region under heating, saddle instability may occur if

$$\alpha_x > \alpha_x^{\Delta}. \tag{64}$$

Moreover, because $\rho_p > 0$, $\beta_p < 0$, and $\Lambda < 0$, one has

$$\beta_x^{\tau} > 0. \tag{65}$$

Hence the trace-zero condition lies outside the non-saddle region $\beta_x < 0$. Equivalently, in the destabilizing direction,

$$\alpha_x^{\tau} > \alpha_x^{\Delta} \qquad (T_w > T), \tag{66}$$

so the equilibrium changes directly from stable to saddle. Therefore, in the low-quality transition region there is no Hopf boundary before saddle instability.

3.5.3 High-quality transition region

In the high-quality transition region, the heat transfer coefficient decreases from its two-phase value to its vapor-phase value, and therefore

$$\alpha_x < 0. \tag{67}$$

In this case, the gradient term in (54) is destabilizing only under cooling,

$$T_w < T, \tag{68}$$

because then $\alpha_x A(T_w - T) > 0$. Under heating ($T_w > T$), the same term is negative and again contributes to keeping $\beta_x < 0$.

Accordingly, in the high-quality region under cooling, saddle instability may occur if

$$\alpha_x < \alpha_x^{\Delta}, \tag{69}$$

since now both α_x and α_x^{Δ} are negative.

The trace behavior in the high-quality region depends on the sign of h_p.

Lower-pressure subregime ($h_p > 0$). In the lower-pressure part of the high-quality region, we may still have

$$h_p > 0, \qquad \rho_p > 0, \qquad T_p > 0. \tag{70}$$

Then $\beta_p < 0$, and as in the low-quality case one has $\beta_x^{\tau} > 0$. Hence the trace-zero boundary again lies beyond the saddle boundary in the destabilizing direction. Equivalently,

$$\alpha_x^{\tau} < \alpha_x^{\Delta} \qquad (T_w < T), \tag{71}$$

so the equilibrium changes directly from stable to saddle as α_x becomes more negative. Therefore, no Hopf boundary appears before saddle instability in this subregime.

Higher-pressure subregime ($h_p < 0$). At sufficiently high pressures, above the pressure at which the slope of the saturated-vapor line in the p–h plane changes sign, the constant-x pressure derivative of enthalpy may become negative:

$$h_p < 0, \qquad \rho_p > 0, \qquad T_p > 0. \tag{72}$$

In this case, the pressure sensitivity of the energy balance,

$$\beta_p = -\dot{m}_{\mathrm{in}} h_p - \alpha A T_p,$$

is no longer sign-definite. The first term is now positive, while the second term remains negative. Therefore, higher pressure does not by itself imply a Hopf-type loss of stability; it only makes such a possibility available if h_p is sufficiently negative.

The determinant condition is unchanged: the equilibrium is non-saddle only when

$$\Delta > 0 \quad \Longleftrightarrow \quad \beta_x < 0.$$

A Hopf candidate can occur only if the trace-zero boundary lies inside this non-saddle region. Since

$$\beta_x^{\tau} = \frac{\Lambda \beta_p - \rho K_p}{\Upsilon \rho_p},$$

the necessary condition for a Hopf candidate is

$$\beta_x^{\tau} < 0. \tag{73}$$

Using $\Upsilon > 0$, $\rho_p > 0$, $K_p < 0$, and $\Lambda < 0$, this condition is equivalent to

$$\beta_p > \frac{\rho K_p}{\Lambda}. \tag{74}$$

Because both K_p and Λ are negative, the right-hand side of (74) is positive. Thus β_p must not only become positive; it must become sufficiently positive. Substituting the definition of β_p, this requirement can be written as

$$-\dot{m}_{\mathrm{in}} h_p - \alpha A T_p > \frac{\rho K_p}{\Lambda}, \tag{75}$$

or equivalently,

$$h_p < -\frac{1}{\dot{m}_{\mathrm{in}}} \left(\alpha A T_p + \frac{\rho K_p}{\Lambda} \right). \tag{76}$$

Equation (76) shows that a Hopf candidate requires a rather restrictive condition: h_p must be sufficiently negative so that the pressure-induced enthalpy feedback overcomes both the stabilizing temperature-sensitivity term and the additional positive threshold imposed by the trace condition.

Consequently, three cases are possible in this higher-pressure subregime:

- If $\beta_x^{\tau} > 0$, the trace-zero boundary lies outside the non-saddle region. The equilibrium therefore remains stable until the determinant changes sign, and the loss of stability occurs through the saddle boundary.
- If $\beta_x^{\tau} = 0$, the trace-zero and determinant-zero boundaries coincide. This gives a codimension-two threshold at which $\tau = 0$ and $\Delta = 0$ occur simultaneously.
- If $\beta_x^{\tau} < 0$, the trace-zero condition is reached while $\Delta > 0$. In this case, the equilibrium crosses a Hopf candidate boundary before it reaches the saddle boundary.

For the high-quality transition under cooling, $T_w < T$, the destabilizing direction corresponds to α_x decreasing from zero toward more negative values. The determinant-zero boundary is reached at α_x^{Δ}, while the trace-zero boundary is reached at α_x^{τ}. Since the boundary closer to zero is encountered first, the ordering determines the transition sequence:

$$\alpha_x^{\tau} > \alpha_x^{\Delta} \implies \text{stable} \rightarrow \text{Hopf candidate} \rightarrow \text{saddle},$$

DOI
10.3384/ecp223113

$$\alpha_x^\tau = \alpha_x^\Delta \implies \text{coincident trace/determinant boundary},$$
$$\alpha_x^\tau < \alpha_x^\Delta \implies \text{stable} \to \text{saddle}.$$

Equivalently, in this cooling case a Hopf candidate can occur only when the trace-zero boundary is reached before the determinant-zero boundary as α_x moves in the destabilizing direction. Here the term "Hopf candidate" is used only in the linearized sense ($\Delta > 0, \tau = 0$). Establishing a true Hopf bifurcation, and hence the existence and stability of a self-sustained limit cycle, would require further nonlinear analysis, such as checking the transversality condition and the first Lyapunov coefficient.

3.5.4 Summary regime map

The results above can be summarized as a regime map in the p–h plane, shown schematically in Fig. 5. The map organizes the local stability picture according to phase region, the sign of α_x, the destabilizing heat-transfer direction, and the resulting local mechanism.

In the strictly single-phase liquid region, $\alpha_x = 0$, so $\beta_x < 0$ and the equilibrium is non-saddle under the present approximation. In the single-phase vapor region below the $h_p = 0$ locus, $\alpha_x = 0$ likewise gives $\Delta > 0$, and the trace is automatically negative in the operating regime considered here, so the equilibrium is locally asymptotically stable. Above the $h_p = 0$ locus, the equilibrium remains non-saddle, but the trace must be checked explicitly because β_p is no longer sign-definite.

Within the two-phase region, the bulk regime is characterized by $\alpha_x = 0$, so the equilibrium is again non-saddle. Below the $h_p = 0$ locus, the trace is negative and the equilibrium is locally asymptotically stable. At higher pressures, however, the trace must again be checked explicitly. By contrast, the low-quality and high-quality transition regions are distinguished by nonzero heat-transfer gradients. In the low-quality transition region, $\alpha_x > 0$, and the destabilizing situation is heating, $T_w > T$. In the high-quality transition region, $\alpha_x < 0$, and the destabilizing situation is cooling, $T_w < T$. In both cases, sufficiently steep phase-boundary variation of the heat transfer coefficient can drive a direct stable-to-saddle transition through the determinant boundary.

A further distinction arises in the higher-pressure high-quality regime, where $h_p < 0$ and β_p is no longer sign-definite. In that case, the trace-zero boundary may in principle be reached before the determinant-zero boundary, so a Hopf candidate becomes theoretically possible. As discussed above, this requires more restrictive conditions than the ordinary determinant-driven saddle transition and, in the present model, appears to require operating points close to the critical region.

4 Modelica-Based Case Studies and Numerical Verification

This section presents Modelica-based case studies that verify the analytical stability conditions developed in Section 3. The four cases are selected to mirror the regime structure summarized in Fig. 5, and their schematic locations in the p–h plane are shown in Fig. 6: Case 1 considers the bulk two-phase stable regime, Cases 2 and 3 examine the low-quality and high-quality transition regions where determinant-driven saddle instability may occur, and Case 4 probes the higher-pressure regime in which a Hopf candidate is theoretically possible.

Table 1. Model parameters common to all case studies.

Parameter	Value
Refrigerant	R410A
Control-volume size V [m^3]	3.927e-5
Heat-transfer area A [m^2]	1.571e-2
Reference mass flow rate $\dot{m}_0$ [kg/s]	0.02
Reference pressure drop Δp_0 [Pa]	1e3
$\dot{m}_{\text{in}}$ [kg/s]	0.02
α_{liq} [W/(m^2 K)]	600
α_{vap} [W/(m^2 K)]	300
Low-quality blending interval	[0, 0.05]
High-quality blending interval	[0.95, 1]

4.1 Implementation and evaluation procedure

The refrigerant is taken to be R410A, and all thermodynamic properties and their derivatives are evaluated from the same property model used throughout the paper. The model is implemented in Modelica and simulated in Dymola. For simplicity, the phase-boundary variation of the heat-transfer coefficient is represented by linear blending in the low-quality and high-quality intervals.

For each case, the control-volume geometry, inlet conditions, outlet restriction, and wall temperature are fixed so that an equilibrium point (p_e, x_e) exists. The principal model parameters common to all four cases are listed in Table 1, while the case-dependent heat-transfer coefficients and operating conditions are summarized in Table 2.

The local quantities

$$h_p, \quad h_x, \quad \rho_p, \quad \rho_x, \quad T_p, \quad T_x, \quad \beta_p, \quad \beta_x, \quad \Delta, \quad \tau$$

are then evaluated at the equilibrium. A small perturbation is subsequently applied to the state, and the nonlinear response is compared with the local prediction from the Jacobian.

4.2 Case 1: Stable equilibrium in the bulk two-phase region

The first case considers an equilibrium located in the bulk two-phase region, away from the phase boundaries. Under the present heat-transfer approximation, this corresponds to $\alpha_x = 0$. The selected equilibrium lies below the $h_p = 0$ locus, so that $h_p > 0$, $\rho_p > 0$, and $T_p > 0$. As shown in Section 3, this implies $\beta_x < 0$, hence $\Delta > 0$, and the trace is also negative. The equilibrium is therefore predicted to be locally asymptotically stable.

The numerical simulations confirm this prediction. Small perturbations in both p and x decay in time, and the trajectory returns to the original equilibrium. Fig. 7 shows

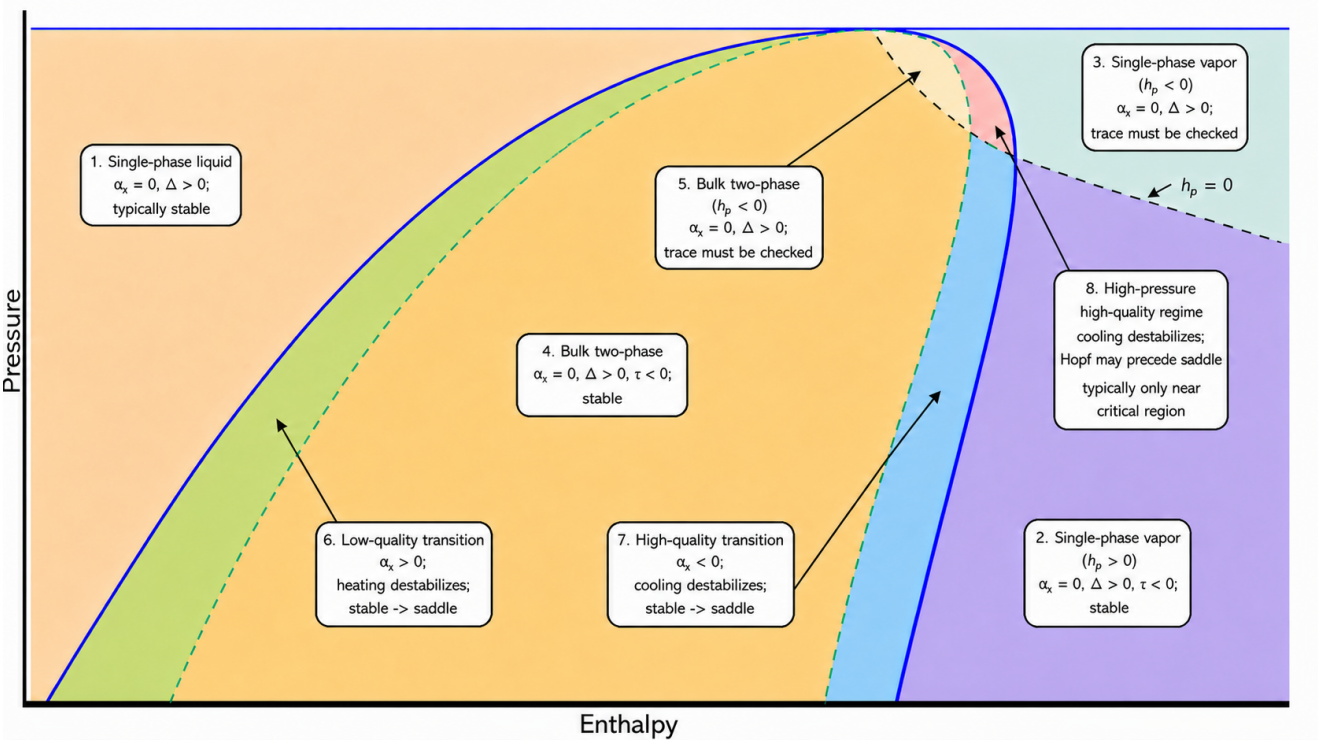


Figure 5. Summary regime map of the local stability mechanisms predicted by the closed-form analysis.

Table 2. Case-dependent operating conditions and equilibrium states.

Parameter	Case 1	Case 2	Case 3	Case 4
α_{2ph} [W/(m^2 K)]	1e4	3e3	3e3	6e3
h_{in} [kJ/kg]	237.15	192.6	423.8	404.0
p_{sink} [bar]	0.7	0.7	1.0	4.5
T_w [°C]	5	1	-1	62

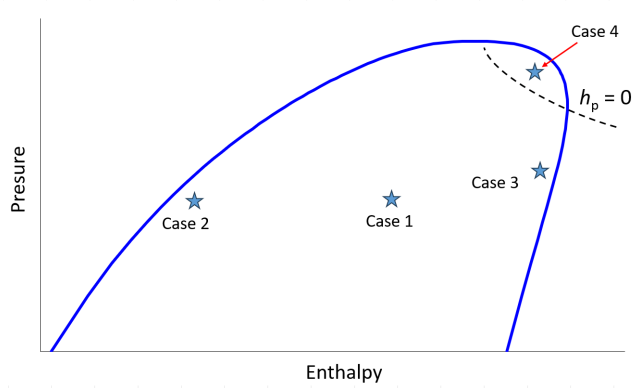


Figure 6. Schematic locations of the four simulation cases in the p–h plane.

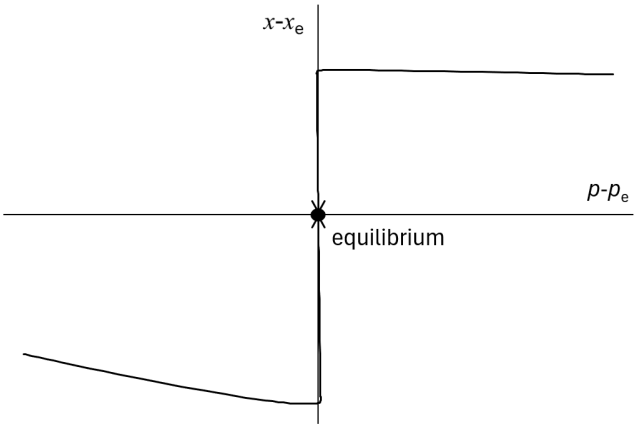


Figure 7. Phase portrait for Case 1 in the bulk two-phase region.

the corresponding phase portrait in the $(p - p_e,\ x - x_e)$ plane for two representative initial perturbations. In both cases, the trajectories converge back to the same equilibrium, confirming that the operating point is locally attracting.

The phase portrait also indicates a clear separation of time scales near the equilibrium. The trajectories approach the equilibrium much more rapidly in the pressure direction, whereas the phase coordinate relaxes more slowly. Thus, for this operating point, the pressure response is faster than the phase-coordinate response.

DOI
10.3384/ecp223113

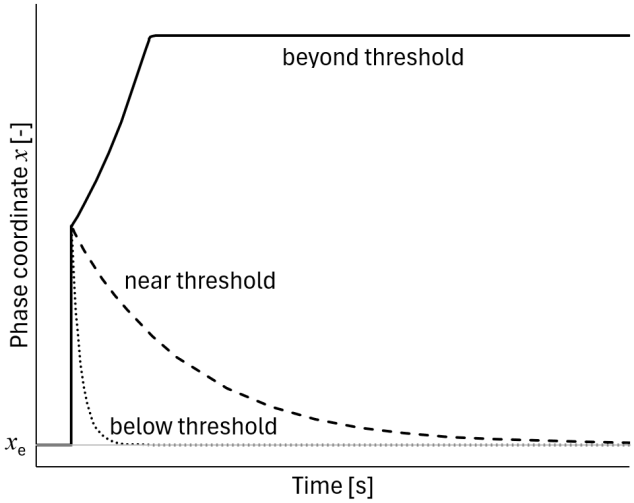
4.3 Case 2: Low-quality transition region under heating

The second case considers an equilibrium in the low-quality transition region, where the heat-transfer coefficient increases from its liquid-phase value to its two-phase value, so that $\alpha_x > 0$. The destabilizing situation is heating, $T_w > T$, because the gradient contribution $\alpha_x A(T_w - T)$ enters β_x with a positive sign. Under the sign pattern $h_p > 0$, $\rho_p > 0$, and $T_p > 0$, the analysis in Section 3 predicts that the trace remains negative throughout the non-saddle region. Hence, as α_x increases, the expected transition is a direct loss of local stability through the determinant boundary α_x^{Δ}, that is, from a stable equilibrium to a saddle.

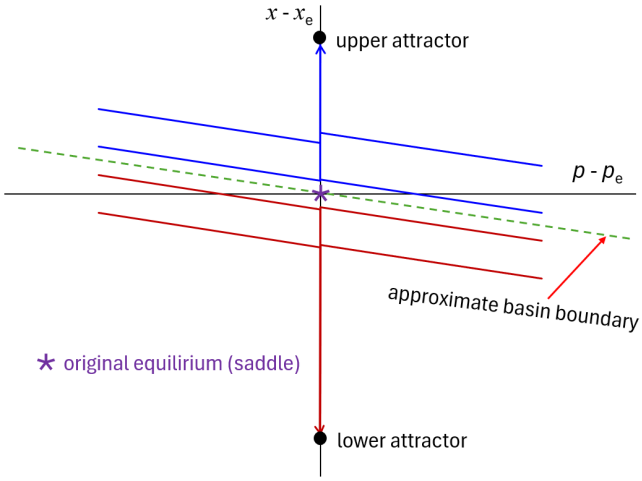
To verify this prediction, the equilibrium is kept in the same low-quality region while the width of the linear blending interval is reduced in the Modelica implementation. This increases the local heat-transfer gradient without changing the thermodynamic regime. Three representative conditions are then considered: one below the determinant threshold, one near it, and one beyond it. Below the threshold, $\beta_x < 0$ and $\Delta > 0$, so the equilibrium remains locally asymptotically stable and a small perturbation returns to the original operating point. Near the threshold, β_x approaches zero and Δ becomes small, so the recovery becomes slow. Beyond the threshold, $\beta_x > 0$ and $\Delta < 0$, and the original equilibrium becomes a saddle.

Fig. 8 summarizes the resulting nonlinear behavior. The time responses in Fig. 8(a) show the three regimes clearly: return to the original equilibrium below the threshold, slow recovery near the threshold, and loss of attraction beyond the threshold. The phase portrait in Fig. 8(b) shows that, after the determinant threshold is crossed, the original equilibrium at (p_e, x_e) no longer attracts nearby trajectories. Instead, the trajectories separate into two groups and converge to an upper or lower attractor. The dashed line indicates the approximate basin boundary inferred from the simulations. In this case, the boundary is not aligned with the coordinate axes, so both pressure and phase-coordinate perturbations influence which attractor is reached.

Thus, Case 2 confirms the determinant-driven stable-to-saddle transition predicted analytically for the low-quality transition region under heating.



(a) Time responses of x for representative operating conditions.



(b) Phase portrait in the $(p - p_e,\ x - x_e)$ plane for a beyond-threshold condition.

Figure 8. Case 2 in the low-quality transition region under heating.

4.4 Case 3: High-quality transition region under cooling

The third case is constructed in a manner parallel to Case 2, but in the high-quality transition region, where the heat-transfer coefficient decreases from its two-phase value to its vapor-phase value, so that $\alpha_x < 0$. The destabilizing situation is now cooling, $T_w < T$, because the gradient contribution $\alpha_x A(T_w - T)$ is then again positive. As discussed in Section 3, sufficiently large negative heat-transfer gradients can drive the determinant through zero and cause the original equilibrium to lose local attraction.

The main point of this case, however, is not merely the threshold crossing itself, but the richer nonlinear behavior that appears beyond it. In the simulations considered here, a beyond-threshold parameter setting is chosen and the model is initialized with two nearby but distinct phase-coordinate perturbations while all other parameters are kept unchanged. Fig. 9 shows the corresponding time responses of the phase coordinate. Although both trajectories start in the neighborhood of the same original equilibrium x_e, they evolve toward different final operating points.

This result indicates that the nonlinear system admits multiple accessible steady equilibria under the same parameter setting. In other words, once the original equilibrium has lost local attraction, the long-time behavior is no longer determined by the parameters alone, but also by the initial condition. The upper and lower final states shown in Fig. 9 therefore represent two distinct attractors in the

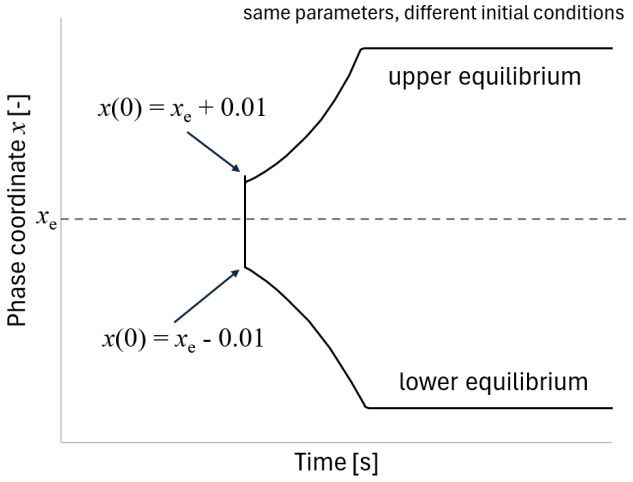


Figure 9. Case 3: time responses of the phase coordinate for identical parameters and different initial conditions.

beyond-threshold regime.

Thus, Case 3 confirms the same determinant-driven loss of local stability identified analytically for the high-quality transition region under cooling. In particular, the coexistence of multiple attractors implies multiple basins of attraction and shows that the post-threshold dynamics cannot be characterized by a single attracting operating point.

4.5 Case 4: Higher-pressure regime

The fourth case examines the higher-pressure regime in which $h_p < 0$, so that β_p is no longer sign-definite. In this subregime, the determinant is still controlled by β_x, but the trace must be checked explicitly. As shown in Section 3, this change in trace structure opens the possibility that the trace-zero boundary may be reached before the determinant-zero boundary. Accordingly, three scenarios are theoretically possible: a direct stable-to-saddle transition, a coincident trace-zero and determinant-zero boundary, or a Hopf candidate if the trace-zero boundary is encountered while $\Delta > 0$.

The purpose of this case is therefore not simply to reproduce another stable-to-saddle transition, but to assess whether the higher-pressure regime can bring the trace-zero boundary α_x^τ close to the determinant-zero boundary α_x^Δ. To this end, the operating point is moved into a region where $h_p < 0$, and the corresponding local stability quantities and nonlinear response are then assessed.

The numerical exploration shows that, in the operating range considered here, the determinant-zero boundary is reached well before the trace-zero boundary. Equivalently, as α_x decreases from zero toward more negative values, α_x^Δ is encountered before α_x^τ. In this cooling case, this corresponds to

$$\alpha_x^\tau < \alpha_x^\Delta < 0.$$

Bringing the two thresholds closer would require h_p to become sufficiently small and strongly negative, so that the pressure-induced enthalpy feedback makes β_p large enough to satisfy the trace-zero condition within the non-saddle region. In the present model, this appears to occur only very near the critical region. Thus, although a Hopf candidate is theoretically possible in the higher-pressure regime, the dominant numerically accessible mechanism in the present study remains the direct determinant-driven stable-to-saddle transition.

5 Conclusions

This paper developed a pressure–phase-coordinate control-volume model and derived closed-form local stability conditions for compressible two-phase flow. The main analytical result is that saddle versus non-saddle behavior is determined directly by the sign of β_x, the phase-coordinate sensitivity term in the energy balance.

The analysis shows that strictly single-phase equilibria and the bulk two-phase region are non-saddle under the present approximation. By contrast, the numerical case studies show that the dominant instability mechanism in the operating range considered here is localized near phase boundaries: sharp variation of the heat transfer coefficient near the liquid-to-two-phase and two-phase-to-vapor transitions can drive a direct stable-to-saddle transition.

The simulations further show that loss of local stability need not appear as unbounded divergence. Instead, the state may depart from the original equilibrium and converge to a different stable operating point, and in some beyond-threshold regimes multiple attractors may coexist.

A Hopf candidate is theoretically possible in a higher-pressure regime where $h_p < 0$ and β_p is no longer sign-definite, but this requires more restrictive conditions than the determinant-driven saddle transition and was not found to be the dominant behavior in the cases studied here.

From a modeling and control perspective, the derived stability conditions provide a compact local diagnostic for identifying sensitive operating regions in distributed thermal-fluid models. In particular, the sign of β_x can be monitored along a nominal operating trajectory to determine whether an individual control volume remains non-saddle or approaches a determinant-driven loss of local attraction. Operating points with β_x close to zero are of particular practical interest because the determinant becomes small, the local recovery becomes slow, and the model may become more sensitive to disturbances, parameter uncertainty, and numerical regularization. This information can support operating-envelope selection, heat-transfer-model regularization, spatial-discretization decisions, and gain-scheduled, supervisory, or optimization-based control design. The criterion should nevertheless be interpreted as a local screening tool rather than as a complete stability condition for the full vapor-compression cycle.

The present result is also qualitatively consistent with a common observation in cycle-level simulations, namely that oscillatory behavior is more often seen in the high-

DOI
10.3384/ecp223113

pressure side than in the evaporating pressure. A possible reason is that condenser models often traverse a large portion of the two-phase region, including the phase-boundary transition regions identified here as locally destabilizing, whereas evaporator inlet states may already be beyond the low-quality transition region and therefore spend less of the evaporation process in that locally sensitive zone.

Although the analysis is carried out in pressure–phase coordinates, the same local instability mechanism is expected to persist in more conventional pressure–enthalpy formulations and in larger distributed models built from the same control-volume and flow-restriction ingredients.

References

Eborn, Jonas and Karl J. Åström (2000). "Modeling of a Boiler Pipe with Two-Phase Flow Instabilities". In: *Modelica Workshop 2000 Proceedings*. Lund, Sweden, pp. 79–88.

Kakaç, S. and B. Bon (2008). "A Review of Two-Phase Flow Dynamic Instabilities in Tube Boiling Systems". In: *International Journal of Heat and Mass Transfer* 51.3–4, pp. 399–433.

Laughman, Christopher and Hongtao Qiao (2018). "On Closure Relations for Dynamic Vapor Compression Cycle Models". In: *American Modelica Conference*.

Laughman, Christopher, Hongtao Qiao, Scott Bortoff, and Vedang Deshpande (2026). "On State Selection for Vapor Compression Cycle Models with Refrigerant Mixtures". In: *21st International Refrigeration and Air-Conditioning Conference at Purdue*.

Qiao, Hongtao, Vikrant Aute, and Reinhard Radermacher (2015). "Transient Modeling of a Flash Tank Vapor Injection Heat Pump System – Part I: Model Development". In: *International Journal of Refrigeration* 49, pp. 169–182. ISSN: 0140-7007.

Qiao, Hongtao, Takashi Kobayashi, Christopher Laughman, and Scott Bortoff (2023). "On the Characteristics of One-Dimensional Compressible Flow". In: *The 15th International Modelica Conference*.

Qiao, Hongtao and Christopher Laughman (2021). "Modeling and Analysis of Pressure Drop Oscillations in Horizontal Boiling Flow". In: *Proceedings of the 18th International Refrigeration and Air Conditioning Conference at Purdue*. Paper 2190.

Ruspini, Leonardo C., Christian P. Marcel, and Alejandro Clausse (2014). "Two-Phase Flow Instabilities: A Review". In: *International Journal of Heat and Mass Transfer* 71, pp. 521–548.

Yadigaroglu, G. (1981). "Two-Phase Flow Instabilities and Propagation Phenomena". In: *Thermal-Hydraulics of Two-Phase Systems for Industrial Design and Nuclear Engineering*. Ed. by J. M. Delhaye, M. Giot, and M. L. Riethmuller. New York: Hemisphere Publishing Corporation. Chap. 17.

DOI
10.3384/ecp223113

A Brief Methodological Reconsideration of Modelica-Based Hydraulic System Modeling and Xu's Method

Bo Wang[1] YangZeng Xu[1]

[1]Guangdong Xu Yangzeng Digital Hydraulic Research Institute, China,
wangbo.hualin@163.com 672445327@qq.com

Abstract

This paper briefly reviews the engineering development of Modelica-based hydraulic system modeling. On this basis, it reconsiders the issue of reasonable decomposition in component-based hydraulic modeling and discusses the methodological relevance of Xu's method for the development of Modelica hydraulic model libraries. Fine-grained component decomposition facilitates rapid assembly and reuse, especially for users without extensive hydraulic modeling expertise. However, excessive subdivision may also introduce duplicated pressure states, hidden volumes, artificial fast time constants, increased numerical stiffness, and ambiguous physical semantics. Through order-of-magnitude examples involving pilot-operated relief valves, pressure compensators in multi-way valves, short manifold passages, hydraulic cylinder chambers, and accumulator groups, the paper illustrates that the appropriate component boundary depends on the modeling objective, the dominant physical mechanism, the available parameters, and the quantities to be validated.

Keywords: Modelica; Hydraulics System Modeling ; Xu's Method ;

1 Introduction

1.1 Overview

The idea of representing complex engineering systems through interacting components has a long historical foundation. In classical mechanics, d'Alembert's principle made it possible to formulate dynamic behavior in terms of generalized equilibrium. In circuit theory, Kirchhoff's laws established conservation-based interconnection constraints for lumped electrical networks. These traditions provided early examples of how system behavior can be described through local component relations and global connection laws.

During the twentieth century, cross-domain analogies further extended this way of thinking. Mechanical, electrical, hydraulic, pneumatic, and thermal systems were increasingly described through analogous effort-flow or across-through variables. Equivalent-circuit methods and bond-graph theory made this analogy explicit by treating energy exchange through ports as a common modeling structure across different physical domains. In the field of fluid power, systematic modeling traditions also developed around the analysis of hydraulic and pneumatic components, where pressure, flow rate, force, displacement, and motion must be represented in a coupled manner.

Modelica inherits and extends this component-based physical modeling tradition. As an equation-based, acausal, object-oriented modeling language, Modelica allows components to be described by physical equations rather than by fixed input-output causality. This makes it particularly suitable for multi-domain systems, where hydraulic, mechanical, electrical, thermal, and control subsystems interact through bidirectional physical relations. In this context, the central issue is not merely whether a system should be decomposed into components, but how the boundaries, states, and abstraction levels of these components should be defined.

1.2 The Development of Modelica-Based Hydraulic System Modeling

Modelica has been widely applied in many engineering domains, with particularly significant developments in thermo-fluid and fluid-power modeling. The development of reusable fluid interfaces, media models, stream connectors, and related library architectures provided important foundations for modeling pressure-flow-energy interactions in component-oriented system libraries. These developments also influenced the modeling of hydraulic systems, where pressure, flow rate, oil compressibility, valve characteristics, actuator motion, and control dynamics must be handled within a unified framework.

The early stage of Modelica-based hydraulic modeling was mainly concerned with the construction of reusable component libraries and the demonstration of multi-domain simulation capability. HyLib organized hydraulic components such as pumps, valves, cylinders, orifices, pipelines, accumulators, and check valves into reusable component models for engineering design and education [4]. These early libraries enabled hydraulic circuits to be built in a form close to engineering schematics, making Modelica accessible to engineers familiar with hydraulic circuit diagrams. At the same time, hydraulic subsystems began to be combined with mechanical structures,

electrical systems, control logic, and real-time simulation environments. Representative examples include the electronic-hydraulic-mechanical system of a universal testing machine [8], Toyota's real-time integration study involving engines, transmissions, hydraulics, and braking systems [25], HIL simulation of the hydraulics and mechanics of an automatic gearbox [22], and object-oriented inverse modeling of multi-domain aircraft equipment systems [2].

A second stage can be characterized by the expansion of Modelica-based hydraulic modeling into engineering applications where hydraulic components were no longer isolated models, but parts of larger multi-domain systems. Representative studies include mechanical-hydraulic simulation of an excavator [5], the development of a simplified transmission hydraulics library based on Modelica.Fluid [26], automotive hydraulic system modeling using the ActuationHydraulics library [12], and HIL simulation of an aircraft thrust reverser hydraulic system [31]. These studies show that Modelica-based hydraulic modeling gradually moved from library construction and circuit-level assembly toward engineering applications involving multi-domain coupling, real-time execution, and control validation.

Since the 2010s, the research scope has further expanded toward component-level physics, high-dynamic hydraulic phenomena, and engineering validation. Examples include real-time model reduction for construction machines [19], co-simulation of a hydraulic braking system with a finite-element tire model [23], strong coupling between Modelica system-level models and detailed CFD models for transient simulation of hydraulic components [29], control-volume modeling of gear pumps [7], wear modeling of hydraulic axial piston pumps [3], hydromechanical double-clutch actuators [20], hydraulic shock absorber models [13], hydro-mechanical synchronization systems for double-clutch transmissions [14], hydro-mechanical systems in an OpenModelica real-time toolchain [30], power- or bond-graph-based modeling of industrial hydraulic plants [1], physical design models of hydraulic valves [6], pump efficiency estimation based on the Euler number [11], architecture validation of hydraulic brake systems [16], and data reconciliation for industrial hydraulic experiments [17].

Taken together, some works suggest that the development of Modelica-based hydraulic modeling is not a simple movement from coarse models to ever finer component decomposition. Rather, it reflects a gradual diversification of modeling purposes. Some applications require detailed component-level models for design, diagnosis, or local dynamic analysis, whereas others require functional, reduced-order, or real-time models for system-level simulation and control validation. The development of Modelica hydraulic libraries should therefore be understood not only as an increase in the number of available components, but also as an evolution in model hierarchy, component boundary definition, and verification practice.

1.3 Problem Statement and Paper Organization

Although component-based modeling is one of the main strengths of Modelica, component decomposition itself does not automatically guarantee a physically meaningful or numerically efficient model. In hydraulic systems, the definition of component boundaries is closely related to the placement of pressure states, the representation of oil compressibility, the treatment of valve openings and local volumes, and the numerical stiffness of the resulting equation system. The central question considered here is how reasonable component boundaries can be defined in Modelica-based hydraulic libraries when the physical structure of hydraulic components does not always coincide with the most appropriate numerical model structure.

The paper is organized as follows. Section 2 summarizes several recurring challenges in Modelica-based hydraulic modeling, especially ambiguous decomposition boundaries, duplicated states, numerical stiffness, parameter uncertainty, and reproducibility. Section 3 revisits Xu's method as a hydraulic-domain-oriented modeling philosophy. Section 4 then uses representative hydraulic structures to discuss how reasonable decomposition can be estimated and judged from the viewpoints of physical mechanism, dominant time scale, available parameters, and validation objective. Section 5 concludes with implications for the design of Modelica hydraulic libraries.

2 Challenges in Modelica-Based Hydraulic Modeling

2.1 Ambiguous Decomposition Boundaries and State Placement

A central unresolved issue in Modelica-based hydraulic modeling is the lack of a unified criterion for defining reasonable decomposition boundaries. Hydraulic valves, pilot circuits, short manifold passages, actuator chambers, pump chambers, accumulator groups, and closed hydraulic networks may require different decomposition strategies depending on the simulation objective. For some applications, high-fidelity component-level models are necessary; for others, functional models or equivalent models may be more appropriate.

This ambiguity is reflected in several strands of research. Physical design models of hydraulic valves show the importance of detailed geometry and component-level parameterization when the purpose is valve design and dynamic characteristic analysis [6]. Strong coupling between Modelica system-level models and CFD models shows that local three-dimensional flow details may be important for certain hydraulic

DOI
10.3384/ecp223127

components [29]. Conversely, industrial hydraulic plant modeling based on generalized power variables emphasizes the system-level energy structure rather than detailed geometric decomposition [1]. Research on dynamic overconstrained connectors further shows that closed incompressible hydraulic systems can produce structurally difficult connection problems when valve switching changes the effective topology of the network [27].

Therefore, the reasonable boundary of a hydraulic model cannot be defined by geometry alone. It must be determined by the intended use of the model, the required time scale, the available parameters, the desired numerical robustness, and the physical quantities to be validated. The core issue in Modelica-based hydraulic modeling has thus shifted from how to connect more components quickly to how to define appropriate model boundaries, state locations, reduction strategies, parameterization methods, and verification frameworks according to engineering objectives.

2.2 Numerical Stiffness, Small Volumes, and Duplicated States

The relevant Modelica Conference papers show that hydraulic modeling has long faced numerical difficulties associated with high bulk modulus, small fluid volumes, strong throttling, valve switching, friction, and rapid pressure-wave propagation. These characteristics can easily lead to strong stiffness, dense events, ill-conditioned algebraic loops, initialization difficulties, and, in some cases, high-index DAE systems. Such issues are already visible in early multi-domain hydraulic models, including the universal testing-machine model [8], Toyota's HILS integration-method study [25], the simplified transmission hydraulics library [26], and automotive hydraulic applications using ActuationHydraulics [12].

A particularly important source of numerical difficulty is excessive subdivision of hydraulic volumes and restrictions. If every small cavity, short passage, valve opening, leakage path, or local restriction is modeled as an independent component, the model may contain duplicated pressure states and unnecessary compliance effects. This can make the mathematical structure more complicated without necessarily improving the system-level fidelity of the model. In this sense, the numerical behavior of a hydraulic model is closely related to the modeling decision of where pressure states are introduced and where hydraulic capacitance is represented.

The transmission line model provides another perspective on this issue. Instead of representing long fluid lines only by repeated lumped-parameter segments, transmission-line-based modeling offers an alternative approach for fast fluid-line dynamics and transient friction [28]. This suggests that the appropriate representation of hydraulic lines, volumes, and wave dynamics depends on the time scale and physical objective of the simulation, rather than on a fixed rule of geometric subdivision.

2.3 Parameter Availability and Physical Credibility

A more detailed model is not necessarily a more credible model. Hydraulic components often contain valve openings, leakage paths, small chambers, friction surfaces, manufacturing tolerances, and flow coefficients that are difficult to identify from available engineering data. When these quantities are unknown or only approximately known, a highly detailed model may introduce parameters that cannot be physically justified or validated.

This does not mean that detailed models are unnecessary. On the contrary, detailed models are essential for component design, fault diagnosis, local flow analysis, high-frequency dynamics, and the investigation of specific physical mechanisms. The key point is that model detail should be consistent with the modeling objective and with the data available for parameterization and validation.

This problem appears in several engineering applications. Automatic gearbox HIL simulation [22], aircraft thrust reverser HIL simulation [31], real-time simulation with industrial Rexroth controllers [18], and virtual end-of-line testing of automatic transmissions [21] all show that engineering hydraulic models must strike a compromise among physical detail, parameter availability, real-time performance, and numerical robustness. Similarly, models of gear pumps [7], axial piston pumps [3], hydraulic valves [6], and hydro-mechanical actuators [20] require different levels of physical detail depending on whether the purpose is system-level response prediction, component design, fault diagnosis, or control validation.

Therefore, reasonable decomposition should not be determined solely by the physical existence of small geometric features. It should also depend on whether the corresponding parameters can be obtained, whether the introduced states can be validated, and whether the additional model detail improves the prediction relevant to the simulation objective.

2.4 Reproducibility, Toolchains, and Commercial Libraries

Another challenge concerns reproducibility and transparency. Modelica-based hydraulic studies often rely on mature commercial libraries, proprietary toolchains, or specialized co-simulation environments. These tools significantly improve engineering usability and industrial adoption, but they may also make it difficult for readers to fully reproduce internal component implementations when the underlying library models are not publicly available.

This does not diminish the engineering value of these libraries. Rather, it highlights a tension between industrial usability and academic reproducibility. This issue is

visible in applications involving HyLib [4], ActuationHydraulics [12], Modelon Hydraulics [9], Dymola-based multi-domain simulation [5], MWorks-based hydraulic component modeling [7], Simpack-FMU co-simulation [15], Abaqus-Dymola coupling [23], and toolchains involving FMI Toolbox or real-time control platforms [18]. These papers are valuable because they demonstrate engineering feasibility and application potential, but the degree of reproducibility depends on how much of the component implementation, parameterization, and solver configuration is available to the reader.

For this reason, future hydraulic model libraries should pay more attention not only to model functionality, but also to documentation, parameter transparency, test cases, benchmark models, and validation procedures. In particular, when a model is intended for academic communication or cross-platform reuse, its component assumptions and numerical treatment should be made as explicit as possible.

These challenges suggest that the problem of decomposition is not only a software-architecture issue, but also a hydraulic-domain modeling issue. It is therefore useful to revisit modeling traditions that were developed specifically for large-scale hydraulic systems. Xu's method is considered here from this perspective.

3 Xu‘s Method

3.1 Historical Background and Scope

The initial conception of Xu's method for hydraulic systems can be traced to the early 1980s and was associated with research activities at the Fluid Power Research Center of Oklahoma State University. In 1984, Xu Yangzeng and his collaborators presented a study on the performance-parameter method for the simulation of large-scale hydraulic systems at the 40th National Conference on Fluid Power and Control held in Chicago.

After returning to China, Xu further advanced this line of research with his team at Gansu University of Technology. The method was gradually developed into a relatively complete theoretical framework, encompassing modeling principles, numerical solution procedures, and software implementation. From 1986 to 1989, this method was advanced as a project funded by the National Natural Science Foundation of China, entitled 'Performance-Parameter Mathematical Modeling Method for Hydraulic Large-System Simulation.' In the early 1990s, it was also applied to the national key research project under the Seventh Five-Year Plan, 'Computer-Aided Design of Hydraulic Speed-Regulation Systems,' which led to the development of the hydraulic large-system simulation software package PERSIM.

Although the method had been theoretically formulated and tested in engineering applications, its further development was constrained by the limited software capabilities of the 1990s and by the comparatively immature industrial-software ecosystem in China at that time. As a result, it did not evolve into a continuously developed modeling and simulation framework, and its subsequent influence remained limited.

3.2 Modeling Principle

The core of Xu's method is to regard a hydraulic system or hydraulic component as a large-scale system composed of a set of standard submodels assembled according to the actual hardware connection relationships. Instead of relying on intermediate graphical representations such as bond graphs, signal-flow graphs, or block diagrams, the method directly takes the hydraulic circuit diagram, or an equivalent system diagram familiar to practicing engineers, as the basis for model construction.

Through component topological characteristics, interface relationships, boundary conditions, and feedback relations, basic submodels such as hydraulic resistance, hydraulic capacitance, hydraulic inertance, and feedback-type hydraulic resistance are combined into a system-level model. The resulting mathematical model preserves the original physical relationships among pressure, flow rate, displacement, velocity, and force in hydraulic components. Its mathematical form is therefore not a conventional explicit state-space equation, but an implicit state equation that more closely reflects the engineering and physical structure of the system.

The significance of this approach lies in its transformation of complex hydraulic modeling into an engineering-oriented, modular, and standardized procedure. It shifts system modeling from a process dominated mainly by mathematical transformation to one guided primarily by hydraulic domain knowledge. In this respect, Xu's method emphasizes that the elementary units of hydraulic modeling should be selected according to physical mechanism and engineering function, rather than according to an abstract requirement of geometric subdivision.

3.3 Numerical Solution Concept

At the level of simulation and numerical solution, Xu's method is coupled with a direct algebraic solution method that unifies static and dynamic simulation. When the derivative terms in the implicit state equations are set to zero, the system model is reduced to a set of static nonlinear algebraic equations. The steady operating point can then be obtained using algebraic iterative methods such as the Newton-Raphson method.

In dynamic simulation, the derivatives of the state variables are treated as unknowns to be solved. These derivative values are first obtained by solving the corresponding algebraic equation system and are then integrated to produce the dynamic response. In this way, static and dynamic simulations do not require two entirely separate modeling and solution frameworks. Instead, they are treated within the same implicit-equation structure and under the same class of algebraic solution procedures.

DOI
10.3384/ecp223127

This numerical idea is closely related to the practical difficulties of hydraulic system simulation, including initial-value problems, discontinuities, stiffness, and nonlinear algebraic coupling. However, Xu's method should not be regarded as equivalent to the DAE solution framework used in modern Modelica tools. In Modelica, system equations are processed through symbolic manipulation, connection equation generation, index reduction, initialization handling, and numerical integration by DAE or ODE solvers. Xu's method, by contrast, was formulated around hydraulic large-system implicit state equations and a direct algebraic solution procedure in which state derivatives are solved as unknowns before integration.

The relevance of Xu's method to Modelica therefore lies not in software implementation equivalence, but in its methodological awareness that hydraulic-system simulation requires careful coordination between model structure, state selection, and numerical solution strategy. This awareness remains important for the design of Modelica hydraulic libraries, especially when component decomposition affects the number and location of pressure states.

3.4 Methodological Relevance to Modelica Hydraulic Libraries

Although the original software implementation and numerical environment of Xu's method did not evolve into a mainstream contemporary modeling framework, the method retains methodological relevance for the development of hydraulic model libraries. Its lasting value lies less in its specific numerical implementation than in its modeling philosophy: a hydraulic system should not be decomposed excessively into unnecessarily fine-grained elementary units. Instead, the model should retain an appropriate level of aggregation so that the main physical behavior of components and subsystems can be represented efficiently and meaningfully.

This idea is compatible with an important principle of Modelica-based physical modeling. Component-based modeling does not imply an indiscriminate subdivision of a system into the smallest possible parts. Rather, it requires the selection of suitable component boundaries and abstraction levels according to the purpose of simulation. This is consistent with the distinction between functional and predictive models in Modelica-based transmission hydraulics, where system-level simulation does not necessarily require every valve orifice, chamber, and local throttling structure to be fully expanded [26].

This parallel does not imply a direct historical influence between Xu's method and later Modelica-based hydraulic modeling. Rather, it shows that both perspectives recognize the importance of selecting model boundaries according to simulation objectives. Therefore, revisiting Xu's method is not intended to replace Modelica's acausal modeling paradigm. Instead, it provides a hydraulic-domain-oriented perspective for reconsidering how component boundaries should be defined in Modelica hydraulic libraries.

From this perspective, reasonable decomposition should preserve the dominant physical mechanisms of hydraulic components while avoiding unnecessary state duplication, hidden volumes, and excessive numerical stiffness. This provides a methodological bridge between traditional hydraulic-domain modeling experience and the architecture design of modern Modelica hydraulic libraries.

4 Representative Examples of Reasonable Decomposition

The following examples are not intended to provide complete component models. They are order-of-magnitude estimates used to illustrate how the same hydraulic structure can lead to different model boundaries depending on the simulation objective, the dominant physical mechanism, the available parameters, and the quantities to be validated. The numerical values used below should therefore be understood as illustrative estimates rather than universal design constants.

4.1 Pilot-Operated Relief Valves

A pilot-operated relief valve is a typical example in which excessive fine-grained decomposition may introduce unnecessary pressure states and obscure the dominant physical mechanism. Internally, such a valve may include a main spool or poppet, a damping orifice, a pilot valve, a control chamber, spring forces, leakage paths, and return passages. If the valve is decomposed only according to the smallest visible hydraulic elements, the damping orifice, pilot-valve inlet, main-valve upper chamber, pressure-force interface, leakage path, and local passage volumes may all be represented as separate components. This modeling style is convenient for graphical assembly, but it may also create artificial pressure states in regions that physically share the same control volume or are connected by very short passages.

For many system-level simulations, the dominant dynamics of a pilot-operated relief valve can be represented by a single control-chamber pressure, the main moving element, and the relevant nonlinear hydraulic resistances. A compact integrated model may be estimated by the following relations:

$$q_d = C_d A_d \sqrt{\left(\frac{2(p_s - p_c)}{\rho}\right)} \quad (1)$$

$$q_p = C_p A_p(x_p) \sqrt{\frac{2(p_c - p_t)}{\rho}} \quad (2)$$

$$\left(\frac{V_c}{\beta_e}\right) \frac{dp_c}{dt} = q_d - q_p - A_m \frac{dx_m}{dt} \quad (3)$$

$$m_m \frac{d^2 x_m}{dt^2} = A_s p_s - A_c p_c - k_m x_m - c_m \frac{dx_m}{dt} - F_0 \quad (4)$$

Here p_c is the common control-chamber pressure, V_c is the effective control volume, q_d is the flow through

the damping orifice, q_p is the pilot-stage flow, and x_m is the main-valve displacement. Equation (3) shows that the control region contributes one physically meaningful pressure state when it is treated as a shared control chamber. If the pilot valve is treated quasi-statically, the integrated model has one pressure state and two mechanical states. If the pilot valve dynamics are also retained, two additional mechanical states may be introduced:

$$N_{integrated} \approx 3 \text{ (quasi-static pilot valve) (5a)}$$

$$N_{integrated} \approx 5 \text{(dynamic pilot valve) (5b)}$$

By contrast, if the same physical control region is artificially split into four small volumes, the model may introduce four pressures p_c1, p_c2, p_c3, and p_c4. A possible split representation is:

$$\begin{aligned} \left(\frac{V_{c1}}{\beta_e}\right)\cdot\frac{dp_{c1}}{dt} &= q_d - q_{12} \\ \left(\frac{V_{c2}}{\beta_e}\right)\cdot\frac{dp_{c2}}{dt} &= q_{12} - q_{23} \\ \left(\frac{V_{c3}}{\beta_e}\right)\cdot\frac{dp_{c3}}{dt} &= q_{23} - q_p \end{aligned} \quad (6)$$

$$\left(\frac{V_{c4}}{\beta_e}\right)\cdot\frac{dp_{c4}}{dt} = q_{34} - A_m\frac{dx_m}{dt} \quad (7)$$

$$N_{split} \approx 6 \text{ (main valve only),}$$

$$N_{split} \approx 8 \text{ (with pilot-spool dynamics) (8)}$$

If these pressures actually correspond to one control chamber or to strongly coupled small regions, then physically p_c1 ≈ p_c2 ≈ p_c3 ≈ p_c4. The additional pressure states mainly describe fast internal equalization rather than the dominant valve behavior. Under a local linearization q = G Δp, the hydraulic capacitance of a virtual chamber is C_h = V_ε/β_e, and the pressure equalization time constant can be estimated as:

$$\tau = \frac{C_h}{G} = \frac{V_\varepsilon}{\beta_e G} \quad (9)$$

$$\tau = \frac{2\times10^{-8}}{1.4\times10^{9}\times 2\times10^{-12}} \approx 7.1\times10^{-6}\,s \quad (10)$$

This estimate uses V_ε = 0.02 cm³ = 2 × 10⁻⁸ m³, β_e = 1.4 × 10⁹ Pa, and G = 2 × 10⁻¹² m³/(s·Pa). The resulting time constant is several orders of magnitude smaller than a typical main-spool motion time scale of approximately 1-10 ms. Such a small artificial time constant may increase stiffness while contributing little to the system-level behavior to be validated. Therefore, for system-level pressure-regulation analysis, it is often more appropriate to treat the control chamber, damping orifice, pressure-force relation, and feedback relation as parts of one integrated valve submodel rather than as many independently connected external components.

4.2 Pressure Compensators in Multi-Way Valves

Pressure compensators in load-sensing multi-way valves provide another example in which the appropriate model boundary is often functional rather than purely geometric. A valve section may contain a main metering edge, a compensator spool, load-sensing passages, small internal chambers, spring chambers, and local restrictions. If each of these geometric features is represented as an independent hydraulic component with its own volume, the resulting model may contain many pressure states that mainly describe internal pressure equalization within the valve block.

The main hydraulic function of a pressure compensator is to maintain an approximately constant pressure drop across the metering orifice. For the i-th valve section, the metering flow can be written in the usual orifice form:

$$q_i = C_q A_i(x_i)\sqrt{2\frac{\Delta p_i}{\rho}}$$

$$\Delta p_i = p_{u,i} - p_{L,i} \quad (11)$$

$$A_c(p_{u,i} - p_{L,i}) = k_c x_c + F_c \quad (12)$$

$$p_{u,i} - p_{L,i} \approx \Delta p_0, q_i \approx C_q A_i(x_i)\sqrt{2\frac{\Delta p_0}{\rho}} \quad (13)$$

Equation (12) is a simplified quasi-static force balance of the compensator spool, where inertia, damping, friction, and flow forces are neglected for a functional-level approximation. Under a fast-compensation or quasi-steady approximation, the compensator enforces an approximately constant pressure margin, as shown in Equation (13). In this representation, the compensator is not modeled as an isolated variable orifice, but as a functional hydraulic unit that regulates the relationship among supply pressure, load pressure, metering area, and flow rate.

The state-count difference can be substantial. If each valve section is decomposed into a main-valve upstream chamber, a main-valve downstream chamber, a compensator upstream chamber, a compensator downstream chamber, and a load-sensing pressure chamber, five pressure states may be added per section. For a four-section multi-way valve, this gives an estimated twenty additional pressure states. A higher-level functional model may instead retain one compensator-spool state per section, or even use a quasi-static relation when the compensator is sufficiently fast:

$$N_extra \approx 4 \times 5 = 20,\ N_comp \approx 4,\ \Delta N \approx 16 \quad (14)$$

These additional states are not necessarily associated with the dominant motion of the boom, arm, bucket, steering actuator, or other hydraulic loads. They often correspond to small internal passages and rapidly equilibrating valve-block chambers. Therefore, for system-level analysis of actuator flow distribution, load-sensing behavior, or pressure-margin regulation, a functional compensator model may be more meaningful and numerically robust than a purely geometric decomposition. Detailed compensator models remain necessary when the objective is valve design, spool oscillation analysis, manufacturing-sensitivity investigation, or low-pressure stability analysis.

DOI 10.3384/ecp223127

4.3 Short Manifold Passages

Short passages in hydraulic manifolds are a common source of excessive decomposition. A hydraulic manifold may contain drilled holes, cross passages, local intersections, plugs, and short connecting channels. If each passage is modeled as an independent dynamic pipe segment with its own resistance, inertance, and compressible volume, the model may contain many small pressure states and fast hydraulic time constants.

For a short manifold passage with length L = 30 mm = 0.03 m and oil acoustic wave speed a = 1200 m/s, the pressure-wave transit time is:

$$t_w = \frac{L}{a} = \frac{0.03}{1200} = 2.5 \times 10^{-5}\, s \quad (15)$$

$$\frac{t_w}{t_s} = \frac{2.5 \times 10^{-5}}{0.05-0.5} \approx 5 \times 10^{-5} \sim 5 \times 10^{-4} \quad (16)$$

Equation (16) corrects the dimensional order of the estimate: if the dominant actuator response time is approximately t_s = 0.05-0.5 s, the wave transit time of the short passage is four to five orders of magnitude smaller. Modeling every short passage as an independent dynamic pipe segment may therefore introduce stiffness without improving the prediction of the system-level response.

A more appropriate representation is often to combine several short passages into an equivalent manifold node:

$$V_M = \Sigma V_j, \left(\frac{V_M}{\beta_e}\right) \cdot \frac{dp_M}{dt} = \Sigma q_{in} - \Sigma q_{out} \quad (17)$$

$$\Delta p_j = R_j \; q_j \, |q_j| \quad (18)$$

$$\left(\frac{V_j}{\beta_e}\right) \cdot \frac{dp_j}{dt} = q_{\{j-1\}} - q_j \quad (19)$$

$$N_split \approx 20, N_lumped \approx 6, \Delta N \approx 14 \quad (20)$$

In this aggregated representation, the short passages still contribute pressure loss, as indicated by Equation (18), but they are not all assigned independent compressible states as in Equation (19). If the simulation objective is water-hammer analysis, high-frequency acoustic behavior, or local cavitation in the manifold, a more detailed dynamic representation may be justified. For ordinary system-level actuator simulation, however, an equivalent manifold-node representation is often more consistent with both numerical efficiency and physical relevance.

4.4 Hydraulic Cylinder Chambers

Hydraulic cylinder chambers are natural control volumes in hydraulic modeling. The chamber pressure directly determines piston force, while piston displacement changes chamber volume. In ordinary lumped system-level simulations, the chamber itself is usually the appropriate control volume. For the cap-end chamber of a hydraulic cylinder, a typical pressure equation may be written as follows:

$$\left(\frac{V_{A(x)}}{\beta_e}\right) \cdot \frac{dp_A}{dt} = q_A - A_A \cdot \frac{dx}{dt} \quad (21)$$

$$V_A(x) = V_{A0} + A_A \, x \quad (22)$$

Here p_A is the chamber pressure, q_A is the net inflow, A_A is the piston area, and x is the piston displacement. Equations (21) and (22) form a physically meaningful hydraulic capacitance because they directly couple oil compressibility with mechanical motion. For such a chamber, an integrated model normally requires one hydraulic pressure state, p_A, in addition to the mechanical states of the piston or load.

If the same chamber is decomposed into a fixed-volume component, a swept-volume component, a pressure-area component, and an external flow-summing component, additional pressure variables may appear even though they represent the same physical chamber. Extra algebraic constraints are then required:

$$p_{A0} = p_{Ax} = p_{Ai}, V_A = V_{A0} + A_A \, x \quad (23)$$

This decomposition does not add physical information. It splits a clear control-volume equation into several coupled components and may create redundant variables or algebraic constraints. Therefore, for ordinary hydraulic cylinder modeling, the chamber itself is usually an appropriate component boundary. A single lumped chamber may be insufficient only when the cylinder is very long, when pressure-wave dynamics are relevant, when trapped sub-volumes exist, when internal damping structures dominate the response, or when cavitation and thermal effects are central to the simulation objective.

4.5 Accumulator Groups in Hydraulic Pitch Systems

Accumulator groups in hydraulic pitch systems provide another example where functional aggregation may be more appropriate than strict geometric decomposition. In wind-turbine hydraulic pitch systems, accumulators are often used to provide emergency energy storage so that blades can be pitched to a safe position during power loss or failure conditions. At the system level, the main questions are whether the actuators can complete the pitch motion within the required time and whether the accumulator pressure remains sufficient throughout the process.

For a single gas-charged accumulator, a simplified polytropic relation may be written as follows:

$$p_g V_g^{\,n} = C, p_h = p_g, q_h = -\frac{dV_g}{dt} \quad (24)$$

$$V_{g,eq} = \Sigma V_{g,i}, p_g \, V_{g,eq}^{\,n} = C_{eq} \quad (25)$$

$$N_three \approx 3, N_eq \approx 1 \quad (26)$$

$$N_split \approx 3 + 3 \times 2 = 9 \quad (27)$$

If three accumulators are connected in parallel, have similar precharge pressures, similar thermal assumptions, and act as a collective energy-storage unit connected to a common pressure node, they may be aggregated into an equivalent accumulator. The number of gas states is then reduced from three to one, as indicated by Equation (26). If each accumulator inlet is further decomposed into an isolation-valve upstream chamber, an isolation-valve downstream chamber, a check-valve chamber, and small

connecting volumes, the estimated state count may increase substantially, as shown in Equation (27).

This does not mean that accumulator aggregation is always appropriate. If unequal precharge, individual accumulator failure, thermal behavior, valve malfunction, or branch interaction is the object of study, individual accumulator models are necessary. However, if the accumulators are physically similar, connected to a common manifold, and used as a collective energy-storage group, an equivalent model may better match the system-level objective and the available validation data.

4.6 Summary of the Examples

The preceding examples show that reasonable decomposition in hydraulic modeling cannot be reduced to a universal rule such as 'more detailed is always better' or 'fewer states are always better.' The appropriate level of decomposition depends on the relation among physical mechanism, simulation objective, available parameters, dominant time scale, numerical robustness, and validation data.

Three modeling purposes should be distinguished. The first purpose is rapid model assembly. This purpose tends to favor small reusable components such as orifices, chambers, pressure-area elements, springs, dampers, leakage paths, pipe segments, and interface adapters. Such decomposition is useful for graphical modeling and library reuse, but it can also introduce duplicated states and hidden volumes if applied mechanically.

The second purpose is expert-guided low-order modeling. This is closer to the modeling philosophy of Xu's method. The modeler first determines which leakages can be neglected, which small volumes should be aggregated, which masses and inertances must be retained, which passages should be merged, and which chambers are essential. The hydraulic component is then transformed into an equivalent internal system composed of hydraulic resistances, hydraulic capacitances, hydraulic inertances, feedback hydraulic resistances, and mechanical submodels.

The third purpose is objective-oriented model selection. If the research question concerns spool vibration, local cavitation, valve-flow forces, water hammer, or high-frequency dynamics, detailed decomposition is necessary. If the research question concerns actuator motion, pressure regulation, system power distribution, emergency energy supply, or control validation, excessive fine-grained decomposition may increase stiffness and reduce interpretability without improving the relevant prediction.

Thus, the key issue is not whether a hydraulic model should be detailed or simplified, but whether its component boundaries correspond to the dominant physical behavior and to the quantities that must be validated. The estimates in this section are intended to support this judgment, not to replace detailed parameter identification or validation when such validation is required.

5 Conclusion

This paper has reconsidered Modelica-based hydraulic modeling from the perspective of reasonable component decomposition. The review in Section 1 shows that Modelica hydraulic modeling has developed from early component libraries and schematic-style assembly toward multi-domain engineering applications, component-level physical design, real-time simulation, and validation-oriented modeling. The challenges discussed in Section 2 further show that hydraulic models are particularly sensitive to state placement, small volumes, valve switching, parameter availability, and numerical stiffness. Section 3 then reinterprets Xu's method as a hydraulic-domain-oriented modeling philosophy rather than as a replacement for Modelica's acausal modeling paradigm.

The examples in Section 4 show that excessive fine-grained decomposition may introduce duplicated pressure states, hidden volumes, artificial fast time constants, additional algebraic constraints, and unclear physical semantics. In pilot-operated relief valves, several internal regions may share one physically meaningful control-chamber pressure. In pressure-compensated multi-way valves, the compensator may be better represented as a functional pressure-margin regulator when system-level flow distribution is the modeling objective. In short manifold passages, many small geometric channels may be better treated as equivalent hydraulic-network nodes with algebraic pressure losses. In hydraulic cylinders, the chamber itself is often the natural control volume. In accumulator groups, an equivalent accumulator may be more appropriate than a detailed set of individual units when only group-level energy storage is relevant.

These examples do not imply that detailed models are inferior. Detailed component models remain necessary for valve design, fault diagnosis, local flow analysis, cavitation, water hammer, high-frequency dynamics, and component-level validation. The point is that model detail must be justified by the simulation objective, the available parameters, and the validation data. A model with more components and more states is not automatically more physical; it may simply be more fragmented.

For Modelica hydraulic libraries, the main implication is that detailed component models and higher-level functional or equivalent models should coexist. Component documentation should clearly state the intended use of each model, the physical assumptions behind it, and the meaning of its states. Pressure states should correspond to meaningful hydraulic capacitances rather than arbitrary geometric fragments. When Xu's method is used as a methodological reference, its value lies in emphasizing hydraulic function, dominant physical relations, equivalent internal system structure, and appropriate aggregation.

Finally, models inspired by Xu's method must still satisfy the requirements of modern DAE-based Modelica

DOI
10.3384/ecp223127

tools when they are implemented in Modelica. The modeler must distinguish physical states from algebraic variables, regularize orifice equations, smooth valve-area transitions, scale residuals, design consistent initialization, and expose suitable interfaces for linearization when needed. These tasks are often hidden inside mature hydraulic libraries, but they cannot be ignored when constructing custom equivalent models. Therefore, the modern relevance of Xu's method lies not in avoiding Modelica's numerical framework, but in helping define clearer and more meaningful hydraulic component boundaries within that framework.

Acknowledgements

The authors gratefully acknowledge the long-standing contributions of all individuals and companies in the Modelica community to the development of Modelica, especially to its application in hydraulic system modeling and simulation.

References

[1] Alkov, Ilja, Robin Diekmann, and Dirk Weidemann (2014). 'A Generalized Power-Based Modelica Library with Application to an Industrial Hydraulic Plant.' In: Proceedings of the 10th International Modelica Conference. Lund, Sweden, March 10-12, 2014. Linköping Electronic Conference Proceedings 96, pp. 617-626. DOI: 10.3384/ecp14096617.

[2] Bals, Johann, Gerhard Hofer, Andreas Pfeiffer, and Christian Schallert (2003). 'Object-Oriented Inverse Modelling of Multi-Domain Aircraft Equipment Systems with Modelica.' In: Proceedings of the 3rd International Modelica Conference. Linköping, Sweden, November 3-4, 2003, pp. 377-384. DOI: 10.5555/3338027.3338038.

[3] Bayer, Christian and Olaf Enge-Rosenblatt (2011). 'Modeling of hydraulic axial piston pumps including specific signs of wear and tear.' In: Proceedings of the 8th International Modelica Conference. Dresden, Germany, March 20-22, 2011. Linköping Electronic Conference Proceedings 63, pp. 461-466. DOI: 10.3384/ecp11063461.

[4] Beater, Peter (2000). 'Modeling and Digital Simulation of Hydraulic Systems in Design and Engineering Education using Modelica and HyLib.' In: Modelica Workshop 2000 Proceedings. Lund, Sweden, October 23-24, 2000, pp. 33-40. URL: https://modelica.org/events/workshop2000/proceedings/Beater.pdf.

[5] Beater, Peter and Martin Otter (2003). 'Multi-Domain Simulation: Mechanics and Hydraulics of an Excavator.' In: Proceedings of the 3rd International Modelica Conference. Linköping, Sweden, November 3-4, 2003, pp. 331-340. URL: https://modelica.org/events/Conference2003/papers/h36_excavator_Beater_bw.pdf.

[6] Chandrasekar, Sureshkumar and Hubertus Tummescheit (2014). 'Physical Design of Hydraulic Valves in Modelica.' In: Proceedings of the 10th International Modelica Conference. Lund, Sweden, March 10-12, 2014. Linköping Electronic Conference Proceedings 96, pp. 627-636. DOI: 10.3384/ecp14096627.

[7] Chen, Liping, Yan Zhao, Fanli Zhou, Jianjun Zhao, and Xianzhao Tian (2011). 'Modeling and Simulation of Gear Pumps based on Modelica/MWorks.' In: Proceedings of the 8th International Modelica Conference. Dresden, Germany, March 20-22, 2011. Linköping Electronic Conference Proceedings 63, pp. 421-429. DOI: 10.3384/ecp11063421.

[8] Clauß, Christoph and Peter Beater (2002). 'Multidomain Systems: Electronic, Hydraulic, and Mechanical Subsystems of an Universal Testing Machine Modeled with Modelica.' In: Proceedings of the 2nd International Modelica Conference. Oberpfaffenhofen, Germany, March 18-19, 2002, pp. 25-30. URL: https://modelica.org/events/Conference2002/papers/p04_Clauss.pdf.

[9] Coïc, Clément, Johan Andreasson, Johan Åkesson, Anand Pitchaikani, and Hemanth Sattenapalli (2020). 'Collaborative Development and Simulation of an Aircraft Hydraulic Actuator Model.' In: Proceedings of Asian Modelica Conference 2020. Tokyo, Japan, October 8-9, 2020. Linköping Electronic Conference Proceedings 174, pp. 67-75. DOI: 10.3384/ecp202017467.

[10] Ensbury, Theodor, Peter Harman, and Mike Dempsey (2018). 'Modelling and Development of a Pseudo-Hydraulic Power Steering Model for use in Real-Time Applications.' In: Proceedings of the 2nd Japanese Modelica Conference. Tokyo, Japan, May 17-18, 2018. Linköping Electronic Conference Proceedings 148, pp. 137-146. DOI: 10.3384/ecp18148137.

[11] Fu, Hongxiang, David Blum, and Michael Wetter (2022). 'Fan and Pump Efficiency in Modelica based on the Euler Number.' In: Proceedings of the American Modelica Conference 2022. Dallas, TX, USA, October 26-28, 2022. Linköping Electronic Conference Proceedings 186, pp. 19-25. DOI: 10.3384/ECP2118619.

[12] Harman, Peter (2006). 'Modelling Automotive Hydraulic Systems using the Modelica ActuationHydraulics Library.' In: Proceedings of the 5th International Modelica Conference. Vienna, Austria, September 4-5, 2006, pp. 399-403. URL: https://modelica.org/events/modelica2006/Proceedings/sessions/Session409.pdf.

[13] Hou, Yuming, Lingyang Li, Ping He, Yunqing Zhang, and Liping Chen (2011). 'Shock Absorber Modeling and Simulation Based on Modelica.' In: Proceedings of the 8th International Modelica Conference. Dresden, Germany, March 20-22, 2011. Linköping Electronic Conference Proceedings 63, pp. 843-846. DOI: 10.3384/ecp11063843.

[14] Huang, Hua, Sebastian Nowoisky, René Knoblich, and Clemens Gühmann (2012). 'Modeling and Testing of the Hydro-Mechanical Synchronization System for a Double Clutch Transmission.' In: Proceedings of the 9th International Modelica Conference. Munich, Germany, September 3-5, 2012. Linköping Electronic Conference Proceedings 76, pp. 287-294. DOI: 10.3384/ecp12076287.

[15] Ishida, Nobumasa and Hideyuki Muramatsu (2018). '[Industrial paper] Application for Optimization of Control Parameters for Multi-body and Hydraulics System by using FMU.' In: Proceedings of the 2nd Japanese Modelica Conference. Tokyo, Japan, May 17-18, 2018. Linköping Electronic Conference Proceedings 148, pp. 169-173. DOI: 10.3384/ecp18148169.

[16] Kang, Sinyoung, Marcel Gottschall, Sunghyun Cho, and Torsten Blochwitz (2025). 'Requirements-based, early stage

Architecture Performance Validation on a Brake System Use Case.' In: Proceedings of the Asian Modelica Conference 2024. Jeju, Korea, December 12-13, 2024. Linköping Electronic Conference Proceedings 217, pp. 93-102. DOI: 10.3384/ecp21793.

[17] Mazurié, Baptiste, Audrey Jardin, Pascal Borel, Didier Boldo, Frans Davelaar, and Luis Corona Mesa-Moles (2025). 'Data Reconciliation for Industrial Experiments.' In: Proceedings of the 16th International Modelica & FMI Conference. Lucerne, Switzerland, September 8-10, 2025. Linköping Electronic Conference Proceedings 218, pp. 45-54. DOI: 10.3384/ecp21845.

[18] Menager, Nils, Rüdiger Kampfmann, Niklas Worschech, and Lars Mikelsons (2015). 'Suitability of Different Real-Time Solvers for a Model-Based Engineering Toolchain using Industrial Rexroth Controllers.' In: Proceedings of the 11th International Modelica Conference. Versailles, France, September 21-23, 2015. Linköping Electronic Conference Proceedings 118, pp. 883-891. DOI: 10.3384/ecp15118883.

[19] Mikelsons, Lars, Hongchao Ji, Thorsten Brandt, and Oliver Lenord (2009). 'Symbolic Model Reduction Applied to Realtime Simulation of Construction Machines.' In: Proceedings of the 7th International Modelica Conference. Como, Italy, September 20-22, 2009. Linköping Electronic Conference Proceedings 43, pp. 765-774. DOI: 10.3384/ecp09430136.

[20] Nowoisky, Sebastian, Chi Shen, and Clemens Gühmann (2011). 'Detailed Model of a Hydromechanical Double Clutch Actuator with a Suitable Control Algorithm.' In: Proceedings of the 8th International Modelica Conference. Dresden, Germany, March 20-22, 2011. Linköping Electronic Conference Proceedings 63, pp. 467-473. DOI: 10.3384/ecp11063467.

[21] Roper, Jan, Jens Gores, and Clemens Gühmann (2015). 'Modeling of an Automatic Transmission for the Evaluation of Test Procedures in a Virtual End-of-Line Test Bench.' In: Proceedings of the 11th International Modelica Conference. Versailles, France, September 21-23, 2015. Linköping Electronic Conference Proceedings 118, pp. 547-556. DOI: 10.3384/ecp15118547.

[22] Schlegel, Clemens, Marco Bross, and Peter Beater (2002). 'HIL-Simulation of the Hydraulics and Mechanics of an Automatic Gearbox.' In: Proceedings of the 2nd International Modelica Conference. Oberpfaffenhofen, Germany, March 18-19, 2002, pp. 67-75. URL: https://modelica.org/events/Conference2002/papers/p10_Schlegel.pdf.

[23] Schofield, Brad, Harish Surendranath, Magnus Gäfvert, and Victor Oancea (2009). 'Interfacing Abaqus with Dymola: A High Fidelity Anti-Lock Brake System Simulation.' In: Proceedings of the 7th International Modelica Conference. Como, Italy, September 20-22, 2009. Linköping Electronic Conference Proceedings 43, pp. 833-838. DOI: 10.3384/ecp09430110.

[24] Shangguan, Duansen, Liping Chen, Jianwan Ding, and Yuhui Liu (2019). 'Modeling and Simulation of Dual Redundant Electro-Hydrostatic Actuation System with Special Focus on model architecting and multidisciplinary effects.' In: Proceedings of the 13th International Modelica Conference. Regensburg, Germany, March 4-6, 2019. Linköping Electronic Conference Proceedings 157, pp. 463-470. DOI: 10.3384/ecp19157463.

[25] Soejima, Shinichi and Takashi Matsuba (2002). 'Application of mixed mode integration and new implicit inline integration at Toyota.' In: Proceedings of the 2nd International Modelica Conference. Oberpfaffenhofen, Germany, March 18-19, 2002, pp. 65-71. URL: https://modelica.org/events/Conference2002/papers/p09_Soejima.pdf.

[26] Tiller, Michael M. (2005). 'Development of a Simplified Transmission Hydraulics Library based on Modelica.Fluid.' In: Proceedings of the 4th International Modelica Conference. Hamburg, Germany, March 7-8, 2005, pp. 237-243. URL: https://modelica.org/events/Conference2005/online_proceedings/Session3/Session3b4.pdf.

[27] Tinnerholm, John, Francesco Casella, and Adrian Pop (2022). 'Towards Modeling and Simulation of Dynamic Overconstrained Connectors in Modelica.' In: Proceedings of Asian Modelica Conference 2022. Tokyo, Japan, November 24-25, 2022, pp. 35-44. DOI: 10.3384/ecp19335.

[28] Velut, Stéphane and Hubertus Tummescheit (2011). 'Implementation of a transmission line model for fast simulation of fluid flow dynamics.' In: Proceedings of the 8th International Modelica Conference. Dresden, Germany, March 20-22, 2011. Linköping Electronic Conference Proceedings 63, pp. 446-453. DOI: 10.3384/ecp11063446.

[29] Viel, Antoine (2011). 'Strong Coupling of Modelica System-Level Models with Detailed CFD Models for Transient Simulation of Hydraulic Components in their Surrounding Environment.' In: Proceedings of the 8th International Modelica Conference. Dresden, Germany, March 20-22, 2011. Linköping Electronic Conference Proceedings 63, pp. 256-265. DOI: 10.3384/ecp11063256.

[30] Worschech, Niklas and Lars Mikelsons (2012). 'A Toolchain for Real-Time Simulation using the OpenModelica Compiler.' In: Proceedings of the 9th International Modelica Conference. Munich, Germany, September 3-5, 2012. Linköping Electronic Conference Proceedings 76, pp. 839-846. DOI: 10.3384/ecp12076839.

[31] Zhao, Jianjun, Ziqiang Li, Jianwan Ding, Liping Chen, Qifu Wang, Qing Lu, Hongxin Wang, and Shuang Wu (2009). 'HIL Simulation of Aircraft Thrust Reverser Hydraulic System in Modelica.' In: Proceedings of the 7th International Modelica Conference. Como, Italy, September 20-22, 2009. Linköping Electronic Conference Proceedings 43, pp. 178-183. DOI: 10.3384/ecp09430040.

FMU-based Training of Physics-Informed Neural Nets for Control

Yiding Wang[1] Luca Bascetta[1] Gianni Ferretti[1]

[1] Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Italy,
{yiding.wang, luca.bascetta, gianni.ferretti}@polimi.it

Abstract

Physics-Informed Neural Nets for Control (PINCs) extend traditional Physics-Informed Neural Networks (PINNs) by incorporating control inputs and variable initial conditions, therefore enabling autoregressive long-horizon simulation by chaining short-horizon predictions, and making them suitable for control-oriented applications. However, the existing literature commonly adopts PINC training procedures where the governing dynamics are embedded in an explicit analytical form, whereas in many engineering workflows, the system models are often available only as Functional Mock-up Units (FMUs). This paper proposes an FMU-based training framework for PINC that uses an FMI 3.0 Model Exchange (ME) FMU directly as the description of system dynamics. The physics-informed residual is evaluated in the forward pass by querying the FMU for state derivatives, while gradient backpropagation is enabled through FMI 3.0 adjoint derivatives integrated into PyTorch via a custom `autograd.Function`. Case studies on a Van der Pol oscillator and on a mechanical system show that the proposed framework can train PINCs without manually implementing the analytical governing equations and can achieve long-horizon simulation performance comparable to conventional ODE-based training when the required FMI 3.0 features are available.

Keywords: Physics-Informed Neural Network, Functional Mock-up Unit, Functional Mock-up Interface 3.0, Adjoint derivatives

1 Introduction

Physics-Informed Neural Networks (PINNs) (Raissi et al. 2019) have emerged as a promising framework for solving differential equations, performing system identification, and learning system dynamics as surrogate models (Karpatne et al. 2022). Unlike conventional machine learning methods that rely solely on measured data, PINNs incorporate governing differential equations into the training loss, which could efficiently improve sample efficiency and interpretability of the trained model. Such properties are especially attractive in engineering domains where real data are scarce, expensive to obtain, or insufficiently informative over the full operating envelope. Recent surveys (Farea et al. 2024; Luo et al. 2025; Zhang et al. 2026) provide comprehensive overviews of physics-informed neural networks, covering developments in network architectures, loss design, feature embedding, and applications across various engineering fields. They also identify persistent challenges related to computational cost, ill-conditioned loss functions, training convergence, and the effective incorporation of physical knowledge.

From a control perspective, however, standard PINNs have important limitations. In their original form, they are typically trained for fixed boundary conditions within a fixed time horizon, which makes them unsuitable for handling varying initial conditions and control inputs, and also limits their use for long-horizon prediction. To address this issue, Physics-Informed Neural Nets for Control (PINC) (Antonelo et al. 2024) was recently proposed as a control-oriented extension of PINNs. The PINC network is conditioned on time, initial states and control inputs over a short interval. Long-horizon simulation is then obtained by chaining the short-horizon predictions in an autoregressive manner, which makes the approach suitable for control-oriented tasks. Successful applications of PINC to simulation and control of nonlinear systems have been demonstrated by Antonelo et al. (2024) for the Van der Pol oscillator, the quadruple tank system and electric submersible pumps, by Nicodemus et al. (2022) for multi-link manipulators and by Amer et al. (2025) for underwater vehicles.

Despite these promises, a major practical barrier remains in the training of PINC. Existing literature usually assumes access to the governing dynamics in an explicit and differentiable form, typically as a system of ordinary, or partial, differential equations (ODE/PDE) that can be evaluated when forming the physics-informed residual and differentiated during backpropagation through the computation graph built by modern deep learning frameworks such as PyTorch (Paszke, Gross, Massa, et al. 2019) or TensorFlow (Martín Abadi et al. 2015). However, this assumption can be restrictive in many industrial applications where the underlying ODEs may be difficult to access in a convenient analytical form. Even when the equations are available, manual implementation could be error-prone and must be repeated for each new system.

On the other hand, in many practical engineering workflows, systems are often modeled in the Modelica language and exported as Functional Mock-up Units (FMUs) under the Functional Mock-up Interface (FMI) standard (Modelica Association 2026). In this form, high-fidelity models can be exchanged, simulated and reused across different software environments without

requiring direct access to the full governing equations. Consequently, an FMU may already encapsulate physical knowledge, validated behavior, and reusable model structure of the system, while the corresponding analytical ODE representation remains inaccessible or inconvenient to extract for conventional physics-informed machine learning methods. This creates a discrepancy between the requirements of conventional physics-informed learning and the system representations that are actually available in engineering practice.

Recent studies have begun to explore approaches for integrating FMI-based models with machine learning frameworks. To list but a few, the NeuralFMU approach (Thummerer, Mikelsons, et al. 2021; Thummerer, Jarmolowitz, et al. 2025) integrates FMUs into trainable neural-network architectures, enabling data-driven components to augment physics-based models. In parallel, FMU-Gym (Wrede et al. 2024) and train_fmu_gym (Ultsch et al. 2025) use FMUs as standardized simulation environments for training and evaluating reinforcement learning controllers. Together, these studies demonstrate the potential of FMI as an interoperability layer between physics-based modeling and machine learning. However, these approaches mainly focus on hybrid-model correction and reinforcement-learning environments, rather than directly using the dynamics and derivative capabilities of an FMU to construct and differentiate a physics-informed training loss.

To close this gap, this work proposes an FMU-based PINC training framework. Rather than introducing a new network architecture, the contribution lies in a new training pathway: the proposed method leverages an FMI 3.0 (Junghanns et al. 2021) Model Exchange (ME) FMU directly as the source of system dynamics, instead of relying on manually implemented analytical expressions of the underlying ODEs. Under assumptions of smooth ODE dynamics, absence of events, and availability of FMI 3.0 adjoint derivatives, the physics-informed residual in the forward pass is evaluated by querying the FMU for state derivatives, while gradient backpropagation is enabled through adjoint derivatives exposed by the FMI 3.0 interface (Junghanns et al. 2021; Baydin et al. 2017). The proposed framework is implemented in FMPy (Dassault Systèmes 2026) and PyTorch (Paszke, Gross, Massa, et al. 2019). Since FMU-related operations are not natively supported by PyTorch, a custom `autograd.Function` is defined, such that the FMU-based gradient backpropagation could be carried out.

The remainder of the paper is organized as follows. Section 2 provides background on PINC and FMI 3.0 ME FMUs, as well as an overview of PyTorch custom `autograd.Function`. Section 3 formulates the problem of training a PINC with an FMU, while listing fundamental assumptions adopted in this work. Section 4 introduces the proposed method, namely the `autograd.Function`, and other implementation details. In Section 5, examples are demonstrated showing the performance of the proposed training strategy. Finally, Section 6 concludes the paper and outlines directions for future work.

2 Background

The proposed training framework of PINC mainly relies on the definition of a custom `autograd.Function` that involves the adjoint derivative exposed by the ME FMU which is a new feature of FMI 3.0. In this section, a brief introduction to the related topics will be given.

2.1 Physics-Informed Neural Nets for Control

The original PINC formulation extends standard PINNs by augmenting the input space with initial state and control input over a short time interval, in addition to time instants. Consider a controlled dynamical system described by the ODE:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$, $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ are the vectors of system states and control inputs.

A PINC is a neural network $\mathscr{N}$ parameterized by $\boldsymbol{\theta}$ that approximates the state trajectory in a short horizon $[0, T]$, namely,

$$\hat{\mathbf{x}}(t) = \mathscr{N}_{\boldsymbol{\theta}}(t, \mathbf{x}_0, \mathbf{u}), \quad t \in [0, T] \tag{2}$$

where $\mathbf{x}_0$ is the initial state and $\mathbf{u}$ is the control input assumed to be constant over the interval $[0, T]$.

Training is performed offline by minimizing a composite loss

$$\mathscr{L} = \mathscr{L}_{\mathtt{IC}} + \lambda \mathscr{L}_{\mathtt{PHY}} \tag{3}$$

where $\lambda > 0$ is a weight factor. The initial-condition loss term enforces consistency at $t = 0$,

$$\mathscr{L}_{\mathtt{IC}} = \frac{1}{N_{\mathtt{IC}}} \sum_{j=1}^{N_{\mathtt{IC}}} \left\| \hat{\mathbf{x}}(0)^{(j)} - \mathbf{x}_0^{(j)} \right\|_2^2 \tag{4}$$

where $\hat{\mathbf{x}}(0)^{(j)}$ is the initial state prediction for $\left(\mathbf{x}_0^{(j)}, \mathbf{u}^{(j)}\right)$ pair. The physics-informed term is evaluated at collocation points $\left(t_k, \mathbf{x}_0^{(k)}, \mathbf{u}^{(k)}\right)$ with $t_k \in [0, T]$,

$$\mathscr{L}_{\mathtt{PHY}} = \frac{1}{N_{\mathtt{col}}} \sum_{k=1}^{N_{\mathtt{col}}} \left\| \mathbf{r}_{\mathbf{x}}\left(t_k, \mathbf{x}_0^{(k)}, \mathbf{u}^{(k)}\right) \right\|_2^2$$

where the residual is defined as:

$$\mathbf{r}_{\mathbf{x}}(t, \mathbf{x}_0, \mathbf{u}) = \dot{\hat{\mathbf{x}}}(t) - f(\hat{\mathbf{x}}(t), \mathbf{u})$$

in which the derivative of $\hat{\mathbf{x}}$ is realized by PyTorch Automatic Differentiation (Paszke, Gross, Chintala, et al. 2017).

Long-range simulation is achieved by chaining short-horizon predictions in an autoregressive form, i.e. the predicted terminal state of interval $K_{\mathtt{T}}$ is used as the initial state of interval $K_{\mathtt{T}} + 1$:

$$\hat{\mathbf{x}}_0[K_{\mathtt{T}} + 1] = \mathscr{N}_{\boldsymbol{\theta}}(T, \hat{\mathbf{x}}_0[K_{\mathtt{T}}], \mathbf{u}[K_{\mathtt{T}}]) \tag{5}$$

By recursively applying (5), PINC is able to simulate state trajectories over arbitrarily long horizons that are not fixed during training.

DOI
10.3384/ecp223137

2.2 PyTorch Custom `autograd.Function`

In standard deep learning workflows, backpropagation is handled by automatic differentiation engines. Specifically, PyTorch performs reverse-mode automatic differentiation by constructing a dynamic computational graph from tensor operations executed in the forward pass. When the relevant operations are fully supported by PyTorch, the associated gradients are generated automatically during backpropagation. However, this is not true when the forward pass computation involves external operations, whose contribution to the gradients is not available unless explicitly supplied.

To maintain the effective gradient flow, PyTorch provides the `torch.autograd.Function` class. By subclassing it, a user-defined operation is allowed to be inserted into the computational graph together with its backward rule. Consider an external operation h, with $\mathbf{x}$ and $\mathbf{y}$ denoting its input and output, respectively.

$$\mathbf{y} = h(\mathbf{x})$$

as a component of a scalar loss $\mathscr{L}$. The custom `autograd.Function` requires the definition of two static methods:

- `forward()`: performs the evaluation of h and saves necessary intermediate variables for the backward pass.
- `backward()`: receives the upstream gradient $\partial\mathscr{L}/\partial\mathbf{y}$ and possibly, the saved intermediate variables, and returns the gradient with respect to the inputs, calculated by a vector-Jacobian product (VJP) of the following form

$$\frac{\partial \mathscr{L}}{\partial \mathbf{x}} = \frac{\partial \mathscr{L}}{\partial \mathbf{y}} \frac{\partial h}{\partial \mathbf{x}}$$

In this way, a custom `autograd.Function` makes it possible to embed external model evaluations into a PyTorch-based training pipeline while preserving the gradient information required for end-to-end optimization.

2.3 FMI 3.0 Model Exchange

The Functional Mock-up Interface (FMI) is a tool-independent standard that defines a container and application programming interface (API) for Model Exchange (ME) or Co-Simulation (CS) of dynamic models across different simulation tools. The physical realization of this standard is the Functional Mock-up Unit (FMU) which is a specialized archive that can be distributed. The present work is concerned specifically with FMI 3.0 ME FMUs for which the solver remains external. It is particularly attractive since it supplies state derivative evaluations when the continuous states and control inputs are properly set, as illustrated in Figure 1, while avoiding the need to perform a full co-simulation step.

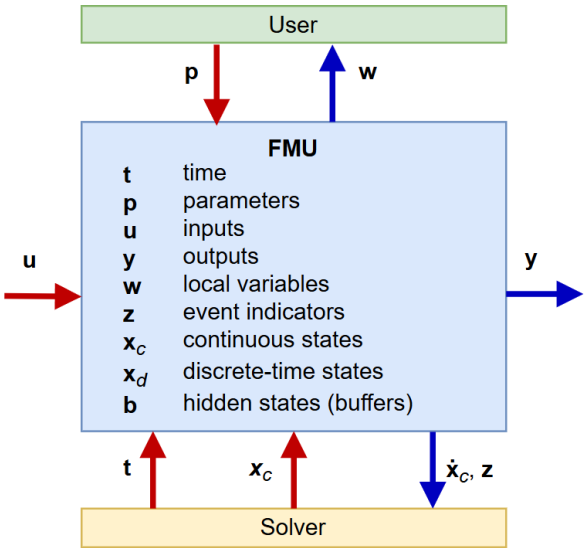


Figure 1. Data flow of FMI 3.0 ME FMU.

2.3.1 Adjoint Derivative

FMI 3.0 introduces several new features, one of the most relevant to this work is the support for adjoint derivatives. Suppose the FMU provides a differentiable mapping

$$\mathbf{v}_{unknown} = \mathbf{g}(\mathbf{v}_{known})$$

and denote the Jacobian by

$$\mathbf{J} = \frac{\partial \mathbf{g}}{\partial \mathbf{v}_{known}}$$

Given a seed vector $\mathbf{v}_{seed}$, the call of the FMI 3.0 API function `fmi3GetAdjointDerivative` provides a VJP,

$$\mathbf{v}^{\top}_{sensitivity} = \mathbf{v}^{\top}_{seed} \cdot \mathbf{J} \tag{6}$$

rather than $\mathbf{J}$ itself. This feature allows a direct calculation of a VJP without explicitly assembling the full Jacobian $\mathbf{J}$, and such a VJP is required in reverse-mode automatic differentiation, where gradients of a scalar loss are propagated backward through intermediate variables. Consequently, the adjoint derivative can provide a natural bridge between the FMU-based model evaluation and gradient-based neural network training procedure.

3 Problem Formulation and Assumptions

Consider a controlled dynamical system governed by

$$\dot{\mathbf{x}}(t) = f\left(\mathbf{x}(t), \mathbf{u}(t)\right), \quad \mathbf{x}(t) \in \mathbb{R}^{n_x}, \ \mathbf{u}(t) \in \mathbb{R}^{n_u}$$

The explicit analytical form of f is assumed to be unavailable or inconvenient to use directly. Instead, the system dynamics are accessible through an FMI 3.0 ME FMU, which is denoted by

$$\dot{\mathbf{x}}(t) = f_{\text{FMU}}\left(\mathbf{x}(t), \mathbf{u}(t)\right) \tag{7}$$

The objective is to train a PINC surrogate model of the form (2) that approximates the short-horizon state trajectory of (7) without requiring a manually implemented analytical right-hand side (RHS).

More specifically, in the present setting, the overall training loss preserves the same structure introduced in Section 2, but with an FMU-based physics-informed loss term:

$$\mathscr{L}_{\mathrm{PHY}}^{\mathrm{FMU}} = \frac{1}{N_{\mathtt{col}}} \sum_{k=1}^{N_{\mathtt{col}}} \left\| \mathbf{r}_{\mathrm{FMU}} \left(t_k, \mathbf{x}_0^{(k)}, \mathbf{u}^{(k)} \right) \right\|_2^2 \tag{8}$$

in which

$$\mathbf{r}_{\mathrm{FMU}}(t, \mathbf{x}_0, \mathbf{u}) = \dot{\hat{\mathbf{x}}}(t) - f_{\mathrm{FMU}}(\hat{\mathbf{x}}(t), \mathbf{u}) \tag{9}$$

is the physics-informed residual obtained from the FMU.

In this formulation, the FMU directly provides the model information required to evaluate the residual, thereby eliminating the need for an analytical function f. The main difficulty, however, lies in gradient propagation during training, since the FMU evaluation is external to the computational graph of the deep-learning framework.

3.1 Assumptions

The proposed FMU-based PINC training method is developed under the following assumptions:

1. **Continuous-time ODE representation.** The autonomous system is assumed to admit a continuous-time state-space description of the form (1). In the present work, the method is applied to ODE systems rather than general DAE systems.

2. **State consistency.** The state vector, as the output of the PINC, coincides with the continuous state vector used by the FMU for derivative evaluation, so that the network prediction can be directly used for FMU evaluation.

3. **Local smoothness in the training domain.** The system dynamics are assumed to be smooth over the region of interest. In particular, no state events, time events, or discontinuities should occur.

4. **Availability of an FMI 3.0 Model Exchange FMU.** The system model is available as an FMI 3.0 ME FMU, for which continuous states and inputs can be set externally and the corresponding state derivatives can be obtained directly without the need for a full time integration step.

5. **Accessible variables and value references.** The FMU model description exposes the variables corresponding to the continuous states, control inputs, and state derivatives, and their value references can be extracted consistently.

6. **Availability of adjoint derivatives.** The FMU supports the FMI 3.0 adjoint derivative for the variables involved in the residual evaluation. Consequently, it would be possible to compute the VJP required in the backward pass, at least for the state-related variables that influence training. [1]

7. **Constant control input over the training interval.** This is the assumption inherited from the original PINC formulation.

4 FMU-based PINC Training Method

This section presents the FMU-based PINC training framework. The main idea is to preserve the original PINC formulation while replacing the manually implemented analytical RHS with an FMI 3.0 ME FMU. Specifically, during training, a custom `autograd.Function` calculates the physics-informed loss on collocation points during the forward pass, while in the backward pass, the FMI 3.0 adjoint derivative is applied to enable gradient backpropagation for network parameter update.

4.1 System Information and FMU Preparation

Under the assumptions made in Section 3.1, the FMI 3.0 ME FMU exposes continuous states $\mathbf{x}$, inputs $\mathbf{u}$ and state derivatives $\dot{\mathbf{x}}$. These quantities, specifically, the corresponding unique handles for information exchange between the FMU and importer, namely, value references, are extracted from the model description and stored accordingly, making a structured access to the FMU possible during training. Additionally, with the number of state and input variables, the proposed method automatically determines the number of inputs and outputs of the neural network, setting them to $1+n_x+n_u$ and n_x, respectively.

Further, to reduce the computational overhead, the FMU is instantiated once by calling the API function `fmi3InstantiateModelExchange` and kept alive throughout the entire training process. This effectively avoids the computational overhead introduced by repeated instantiation and initialization of the FMU since for each collocation point, the continuous states and inputs must be set and the state derivatives must be evaluated by the FMU.

4.2 FMU-based Loss Calculation

The overall training procedure adopts the general idea introduced by Antonelo et al. (2024). The training dataset consists of $N_{\mathtt{IC}}$ initial-condition samples of the form $(0, \mathbf{x}_0, \mathbf{u})$, and $N_{\mathtt{col}}$ collocation points of the form $(t, \mathbf{x}_0, \mathbf{u})$, $t \in [0, T]$, both of which are randomly sampled, denoted by $\mathscr{D}_{\mathtt{IC}}$ and $\mathscr{D}_{\mathtt{col}}$ for clarity, respectively.

In the forward pass, the initial condition loss (4) is evaluated on $\mathscr{D}_{\mathtt{IC}}$. Given the FMU-based physics-informed

[1] The current work focuses on FMI 3.0 ME FMU providing adjoint derivative. If the available FMU is only capable of providing directional derivatives, a similar approach can be formulated.

DOI
10.3384/ecp223137

loss defined in (8), the key quantity to be evaluated by the custom `autograd.Function` is the residual (9):

$$\mathbf{r}_{\texttt{FMU}}(t,\mathbf{x}_0,\mathbf{u}) = \dot{\hat{\mathbf{x}}}(t) - f_{\texttt{FMU}}(\hat{\mathbf{x}}(t),\mathbf{u})$$

where $\hat{\mathbf{x}}(t) = \mathscr{N}_{\boldsymbol{\theta}}(t,\mathbf{x}_0,\mathbf{u})$ denotes the network prediction.

At each collocation point in $\mathscr{D}_{\texttt{col}}$, the network first predicts $\hat{\mathbf{x}}(t)$, then PyTorch automatic differentiation is used to calculate the derivative of the network output with respect to the input time $\dot{\hat{\mathbf{x}}}(t)$. The FMU is evaluated at $(\hat{\mathbf{x}}(t),\mathbf{u})$ to obtain the state derivatives $f_{\texttt{FMU}}(\hat{\mathbf{x}}(t),\mathbf{u})$.

The total loss can thus be assembled according to (3).

4.3 FMU-based Backpropagation

As previously mentioned, the challenge of the proposed method lies in the backpropagation, since the FMU evaluation is not natively supported by PyTorch. Thus, the custom `autograd.Function` must manually define the derivative of the FMU-related operation, ensuring that the chain rule can be applied across the computational graph.

Since $\mathscr{L}_{\texttt{IC}}$ is computed entirely by PyTorch-supported operations, the backpropagation of this loss component is carried out directly by the native automatic differentiation mechanism of PyTorch. It is thus reasonable to analyze only the $\mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}$ term. Note that $\mathbf{u}$ is part of the training sample and is therefore independent of the network parameters $\boldsymbol{\theta}$. Applying the chain rule to $\mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}$ gives

$$\frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \boldsymbol{\theta}} = \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \dot{\hat{\mathbf{x}}}}\frac{\partial \dot{\hat{\mathbf{x}}}}{\partial \boldsymbol{\theta}} + \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \hat{\mathbf{x}}}\frac{\partial \hat{\mathbf{x}}}{\partial \boldsymbol{\theta}} \tag{10}$$

so that the backward pass reduces to finding the gradients of $\mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}$ with respect to $\dot{\hat{\mathbf{x}}}$ and $\hat{\mathbf{x}}$.

Recalling the definition of $\mathbf{r}_{\texttt{FMU}}$ (9), it follows that

$$\frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \dot{\hat{\mathbf{x}}}} = \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \mathbf{r}_{\texttt{FMU}}}\frac{\partial \mathbf{r}_{\texttt{FMU}}}{\partial \dot{\hat{\mathbf{x}}}} = \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \mathbf{r}_{\texttt{FMU}}} \tag{11}$$

while

$$\begin{aligned}\frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \hat{\mathbf{x}}} &= \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \mathbf{r}_{\texttt{FMU}}}\frac{\partial \mathbf{r}_{\texttt{FMU}}}{\partial \hat{\mathbf{x}}} \\ &= -\frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \mathbf{r}_{\texttt{FMU}}}\frac{\partial f_{\texttt{FMU}}(\hat{\mathbf{x}},\mathbf{u})}{\partial \hat{\mathbf{x}}}\end{aligned} \tag{12}$$

To this end, define the seed vector in (6) using the upstream gradient such that

$$\mathbf{v}_{seed}^{\top} = \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \mathbf{r}_{\texttt{FMU}}}$$

The FMI 3.0 adjoint derivative function call returns

$$\mathbf{v}_{sensitivity}^{\top} = \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \mathbf{r}_{\texttt{FMU}}}\frac{\partial f_{\texttt{FMU}}(\hat{\mathbf{x}},\mathbf{u})}{\partial [\hat{\mathbf{x}},\mathbf{u}]}$$

from which the quantities in (11) and (12) can be written as

$$\begin{aligned}\frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \dot{\hat{\mathbf{x}}}} &= \mathbf{v}_{seed}^{\top} \\ \frac{\partial \mathscr{L}_{\texttt{PHY}}^{\texttt{FMU}}}{\partial \hat{\mathbf{x}}} &= -\mathbf{v}_{sensitivity,\hat{\mathbf{x}}}^{\top}\end{aligned}$$

where $\mathbf{v}_{sensitivity,\hat{\mathbf{x}}}^{\top}$ denotes the part of $\mathbf{v}_{sensitivity}^{\top}$ corresponding to the n_x state variables in the known-variable vector $\mathbf{v}_{known} = [\hat{\mathbf{x}}^{\top},\mathbf{u}^{\top}]^{\top}$.

Consequently, the custom `autograd.Function` returns the two gradients required by PyTorch without the need to assemble the full Jacobian thanks to the adjoint derivative feature of the FMI 3.0. Once these quantities are supplied, the remaining gradient propagation to the network parameters $\boldsymbol{\theta}$ is handled by standard PyTorch automatic differentiation through (10). In this way, the chain rule is preserved over the FMU-based residual, enabling the use of conventional gradient-based optimizers for training the PINC.

5 Case Studies and Results

This section presents case studies used to validate the proposed method: the Van der Pol oscillator and a mechanical system, where the analytical ODE is assumed to be difficult to obtain. In both cases, the FMU-based training strategy is evaluated through long-horizon autoregressive simulation and compared against a Dymola reference simulation. All the neural network training and the corresponding Dymola simulation were performed on a personal computer with an Intel® Core™ Ultra 9 185H CPU, 32 GB of RAM, and an NVIDIA GeForce RTX 4070 Laptop GPU with 8 GB memory.

5.1 Van der Pol Oscillator

The Van der Pol oscillator is adopted as the first case study, since it has been used as a standard benchmark in the PINC literature and a reference strategy based on the analytical ODE-based training is available (Antonelo et al. 2024). This makes it a suitable starting point for assessing whether the proposed FMU-based training strategy can recover comparable PINC behavior when the analytical RHS is no longer explicitly coded.

The governing equations of the Van der Pol oscillator are as follows:

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \mu(1-x_1^2)x_2 - x_1 + u\end{aligned} \tag{13}$$

where $\mu = 1$, $\mathbf{x} = [x_1\ x_2]^{\top}$ and u are the state vector and control input, respectively. In the present work, these equations are not directly used during the training. Instead, a Modelica model has been established implementing the system dynamics as shown in Figure 2, and exported as an FMI 3.0 ME FMU with `providesAdjointDerivatives="true"`.

Accordingly, the FMPy implementation successfully extracts the states, inputs and state derivatives as listed in Listing 1.

Listing 1. FMU model interface information of the Van der Pol oscillator.

```
FMI version: 3.0.1
The system has 2 derivatives, 2 states, 1
    ↪ inputs
```

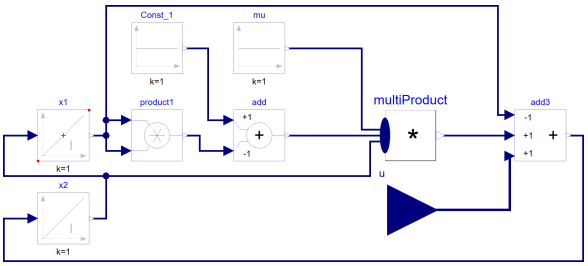


Figure 2. Modelica model of the Van der Pol oscillator.

```
Left-hand Side Element (Derivatives):
der(x1.y), der(x2.y)
Right-hand Side Element (States+Inputs):
x1.y, x2.y, u
```

which corresponds to the states and inputs defined in the analytical ODE (13).

To ensure meaningful comparison with the strategy reported in Antonelo et al. (2024), the same sampling region is adopted, namely

$$\texttt{x1.y},\ \texttt{x2.y} \in [-3,3], \quad \texttt{u} \in [-1,1]$$

and the same short training horizon is considered,

$$T = 0.5 \text{ s}$$

The network employed in this study is a fully connected feedforward neural network with 4 inputs and 2 outputs, and with 4 hidden layers of width 64. Based on a training set with $N_{\texttt{IC}} = 1000$ initial-condition samples and $N_{\texttt{col}} = 10^5$ collocation points, the network is trained with the Adam optimizer (Kingma and Ba 2017) using a learning rate 1.5×10^{-3} for 4000 epochs, with $\lambda = 1$ as the weight of $\mathscr{L}_{\texttt{PHY}}$.

In order to evaluate the performance of short-horizon prediction obtained from the trained PINC, 10^4 random initial state-input pairs are uniformly sampled from their corresponding admissible regions. For each sampled case, both the PINC and the FMU are deployed over the same short prediction horizon $T = 0.5$ s, same as the short training horizon. `SciPy.integrate.BDF` (Virtanen et al. 2020) is selected as the external solver to simulate the available ME FMU. The prediction accuracy is evaluated by comparing the state trajectories generated by both models on the common time grid $\{t_k\}_{k=1}^{N_t}$, $N_t = 21$ over the interval $[0,T]$. Let $\mathbf{x}^{\texttt{PINC}}(t_k) \in \mathbb{R}^{n_x}$ and $\mathbf{x}^{\texttt{FMU}}(t_k) \in \mathbb{R}^{n_x}$ denote the state vectors predicted by the PINC and the FMU at time instant t_k, respectively. The instantaneous state-prediction error is then defined as

$$\mathbf{e}(t_k) = \mathbf{x}^{\texttt{PINC}}(t_k) - \mathbf{x}^{\texttt{FMU}}(t_k)$$

Based on this error, the trajectory prediction quality over the entire short horizon is quantified by the Root Mean Square Error (RMSE),

$$\texttt{RMSE}_{\texttt{traj}} = \sqrt{\frac{1}{N_t n_x} \sum_{k=1}^{N_t} \sum_{i=1}^{n_x} \left(x_i^{\texttt{PINC}}(t_k) - x_i^{\texttt{FMU}}(t_k)\right)^2}$$

In addition, the quality of the predicted initial and terminal states is evaluated by the corresponding state-wise RMSE values,

$$\texttt{RMSE}_{\texttt{init}} = \sqrt{\frac{1}{n_x} \sum_{i=1}^{n_x} \left(x_i^{\texttt{PINC}}(0) - x_i^{\texttt{FMU}}(0)\right)^2}$$

and

$$\texttt{RMSE}_{\texttt{final}} = \sqrt{\frac{1}{n_x} \sum_{i=1}^{n_x} \left(x_i^{\texttt{PINC}}(T) - x_i^{\texttt{FMU}}(T)\right)^2}$$

The statistical results are demonstrated in Figure 3, box plots of $\texttt{RMSE}_{\texttt{init}}$, $\texttt{RMSE}_{\texttt{final}}$, and $\texttt{RMSE}_{\texttt{traj}}$, showing that a low prediction error over the entire short-horizon trajectory can be achieved via the PINC simulation.

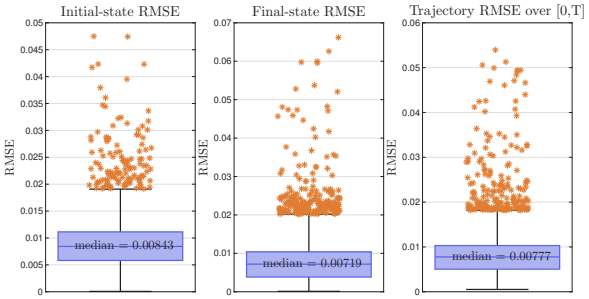


Figure 3. Box plot of prediction quality metrics across sampled initial conditions and constant inputs: initial state RMSE, final state RMSE, and trajectory RMSE over the short horizon $[0,T]$.

Meanwhile, the computational efficiency is quantified by the execution time required to complete one short-horizon simulation with N_t intermediate evaluation instants. As it can be observed from Figure 4, the trained PINC significantly accelerates the computational speed by around 30 times, compared to the simulation obtained from FMU.

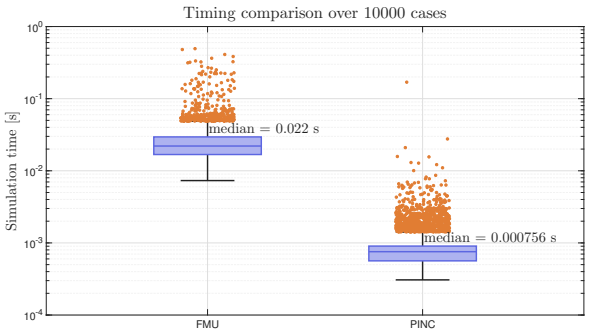


Figure 4. Box plot of short-horizon simulation time for the FMU model and the trained PINC over the same set of sampled initial conditions and inputs. The median FMU and PINC simulation times are 0.022 s and 7.56×10^{-4} s, respectively.

In order to compare the performance of long-horizon simulation, a second PINC, with the same structure, is trained with the conventional ODE-based framework (Antonelo et al. 2024), using the same hyperparameters

DOI
10.3384/ecp223137

and optimizer settings as the FMU-based PINC training. In a single indicative wall-clock measurement, the required time for the ODE-based training is approximately 6 h 1 min 43 s, while the FMU-based training required approximately 7 h 59 min 10 s, corresponding to an increase of 32.47% in this test case.

Both PINCs are evaluated in an autoregressive long-horizon simulation over a duration of 15 s. The input amplitude is randomly sampled in $[-1,1]$ and held constant for each short interval $T = 0.5$ s, while the initial state of x_1 and x_2, and correspondingly `x1.y` and `x2.y`, are randomly sampled from $[-3,3]$. The same input sequence and initial values are applied to the Modelica model in Dymola to generate the reference trajectory.

The autoregressive long-horizon simulation results are presented in Figure 5. As shown, both PINC models, trained using the FMI 3.0 ME FMU and the analytical ODE, accurately capture the system response over the full simulation horizon relative to the Dymola simulation baseline. The corresponding absolute state-prediction errors are presented in Figure 6. Using the Dymola simulation trajectories as reference, the FMU-based PINC achieves RMSE values of $\mathrm{RMSE}_{\texttt{x1.y}} = 0.021$ and $\mathrm{RMSE}_{\texttt{x2.y}} = 0.033$, with maximum absolute errors of 0.058 and 0.090, respectively. The ODE-based PINC yields $\mathrm{RMSE}_{x_1} = 0.034$ and $\mathrm{RMSE}_{x_2} = 0.043$, with corresponding maximum absolute errors of 0.081 and 0.159. For the considered long-horizon trajectory, both PINCs achieve comparable prediction accuracy, with the FMU-based PINC yielding slightly lower RMSE and maximum absolute errors. This confirms that the FMI 3.0 ME FMU can be used to train the PINC for the Van der Pol oscillator while achieving a performance comparable to that obtained with the analytical RHS expression.

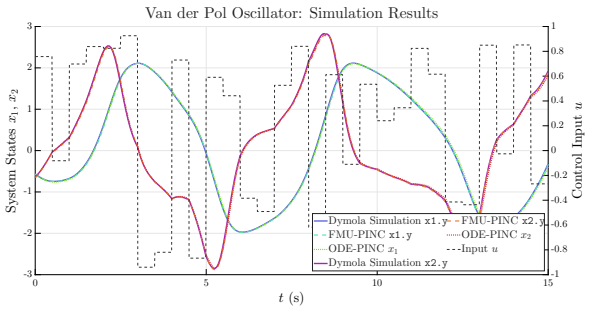


Figure 5. Autoregressive long-horizon simulation of the Van der Pol oscillator under piecewise-constant random control input (gray dashed line). The state evolution of Dymola simulation corresponds to the solid lines, the state predictions obtained from FMU-based and ODE-based training are plotted with colored dashed lines and dotted lines, respectively.

5.2 Mechanical System

The second case study considers a two-degree-of-freedom translational mechanical system that can be expressed as an ODE system. As shown in Figure 7, the left mass is connected to a fixed boundary through a spring-

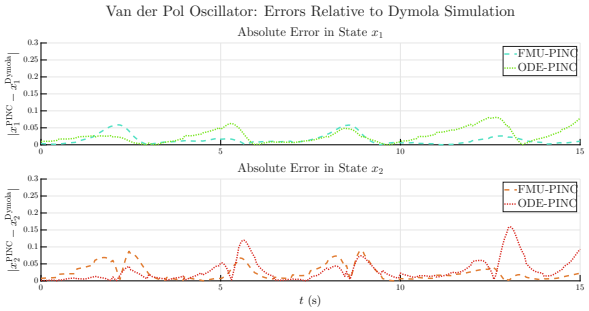


Figure 6. Absolute state-prediction errors for the Van der Pol oscillator, using the Dymola simulation as reference.

damper element with stiffness $c = 5$ N/m and damping $d = 2$ N·s/m. The two masses are coupled through an intermediate compliant connection formed by a spring with stiffness $c = 3$ N/m and a second spring-damper element with stiffness $c = 4$ N/m and damping $d = 3$ N·s/m. Both masses have $m = 1$ kg. In addition, two independent forces are applied to the system as external inputs. For this mechanical system, the analytical ODEs are not derived in advance; instead, only the FMI 3.0 ME FMU is exported with `providesAdjointDerivatives="true"` for training.

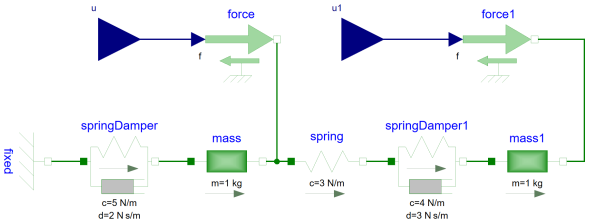


Figure 7. Modelica model of an example mechanical system.

As a first step, the proposed method identified the listed states, inputs and state derivatives, as shown in Listing 2.

Listing 2. FMU model interface information of the Mechanical System.

```
FMI version: 3.0.1
The system has 5 derivatives, 5 states, 2
    ↪ inputs
Left-hand Side Element (Derivatives):  der(
    ↪ springDamper.s_rel), der(springDamper.
    ↪ v_rel), der(mass1.s), der(mass1.v),
    ↪ der(springDamper1.s_rel)
Right-hand Side Element (States+Inputs):
    ↪ springDamper.s_rel, springDamper.v_rel
    ↪ , mass1.s, mass1.v, springDamper1.
    ↪ s_rel, u, u1
```

Sampling ranges of the state variables for constructing the training set were determined empirically through a preliminary simulation-based analysis. Specifically, given input ranges

$$\texttt{u} \in [-1,1], \quad \texttt{u1} \in [-2,2]$$

and an interval

$$T = 1 \text{ s}$$

The Modelica model is first excited with stochastic input using the block `Modelica.Blocks.Noise.UniformNoise` while setting its parameter `samplePeriod` to T, and `y_min`, `y_max` to the corresponding limits of each input. A sufficiently long-duration simulation is performed to observe the extrema of each state variable, and the sampling range for each state variable is then defined as the interval between these observed maxima and minima, extended by some margin. Consequently, these ranges are defined as

$$\begin{aligned} \texttt{springDamper.s_rel} &\in [-1,1] \\ \texttt{springDamper.v_rel} &\in [-1.5,1.5] \\ \texttt{mass1.s} &\in [-3,3] \\ \texttt{mass1.v} &\in [-3,3] \\ \texttt{springDamper1.s_rel} &\in [-1,1] \end{aligned}$$

within which $N_{\texttt{IC}} = 1000$ initial-condition samples and $N_{\texttt{col}} = 2 \times 10^5$ collocation points are generated.

The employed network therefore has 8 inputs and 5 outputs. The width and depth of the hidden part remain the same as in the Van der Pol case. Training is performed for 5000 epochs while other hyperparameters remain unchanged.

After training, the same short-horizon prediction evaluation is performed, and the corresponding metrics are reported in Figure 8 and Figure 9, respectively. In this case, the PINC is capable of simulating the system dynamics with acceptable errors, while it provides roughly a 100× speedup, compared to the simulation of FMU. A possible explanation for this even larger acceleration could be that, since in this mechanical system the dynamics are more complex than the Van der Pol oscillator, the external solver requires greater computational effort to execute the time integration. Meanwhile, the PINC structure remains similar to the previous case, thus its inference time may not change drastically.

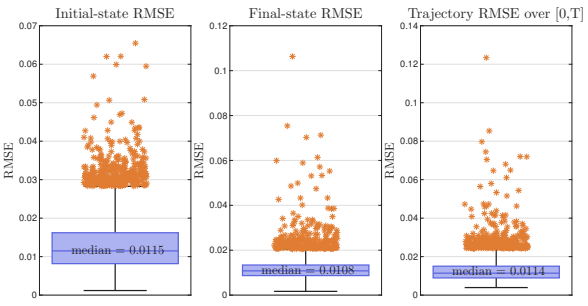


Figure 8. Box plot of prediction quality metrics across sampled initial conditions and constant inputs: initial state RMSE, final state RMSE, and trajectory RMSE over the short horizon $[0, T]$.

Similarly, the resulting PINC is applied for a long-horizon simulation whose results are compared to the simulation in Dymola, with the same randomly sampled input sequence and a fixed initial state $\mathbf{x} = 0$. The state evolutions and the absolute state-prediction errors are plotted in Figure 10 and Figure 11, respectively. Using the Dymola simulation as reference, the state-wise RMSE and maximum absolute errors are summarized in Table 1. These quantitative results are consistent with the close agreement observed between the predicted and reference trajectories over the 30 s simulation, indicating that the system dynamics encapsulated in the FMU have been successfully learned by the neural network.

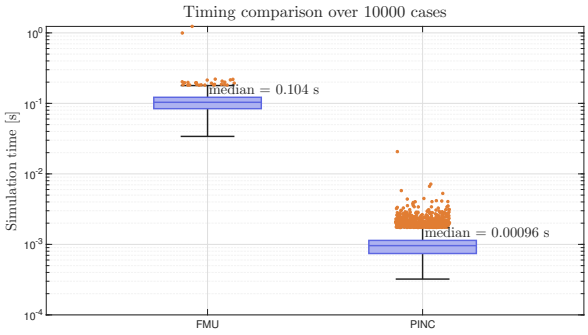


Figure 9. Box plot of short-horizon simulation time for the FMU model and the trained PINC over the same set of sampled initial conditions and inputs. The median FMU and PINC simulation times are 0.104 s and 9.6×10^{-4} s, respectively.

Table 1. Error metrics using Dymola simulation as reference

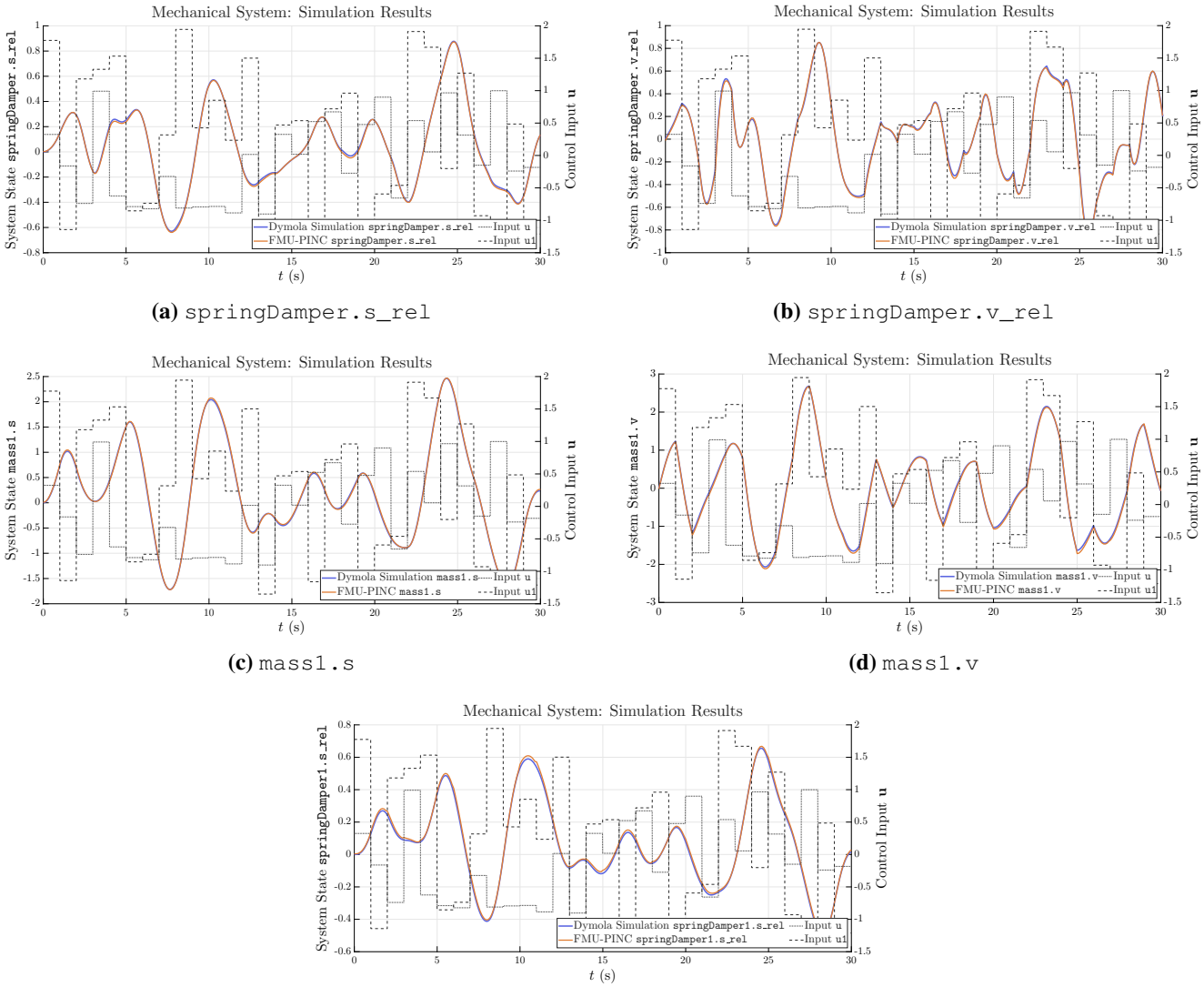
State	RMSE	Max. Abs. Error
`springDamper.s_rel`	0.008	0.018
`springDamper.v_rel`	0.012	0.043
`mass1.s`	0.017	0.045
`mass1.v`	0.035	0.087
`springDamper1.s_rel`	0.012	0.032

6 Conclusion

This paper presented an FMU-based training framework for Physics-Informed Neural Nets for Control. The proposed approach enables PINC training directly from FMI 3.0 Model Exchange FMUs, without requiring a manually implemented analytical right-hand side. In the forward pass, the physics-informed residual is evaluated by querying the FMU for state derivatives. In the backward pass, gradient propagation is enabled through FMI 3.0 adjoint derivatives integrated into PyTorch through a custom `autograd.Function`.

The method was validated on two case studies. For the Van der Pol oscillator, the FMU-based implementation achieved prediction quality comparable to that of a conventional ODE-based PINC. For the mechanical system example, the proposed framework successfully learned the system dynamics directly from the FMU and reproduced the Dymola reference trajectories in a long-horizon autoregressive simulation. The timing comparison further

DOI
10.3384/ecp223137



(a) `springDamper.s_rel` **(b)** `springDamper.v_rel`

(c) `mass1.s` **(d)** `mass1.v`

(e) `springDamper1.s_rel`

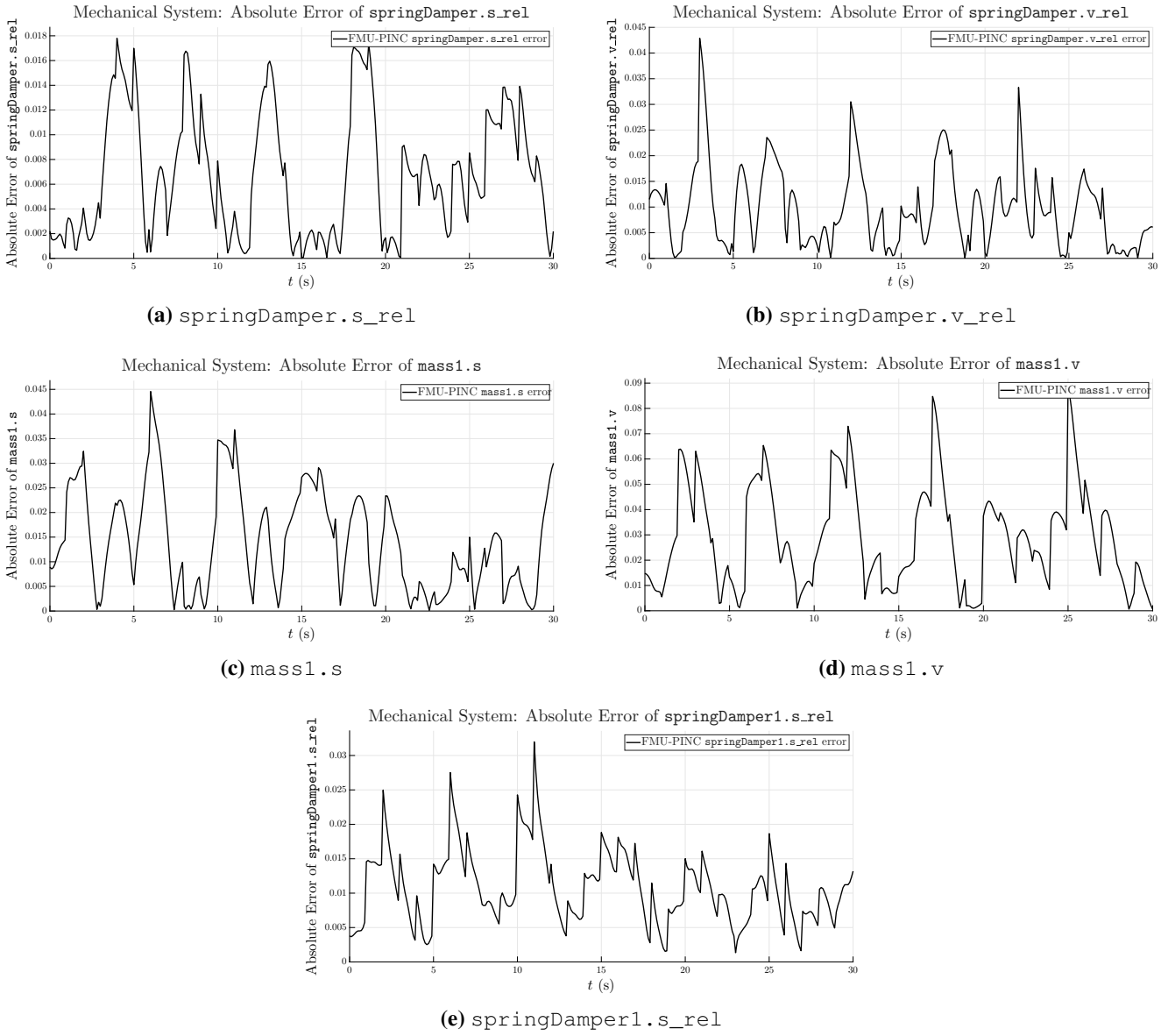
Figure 10. Autoregressive long-horizon simulation of the mechanical system under piecewise-constant random control inputs (gray dashed and dotted lines). In the plot of each identified state, the blue line plots the Dymola simulation baseline, while the orange line plots the PINC simulation result.

indicates that PINC can serve as an efficient surrogate for FMU models, substantially reducing simulation cost, with potentially larger gains observed for more complex systems.

These results indicate that the proposed training framework is applicable when the system dynamics are encapsulated in an FMI 3.0 ME FMU, the PINC outputs can be mapped directly to the FMU continuous states, and the dynamics remain smooth within the considered operating region. Under these conditions, the FMI 3.0 ME FMU can serve as a reusable source of physics information for training control-oriented neural surrogates. This is particularly relevant to engineering workflows in which high-fidelity Modelica models already exist.

However, the current formulation has several limitations. It is restricted to continuous-time ODE systems and does not explicitly handle DAEs, state or time events, or discontinuities. Furthermore, the implementation requires an FMI 3.0 ME FMU that provides adjoint derivatives, which, additionally, limits its applicability to a subset of FMUs exported by software tools that fully support this feature. Finally, although the reported training time comparison may not be representative of general cases, FMU-based training is slower than the pure ODE-based training because the FMU is evaluated on the CPU, while network training is performed on the GPU. These computations are not overlapped: FMU evaluation leaves the GPU idle, whereas network forward and backward passes leave the CPU-side FMU evaluation inactive. As a result, CPU–GPU transfers, synchronization, and possible underutilization of both devices introduce additional overhead relative to conventional ODE-based training.

Future work will therefore focus on relaxing the assumptions, extending the framework to more complex



(a) `springDamper.s_rel` **(b)** `springDamper.v_rel`

(c) `mass1.s` **(d)** `mass1.v`

(e) `springDamper1.s_rel`

Figure 11. Absolute state-prediction errors for the mechanical system, using the Dymola simulation as reference.

Modelica models, investigating DAE systems and improving the computational efficiency of the training implementation.

References

Amer, Abdelhakim, David Felsager, Yury Brodskiy, and Andriy Sarabakha (2025). "Modelling of Underwater Vehicles using Physics-Informed Neural Networks with Control". In: *2025 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. DOI: 10.1109/IJCNN64981.2025.11228873.

Antonelo, Eric Aislan, Eduardo Camponogara, Laio Oriel Seman, Jean Panaioti Jordanou, Eduardo Rehbein de Souza, and Jomi Fred Hübner (2024). "Physics-informed neural nets for control of dynamical systems". In: *Neurocomputing* 579, p. 127419. DOI: https://doi.org/10.1016/j.neucom.2024.127419.

Baydin, Atılım Günes, Barak A. Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind (2017). "Automatic differentiation in machine learning: a survey". In: *Journal of Machine Learning Research* 18, pp. 5595–5637.

Dassault Systèmes (2026). *FMPy: A Python library to simulate Functional Mock-up Units (FMUs)*. URL: https://github.com/CATIA-Systems/FMPy.

Farea, Amer, Olli Yli-Harja, and Frank Emmert-Streib (2024). "Understanding Physics-Informed Neural Networks: Techniques, Applications, Trends, and Challenges". In: *AI* 5, pp. 1534–1557. DOI: 10.3390/ai5030074.

Junghanns, Andreas et al. (2021). "The Functional Mock-up Interface 3.0 - New Features Enabling New Applications". In: *Proceedings of 14th Modelica Conference 2021, Linköping, Sweden, September 20-24, 2021*, pp. 17–26. DOI: 10.3384/ecp2118117.

Karpatne, Anuj, Ramakrishnan Kannan, and Vipin Kumar (2022). *Knowledge guided machine learning: Accelerating discovery using scientific knowledge and data*. CRC Press. ISBN: 9781003143376.

Kingma, Diederik P. and Jimmy Ba (2017). *Adam: A Method for Stochastic Optimization*. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980.

Luo, Kuang et al. (2025). "Physics-informed neural networks for PDE problems: A comprehensive review". In: *Artificial Intelligence Review* 58, p. 323.

Martín Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. URL: https://www.tensorflow.org/.

Modelica Association (2026). *Functional Mock-up Interface*. URL: https://fmi-standard.org/ (visited on 2026-04-04).

Nicodemus, Jonas, Jonas Kneifl, Jörg Fehr, and Benjamin Unger (2022). "Physics-informed Neural Networks-based Model Predictive Control for Multi-link Manipulators". In: *IFAC-PapersOnLine* 55.20, pp. 331–336. DOI: https://doi.org/10.1016/j.ifacol.2022.09.117.

Paszke, Adam, Sam Gross, Soumith Chintala, et al. (2017). "Automatic Differentiation in PyTorch". In: *Proceedings of the 31st Conference on Neural Information Processing Systems*.

Paszke, Adam, Sam Gross, Francisco Massa, et al. (2019). "PyTorch: an imperative style, high-performance deep learning library". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Curran Associates Inc.

Raissi, Maziar, Paris Perdikaris, and George E Karniadakis (2019). "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational Physics* 378, pp. 686–707. DOI: https://doi.org/10.1016/j.jcp.2018.10.045.

Thummerer, Tobias, Fabian Jarmolowitz, Daniel Sommer, and Lars Mikelsons (2025). "Br (e) aking the boundaries of physical simulation models: Neural Functional Mock-up Units for Modeling the Automotive Braking System". In: *Modelica Conferences*, pp. 575–583.

Thummerer, Tobias, Lars Mikelsons, and Josef Kircher (2021). "NeuralFMU: towards structural integration of FMUs into neural networks". In: *Modelica Conferences*, pp. 297–306.

Ultsch, Johannes, Kenan Ahmic, and Jonathan Brembeck (2025). "train_fmu_gym: A Functional Mock-up Unit-Based Framework to Train Reinforcement Learning Agents for Multi-Physical Systems". In: *IEEE Access* 13, pp. 208845–208867. DOI: 10.1109/ACCESS.2025.3636787.

Virtanen, Pauli et al. (2020). "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17, pp. 261–272. DOI: 10.1038/s41592-019-0686-2.

Wrede, Konstantin, Chenzi Huang, Tommy Wohlfahrt, and Nick Hartmann (2024). "FMUGym: An Interface for Reinforcement Learning-based Control of Functional Mockup Units under Uncertainties". In: *31st Int. Workshop on Intelligent Computing in Engineering, EG-ICE*, pp. 647–656.

Zhang, Weiwei, Wei Suo, Jiahao Song, and Wenbo Cao (2026). "Physics-informed neural networks (PINNs) as intelligent computing technique for solving partial differential equations: Limitation and future prospects". In: *SCIENCE CHINA Physics, Mechanics & Astronomy* 69, p. 214602.

DOI
10.3384/ecp223137

Simulation and Verification of Marine Power System Based on Data, Models, and Mock-ups

Ziyi Zou [1] Luotao Xie [1] Ruirui Zeng [2] Yangzeng Xu [3]
Guobing Chen [1*] Zichun Yang [1]

[1] Naval University of Engineering, China

[2] Suzhou Tongyuan Software&Control Technology Co., Ltd, China

[3] Shanghai Hydraulics Pneumatics & Seals Trade Association, China

Abstract

Conventional simulation methods relying on single models or empirical formulas fail to support intelligent, full-lifecycle simulation and verification for complex systems such as marine power systems. Multi-domain modeling languages like Modelica, while powerful, remain at the tool level and cannot resolve bottlenecks in data fusion, high-confidence simulation, or closed-loop verification. To overcome these limitations, this paper proposes a system simulation methodology based on data, models, and mock-ups. The methodology integrates full-lifecycle multi-source data into simulation, employs simulation models to construct a digital mock-up, and achieves tight integration and closed-loop iteration among data, models, and the physical mock-up among data, models, and the physical mock-up, thereby establishing high-confidence system-level simulation across design, testing, service, and maintenance. A digital twin of a marine power system is developed using Modelica with real-time multi-dimensional data interaction. Experimental validation under four typical operating conditions (shutdown, 0kW, 30kW, 60kW) shows that simulated key parameters (steam pressure, turbine speed, output power) agree closely with measured values, with all relative errors controlled within 8%. These results demonstrate the feasibility, practicality, and engineering applicability of the proposed methodology, confirming that integrating data, models, and mock-ups effectively supports intelligent full-lifecycle operation and maintenance of complex systems and provides an implementable pathway for the intelligent transformation of systems engineering.

Keywords: System simulation methodology; Modelica; Data, models, and mock-ups; Marine power system; Digital twin

1 Introduction

The marine power system, as the core energy and propulsion unit of a ship, has operational reliability that directly determines vessel safety and stability. Under complex marine environments and varying operating conditions, this system features strong multi-physics coupling, making system-level simulation with sufficient fidelity is crucial for improving overall ship reliability and performance. As equipment system complexity increases, traditional simulation methods based on single models or empirical formulas can no longer accurately capture the dynamic evolution mechanisms and nonlinear response characteristics of the system. In practical engineering applications, such methods face multiple challenges: lengthy model development cycles, high difficulty in cross-domain coupled modeling, trade-offs between simulation accuracy and real-time capability, and a lack of systematic collaborative verification. Overcoming the limitations of conventional simulation methods and achieving high-fidelity dynamic simulation and verification of complex systems has thus become an urgent issue in high-fidelity system simulation for marine power technology.

Multi-domain unified modeling and simulation verification, as an important trend in complex system simulation, aims to transcend the capability constraints of traditional single-domain simulation. Among the available techniques, the one-dimensional lumped-parameter modeling approach has become the mainstream system-level simulation paradigm. Owing to its high computational efficiency and real-time simulation capability, this approach represents a promising pathway for analyzing the dynamic properties of complex systems. Tools such as Simulink and AMESim abstract hydraulic, control, mechanical, and other subsystems into mathematical equations and are widely used in control system design and hardware-in-the-loop testing[1-3]. Moreover, the Modelica language, distinguished by its equation-based, object-oriented, and acausal modeling paradigm, demonstrates significant advantages in multi-domain unified modeling[4-5]. Compared with signal-flow-oriented modeling in Simulink, Modelica represents physical systems declaratively through acausal equations. Although Simscape and AMESim also support physical-network modeling, Modelica provides a language-level, tool-independent specification that is advantageous for reusable cross-domain model libraries and standardized model exchange. It is particularly well-suited for

modeling environments that involve complex systems spanning mechanical, thermal, electrical, control, and other domains[6-10].

Recent studies have applied Modelica to marine power and propulsion systems at different scopes. Zeng et al. combined a mechanism model with multi-source data to construct a digital twin of a marine condensate feedwater subsystem, achieving static errors below 2% and consistent dynamic trends[11]. Their subsequent system-level study developed a Modelica-based multi-domain digital model of a marine steam power system for variable-condition and fault simulation, with key-parameter errors within ±5%[12]. Wang et al. further validated a modular Modelica model of a marine wet-steam turbine against a physical test platform under steady and variable operating conditions, with errors below 5%[13]. For virtual-physical verification, Zhang et al. integrated OpenModelica with Arduino to implement hardware-in-the-loop simulation of a nuclear-powered engineering ship, demonstrating steady- and transient-state hardware feedback[14]. These studies confirm the applicability of Modelica to hierarchical modeling, model calibration, and virtual-physical verification in marine systems; however, they mainly address specific subsystems, modeling tasks, or verification scenarios and do not yet provide a standardized methodology that continuously coordinates full-lifecycle heterogeneous data, simulation models, digital mock-ups, and physical mock-ups.

Accordingly, the remaining gap is not the lack of a multi-domain modeling language, but the lack of an engineering methodology that organizes data, models, and mock-ups into a closed-loop process. Modelica alone does not prescribe how multi-source data should be fused, how models should be calibrated and updated across lifecycle stages, or how digital and physical mock-ups should collaborate for high-confidence verification. This limits the transfer of isolated modeling achievements into lifecycle-oriented engineering practice for complex marine power systems.

In this paper, we adopt the following definitions: a digital mock-up refers to a system-level virtual integration platform that aggregates various simulation models (component, subsystem, and system-level) along with geometric, functional, and performance descriptions, enabling multi-domain co-simulation and parameter exchange. A digital twin, on the other hand, extends the digital mock-up by establishing real-time bidirectional data linkage with the physical counterpart, supporting online state synchronization, adaptive model updating, and predictive decision-making. Thus, the digital mock-up serves as the core simulation engine, while the digital twin represents the complete cyber-physical system with live data interaction—a distinction that is maintained throughout this work.

To address these limitations, this paper proposes a system simulation methodology based on data, models, and mock-ups, focusing on multi-source data-driven modeling, model–mock-up collaboration, and digital twin-oriented verification. Using a marine power system as a case, a system-level simulation model is developed with Modelica, and the methodology is applied to design the system simulation architecture, analyze simulation results, and realize digital twin verification, providing a methodological reference for standardizing simulation workflows and promoting engineering applications.

2 System Simulation Methodology

Addressing the challenges that complex systems face in multi-domain unified modeling, high-fidelity simulation, and model verification, this paper proposes a system simulation methodology with data serving as the carrier and bidirectionally driven by simulation models and mock-ups. This methodology conducts system simulation computations based on full-process data and uses simulation models to support the construction of a digital mock-up, ultimately achieving high-fidelity simulation and dynamic evolution throughout the entire life cycle of a complex system. The following exposition is organized around three aspects: multi-source data-driven modeling, model–mock-up fusion, and the system simulation methodology.

2.1 Multi-Source Data-Driven Modeling

Traditional simulation methods are limited to theoretical modeling in the equipment design stage, lacking integration of full-lifecycle multi-source heterogeneous data for dynamic correction, iteration, or scenario expansion, thus failing to support full-lifecycle performance assessment, state extrapolation, and fault prediction. To address this, this paper proposes a multi-source data-driven system simulation methodology, which takes full-lifecycle data (design parameters, bench test data, field monitoring data, historical fault records) as the core to establish a simulation framework covering test verification and service operation.

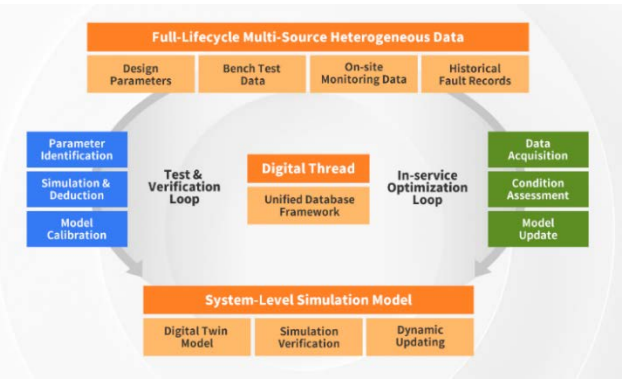


Figure 1. Multi-source data-driven simulation framework. Note: The test verification loop achieves model parameter identification, simulation extrapolation and online correction

DOI
10.3384/ecp223149

through bidirectional feedback between test/design data and the simulation model; the service optimization loop uses field monitoring and fault data to calibrate parameters and expand simulation scenarios; the two loops interwork via a unified database and digital thread to support model iterative optimization.

2.2 Model–Mock-Up Fusion

Traditional practice often shows significant discrepancies between digital mock-ups and actual equipment due to model simplification and insufficient effective data. To address this, this paper proposes a model–mock-up fusion architecture enabling deep collaboration among digital models, digital mock-ups, and physical mock-ups. Digital models are precise mathematical descriptions of specific physical fields or subsystems, categorized by object (component, subsystem, system) and physical domain. The digital mock-up integrates various simulation models with geometric, functional, and performance models into a system-level simulation platform via parameter transfer and model order reduction. A bidirectional interaction mechanism is established: the forward link uses high-fidelity models to guide physical mock-up experiment design; the reverse link calibrates simulation models with measured data, supplemented by data-driven surrogate models for subsystems with unclear mechanisms. Based on Modelica and standardized interfaces, different models achieve parameter transfer and reuse across design, verification, and operation and maintenance stages.

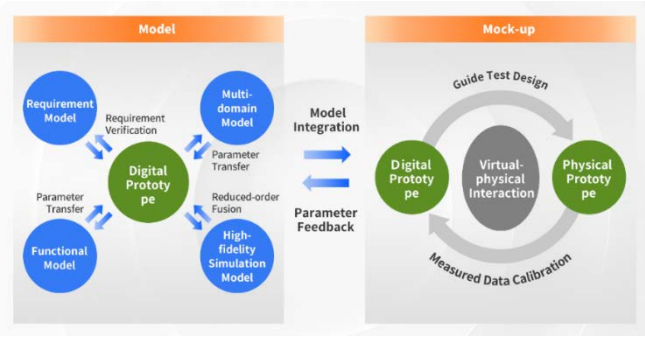


Figure 2. Model–mock-up fusion architecture. Note: The architecture realizes deep collaboration among digital models, digital mock-up and physical mock-up; it completes model calibration, experiment design guidance and multi-domain model integration to support full-lifecycle model iteration and state synchronization.

As shown in Figure 2, the proposed architecture realizes deep collaboration among digital models, digital mock-up, and physical mock-up, with core functions including model calibration by physical data, experiment design guidance by simulation models, and multi-domain model integration to support full-lifecycle model iteration and state synchronization. In summary, this mechanism resolves the isolation between models and mock-ups in traditional simulation, laying an architectural foundation for full-lifecycle high-fidelity simulation.

2.3 System Simulation Methodology Based on Data, Models, and Mock-ups

Integrating the above ideas, this paper develops a system simulation methodology featuring the triadic synergy of data, models, and mock-ups. Taking data as the core driving source and models/mock-ups as two key carriers, this methodology establishes a full-lifecycle closed-loop simulation and verification framework, as shown in Figure 3. The triadic synergy architecture consists of three layers.

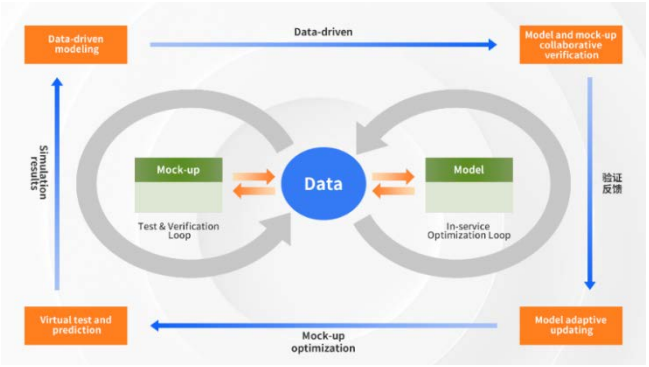


Figure 3. Triadic synergy methodology of data, models, and mock-ups. Note: It consists of data, model and mock-up layers, with a four-step closed-loop simulation flow, realizing full-lifecycle simulation and verification.

The data layer is responsible for the acquisition, fusion and storage of full-lifecycle multi-source heterogeneous data, providing fundamental support for parameter identification, boundary condition setting and uncertainty quantification. The model layer integrates multiple simulation models, adopts a hybrid strategy of physics-based and surrogate models to balance accuracy and computational efficiency, and continuously improves model fidelity through parameter sensitivity analysis and data-driven calibration. The mock-up layer includes physical and digital mock-ups: the former provides high-confidence data for model validation, while the latter integrates simulation models to generate virtual test cases and guide physical mock-up design.

The closed-loop simulation flow includes four steps: data-driven modeling for parameter identification and uncertainty quantification; model–mock-up collaborative verification for digital mock-up construction and simulation extrapolation; model adaptive updating via online parameter calibration and surrogate modeling; and virtual testing and prediction for dynamic response evaluation and fault evolution analysis to support decision-making.

Compared with traditional methods, the proposed methodology has four improvements: the driving core shifts from model-driven to triadic synergy; verification changes from late-stage physical testing to real-time virtual-physical interactive verification; optimization

extends from static tuning to full-lifecycle dynamic adaptive optimization; data value evolves from unidirectional transmission to multi-directional interaction. It provides a standardizable framework for complex equipment simulation, laying a methodological foundation for digital twin applications.

3 System Simulation Model Construction

To validate the effectiveness of the system simulation methodology proposed in this paper, a marine steam turbine for power generation is taken as the research object. Using the Modelica language, modular decomposition and equation-based modeling are performed to construct a solvable and scalable system-level simulation model. This model can not only accurately capture the thermal and mechanical dynamic characteristics of the steam turbine under multiple operating conditions, but also serves as the core carrier for subsequent data fusion and calibration as well as virtual–physical interactive verification, thereby providing a model foundation for high-fidelity simulation analysis and engineering applications.

3.1 Component Model Construction

Based on the marine power system architecture, a steam power system simulation model library is constructed using the Modelica multi-domain unified modeling specification. The steam power system mainly includes equipment such as steam turbines, condenser and valves. Below are the physical principles of some of the main components and equipment models.

- **Steam turbine**

The main function of a steam turbine model is to calculate the power output generated by high-temperature and high-pressure steam through expansion. In this article, a mathematical model of the steam turbine is established based on the Rankine cycle formula. The steam flow rate at the outlet of the steam turbine is given by:

$$\frac{G_E}{G_{E0}} = \sqrt{\frac{p_E^2 - p_L^2}{p_{E0}^2 - p_{L0}^2}} \sqrt{\frac{T_{E0}}{T_E}} \tag{1}$$

Steam outlet specific enthalpy:

$$h_L = h_E - \Delta h_i \eta_i \tag{2}$$

Turbine power:

$$P_t = G_E(h_E - h_L) \tag{3}$$

Among them, G_{E0} represents the rated steam flow rate, p_{E0} represents the rated steam inlet pressure, T_{E0} represents the rated steam inlet temperature, p_{L0} represents the rated steam outlet pressure, p_L represents the actual steam outlet pressure.

- **Condenser**

The condenser is a common multi-inlet shell and tube heat exchanger in the thermal system. The cooling water flows through the pipe side by the circulating pump, and the steam passes over the outside of the tube in the shell side for condensation and heat release. The heat is transferred to the cooling water through the tube wall, and the lower temperature cooling water keeps the condenser high vacuum through continuous circulation.

Shell side steam mass conservation equation:

$$\frac{dm_g}{dt} = q_{m,in} + \sum q_{m,in}^{OT} + q_{m,HW} - q_{m,p} - q_{m,con} \tag{4}$$

In this equation, m_g represents the steam mass within the condenser, $q_{m,in}$ denotes the steam mass flow rate of exhaust steam from the turbine , $q_{m,HW}$ indicates the steam mass flow rate entering the steam space through the heat well , $q_{m,p}$ is the steam mass flow rate extracted by the extraction steam trap, $q_{m,con}$ refers to the condensate flow rate at the steam space outlet of the condenser.

Shell side steam energy conservation equation:

$$\frac{d(m_g h_g)}{dt} = q_{m,in} h_{in} + \sum \left(q_{m,in}^{OT} h_{in}^{OT} \right) - q_{m,p} h_p - q_{m,con} h_{con} \tag{5}$$

Where h_g denotes the enthalpy of steam in the condenser, h_{in} represents the enthalpy of exhaust steam from the turbine, h_p indicates the extracted steam enthalpy from the extraction steam trap, h_{con} is the enthalpy of condensate at the steam space outlet of the condenser.

Tube side thermal balance equation:

$$Q = KA\Delta T_m = q_{m,W}(h_{W,L} - h_{W,E}) = q_{m,G}(h_G - h_C) \tag{6}$$

Where Q represents condensation heat transfer, K denotes the total heat transfer coefficient , A is the condensation heat transfer area, ΔT_m is the logarithmic mean temperature difference, $q_{m,W}$ is the cooling water flow rate, $q_{m,G}$ is the exhaust steam flow rate, $h_{W,E}$ is the specific enthalpy of cooling water at inlet , $h_{W,L}$ is the outlet temperature of cooling water, h_G is the specific enthalpy of exhaust steam and h_C is the specific enthalpy of condensate.

DOI 10.3384/ecp223149

The overall heat transfer coefficient K is calculated as:

$$K = 794 \times 4186 / 3600 \beta \sqrt{v_W} \cdot \sqrt[4]{\frac{T_{W,L} + T_{W,E}}{2}} + 17.8 \quad (7)$$

Where β is the fouling coefficient, v_W is the cooling water flow velocity, $T_{W,L}$ is the inlet temperature of cooling water, and $T_{W,E}$ is the outlet temperature of cooling water.

- **Steam regulating value**

Steam control valve is used to control the steam flow rate entering the steam turbine, thereby controlling the output power of the steam turbine. Assuming that the steam flows adiabatically and isoentropically in the valve, only the flow rate produced by the pressure difference between the inlet and outlet is considered. The momentum equation is:

$$q_{m_E} = k_{mx} k_v \left(1 - 0.367 \frac{X}{X_{cr}}\right) \sqrt{\rho_E p_E X} \quad (8)$$

$$X_{cr} = 1 - \left(\frac{2}{K+1}\right)^{\frac{K}{K-1}} \quad (9)$$

$$k_{mx} = \sqrt{\frac{2KA_{max}^2}{K+1}} \quad (10)$$

Where q_{m_E} is the steam mass flow, ρ_E is the steam density, p_E is the inlet steam pressure, X is the valve pressure loss ratio, X_{cr} is the critical pressure loss ratio, K is the heat capacity ratio, k_{mx} is the valve characteristic correction coefficient, and A_{max} is the maximum opening flow area of the valve.

3.2 System Model Construction

Based on the general model library of steam power systems, equipment, subsystem, and system models are built from bottom to top according to the system design architecture. Figure 4 shows the sub-system simulation model of the steam power system.

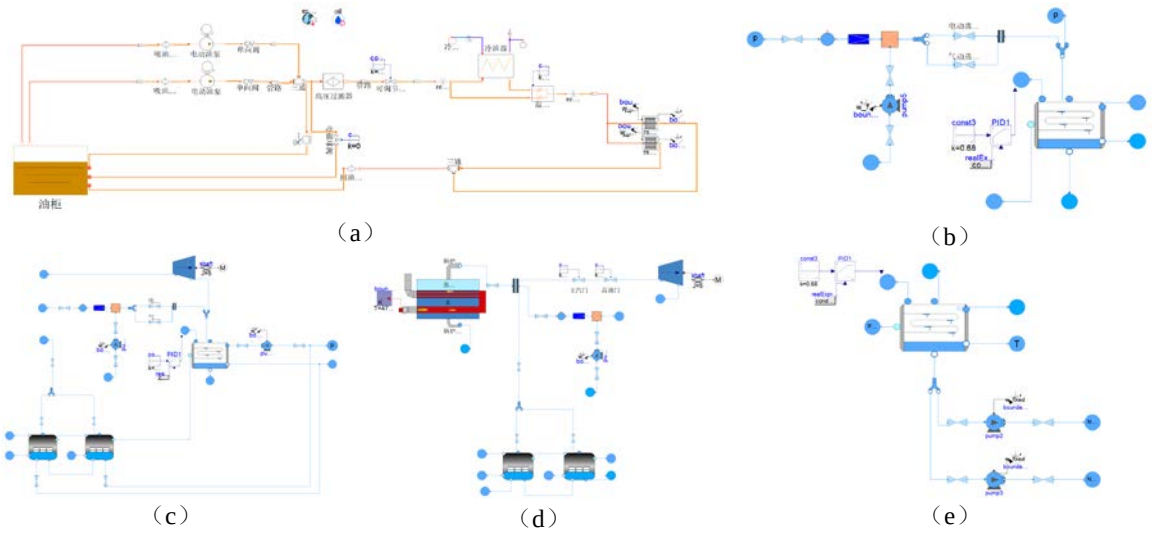


Figure 4. Simulation models of steam power system subsystems: (a) Oil System, (b) Steam-Seal System, (c) Exhaust-Steam System, (d) Steam System, (e) Condensate System.

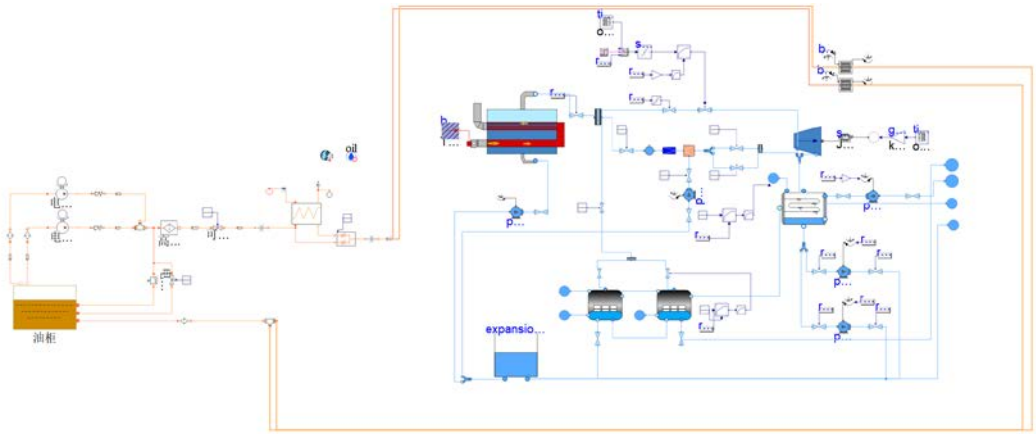


Figure 5. Simulation model of marine steam power system.

After completing the construction of subsystem simulation models, they are connected according to the interface relationships between the subsystems of the steam power system based on its logical architecture. Finally, the steam power system simulation model is obtained (shown in Figure 5). The steam power system consists of a steam system, condensate system, low-pressure steam system, cooling water system, seal system and lubrication oil system. Its main equipment includes steam turbine, boiler feed pump, condenser, condensate pump, ejector, deaerator, lubricating oil cooler, lubricating oil pump, valves, and pipeline accessories. The control logic in the system includes condenser water level control and lubrication oil temperature control.

4 Application Verification and Results Analysis

As a vital subsystem of large vessels, the steam power system comprises over ten subsystems and thousands of equipment units. It converts thermal energy into mechanical and electrical power. Its operational reliability directly affects the safe and reliable operation of the ship. During navigation, the vessel is subjected to high salt spray and variable sea conditions, resulting in a harsh operating environment that increases the load on the power system and imposes higher demands on its safe and stable operation. In this section, the system simulation methodology based on data, models, and mock-ups proposed in this paper is applied to the simulation analysis of a large marine steam power system, providing support for its safe and reliable operation.

4.1 System Simulation Architecture Design

The establishment of a multi-source data-driven full-process simulation system is the prerequisite for application verification of the proposed methodology. Accordingly, a layered simulation architecture for the large marine steam power system under unified data management is designed (see Figure 6). Within this architecture, a unified database serves as the hub for data storage and interaction, the MWORKS.Sysplorer platform acts as the core for model computation and virtual-physical interaction, and the physical test platform together with the operation and maintenance interface is deeply integrated to enable coordinated data links, full-element integration of models and mock-ups, and dynamic virtual-physical evolution.

The architecture comprises four layers. The data acquisition and storage layer employs MySQL for multi-source heterogeneous data (design parameters, bench test data, field monitoring data, fault records). The data processing and business logic layer, developed in Python, provides APIs for health assessment, fault prediction, and model calibration. The application and interaction layer includes fault diagnosis/prediction software and an operation and maintenance interface. The virtual-physical interaction and simulation platform layer, centered on MWORKS.Sysplorer, hosts digital mock-ups and establishes a closed-loop mechanism of data-driven, model computation, mock-up verification, and dynamic updating.

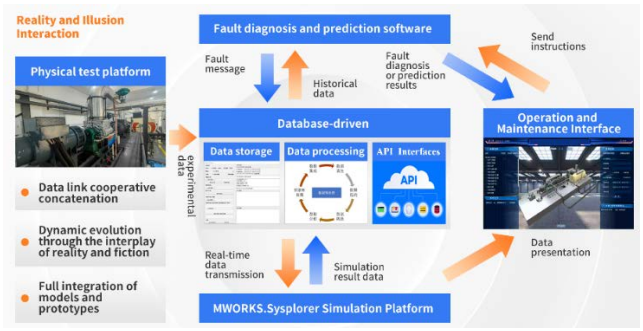


Figure 6. Unified Data Management Architecture for Power System.

The unified data management architecture overcomes limitations of traditional simulation, including data isolation, model–mock-up disconnection, and fragmented workflows. It enables simulation analysis to no longer rely on a single data source or isolated models; instead, multi-dimensional data can be integrated to form a closed-loop simulation system. Driven by the synergy of multiple modules, multi-domain simulation computation enables digital twin applications at multiple levels, including simulation analysis, condition assessment, and diagnostic prediction.

4.2 System Simulation Results Analysis

To validate the fidelity and accuracy of the developed simulation model for large-scale marine steam power systems, comparative verification experiments were conducted on a physical test platform, with simulation results benchmarked against measured test data. This ensures the model accurately captures the operational characteristics of the real system, thereby providing reliable support for full-lifecycle simulation and subsequent engineering applications.

The physical test platform used in this study consists of a real steam turbine generator set, supporting auxiliary systems (fuel, feedwater, condenser, lubricating oil), and a data acquisition and control system. This platform is connected to the simulation model via standard OPC UA and TCP/IP interfaces, enabling simultaneous operation of the physical plant and the digital mock-up. The digital twin is realized by feeding real-time sensor data (pressure, temperature, speed, power) into the simulation model for online calibration and state estimation, while the simulation model provides reference predictions and virtual scenarios to guide test procedures. Figure 6 illustrates the overall architecture of this data-driven closed-loop system, where the physical test platform and the simulation model exchange data through a unified

DOI
10.3384/ecp223149

database and middleware, forming the core of the virtual-physical interaction.

The experimental procedure strictly adheres to the actual operational logic of the system. First, core subsystems of the test platform, including fuel supply, feedwater, steam generation, circulating water, lubricating oil, and the steam turbine, were sequentially activated. Concurrently, the digital mock-up and simulation model were initialized. Standardized interfaces were employed to enable real-time data interaction between the physical platform and the simulation model, ensuring state synchronization and eliminating the influence of timing offsets on verification results.

To comprehensively assess the model's adaptability and accuracy across diverse operating conditions, four representative scenarios were defined: shutdown, 0 kW steady-state, 30 kW steady-state, and 60 kW steady-state operations, covering startup, shutdown, and variable-load regimes. Following the readiness check of all subsystems, the turbine speed was gradually increased to stabilize at 3000 r/min, corresponding to the 0 kW steady-state condition. First- and second-stage loads were then incrementally applied to achieve the 30 kW and 60 kW steady-state conditions. During each steady-state phase (duration$\geqslant$30 min), key system parameters were acquired via sensors, while corresponding simulation outputs were synchronously logged to ensure temporal consistency between measured and simulated data.

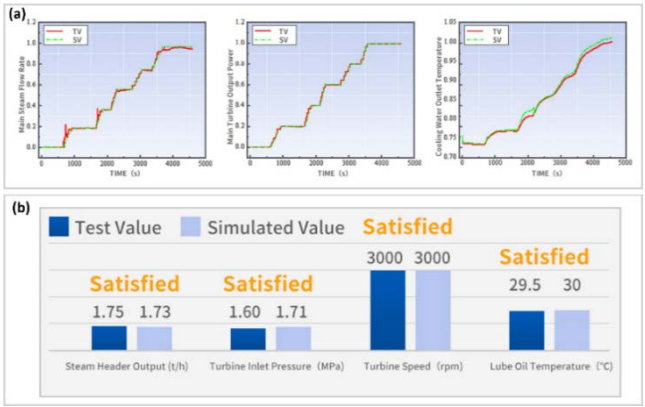


Figure 7. Comparison between measured and simulated key parameters.

Figure 7 presents comparisons between measured and simulated values of critical parameters (e.g., steam pressure, turbine speed, and unit power output) under steady-state and condition-switching processes. The results show that, across all four conditions, the trends of core parameters are highly consistent between measured and simulated data, with no significant discrepancies. Moreover, relative errors of all parameters are controlled within 8%, satisfying the accuracy requirements for real-time simulation and engineering deployment of complex systems. These findings demonstrate that the proposed data-model-mockup integrated simulation methodology enables accurate reproduction of the operational behavior of the target marine steam power system, thereby validating the effectiveness and engineering applicability of the developed model and method.

4.3 Digital Twin-Oriented Application

Based on the simulation architecture and high-fidelity simulation model established previously, and leveraging the advantages of full-life-cycle multi-source data driving, this paper further expands engineering applications oriented toward the digital twin. An intelligent operation and maintenance visualization platform for the large marine steam power system based on data-knowledge fusion is constructed (with the interface shown in Figure 8). This extends the simulation method from "model verification" to "engineering operation and maintenance," fully realizing the engineering value of the proposed system simulation methodology and providing technical support for the full-life-cycle intelligent operation and maintenance of the marine steam power system.

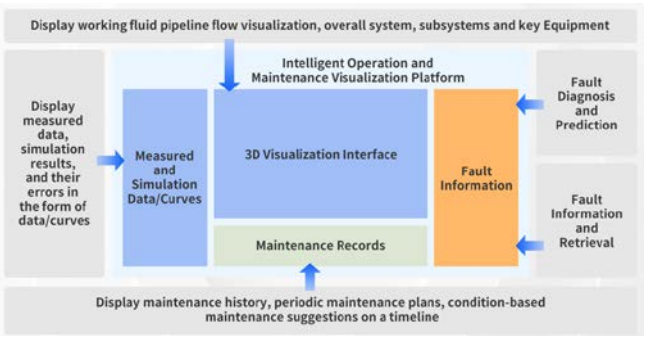


Figure 8. Intelligent Operation and Maintenance Platform Interface.

This intelligent operation and maintenance visualization platform is built upon the proposed data-model-mock-up triadic collaborative simulation framework. It integrates full-life-cycle multi-source data, maintenance archives, and domain knowledge of the large marine steam power system, achieving deep convergence of simulation models, physical test data, and intelligent operation and maintenance management, thus establishing a full-process predictive operation and maintenance system covering fault prediction, diagnosis, maintenance, and optimization. The platform features four key technical advantages: full-life-cycle multi-source data fusion forming a data-knowledge dual-driven framework, real-time dynamic linkage via standardized APIs, visual intelligent decision support, and predictive proactive operation and maintenance enabling early fault warning and intelligent maintenance planning. Based on these, the platform realizes four core engineering functions: real-time system monitoring with simulation-measurement comparison, intelligent fault diagnosis and remaining useful life prediction, unified archiving of full-life-cycle operation

and maintenance data, and data-driven preventive maintenance decision optimization.

In summary, the developed platform serves as the engineering application carrier of the proposed methodology, verifying its applicability in digital twin scenarios, overcoming limitations of traditional operation and maintenance modes, and offering a universal technical solution for full-life-cycle intelligent management of complex marine power systems.

5 Conclusion

This paper proposes a system simulation methodology based on data, models, and mock-ups to support system-level, full-lifecycle simulation and verification of complex equipment. Taking a large marine steam power system as an example, a high-fidelity simulation model is built using Modelica, and validation under multiple operating conditions confirms that all relative errors between simulated and measured key parameters are within 8%, demonstrating the effectiveness and engineering applicability of the method.

On this basis, a digital twin-oriented intelligent operation and maintenance visualization platform is built, realizing core functions such as real-time monitoring, fault diagnosis and prediction, maintenance record management, and planned maintenance decision-making, thereby promoting the transformation from corrective maintenance to predictive maintenance.

The proposed methodology offers a standardizable and generalizable technical framework for full-lifecycle state evolution analysis, high-fidelity simulation, and intelligent operation and maintenance of complex equipment, providing significant reference value for the digital transformation of systems engineering. Future work will focus on multi-physics coupling acceleration algorithms, online adaptive model update strategies, and a generalized framework applicable to different equipment types.

References

Lee S J, Kwak S K, Kim S H, et al. Test platform development of vessel's power management system using hardware-in-the-loop simulation technique[J]. Journal of Electrical Engineering & Technology, 2017, 12(6): 2298-2306.

Zhao C C, Guo J, Xu K W, et al. Modeling and simulation verification of AUV propulsion system based on AMESim[J]. Journal of Unmanned Undersea Systems, 2021, 29(1): 97-103.

Perabo F, Karbalaye Zadeh M. Modelling of a shipboard electric power system for hardware-in-the-loop testing[C]//2020 IEEE Transportation Electrification Conference & Expo. IEEE, 2020: 69-74.

Fritzson P. Modelica—A cyber-physical modeling language and the OpenModelica environment[C]//2011 7th International Wireless Communications and Mobile Computing Conference. IEEE, 2011: 1648-1653.

Qiu K, Yang J, Gao Z, et al. A review of Modelica language in building and energy: Development, applications, and future prospect[J]. Energy and Buildings, 2024, 308: 113998.

Gai J, Fang J, Du X, et al. A novel Modelica-based reliability modeling approach for ship electric propulsion systems[J]. Measurement Science and Technology, 2025, 36(1): 016043.

Guo M M, Hao Y, Nižetić S, et al. Modelica-based heating surface modelling and dynamic exergy analysis of 300 MW power plant boiler[J]. Energy Conversion and Management, 2024, 312: 118557.

Casella F, Leva A. Modelica open library for power plant simulation: design and experimental validation[C]//Proceeding of the 2003 Modelica conference, Linkoping, Sweden. 2003.

Elsheikh A, Awais M U, Widl E, et al. Modelica-enabled rapid prototyping of cyber-physical energy systems via the functional mockup interface[C]//2013 workshop on modeling and simulation of cyber-physical energy systems (MSCPES). IEEE, 2013: 1-6.

Anbarasu S. Multi-Domain Modeling, Simulation and Optimization of Integrated Energy Systems[M]. The Pennsylvania State University, 2025.

Zeng G Q, Ma G C, Zhang L, et al. Virtual real mapping method of marine condensate feed water system based on multi-domain modeling[J]. Chinese Journal of Ship Research, 2022, 17(S1): 81-91, 98.

Zeng G, Wang J, Zhang L, et al. Multi-domain modeling and analysis of marine steam power system based on digital twin[J]. Journal of Marine Science and Engineering, 2023, 11(2): 429.

Wang J, Zeng G, Xie X, et al. Simulation and characterization of wet-steam turbine system for power generation based on Modelica[J]. Ship Science and Technology, 2025, 47(4): 105-111.

Zhang A, He X, Cammi A, et al. Modelica and Arduino-based hardware-in-the-loop simulation of a nuclear-powered engineering ship[J]. Nuclear Engineering and Design, 2024, 429: 113650.

DOI 10.3384/ecp223149

An FMI-Based Multiscale Workflow for Multi-Scenario Local Accident Assessment in LBE Reactor Rod Bundles

Xinheng Zhao[1] Guangliang Chen*[1] Minjun Peng[1] Xiang Zhang[1]

[1]Harbin Engineering University, Harbin, Heilongjiang 150001, China

Abstract

System-level thermal-hydraulic models provide bulk quantities but cannot directly resolve rod-wise temperatures or hotspot migration during local accidents. This study develops an FMI-based multiscale workflow that couples a Modelica primary-loop FMU with an OpenFOAM-derived reduced-order model of a 3×3 lead-bismuth eutectic rod bundle through a Python driver. Four scenarios are evaluated: baseline operation, single-rod local overpower, and local overpower combined with mild or moderate loss of flow. The coupled model predicts bundle-average outlet temperature, nine rod temperatures, hotspot temperature, and limiting-rod location. The hotspot-to-outlet temperature difference increases from 0.803 K in the baseline case to 18.274 K under local overpower with moderate flow reduction, while the hotspot migrates to the south-east rod. OpenFOAM calculations at 15 s and 40 s identify the same limiting rod. All four 60 s scenarios require 0.319 s of online execution time, demonstrating faster-than-real-time local accident assessment.

Keywords: FMI, Modelica, reduced-order model, local accident assessment

1 Introduction

Thermal-hydraulic processes in nuclear reactors exhibit pronounced multiscale characteristics, spanning from system-level transport in the primary and secondary circuits to subchannel flow, rod-surface heat transfer, and local temperature distributions within fuel assemblies. System thermal-hydraulic codes have long served as essential tools for reactor design and safety analysis because of their high computational efficiency, numerical robustness, and suitability for long-duration transient simulations. However, these models generally rely on one-dimensional or lumped-parameter formulations and primarily provide bulk quantities such as mass flow rate, pressure drop, and average temperature. As a result, they have limited capability to resolve spatial phenomena such as subchannel mixing, rod-to-rod temperature differences, and local hotspot migration within fuel assemblies. Computational fluid dynamics (CFD) can provide high-resolution three-dimensional flow and temperature fields, but full transient CFD simulations of complex rod-bundle geometries remain computationally expensive. Consequently, multiscale coupling among system, subchannel, and CFD codes has become an important direction in nuclear thermal-hydraulic research by combining the complementary strengths of models at different scales (D'Auria et al. 2021; Sanchez-Espinoza et al. 2023; Sun et al. 2021).

Accurate prediction of local temperature fields is particularly important for lead-bismuth eutectic (LBE)-cooled fuel assemblies. LBE has a low Prandtl number and strong molecular thermal diffusivity, resulting in a marked difference between turbulent momentum transport and turbulent heat transport. Heat-transfer models tailored to liquid-metal characteristics are therefore required. Recent studies have developed OpenFOAM-based numerical methods for heat-transfer analysis in LBE rod bundles and investigated local temperature variations on fuel-rod surfaces and within individual subchannels of wire-wrapped bundles (Wu et al. 2024). In addition to coolant properties and bundle geometry, rod-wise power distribution also has a strong influence on the internal thermal-hydraulic behavior of a fuel assembly. Studies of non-uniform rod-power distributions have shown that high-power rods alter the temperature, flow, and heat-transfer characteristics of adjacent subchannels and significantly affect peak cladding temperature and temperature uniformity within the assembly. These effects become more pronounced under reduced-flow or natural-circulation conditions (Luo et al. 2025). Therefore, strongly localized disturbances, such as a single-rod local overpower event, may produce substantial rod-wise temperature increases and hotspot migration that cannot be fully represented by the bundle-average outlet temperature.

To describe both system dynamics and local spatial phenomena, coupling between system thermal-hydraulic codes and CFD solvers has been widely investigated. Previous studies have shown that multiscale coupling can preserve system-level transient capability while improving the spatial resolution of local flow, temperature distributions, and accident responses (Sanchez-Espinoza et al. 2023; Sun et al. 2021). In such frameworks, the system model describes loop-scale mass, momentum, and energy transport, whereas the CFD model resolves three-dimensional thermal-hydraulic behavior within selected components. This approach provides an important basis for evaluating local reactor safety parameters and has promoted coordinated simulation across different spatial and

*Corresponding author. Email: chenguangliang@hrbeu.edu.cn

temporal scales.

With the growing demand for online monitoring, multi-scenario evaluation, and digital-twin applications, direct online CFD calculations are often unable to satisfy rapid prediction requirements. Reduced-order and surrogate models can extract the dominant spatial features from high-fidelity CFD data and reconstruct local flow or temperature fields at much lower online computational cost. Proper orthogonal decomposition (POD)-based methods generally identify dominant spatial modes from CFD snapshots and then use regression models or neural networks to predict the corresponding modal coefficients under different operating conditions. POD combined with radial-basis-function neural networks has been applied to rapid flow-field prediction in fuel-rod bundles, achieving substantial improvements in computational efficiency while retaining good spatial prediction accuracy (Min et al. 2024). Reduced-order modeling has also been applied to nuclear thermal-hydraulic transient and accident analysis, providing efficient tools for multi-query calculations, design optimization, and uncertainty analysis (Bang et al. 2025). These studies demonstrate that CFD-derived reduced-order models provide an effective means of rapidly reconstructing internal thermal-hydraulic states in rod bundles.

Standardized integration among heterogeneous modeling tools is another key requirement for multiscale co-simulation. The Functional Mock-up Interface (FMI) encapsulates models as Functional Mock-up Units (FMUs) and provides a unified interface for model exchange and co-simulation across different software environments and solvers. Recent studies have applied FMI to multiscale and multi-fidelity nuclear-reactor simulations, enabling modular coupling among reactor-physics, fuel-performance, and multiphysics solvers and demonstrating its applicability to fast transient processes (Guilbaud et al. 2026). An FMI-based framework has also been used for real-time reconstruction of rod-bundle hydraulic fields by coupling a Modelica system model with POD-based surrogates through a Python execution layer (Zhao et al. 2026). However, its applicability to rod-wise temperature prediction and accident-induced hotspot migration has not been assessed.

To address this gap, a Sysplorer/Modelica primary-loop model is exported as a co-simulation FMU and coupled, through a Python driver, with a CFD-derived reduced-order model of a 3×3 LBE rod bundle. The system model supplies time-dependent mass flow rate, inlet temperature, pressure, and total power, while the reduced-order model predicts the bundle-average outlet temperature, rod-wise temperatures, hotspot-to-outlet temperature difference, limiting-rod location, and outlet temperature field. Four scenarios are considered: a baseline case, a single-rod local overpower case, and two cases combining local overpower with different degrees of flow reduction. Model predictions are assessed using independent steady-state cases and OpenFOAM calculations at 15 s and 40 s during the combined local-overpower and moderate-loss-of-flow transient.

2 Methodology

2.1 Multiscale Co-Simulation Framework

A multiscale computational framework consisting of a system-level thermal-hydraulic model, a local rod-bundle reduced-order model, and a Python-based co-simulation driver was developed, as shown in Fig. 1. The system-level model calculates the primary-loop transient and provides the mass flow rate, core inlet temperature, inlet pressure, and total power. Based on these transient boundary conditions, the local rod-bundle model predicts the bundle-average outlet temperature, the temperatures of the nine rods, the hotspot location, and the outlet temperature field.

The system-level model is encapsulated as a Functional Mock-up Unit (FMU) using the Functional Mock-up Interface (FMI). The Python driver performs FMU instantiation, time advancement, accident-parameter updates, variable exchange, and reduced-order-model evaluation. The system-scale transient calculation and rod-bundle-scale temperature prediction are thereby executed on a common time axis, enabling coordinated analysis of loop dynamics and local thermal-hydraulic responses.

2.2 System-Level Thermal-Hydraulic Model

The primary-loop system is developed in Modelica and exported as a co-simulation FMU. The model describes the mass, momentum, and energy transport processes in the loop and includes the primary circulation pump, core channel, heat-transfer components, and connecting pipes. The FMU outputs the primary-loop mass flow rate, core inlet temperature, inlet pressure, and total power to the co-simulation driver.

Under nominal operation, the primary circulation pump remains at its rated speed. Under loss-of-flow conditions, the pump speed decreases according to a prescribed accident profile, and the system model calculates the resulting transient variations in the loop mass flow rate, inlet temperature, pressure, and other system variables. The mass flow rate, inlet temperature, and total power are supplied to the local rod-bundle reduced-order model, while the inlet pressure is retained as a system-level state variable.

2.3 Local Rod-Bundle Model and Definition of Quantities

The local thermal-hydraulic region is represented by a 3×3 LBE rod bundle with a heated length of 1.0 m, a rod diameter of 10 mm, a rod pitch of 14 mm, and a square duct width of 54 mm. The nine rod positions are labeled as NW–N–NE, W–C–E, and SW–S–SE.

The local overpower disturbance is imposed on the south-east rod, denoted SE. The total bundle power is maintained during the disturbance. The local overpower therefore represents a redistribution of rod-wise power rather than an overall increase in bundle power.

DOI
10.3384/ecp223157

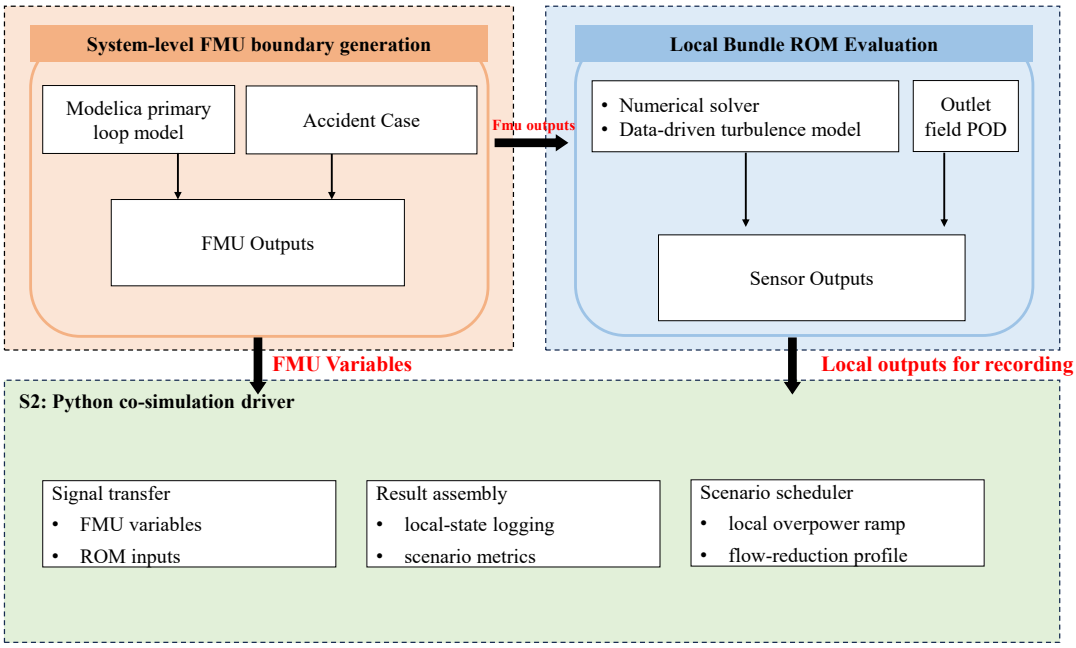


Figure 1. FMI-based multiscale co-simulation framework. The system-level FMU provides transient boundary conditions, the local reduced-order model predicts the bundle-average outlet temperature, rod-wise temperatures, hotspot location, and outlet temperature field, and the Python driver coordinates model execution and data exchange.

The temperature associated with rod i at a given time is denoted by $T_{\mathrm{rod},i}$. The hotspot temperature is defined as the maximum temperature among the nine rods:

$$T_{\mathrm{hotspot}} = \max_{i=1,\ldots,9} T_{\mathrm{rod},i}.$$

The rod corresponding to this maximum is defined as the limiting rod at the current time. The temperature difference between the hottest and second-hottest rods is also calculated to distinguish a clear hotspot location from a near-tied rod-temperature state.

The outlet temperature T_{out} is defined as the area-averaged temperature over the bundle outlet plane:

$$T_{\mathrm{out}} = \frac{\int_{A_{\mathrm{out}}} T(\boldsymbol{x})\,\mathrm{d}A}{\int_{A_{\mathrm{out}}} \mathrm{d}A}.$$

In addition to the scalar outlet-average temperature, the local model reconstructs the outlet temperature distribution to characterize the spatial temperature non-uniformity at the bundle exit.

2.4 Reduced-Order Model Construction

The local rod-bundle response is represented by a reduced-order model constructed from OpenFOAM simulation data. The model inputs are the mass flow rate, inlet temperature, total bundle power, and local overpower factor. The predicted outputs include the bundle-average outlet temperature, the temperatures of the nine rods, and the POD modal coefficients of the outlet temperature field.

The scalar thermal-hydraulic responses are predicted using ridge regression. The original input variables are expanded into a 19-term feature set containing linear, higher-order, and interaction terms to represent the nonlinear relationships among mass flow rate, inlet temperature, total power, and local power distribution. The ridge-regression coefficients are determined by minimizing

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left[\|\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}\|_2^2 + \lambda \|\boldsymbol{\beta}\|_2^2 \right],$$

where $\boldsymbol{X}$ is the feature matrix, $\boldsymbol{y}$ is the target response, $\boldsymbol{\beta}$ is the regression coefficient vector, and λ is the regularization parameter.

A local residual-correction model is applied in addition to the global ridge-regression model. For a given input vector $\boldsymbol{\mu}$, the corrected prediction is expressed as

$$\hat{y}(\boldsymbol{\mu}) = \hat{y}_{\mathrm{global}}(\boldsymbol{\mu}) + \hat{r}_{\mathrm{local}}(\boldsymbol{\mu}),$$

where $\hat{y}_{\mathrm{global}}$ is the global ridge-regression prediction and $\hat{r}_{\mathrm{local}}$ is the residual correction obtained from neighboring training samples.

To preserve physical consistency, the model separately predicts the outlet temperature rise above the inlet temperature and the rod superheats above the bundle-average outlet temperature. The outlet temperature rise is defined as

$$\Delta T_{\mathrm{out}} = T_{\mathrm{out}} - T_{\mathrm{in}} \geq 0,$$

and the superheat of rod i relative to the outlet-average temperature is defined as

$$\Delta T_{\mathrm{rod},i} = T_{\mathrm{rod},i} - T_{\mathrm{out}} \geq 0.$$

The outlet and rod-wise temperatures are reconstructed as

$$T_{\mathrm{out}} = T_{\mathrm{in}} + \Delta T_{\mathrm{out}},$$

$$T_{\mathrm{rod},i} = T_{\mathrm{out}} + \Delta T_{\mathrm{rod},i}.$$

The hotspot temperature is subsequently determined from the maximum of the nine rod temperatures. This reconstruction maintains

$$T_{\mathrm{hotspot}} \geq T_{\mathrm{out}} \geq T_{\mathrm{in}}.$$

Proper orthogonal decomposition (POD) is used to reduce the dimensionality of the outlet temperature field (Sirovich 1987). For outlet-temperature snapshots obtained under different operating parameters $\boldsymbol{\mu}$, the field is expressed as

$$T(\boldsymbol{x};\boldsymbol{\mu}) = \overline{T}(\boldsymbol{x}) + \sum_{k=1}^{N_r} a_k(\boldsymbol{\mu})\phi_k(\boldsymbol{x}),$$

where $\boldsymbol{x}$ denotes the spatial coordinate, $\overline{T}(\boldsymbol{x})$ is the mean temperature field, $\phi_k(\boldsymbol{x})$ is the kth POD spatial mode, $a_k(\boldsymbol{\mu})$ is the corresponding modal coefficient, and N_r is the number of retained modes.

During online evaluation, the regression model predicts the modal coefficients from the current operating parameters. The outlet temperature field is then reconstructed through modal superposition. All outlet temperature fields are mapped onto a common set of 2866 spatial sampling points.

2.5 Training Database

The reduced-order model is constructed from 120 steady-state OpenFOAM simulations. The database uses the mass flow rate, inlet temperature, total bundle power, and local overpower factor as input parameters and covers the operating range considered in the accident analyses. Each operating condition contains the bundle-average outlet temperature, the temperatures of the nine rods, and the outlet-plane temperature field.

All samples are post-processed using consistent definitions of the local thermal-hydraulic quantities. The hotspot temperature is calculated as

$$T_{\mathrm{hotspot}} = \max_{i=1,\ldots,9} T_{\mathrm{rod},i}.$$

The bundle-average outlet temperature is obtained by area-averaging the outlet temperature field. The outlet fields are interpolated onto the same spatial sampling points to ensure consistent data dimensions and spatial locations across all operating conditions. The parameter ranges of the training database are listed in Table 1.

Table 1. Parameter ranges of the reduced-order-model training database.

Parameter	Range	Number of levels
Mass flow rate / $\mathrm{kg\,s^{-1}}$	20–30	7
Core inlet temperature / K	570–621	13
Bundle power / kW	7.2–10.0	6
Local overpower factor	1.0–6.6	6

2.6 Model Verification

The reduced-order model is evaluated using a three-level verification strategy.

The first level is holdout verification within the training parameter domain. The database is divided into fitting and holdout subsets, with the holdout samples excluded from model fitting. The evaluated quantities include the bundle-average outlet temperature, hotspot temperature, and hotspot-to-outlet temperature difference. The reported error metrics are the mean absolute error, root-mean-square error, and maximum absolute error. The corresponding results are presented in Table 2.

The second level uses four independent steady-state local-overpower cases that are not included in model training. The reduced-order-model predictions are compared with OpenFOAM results for the bundle-average outlet temperature, rod-wise temperatures, hotspot temperature, hotspot-to-outlet temperature difference, and limiting-rod location. This verification level evaluates the model under local power-redistribution conditions relevant to the accident scenarios considered in this study.

The third level addresses the dynamic response under combined local overpower and flow reduction. Independent OpenFOAM calculations are performed at 15 s and 40 s and compared with the FMU–ROM co-simulation results. The comparison quantities include the bundle-average outlet temperature, the temperatures of the nine rods, the hotspot-to-outlet temperature difference, and the limiting-rod location. This level evaluates the prediction of local thermal responses under time-varying system boundary conditions.

2.7 FMU–ROM Dynamic Coupling

The communication step between the FMU and ROM is set to 1.0 s. Within each communication macro-step, the FMU advances using four 0.25 s substeps. The ROM is evaluated once per macro-step, and its outputs are held constant during the subsequent FMU substeps.

At the beginning of each accident scenario, the ROM is evaluated using the FMU outputs at $t = 0$ s to initialize the outlet-average temperature, rod-wise temperatures, hotspot temperature, and outlet temperature field. The co-simulation subsequently proceeds according to the following sequence:

1. Advance the FMU according to the current pump speed and system state to obtain the mass flow rate, inlet temperature, inlet pressure, and total power;

DOI
10.3384/ecp223157

2. Update the local overpower factor and pump speed according to the prescribed accident timing;
3. Transfer the FMU outputs and local overpower factor to the ROM;
4. Evaluate the ROM to obtain the quasi-steady outlet-average temperature, rod-wise temperatures, hotspot location, and outlet temperature field;
5. Apply a first-order dynamic correction to the ROM-predicted local temperature states;
6. Store the system and local rod-bundle states at the current communication time and proceed to the next macro-step.

A first-order thermal-inertia model is applied to the outlet temperature. The effective LBE volume participating in the local transient heat transfer is approximately $2.21 \times 10^{-3}\,\mathrm{m}^3$. For the quasi-steady outlet temperature T^*_{out} predicted by the ROM, the dynamic outlet temperature satisfies

$$\tau_{\mathrm{out}} \frac{\mathrm{d}T_{\mathrm{out}}}{\mathrm{d}t} + T_{\mathrm{out}} = T^*_{\mathrm{out}},$$

where τ_{out} is the outlet-temperature response time scale determined from the effective LBE mass and heat capacity.

Using an explicit discretization, the outlet temperature is updated as

$$T^{n+1}_{\mathrm{out}} = T^n_{\mathrm{out}} + \alpha_{\mathrm{out}} \left(T^{*,n}_{\mathrm{out}} - T^n_{\mathrm{out}} \right),$$

where

$$\alpha_{\mathrm{out}} = \min\left(\frac{\Delta t}{\tau_{\mathrm{out}}}, 1 \right),$$

and Δt is the communication step between the FMU and ROM.

The hotspot temperature is not advanced as an independent absolute-temperature state. Instead, the hotspot-to-outlet temperature difference is defined as

$$\Delta T_{\mathrm{hot}} = T_{\mathrm{hotspot}} - T_{\mathrm{out}} \geq 0.$$

For the quasi-steady hotspot-to-outlet temperature difference $\Delta T^*_{\mathrm{hot}}$ predicted by the ROM, its dynamic response is expressed as

$$\tau_{\mathrm{hot}} \frac{\mathrm{d}\Delta T_{\mathrm{hot}}}{\mathrm{d}t} + \Delta T_{\mathrm{hot}} = \Delta T^*_{\mathrm{hot}}.$$

The corresponding discrete update is

$$\Delta T^{n+1}_{\mathrm{hot}} = \Delta T^n_{\mathrm{hot}} + \alpha_{\mathrm{hot}} \left(\Delta T^{*,n}_{\mathrm{hot}} - \Delta T^n_{\mathrm{hot}} \right),$$

where

$$\alpha_{\mathrm{hot}} = \min\left(\frac{\Delta t}{\tau_{\mathrm{hot}}}, 1 \right).$$

The hotspot temperature is reconstructed as

$$T^{n+1}_{\mathrm{hotspot}} = T^{n+1}_{\mathrm{out}} + \Delta T^{n+1}_{\mathrm{hot}}.$$

The dynamic reconstruction therefore maintains

$$T_{\mathrm{hotspot}} \geq T_{\mathrm{out}} \geq T_{\mathrm{in}}$$

throughout the transient calculation.

2.8 Accident Scenarios

Four dynamic scenarios are considered to investigate the effects of local power redistribution and its combination with coolant-flow reduction:

- **Baseline case:**

  The primary circulation pump remains at its rated speed, and the nine rods have a uniform power distribution.

- **Local overpower case:**

  The local overpower factor of the south-east rod increases linearly from 1.0 to 5.8 between $t = 10\,\mathrm{s}$ and $t = 12\,\mathrm{s}$, while the total bundle power is maintained.

- **Local overpower with mild loss of flow:**

  The same rod-wise power redistribution is applied, while the normalized pump speed decreases from 1.0 to 0.9. The system model calculates the resulting loop-flow and inlet-state responses.

- **Local overpower with moderate loss of flow:**

  The same local power disturbance is applied, while the normalized pump speed decreases from 1.0 to 0.8, producing a more pronounced loss-of-flow transient.

All four scenarios use the same initial operating condition and are simulated for 60 s. The effects of local power redistribution and coolant-flow reduction are evaluated by comparing the dynamic evolution of the bundle-average outlet temperature, rod-wise temperatures, hotspot temperature, hotspot-to-outlet temperature difference, and limiting-rod location.

3 Results

3.1 Reduced-Order Model Verification

The reduced-order model was first evaluated using hold-out data within the training parameter domain. Table 2 presents the prediction errors for the bundle-average outlet temperature, hotspot temperature, and hotspot-to-outlet temperature difference. The mean absolute errors of the bundle-average outlet temperature and hotspot temperature are 1.428 K and 0.959 K, respectively. The mean absolute error, root-mean-square error, and maximum absolute error of the hotspot-to-outlet temperature difference

are 0.552 K, 0.919 K, and 2.665 K, respectively. These results show that the reduced-order model captures the principal local temperature differences within the training parameter domain.

Table 2. Holdout verification errors within the training parameter domain. All errors are in K.

Quantity	MAE	RMSE	Max. error
Bundle-average outlet temperature	1.428	2.625	8.396
Hotspot temperature	0.959	2.097	8.148
Hotspot-to-outlet difference	0.552	0.919	2.665

Four independent steady-state local-overpower cases were subsequently used to assess the model outside the fitting data used for the corresponding predictions. The verification statistics are listed in Table 3. The mean absolute errors of the bundle-average outlet temperature, hotspot temperature, and hotspot-to-outlet temperature difference are 4.833 K, 2.686 K, and 2.146 K, respectively. The independent cases exhibit larger outlet-temperature errors than the holdout data, while the hotspot-temperature and hotspot-to-outlet temperature-difference errors remain within a few kelvin.

The reduced-order model correctly identifies the limiting rod in three of the four independent cases. The only difference in limiting-rod location occurs at a local overpower factor of 2.2, for which the predicted temperature difference between the hottest and second-hottest rods is only 0.003 K. The position difference in this case is therefore associated with a near-tied rod-temperature state rather than a pronounced discrepancy in the predicted temperature distribution.

Table 3. Verification errors for the independent steady-state local-overpower cases. All errors are in K.

Quantity	MAE	RMSE	Max. error
Bundle-average outlet temperature	4.833	4.833	4.976
Hotspot temperature	2.686	2.813	4.086
Hotspot-to-outlet difference	2.146	2.330	3.008

3.2 Steady-State Temperature Response to Local Overpower

Table 4 presents the steady-state temperature response as the local overpower factor of the south-east rod is varied. The total bundle power is maintained in all cases; therefore, the temperature changes reflect the local effect of rod-wise power redistribution.

As the local overpower factor increases from 2.2 to 5.8, the outlet-average temperature decreases from 621.886 K to 621.632 K, corresponding to a decrease of only 0.254 K. Over the same range, the hotspot temperature increases from 621.964 K to 622.542 K, an increase of 0.578 K, while the hotspot-to-outlet temperature difference increases from 0.078 K to 0.909 K.

With the total bundle power fixed, rod-wise power redistribution produces little change in the outlet-average temperature but increases local temperature non-uniformity. Assigning a larger fraction of the bundle power to the south-east rod raises its local temperature while slightly reducing the outlet-average temperature. The resulting rod-wise thermal response is therefore not represented by the outlet-average temperature alone.

Table 4. Steady-state temperature response to local overpower of the south-east rod. The total bundle power is maintained in all cases, and all temperatures are in K.

Overpower factor	Outlet temperature	Hotspot temperature	$\Delta T_{\mathrm{hot-out}}$
2.2	621.886	621.964	0.078
4.2	621.750	622.407	0.657
5.0	621.692	622.485	0.793
5.8	621.632	622.542	0.909

3.3 Dynamic Response to Local Overpower and Flow Reduction

The principal results of the four dynamic scenarios are summarized in Table 5. The hotspot-temperature histories and hotspot-to-outlet temperature differences are presented side by side in Fig. 2. Because all scenarios share the same pre-accident evolution, the thermal states at the end of the transient are used for comparison.

At the end of the baseline case, the hotspot-to-outlet temperature difference is 0.803 K. After the local overpower disturbance is imposed on the south-east rod, this difference increases to 4.914 K. When the same local overpower is combined with mild and moderate loss of flow, the final hotspot-to-outlet temperature differences increase further to 13.933 K and 18.274 K, respectively. The reduction in coolant flow therefore substantially amplifies the temperature non-uniformity produced by the local power disturbance.

At the same time, the final outlet-average temperature decreases from 618.507 K in the baseline case to 611.167 K, 582.808 K, and 571.676 K in the three accident scenarios. The final hotspot temperature also decreases across these scenarios. The reduction in absolute temperature level is primarily associated with the transient variation of the system-side inlet temperature, whereas the increasing hotspot-to-outlet temperature difference reflects the combined effects of local power concentration and reduced coolant flow.

A decrease in the outlet-average temperature therefore does not necessarily indicate a more uniform internal bundle temperature distribution. Under combined local overpower and flow reduction, the outlet-average temperature and the local hotspot deviation exhibit different trends. The hotspot-to-outlet temperature difference provides a more direct measure of the increasing local temperature non-uniformity than the outlet-average temperature alone.

The local overpower disturbance also changes the location of the limiting rod. At the end of the baseline case, the highest rod temperature occurs at the north rod. After the local overpower disturbance is imposed, the limiting rod

DOI
10.3384/ecp223157

Table 5. Local thermal-hydraulic quantities at the end of the four dynamic scenarios.

Scenario	Final outlet / K	Final hotspot / K	Final $\Delta T_{\mathrm{hot-out}}$ / K	Final limiting rod
Baseline	618.507	619.310	0.803	North
Local overpower	611.167	616.081	4.914	South-East
Local overpower with mild loss of flow	582.808	596.741	13.933	South-East
Local overpower with moderate loss of flow	571.676	589.950	18.274	South-East

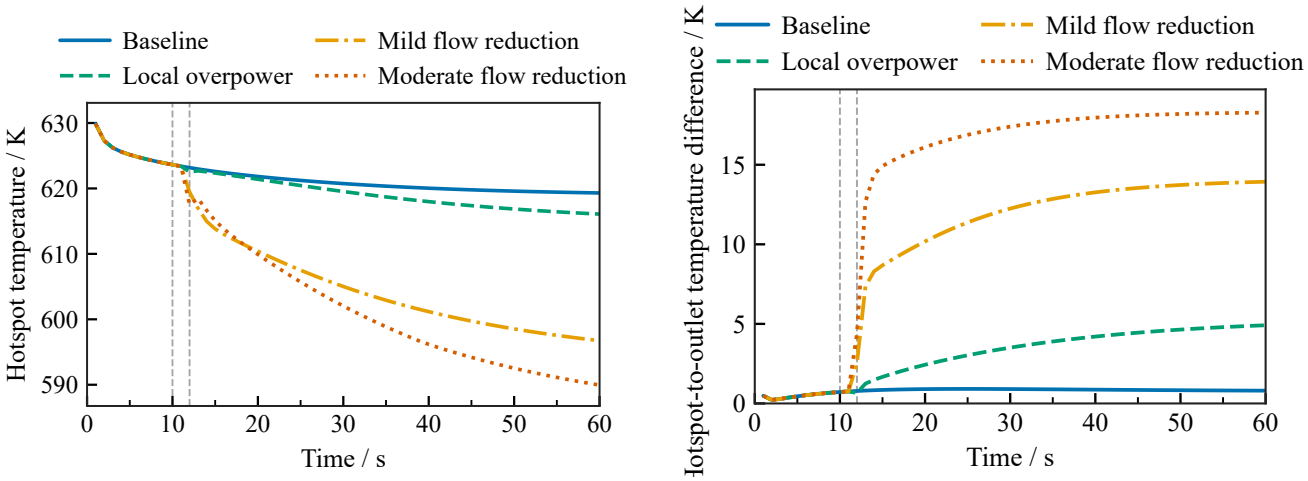


Figure 2. Dynamic thermal responses for the four accident scenarios. Left: hotspot-temperature histories. Right: hotspot-to-outlet temperature differences. The vertical dashed lines at 10 s and 12 s mark the beginning and completion of the prescribed disturbance ramp, respectively.

shifts to the south-east rod. The south-east rod remains the limiting rod when the disturbance is combined with flow reduction, and its temperature deviation from the outlet average increases further.

To compare the rod-wise temperature non-uniformity independently of the absolute temperature level, the temperature deviation of rod i from the nine-rod mean is defined as

$$\delta T_{\mathrm{rod},i} = T_{\mathrm{rod},i} - \overline{T}_{\mathrm{rod}},$$

where

$$\overline{T}_{\mathrm{rod}} = \frac{1}{9}\sum_{i=1}^{9} T_{\mathrm{rod},i}.$$

Figure 3 presents the terminal distributions of $\delta T_{\mathrm{rod},i}$ at 60 s. The baseline case remains nearly uniform, with deviations confined to approximately 0.5 K from the nine-rod mean. Under local overpower, the temperature deviation of the south-east rod reaches 3.4 K. This value increases to 9.5 K and 12.4 K when the local power disturbance is combined with mild and moderate loss of flow, respectively. The increasing positive deviation of the south-east rod, together with the elevated temperatures of the adjacent east and south rods, demonstrates the progressive concentration of the local thermal response near the overpowered position.

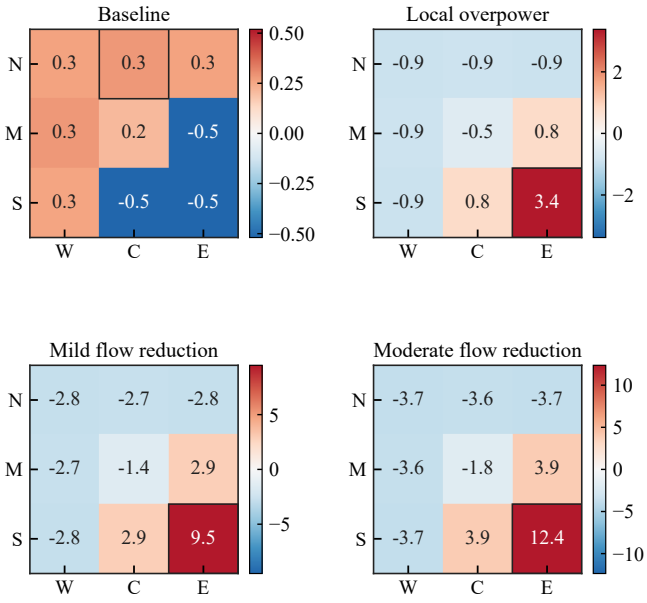


Figure 3. Terminal rod-temperature deviations from the nine-rod mean at 60 s. Positive values denote rods hotter than the nine-rod mean, whereas negative values denote rods colder than the mean. Each panel uses an independently normalized color scale to emphasize the rod-wise temperature non-uniformity within the corresponding scenario. The outlined cell denotes the numerical maximum; the leading rods in the baseline case remain nearly tied.

3.4 Hotspot Migration

To distinguish numerical changes in rod ranking from physically meaningful hotspot migration, Fig. 4 presents the complete temperature histories of the nine rods for the local-overpower case combined with moderate loss of flow. Figure 5 combines the temperature difference between the hottest and second-hottest rods with the time-dependent limiting-rod label.

The first change in hotspot label occurs at approximately 4 s, when the hottest-rod label changes from the north-west rod to the north rod. The temperature difference between these two rods is only 1.26×10^{-5} K, which is substantially smaller than the rod-to-rod temperature differences produced during the accident phase. This event is therefore interpreted as a numerical ranking change under a near-tied temperature state rather than a distinct spatial migration of the hotspot.

The second change occurs at approximately 12 s, corresponding to the completion of the local overpower ramp applied to the south-east rod. At this time, the temperature of the south-east rod exceeds that of the second-hottest rod by 2.18 K. The temperature separation subsequently increases, and the south-east rod remains the hottest rod throughout the remainder of the transient. This behavior indicates a clear migration of the hotspot to the locally overpowered rod.

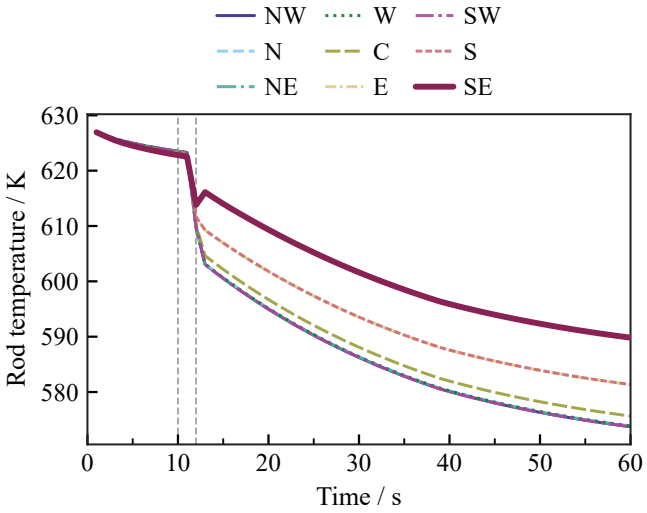


Figure 4. Temperature histories of the nine rods for the local-overpower case combined with moderate loss of flow. The south-east rod is highlighted by the thicker line. The vertical dashed lines at 10 s and 12 s mark the beginning and completion of the prescribed disturbance ramp, respectively.

3.5 OpenFOAM Verification of Hotspot Location and Dynamic Temperature Response

OpenFOAM calculations were performed at 15 s and 40 s to assess the post-disturbance local thermal response predicted by the reduced-order model. The comparison includes the bundle-average outlet temperature, hotspot temperature, hotspot-to-outlet temperature difference, and

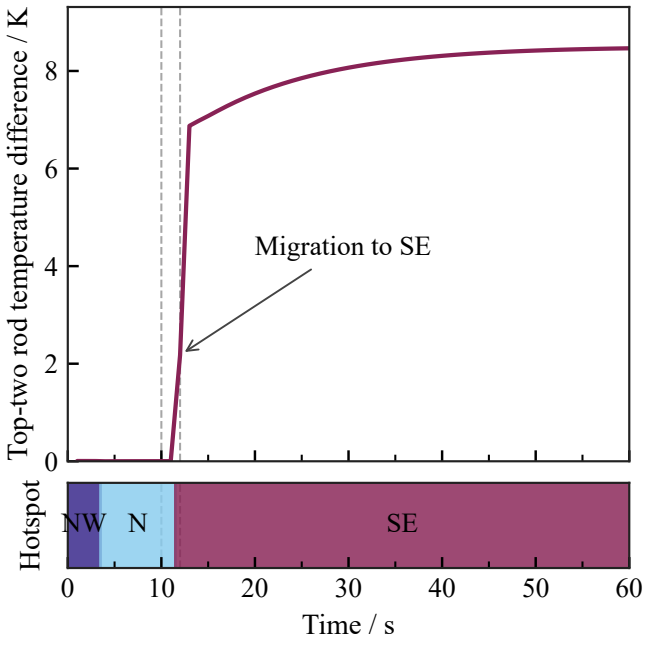


Figure 5. Hotspot migration for the local-overpower case combined with moderate loss of flow. The upper panel shows the temperature difference between the hottest and second-hottest rods, while the lower band indicates the limiting-rod label. The early change from the north-west rod to the north rod occurs under a near-tied state, whereas the transition to the south-east rod after the local-overpower ramp represents a distinct hotspot migration.

limiting-rod location. The quantitative results are summarized in Table 6.

At both verification times, the highest temperature in the OpenFOAM results occurs at the south-east rod, consistent with the limiting-rod location predicted by the reduced-order model. The agreement in limiting-rod location supports the predicted post-disturbance hotspot migration and shows that the south-east rod continues to control the maximum local temperature after the local overpower disturbance.

At 15 s, the reduced-order model underpredicts the hotspot temperature, bundle-average outlet temperature, and hotspot-to-outlet temperature difference by 5.75 K, 3.05 K, and 2.69 K, respectively. The model therefore underestimates the magnitude of the local thermal response during the early post-disturbance stage.

At 40 s, the hotspot-temperature and outlet-temperature errors are −2.72 K and −2.98 K, respectively, while the hotspot-to-outlet temperature-difference error decreases to 0.26 K. As the transient develops, the reduced-order prediction of the local temperature non-uniformity approaches the OpenFOAM result, although the absolute temperature predictions retain a negative bias of approximately 3 K.

The rod-wise ranking and quantitative temperature comparisons show that the reduced-order model identifies the post-disturbance limiting rod and reproduces the evolution of the hotspot deviation from the outlet-average temperature. The absolute temperature level is underestimated dur-

DOI
10.3384/ecp223157

Table 6. OpenFOAM verification for the local-overpower case combined with moderate loss of flow. All temperature quantities and errors are in K.

Time / s	Quantity	ROM	CFD	Error
15	Hotspot temperature	614.01	619.76	-5.75
15	Outlet-average temperature	598.98	602.03	-3.05
15	Hotspot-to-outlet difference	15.03	17.72	-2.69
40	Hotspot temperature	595.90	598.62	-2.72
40	Outlet-average temperature	577.93	580.91	-2.98
40	Hotspot-to-outlet difference	17.97	17.71	+0.26

ing the early stage of the disturbance, while the prediction of the hotspot-to-outlet temperature difference improves during the later transient.

3.6 Computational Performance

On the computational platform used in this study, the four 60 s dynamic scenarios require a total online execution time of 0.319 s. This time includes FMU advancement, accident-parameter updates, reduced-order-model evaluation, and result recording, but excludes the offline generation of the OpenFOAM database and reduced-order-model training.

The FMU–ROM co-simulation framework therefore completes the multi-scenario analysis in substantially less time than the physical duration of the simulated accidents. This computational performance enables faster-than-real-time comparison of local accident scenarios and rapid prediction of internal rod-bundle hotspot states.

4 Conclusions and Outlook

An FMI-based multiscale co-simulation framework was developed for local accident assessment in an LBE-cooled rod bundle. The framework couples a Modelica primary-loop FMU with a CFD-derived reduced-order model and provides bundle-average outlet temperature, rod-wise temperatures, hotspot location, and outlet-field information under time-dependent system boundary conditions.

Local power redistribution and coolant-flow reduction were found to increase rod-bundle temperature non-uniformity without producing a corresponding increase in the bundle-average outlet temperature. Across the investigated dynamic scenarios, the final hotspot-to-outlet temperature difference increased from 0.803 K in the baseline case to 18.274 K under combined local overpower and moderate loss of flow. The limiting location migrated to the locally overpowered south-east rod, and OpenFOAM calculations at two representative times identified the same limiting rod. The comparison also indicated that the main early-transient discrepancy lies in the absolute temperature magnitude rather than in the predicted hotspot location.

The four 60 s scenarios required a total online execution time of 0.319 s. The coupled framework therefore provides faster-than-real-time estimates of internal hotspot states that are not available from system-level outlet quantities alone. Future work will consider blockage, inlet-flow maldistribution, multiple-rod power anomalies, and uncertainty quantification of the reduced-order predictions.

References

Bang, Young Suk, Jaeseok Heo, Ha Neul Na, and Kyoung-Ho Kang (2025). “Reduced order models for thermal hydraulic transient analysis in nuclear engineering”. In: *Annals of Nuclear Energy* 211, p. 111002.

D’Auria, F, N Aksan, H Glaeser, et al. (2021). “Physical Phenomena in Nuclear Thermal Hydraulics and Current Status”. In: *TECNICA ITALIANA* 65.1, pp. 1–11.

Guilbaud, Thomas, Carlo Fiorina, Alessandro Scolaro, and Andreas Pautz (2026). “Investigation of FMI-based multi-fidelity simulation for nuclear cores: Application to ADS beam trips”. In.

Luo, Guanwen et al. (2025). “Thermal-hydraulic analysis of a 19-rod bundle LBE cooled fuel assembly with non-uniform rods power distribution by numerical simulation”. In: *Annals of Nuclear Energy* 215, p. 111242.

Min, Guangyun, Yu Ma, Yahui Wang, and Naibin Jiang (2024). “Flow fields prediction for data-driven model of 5× 5 fuel rod bundles based on POD-RBFNN surrogate model”. In: *Nuclear Engineering and Design* 422, p. 113117.

Sanchez-Espinoza, Victor Hugo, Kanglong Zhang, Alejandro Campos Munoz, and Michael Böttcher (2023). “KIT Multi-scale thermal–hydraulic coupling methods for improved simulation of nuclear power plants”. In: *Nuclear Engineering and Design* 405, p. 112218.

Sirovich, Lawrence (1987). “Turbulence and the Dynamics of Coherent Structures. Part I: Coherent Structures”. In: *Quarterly of Applied Mathematics* 45.3, pp. 561–571.

Sun, Lin, Minjun Peng, Genglei Xia, Xuesong Wang, and Mingyu Wu (2021). “Development and validation of multi-scale coupled thermal-hydraulic code combining RELAP5 and fluent code”. In: *Frontiers in Energy Research* 8, p. 613852.

Wu, Jie, Xingkang Su, Ziqi Gong, Jiejin Cai, and Long Gu (2024). “Numerical heat transfer to LBE flow in a wire-wrapped 19-rod bundle using an OpenFOAM-based solver: fourParameterFoam”. In: *Nuclear Engineering and Design* 418, p. 112917.

Zhao, Xinheng, Minjun Peng, and Guangliang Chen (2026). “A real-time digital twin framework for rod-bundle hydraulics using POD-based surrogates and FMI-compliant system coupling”. In: *Annals of Nuclear Energy* 238, p. 112514.

DOI
10.3384/ecp223157

High-Pressure Liquid–Gas Ejector Models for System Simulation in Modelica

Jinglong Zhou[1] Jie Cheng[1,*] Luchao She[1] Guangming Fan[1] Jianjun Wang[1]

[1]College of Nuclear Science and Technology, Harbin Engineering University, Harbin, China
[*]Corresponding author: jcheng@hrbeu.edu.cn

Abstract

System simulation of high-pressure liquid–gas ejectors requires compact pressure–flow representations. This work compares two reduced-order models of the same fixed-geometry water–methane ejector against a 21-point CFD pressure-response database: an effective-area one-dimensional model and an explicit quadratic surrogate derived from the CFD data. With leave-one-out errors normalized by the global CFD response range, the surrogate gives NRMSEs of 1.19%, 0.80%, and 0.71% for primary flow, secondary flow, and entrainment ratio; the 1D model gives 1.31%, 5.49%, and 5.52%. Both models are implemented in Modelica and exported through FMI 2.0. The surrogate CS FMU matches the native Modelica scheduling trajectory to machine precision. Coupled to a first-order back-pressure actuator, the surrogate supports inverse entrainment-ratio scheduling with settling times of 7.0 and 7.6 s. The two formulations therefore offer complementary choices for reusable system simulation.

Keywords: liquid–gas ejector, Modelica, Functional Mock-up Interface, one-dimensional model, CFD-derived surrogate, system simulation

1 Introduction

A high-pressure motive stream can entrain and compress a lower-pressure stream in an ejector without moving parts. The resulting performance is set by nozzle acceleration, suction, mixing, and pressure recovery. CFD resolves these processes in detail, but a system model may evaluate the component many times, making direct CFD impractical for routine simulation (Aidoun et al. 2019; Koirala et al. 2021; Yadav and Patwardhan 2008). A useful Modelica component must therefore retain the relevant pressure–flow dependence without carrying the cost of the full flow solver.

Two reduced-model families are well suited to that task. One-dimensional (1D) formulations preserve conservation-law structure and use a small set of interpretable closure parameters (Antonio et al. 2012; Aidoun et al. 2019). CFD-derived response models take a different route: they fit the performance map of a fixed geometry and can be evaluated as explicit algebraic relations. CFD-based ejector models have already been used in OpenModelica system studies (Haida, Smolka, et al. 2018; Haida, Fingas, et al. 2019), while Modelica and the Functional Mock-up Interface (FMI) provide a standard route from component equations to reusable simulation artifacts (Blochwitz et al. 2011; Modelica Association Project FMI 2014; Lindqvist et al. 2023).

Here the two approaches are examined on the same high-pressure water–methane geometry and the same CFD pressure-response database. They use identical pressure inputs and target outputs, so differences in primary flow, secondary flow, and entrainment ratio arise from model form rather than from changes in geometry or operating data. Both are implemented behind a common Modelica interface and exported through FMI 2.0. The CFD-derived model is then used on the characterized back-pressure branch in an inverse scheduling example with a first-order actuator. The emphasis is on a clear model hierarchy, a like-for-like assessment, and reusable deployment on a common system interface.

2 Ejector CFD Characterization

2.1 Geometry and Numerical Method

The geometry follows the configuration reported by She (2020): a converging primary nozzle discharges into a suction chamber connected to an oblique secondary inlet, followed by a constant-area mixing chamber and a diffuser. Figure 1 defines the pressure boundaries and principal dimensions, and Table 1 collects the main dimensions and reference operating condition. Water enters through the primary nozzle at P_p, methane through the secondary branch at P_s, and the mixed stream leaves against the back pressure P_c.

The three-dimensional CFD calculation was performed in ANSYS Fluent using a Volume of Fluid (VOF) formulation with the standard k–ε turbulence model. Pressure-driven entrainment and back-pressure sensitivity are well established in gas–liquid ejector CFD and in broader ejector experiments (C. Li and Y.-Z. Li 2011; Sriveerakul et al. 2007; Wang et al. 2017). A structured hexahedral grid with 327 709 cells was selected from a seven-grid study covering 147 921–632 790 cells. Beyond the selected grid, the reported primary flow, secondary flow, and entrainment ratio no longer change at the precision of the mesh-sensitivity data. The main numerical settings are listed in Table 2. For the source CFD formulation, separate benchmark cases reported entrainment-ratio deviations of 6.25–12.35% and

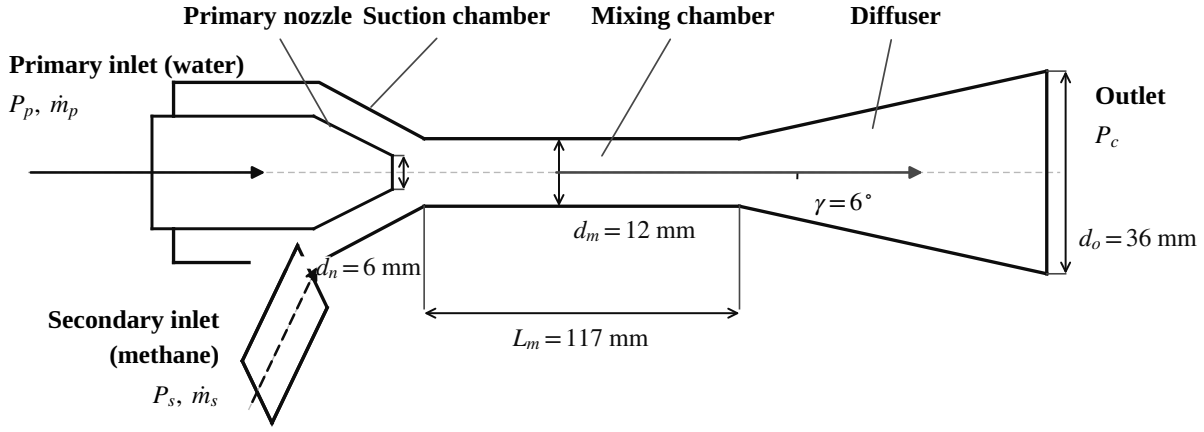


Figure 1. Schematic of the high-pressure liquid–gas ejector showing the principal dimensions and pressure boundary variables; dimensions are not drawn to scale.

Table 1. Main geometry and reference operating condition.

Quantity	Value
Nozzle outlet diameter, d_n	6 mm
Secondary inlet diameter, d_2	25 mm
Nozzle-exit position, L_{NXP}	11.8 mm
Mixing chamber, $d_m \times L_m$	12×117 mm
Diffuser, $d_o \times L_d$	36×114 mm
Diffuser angle, γ	6°
Working fluids	water / methane
Reference pressures, $P_p/P_s/P_c$	20/12/14 MPa
Temperature	298.15 K

Table 2. CFD formulation and numerical settings.

Item	Setting / criterion
Multiphase model	VOF
Turbulence model	standard k–ε
Mesh topology	structured hexahedral
Selected grid	327 709 cells
Grid study	147 921–632 790 cells
General residuals	$< 10^{-3}$
Energy residual	$< 10^{-6}$
Mass imbalance	$< 0.5\%$
Walls	no-slip, adiabatic

critical-back-pressure deviations of 9.09–9.91%, providing a method-level check independent of the reduced-model assessment (She 2020).

2.2 Reference Flow and Operating Data

At the reference condition, the primary jet accelerates through the nozzle and produces a low-pressure, high-velocity region just downstream of the exit. Figure 2 shows the corresponding pressure, velocity, and methane-volume-fraction contours adapted from She (2020). The minimum static pressure is 11.98 MPa and the maximum velocity magnitude is 128.09 m s^{-1}, both about 7.12 mm downstream of the nozzle exit; at the exit itself, the values are 15.36 MPa and 109.38 m s^{-1}. The pressure depression draws methane into the suction chamber, mixing develops downstream, and the diffuser recovers static pressure. The reduced models do not reproduce these local fields. Instead, they retain the pressure–flow response that results when the inlet and outlet pressures change.

The pressure-response database contains 21 unique CFD operating points arranged as controlled sweeps about a common reference state. The ranges are 17.5–30 MPa in P_p, 12–18 MPa in P_s, and 0–14.5 MPa in P_c. At the reference condition, $\dot{m}_{p,0} = 3.1751$ kg s^{-1}, $\dot{m}_{s,0} = 0.2867$ kg s^{-1}, and $\mu_0 = 0.09030$. The entrainment ratio is

$$\mu = \frac{\dot{m}_s}{\dot{m}_p}. \tag{1}$$

Additional CFD states on the reverse-flow side of the operating boundary are retained as regime markers for the status output of the Modelica component.

3 Reduced Ejector Models

3.1 Effective-Area One-Dimensional Model

The one-dimensional model keeps a control-volume description of the two inlet streams. Effective-area and loss coefficients account for contraction, mixing, and diffuser effects that are not resolved explicitly. With P_e denoting an effective pressure at the mixing entry, the inlet relations are

$$\begin{aligned} u_q &= \sqrt{\frac{2(P_q - P_e)}{(1+K_q)\rho_q}}, \\ \dot{m}_q &= C_{A,q} A_q \rho_q u_q, \qquad q \in \{p, s\}, \end{aligned} \tag{2}$$

where A_q, ρ_q, and $C_{A,q}$ denote inlet area, density, and effective-area coefficient, respectively. The branch loss term is $K_q = K_n$ for $q = p$ and $K_q = K_s$ for $q = s$. Pressures in Eqs. (2) and (3) are evaluated in consistent SI units; pressure increments in the surrogate model below are expressed in MPa to match the fitted coefficients.

DOI
10.3384/ecp223167

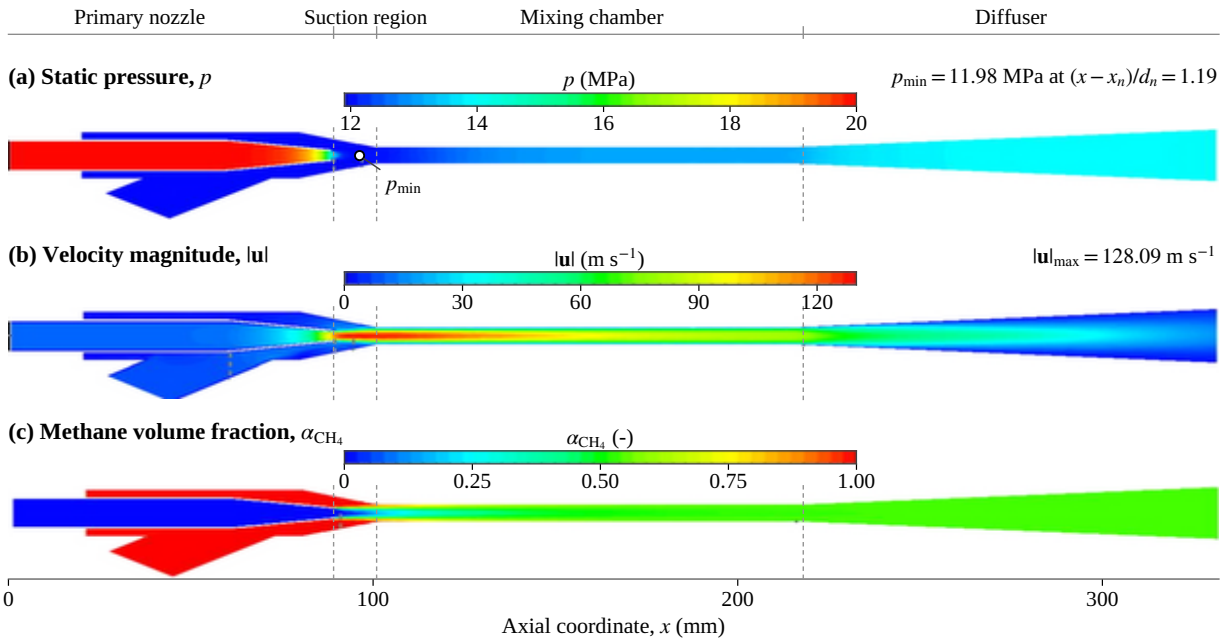


Figure 2. Reference axial-plane CFD contours at $P_p = 20$ MPa, $P_s = 12$ MPa, and $P_c = 14$ MPa: (a) static pressure, (b) velocity magnitude, and (c) methane volume fraction. Adapted from She (2020).

With $Q_m = \dot{m}_p/\rho_p + \dot{m}_s/\rho_s$, $u_m = Q_m/A_m$, $u_o = Q_m/A_d$, and $\rho_m = (\dot{m}_p + \dot{m}_s)/Q_m$, the mixing and diffuser closure is written as

$$P_f = P_e + \frac{\dot{m}_p u_p + \dot{m}_s u_s - (\dot{m}_p + \dot{m}_s) u_m}{A_m} - K_m \frac{\rho_m u_m^2}{2}, \qquad P_c = P_f + \frac{\rho_m}{2}(u_m^2 - u_o^2) - K_d \frac{\rho_m u_o^2}{2}. \tag{3}$$

Here A_m and A_d are the mixing-chamber and diffuser-outlet areas, respectively, and P_f is the pressure at the mixing-section exit. For each boundary condition, the effective mixing-entry pressure P_e is obtained from the outlet-pressure relation. The three fitted closure parameters, $C_{A,p}$, $C_{A,s}$, and K_m, were obtained by bounded nonlinear least squares over the CFD database; the objective uses range-scaled primary- and secondary-flow residuals while the outlet-pressure relation is enforced through the implicit solution for P_e. The remaining loss coefficients and both phase densities were prescribed and held fixed. The density values therefore form part of the compact fixed-property 1D representation over the investigated operating range. The fitted and fixed quantities are summarized in the left block of Table 3.

3.2 CFD-Derived Pressure-Response Model

The CFD-derived model maps the three boundary pressures directly to the primary and secondary mass-flow rates. The CFD database consists of controlled pressure sweeps about the reference condition, so the individually characterized pressure contributions are combined additively:

$$\dot{m}_j^{\mathrm{sur}} = \dot{m}_{j,0} + \sum_{k \in \{p,s,c\}} \Delta\dot{m}_{jk}, \qquad \Delta\dot{m}_{jk} = a_{jk}\Delta P_k + b_{jk}(\Delta P_k)^2, \tag{4}$$

where $j \in \{p, s\}$ and $\Delta P_k = P_k - P_{k,0}$ in MPa. For each pressure–output pair, a_{jk} and b_{jk} were obtained by linear least squares with the common CFD reference response retained as the intercept. The polynomial form is continuously differentiable and translates directly into Modelica equations without interpolation tables. The fitted coefficients are listed in the right block of Table 3.

Linear, quadratic, cubic, piecewise-linear (PWL), and monotone piecewise cubic Hermite (PCHIP) alternatives were assessed on the same pressure-response series, spanning compact global response surfaces and local interpolation forms commonly used in computer experiments and surrogate modeling (Sacks et al. 1989; Forrester et al. 2008). The PCHIP candidate follows the monotone construction of Fritsch and Carlson (1980). Interior-point leave-one-out (LOO) tests and endpoint holdouts were quantified with

$$\mathrm{NRMSE} = \frac{\sqrt{N^{-1}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}}{y_{\max} - y_{\min}}. \tag{5}$$

For family screening, the denominator is the range of the corresponding pressure-sweep/output series. The quadratic model gives a maximum interior-point LOO NRMSE of 2.71% and an endpoint-holdout maximum of 8.75%. Cubic and PCHIP reduce the endpoint maximum to 2.64% and 1.90%, respectively. The quadratic form is retained because it is globally smooth, explicit, and compact—only two coefficients are needed for each pressure–output contribution—while its interior-point error remains low enough for the intended system studies.

3.3 Comparative Assessment

Figure 3 compares both reduced models with the CFD points along the three pressure sweeps. The primary-flow

Table 3. Parameters of the two reduced models. The 1D block distinguishes globally fitted closure parameters from prescribed quantities; surrogate pressure increments are expressed in MPa.

(a) Effective-area 1D model

Parameter	Value	Treatment
$C_{A,p}$	0.899635	fitted
$C_{A,s}$	0.481877	fitted
K_m	0.083306	fitted
K_n	0.05	fixed
K_s	0.10	fixed
K_d	0.10	fixed
ρ_p (kg m^{-3})	991.7	fixed
ρ_s (kg m^{-3})	77.166	fixed

(b) Quadratic CFD-derived model

Relation	a_{jk}	b_{jk}
$P_p \to \dot{m}_p$	0.202589	-0.004415
$P_p \to \dot{m}_s$	0.094642	-0.003248
$P_s \to \dot{m}_p$	-0.183681	-0.006106
$P_s \to \dot{m}_s$	0.298605	0.029386
$P_c \to \dot{m}_p$	-0.016311	0.001481
$P_c \to \dot{m}_s$	-0.466796	-0.007058

a_{jk} in kg s^{-1} MPa^{-1}; b_{jk} in kg s^{-1} MPa^{-2}.

curves are close to one another. Larger differences appear in the secondary stream: the globally calibrated 1D model follows the overall trend, while the surrogate tracks the CFD response more closely as secondary pressure and back pressure change.

The like-for-like comparison uses the reference state as a fixed anchor and withholds each of the remaining 20 CFD points in turn. Within every fold, the surrogate coefficients and the fitted 1D parameters $(C_{A,p}, C_{A,s}, K_m)$ are re-estimated; K_n, K_s, K_d, ρ_p, and ρ_s remain fixed. Errors are normalized by the global CFD range of each response. The surrogate gives NRMSEs of 1.19%, 0.80%, and 0.71% for $\dot{m}_p$, $\dot{m}_s$, and μ, compared with 1.31%, 5.49%, and 5.52% for the 1D model. Figure 4(b) thus shows almost the same primary-flow accuracy, but a clear separation for secondary flow and entrainment ratio.

Both descriptions are inexpensive compared with a CFD solve. The surrogate evaluates the pressure–flow map explicitly, whereas the 1D formulation also solves for the effective mixing-entry pressure. For the scheduling study, the surrogate is preferred because it reproduces the secondary-flow response more closely and provides an explicit relation that can be inverted along the characterized back-pressure branch.

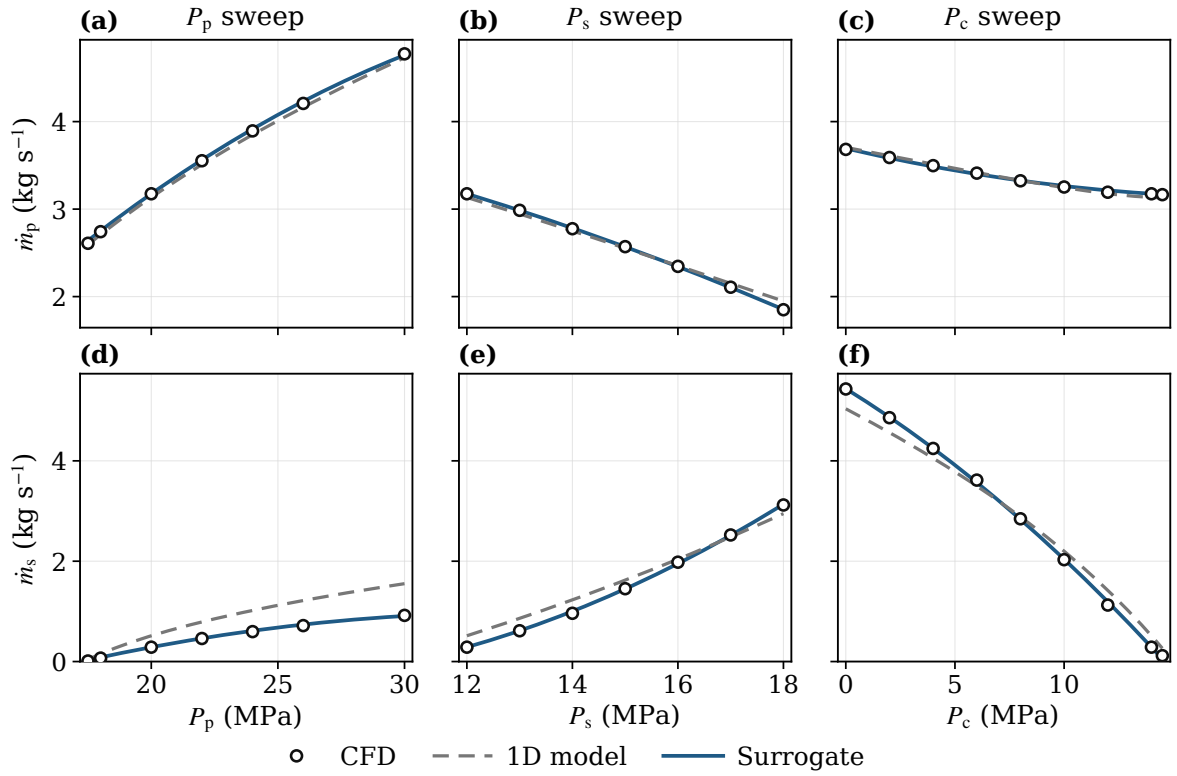


Figure 3. Pressure-response comparison of CFD data, the 1D model, and the CFD-derived surrogate. Panels (a)–(c) show primary mass flow and panels (d)–(f) secondary mass flow.

DOI
10.3384/ecp223167

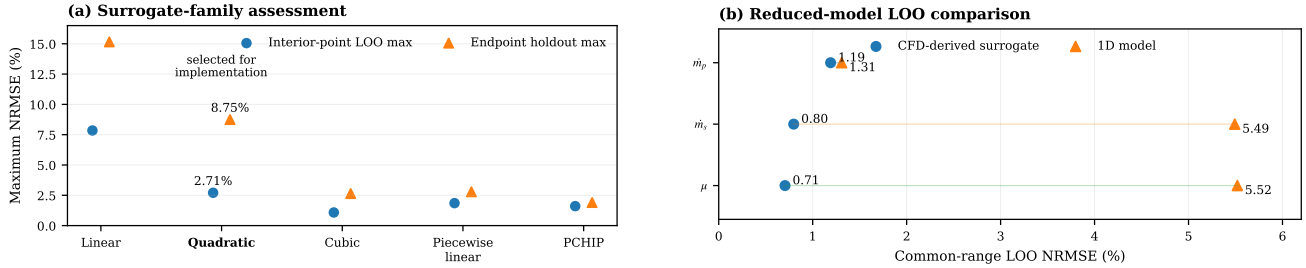


Figure 4. Model assessment. Panel (a) compares interior-point LOO and endpoint-holdout errors across candidate surrogate families, normalized by the range of each pressure-sweep/output series. Panel (b) uses the common global CFD response range for like-for-like comparison of the surrogate and 1D model.

4 Modelica and FMI Deployment

4.1 FMI Export and Numerical Verification

Both reduced models expose the same Modelica interface. The inputs are the three pressures (P_p, P_s, P_c), and the performance outputs are $(\dot{m}_p, \dot{m}_s, \mu)$. The CFD-derived component additionally reports operating-domain status information. Figure 5 shows the FMI exports and the surrogate scheduling path.

The components were written in Modelica 3.6 (Modelica Association 2023) and compiled with OpenModelica 1.26.3 (Open Source Modelica Consortium 2026). The 1D model was exported as an FMI 2.0 Model Exchange (ME) FMU, while the surrogate was exported in both ME and Co-Simulation (CS) forms. The 1D FMU has three inputs and three outputs. The surrogate also exposes three operating-domain diagnostics for documented-box membership, documented-axis membership, and status. Its physical outputs remain $\dot{m}_p$, $\dot{m}_s$, and μ. All three FMUs are algebraic and contain no internal continuous states.

Implementation verification was performed on the surrogate CS FMU using all 2 006 points of the scheduling trajectory. The differences from the native OpenModelica result are listed in Table 4 and remain at machine precision. This test checks consistency of the exported software artifact; it is separate from the CFD-based assessment of model accuracy (Oberkampf et al. 2004).

4.2 Inverse Back-Pressure Scheduling

The system example inverts the surrogate to obtain a back-pressure command from an entrainment-ratio target. The inlet pressures remain fixed at $P_p = 20$ MPa and $P_s = 12$ MPa. Along this characterized normal-flow branch, μ varies monotonically with P_c; each target used here therefore has a unique back-pressure command within the modeled range. A first-order actuator follows the scheduled pressure $P_c^\star$:

$$\begin{aligned} \mu(P_{p,0}, P_{s,0}, P_c^\star) &= \mu_{\text{tar}}, \\ \tau \dot{P}_c &= P_c^\star - P_c, \qquad \tau = 2 \text{ s}. \end{aligned} \tag{6}$$

The target is 0.20 over 0–20 s, 0.80 over 20–60 s, and 0.40 thereafter. These values correspond to back-pressure commands of 13.24, 8.48, and 11.78 MPa. At the three steady commands, the surrogate predicts $(\dot{m}_p, \dot{m}_s) = (3.188, 0.638)$, $(3.310, 2.648)$, and $(3.219, 1.287)$ kg s^{-1}. The secondary flow changes far more than the primary flow, consistent with the back-pressure sweep in Figure 3. Figure 6 shows the target, pressure response, and mass-flow histories.

The inverse map supplies the steady back-pressure command; the actuator supplies the dynamics. Using the settling condition $|\mu - \mu_{\text{tar}}| \le \max(0.01, 0.02|\mu_{\text{tar}}|)$ and requiring it to hold until the next target change, the two

Table 4. Native Modelica–surrogate CS FMU agreement over 2 006 application points.

Output	Max. $\lvert\Delta y\rvert$	RMSE
$\dot{m}_p$ (kg s^{-1})	1.33×10^{-15}	3.92×10^{-16}
$\dot{m}_s$ (kg s^{-1})	3.77×10^{-15}	1.41×10^{-15}
μ (-)	1.05×10^{-15}	4.42×10^{-16}

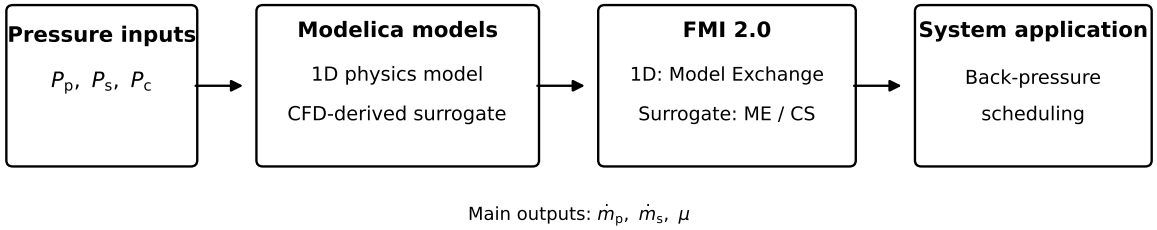


Figure 5. Modelica components, FMI 2.0 exports, and the inverse back-pressure scheduling path.

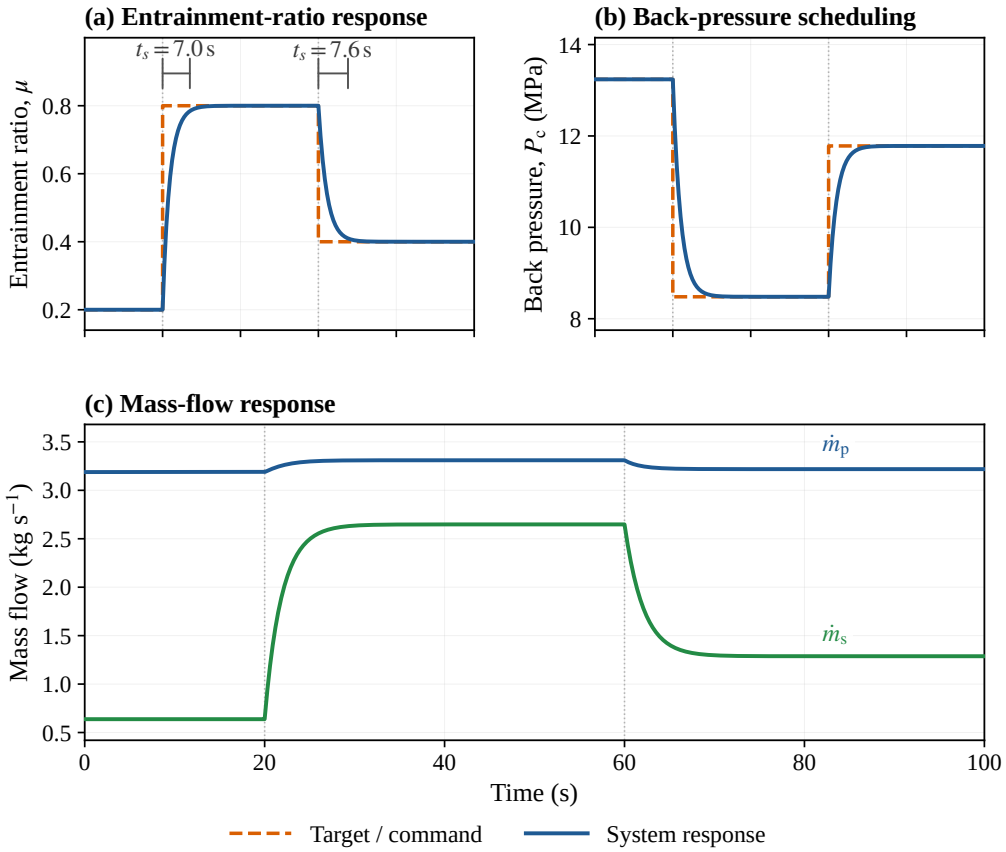


Figure 6. Inverse back-pressure scheduling: (a) entrainment-ratio target and response, (b) commanded and realized back pressure, and (c) primary and secondary mass-flow rates. Target changes occur at 20 and 60 s.

transitions settle in 7.0 and 7.6 s. These times describe the actuator-coupled system response, not an intrinsic fluid time constant of the algebraic ejector model.

5 Discussion

The results suggest distinct roles for the two reduced models. The 1D formulation keeps the control-volume structure visible and reproduces the primary-flow response almost as closely as the surrogate. Its larger error appears in the secondary stream, where suction and mixing are represented through lumped closures. The CFD-derived model fits the fixed-geometry pressure–flow behavior directly and therefore follows the secondary-flow response more closely. This pattern is consistent with the established use of one-dimensional ejector models in system studies (Antonio et al. 2012; Aidoun et al. 2019).

Because both components use the same Modelica interface, changing the ejector representation does not require changes elsewhere in the system model. The 1D component is useful when control-volume interpretation matters; the surrogate is better suited to repeated evaluation of the characterized CFD map. The common interface also permits model-form sensitivity to be examined without rebuilding the surrounding simulation.

For deployment, the quadratic form was selected for its compact, globally smooth algebraic structure. Cubic and PCHIP alternatives reduce endpoint errors, whereas the quadratic model remains explicit and requires no lookup table. Similar CFD-based reduced ejector models have been used in OpenModelica system studies (Haida, Smolka, et al. 2018; Haida, Fingas, et al. 2019). In the present model, the three independently characterized pressure contributions are superposed. The scheduling example uses only the directly characterized P_c branch, with P_p and P_s fixed at their reference values.

Operating-regime information is handled separately from the normal-flow regression. Reverse-flow CFD cases define status outputs that the surrounding Modelica system can monitor while the pressure–flow relation remains compact. The FMI comparison addresses another question altogether: whether the exported surrogate reproduces the native Modelica calculation (Oberkampf et al. 2004). It is therefore treated as implementation verification rather than as evidence of physical accuracy.

6 Conclusions

Two reduced representations of the same high-pressure water–methane ejector were evaluated against a 21-point CFD pressure-response database. The effective-area 1D model preserves a compact control-volume description, whereas the quadratic surrogate reproduces the characterized pressure–flow map directly. Under the common

DOI
10.3384/ecp223167

global-range LOO comparison, primary-flow NRMSE differs little between the 1D model and surrogate (1.31% and 1.19%). The separation is larger for secondary flow and entrainment ratio: the surrogate reduces NRMSE from 5.49% to 0.80% and from 5.52% to 0.71%. Cubic and PCHIP candidates improve endpoint interpolation, but the quadratic form was retained because it combines low interior-point error with a compact, smooth algebraic expression suited to repeated Modelica evaluation.

Both models were exported through FMI 2.0 from the same Modelica interface. On the scheduling trajectory, the surrogate CS FMU reproduces the native OpenModelica calculation to machine precision. The inverse scheduling example maps entrainment-ratio targets of 0.20, 0.80, and 0.40 to back-pressure commands of 13.24, 8.48, and 11.78 MPa. With the 2 s actuator, the two target changes settle in 7.0 and 7.6 s. Taken together, the two reduced models form a consistent hierarchy: the 1D formulation supplies a physics-based system representation, while the CFD-derived surrogate supplies a compact fixed-geometry response model for reusable Modelica/FMI simulations.

References

Aidoun, Zine, Khaled Ameur, Mehdi Falsafioon, and Messaoud Badache (2019). "Current Advances in Ejector Modeling, Experimentation and Applications for Refrigeration and Heat Pumps. Part 1: Single-Phase Ejectors". In: *Inventions* 4.1, p. 15. DOI: 10.3390/inventions4010015.

Antonio, Yveline Marnier, Christelle Périlhon, Georges Descombes, and Claude Chacoux (2012). "Thermodynamic Modelling of an Ejector with Compressible Flow by a One-Dimensional Approach". In: *Entropy* 14.4, pp. 599–613. DOI: 10.3390/e14040599.

Blochwitz, Torsten et al. (2011). "The Functional Mockup Interface for Tool Independent Exchange of Simulation Models". In: *Proceedings of the 8th International Modelica Conference*, pp. 105–114. DOI: 10.3384/ecp11063105.

Forrester, Alexander I. J., András Sóbester, and Andy J. Keane (2008). *Engineering Design via Surrogate Modelling: A Practical Guide*. Chichester, UK: Wiley.

Fritsch, Frederick N. and Ralph E. Carlson (1980). "Monotone Piecewise Cubic Interpolation". In: *SIAM Journal on Numerical Analysis* 17.2, pp. 238–246. DOI: 10.1137/0717021.

Haida, Michal, Rafal Fingas, Wojciech Szwajnoch, Jacek Smolka, Michal Palacz, Jakub Bodys, and Andrzej J. Nowak (2019). "An Object-Oriented R744 Two-Phase Ejector Reduced-Order Model for Dynamic Simulations". In: *Energies* 12.7, p. 1282. DOI: 10.3390/en12071282.

Haida, Michal, Jacek Smolka, Armin Hafner, Ziemowit Ostrowski, Michal Palacz, Andrzej J. Nowak, and Krzysztof Banasiak (2018). "System Model Derivation of the CO2 Two-Phase Ejector Based on the CFD-Based Reduced-Order Model". In: *Energy* 144, pp. 941–956. DOI: 10.1016/j.energy.2017.12.055.

Koirala, Ramesh, Quang L. Ve, Bin Zhu, Kiao Inthavong, and Abhijit Date (2021). "A Review on Process and Practices in Operation and Design Modification of Ejectors". In: *Fluids* 6.11, p. 409. DOI: 10.3390/fluids6110409.

Li, Chuan and Yi-Zhao Li (2011). "Investigation of Entrainment Behavior and Characteristics of Gas–Liquid Ejectors Based on CFD Simulation". In: *Chemical Engineering Science* 66.3, pp. 405–416. DOI: 10.1016/j.ces.2010.10.041.

Lindqvist, Oskar, Rickard Hällqvist, and Raghu Chaitanya Munjulury (2023). *Modelica Association Standards and Surrogate Modeling to Enable Multi-Fidelity Simulations*. Modelica Conferences. URL: https://www.ecp.ep.liu.se/index.php/modelica/article/view/915 (visited on 2026-08-13).

Modelica Association (2023). *Modelica Language Specification, Version 3.6*. URL: https://specification.modelica.org/maint/3.6/MLS.html (visited on 2026-08-13).

Modelica Association Project FMI (2014). *Functional Mock-up Interface for Model Exchange and Co-Simulation, Version 2.0*. URL: https://fmi-standard.org/assets/releases/FMI_for_ModelExchange_and_CoSimulation_v2.0.pdf (visited on 2026-08-13).

Oberkampf, William L., Timothy G. Trucano, and Charles Hirsch (2004). "Verification, Validation, and Predictive Capability in Computational Engineering and Physics". In: *Applied Mechanics Reviews* 57.5, pp. 345–384. DOI: 10.1115/1.1767847.

Open Source Modelica Consortium (2026). *OpenModelica User's Guide*. URL: https://openmodelica.org/doc/OpenModelicaUsersGuide/latest/ (visited on 2026-08-13).

Sacks, Jerome, William J. Welch, Toby J. Mitchell, and Henry P. Wynn (1989). "Design and Analysis of Computer Experiments". In: *Statistical Science* 4.4, pp. 409–423. DOI: 10.1214/ss/1177012413.

She, Luchao (2020). "Study on Performance and Structure Design of Liquid-Gas Ejector". Master's thesis, in Chinese. MA thesis. Harbin, China: Harbin Engineering University.

Sriveerakul, Thanarath, Satha Aphornratana, and K. Chunnanond (2007). "Performance Prediction of Steam Ejector Using Computational Fluid Dynamics: Part 1. Validation of the CFD Results". In: *International Journal of Thermal Sciences* 46.8, pp. 812–822. DOI: 10.1016/j.ijthermalsci.2006.10.014.

Wang, Lei, Chen Wang, Wenxiu Hou, Hongxia Zhao, and Hailun Zhang (2017). "Experimental Investigation on Ejector Performance Near Critical Back Pressure". In: *International Journal of Refrigeration* 80, pp. 158–168. DOI: 10.1016/j.ijrefrig.2017.05.010.

Yadav, R. L. and A. W. Patwardhan (2008). "Design Aspects of Ejectors: Effects of Suction Chamber Geometry". In: *Chemical Engineering Science* 63.15, pp. 3886–3897. DOI: 10.1016/j.ces.2008.04.012.

DOI
10.3384/ecp223167

An FMI Co-Simulation Framework for Day-ahead Optimization of District Cooling Systems Based on Modelica

Weidong Ma*, Feng Sha, Xiang Li, Rui Gao

Nanjing Yuansi SimTek Co., Ltd., China {weidong.ma, feng.sha, xiang.li, rui.gao}@simtek.cc

Abstract

The high penetration of renewable energy increases grid volatility, necessitating flexible demand-side resources such as District Cooling Systems (DCS) with thermal storage. However, the multi-timescale coupling of equipment, pipelines, and buildings makes it difficult to integrate high-fidelity physical models with optimization. This paper proposes an FMI-based co-simulation framework for day-ahead DCS optimization. A full-chain Modelica model is exported as an FMI 2.0 Co-Simulation FMU and coupled with MATLAB optimizers. The main contribution is a standardized FMU-in-the-loop workflow for day-ahead mixed-integer nonlinear programming (MINLP) scheduling with discrete unit commitment, together with a systematic quantification of the accuracy-runtime trade-off induced by FMI communication settings. Sensitivity analysis identifies a practical configuration for the investigated case, while a comparison of the genetic algorithm (GA) and surrogateopt reveals the solution-quality and computation-time trade-off. Compared with storage-priority control, the cost-optimization scenario reduces operating costs by 0.97%, the energy-optimization scenario reduces electricity consumption by 4.58%, and the carbon-optimization scenario reduces CO_2 emissions by 4.20%. The physical feasibility of the generated schedules is verified by dynamic FMU simulation.

Keywords: District cooling system, Day-ahead Optimization, FMI, co-simulation, Modelica

1 Introduction

Renewable energy integration and air-conditioning electrification intensify grid supply-demand fluctuations, making demand-side flexibility imperative (Gielen 2018; IEA 2021; Jordehi 2019). District cooling systems (DCS) with thermal storage leverage thermal inertia for load-shifting, offering an effective pathway to enhance demand-side electricity flexibility.

However, compared with conventional central air-conditioning systems, DCS exhibit greater structural complexity, involving the coupling of multiple components including chiller plants, thermal storage devices, distribution networks, and end-use cooling. The system flexibility exploitation critically depends on the sophistication of operational strategies. Traditional rule-based control strategies, such as chiller sequencing based on return water temperature deviations (Huang et al. 2016; Li et al. 2014), although reliable in engineering practice, struggle to accommodate multi-scenario operational optimization (Dai et al. 2025; Shan et al. 2016; Zou et al. 2023). Therefore, constructing high-fidelity simulation models and integrating them with advanced optimization algorithms to generate day-ahead control strategies has become crucial for achieving flexible DCS operations.

With the advancement of computational power and optimization theory, mathematical-programming-based control strategy generation has become an important research direction. For mixed-integer nonlinear programming (MINLP) problems with discrete variables and non-convex characteristics, heuristic methods such as the genetic algorithm (GA) (Chan, Chang, and Huang 2017; Du, Li, and Li 2020) and particle swarm optimization (PSO) (Karami and Wang 2018) are widely used because they are robust to non-convexity.

The fidelity of simulation models is a decisive factor in the credibility of optimization results. Current research mainly adopts two modelling paradigms: data-driven black-box models trained from historical operational data and mechanism-driven white-box models constructed based on mass and energy conservation equations. The black-box models offer high computational efficiency but suffer from limited generalisability beyond the range of training data and lack interpretability of underlying physical mechanisms. While white-box models offer strong physical interpretability and adaptability across operating conditions, their modelling complexity is comparatively high. In the domain of cooling system simulation, EnergyPlus, TRNSYS, and Modelica are three mainstream tools. Modelica offers unique advantages for multi-timescale coupling problems (equipment-level: second-scale, dynamics; pipe network-level: minute-scale, delays; building-level: hourly-scale, thermal inertia) through its equation-based paradigm and the ASHRAE-validated Buildings library (Hinkelman et al. 2022).

Nevertheless, physical simulation platforms typically lack native optimization interfaces, making it difficult to directly integrate "high-fidelity models" with "efficient optimization algorithms", thereby constraining the engineering deployment of sophisticated control strategies. To overcome this bottleneck, four approaches for co-simulation between Modelica model and optimization tools have been explored:

1. Script-driven loose coupling: This approach automates data exchange between MATLAB and Dymola by generating Dymola scripts (Karami et al. 2018). Although simple to implement, it requires restarting the simulation environment for each iteration, leading to a substantial increase in total computational time over hundreds of iterations, thus limiting its engineering feasibility.

2. Dedicated toolbox coupling: Utilising toolkits such as OMMATLAB enables direct connection between OpenModelica and MATLAB. However, this approach is restricted to the OpenModelica platform and has limited generalisability (Alizadeh et al. 2023).

3. Tool-level coupling: The Optimica–CasADi integration leverages the modeling and simulation platform Modelon Impact (via its Optimica compiler) and the automatic differentiation-based optimization framework CasADi to perform simulation-based optimization. Currently, Optimica is the only Modelica simulation tool that supports seamless integration; however, this comes at the cost of reduced model fidelity (Köppen et al. 2022).

4. FMI-based standardised coupling: Modelica models are exported as FMU following the FMI standard, enabling cross-tool integration with MATLAB/Simulink and Python (Müller et al. 2025). The model is exported based on the FMI 2.0 standard as a co-simulation FMU, stably handling stiff thermo-hydraulic dynamics (time constants from seconds to hours). This approach combines portability (model once, reuse across platforms) with computational efficiency (memory-level data exchange) and has become an industrial standard for complex-system co-simulation.

Despite the growing maturity of FMI technology, the application of a high-fidelity FMU to day-ahead DCS scheduling with discrete unit commitment remains insufficiently documented. Existing DCS studies mainly emphasize physical modeling and simulation, while real-time MPC studies often use reduced-order models. The sensitivity of FMI communication parameters to both simulation error and optimization runtime is also rarely quantified in a DCS scheduling context.

This study addresses these gaps through three contributions: (i) a standardized FMI 2.0 Co-Simulation workflow that embeds a high-fidelity, full-chain DCS FMU in an external MATLAB optimization loop; (ii) a MINLP formulation that schedules discrete chiller commitment and thermal-storage operation over a 24 h horizon; and (iii) a sensitivity analysis that quantifies how the FMI communication interval and numerical tolerance affect simulation error and optimization runtime. The positioning is distinct from related work. Hinkelman et al. (2022) demonstrated detailed Modelica-based district cooling modeling and simulation, but did not address standardized FMU-in-the-loop day-ahead MINLP scheduling with discrete unit commitment. Spawn couples the Modelica Buildings Library with EnergyPlus for building and HVAC co-simulation (Wetter et al. 2024), while the present work focuses on external optimization of a chilled-water-storage DCS through FMI. These differences define the scope of the contribution.

2 Methodology

2.1 Overall Architecture

This paper proposes an operational strategy optimization architecture for district cooling systems based on FMI co-simulation framework, as illustrated in Figure 1. The workflow comprises: (i) Modelica model export as FMU; (ii) Simulink import with Workspace variable binding; (iii) MATLAB optimization function construction with FMU invocation; (iv) heuristic solver execution, forming a closed-loop 'decision-simulation-evaluation' iteration.

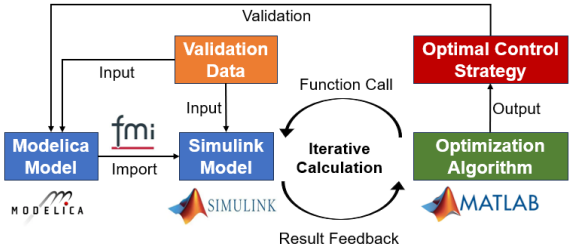


Figure 1. Architecture of the FMI-based co-simulation framework integrating the Modelica and MATLAB.

2.2 Modeling using Modelica Language

Modelica is well suited to multi-physics system simulation, while the Modelica Buildings Library provides validated components for heating, ventilation, and air-conditioning systems (Wetter et al. 2014). The case study selects a district cooling energy station with chilled-water storage in Shanghai, China. A dynamic model covering the entire refrigeration-distribution-consumption chain is constructed using the Modelica Standard Library and the Buildings Library.

The case study is located in a hot-summer and cold-winter climate zone, providing air-conditioning chilled water to 16 commercial and public buildings in the surrounding area during summer. The system design parameters are as follows: peak cooling load of 55,792 kW, typical design day cooling capacity of 576,417 kWh,

DOI
10.3384/ecp223175

and a configured chilled water storage tank with a volume of 20,000 m³. The constructed simulation model comprises four core modules: the chiller plant model, the distribution pipe network model, the building thermal balance model, and the data interaction and control interface model, enabling full-process dynamic simulation from cooling source generation to cooling distribution and end-use consumption.

2.2.1 Chiller Plant Model

The system architecture of the chiller plant is shown in Figure 2. The system comprises two base-load chillers (3,518 kW each), three storage chillers (7,030 kW each), ten air-source heat pumps (1,354 kW each), and a 20,000 m³ stratified storage tank. The component models are selected from the Buildings library (e.g., `Buildings.Fluid.Chillers.ElectricEIR`) and customised for special operating conditions based on the DOE-2 fitting method. Thus, component-level simulation models meeting engineering accuracy requirements can be rapidly generated with equipment design parameters and a small amount of operational data.

The chiller plant model shown in Figure 3 encapsulates single chiller models, their associated cooling towers, cooling water pumps, and control components, and constructs multi-unit parallel logic to achieve a simplified representation of the coordinated operation of multiple identical chillers in parallel, thereby reducing model complexity. This model supports dynamic regulation of system cooling capacity through discrete variables (unit on/off status), providing a reliable model foundation for subsequent optimization strategy generation.

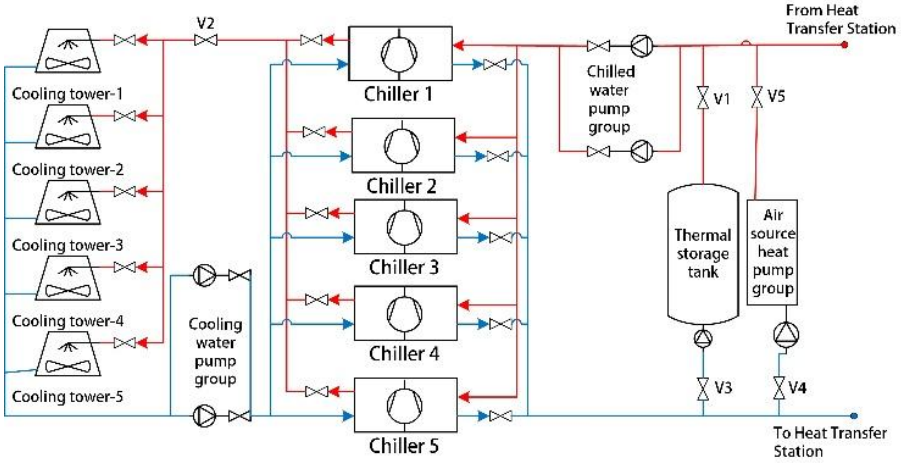


Figure 2. System diagram of chilled water storage district cooling system in summer cooling mode.

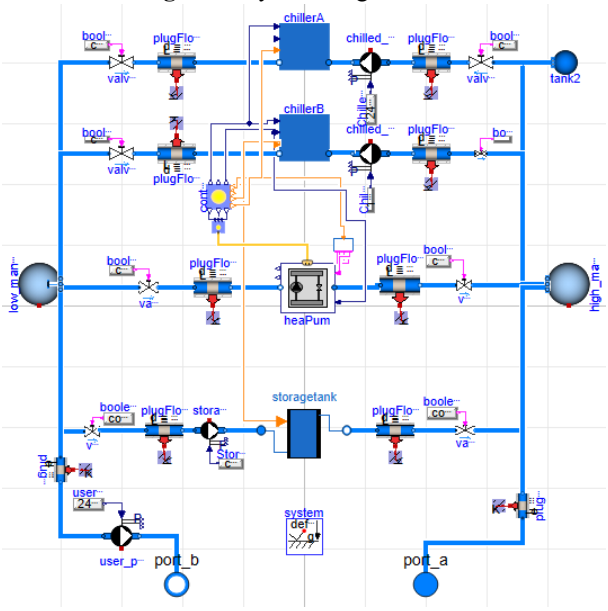


Figure 3. Chiller plant model.

2.2.2 Distribution Pipe Network Model

District energy systems typically involve long chilled water transportation distances; in this case study, the furthest branch reaches 1.2 km, making the time-lag effect and thermal inertia of the pipe network significant factors influencing system dynamic response characteristics. The case study serves 16 commercial buildings, each equipped with an energy transfer station (ETS). The secondary circulation loop between the chiller plant and the end-users adopts a variable flow design, and the overall distribution pipe network is divided into four independent branches, with its topology shown in Figure 4.

PlugFlowPipe components capture temperature wave propagation delays ($\tau = L/v$) while controlling computational complexity through the method of characteristics. Given the differentiated hydraulic characteristics of the four branches, hydraulic–thermal coupled dynamic models based on distributed parameters are constructed for a typical branch (Figure 5). These pipe network models capture hydraulic coupling relationships among multiple energy transfer stations, simulate flow redistribution within the pipe network caused by local adjustments, and quantify the contribution of pipe network thermal inertia to system flexible regulation, providing a physical basis for temporal optimization of storage strategies.

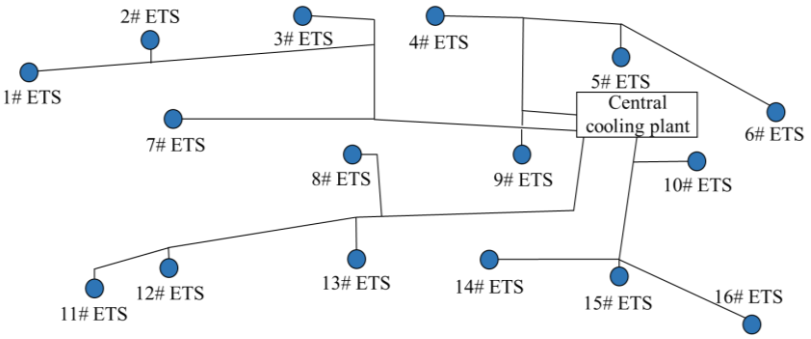


Figure 4. Distribution pipe network structure.

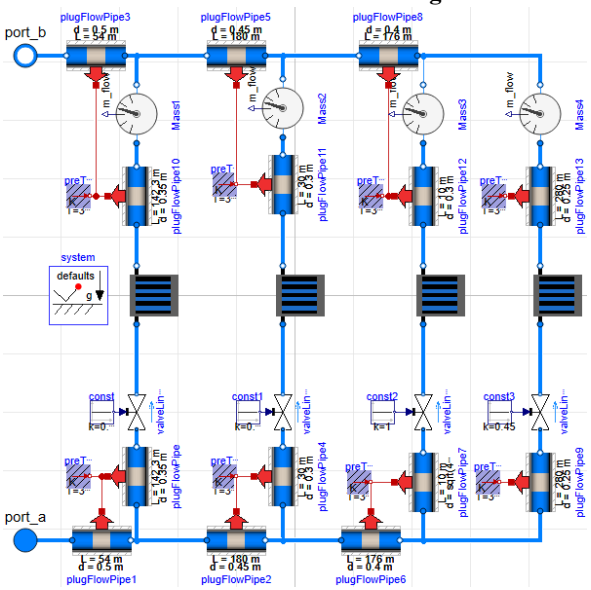


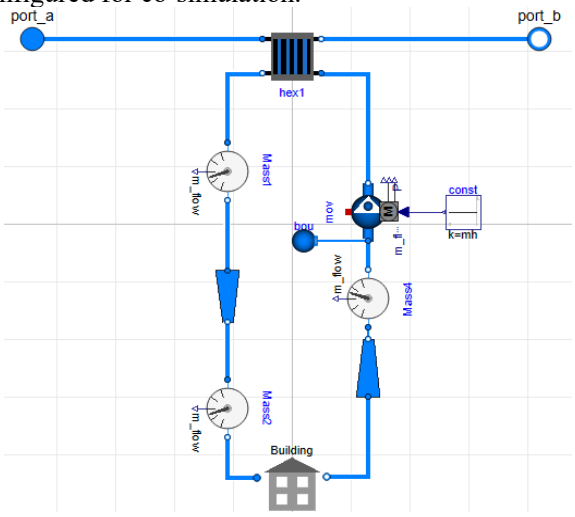
Figure 5. Simulation model of a single branch of the distribution pipe network.

2.2.3 Building Thermal Balance Model

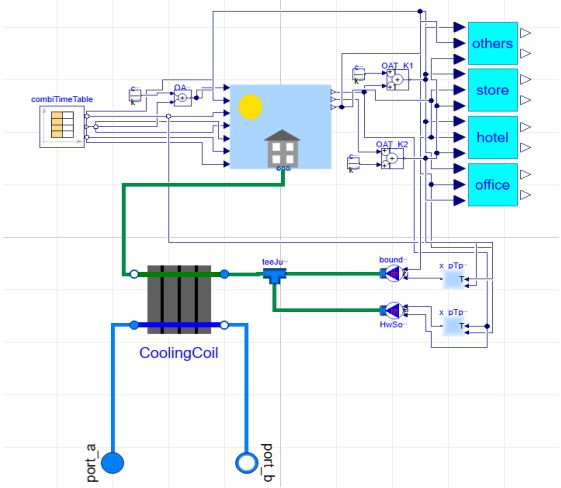
The building thermal balance model is primarily used to simulate the dynamic thermal and moisture balance processes of the air-conditioning system under cooling and dehumidification conditions. This model consists of four key subsystems: the chilled water secondary circulation model, the air handling unit model, the single-zone thermal and moisture balance model, and the thermal comfort quantitative evaluation model. Figure 6(a) illustrates the chilled water secondary circulation simulation model from the energy transfer station to a single building, where the Building component represents an individual building, with its internal structure shown in Figure 6(b). Zone thermal dynamics are simulated using a first-order RC network (`Buildings.ThermalZones.ReducedOrder.RC.OneElement`), with PMV-PPD thermal comfort evaluation per ASHRAE Standard 55.

2.2.4 Data Interaction and Operation Control Interface

After integrating the three modules (the chiller plant, distribution pipe network, and building thermal balance model) into the DCS model, dynamic simulation is performed by inputting operating condition parameters to obtain hourly operational data. Four input categories (weather, economic, load, control) and four output categories (energy, thermal, comfort, emissions) are configured for co-simulation.



(a) energy transfer station model.



(b) zone thermal balance model.

Figure 6. Building thermal balance simulation model.

2.3 FMU Export and Configuration

The validated Modelica model is exported as an FMU via FMI 2.0 Co-simulation, with the internal DASSL solver for stiff thermo-hydraulic dynamics while enabling memory-level data exchange through MATLAB Workspace binding.

DOI 10.3384/ecp223175

2.4 Day-ahead Optimization Algorithm Integration

This study focuses on day-ahead operational strategy optimization for storage-equipped district cooling systems, selecting a 24-hour hourly optimization time horizon. This choice is based on three considerations: (i) current demand response policies require demand-side flexible resources to declare regulation plans 24 hours in advance; (ii) district cooling systems possess significant thermal inertia (storage tank capacity of 20,000 m³), enabling day-ahead regulation to cover major load shifting requirements; (iii) the physical model has a single simulation time which aligns with the day-ahead decision-making time window.

The number of operating chillers of each type and the charging/discharging variables of the thermal storage device are selected as optimization variables. Auxiliary equipment such as pumps and cooling towers are interlocked to start and stop with their corresponding chillers, and the chilled water supply temperature is fixed at the design value.

The day-ahead optimization (24-hour horizon, hourly resolution) minimizes a scalar objective function. The decision variables include continuous variables ($Q_{tank,i}$) and integer variables ($n_{chillerA,i}$, $n_{chillerB,i}$, $n_{hp,i}$), rendering the problem a MINLP. The unified formulation is shown in Equation (1):

$$\begin{aligned} &min\ f(Q_{tank}, n_{chillerA}, n_{chillerB}, n_{hp}) \\ &= \sum_{i=1}^{a} g(Q_{tank,i}, n_{chillerA,i}, n_{chillerB,i}, n_{hp,i}) \end{aligned} \quad (1)$$

where g(·)is the performance indicator function decided based on optimization scenarios: daily energy consumption (Strategy 1), time-of-use electricity cost (Strategy 2), carbon emissions (Strategy 3).

Constraints include cooling supply-demand balance, indoor thermal comfort, storage capacity limits, and equipment rating bounds, as follows:

$$Q_{\mathrm{cs},i} + Q_{\mathrm{chiller},i} \geq Q_{\mathrm{c},i} \quad (2)$$

$$24℃ \leq T_{room,i} \leq 26℃ \quad (3)$$

$$0 \leq E_{tank,i} \leq E_{tank,max} \quad (4)$$

$$-Q_{\mathrm{sa}} \leq Q_{\mathrm{cs},i} \leq Q_{\mathrm{sa}} \quad (5)$$

$$0 \leq Q_{\mathrm{chiller},i} \leq Q_{\mathrm{ref}} \quad (6)$$

where i is the time index; $Q_{\mathrm{cs},i}$ is the hourly cooling discharge power of the thermal storage device (kW); $Q_{\mathrm{chiller},i}$ is the hourly cooling capacity of the chillers (kW); $Q_{\mathrm{c},i}$ is the hourly user cooling load (kW); $E_{tank,i}$ is the hourly effective stored cooling capacity of the storage tank (kWh); $E_{tank,max}$ is the maximum storage capacity of the storage tank (kWh); Q_{sa} is the maximum cooling discharge power of the storage system (kW); $-Q_{\mathrm{sa}}$ is the maximum cooling charging power of the storage system (kW); Q_{ref} denotes the total rated cooling capacity of all available chiller and heat pump units (kW). The unit-commitment variables $n_{chillerA,i}$, $n_{chillerB,i}$ and $n_{hp,i}$ are restricted to non-negative integers, reflecting the discrete on/off nature of equipment dispatch.

This study employs the Global Optimization Toolbox in MATLAB to solve the optimization problems. The Global Optimization Toolbox provides solvers based on various optimization algorithms for MINLP problems addressed in this paper. To compare optimization performance, both the surrogateopt and GA algorithms are used to solve the objective functions.

3 Case Study

This section validates the proposed framework using a district cooling station under construction in Shanghai. Because the target project is not yet operational, model accuracy is evaluated by cross-case comparison with measured data from an existing reference system. The framework then generates day-ahead operational strategies and compares optimizer performance. Claims about the target project are therefore simulation-based until direct measured data become available.

3.1 Verification of Modeling Approach Using a Reference System

Given that the case study project has been under construction, actual operating data are currently unavailable for model validation. Model accuracy was validated against measured data from an operational reference system located in a hot-summer and cold-winter climate zone, serving a total building area of 225,191 m² with a design cooling load of 12,036 kW.

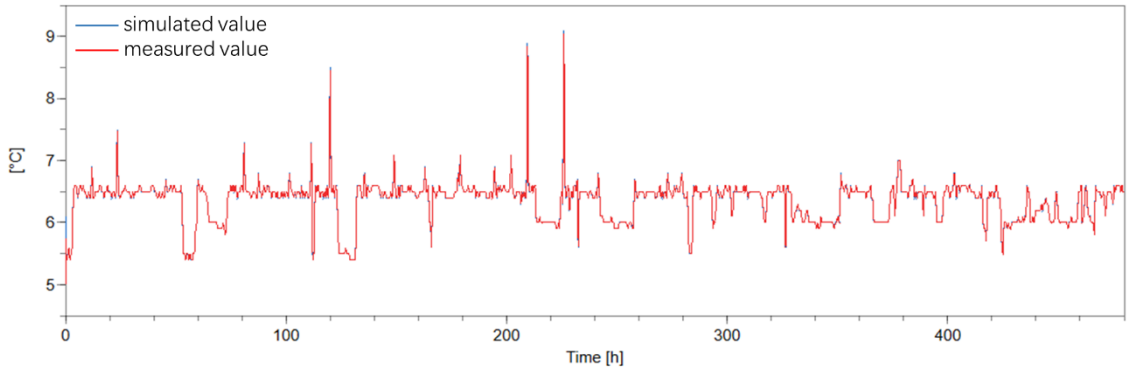
The chiller model was calibrated using the reference curve method (Hydeman et al. 2002). A performance curve providing the best fit (`ElectricEIRChiller_Trane_CVHF_1758kW_6_87COP_Vanes`) was selected from 157 candidates and achieved a CVRMSE of 3.17% for chiller power and 0.28% for cooling capacity (Figure 7a, 7b, 7c). The absolute error in cooling tower outlet water temperature was within ±0.8°C, the chilled water outlet temperature error was less than 0.5°C (Figure 7d, 7e).

It should be noted that the current validation focuses on verifying the modeling methodology and the applicability of the component libraries. Future work will involve direct model calibration based on measured data once the project becomes operational.

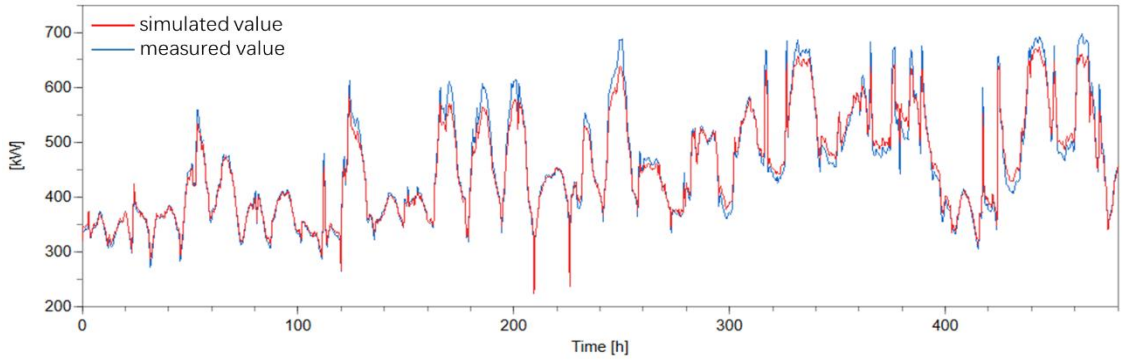
3.2 Operational Strategy Results

The hourly data of a typical summer day was chosen as boundary conditions. Figure 8 illustrates the 24 h electricity-price vector and peak/valley periods, the hourly grid-emission-factor vector, and the hourly weather data. Baseline is storage-priority control strategy, priority discharging during peak tariff periods, with any cooling shortage supplemented by chillers. The optimal operational strategies under three scenarios are

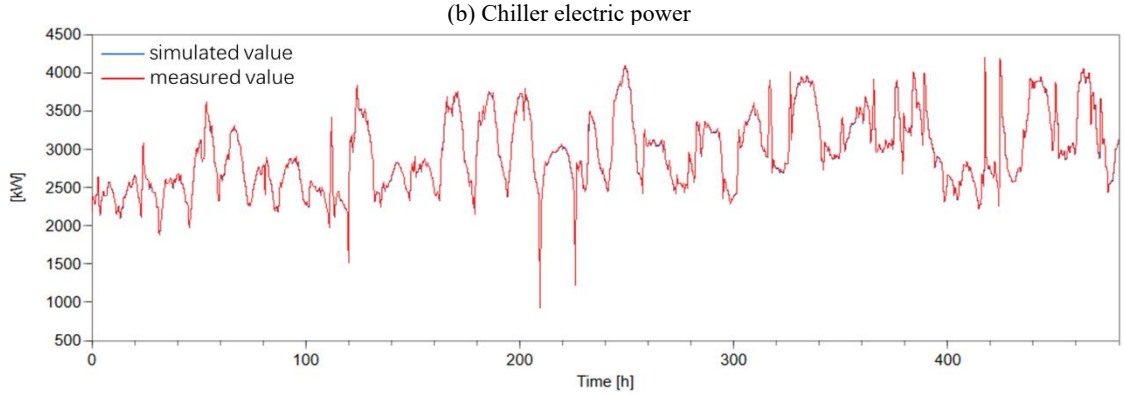
solved to quantitatively evaluate the effectiveness of the proposed co-simulation framework. Figure 9 illustrates the hourly dispatch status and system power consumption under different strategies, while Table 1 summarizes the simulation results of core performance indicators. The system-level daily energy efficiency ratio (EER) is defined as the ratio of total cooling supplied in kWh and total electricity consumed by the chillers and auxiliary equipment in kWh (pumps, cooling towers, and heat pumps are included). EER is therefore dimensionless.



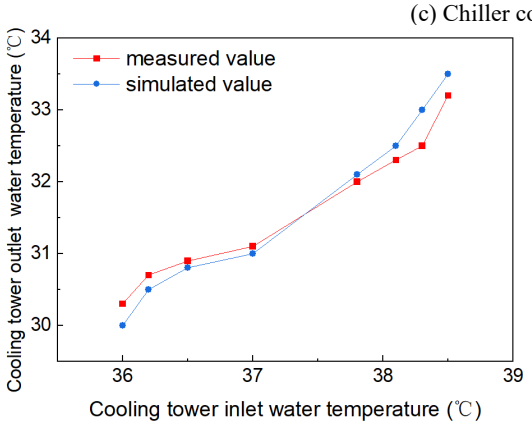
(a) Chilled water outlet temperature



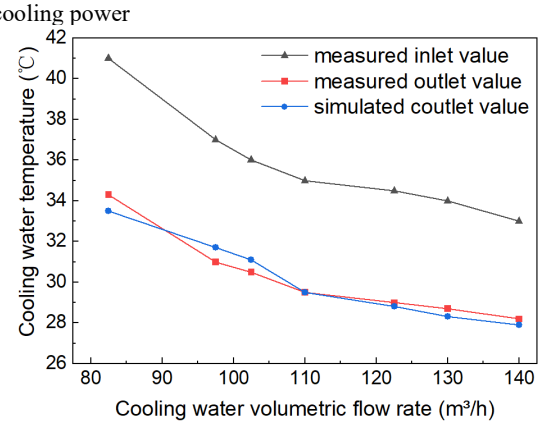
(b) Chiller electric power



(c) Chiller cooling power



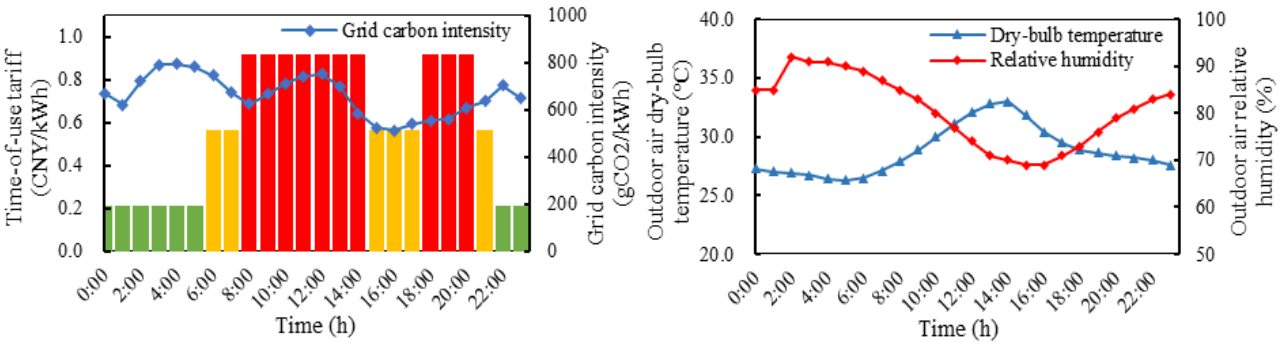
(d) constant cooling water flow rate



(e) variable cooling water flow rate

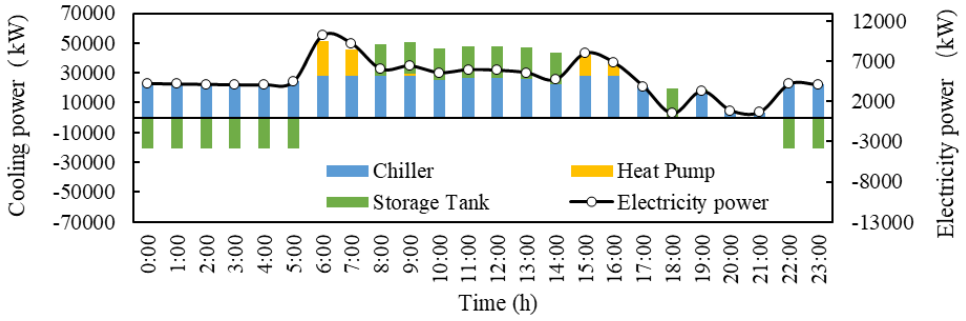
Figure 7. Comparison of measured and simulated values of key parameters in the district cooling system.

DOI
10.3384/ecp223175

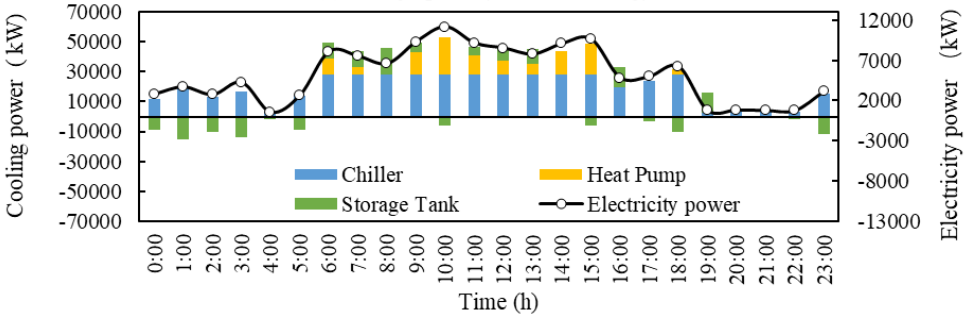


(a) hourly electricity boundary conditions (b) hourly meteorological boundary conditions

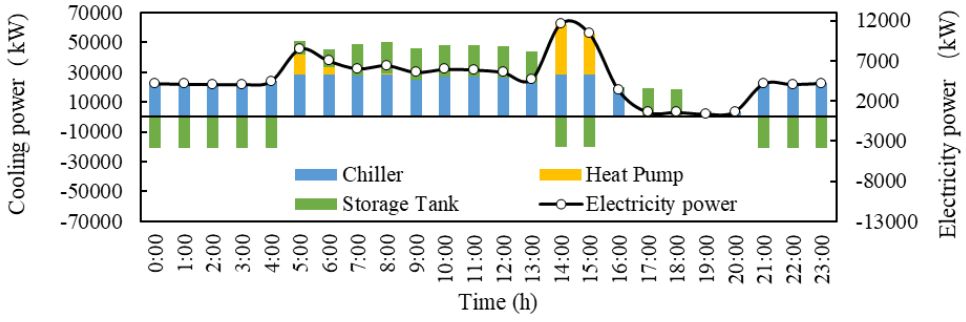
Figure 8 Hourly boundary conditions for typical summer day.



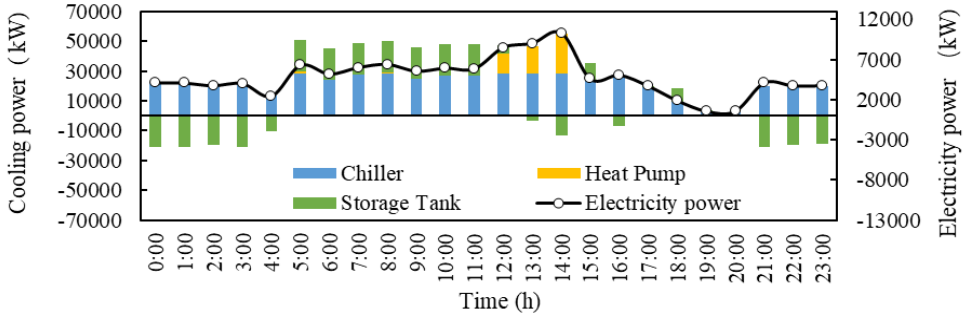
(a) storage-priority control strategy



(b) minimum daily energy consumption strategy



(c) minimum daily operating cost strategy



(d) minimum daily carbon emission strategy

Figure 9. Hourly cooling supply and power consumption of the district energy system under different strategies on a typical day.

Table 1. Quantitative evaluation indicators of district energy system operational performance on a typical day.

Indicator	*Baseline*	*Strategy 1*	*Strategy 2*	*Strategy 3*
Cooling supply (kWh)	582,809	582,809	582,809	582,809
Electricity consumption (kWh)	125,410	119,662(-4.58%)	122,063(-2.67%)	120,600(-3.84%)
Operating cost (RMB)	79,494	86,374(+8.66%)	78,720(-0.97%)	85,377(+7.40%)
Carbon emissions ($kgCO_2$)	74,269	71,241(-4.08%)	72,271(-2.69%)	71,147(-4.20%)
Energy efficiency ratio	4.65	4.87	4.77	4.83

As shown in Figure 9 (b), the minimum energy consumption strategy dynamically adjusts the number of operating chillers by leveraging the thermal storage device, reducing the amount of stored energy within the day. Strategy 1 improves EER from 4.65 to 4.87 and reduces daily electricity consumption by 4.58% by optimising unit commitment to maintain high-efficiency partial-load operation.

As shown in Figure 9 (c), the minimum-cost strategy charges storage when electricity is less expensive and discharges it during peak-tariff periods, while the chiller commitment is shifted away from the most expensive hours. This reduces operating cost by 0.97% (774 RMB/day). Table 1 reports electricity consumption as 122,063 kWh, a 2.67% reduction relative to the baseline. Although Strategy 2 achieves a 0.97% cost reduction relative to the baseline, the absolute savings (774 RMB/day) must be contextualized against operational uncertainties.

As shown in Figure 9 (d), the minimum daily carbon emissions strategy achieves 4.20% emissions reduction, limited by the flat grid carbon intensity profile (0.58-0.62 $kgCO_2$/kWh) and storage cycle exergy losses.

3.3 Algorithm Performance Analysis

This section compares GA and surrogateopt in terms of convergence, solution quality, and computational efficiency using the MATLAB Global Optimization Toolbox. All experiments were run on an Intel® Core™ i7-10750H CPU @ 2.60GHz with 32 GB RAM.

3.3.1 Sensitivity Analysis of FMI Parameters

The computational efficiency of the co-simulation is constrained by the simulation time per FMU call, which is affected by FMI communication parameters such as step size and tolerance. To quantify the impact of FMI communication parameters, a reference "ground-truth" simulation is first executed with a step size of 1 s and a tolerance of 1e-6, serving as the benchmark. Under different fixed tolerance settings, the average time and simulation result error for a single FMU call were tested across various communication step sizes, as shown in Figure 10. The results indicate that step sizes below 900 s increase interface exchange frequency; beyond 900 s, computation time stabilises as the internal solver leverages larger integration steps. Tolerance dominates the accuracy-efficiency trade-off.

Based on the trade-offs observed with the GA under different FMI configurations, the recommended configuration (tolerance 1e-3, step size 900 s) ensures a power prediction RMSE of less than 4 kW while keeping the single optimization evaluation time within 20 seconds, meeting the engineering time requirements for day-ahead strategy generation.

3.3.2 Sensitivity Analysis of Optimization Algorithm Parameters

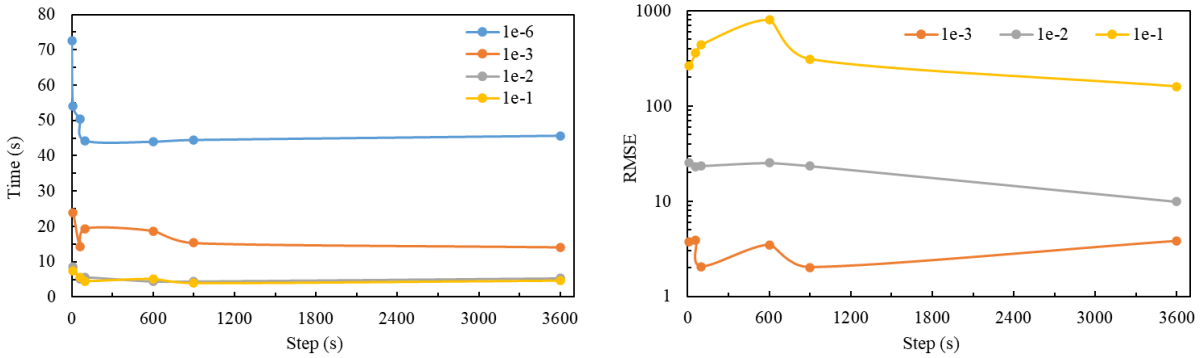
Within the FMI co-simulation framework, each evaluation of the objective function by the optimization algorithm requires a complete dynamic simulation using the FMU. Therefore, the algorithm's efficiency in terms of the number of simulations directly impacts the total optimization time. The baseline parameters for the optimization algorithms are set as follows: (1) GA: population size of 100, maximum generations of 200, crossover probability of 0.8, and adaptive mutation probability (1/chromosome length) dynamically adjusted during evolution to avoid premature convergence; (2) surrogateopt: termination condition set to a maximum of 4,000 function evaluations, with an objective function tolerance of 0.01.

The simulation results for the two optimization algorithms are shown in Figure 11. GA achieves the best solution (78,720 RMB) after 20,000 evaluations with strong global search capability, and surrogateopt converges rapidly initially via RBF surrogate approximation, reaching 95% of GA optimum by 1,681 evaluations, but plateaus at 80,016 RMB due to FMU nonlinearity.

To evaluate the convergence stability of the algorithms, multiple independent runs were conducted for the "cost minimization" objective using various GA parameter settings. The optimal objective value and computation time for each run are recorded in Table 2. The GA achieves significantly lower optimal solutions than surrogateopt, indicating stronger robustness under random initialization. However, the GA has a much longer total computation time. Within the permissible time window for day-ahead optimization (24 hours), surrogateopt demonstrates a speed advantage by reducing FMU calls through the surrogate model. In

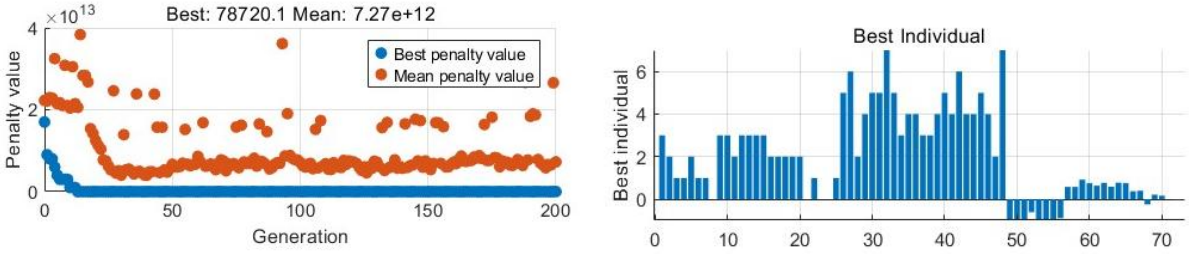
DOI
10.3384/ecp223175

summary, the accuracy of the surrogateopt algorithm's surrogate model is limited by the nonlinearity of the FMU input-output mapping. In contrast, although the GA requires more function evaluations, it more thoroughly explores the complex solution space represented by the high-fidelity FMU.



(a) Computation time (b) Simulation error

Figure 10. Performance of a single FMU simulation under different tolerances and step sizes.



(a) Optimizing process (b) Best evaluations

Figure 11. Convergence curves of the optimal objective function value over iterations.

Table 2. Comparison of convergence performance of different optimization algorithms.

Algorithm	*Optimal Solution (RMB)*	*Evaluations*	*Evaluations to 95% of GA Optimum*
Surrogateopt	80,016	4,000	1,681
GA (Pop size 100)	78,720	20,000	7,800
GA (Pop size 50)	78,720	10,000	9,000
GA (Pop size 20)	84,858	2,000	/

The algorithm performance analysis reveals the global convergence advantage of GA for MINLP problems. Considering convergence, robustness, and computational efficiency, the following engineering recommendation is proposed: for day-ahead optimization of district cooling systems, which are MINLP problems with discrete variables, it is recommended to use the surrogateopt algorithm with a 900-second FMI communication step size. Although this combination may be prone to local optima, it meets the time window requirements for day-ahead optimization of large-scale district cooling systems. For refined strategy effectiveness studies, GA can be employed to obtain a better solution.

4 Conclusion

This paper proposes and validates an FMI-based co-simulation optimization framework that places a district-cooling FMU inside an external day-ahead MINLP optimization loop. The principal contribution is the standardized FMU-in-the-loop scheduling workflow with discrete unit commitment and the associated FMI accuracy-runtime sensitivity analysis; the Modelica model serves as the physical validation vehicle.

(1) A full-chain dynamic simulation model covering the chiller plant, distribution network, and building thermal dynamics was developed and cross-case validated. The reference-system validation achieved a chiller-power CVRMSE of 3.17%.

(2) FMI 2.0 Co-Simulation enables standardized integration between the physical DCS model and MATLAB optimization. Compared with baseline storage-priority control, the cost-optimization scenario reduces operating costs by 0.97%, the energy-optimization scenario reduces electricity consumption by 4.58%, and the carbon-optimization scenario reduces CO_2 emissions by 4.20%, while the FMU verifies physical feasibility.

(3) For the investigated case, a 900 s FMI communication interval and a 1e-3 relative tolerance provide a practical accuracy-runtime compromise. GA obtains the best reported cost solution (78,720 RMB), while surrogateopt converges faster and reaches the 95%-of-GA benchmark after 1,681 evaluations. The

recommended optimizer therefore depends on the available day-ahead runtime budget and must be supported by measured total runtime rather than evaluation count alone.

In summary, the proposed framework bridges high-fidelity Modelica simulation and external optimization through a reusable FMI 2.0 Co-Simulation interface. Its demonstrated contribution is a verifiable and portable workflow for day-ahead scheduling with discrete equipment decisions, together with a practical method for tuning communication parameters. Future work will add direct calibration with measured data, multi-day and seasonal validation, and stochastic or robust optimization to account for load, weather, tariff, and carbon-intensity forecast errors.

Acknowledgements

No financial support was received for this work. The authors have no affiliations with or involvement in any organization or entity with any financial interest in the subject matter discussed in this manuscript.

References

Alizadeh, Mohammadhadi, Ali Sahlodin, Arunkumar Palanisamy, Francesco Casella, and Peter Fritzson (2023). "Application of the OpenModelica-MATLAB Interface to Integrated Simulation and Successive Linearization Based Model Predictive Control". In: *Modelica Conferences*. DOI: 10.3384/ecp204293.

Chan, T., Y. Chang, and J. Huang (2017). "Application of artificial neural network and genetic algorithm to the optimization of load distribution for a multiple-type-chiller plant". In: *Building Simulation* 10.5, pp. 711–722. DOI: 10.1007/s12273-017-0363-2.

Dai, C., T. Ma, Y. Zhang, et al. (2025). "Improvement of Buildings energy flexibility with PV battery system based on prediction and load management". In: *Building Simulation* 18.1, pp. 65–85. DOI: 10.1007/s12273-024-1216-4.

Du, Jiamin, Shuhong Li, and Xinmei Li (2020). "Energy-saving optimization of cooling system based on ResNet-GWO". In: *Journal of Southeast University (Natural Science Edition)* 50.5, pp. 866–874. DOI: 10.3969/j.issn.1001-0505.2020. 05.011.

Gielen, Dolf (2018). *Global Energy Transformation: A Roadmap to 2050*. Tech. rep. Abu Dhabi: IRENA. URL: https://sdgs.un.org/publications/global-energy-transformation-roadmap-2050-2019-edition-33240.

Hydeman, M., K.L. Jr, and A.L. Dexter (2002). "Tools and techniques to calibrate electric chiller component models." *ASHRAE Transactions* 108. pp. 733–741.

Hinkelman, K., J. Wang, W. Zuo, et al. (2022). "Modelica-based modeling and simulation of district cooling systems: A case study". In: *Applied Energy* 311, p. 118654. DOI: 10.1016/j.apenergy.2022.118654.

Huang, S., W. Zuo, and M. D. Sohn (2016). "Amelioration of the cooling load based chiller sequencing control". In: *Applied Energy* 168, pp. 204–215. DOI: 10.1016/j.apenergy. 2016.01.035.

International Energy Agency (2021). *An Energy Sector Roadmap to Carbon Neutrality in China*. Tech. rep. Paris: IEA. URL: https://www.iea.org/events/an-energy-sector-roadmap-to-carbon-neutrality-in-china.

Jordehi, A. R. (2019). "Optimization of demand response in electric power systems, a review". In: *Renewable and Sustainable Energy Reviews* 103, pp. 308–319. DOI: 10.1016/j.rser.2018.12.054.

Karami, Majid and Liping Wang (2018). "Particle Swarm optimization for control operation of an all-variable speed water-cooled chiller plant". In: *Applied Thermal Engineering* 130, pp. 962–978. DOI: 10.1016/j.applthermaleng.2017 11.037.

Köppen, A., F. Magnusson, S. Velut, et al. (2022). "Techno-economic assessment of an industrial project towards carbon neutrality". In: *Proceedings of Asian Modelica Conference 2022.* Tokyo, Japan, November 24-25, 2022. DOI: 10.3384/ ecp19325

Li, Z., G. Huang, and Y. Sun (2014). "Stochastic chiller sequencing control". In: *Energy and Buildings* 84, pp. 203–213. DOI: 10.1016/j.enbuild.2014.07.072.

Müller, S., A. Dahash, S. Akbar, et al. (2025). "Integrating a Seasonal Thermal Energy Storage FMU in a MATLAB/Simscape Thermal Source Network Model". In: *Proceedings of the 16th International Modelica & FMI Conference*, ed. D. Zimmer and U. C. Müller, Vol. 218, pp. 851–860. DOI: 10.3384/ecp218851.

Shan, K., S. Wang, D. Gao, et al. (2016). "Development and validation of an effective and robust chiller sequence control strategy using data-driven models". In: *Automation in Construction* 65, pp. 78–85. DOI: 10.1016/j.autcon.2016. 01.005.

Thangavelu, S. R., A. Myat, and A. Khambadkone (2017). "Energy optimization methodology of multi-chiller plant in commercial Buildings". In: *Energy* 123, pp. 64–76. DOI: 10.1016/j.energy.2017.01.116.

Wetter, M., Benne, K., Tummescheit, H., & Winther, C. (2024). Spawn: coupling Modelica Buildings Library and EnergyPlus to enable new energy system and control applications. Journal of Building Performance Simulation, 17(2), 274–292. DOI: 10.1080/19401493.2023.2266414.

Wetter, M., Zuo, W., Nouidui, T. S., & Pang, X. (2014). Modelica Buildings library. Journal of Building Performance Simulation, 7(4), 253–270. DOI: 10.1080/19401493.2013. 765506

Zhou, X., J. Yu, W. Zhang, et al. (2022). "A multi-objective optimization operation strategy for ice-storage air-conditioning system based on improved firefly algorithm". In: *Buildings Services Engineering Research & Technology* 43.2, pp. 161–178. DOI: 10.1177/01436244211045570.

Zou, W., Y. Sun, D. Gao, et al. (2023). "Robust enhancement of chiller sequencing control for tolerating sensor measurement uncertainties through controlling small-scale thermal energy storage". In: *Energy* 280, p. 128152. DOI: 10.1016/j.energy. 2023.128152.

An Equation-Oriented Modeling Framework for Thermodynamic Cycle Systems toward MBSE: A Case Study on Supercritical CO_2 Brayton Cycles

Bai Xin-yu Xia Geng-lei * Zhou Tao Jiao Guang-hui

School of Nuclear Science and Technology, Harbin Engineering University, China, @xiagenglei@163.com

Abstract

Complex thermodynamic cycle systems play a critical role in advanced energy applications, where modeling and simulation constitute a fundamental component of Model-Based Systems Engineering (MBSE). Conventional sequential modular approaches rely heavily on predefined calculation sequences and empirical parameter settings, leading to limited adaptability for complex cycle topologies with strong coupling, flow splitting, and multiple recuperation processes.

This study proposes an equation-oriented (EO) modeling framework for thermodynamic cycle systems, in which the entire cycle is formulated as a coupled nonlinear algebraic equation system and solved simultaneously. Standardized algebraic component models are established for compressors, turbines, and recuperators. An ε–NTU-based heat exchanger formulation is adopted to avoid the predefined minimum temperature difference commonly required in conventional approaches. In addition, a bi-level solution strategy combining outer-layer parameter scanning and inner-layer Newton iterations is introduced to handle additional design degrees of freedom.

The proposed framework is validated using supercritical CO_2 simple recuperated and recompression Brayton cycles. Simulation results show that the predicted cycle efficiencies agree well with published reference data, with deviations below 1% under all investigated operating conditions. The framework demonstrates stable convergence behavior and strong adaptability to complex thermodynamic topologies.

The proposed approach exhibits acausal, modular, and topology-decoupled characteristics consistent with the declarative modeling philosophy of Modelica. It provides a unified and extensible solution framework for MBSE-oriented modeling and simulation of complex thermodynamic energy systems.

Keywords: Thermodynamic cycle systems, Equation-oriented method, Supercritical CO_2 Brayton cycle, ε–NTU method, MBSE

1 Introduction

Thermodynamic cycle systems are widely employed in advanced energy applications, including nuclear power systems, concentrated solar power generation, and industrial waste heat recovery systems(Ahn et al. 2015; Dostal 2004; Driscoll and Hejzlar 2008). With the increasing complexity of advanced cycle configurations involving high operating parameters, multiple recuperators, and strongly coupled components, system-level modeling and simulation have become increasingly important for performance evaluation,design optimization, and system integration.

At present, thermodynamic cycle simulations are predominantly based on the Sequential Modular Approach (SMA), in which component models are solved sequentially following the working-fluid flow path(Carstens 2007). Although straightforward for relatively simple cycles, this approach suffers from several limitations when applied to complex thermodynamic systems. First, the computational sequence strongly depends on the cycle topology, requiring substantial restructuring of the solution procedure when the configuration changes. Second, multi-recuperation, flow-splitting, and mixing structures often introduce nested iterative loops, significantly increasing implementation complexity(Fleming et al. 2012; Pasch et al. 2012). Third, empirical parameters such as minimum temperature difference in heat exchangers are typically predefined, limiting model reusability and automated design optimization capabilities.

Equation-oriented (EO) methods provide an alternative modeling paradigm for addressing these challenges(Chen 2019). In EO formulations, the entire thermodynamic system is represented as a unified set of implicit nonlinear algebraic equations, where all component governing equations and conservation constraints are solved simultaneously without explicitly specifying the computational sequence. Compared with sequential approaches, EO methods are naturally suited for strongly coupled systems with complex topologies and multiple recuperation processes.

The modeling philosophy of EO methods is inherently consistent with the declarative modeling paradigm adopted in MBSE-oriented Modelica environments(Fritzson 2014; Modelica Association 2023). Both approaches employ acausal modeling principles, where only physical constraints are defined without prescribing explicit input–output causality. System equations are generated from component interconnections rather than manually arranged execution sequences. In addition, both frameworks emphasize modular component modeling and equation-variable balancing principles. In EO formulations, maintaining consistency between the number of equations and unknown variables corresponds closely to the balanced model principle in Modelica, which provides a theoretical basis for automated system assembly and unified solution procedures.

Despite the conceptual alignment between EO methods and MBSE principles, the practical implementation of MBSE in thermodynamic cycle systems still faces several challenges. First, the highly nonlinear variations of thermophysical properties near the critical point impose stringent requirements on solver robustness. Second, large-scale nonlinear equation systems automatically generated from complex topologies often lead to convergence difficulties without carefully designed initialization strategies. Third, the coexistence of state variables and design variables in thermodynamic systems requires a clear separation mechanism to avoid underdetermined or ill-conditioned system configurations. These practical difficulties have limited the systematic application of declarative modeling approaches in this domain.

Existing studies on EO methods have mainly focused on process optimization and performance analysis of specific energy systems(Chen 2019; Dyreby et al. 2014). However, research on modular EO modeling frameworks for thermodynamic cycle systems remains relatively limited. In particular, for advanced supercritical CO_2 Brayton cycles with recompression and multi-recuperation structures, most existing studies rely on sequential approaches or direct implementation using commercial simulation platforms(Crespi et al. 2017; Kulhánek and Dostal 2011), while systematic discussions on the EO modeling methodology itself are still scarce.

Therefore, this work develops an EO-based modeling framework for thermodynamic cycle systems and validates it using supercritical CO_2 simple recuperated and recompression Brayton cycles. The study focuses on modular component equation construction, equation assembly for complex topologies, and treatment of additional design degrees of freedom, providing a reference framework for MBSE-oriented thermodynamic system modeling and simulation.

2 Equation-Oriented Modeling Method

2.1 Fundamental Formulation and Solution Strategy

The core concept of the Equation-Oriented (EO) method is to represent the entire thermodynamic cycle as a unified nonlinear algebraic system by simultaneously coupling component governing equations with mass and energy conservation constraints. All state variables are solved in a fully coupled manner within a single solution framework(Chen 2019).

Mass and energy conservation relations must be satisfied at all splitting, mixing, and junction nodes throughout the cycle system:

$$\sum \dot{m}_{\text{in}} = \sum \dot{m}_{\text{out}} \tag{1}$$

$$\sum (\dot{m}h)_{\text{in}} = \sum (\dot{m}h)_{\text{out}} \tag{2}$$

For a thermodynamic system containing n independent unknown variables, the EO model can be generally expressed as:

$$F(x) = 0 \tag{3}$$

$$x = [x_1, x_2, \ldots, x_n]^{\mathrm{T}} \tag{4}$$

where x denotes the vector of unknown variables, primarily including key state-point enthalpies and the total mass flow rate in the present study. $F(x)$ represents the residual vector, whose individual components correspond to component process equations, heat transfer constraints, and node conservation equations. The system is considered converged when all residuals approach zero and all physical constraints are simultaneously satisfied.

A Newton–Raphson method with backtracking line search is employed to solve the nonlinear equation system. The solution procedure is summarized as follows:

1. Initialize the iteration vector $x^{(k)}$ and evaluate the residual vector $F(x^{(k)})$;
2. Check convergence. The iteration terminates when the infinity norm of the residual satisfies:
$$\| F(x^{(k)}) \|_{\infty} < 10^{-6} \tag{5}$$
3. Construct the Jacobian matrix J using a central-difference approximation:
$$J_{ij} = \frac{F_i(x + \delta e_j) - F_i(x - \delta e_j)}{2\delta} \tag{6}$$
where e_j denotes the unit basis vector. The perturbation step size is selected as: $\delta = 10^{-6} \cdot max\,(|x_j|, 1.0)$ to balance numerical accuracy and stability.
4. Solve the linear correction equation to obtain the Newton iteration step:
$$J \cdot \Delta x = -F(x) \tag{7}$$

To avoid divergence or thermophysical property violations caused by excessively large iteration steps, a

DOI
10.3384/ecp223185

backtracking line-search strategy is introduced. The initial step-size factor is set to $\alpha=1.0$. If the residual reduction criterion is not satisfied, the step size is progressively reduced according to $\alpha=0.5,0.25,\ldots$ until the following condition is satisfied:

$$\| F(x^{(k)}+\alpha\Delta x)\|_{\infty} < \| F(x^{(k)})\|_{\infty} \quad (8)$$

In addition, equation-variable consistency is strictly maintained throughout the EO formulation to ensure structural solvability and avoid singular system configurations.

2.2 Component Model Formulation

Using the supercritical CO_2 Brayton cycle as an example, all components in the EO framework are described using standardized algebraic equations. Each component model only defines its own physical constraints and remains independent of the overall cycle topology, enabling flexible reuse under different system configurations.

2.2.1 Compressor and Turbine Models

Both compressors and turbines are modeled using conventional isentropic-efficiency formulations, which are widely adopted for representing irreversible losses in thermodynamic cycles(Cengel and Boles 2019).

For the compressor, the isentropic outlet enthalpy is first determined from the inlet state and outlet pressure using thermophysical property relations. Considering irreversible compression losses, the actual outlet enthalpy is calculated by:

$$h_{\text{out}} = h_{\text{in}} + \frac{h_{\text{out},s} - h_{\text{in}}}{\eta_c} \quad (9)$$

where η_c denotes the compressor isentropic efficiency.

Similarly, for the turbine, the actual outlet enthalpy is obtained from the isentropic expansion process:

$$h_{\text{out}} = h_{\text{in}} - \eta_t \left(h_{\text{in}} - h_{\text{out},s} \right) \quad (10)$$

where η_t represents the turbine isentropic efficiency. Both efficiencies are treated as prescribed design parameters characterizing component performance.

2.2.2 Recuperator ε–NTU Heat Transfer Model

The recuperator is modeled using the ε–NTU effectiveness method(Cengel and Boles 2019). Compared with conventional approaches based on predefined minimum temperature differences, the ε–NTU formulation is more naturally compatible with EO-based declarative equation assembly.

The heat transfer relation of the recuperator is expressed as:

$$Q = \varepsilon \cdot C_{\min} \cdot (T_{\text{hot,in}} - T_{\text{cold,in}}) \quad (11)$$

where ε is the heat exchanger effectiveness, C_{min} denotes the minimum heat capacity rate between the hot and cold streams, and $T_{h,in}$ and $T_{c,in}$ are the inlet temperatures of the hot and cold sides, respectively.

Energy conservation equations are simultaneously imposed on both sides of the recuperator. The heat release equation for the hot side is:

$$Q = \dot{m}_{\text{hot}} (h_{\text{hot,in}} - h_{\text{hot,out}}) \quad (12)$$

The heat absorption equation for the cold side is:

$$Q = \dot{m}_{\text{cold}} (h_{\text{cold,out}} - h_{\text{cold,in}}) \quad (13)$$

Under known inlet states and mass flow conditions, the heat transfer rate and outlet thermodynamic states can be solved simultaneously.

2.2.3 Flow Splitting and Mixing Model for the Recompression Cycle

In the recompression Brayton cycle, the working fluid is divided downstream of the low-temperature recuperator, where one branch enters the main compressor and the other enters the recompressor(Dyreby et al. 2014). The split ratio is defined as:

$$\dot{m}_{\text{rc}} = \text{split} \cdot \dot{m} \quad (14)$$

$$\dot{m}_{\text{mc}} = (1-\text{split}) \cdot \dot{m} \quad (15)$$

where $\dot{m}_{rc}$ and $\dot{m}_{mc}$ denote the mass flow rates through the recompressor branch and main compressor branch, respectively, and $\dot{m}$ represents the total cycle mass flow rate.

The enthalpy at the mixing point is determined by the energy conservation relation:

$$h_{\text{mix}} = (1-\text{split})h_3 + \text{split} \cdot h_9 \quad (16)$$

The split ratio represents a system-level design degree of freedom rather than a state variable determined by conservation equations. If the split ratio were included as an additional unknown in the inner Newton iteration, the resulting system would be underdetermined—with more unknowns than equations — unless an additional constraint equation is introduced. In the conceptual design phase, such additional constraints are typically not yet specified; instead, the designer is more interested in determining which split ratio yields the maximum cycle efficiency for a given set of component performance parameters. Therefore, the split ratio is treated as an outer-layer design variable optimized through parametric scanning, while the inner-layer Newton iteration solves only the thermodynamic state variables. This bi-level strategy decouples design variable selection from state variable solution, providing a systematic approach for handling design degrees of freedom in the early design stage. It should be emphasized that this treatment does not represent a limitation of the EO formulation, but rather a design-oriented strategy for handling optimization variables. If additional constraints become available in later design phases, the split ratio can also be directly incorporated into the inner-layer solution within the same framework.

2.2.4 Component Model Fidelity and Extensibility

It should be noted that the constant isentropic efficiency model and constant heat exchanger effectiveness model adopted in this study represent an appropriate level of model fidelity for the conceptual design phase of thermodynamic systems. During this phase, the system topology and key parameters are yet to be finalized, and simplified component models enable rapid performance evaluation and comparison among different cycle configurations. When entering the detailed design phase, these models can be replaced by higher-fidelity component characteristic models without modifying the system-level equation assembly process. For example, turbine efficiency can be expressed as a function of the pressure ratio and rotational speed, and heat exchanger effectiveness can be expressed as a function of NTU and flow arrangement. Owing to the modular nature of the EO framework, such replacements only require modifications within the corresponding device classes, while the topology-driven automatic equation assembly process remains unchanged.

2.3 Numerical Stability and Convergence Strategy

The EO equation system for thermodynamic cycles is characterized by strong nonlinearity and tight coupling. Near the supercritical region, thermophysical properties of CO_2 exhibit steep gradients(Brun et al. 2017), which may lead to iteration divergence or thermodynamic property evaluation failures.

To improve numerical robustness and convergence stability, the following strategies are adopted:

1. Initial guesses are assigned based on the thermodynamic characteristics of each component: the compressor outlet enthalpy is initialized as the sum of the inlet enthalpy and a typical compression work increment; the turbine outlet enthalpy is initialized as the inlet enthalpy minus a typical expansion work increment; and the recuperator outlet enthalpies are initialized near the midpoint between the respective inlet values. These estimates serve only as iteration starting points with clear physical interpretations, rather than arbitrary assignments. The sensitivity of the solution to initial guesses is further analyzed in Section 3.4. For multi-condition calculations, converged solutions from neighboring operating conditions are used as initial guesses to accelerate convergence;
2. A backtracking line-search strategy is employed to adaptively reduce the Newton step size, and iteration updates are accepted only when the residual norm decreases;
3. Boundary protections are imposed on temperature, enthalpy, and specific heat capacity. When abnormal thermophysical property evaluations occur, the solution automatically rolls back to the previous valid state and reduces the iteration step size.

These treatments significantly improve the numerical stability of the EO framework under supercritical operating conditions.

3 Validation Using Supercritical CO_2 Brayton Cycles

To validate the applicability of the proposed EO framework to complex thermodynamic systems, supercritical CO_2 simple recuperated and recompression Brayton cycles are selected as representative case studies (Dostal 2004; Dyreby et al. 2014). Both cycles involve compressors, turbines, recuperators, and strongly coupled heat transfer processes, while the recompression cycle additionally includes flow-splitting and mixing structures, providing an effective benchmark for evaluating the adaptability of the EO framework to complex topologies.

Both cycle models are constructed using the same component model library. Different system configurations are established solely through topology-dependent equation assembly without modifying individual component models, demonstrating the modularity and topology scalability of the EO framework.

3.1 EO Model of the Simple Recuperated Cycle

The simple supercritical CO_2 recuperated Brayton cycle consists of a main compressor, recuperator, reactor core, turbine, and precooler. The state-point numbering scheme is illustrated in Fig. 1.

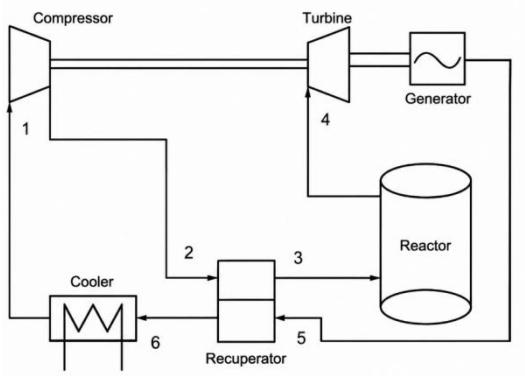


Fig. 1. Configuration and state-point numbering of the supercritical CO_2 simple recuperated Brayton cycle

The unknown variable vector is selected as:

$$x = [h_2, h_3, h_5, h_6, \dot{m}]^T \tag{17}$$

By combining the compressor, turbine, and ε–NTU recuperator models introduced in Section 2.2 with node conservation relations, the following closed EO equation system is obtained:

F_1(compressor equation):

$$h_2 = h_1 + \frac{h_{2s} - h_1}{\eta_c} \tag{18}$$

DOI 10.3384/ecp223185

F_2(turbine equation) :

$$h_5 = h_4 - \eta_t (h_4 - h_{5s}) \tag{19}$$

F_3(cold-side recuperator equation) :

$$\dot{m}(h_3 - h_2) = \varepsilon C_{\min}(T_5 - T_2) \tag{20}$$

F_4(hot-side recuperator equation) :

$$\dot{m}(h_3 - h_2) = \varepsilon C_{\min}(T_5 - T_2) \tag{21}$$

F_5(reactor-core heat balance equation) :

$$\dot{m}(h_4 - h_3) = Q_{\text{in}} \tag{22}$$

These equations are coupled through heat-transfer rates and thermodynamic state variables, enabling simultaneous system-level solution. In particular, F_3 and F_4 are connected through the same recuperator heat-transfer rate, reflecting the fully coupled nature of the EO framework.

3.2 EO Model of the Recompression Cycle

The recompression Brayton cycle extends the simple recuperated cycle by introducing a recompressor, a low-temperature recuperator (LTR), a high-temperature recuperator (HTR), and additional flow-splitting and mixing structures, as illustrated in Fig. 2.

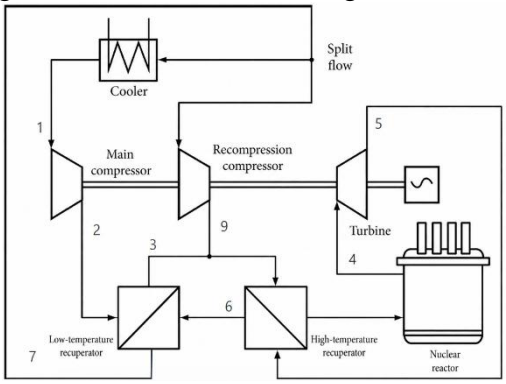


Fig. 2. Configuration and state-point numbering of the supercritical CO_2 recompression Brayton cycle

The unknown variable vector is selected as:

$$x = [h_2, h_3, h_5, h_6, h_7, h_9, \dot{m}]^{\mathsf{T}} \tag{23}$$

Using the modular component equations and node conservation relations described in Section 2.2, the EO equation system is assembled as follows:

F_1(main compressor equation):

$$h_2 = h_1 + \frac{h_{2s} - h_1}{\eta_c} \tag{24}$$

F_2(recompressor equation):

$$h_9 = h_7 + \frac{h_{9s} - h_7}{\eta_c} \tag{25}$$

F_3(turbine equation):

$$h_5 = h_4 - \eta_t (h_4 - h_{5s}) \tag{26}$$

F_4(LTR cold-side equation):

$$\dot{m}_{\text{mc}}(h_3 - h_2) = Q_{\text{LTR}} \tag{27}$$

F_5(LTR hot-side equation):

$$\dot{m}(h_6 - h_7) = Q_{\text{LTR}} \tag{28}$$

F_6(HTR hot-side equation):

$$\dot{m}(h_5 - h_6) = Q_{\text{HTR}} \tag{29}$$

F_7(reactor-core and HTR cold-side coupled equation):

$$\dot{m}(h_4 - h_{\text{mix}}) = Q_{\text{HTR}} + Q_{\text{core}} \tag{30}$$

The mixing-point enthalpy is determined from the energy conservation relation, while the heat transfer rates Q_{LTR} and Q_{HTR} are evaluated using the ε–NTU formulations presented in Section 2.2.2.

The split ratio represents an additional design variable that cannot be directly determined by the seven thermodynamic state equations. Therefore, a bi-level solution strategy is employed, as illustrated in Fig. 3. In the outer layer, the split ratio is scanned within a prescribed range. In the inner layer, the EO equation system is solved using Newton iterations under each split-ratio condition. The optimal split ratio is finally determined based on the maximum cycle efficiency criterion.

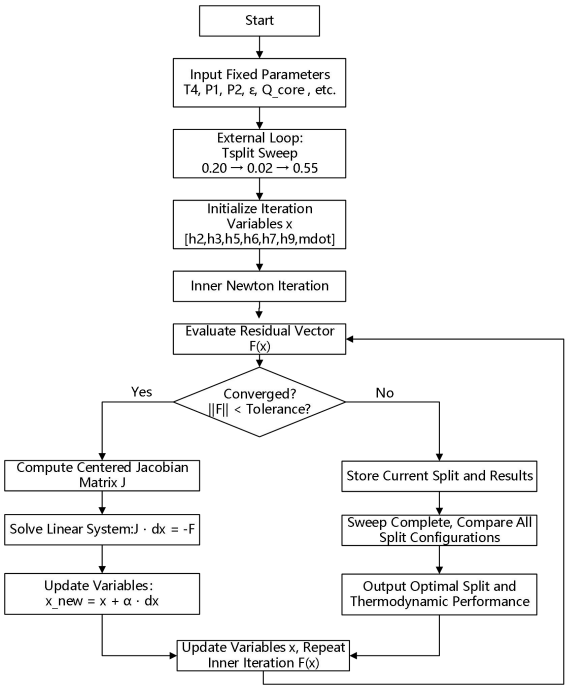


Fig. 3. Bi-level solution strategy of the equation-oriented framework

3.3 Validation of the EO Framework

Under the reference operating condition ($T_{max} = 600\,°C$, $T_1 = 32\,°C$, $P_2 = 25\,MPa$, $P_1 = 7.7\,MPa$, $Q_{core} = 20\,MW$), the EO simulation results are compared with published reference data reported by Dostal et al.

The results indicate that the predicted efficiencies of both the simple recuperated and recompression cycles agree well with published reference values. For the recompression cycle at $T_{max} = 600\,°C$, the predicted thermal efficiency reaches 48.09%, which falls within the 46%–48% range reported in the literature.

All investigated operating conditions achieve stable convergence, and the energy imbalance between the hot and cold sides of the recuperators remains below 0.2%.

The convergence behavior of the EO framework is also evaluated. Under the reference operating condition, the simple recuperated cycle model converges within 20 to 30 Newton iterations to a residual below 10^{-6}. The recompression cycle model, owing to the bi-level solution strategy, achieves stable convergence at each discretized split ratio value. In conventional sequential modular implementations, such strongly coupled structures are usually handled through nested iterations between individual components. The EO formulation avoids explicitly designed iteration loops by solving all component equations simultaneously, which helps reduce the implementation complexity for complex cycle topologies.

These results demonstrate that the proposed EO framework can effectively handle strongly coupled heat-transfer processes and flow-splitting structures in complex thermodynamic cycles. Compared with conventional sequential approaches, the proposed method eliminates the need for predefined heat-exchanger terminal temperature differences while simultaneously satisfying all conservation equations during each iteration step. In addition, the framework exhibits strong topology adaptability and component reusability.

Table 1 Comparison between present results and published reference data

	Initial Settings			Calculated Parameters		
Cycle Type	T_{max} (°C)	P_{max} (MPa)	ε	Literature Efficiency (Source)	Present Efficiency	Deviation
Simple	650	15.4	0.90	27%~30% (Dostal 2004)	28.08%	Within range
Recompression	550	25.0	0.95	45%~46% (Dostal 2004)	45.39%	<1%
Recompression	600	25.0	0.95	46%~48% (Dostal 2004)	48.09%	<1%

3.4 Parameter Sensitivity and Convergence Analysis

3.4.1 Split Ratio Sensitivity

The bi-level solution strategy with outer-layer split ratio scanning is a distinctive feature of the proposed EO framework. To evaluate the effectiveness of this strategy, the variation of the recompression cycle efficiency with the split ratio is analyzed under the reference operating condition. The split ratio is scanned within a predefined feasible range, and the corresponding cycle efficiency is evaluated for each candidate value.

The efficiency variation exhibits a clear single-peak pattern, with the optimal split ratio located at approximately 0.48. The efficiency increases gradually as the split ratio approaches the optimum and decreases as it deviates, indicating a relatively flat optimum region that is favorable for practical engineering design. This characteristic suggests that near-optimal performance can be achieved even without precise split ratio control. The outer-layer scanning strategy successfully captures this design optimum without requiring additional closure equations, demonstrating its effectiveness for handling design degrees of freedom in the conceptual design phase.

3.4.2 Numerical Robustness with Respect to Initial Guesses

The numerical robustness of the EO framework is evaluated by perturbing the initial guesses of the unknown variables. For the simple recuperated cycle, the initial values of the state variables were randomly perturbed within $\pm$ 20% of their baseline estimates. All tested perturbation cases converged stably to the same solution within approximately 30 Newton iterations, indicating that the final solution is insensitive to the choice of initial guesses. This robustness is attributed to the backtracking line-search strategy and the thermophysical property protection mechanisms introduced in Section 2.3.

4 Discussion on MBSE-Oriented Modeling Compatibility

The proposed EO framework exhibits strong consistency with the declarative modeling philosophy adopted in MBSE-oriented Modelica environments(Fritzson 2014; Modelica Association 2023).

First, each component model only defines its own physical constraint equations without explicitly specifying input–output causality. System-level coupling is established through thermodynamic state variables, which is consistent with the acausal connection mechanism in Modelica(Fritzson 2014).

DOI 10.3384/ecp223185

Second, the EO framework automatically assembles system equations according to component interconnections rather than manually defining the execution sequence of individual devices. Component models describe only constitutive physical relations, while the system-level topology determines the final equation assembly. This structure provides strong modularity and topology-decoupling capability(Modelica Association 2023).

In addition, the EO framework strictly maintains consistency between the number of equations and unknown variables during system solution, corresponding to the balanced model principle in Modelica(Fritzson 2014; Modelica Association 2023). For additional design degrees of freedom such as the split ratio in recompression cycles, outer-layer optimization and inner-layer state solution are decoupled to avoid structural singularities.

Therefore, the proposed EO framework provides a potential foundation for MBSE-oriented unified modeling of thermodynamic systems. Future work may further extend the component model library and integrate the framework with Modelica platforms such as OpenModelica and Dymola to realize automated topology generation and co-simulation of complex thermodynamic systems.

5 Conclusions

This study proposes an equation-oriented modeling framework for thermodynamic cycle systems and validates the framework using supercritical CO_2 simple recuperated and recompression Brayton cycles as representative examples.

The main conclusions are summarized as follows:

1. The EO framework achieves unified modeling of different thermodynamic cycle topologies through modular component equations and fully coupled implicit solution procedures. Component equations remain decoupled from the system topology, enabling model reuse across different cycle configurations.
2. The $\varepsilon - NTU$ heat exchanger formulation replaces the predefined minimum temperature difference commonly adopted in sequential approaches, allowing heat-transfer processes to be determined self-consistently from physical equations, thereby improving topology adaptability and design-variable handling capability.
3. For the split-ratio design degree of freedom in the recompression cycle, a bi-level strategy combining outer-layer scanning and inner-layer Newton iterations is proposed. This approach decouples design variable selection from state variable solution, providing a systematic method for handling design degrees of freedom without introducing structural singularities.
4. Comparisons with published reference data demonstrate that the proposed EO framework can accurately predict the performance of supercritical CO_2 Brayton cycles, with efficiency deviations below 1% and stable convergence achieved under all investigated operating conditions.

The proposed framework exhibits acausal, modular, and topology-decoupled characteristics consistent with the declarative modeling philosophy of Modelica. It provides a unified and extensible solution path for MBSE-oriented modeling and simulation of complex energy systems.

References

Ahn, Y., S. J. Bae, M. Kim, et al. (2015). "Review of supercritical CO_2 power cycle technology and current status of research and development." Nuclear Engineering and Technology 47(6): 647–661.

Brun, K., P. Friedman, and R. Dennis (2017). Fundamentals and Applications of Supercritical Carbon Dioxide (sCO_2) Based Power Cycles. Cambridge: Woodhead Publishing.

Carstens, N. A. (2007). Control Strategies for Supercritical Carbon Dioxide Power Conversion Systems. Doctoral dissertation. Cambridge: Massachusetts Institute of Technology.

Cengel, Y. A. and M. A. Boles (2019). Thermodynamics: An Engineering Approach. 9th ed. New York: McGraw-Hill.

Chen, Y. (2019). Equation-Oriented Optimization of Energy Systems: Methodologies and Applications. Amsterdam: Elsevier.

Crespi, F., G. Gavagnin, D. Sánchez, et al. (2017). "Supercritical carbon dioxide cycles for power generation: A review." Applied Energy 195: 152–183.

Dostal, V. (2004). A Supercritical Carbon Dioxide Cycle for Next Generation Nuclear Reactors. Doctoral dissertation. Cambridge: Massachusetts Institute of Technology.

Driscoll, M. J. and P. Hejzlar (2008). "Supercritical CO_2 Brayton cycles for nuclear power applications." ASME Turbo Expo 2008: Power for Land, Sea, and Air. Berlin: ASME, pp. 457–465.

Dyreby, J., S. Klein, G. Nellis, et al. (2014). "Design considerations for supercritical carbon dioxide Brayton cycles with recompression." Journal of Engineering for Gas Turbines and Power 136(10): 101702.

Fleming, D., T. Holschuh, T. Conboy, et al. (2012). "Scaling considerations for a multi-megawatt class supercritical CO_2 Brayton cycle and path forward for commercialization." ASME Turbo Expo 2012. Copenhagen: ASME, GT2012-68484.

Fritzson, P. (2014). Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A

Cyber-Physical Approach. 2nd ed. Hoboken: Wiley-IEEE Press.

Kulhánek, M. and V. Dostal (2011). "Supercritical CO_2 cycles for nuclear power plants." Supercritical CO_2 Power Cycle Symposium. Boulder.

Modelica Association (2023). Modelica Language Specification, Version 3.6. Linköping: Modelica Association.

Pasch, J., T. Conboy, D. Fleming, et al. (2012). Supercritical CO_2 Recompression Brayton Cycle: Completed Assembly Description. SAND2012-9546. Albuquerque: Sandia National Laboratories.

DOI
10.3384/ecp223185

A Three-Level MBSE Logical Design Framework and Requirements-Traceability Validation for ICF Ultrafast Imaging Diagnostics Instruments

Liqiong Xia[1,2] Jinhua Zheng[1,2]* Youchen Li[1,2]
Huabing Du[1,2] Lin Hu[1,2] Jianhua Zheng[1,2]
Chaoxin Chen[1,2] Xing Zhang[1,2]

[1]Laser Fusion Research Center, China Academy of Engineering Physics, Mianyang 621900, China
[2]National Key Laboratory of Plasma Physics

Abstract

The ultrafast evolution of the physical process in the laser driven inertial confinement (ICF) fusion requires ultrafast imaging diagnostic instruments. The traditional commercial instruments are not able to meet the ICF experiment requirement. It is a technical challenge to develop the diagnostic instruments with an extreme capbility. This work proposes a three-layer MBSE logical design framework for an ultrafast framing imager. The framework comprises an operational analysis layer, a system design layer, and a component design layer. Using SysML as the unified modeling language, it progressively decomposes the experimental objective of acquiring picosecond-scale plasma-evolution images into system indicators, functional nodes, logical components, and module-level parameters.

Methodologically, the study establishes a bidirectional requirements-traceability mechanism and introduces a parameter-budget model for temporal-resolution allocation. Taking system temporal resolution <= 80 ps as the representative case, a complete decomposition chain is constructed from the user requirement down to trigger-synchronization jitter, gating-pulse jitter, and MCP response time. Process-based comparative analysis further confirms the effectiveness of the framework in requirement allocation, change-impact localization, and cross-disciplinary collaboration.

The case study shows that, at the same problem scale, the MBSE approach reduces the time required for requirement-to-module allocation from approximately 6.4 h to 2.1 h, and shortens the impact-localization time for tightening the target from 0 ps to 50 ps from 47 min to 11 min, while significantly reducing omissions. Model statistics indicate that the case includes 5 user requirements, 8 system indicators, 17 functional nodes, 12 key interfaces, and 46 traceability links, demonstrating not only conceptual value but also maintainability at a practical engineering scale. The proposed method therefore provides a useful foundation for the systematic design, rapid iteration, and subsequent automated verification of laser-fusion diagnostic equipment.

Keywords: MBSE; SysML; laser fusion; ultrafast framing imager; requirements traceability; change-impact analysis; system architecture.

1 Introduction

In the laser driven inertial confinement fusion (ICF), the interaction time of laser and target is in only several ns, the duration of the fusion plasma is about 100ps. It is very important and necessary to observe the target evolution and diagnose the plasma parameters. The ultrafast imaging diagnostics instruments, for example, the X-ray framing camera, is indispensable in the ICF experiments. Their performance directly determines the ability to resolve transient physical processes. Considering the systematic exposition of indirect-drive ICF target physics,the X-ray framing camera is required to achieve a time resolotion of less than 100ps. Previous studies consistently indicate that temporal resolution, gated-shutter performance, and synchronous triggering capability are the fundamental performance determinants of ultrafast diagnostic equipment [1].

Related research has also continued to advance in China. Early efforts investigated the structure and operating principles of MCP-gated picosecond X-ray framing cameras, and recent studies have further extended this work to the design and simulation of ultrafast four-frame CMOS circuits for ICF diagnostics, reflecting the evolution of diagnostic chains from MCP gating toward higher integration and single-line-of-sight imaging [3-4]. Meanwhile, MBSE research in China has progressed from methodological introduction to application in complex engineering systems. For example, SysML-based MBSE methods have been applied to the logical design of complete trainsets, demonstrating the engineering feasibility of model-driven approaches for cross-disciplinary collaboration and hierarchical modeling of complex systems [4].

Nevertheless, most existing studies, both domestic and international, focus either on diagnostic-device development or on generic MBSE methodologies. The former primarily aims to improve spatiotemporal resolution and hardware realization [2,3], whereas the latter concentrates on requirements, behavior, structure, parametric modeling, and traceability consistency, thereby forming a knowledge base rooted in SysML/UML specifications, methodological guidance, traceability theory, and industrial adoption studies [5-13]. For scientific instruments such as ICF ultrafast imaging systems, which are simultaneously characterized by stringent timing constraints, multidisciplinary interfaces, and rapid change propagation, a lightweight logical design framework is still lacking that can robustly map high-level experimental requirements to module parameters and directly support rapid localization of change impacts [5,11-13].

From the perspective of model-based implementation continuity, Modelica provides a useful complement to the proposed three-layer MBSE framework. Where SysML is effective for organizing requirements, functions, interfaces, and logical allocation, Modelica is better suited for constructing executable multi-domain models that capture the coupled dynamic behavior of control, pulse-generation, detection, and timing-transfer links. For an ICF ultrafast imaging instrument, temporal resolution is affected not only by static parameter assignment but also by transient interactions among trigger distribution, gate-drive response, circuit parasitics, and detector-side conversion processes. Introducing Modelica-oriented behavioral refinement after the logical-design stage would therefore make it possible to verify whether the parameter-budget assumptions embedded in the traceability chain remain valid under dynamic operating conditions, and would provide a clearer bridge from system logic to simulation-supported design decisions[14-17].

Against this background, the present work proposes a three-layer MBSE logical design framework for an ICF ultrafast framing imager and reinforces the engineering argument through a requirements-traceability chain and process-oriented validation. The objective is not to replace detailed optical or electronic design, but to provide a verifiable logical backbone from experimental requirements to module parameters. Specifically, the study: tailors a generic multi-layer MBSE methodology into a three-layer architecture for the ICF scenario; establishes a bidirectional traceability mechanism from user requirements to module parameters and introduces a temporal-resolution budget model; and validates the framework through case-based process evaluation, change analysis, and complexity statistics, thereby closing the loop with the concluding claims of higher efficiency, lower omission rates, and support for subsequent automated verification.

2 Related Work and Problem Definition

2.1 Overview of Mainstream MBSE Methods

Internationally, several mature MBSE methods and engineering workflows have been established, including Harmony SE, CESAM, Arcadia, and MagicGrid. These methods provide relatively complete support in execution logic, architectural coordination, hierarchical decomposition, and consistency control [15-18]. Meanwhile, SysML/UML specifications and their engineering guides offer a common basis for the unified expression of requirements, behavior, structure, and parameters, while studies on requirements traceability and industrial adoption further demonstrate the value of MBSE in complex-system collaboration, impact analysis, and consistency management [6-14].

Table 1 Comparison of mainstream MBSE methods and the adaptation features of the proposed framework

Method	*Typical layering*	*Modeling focus*	*Traceability depth*	*Adaptability to ICF ultrafast diagnostics*
OOSEM	Requirements-use cases-architecture	Object-oriented decomposition	Medium	Suitable for functional organization, but insufficient for a picosecond-level parameter closed loop.
Harmony SE	Requirements-state-implementation	Execution logic and state machines	Medium	Suitable for control logic, but limited in cross-domain optical-electrical-mechanical parameter mapping.
CESAM	Mission-capability-system-subsystem	Enterprise-level architecture	Medium	Suitable for large-scale industrial collaboration, but relatively coarse-grained.
Arcadia	OA-SA-LA-PA	From operational scenarios to architecture	High	Clear layering and suitable as a parent method.

DOI
10.3384/ecp223193

MagicGrid	Problem-solution-implementation multi-grid	White-box analysis and consistency	High	Suitable for unified modeling and intra-layer consistency control.
Proposed framework	OA-SD-CD three-layer architecture	Closed loop from requirements to module parameters	High	Tailored to ICF timing chains and convenient for rapid iteration.

Overall, Arcadia and MagicGrid are the closest to the modeling objectives of this work. Arcadia emphasizes progressive decomposition from the operational environment to the architectural layer, whereas MagicGrid highlights white-box analysis and model-consistency control [15,18]. However, directly adopting these methods still cannot fully satisfy the engineering demand of the ICF ultrafast-diagnostics scenario for a vertically closed mapping from extreme physical constraints to system functional chains and further to module parameters. Tailoring the method to the characteristics of the instrument timing chain, together with a clearer requirements-traceability and parameter-budget mechanism, therefore remains necessary [6,12,14].

2.2 Engineering Challenges of ICF Ultrafast Imaging Instruments

The engineering challenges of ICF ultrafast imaging instruments can be summarized in three aspects. First, the core indicators are strongly coupled. Overall temporal resolution is not determined by a single device, but jointly by the synchronization accuracy of the external master trigger, gating-pulse jitter, MCP response characteristics, and CMOS sampling timing. Second, the interfaces are inherently cross-disciplinary. The high-voltage gating chain, photoelectric conversion chain, and control-configuration chain belong to different teams; without a unified model, indicators can be specified without clearly assigned responsibility. Third, change propagation is rapid. When the experimental requirement is tightened from 100 ps to 80 ps, the impact is not confined to a single module, but propagates along functional-node, interface, and parameter chains.

2.3 Research Objectives and Contributions

In response to these issues, this study aims to establish a streamlined MBSE logical design framework for ICF ultrafast imaging diagnostic equipment and to verify its engineering effectiveness through an explicit traceability chain. The contributions are threefold: an OA-SD-CD three-layer architecture tailored to the ICF scenario to reduce the modeling burden associated with generic multi-layer methods; a bidirectional traceability mechanism from user requirements to module parameters, together with a temporal-resolution budget model; and case-based process evaluation with comparative results and complexity statistics, so that the conclusions are directly supported by method validation.

3 Three-Layer MBSE Logical Design Framework

3.1 Overall Framework Structure

A top-down three-layer logical architecture is adopted, consisting of the operational analysis (OA) layer, the system design (SD) layer, and the component design (CD) layer. The OA layer converts experimental tasks and observation objectives into system-level indicators; the SD layer transforms system indicators into functional activities and logical components; and the CD layer further translates logical components into module-level parameters, interface constraints, and verification conditions.

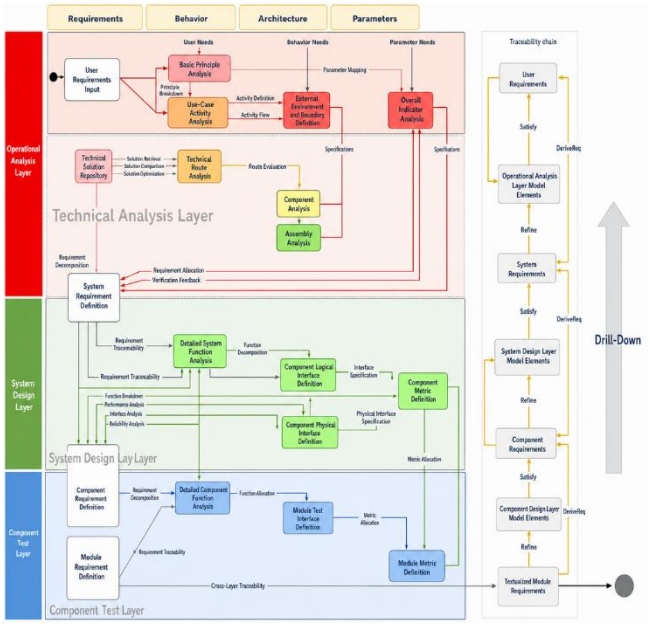


Fig. 1 Schematic of the three-layer MBSE logical design framework.

Compared with generic MBSE architectures containing five or more layers, the proposed three-layer framework is better suited to ultrafast diagnostic-instrument development, where requirements are dense, cross-disciplinary iteration is rapid, and implementation cycles are short. The purpose is not to sacrifice modeling rigor, but to compress the hierarchy while preserving the main traceability backbone, thereby making the model easier to maintain collaboratively by optical, electronic, and control teams.

3.2 Operational Analysis Layer

The operational analysis layer focuses on what tasks the system must accomplish within the experimental environment. In the present case, the top-level user requirement is defined as the acquisition of picosecond-scale plasma-evolution images. This layer further quantifies the requirement into system indicators, including temporal resolution, synchronization error below a specified threshold, coordination between gating and acquisition, compatibility with the facility master trigger, and operational safety. Its key output is therefore not a structural design, but a set of quantified constraints that can be accepted by downstream layers.

3.3 System Design Layer

The system design layer focuses on how the system satisfies indicator constraints through functional coordination. Around the temporal-resolution objective, the study identifies core functional nodes such as trigger reception, synchronization calibration, gating-pulse generation, MCP shutter control, CMOS external-trigger sampling, image buffering, and safe power-down. Activity diagrams play a central role at this layer because they not only describe functional sequencing, but also clarify control and data flows, enabling different disciplines to establish a shared understanding of the same timing chain.

3.4 Component Design Layer

The component design layer focuses on how logical components are mapped to module-level implementation constraints. In this work, the system is divided into three logical components: the control unit, the electronic pulse unit, and the optical detection unit. The control unit handles master-trigger reception, timing scheduling, and configuration distribution; the electronic pulse unit generates and shapes high-voltage gating pulses; and the optical detection unit coordinates MCP response with CMOS sampling. Through parametric-constraint modeling at this layer, system indicators can be further decomposed into concrete engineering quantities such as jitter, delay, bandwidth, response time, and interface consistency.

3.5 Cross-Layer Requirements Traceability and Time-Budget Allocation Model

To translate the high-level indicator of system temporal resolution <= 80 ps into component-level design constraints, a time-budget allocation model is introduced. By considering the combined effects of master-trigger synchronization, gating-pulse jitter, MCP response time, CMOS sampling-timing error, and environmental and calibration residuals, the system temporal resolution can be expressed in root-mean-square form as follows:

$$T_sys = sqrt(T_sync^2 + T_gate^2 + T_MCP^2 + T_cmos^2 + T_margin^2)$$

$$T_sync <= 25psT_gate <= 30psT_MCP <= 45psT_cmos <= 35psT_margin <= 20psT_sys71.9ps100psI$$

In the case study, conservative budgets are assigned to the contributing terms. After root-sum-square synthesis, the resulting value remains within the target limit and preserves a practical engineering margin. This budget model is not intended to replace detailed circuit or optical design, but rather to provide a unified top-level constraint boundary for cross-team collaboration.

Table 2 Hierarchical decomposition of the time-resolution requirement and parameter allocation

Level	*ID*	*Item/constraint*	*Responsible entity*
User requirement	UR-01	Acquire images of plasma evolution on the picosecond timescale	Experimental user / system integrator
System indicator	SI-01	System temporal resolution <= 100 ps	System integrator
Functional node	FN-01	Trigger reception and synchronization calibration	Control unit (CU)
Functional node	FN-02	Gating-pulse generation and shaping	Electronic pulse unit (EPU)
Functional node	FN-03	MCP shutter control	EPU + ODU
Functional node	FN-04	CMOS external-trigger sampling	Optical detection unit (ODU)
Module parameter	MI-01	Trigger-synchronization jitter <= 25 ps	CU
Module parameter	MI-02	Gating-pulse jitter <= 30 ps	EPU
Module parameter	MI-03	Effective MCP response time <= 45 ps	ODU

In addition to the time budget, this study also uses a traceability-coverage metric to evaluate model completeness. If the number of established valid traceability links is N_linked and the total number of objects requiring traceability is N_total, the traceability coverage is defined as follows:

$$C_t = N_linked/N_total$$

C_t When the metric approaches 1, the model has a relatively complete mapping among requirements,

DOI 10.3384/ecp223193

functions, structures, and parameters, which can significantly reduce the risk of broken design traceability.

4 Case Modeling of the Ultrafast Framing Imager

4.1 System Context and Service Use Cases

For the ICF ultrafast framing imager, the first modeling task is to define the system boundary explicitly and distinguish among the facility master-control system, the experimental target area, and the internal logic of the instrument. This step clarifies the responsibility boundary between external constraints and internal implementation at an early stage, thereby preventing ambiguity in subsequent interface definitions.

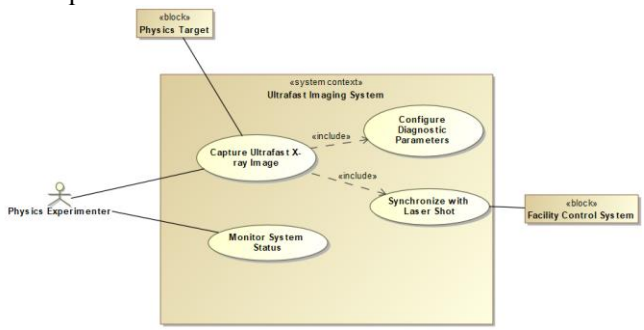


Fig. 2 System-context and service-use-case diagram.

At this layer, the core service use case is defined as acquiring ultrafast X-ray images. Its inputs include the facility master trigger, experimental configuration parameters, and operating-mode selection, whereas its outputs include image data, status information, and the required protection actions. The use case is directly associated with the system indicators in the operational analysis layer and therefore provides the constraint source for subsequent functional modeling.

4.2 Functional-activity Modeling and Logical Allocation

To ensure the determinacy of the timing chain, an activity diagram is used to describe the system behavior sequence in a typical experiment: trigger reception and synchronization calibration, gating-pulse generation, MCP shutter driving, CMOS external-trigger sampling, image buffering, and power-down protection. By placing control logic, electronic pulses, and optical detection in parallel through activity partitions (swimlanes), cross-disciplinary teams can intuitively identify functional ownership and interface interactions.

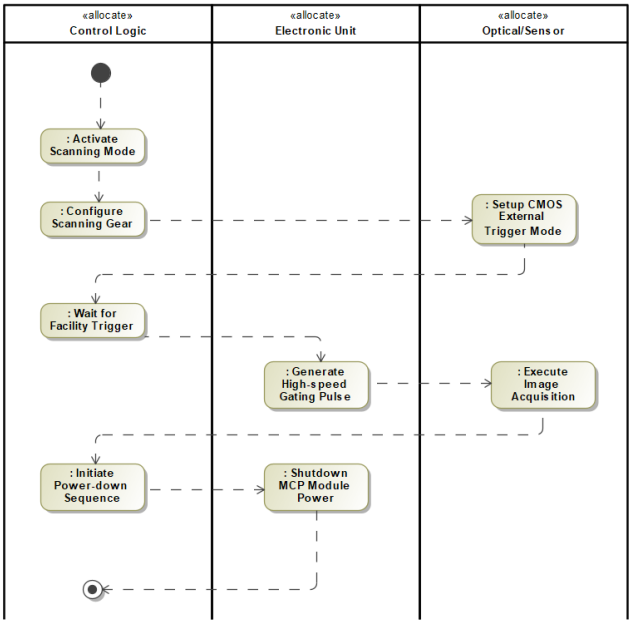


Fig. 3 System activity diagram.

The value of this model does not lie in replacing low-level implementation code, but in expressing transient timing logic in a unified manner from a systems-engineering perspective. For ultrafast imaging diagnostic systems, such unified expression can effectively reduce locally optimal but globally inconsistent situations such as control being satisfied while electronics are not, or electronics being satisfied while detection is mismatched.

4.3 Logical Components, Interfaces, and Parameter Constraints

Based on the activity-diagram results, the case is partitioned into three logical components: the control unit, the electronic pulse unit, and the optical detection unit. The control unit is responsible for receiving the master trigger, switching operating modes, and issuing configuration commands; the electronic pulse unit generates high-speed gating pulses and performs timing shaping; and the optical detection unit executes MCP control and CMOS sampling. The main internal interfaces include the master-trigger interface, the configuration bus, the high-voltage gating interface, the power-control interface, and the image-data interface.

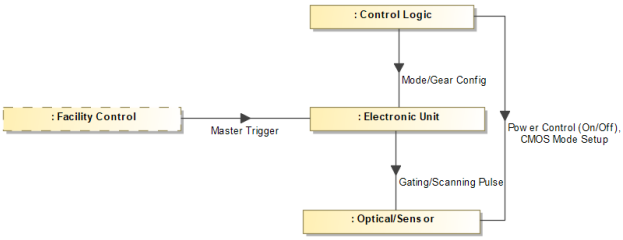


Fig. 4 Internal block diagram of logical components.

Table 3 Examples of logical components, interfaces, and key parameter constraints

Logical component	*Key interface*	*Core parameter*	*Constraint description*
Control unit (CU)	Master-trigger input, configuration bus	Trigger-synchronization error	Ensures consistency between the external reference and the internal timing sequence
Electronic pulse unit (EPU)	High-voltage gating interface, pulse-shaping interface	Pulse jitter, rise time	Determines the determinacy of the gating window
Optical detection unit (ODU)	MCP control interface, CMOS sampling interface	MCP response, sampling-timing error	Determines the effective imaging exposure and sampling synchronization

Through synchronized modeling of logical components and interfaces, the system no longer advances merely by stacking single-module indicators. Instead, interface boundaries explicitly bind component responsibilities, parameter constraints, and verification methods, thereby providing a unified basis for subsequent bidirectional traceability and integration testing.

4.4 Model-Building Principles and Consistency Rules

To ensure model maintainability, nine consistency rules are defined in the case, including the following: each system indicator shall be associated with at least one functional node; each key functional node shall be allocated to at least one logical component; each key interface shall correspond to both upstream and downstream responsible entities; and each module-level parameter shall be traceable to at least one system indicator. Combined later with SysML v2 or automated checking scripts, these rules can serve as the basis for semi-automated consistency verification.

5 5 Requirements-Traceability Validation and Results Analysis

5.1 Example of a Complete Requirements Traceability Chain

To verify the effectiveness of the proposed method, system temporal resolution <= 100 ps is selected as the representative core case for vertical expansion. The complete chain can be described as follows: user requirement UR-01 is to acquire images of picosecond-scale plasma evolution; system indicator SI-01 is system temporal resolution <= 100 ps; satisfying this indicator requires four key functions, namely trigger reception and synchronization calibration, gating-pulse generation, MCP shutter control, and CMOS external-trigger sampling; these functions are allocated to the control unit, electronic pulse unit, and optical detection unit; and parameter constraints such as MI-01 to MI-04 are finally established at the module layer.

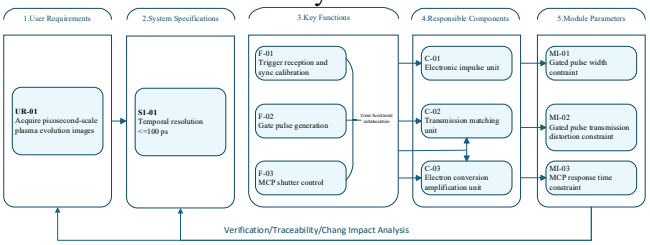


Fig. 5 Closed-loop schematic of the requirements traceability chain.

This chain indicates that temporal resolution is not a parameter of an isolated device, but the integrated outcome of functions, components, and interfaces. Only by explicitly binding functional nodes to responsible components can the implementation stage avoid broken traceability, in which requirements remain documented but fail to land on module parameters.

5.2 Change-Impact Analysis under Bidirectional Traceability

The value of bidirectional traceability lies not only in identifying the source, but also in supporting change. When the system temporal-resolution target is tightened from 100 ps to 80 ps, the model can immediately localize the affected objects, including the gating-pulse function, the synchronization-calibration function, key interfaces from the EPU to the ODU, and parameter constraints such as trigger jitter, pulse jitter, and sampling error. Compared with the traditional approach, engineers no longer need to reread the entire document set, but can identify the impact scope directly along the traceability chain.

Table 4 Example of change-impact analysis supported by bidirectional traceability

Change scenario	*Affected objects*	*Main adjustments*	*Recommended verification method*
SI-01: 100 ps -> 50 ps	FN-02、FN-03、FN-04,CU/EPU/ODU	Reallocate the jitter budget, narrow the gating window, and optimize synchronization calib-ration	Timing simulation +interface review + bench testing
MI-02 exceeds the threshold	The EPU and its high-voltage	Optimize pulse shaping and transmission-impedance matchi-ng	High-speed oscilloscope testing + parameter back-tracing

DOI
10.3384/ecp223193

	gating link to the ODU		
Master-trigger delay drift	CU, master-trigger interface, and synchronization strategy	Update the calibration strategy and trigger-compensation parameters	Hardware-in-the-loop validation

Reverse traceability is equally important. When a module parameter exceeds its design threshold during implementation, the engineering team can trace backward from the parameter item to the responsible component, functional node, and system indicator, thereby determining whether the deviation jeopardizes the overall temporal-resolution target. This closed-loop capability is a key reason why the proposed method outperforms traditional document-centric collaboration.

5.3 Validation Case of SysML-Modelica Timing-Deviation Analysis

Under the nominal budget, the synthesized temporal resolution of the instrument is maintained at about 92 ps, leaving only a limited engineering margin relative to the 100 ps requirement. During system integration, however, the measured response of the gating-pulse path shows an abnormal increase in equivalent jitter, accompanied by pulse-edge distortion and a small additional trigger-to-gate delay. The abnormality is observed after the electronic pulse unit has been coupled with the optical detection unit, which indicates that the issue is likely to emerge not from a single isolated component but from the combined effect of pulse generation, interface transmission, and detector-side dynamic response. Within the SysML model, the deviation can therefore be traced to the gating-pulse-generation function, the synchronization calibration function, the interface between the electronic pulse unit and the optical detection unit, and the associated parameter constraints on trigger jitter, pulse-width stability, and detector response delay.

The verification logic can be described in flowchart form as follows: abnormal timing response observed in integration testing, then deviation object registered in the requirement-verification record , then affected logical functions, interfaces, and parameter constraints localized in SysML - measured delay and jitter increments mapped to the corresponding Modelica variables and boundary conditions , then executable timing-chain simulation performed under disturbed conditions, then synthesized temporal resolution compared with the 80 ps requirement and remaining budget,then dominant contributors reverse-localized and mitigation actions assigned,then traceability links, parameter budgets, and component responsibilities updated in the model. In the representative disturbed run, the synthesized result may increase from about 92 ps to about 106 ps, which provides explicit evidence that the system no longer satisfies the requirement and that a design or calibration update is necessary.

Table 4 A Structured validation steps for the SysML-Modelica timing-deviation case

Step	*Input or observation*	*SysML-Modelica action*	*Output or decision*
Abnormality capture	Integration test indicates increased equivalent jitter and additional trigger-to-gate delay in the gating path.	Record the disturbed operating mode, measured delay increment, and pulse-shape abnormality as verification inputs.	Measured deviation set for follow-up analysis.
Traceability localization	Observed anomaly needs architectural interpretation.	Use the SysML traceability chain to localize the affected requirement, functions, interface, logical components, and parameter constraints.	Impacted object set including gating-pulse generation, synchronization calibration, and the EPU-ODU interface.
Executable-model mapping	Affected parameters must be transformed into simulation variables.	Map the delay and jitter increments to Modelica boundary conditions, component parameters, and timing-chain disturbance variables.	Executable disturbed case ready for simulation.
Dynamic verification	Nominal and disturbed cases are both available.	Run the linked Modelica timing-chain model and compare the synthesized temporal resolution against the 80 ps target.	Pass-fail conclusion with explicit margin consumption.

Reverse contributor analysis	The disturbed run exceeds the allowable timing budget.	Identify the dominant contributors through term-level comparison and sensitivity-focused diagnosis across synchronization, pulse-shaping, and detector response links.	Dominant cause list for engineering mitigation.
Model update and closure	Corrective action must be decided and documented.	Update requirement allocation, interface assumptions, component responsibilities, and verification records in the SysML traceability model.	Closed-loop decision record supporting the next design iteration.

The SysML layer ensures that the abnormal phenomenon can be connected to explicit requirement, function, interface, and component objects, while the linked Modelica layer tests whether the observed disturbance truly changes the system-level timing outcome. The result is a closed-loop argument from measured anomaly to architecture-level decision, enabling engineers to determine whether the proper corrective action is tighter synchronization calibration, pulse-shaping redesign, interface optimization, detector-side delay compensation, or an update to the requirement-allocation record itself. For ICF ultrafast imaging diagnostics, where acceptable timing margin is small and cross-domain coupling is strong, this capability materially improves the practical credibility of the MBSE process.

5.4 SysML-Modelica Association for Executable MBSE

Within the three-layer logical framework, SysML and Modelica jointly bridge descriptive and executable models. SysML characterizes system requirements, functions, components, interfaces and traceability, whereas Modelica implements them into executable models for time-domain behaviors, signals, pulses and multidisciplinary interactions. System requirements, constraints and logical structures are respectively converted into Modelica parameters, objectives and component models, with interfaces matched to model connectors and variables.

The improved MBSE framework supports both static consistency checking and dynamic performance validation. For ICF ultrafast imaging diagnostics, this linkage effectively addresses design risks caused by mismatches between static architecture and dynamic responses including transients, synchronization errors and interface delays. It enhances the traceability from requirement decomposition to behavioral verification and facilitates digital thread development.

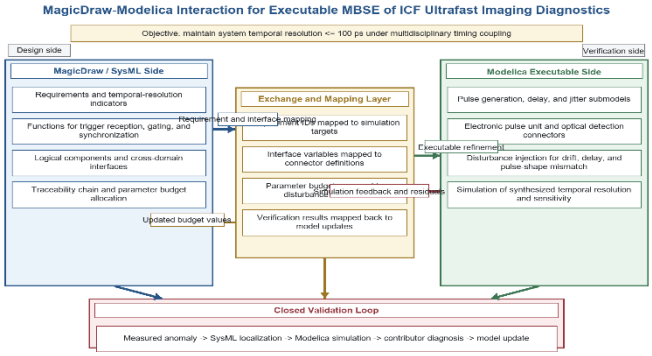


Fig. 6 Schematic interaction between MagicDraw and Modelica for executable MBSE and timing-chain verification.

5.5 Comparison with the Traditional Document-Centric Method

To provide clearer methodological validation, this paper performs a process-level comparison based on two representative engineering tasks: decomposing and locating requirements down to components and modules, and identifying affected objects when the temporal-resolution target changes. Although the comparison is not based on large-sample statistics, it is conducted under the same case scale and is sufficient to demonstrate the advantages of the model-driven approach in execution efficiency and completeness.

Table 5 Process-level comparison between the traditional document-centric method and the MBSE method

Task	*Traditional document-centric method*	*MBSE method*	*Improvement*
Allocation from requirements to components/modules	About 6.4 h, with 3 missed items	About 2.1 h, with 0 missed items	Time reduced by 67.2%
Impact localization for 100 ps > 80 ps	About 47 min, with 2 missed items	About 11 min, with 0 missed items	Time reduced by 76.6%
Cross-disciplinary interface confirmation	Dependent on multiple rounds of manual cross-checking	Located directly through the interface model	Communication iterations are significantly reduced

DOI
10.3384/ecp223193

The comparison indicates that the primary bottleneck of the traditional document-centric method lies in chain identification through manual reading and cross-document checking, whereas MBSE can directly localize objects through explicit model relationships. Accordingly, MBSE demonstrates higher efficiency and a lower omission rate in both requirement decomposition and change-impact analysis.

5.6 Statistics of Model Complexity and Maintainability

To demonstrate that the case is not merely a conceptual schematic but a manageable model of practical engineering scale, the study counts eight categories of objects, including user requirements, system indicators, functional nodes, logical components, key interfaces, module parameters, traceability links, and consistency rules.

Table 6 Statistics of the case-model complexity

Metric	*Count*	*Description*
User requirement	5	Including time resolution, synchronization capability, and imaging-mode configuration
System indicator	8	Covering constraints on timing, interfaces, power supply, and data links
Functional node	17	Obtained by decomposing the activity diagram and control flow
Logical component	3	Control unit, electronic pulse unit, and optical detection unit
Key interface	12	Including the master trigger, configuration bus, high-voltage gating, and power control
Module parameter	18	Covering jitter, delay, response time, sampling error, and related quantities
Traceability links	46	Forming a cross-layer closed loop from UR to MI
Consistency rules	9	Used to check the completeness of interfaces, indicators, and parameter allocation

According to the statistics, the traceability coverage of the case model can be expressed as C_t = 46 / 46 = 1.00, indicating that all key objects defined in this study have been incorporated into the traceability chain. Although the model is substantially larger than a conceptual schematic, its complexity remains within the maintainable range of an engineering team, further demonstrating the practicality of the three-layer architecture relative to more complicated generic frameworks.

6 6 Discussion

6.1 Engineering Applicability and Generalizability

The engineering value of the proposed method is reflected in two respects. First, it provides a unified logical-expression environment for high-precision diagnostic equipment, enabling optical, electronic, and control teams to collaborate around the same timing chain. Second, it preserves the interface conditions required for further integration with simulation, testing, and verification tools. The framework is therefore applicable not only to conceptual design, but also as a higher-level logical structure for subsequent detailed design and integration verification. It also has transfer potential for other scientific instruments characterized by strong timing constraints and multidisciplinary interfaces, such as high-speed gated detection systems and synchronized measurement systems.

6.2 Future Research Directions

Future work will proceed in three directions. First, by leveraging the data-oriented expressive capability of SysML v2, the current graphical model will be transformed into a structured model that can be accessed directly through interface services. Second, timing simulation, interface-consistency checking, and requirement-verification rules will be incorporated into a unified verification pipeline. Third, AI agents will be explored for assisting long-chain traceability navigation, model-consistency checking, and design-change summarization, thereby further improving the level of engineering automation.

A further direction is to establish a SysML-Modelica collaborative workflow, requirement objects, interface definitions, and parameter constraints from the MBSE model could be mapped to Modelica components, connectors, and simulation parameters so that the timing chain of the instrument can be evaluated under multiple operating scenarios. This would support not only forward verification of temporal-resolution compliance, but also backward analysis of which component-level dynamic effects are responsible when simulation results deviate from requirement targets. Combined with automated parameter synchronization and batch simulation, Modelica could further strengthen the closed loop among requirements decomposition, architecture refinement, and verification evidence, thereby improving the practical depth of digital engineering for ICF ultrafast imaging diagnostics.

7 Conclusion

This paper proposes a three-layer MBSE logical design framework for an ICF ultrafast imaging diagnostic system and establishes a complete traceability chain from user requirements to module parameters around the core indicator of system temporal resolution <= 80 ps. Through the time-budget model, bidirectional change analysis, process-level comparison, and model-complexity

statistics, the completeness of the methodological argument is further strengthened.

The results demonstrate that the proposed framework can stably propagate high-level experimental requirements to the module-parameter level and can rapidly localize impacts under requirement adjustment or parameter deviation, with higher efficiency and lower omission rates. For complex systems such as laser-fusion diagnostic equipment, which demand high timing precision and multidisciplinary collaboration, the method can serve as an important foundation for engineering design and subsequent automated verification.

8 References

[1] LINDL J. Development of the indirect-drive approach to inertial confinement fusion and the target physics basis for ignition and gain[J]. Physics of Plasmas, 1995, 2(11): 3933-4024.

[2]CHENG Jinxiu, WEN Tianshu, ZHU Zongyuan, CHANG Zenghu, SHAN Bing, LIU Jinyuan. Progress in the development of MCP-gated picosecond X-ray framing cameras[J]. Optics and Precision Engineering, 1996(1): 44-48.

[3] CAI Houzhi, HUANG Xiaoya, YANG Kaizhi, MA Youlin, XIE Zhaoyang, LIU Jinyuan, XIANG Lijuan. Design and simulation of an ultrafast four-frame CMOS circuit[J]. High Power Laser and Particle Beams, 2024, 36: 122002. DOI: 10.11884/HPLPB202436.240218.

[4] WANG Baomin, CHEN Bo, ZHANG Shicong. Research method and application of whole-train logical design based on MBSE[J]. China Railway Science, 2021, 42(6): 143-151. DOI: 10.3969/j.issn.1001-4632.2021.06.15.

[5] CLELAND-HUANG J, GOTEL O, ZISMAN A, et al. Software and Systems Traceability[M]. London: Springer, 2012.

[6] DELLIGATTI L. SysML Distilled: A Brief Guide to the Systems Modeling Language[M]. Boston: Addison-Wesley Professional, 2013.

[7] OBJECT MANAGEMENT GROUP. OMG Systems Modeling Language (OMG SysML) Version 1.6[R]. Needham: OMG, 2019.

[8] OBJECT MANAGEMENT GROUP. Systems Modeling Language (SysML) v2 API and Services Specification[R]. Needham: OMG, 2025.

[9] FRIEDENTHAL S, MOORE A, STEINER R. A Practical Guide to SysML: The Systems Modeling Language[M]. Waltham: Morgan Kaufmann, 2014.

[10] OBJECT MANAGEMENT GROUP. Unified Modeling Language (OMG UML) Specification Version 2.5.1[R]. Needham: OMG, 2017.

[11] RAMOS A L, FERREIRA J V, BARCELO J. Model-based systems engineering: An emerging approach for modern systems[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), 2012, 42(1): 101-111. DOI: 10.1109/TSMCC.2011.2106495.

[12] HONOUR E C. Understanding the Value of Systems Engineering[C]//INCOSE International Symposium. 2004.

[13] CHAMI M, BRUEL J M. A survey on MBSE adoption challenges[C]//INCOSE EMEA Sector Systems Engineering Conference. Berlin, Germany, 2018: 1-16.

[14] ZELENEVICIUS A, ALEKSANDRAVICIUS A. MagicGrid Book of Knowledge[M]. No Magic / Dassault Systemes, 2020.

[15] HOFFMANN H P. Harmony a Model-Based Systems Engineering Process[R]. IBM Corporation, 2015.

[16] KROB D. CESAMES Systems Architecture Method (CESAM): A Pocket Guide[M]. CESAMES, 2017.

[17] VOIRIN J L. Model-based System and Architecture Engineering with the Arcadia Method[M]. London: ISTE Press - Elsevier, 2017.

Research on Multi-disciplinary Unified Modeling and Simulation Technology of Hualong One System

Peng Cuiting, Luo Fuhong, Lv Hong

State Key Laboratory of Nuclear Power Safety Technology and Equipment, China Nuclear Power Engineering Co., Ltd., Shenzhen, Guangdong, 518116, China, 1786410667@qq.com

Abstract

Nuclear power systems are complex multi-disciplinary systems involving thermal hydraulics, fluid mechanics, control engineering and electrical engineering. Comprehensive performance analysis and optimization of such systems require coupled multi-disciplinary modeling and simulation. Nevertheless, co-simulation via interfaces between separate single-discipline simulation software brings numerous inherent drawbacks. Hualong One is China's domestically developed third-generation pressurized water reactor. To accelerate system modeling and simulation efficiency, this paper independently develops the unified multi-disciplinary modeling library CGN_HPR V1.0 based on Modelica and the hierarchical modeling method, with the domestic MWorks simulation platform as the core development environment. Adopting a four-tier architecture consisting of basic, component, equipment and system levels, the library supports modular, parameterized and visual modeling for streamlined simulation analysis, and addresses the technical bottleneck of insufficient deep coupling in traditional multi-disciplinary simulation workflows. Typical operating conditions of Hualong One are modeled and simulated using the proposed library, which preliminarily verifies the validity of the Modelica-based system simulation model. The results demonstrate that the Modelica-enabled simulation framework enables seamless integration of cross-disciplinary subsystems, mitigates the difficulties in multi-disciplinary coupled simulation for nuclear power plants, and improves overall system analysis and design capacity, providing key technical support for unit safety assessment and operational optimization.

Keywords: HPR1000; Multi-disciplinary Coupling Modeling; Modelica; Nuclear Power System

1 Introduction

Nuclear power system is a complex engineering physical system which integrates fluid, thermal, electronics, control, hydraulic and other disciplines. Its development is a multidisciplinary system engineering. Comprehensive analysis and optimization of overall system performance require integrated modeling and simulation across multiple physical domains. From the 1970s to 1990s, a batch of representative single-discipline modeling and simulation software appeared in the fields of common mechanism, electronics, control and energy in engineering systems, such as Flowmaster (Zhang et al. 2022; Liang et al. 2023), ANSYS (Jing 2023; Chu and Yan 2018), MATLAB/Simulink (Liu et al. 2025; Jiang 2025; Xu et al. 2025), RELAP5 (Zhou 2017; Miao et al. 2016; Yan 2016), etc. However, to realize the modeling and simulation of multi-disciplinary systems is to conduct co-simulation between different disciplinary tools through the interface between single-disciplinary simulation tools (Wang et al. 2025; Xiang 2025; Liu et al. 2025). Multi-disciplinary modeling and simulation can be realized to a certain extent through co-simulation based on interface, but there are the following problems: first, simulation tools must provide compatible interfaces; second, co-simulation requires decoupling of system domains during modeling, which may affect model fidelity and make it difficult to capture certain behavioral characteristics; third, co-simulation can significantly reduce simulation efficiency and accuracy (Hasanpor Divshali et al. 2021; Gurecky et al. 2021). Therefore, the simulation community has been striving to achieve consistent modeling of physical systems in different disciplines in a unified representation. The Modelica language, as an open-standard, object-oriented, equation-based modeling language, has emerged as a promising approach for unified multi-domain modeling (Fritzson 2015). In the nuclear engineering domain, Modelica has been applied to various reactor types, including PWR modeling at EDF (Souyri et al. 2006), IRIS reactor simulation (Cammi et al. 2005), TRIGA reactor modeling (Boarin et al. 2016), and integral pressurized water reactor health monitoring (Anderson and Coble 2024). These studies demonstrate the feasibility and advantages of Modelica-based unified modeling for nuclear power systems.

As the third-generation pressurized water reactor (PWR) nuclear power technology independently developed by China (Ye 2018), the design concept of Hualong One (HPR1000) integrates international advanced nuclear safety standards and engineering practice experience. As early as the beginning of this

century, China started the research and development of independent third-generation nuclear power technology. Through systematic demonstration and technical research, HPR1000 has successfully achieved the unity of high performance and high safety, and has become one of the core reactor types of China's nuclear power "going out" strategy.HPR1000 adopts the "active + passive" dual safety system design (Jing et al. 2017; Song et al. 2017), which significantly improves the accident prevention and mitigation capabilities.At present, HPR1000 has been commercialized in many projects at home and abroad, such as Fuqing Nuclear Power Station in Fujian Province and Karachi Nuclear Power Station in Pakistan.In order to further optimize its economy and safety, it is of great significance to establish a multi-professional unified simulation platform covering the whole life cycle of design, construction, operation and maintenance to promote the continuous upgrading and internationalization of HPR1000 technology.

This paper focuses on the optimization requirements of nuclear power system modeling and simulation technology, and carries out innovative research on the engineering characteristics of HPR1000.The current nuclear power plant is highly nonlinear, and its multi-parameter interaction and close correlation significantly increase the complexity of system modeling.Traditional modeling methods have efficiency bottlenecks and need to invest a lot of resources to complete basic modeling work, which is significantly different from the development trend of graphical modeling in modern engineering fields.In view of this, this paper proposes to build a hierarchical parameter model library system supporting multi-disciplinary coupling, improve modeling efficiency through modular component packaging and parameter-driven technology, and build a unified model library CGN _ HPR V1.0 covering fluid, heat transfer, control, electrical and other professional fields based on the domestic Mworks (Zhou 2021) industrial software environment. In order to verify the accuracy of the constructed model library and the effectiveness of the modeling method, the simulation and test verification of typical steady-state conditions and transient conditions are carried out based on the developed HPR1000 primary, secondary and tertiary circuit system and control coupling model.

2 Development of the Multi-disciplinary Unified Modeling and Simulation Model Library

As the benchmark of the third generation pressurized water reactor nuclear power technology independently developed by China, the operation mechanism of HPR1000 system presents the core characteristics of deep coupling of multi-physical fields, dynamic correlation of multi-temporal and spatial scales, and precise and complex hierarchical structure, covering many key technical fields such as neutron physics, thermal hydraulics, fluid transport, control strategy, etc.

In this study, the structure and characteristics of the core key system of HPR1000 are analyzed by using the hierarchical modeling strategy and the interdisciplinary integrated simulation technology of Modelica, and the models and functions of the basic level, component level, equipment level and system level are constructed, which covers the unified model architecture of fluid, heat transfer, control, electrical and other professional fields. All physical governing equations are natively compiled and solved within MWorks without cross-code data transmission, eliminating the accuracy loss and computational delay introduced by conventional co-simulation interfaces.

2.1 Object-Oriented Hierarchical Modeling Method

Hierarchical modeling technology realizes the structural deconstruction and simulation analysis of complex engineering systems through standardized hierarchical principles.Based on the object-oriented features of the multi-domain modeling language Modelica, the method constructs reusable modular components through physical topology equivalent mapping.This hierarchical construction mode based on the actual physical architecture significantly improves the flexible reorganization and expansion ability of the model base.The object-oriented physical system description capability makes the hierarchical division of the model more suitable for the real connection relationship of the equipment.The modular encapsulation mechanism effectively supports the iterative accumulation of engineering knowledge and the application of cross-project migration.This modeling paradigm realizes the consistent expression of simulation model and physical entity through the structured mapping of system topology (Lv et al. 2023). Compared with standard Modelica modeling practices, this work introduces HPR1000-specific domain knowledge into the hierarchical decomposition, particularly in the definition of equipment-level and system-level boundaries that reflect the actual design topology of the HPR1000 nuclear island and conventional island.

Based on the object-oriented top-down system decomposition method, the HPR1000 is analyzed hierarchically, as shown in Figure 1.The hierarchical decomposition of HPR1000 system is as follows:

(1) Modeling of HPR1000 subsystem and the whole plant. Subsystem models include primary, secondary and tertiary circuit system, reactor core power control system, coolant temperature control system, pressurizer pressure control system, pressurizer liquid level control system, steam generator liquid level control system, steam turbine bypass control system, etc;

DOI
10.3384/ecp223203

(2) Modeling of core equipment involved in the subsystem model, including pressure vessel model, pressurizer model, steam generator, steam turbine model, heater model, condenser model, deaerator model, moisture separator reheater model, etc.;

(3) Component-level model and basic-level model that can be separated from system-level model and equipment-level model, specifically including component models such as pipes, volumes, valves, pumps, separators and dryers that are essential to the fluid circuit, as well as basic models such as interfaces, thermal components, boundaries and media required for modeling.

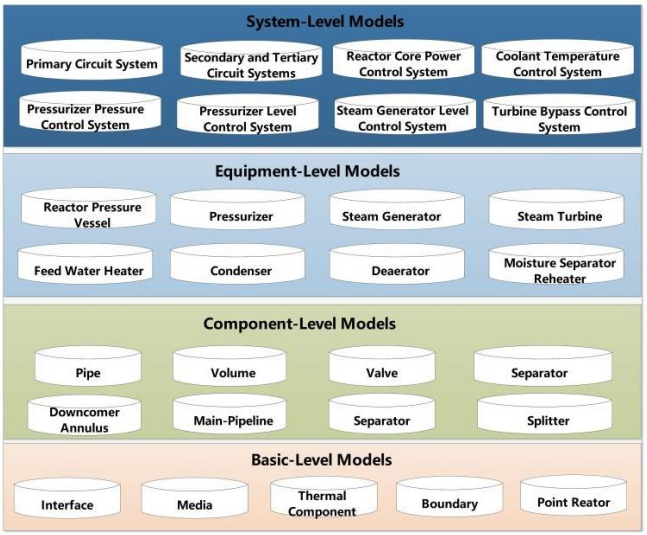


Figure 1. Top-down hierarchical decomposition of HPR1000

2.2 Principles of the Model Library Architecture Design

Building upon the Modelica modeling technology, following the hierarchical modeling idea of "basic level → component level → equipment level → system level", and combining with the equipment structure and spatial arrangement characteristics of HPR1000, the reactor system is disassembled into reusable, verifiable and integrated modeling units step by step, as follows:

(1) The basic level is the bottom core of the model library, which mainly includes the basic control equations for the construction of component-level and equipment-level models, as well as the auxiliary class models that provide various supporting functions.The basic governing equations can be divided into three categories according to the modeling field: the basic equations for core physics modeling, the basic equations for fluid modeling, and the basic equations for thermal component modeling.The auxiliary model library mainly provides supporting resources such as unit library, interface library, medium physical property library, source term coefficient, function library and icon library.The basic-level model does not involve specific equipment structures, but only encapsulates general physical laws and parameters, and provides standardized and reusable modeling units and interfaces for the upper-level model.

(2) Based on the basic level model, the component level constructs a single physical component with specific functions, such as pipes, valves, volumes, etc.The component-level model focuses on the multi-physical field coupling characteristics of a single component, and achieves flexible docking with other components by defining standardized input and output interfaces, which is the core modeling level connecting the preceding and the following.

(3) At the equipment level, multiple component-level models are integrated according to the functional and spatial arrangement to form the core equipment of HPR1000 with complete functions, which is divided into nuclear island equipment and conventional island equipment, such as pressure vessel, steam generator, main pump, pressurizer, steam turbine, condenser, etc.The equipment-level model needs to consider the spatial interactions between components.

(4) The system level is the highest level of the spatial dimension. By integrating the equipment level model, a complete model of the primary circuit system, the secondary circuit system, the tertiary circuit system and the control system of HPR1000 is constructed.The core of the system-level model is to realize the material and energy transfer across equipment and loops, depict the whole process characteristics of the reactor from coolant heating, steam generation to steam turbine power generation, support system-level performance analysis, safety assessment and accident simulation, and is the ultimate application carrier of the modeling and simulation of HPR1000.

2.3 Development of the Model Library

Following the above object-oriented hierarchical modeling method and model library architecture design principle, combined with the exclusive technical characteristics and modeling and simulation requirements of HPR1000, the domestic MWorks simulation tool is used as the core development platform to systematically construct a multi-disciplinary fusion simulation model library adapted to the full scene application of HPR1000, so as to realize the autonomous control of the core modeling results.

The model library strictly follows the four-layer hierarchical architecture of basic level, component level, equipment level and system level, and the layers are progressive and aggregated step by step to form a complete modeling system.The basic-level model library, as the underlying support, covers the basic control equations of fluid, thermal engineering, neutronics and other disciplines, as well as the auxiliary models providing various supporting functions, and provides basic modeling units for the upper-level models;The component-level model library is built based on the basic-level modules, focusing on the core basic components such as pipes, valves and volumes of HPR1000, and

accurately reproducing the core physical characteristics and operation rules of the components;The equipment-level model library takes the component-level model as the core unit, integrates the reactor pressure vessel, steam generator, pressurizer, main pump and other key equipment models of HPR1000, and completely restores the multi-physical field coupling process within the equipment; The system-level model library relies on the equipment-level model to complete the integration of the primary cooling system, the secondary steam power system and the control system according to the actual system topology, so as to realize the complete modeling coverage of the whole system of HPR1000.

Based on the model library development process, the HPR1000 multi-disciplinary unified modeling and simulation model library CGN_HPR V1.0 developed in this paper is shown in Figure 2. In the whole process of model library development, multi-disciplinary coupling logic is deeply integrated to ensure good compatibility and linkage among models at all levels and disciplines, and model parameter calibration and accuracy optimization are completed in combination with the actual operation data and design standards of HPR1000, which not only guarantees the universality and reusability of the model, but also highlights the exclusive of HPR1000. It provides core model support for the follow-up work of HPR1000 full-condition simulation analysis and safety performance evaluation.

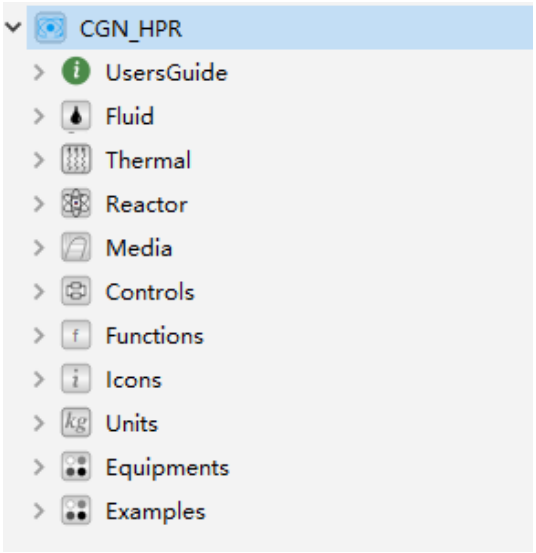


Figure 2. The HPR1000 multi-disciplinary unified modeling and simulation model library CGN_HPR V1.0

3 Simulation results and analysis of typical working conditions

In this paper, the effectiveness and reliability of the multi-disciplinary modeling and simulation model library are verified by the primary, secondary and tertiary circuit system and control coupling model of HPR1000.Based on the developed Modelica primary and secondary circuit equipment models and control system models, a modular modeling method is adopted to decompose each subsystem model into several functionally independent subsystem models, which are assembled and connected according to the interface connection relationship of the actual unit, and then finally integrated into a complete primary circuit system and control coupling model, as well as a secondary and tertiary circuit system and control coupling model. And finally, integrating the two coupling system models to form a final primary, secondary and tertiary circuit system and control coupling model. It should be noted that the verification scope of this study is limited to one steady-state and two transient load-following conditions. While these conditions cover typical operational scenarios, a more comprehensive validation including accident conditions and start-up/shutdown transients would be needed to fully establish the reliability of the model library across the full operating envelope.

3.1 Simulation Results and Analysis of Steady-State Condition

To verify the accuracy of the primary, secondary and tertiary circuit system and control coupling model of HPR1000 based on the multi-disciplinary unified modeling and simulation model library, the simulation and test verification of 100% FP steady-state condition were carried out.

The model parameters are configured according to the 100% FP steady-state operating condition parameters of the actual unit, the output power of the reactor core is 3150MW, the pressure of the pressurizer is maintained at 1505MPa, and the secondary side feedwater flow and enthalpy of each steam generator are in a steady-state state (586.94kg/s, 981.83kJ/kg).The secondary side outlet pressure is 6.817MPa, the speed, flow and head of the main pump are all under the rated operating conditions, and the flow, enthalpy, head and other performance parameters of the steam turbine high pressure cylinder, moisture separator reheater, low pressure cylinder, condenser, low pressure heater, deaerator and high pressure heater of the secondary circuit are all under the rated operating conditions.

Respectively set the parameters of the primary circuit system model, the parameters of the secondary circuit system model and the system simulation parameters under the 100% FP steady state working condition, carry out simulation solution on the system model based on this, and extract the parameters of each key equipment. After the system enters the steady state, the comparison between the key parameters of the main equipment of the primary circuit and the design data is shown in Table 1.The comparison between the key parameters of the main equipment of the secondary circuit and the design data is shown in Table 2. The relative deviation between the simulation value and the design value of the key parameters of the main equipment of the primary and secondary circuits is within 2%.

DOI
10.3384/ecp223203

Table 1. Comparison of Key Parameter Results of Primary circuit

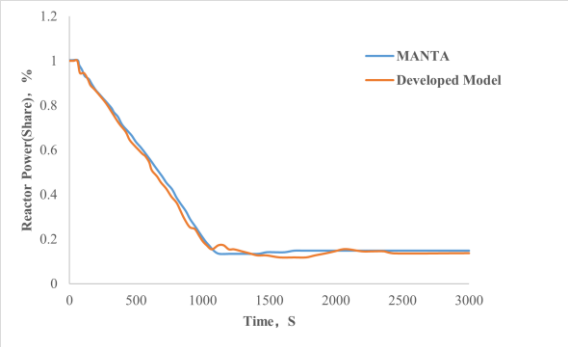
Parameter	Simulation Value	Design Value	Relative Deviation
Coolant flow[Kg/s]	15845	15840	0.03%
Primary circuit pressure[MPa]	15.4	15.5	0.65%
Coolant cold leg temperature[℃]	289.4	289.42	0.007%
Coolant hot leg temperature[℃]	324	324.45	0.14%
Core power[MW]	3150	3150	0
Steam-Generator(SG) heat exchange capacity (single unit)[MW]	1050	1053	0.29%
Feedwater temperature[℃]	228	228	0
SG circulation ratio[-]	5.02	5.05	0.6%
SG steam flow (single unit)[Kg/s]	582.66	586.94	0.73%
SG primary side flow (single unit)[Kg/s]	5281.8	5280	0.04%
SG water level[%]	50.09	50	0.18%
SG Steam dryness[1]	99.83	>99	-
Main pump flow [Kg/s]	5281.76	5280	0.03%

Table 2. Comparison of Key Parameter Results of Secondary circuit

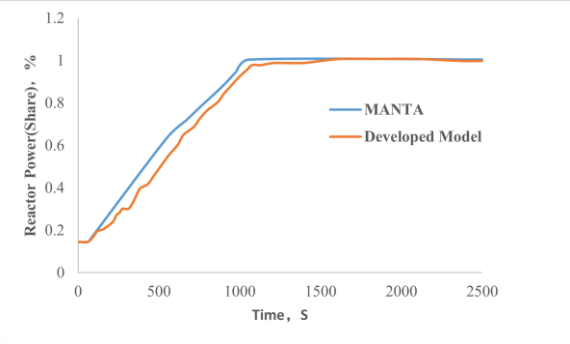
Parameter	Simulation Value	Design Value	Relative Deviation
Steam turbine High-Pressure(HP) cylinder inlet flow[kg/s]	1686.123	1686	0.01%
Moisture separator reheater shell side flow [kg/s]	1320.44	1319	0.11%
Steam turbine Low-Pressure(LP) cylinder inlet flow[kg/s]	1148.122	1142	0.54%
Condenser inlet flow[kg/s]	916.042	931.94	-1.71%
LP heater inlet flow[kg/s]	1033.97	1033	0.09%
HP heater inlet flow[kg/s]	1760.90	1763	-0.12%
SG outlet enthalpy[kJ/kg]	2771.3	2772.5	-0.04%
HP cylinder outlet enthalpy[kJ/kg]	2502	2498.2	0.15%
Moisture separator reheater outlet enthalpy[kJ/kg]	2980	2984.6	-0.15%
LP cylinder outlet enthalpy[kJ/kg]	2312.1	2311.6	0.02%
Condenser outlet enthalpy[kJ/kg]	166.1	167.6	-0.89%
LP heater 1 outlet enthalpy[kJ/kg]	217.6	219.2	-0.73%
LP heater 2 outlet enthalpy[kJ/kg]	352.9	355.6	-0.76%
LP heater 3outlet enthalpy [kJ/kg]	477.2	476.6	0.13%
LP heater 4 outlet enthalpy[kJ/kg]	581.4	583.9	-0.43%
Deaerator outletEnthalpy [kJ/kg]	745.7	744.6	0.15%
Feed water pump outlet enthalpy[kJ/kg]	757.1	756.1	0.13%
HP heater 1 outlet enthalpy[kJ/kg]	902.5	903.3	-0.09%
HP heater 2 outlet enthalpy[kJ/kg]	983.3	982.1	0.12%

3.2 Simulation Results and Analysis of Transient Condition

Under transient conditions, the unit is selected to perform simulation calculation under the typical conditions of linear load reduction from 100% FP to 15% FP at 5% FP/min and linear load increase from 15% FP to 100% FP at 5% FP/min. The calculation results are compared with the calculation results of the third-party software MANTA. MANTA is a mature industry-standard thermal-hydraulic code widely validated for HPR100 conventional island analysis, adopted as the authoritative benchmark for transient comparison in this work. Minor transient mismatches stem from differences in numerical time stepping and control algorithm discretization between MWorks and MANTA, while overall dynamic trajectories remain consistent across both tools. The main parameter changes in the transient process are shown in Figure 3 and Figure 4:



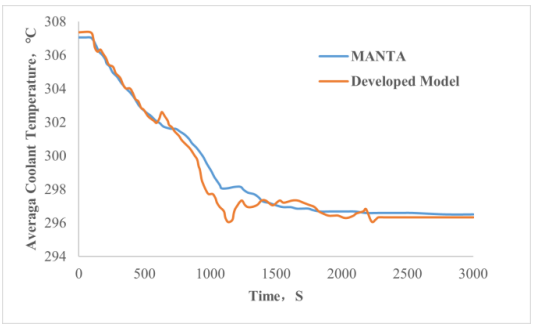
(a)Linear load reduction from 100% FP to 15% FP at 5% FP/min



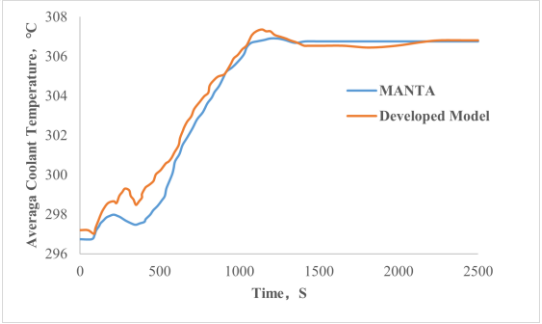
(b)Linear load increase from 15% FP to 100% FP at 5% FP/min

Figure 3. Reactor Power

The comparison results of reactor power show that for the condition of linear load reduction from 100% FP to 15% FP at 5% FP/min, there is only a slight deviation at the overshoot time in the transient, and the final power tends to be stable, which is consistent with the reference simulation results.When the load increases linearly from 15% FP at 5% FP/min to 100% FP, there is only a slight deviation at the overshoot moment in the transient state, and the final power tends to be stable, which is basically consistent with the reference simulation results.



(a)Linear load reduction from 100% FP to 15% FP at 5% FP/min



(b)Linear load increase from 15% FP to 100% FP at 5% FP/min

Figure 4. Average Coolant Temperature

The comparison results of the average temperature of the primary coolant show that the average temperature of the primary coolant decreases gradually after the transient and then tends to be stable under the condition of linear load reduction from 100% FP to 15% FP at 5% FP/min. The simulation results are consistent with the reference simulation results. For the condition of increasing load from 15% FP to 100% FP at 5% FP/min, the average temperature of the primary coolant increases gradually after a small fluctuation, and finally stabilizes near the rated condition, which is consistent with the reference simulation results.

4 Conclusion

Based On Modelica language, a multi-disciplinary unified modeling and simulation model library CGN _ HPR V1.0 is developed to support the modeling, simulation, design and development of HPR1000.The basic simulation model covering nuclear reactor neutron dynamics, thermal engineering, fluid, control and other disciplines integrates the dynamic simulation model of key equipment involved in the dynamic modeling and simulation of HPR1000 system, which provides a model basis for the design and development of HPR1000.

To assess the effectiveness and reliability of the developed library, the authors evaluated coupled models of the primary, secondary, and tertiary circuits and the control system under three conditions: 100% FP steady state, load reduction from 100% FP to 15% FP at 5% FP/min, and load increase from 15% FP to 100% FP at 5% FP/min. The results show that the relative deviation between key simulated parameters and the design values is below 2% under steady-state conditions, while the transient trends are consistent with the reference results from the third-party software. Demonstrated the accuracy of the primary, secondary and tertiary circuit system and control coupling model, and also verified the reliability of the multi-disciplinary unified modeling and simulation model library modeling method and simulation.

References

Zhang, Lide, Li, Jianwei and Tan, Pu (2022). “Application of FlowMaster in Air Volume Regulation of Complex Ventilation Systems of Nuclear Power Plants”. In: Heating Ventilating & Air Conditioning, vol. 52, no. S2, pp. 297-301.

Liang, Tiebo, Wang, Changshuo, Tan, Shuyang and et al. (2023). “Operation Characteristics Analysis of Modular Small Reactor Feedwater Control Valve Based on Flowmaster”. In: Valve, no. 4, pp. 464-467. DOI: 10.16630/j.cnki.1002-5855.2023.04.029.

Jing, Heng (2023). “Wind Load Strength Analysis of VOR Antenna Based on ANSYS Workbench Fluid-Structure Interaction”. In: Value Engineering, vol. 42, no. 31, pp. 114-116.

Chu, Sensen and Yan, Jin (2018). “Simulation on Shrinkage and Creep of Continuous Composite Steel-Concrete Bridge

Based on ANSYS". In: Engineering and Construction, vol. 32, no. 1, pp. 8-10.

Liu, Junfeng, Chen, Quan, Wang, Pengfei and et al. (2025). "Simulation Modeling and Characteristic Analysis of Nuclear Power Unit Feedwater Heating System Based on MATLAB/Simulink". In: Nuclear Science and Engineering, vol. 45, no. 5, pp. 978-986. DOI: 10.20190/j.cnki.02580918.202505017.

Jiang, Jinjin (2025). "Research on LC Filter Non-Complementary Chopper Controlled AC-AC Voltage Regulation Circuit Based on MATLAB/Simulink". In: Automation Application, vol. 66, no. 16, pp. 154-157. DOI: 10.19769/j.zdhy.2025.16.043.

Xu, Xinyi, Chai, Gui'an and Qian, Jiaxin (2025). "Transient Process Analysis of Power System Short-Circuit Faults Based on MATLAB/Simulink". In: China-Arab Science and Technology Forum, no. 7, pp. 97-101.

Zhou, Yueshan (2017). "Simulation Research on Thermal Process and Control System of Nuclear Power Plant Steam Turbine". In: Master's Dissertation, Shanghai Jiao Tong University, Shanghai. DOI: 10.27307/d.cnki.gsjtu.2017.002344.

Miao, Chun, Fang, Shudong and Zang, Xueping (2016). "Transient Simulation of Main Steam Pipeline Rupture for AP1000". In: Journal of Hefei Normal University, vol. 34, no. 6, pp. 33-36.

Yan, Siwei (2016). "Thermal System Modeling of Nuclear Power Plant Conventional Island Based on RELAP5". In: Master's Dissertation, Southeast University, Nanjing.

Wang, Zhitao, Chang, Yingjie, Zheng, Peiying and et al. (2025). "Design and Development of Multi-System Co-Simulation Platform for Gas Turbines". In: Aeroengine, vol. 51, no. 4, pp. 1-9. DOI: 10.13477/j.cnki.aeroengine.2025.04.001.

Xiang, Huazhi (2025). "Research on Thermodynamic Simulation and Optimization Method in Multi-Physics Co-Simulation". In: Master's Dissertation, University of Electronic Science and Technology of China, Chengdu. DOI: 10.27005/d.cnki.gdzku.2025.001116.

Liu, Weicheng, Zhao, Zhengming, Xu, Han and et al. (2025). "Electromagnetic Transient Co-Simulation Interface for Power Electronics and Power System Based on Numerical Integral Substitution". In: Proceedings of the Chinese Society for Electrical Engineering, vol. 45, no. 11, pp. 4394-4405. DOI: 10.13334/j.0258-8013.pcsee.250059.

Hasanpor Divshali, P., Laakso, P., Hänninen, S., Millar, R.J. and Lehtonen, M. (2021). "Electrical and Thermomechanical Co-Simulation Platform for NPP". In: Energies, vol. 14, no. 3, p. 710.

Gurecky, W., De Wet, D., Greenwood, M.S., Salko, B. and Pointer, D. (2021). "Coupling of CTF and TRANSFORM using the Functional Mock-Up Interface". In: Oak Ridge National Laboratory Technical Report.

Fritzson, P. (2015). Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach. IEEE Press/Wiley.

Souyri, A., Bouskela, D., Pentori, B. and Kerkar, N. (2006). "Pressurized Water Reactor Modelling with Modelica". In: Proceedings of the 5th International Modelica Conference, Vienna, Austria.

Cammi, A., Casella, F., Ricotti, M.E. and Schiavo, F. (2005). "Object-Oriented Modeling, Simulation and Control of the IRIS Nuclear Power Plant with Modelica". In: Proceedings of the 4th International Modelica Conference, Hamburg, Germany.

Boarin, S., Cammi, A., Ponciroli, R., Chiesa, D., Previtali, E., Sisti, M., Magrotti, G., Prata, M. and Salvini, A. (2016). "Object-Oriented Modeling and simulation of a TRIGA reactor plant with Dymola". In: Energy Procedia.

Anderson, D. and Coble, J. (2024). "Implementing Component Degradations into a Modelica Model of an iPWR System to Develop Health Monitoring Techniques". In: Nuclear Technology. DOI: 10.1080/00295450.2024.2376996.

Ye, Qizhen (2018). "Adhering to Independent Innovation to Open a New Era of Nuclear Power Development". In: China Nuclear Power, vol. 11, no. 1, pp. 5-10.

Jing, Chunning, Zhao, Ke, Zhang, Liyou and et al. (2017). "Design Concept and Overall Technical Characteristics of Hualong-One". In: China Nuclear Power, vol. 10, no. 4, pp. 463-467.

Song, Daiyong, Zhao, Bin, Yuan, Xia and et al. (2017). "Active and Passive Combined Safety System Design of Hualong-One". In: China Nuclear Power, vol. 10, no. 4, pp. 468-471.

Zhou, Fanli (2021). System Modeling and Simulation Software MWORKS.Sysplorer. Suzhou: Suzhou Tongyuan Soft Control Information Technology Co., Ltd., Dec. 1.

Lv, Hong, Hong, Hao, Li, Wenji and et al. (2023). Introduction and Practical Operation of OpenModelica. Beijing: China Atomic Energy Press.

DOI
10.3384/ecp223203

An FMI-Based Fast Thermal Prediction FMU for Real-Time Digital Twins of Data Center Server Rooms

Haobo Guo[1], Siyuan Yang[2], Chuangkang Yang[3], Xiangmiao Hao[2], Jie Wu[2], Wen-Quan Tao[1]

1 School of Energy and Power Engineering, Xi'an Jiaotong University, Xi'an, China, guohaobo2000@stu.xjtu.edu.cn, wqtao@xjtu.edu.cn

2 Xi'an ShuFeng Technological Information, Ltd., Xi'an, China, yangsiyuan@shufengmht.com, haoxiangmiao_sf@163.com, wujie@shufengmht.com

[3] Alibaba Cloud Computing, Hangzhou, China, yangchuangkang.yck@alibaba-inc.com

Abstract

Data center server rooms require thermal prediction components that can be called by digital twins and system simulation platforms under changing IT loads and cooling conditions. CFD provides detailed spatial information but is too expensive for step-wise online simulation. Machine-learning surrogates are efficient, but offline models often lack state management, rolling prediction, and physical safeguards for continuous simulation. This paper presents an FMI(Functional Mock-up Interface)-compatible fast thermal prediction FMU (Functional Mock-Up Unit) for data center server rooms. The FMU embeds a physics-guided graph neural thermal predictor and exposes standardized FMI inputs and outputs for supply-air temperature, fan state, IT load, historical thermal states, and predicted temperatures. Each simulation step performs input validation, history-window update, model inference, output correction, and FMI variable write-back. A CRAH M1-08 shutdown replay case shows an overall RMSE of 0.284 °C and an average step time of 0.0656 s, supporting near-real-time use in thermal digital twins.

Keywords: FMI, FMU, data center, thermal prediction, digital twin, graph neural network, thermal management

1 Introduction

As data center computing density increases, server-room thermal management has become a critical constraint on computing deployment, energy efficiency, and operational reliability (Masanet et al., 2020; Zhang et al., 2021). In high-density and heterogeneous workload scenarios, IT power variation, cooling-equipment adjustment, local airflow disturbance, and abnormal equipment states can jointly cause rapid temperature changes. If key sensor temperatures or hot spots cannot be predicted in time, the cooling system may respond too late, causing local overheating, increased energy consumption, or operational risk (Du et al., 2023; Zhao et al., 2023).

CFD can provide high-resolution temperature and airflow fields, but its computational cost makes it difficult to use directly for real-time digital twins and system simulation. Data-driven models and surrogate models substantially reduce inference time and provide a feasible route for fast thermal prediction (Asgari et al., 2021a; Asgari et al., 2021b; Athavale et al., 2019). However, an offline prediction model is not automatically an engineering simulation component. A data center digital twin requires a thermal prediction module that receives dynamic supply-air temperatures, fan states, and IT loads over consecutive simulation steps, maintains historical states, and provides stable and physically credible outputs under sensor noise, boundary changes, or out-of-range inputs (Fuller et al., 2020). If an offline model is called only as an external script, inconsistent initial states, abnormal input propagation, and unconstrained outputs can reduce engineering usability.

The Functional Mock-up Interface (FMI) provides a standardized mechanism for integrating models, tools, and simulation platforms (Junghanns et al., 2021). Packaging a fast thermal prediction model as a Functional Mock-up Unit (FMU) allows it to receive dynamic boundary conditions through a unified interface and return key temperature predictions to an external system. In this work, we develop an FMI-compatible fast thermal prediction FMU for real-time digital twins of data center server rooms. The contributions are: (1) an FMU framework that connects external simulation environments and an internal thermal predictor through standardized FMI inputs and outputs; (2) an embedded physics-guided graph neural thermal predictor for sparse-sensor temperature prediction; and (3) a step-wise rolling prediction workflow with input-output safeguards, physics-consistent correction, and conservative fallback.

2 FMI-Based FMU Implementation and Prediction Workflow

The proposed FMU packages fast server-room temperature prediction as a standardized component that

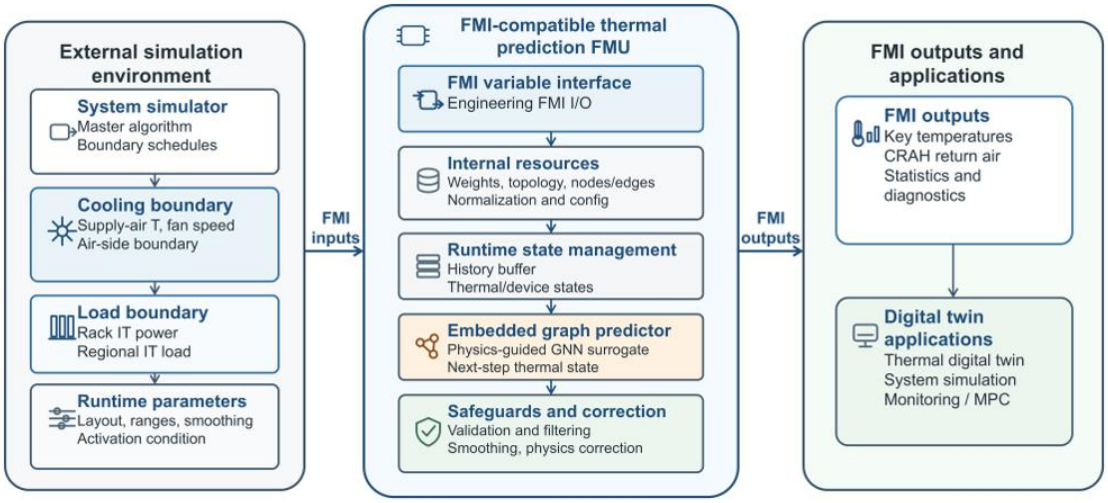


Figure 1. Overall architecture of the FMI-compatible thermal prediction FMU. The external simulation environment passes air-side cooling boundaries, IT load boundaries, and instantiation parameters through FMI inputs. Inside the FMU, resource loading, history-window maintenance, graph-input construction, physics-guided graph prediction, and input-output safeguards are encapsulated. The predicted results are written back as key temperature nodes, CRAH return-air temperatures, room thermal statistics, and diagnostic outputs.

Table 1. FMI variable interface of the thermal prediction FMU.

Variable group	FMI role and update	Meaning and example variables
Cooling boundary	Input; Time-varying	Air-side cooling condition provided by CRAH units Examples: Supply-air temperature; fan frequency or fan speed Notes: Defined at the air-side outlet of the CRAH rather than at the chilled-water side.
Load boundary	Input; Time-varying	Internal heat-source intensity of the server room Examples: Rack-level IT power; regional IT load Notes: Represents dynamic heat generation from IT equipment.
Runtime configuration	Parameter; Set during FMU instantiation or initialization	Room-specific configuration and runtime protection settings Examples: Room or layout type; valid input range; smoothing window; model activation condition Notes: Used to select model resources and protection rules.
Thermal outputs	Output; Time-varying	Predicted thermal states returned to the external simulation environment Examples: Key temperature nodes; CRAH return-air temperature; room thermal statistics Notes: Used by the digital twin, monitoring platform, or control module.
Diagnostic outputs	Output; Time-varying or optional	Runtime status and optional diagnostic information Examples: Step time; validity flag; correction status; protection-mode flag Notes: Used to monitor FMU execution and runtime safeguards.

can be called at each simulation step. It must define external input and output interfaces, organize internal model resources, perform rolling prediction, and maintain stable operation under abnormal inputs or physically inconsistent outputs.

2.1 Overall FMU architecture and FMI interfaces

The FMU behaves externally as a thermal prediction unit with explicit inputs and outputs. At each simulation step, an external digital twin, system simulator, or control module provides current engineering variables. The FMU updates its internal state, constructs model inputs, performs inference and correction, and returns the predicted temperature state for the next time step through FMI output variables. Figure 1 illustrates the overall architecture and information flow of the FMI-compatible thermal prediction FMU, while the corresponding input, output, and configuration variables are summarized in Table 1.

Dynamic inputs are divided into cooling boundary variables and load boundary variables, while runtime configuration parameters are used for instantiation, model selection, and protection-rule settings. Dynamic inputs are divided into cooling boundary variables and load

DOI
10.3384/ecp223211

boundary variables, while runtime configuration parameters are used for instantiation, model selection, and protection-rule settings. The prediction target is the air-side thermal response in a server room. Therefore, the cooling boundary is defined at the air-side outlet of CRAH (Computer Room Air Handler) units rather than at the chilled-water side. Chilled-water temperature, valve opening, and water-side heat transfer belong to upstream chiller or air-handling models. Their effect on the room is represented by the air temperature and supply-air capacity entering the room. Thus, cooling boundary variables include supply-air temperature and fan frequency or fan speed, while load boundary variables include rack-level or regional IT power.

Trained model weights, room topology, node positions, edge-attribute definitions, normalization parameters, and runtime configuration are organized as internal FMU resources. For different rooms or layout types, the FMU can load the corresponding topology, point configuration, and model weights according to instantiation parameters without changing the external FMI interface. The external system only exchanges engineering variables with clear physical meanings and does not directly manage graph structures, normalization scales, or tensor formats.

This design converts an offline thermal prediction model into a simulation component with engineering runtime capability. The FMU not only provides a standardized model interface, but also integrates input validation, history-state maintenance, abnormal input filtering, output smoothing, and physics-consistent correction. It can therefore serve as an independent thermal prediction component in a data center thermal digital twin and can be coupled with cooling-system models, monitoring platforms, or control strategies.

2.2 Embedded thermal prediction model

The embedded prediction core is a physics-guided graph neural thermal prediction model. It predicts the next-step key temperature state from the historical thermal state, cooling boundary, and load boundary maintained internally by the FMU. Physics-informed machine learning and graph networks have been widely used to incorporate structural priors into data-driven prediction processes (Karniadakis et al., 2021; Sanchez-Gonzalez et al., 2020).

The model represents the server room as a sparse thermal graph. Nodes correspond to temperature sensors, CRAH supply-air or return-air related positions, or other key thermal locations. Edges describe possible airflow transport, thermal coupling, or cooling influence paths between nodes rather than CFD mesh faces. Edge attributes encode dynamic operating information such as fan state, rack type, or local heat-source intensity.

This representation does not replace dense CFD discretization, nor does it treat temperature prediction as unstructured black-box regression. Instead, it uses the relatively fixed spatial organization of cooling paths, racks, and sensors in a given room. By encoding major influence relationships among sensors, cooling boundaries, and heat sources as a graph, the model introduces physical structural priors under limited sensing conditions.

At each simulation step, the external system provides only current engineering variables. The FMU maintains the historical sequence internally, updates the history window, and constructs graph inputs from the room topology, node definitions, and edge attributes. The model then predicts the next-step sparse thermal state, which is corrected and protected before being written back to FMI output variables. To reduce non-physical jumps, the predictor estimates a thermal-response target from the current boundary conditions and heat load and updates the current thermal state toward that target through dynamic relaxation.

2.3 Step-wise rolling prediction workflow

During initialization, the FMU loads trained model weights, room topology, node positions, edge definitions, normalization parameters, and runtime configuration. It then selects the corresponding model and topology according to the room layout or instantiation parameters and constructs the initial temperature state, equipment state, and history window.

At each `do_step` call, the external system updates current cooling and load boundary variables through the FMI interface. The FMU reads these inputs, performs normalization and validity checking, appends the latest equipment and temperature states to the internal history window, and removes the oldest information to maintain a fixed-length buffer. If the history window is insufficient, the FMU holds the current state or uses a conservative output strategy. Once sufficient history is available, the FMU constructs graph model inputs, performs inference, applies output correction and smoothing, and writes the predicted temperatures to FMI output variables.

This step-wise mechanism allows the FMU to be embedded naturally in dynamic simulation. Unlike one-shot offline prediction, it updates internal states at every time step and can be used for operational replay, online digital-twin state estimation, and future control-strategy coupling.

2.4 Input-output safeguards and physics-consistent correction

To run stably as an FMI component, the FMU integrates input-output safeguards and physics-consistent correction. Input-side safeguards act before model inference. The FMU checks whether supply-air temperature, fan frequency or speed, IT power, and related variables are within preset engineering ranges. When an input is outside the valid range, the FMU switches to a conservative

prediction branch that extrapolates from the previous thermal state and limits output variation. Short-term historical buffers are also used to filter abnormal power spikes caused by sensor noise or communication faults.

Input-side processing also includes history-window maintenance and model-input construction. This solves the state-continuity problem that arises when an offline model is used directly in engineering simulation. The external system reads and writes FMI variables with physical meanings, while the FMU internally maintains the historical states and graph-structured inputs required by the prediction model.

Output-side safeguards act after model inference. During the first valid prediction, the model output may jump relative to the historical state maintained by the FMU. The FMU therefore applies initial compensation to align the predicted output with the existing state. Subsequent output smoothing and online bias correction reduce short-term oscillations and cumulative offsets. For physically meaningful outputs, the FMU further checks consistency with simplified air-side energy relationships. For example, CRAH return-air temperature should remain broadly consistent with supply-air temperature, total IT load, and total airflow, and cold-aisle temperatures should not fall clearly below the effective supply-air baseline. The complete step-wise prediction workflow, including input validation, graph-model inference, output correction, and the conservative fallback branch, is summarized in Figure 2.

3 Case Study and Results

The shutdown of CRAH M1-08 in the first air-conditioning room was used as a representative dynamic disturbance case. The purpose was not to establish a large benchmark, but to evaluate whether the FMU can receive boundary inputs and output key temperature predictions step by step in a cooling-disturbance scenario.

3.1 Test case and replay setup

When a CRAH unit stops operating, local cooling capacity decreases, and the supply-air boundary, airflow distribution, and return-air temperature may change. This can cause gradual warming in some rack or aisle regions. Compared with steady operation data, a CRAH shutdown case better reflects the FMU's ability to track dynamic thermal disturbances and equipment-side thermal response. In this case, the shutdown time was 230 s and the restart time was 640 s.

The case was evaluated using a step-wise replay test. At each time step, an external replay program provided recorded cooling and load boundary variables, including supply-air temperature, fan state, and IT load. The FMU generated next-step temperature predictions from the current input and internally maintained history, and the predictions were compared with measured sensor temperatures. The replay contains 197 time steps with a 10 s sampling interval. The statistics cover 28 room temperature sensors and 16 CRAH return-air temperature points. Because CRAH M1-08 was the shutdown unit, its return-air point was excluded, yielding 43 evaluated temperature nodes. Figure 3 compares measured and

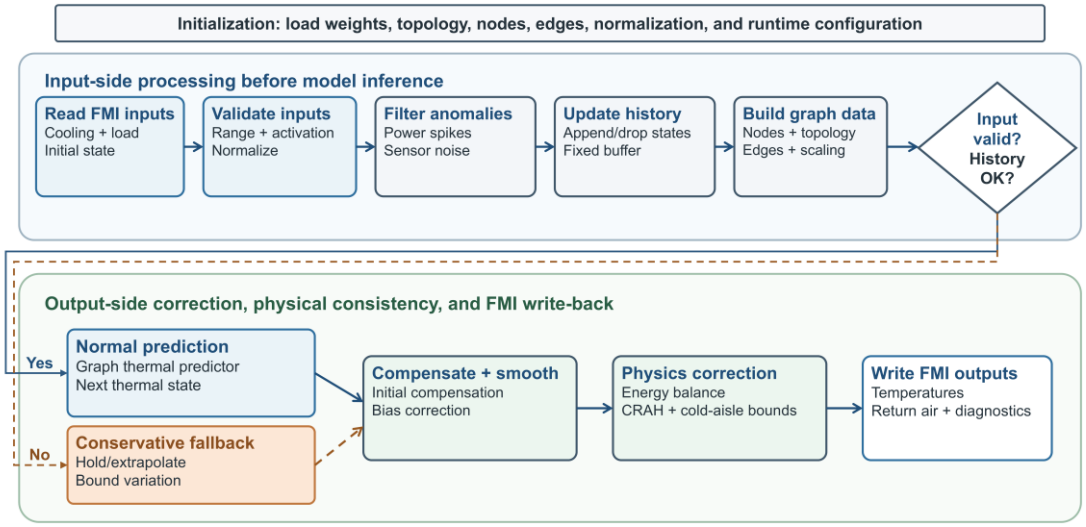


Figure 2. Step-wise prediction workflow and runtime safeguards inside the FMU. In a single do_step call, the FMU reads cooling and load inputs, validates and normalizes them, filters abnormal values, updates the history window, and constructs graph prediction inputs. If the inputs and history are valid, the FMU follows the normal graph-prediction path; otherwise, it switches to a conservative fallback. Before FMI write-back, outputs are processed by initial compensation, smoothing, online bias correction, and physics-consistent correction.

DOI
10.3384/ecp223211

predicted temperatures for Node 25, Node 26, Node 27, and Node 28.

3.2 Prediction accuracy and dynamic response

The FMU achieved high prediction accuracy in the CRAH shutdown replay case. Across 197 time steps and 43 temperature nodes, the overall RMSE was 0.284 °C, the overall MAE was 0.197 °C, and the mean pointwise CV(RMSE) was 0.84%. The worst pointwise CV(RMSE) was 2.55%, corresponding to node TH_26__T_SDO_NTI. These results indicate low errors for most temperature nodes under a dynamic disturbance.

3.3 Runtime performance

Runtime performance is critical for digital twins and system simulation. In this replay case, the FMU average step time was 0.0656 s. With a 10 s data sampling interval, the ratio between computation time and sampling interval was 0.00656, meaning that each step required approximately 0.656% of the sampling interval. The FMU therefore completed step-wise replay much faster than real time while performing input reading,

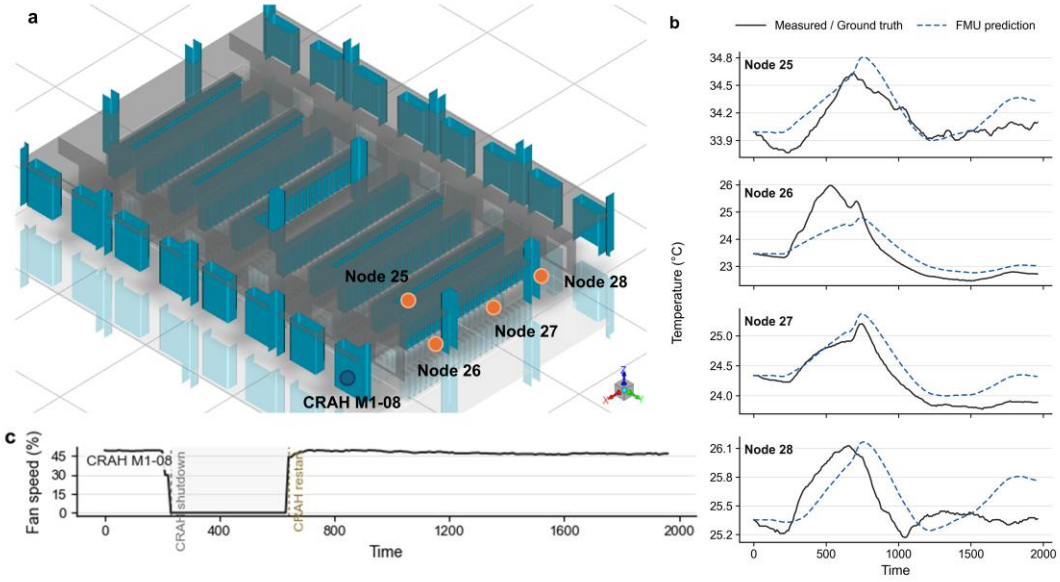


Figure 3. Dynamic thermal response under the CRAH shutdown replay case. Panel a shows the room geometry, the shutdown CRAH M1-08, and representative temperature nodes Node 25, Node 26, Node 27, and Node 28. Panel b compares measured temperatures and FMU predictions. Panel c shows the fan speed of CRAH M1-08 and marks the shutdown event at 230 s and the restart event at 640 s.

Table 2. Prediction accuracy and runtime performance in the CRAH shutdown replay case.

Category	Metric	Value
Case setting	Replay steps	197
Case setting	Shutdown CRAH case	CRAH M1-08 shutdown in the first air-conditioning room
Case setting	Shutdown time	230 s
Case setting	Restart time	640 s
Case setting	Evaluated temperature nodes	43
Accuracy	Overall RMSE	0.284 °C
Accuracy	Overall MAE	0.197 °C
Accuracy	Mean pointwise CV(RMSE)	0.84%
Accuracy	Worst pointwise CV(RMSE)	2.55%
Accuracy	Worst CV(RMSE) node	TH_26__T_SDO_NTI
Hotspot	Hotspot RMSE	0.192 °C
Equipment-side	Return-air RMSE	0.373 °C
Runtime	Average step time	0.0656 s
Runtime	Runtime / sampling interval	0.00656

For thermal-safety-related locations, the hotspot RMSE was 0.192 °C. For equipment-side response, the CRAH return-air RMSE was 0.373 °C. These results show that the FMU tracked both local hotspot temperatures and cooling-equipment-side return-air changes during the shutdown and recovery stages.

history update, model inference, output correction, and variable write-back. The case settings, prediction

accuracy, and runtime performance metrics are summarized in Table 2.

4 Discussion

The main value of this work is the conversion of a fast data center temperature predictor from an offline machine-learning script into an FMI-compatible engineering simulation component. Through FMU packaging, the model can participate in external system simulation, operational replay, and future control-strategy evaluation through standardized interfaces. This is important for data center thermal digital twins because they require not only prediction accuracy, but also stable input handling, step-wise internal state updates, and output variables that can be read directly by other modules.

The embedded physics-guided graph neural predictor handles sparse sensors, complex topology, and dynamic boundary conditions. The graph representation adapts to irregular sensor positions and airflow-coupling relationships in real server rooms. For Modelica/FMI applications, this internal model is not intended to replace all physics-based models. Instead, it serves as a fast surrogate component between high-fidelity CFD and real-time control (Fu et al., 2019; Junghanns et al., 2021).

The engineering safeguards distinguish the FMU from a conventional offline predictor. Real operational data may contain sensor anomalies, power spikes, out-of-range inputs, inconsistent initial states, and output jumps. Directly calling an offline trained model may therefore produce unstable or non-physical predictions. By combining input validity checking, fallback prediction, initial compensation, smoothing, and physics-consistent correction, the FMU improves robustness during dynamic simulation.

This work has limitations. The current study focuses on the fast thermal prediction FMU itself and has not yet completed closed-loop coupling with a full Modelica cooling-system model or controller. The validation is also based on one representative CRAH shutdown case, so generalization should be tested under more dynamic events. Finally, the physics-consistent correction uses engineering approximations rather than rigorous conservation-equation solving. Future work will connect the FMU to data center cooling-system models, MPC frameworks, and energy optimization workflows to build a more complete thermal-management digital twin (Fu et al., 2019).

References

Asgari, S., MirhoseiniNejad, S. M., Moazamigoodarzi, H., Gupta, R., Zheng, R., and Puri, I. K. (2021a). A gray-box model for real-time transient temperature predictions in data centers. Applied Thermal Engineering, 185, 116319. https://doi.org/10.1016/j.applthermaleng.2020.116319

Asgari, S., Moazamigoodarzi, H., Tsai, P. J., Pal, S., Zheng, R., Badawy, G., and Puri, I. K. (2021b). Hybrid surrogate model for online temperature and pressure predictions in data centers. Future Generation Computer Systems, 114, 531-547. https://doi.org/10.1016/j.future.2020.08.029

Athavale, J., Yoda, M., and Joshi, Y. (2019). Comparison of data driven modeling approaches for temperature prediction in data centers. International Journal of Heat and Mass Transfer, 135, 1039-1052. https://doi.org/10.1016/j.ijheatmasstransfer.2019.02.041

Du, Y., Zhou, Z., Yang, X., Yang, X., Wang, C., Liu, J., and Yuan, J. (2023). Dynamic thermal environment management technologies for data center: A review. Renewable and Sustainable Energy Reviews, 187, 113761. https://doi.org/10.1016/j.rser.2023.113761

Fu, Y., Zuo, W., Wetter, M., VanGilder, J., Han, X., and Plamondon, D. (2019). Equation-based object-oriented modeling and simulation for data center cooling: A case study. Energy and Buildings, 186, 108-125. https://doi.org/10.1016/j.enbuild.2019.01.018

Fuller, A., Fan, Z., Day, C., and Barlow, C. (2020). Digital twin: Enabling technologies, challenges and open research. IEEE Access, 8, 108952-108971. https://doi.org/10.1109/ACCESS.2020.2998358

Junghanns, A., Gomes, C., Schulze, C., Schuch, K., Widl, E., Blaesken, M., Zacharias, G., Pillekeit, A., Wernersson, K., Sommer, T., Bertsch, C., Blochwitz, T., and Najafi, M. (2021). The Functional Mock-up Interface 3.0: New features enabling new applications. Linköping Electronic Conference Proceedings, 181, 17-26. https://doi.org/10.3384/ecp2118117

Karniadakis, G. E., Kevrekidis, I. G., Lu, L., Perdikaris, P., Wang, S., and Yang, L. (2021). Physics-informed machine learning. Nature Reviews Physics, 3, 422-440. https://doi.org/10.1038/s42254-021-00314-5

Masanet, E., Shehabi, A., Lei, N., Smith, S., and Koomey, J. (2020). Recalibrating global data center energy-use estimates. Science, 367(6481), 984-986. https://doi.org/10.1126/science.aba3758

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. W. (2020). Learning to simulate complex physics with graph networks. Proceedings of the 37th International Conference on Machine Learning, PMLR 119, 8459-8468. https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html

Zhang, Q., Meng, Z., Hong, X., Zhan, Y., Liu, J., Dong, J., Bai, T., Niu, J., and Deen, M. J. (2021). A survey on data center cooling systems: Technology, power consumption modeling and control strategy optimization. Journal of Systems Architecture, 119, 102253. https://doi.org/10.1016/j.sysarc.2021.102253

Zhao, R., Du, Y., Yang, X., Zhou, Z., Wang, W., and Yang, X. (2023). A critical review on the thermal management of data center for local hotspot elimination. Energy and Buildings, 297, 113486. https://doi.org/10.1016/j.enbuild.2023.113486

A Modelica-FMI digital twin for reinforcement learning-based NOx emission control of boilers under deep peak shaving

Qing Duan[1], Shengshan Bi[1,]*, Xuyao Tang[2], Dongyang Zhou[1,2], Aiguo Xu[3]

*Corresponding Author. bss@mail.xjtu.edu.cn.

[1]Key Laboratory of Thermo–Fluid Science and Engineering, Ministry of Education, School of Energy and Power Engineering, P.R. China, Xi'an Jiaotong University.

[2]Xi'an Thermal Power Research Institute, P.R. China.

[3]Nanjing Yuansi SimTek Co., Ltd.

Abstract

Deep peak-shaving in coal-fired boilers causes non-linear dynamics, rendering traditional PID control insufficient for NOx emission management. This paper proposes a Modelica-based digital twin framework for a 600 MW boiler. The CFD-PINN model is coupled with a physics-based steam-side dynamic model and an adaptive correction layer driven by DCS (Distributed Control System) data to ensure high-precision synchronization under operating condition drift. An intelligent reinforcement learning agent using the Recurrent Proximal Policy Optimization (Recurrent PPO) algorithm is developed to inject residual bias signals into the main loop, optimizing NOx emissions without altering the original architecture. Validated in seven scenarios spanning 300–600 MW, this strategy reduces the 95th-percentile NOx by 19%–23% and emission fluctuations by 32.7%–43.2% across four dynamic scenarios relative to both manually tuned and gain-scheduled PID baselines, while maintaining combustion efficiency. This approach provides a deployable, low-risk pathway for achieving stricter NOx compliance under deep peak shaving operation.

Keywords: Coal-boiler, Digital twin, reinforcement learning, deep peak shaving, NOx emission control.

1 Introduction

The rapid expansion of renewable energy generation in China has shifted the role of coal-fired boilers from that of a traditional base-load power source to that of a dynamic, flexibility-oriented asset for grid balancing(Wu et al., 2025). Against this backdrop, coal-fired boilers require extensive load-shaving operations, sustaining operation at 40%–50% of rated capacity with frequent ramps spanning 30%–60% of full load within minutes(Jin et al., 2023). This operating mode poses a challenge to traditional control schemes. Feedforward(FF) mapping based on load dispatch deteriorates when operating outside its calibrated range. Constant rate of rise PID loops, which are tuned for a single design condition, exhibit significant oscillation and steady-state error during rapid load transients. The most significant operational impact is the deterioration in NOx emissions(Liu and Bansal, 2014, Wang et al., 2025c). Under ultra-low emission standards, NOx peaks rather than time-averaged values are considered a key determinant of a unit's compliance(Wang et al., 2025c).

Scholars have proposed various strategies to address these control challenges. Model Predictive Control (MPC), by coupling a linearized plant model with receding-horizon optimization, demonstrates effective tracking under moderate transients(Wu et al., 2015, Liu and Kong, 2014). However, its industrial adoption in boiler combustion control is hindered by the difficulty in obtaining control-oriented dynamic NOx models that maintain validity across the deep peak shaving range(Huang et al., 2025a). Adaptive and gain-scheduled PID schemes mitigate the calibration fragility of fixed-gain loops, yet they remain fundamentally constrained by the scalar feedback law, which struggles to encode the complex trade-offs between NOx emissions, combustion completeness, and steam-side stability(Fu et al., 2025). Recently, deep reinforcement learning (RL) has emerged as a promising alternative, capable of learning anticipatory, multi-objective control policies directly from high-fidelity simulators without explicit model identification(Wei et al., 2017, Huang et al., 2025b). RL applications have been reported in HVAC, gas turbines, and chemical processing(Spielberg et al., 2019, Sage et al., 2023, Wang et al., 2025b). However, its maturity in coal-fired boilers remains constrained by two primary factors: the prohibitive computational cost of integrating CFD combustion physics into the RL training loop, and the sim-to-real gap caused by intrinsic unmodeled plant drifts (slagging, air ingress, and coal quality variations) that are inherent to live power plant operations(Chen et al., 2023, Wang et al., 2024).

Developing a simulation framework that is simultaneously physically accurate, computationally tractable for RL training, deployable as a standardized co-simulation unit, and maintainable without redesigning the control software represents a third, equally significant challenge. Existing RL studies on boilers rely either on simplified lumped models that sacrifice physical fidelity, or on proprietary simulation environments tightly coupled to specific toolchains, limiting both reproducibility and industrial transferability. No prior work has addressed these integration requirements within a standardized, open modeling framework.

This paper systematically addresses these challenges within a Modelica-based digital twin framework built on the FMI 2.0 co-simulation standard. A four-layer architecture was proposed, comprising an integrated CFD-PINN combustion reduced-order model (ROM), a Modelica steam-water cycle model, a DCS (Distributed Control System) data-driven calibration layer, and a Recurrent PPO agent. The boiler model is compiled as a single co-simulation FMU via Dymola, exposing a clean RealInput/RealOutput interface to the RL training loop through fmpy. The calculation process of the digital twin framework in this work is presented in Figure 1.

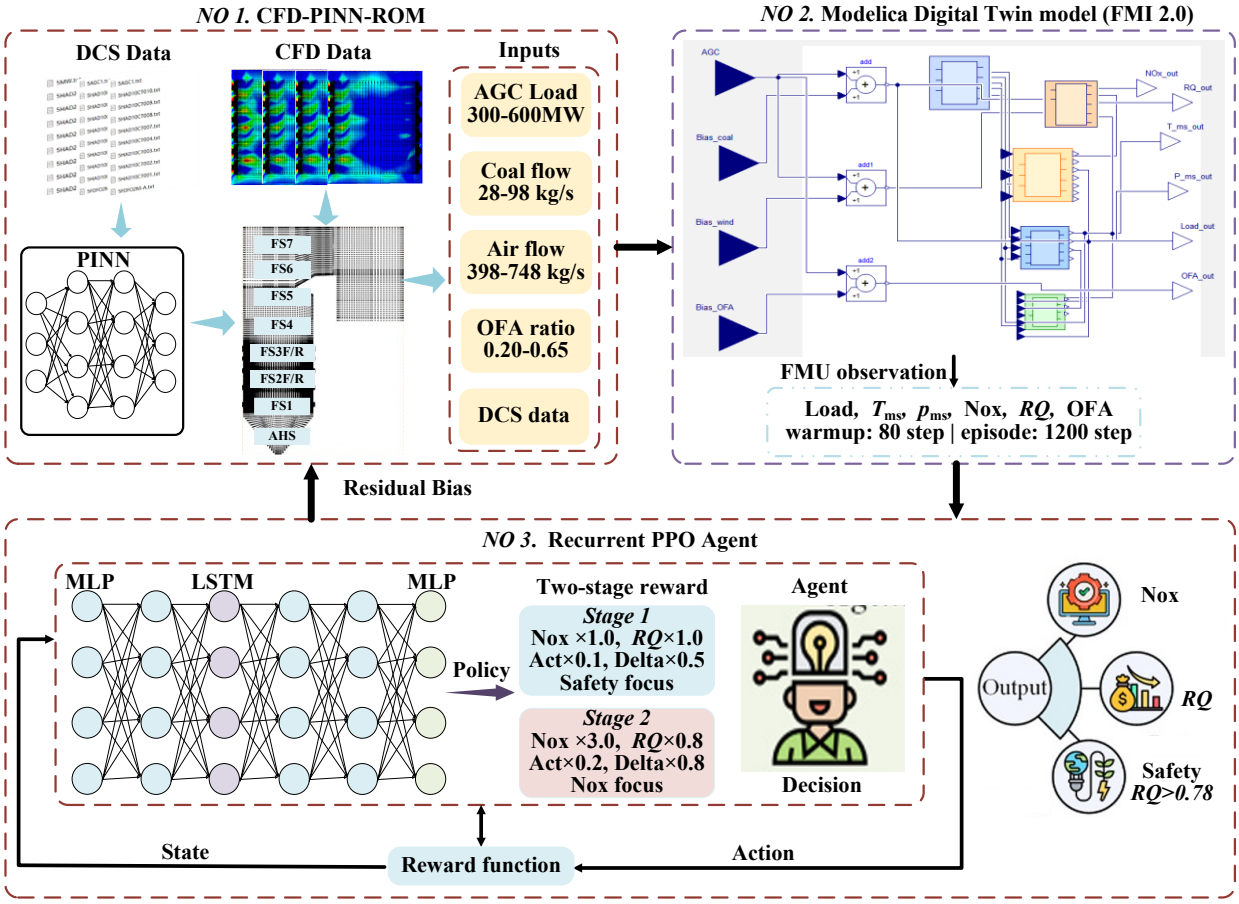


Figure 1. The calculation process of the digital twin framework

This architecture decouples model development from control policy training — the FMU can be updated, re-calibrated, or replaced independently of the RL agent, a property essential for long-term industrial maintainability. The ROM provides a physically consistent surrogate, ensuring the computational feasibility of RL training within the FMU environment. The DCS calibration layer as a pre-processing module, dynamically compensating for long-term operational drifts such as slagging and coal quality fluctuations. This agent is designed as an auxiliary residual bias controller, making fine-tuning adjustments to the existing FF plan without interfering with safety-critical main control loops such as steam temperature and pressure. The main contributions are as follows.

1. A Modelica/FMI co-simulation architecture for RL-based boiler control is proposed and fully implemented. The design addresses concrete FMI engineering challenges — numerical stability under the DASSL solver, smooth table lookups to prevent integrator restarts, and a fixed 1 s communication step size matched to the RL decision frequency.
2. A CFD-PINN-ROM model was developed to bridge the gap between high-fidelity CFD and the computational efficiency required for machine learning, and is integrated into Modelica via CombiTable2D to enable FMU-compatible lookup..
3. A DCS-driven adaptive correction layer was designed, combining a first-order lag residual estimator with adaptive forgetting recursive least squares regression.

DOI
10.3384/ecp223217

This mechanism enables the digital twin to self-calibrate against inherent plant drifts and ensure long-term model accuracy.

4. An intelligent RL agent utilizing Recurrent PPO to optimize NOx emissions, which improves peak emission performance and operational efficiency by dynamically adjusting existing FF control, while strictly preserving the integrity and safety of the primary control loop.

The framework was systematically validated through rigorous comparative analysis against expert-tuned FF and manually tuned PID baselines across a wide range of deep peak shaving cases. The proposed digital twin framework integrates four interrelated subsystems.

2 Methods

The framework comprises four interrelated subsystems, as shown in Figure 1. On the combustion side, Modelica-based empirical relationships have been replaced by a CFD-PINN-ROM model, which maintains high-precision combustion predictions at a computational cost of microseconds.

The dynamic characteristics of the steam side are modelled using physics-based differential equations within the Modelica modelling paradigm(Wang et al., 2025a). To ensure consistency between simulation and reality, an adaptive correction layer derived from the DCS is implemented to compensate for unmodelled physical drifts, such as fouling and variations in coal quality. Finally, a reinforcement learning (RL) agent interacts with the Modelica model via functional simulation units (FMUs) to optimize NOx emissions during deep-trough transients. This decoupled design strategy separates offline model training from real-time execution, ensuring the digital twin remains computationally efficient in industrial control applications.

2.1 Data generation and PINN model

The subject of this study is a 600MW coal–fired boiler that is currently in operation at a certain power plant in China. The boiler model and the process for establishing the CFD database are presented in Figure 2. Combustion data were derived from validated CFD simulations covering the full deep peak shaving range (250–620 MW, airflow 398–748 kg/s)(Zhou et al., 2026). The furnace was divided into ten sections (AHS, FS1–FS7). The average flue gas temperature and wall heat flux density for each section were extracted as the ROM targets.

A physics-informed neural network (PINN) was used to train the CFD dataset to learn the non-linear mapping relationship between operating conditions and combustion output(Raissi et al., 2019, Cai et al., 2021). The input feature vector is shown in Eq (1).

$$x = \left[L, W, m_{coal}, r_{OFA}, Q_{HV} / Q^{1}{}_{HV}, 0, e_{combo} \right] \quad (1)$$

where L is the load (MW). W is the total air flow rate (kg/s). m_{coal} is the total coal flow rate (kg/s). r_{OFA} is the over-fire air ratio. $Q_{\mathrm{HV}}/Q^{1}{}_{\mathrm{HV}}$ is the calorific value of the coal normalized by its design value. e_{combo} is a one-dimensional thermal encoding of the mill combination.

The PINN network architecture incorporates four hidden layers [128, 256, 256, 128] with layer normalization and Tanh activation function(Gong et al., 2025).

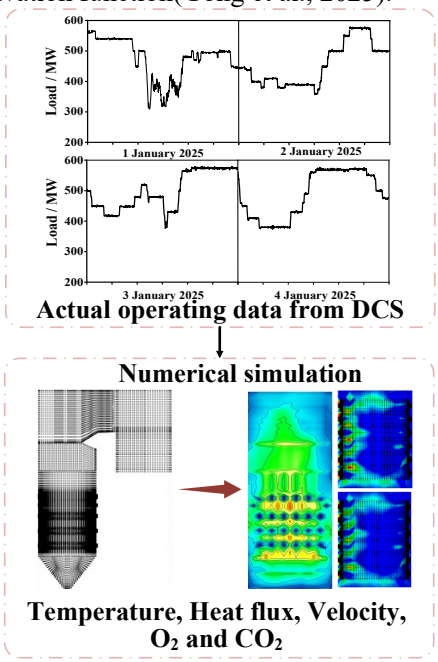


Figure 2. The boiler model and the process of establishing the CFD database

To ensure physical consistency across the entire deep peak shaving range, the composite loss function L as shown in Eq (2).

$$L = \lambda_{data} \cdot L_{data} + \lambda_{mono} \cdot L_{mono} + \lambda_{energy} \cdot L_{energy} + \lambda_{symm} \cdot L_{symm} + \lambda_{bc} \cdot L_{bc} \quad (2)$$

Where L_{data} is the standard supervised MSE on the CFD targets. L_{mono} enforces a monotonically increasing trend in combustion output with respect to load. L_{energy} restricts the total heat release to match the heat input from the coal. L_{symm} penalizes asymmetry in heat flux across the front and rear walls. L_{bc} implements the physical boundary conditions.

The physical constraints were gradually introduced via a linearly increasing schedule. Separate PINN models were trained for the flue gas exit temperature (T_{exit}), combustion efficiency (RQ), segment heat flux density (q_{seg}), and total heat output (Q_{total}). Anisotropic Gaussian smoothing (with standard deviations of 0.6 and 0.8, respectively) and monotonic regression were employed for post-processing to ensure that the load in the T_{exit} grid exhibited a monotonic trend. Linear interpolation was subsequently applied within the unit switching transition zone (380–450 MW) in the RQ grid. Each trained PINN predicts output values across a 20×20 grid covering load and air flow. The generated tables are saved in Modelica CombiTable2D format.

2.2 DCS-driven adaptive correction

The CFD-PINN-ROM model was trained using CFD results. However, factors that may arise during actual power plant operation—such as scale build-up, air leaks, or changes in coal quality—were not taken into account. A two-stage adaptive correction scheme was employed to eliminate these systematic biases. $\Delta t = y_{DCS,t} - y_{ROM,t}$, t is represented as the residual between the DCS measurement and the ROM prediction. The correction term is estimated by the hybrid corrector, as shown in Eq (3).

$$\Delta t = \begin{cases} \Delta t, & if\ t_{steady} \geq 6min\ and\ |\Delta L| < 15MW \\ \phi_t^T \theta_t, & otherwise \end{cases} \quad (3)$$

where t_{steady} is the duration of the current steady-state segment, and ΔL is the load fluctuation rate. In the steady-state regime, the lag-1 residual is adopted to eliminate systematic bias with minimal latency. The state-dependent thresholds (6 min/15 MW) represent the boiler's dominant thermal inertia and primary control capacity, respectively. During the transient, a polynomial regression model based on the feature vector ϕ_t is employed. The parameter vector θ_t is recursively updated via a Recursive Least Squares (RLS) filter with an adaptive forgetting factor λ_t, as shown in Eq (4).

$$\lambda_t = \lambda_t - (\lambda_{base} - \lambda_{min}) \cdot min\left(\frac{|\Delta L|}{L_{rated} \cdot 3\delta_{load}}, 1 \right) \quad (4)$$

This design incorporates a self-calibration mechanism; while the two sensors are insensitive to measurement noise during stable operation, they are able to respond maximally to new DCS evidence when the load changes, thereby bridging the remaining gap between the alternative model trained offline and the actual plant.

2.3 Modelica-based digital twin integration

The boiler is represented by a layered Modelica package, which is translated by Dymola. The complete signal flow for all internal components is shown in Figure 3. This boiler Modelica model comprises five reusable sub-models: DCS boundary, CFD-PINN combustion model, steam-water cycle, convection section heat balance, and energy balance monitor.

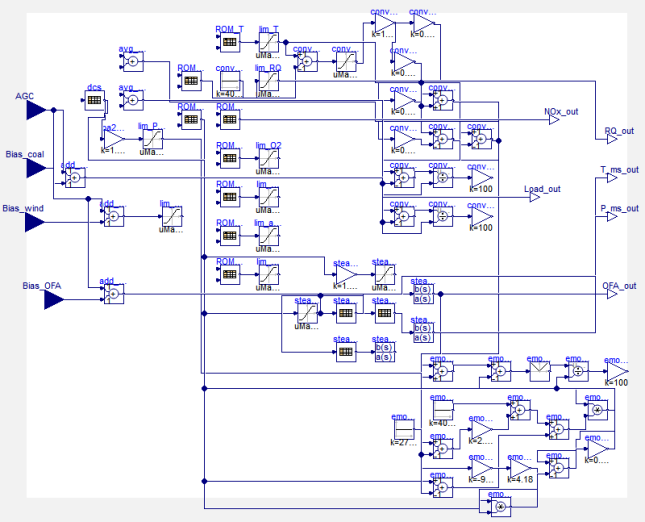


Figure 3. Dynamic model of the coal-fired boiler system

The DCSBoundary encapsulates a single CombiTable1Ds configured with linear segment smoothing. The built in Modelica time variable is directly connected to the table's input, so the block automatically advances through the DCS data at the simulation clock rate without the need for an external driver. Two pairs of redundant temperature sampling points (T_{horiz} and T_{tail}) are averaged using a Modelica.Blocks.Math.Add block. Pressure signals in Pascals are converted to megapascals by Gain block, and a Nonlinear.Limiter prevents division by 0 in downstream calculations by setting a physical lower limit for p_{ms}.

The Combustion ROM integrates offline-trained CFD-PINN data into Modelica based on CombiTable2Ds. Their indices are determined by the AGC load command and the combustion air flow rate. A dedicated Nonlinear.Limiter is connected in series with the air flow input to ensure that the validated CFD training range is enforced before any table lookup. The smooth (1, ·) expression of this wind shear is utilized in the underlying Modelica equations to prevent the integrator in the DASSL solver from restarting. The output is passed through a second Nonlinear. Limiter block, which ensures a physically acceptable output range. The NOx output is not connected to a downstream Limiter, as it serves as the primary reward signal for the RL agent.

The Convective Heat Balance module is used to calculate the energy balance of the convective heat transfer surface downstream of the boiler outlet. The model consists of components from the Blocks. Math and Blocks. Sources libraries. The non-linear limiter block lim_{dT} prevents negative heat release during cold start-up by enforcing max (0, T_{exit} - $T_{exhaust}$). This is an important robustness measure for RL when exploring low-load conditions. The flue gas temperature distribution is reconstructed using a chain of two Math. Add and Math. Gain blocks, thereby yielding the predicted horizontal flue inlet temperature and tail flue outlet temperature. These are compared with DCS data, and the percentage validation residuals $errT_{horiz}$ and $errT_{tail}$ are calculated based on a chain of Math. Add/Division/Gain blocks, serving as the actual output signals for the MSL.

DOI
10.3384/ecp223217

The SteamSide Signal is implemented as a Modelica DAE subsystem comprising three continuous state variables: main steam temperature T_{ms}, pressure p_{ms}, and mass flow rate m_{ms}. Each variable is driven by a first-order lag term. Each first-order lag term is implemented as a Blocks. Continuous. TransferFunction block, with a numerator $b = \{1\}$ and a denominator $a = \{\tau, 1\}$, corresponding to the standard MSL representation of the first-order system $H(s) = 1/(\tau_s+1)$. The time constant is determined from the step response observed by the DCS during the 320 to 608 MW ramp-up test. Initial conditions are specified via the y_{start} parameter in each TransferFunction block. During the FMU compilation process, Dymola maps this parameter to the Modelica initial equation structure to ensure consistent initialization of the differential equations via the DASSL integrator. The load non-linear limiter restricts the combined table input to the range of 300–640 MW. The gain limiter pair calculates the coefficient per unit of heat and uses it as an auxiliary monitoring output. Distributed thermo-hydraulic models based on the ClaRa or ThermoSysPro Modelica libraries increase the number of continuous states by one or two orders of magnitude, which causes the FMU to run too slowly to be used for reinforcement learning training. The three-state DAE sub-model captures the primary thermal inertia timescales of the steam-water cycle while maintaining the number of FMU steps required for the RL training.

The energy balance monitor is an algebraic Modelica sub-model used to calculate the relative difference between the heat released by the combustion and convection sub-models and the enthalpy rise on the steam side.

The boiler model and ModeBlocks.Math.Add components are instantiated. The residual error signal from the RL agent is superimposed via these components onto the operating plan generated by the AGC based on load, primary air flow, and OFA ratio. RealInput connectors (AGC, $Bias_{coal}$, $Bias_{wind}$, $Bias_{OFA}$) and RealOutput connectors (NOx_{out}, RQ_{out}, $T_{ms,\,out}$, $p_{ms,\,out}$, $Load_{out}$, OFA_{out}) define the boundaries of the FMU. The boiler system model is compiled into a co-simulation FMU compliant with the FMI 2.0 standard. The DASSL solver is used internally, and the external communication step size is fixed at Δt = 1s to match the decision-making frequency of the reinforcement learning agent.

2.4 RL control strategy design

Markov decision process formulation. The NOx control problem is formulated as an FMU-based probabilistic Markov decision process (POMDP). At each time step, the agent receives an observation o_t and generates a residual deviation action a_t, which is superimposed onto the intrinsic FF plan. The agent does not replace the existing primary control loops for main steam temperature and pressure. Instead, it operates as a learned secondary controller dedicated to optimizing the combustion side. This combined FF and residual design significantly reduces the policy exploration burden, as a zero policy output reproduces the plant's validated open-loop behavior. Furthermore, if the agent is disabled while online, it provides a natural safety fallback.

A 21-dimensional observation vector is estimated to compensate for the partial observability inherent in the FMU at each time step, as shown in Eq (5).

$$o_t = \left[x_t, x_{t-1}, a_{t-1}, I_t, \Delta e_t\right]^T \tag{5}$$

where x_t is the raw measurement vector. x_{t-1} is its one-step delayed value. $\tilde{a}_{t-1}$ is the previous filtering action. I_t is the tracking error e_t is a leaky integral with a retention factor of 0.99 and a saturation range of ±200, while $\Delta e_t = e_t - e_{t-1}$ is the one-step error derivative.

These are mapped to the physical deviation signal by a scaling factor $s = [6.0\ kg/s, 80.0\ kg/s, 0.15]^T$ per channel. This scaling factor corresponds to approximately ±8% of the rated coal flow rate, ±11% of the rated primary air flow rate, and ±0.15 of the absolute OFA ratio at medium load. A first-order low-pass filter with a coefficient of $\alpha = 0.3$ attenuates high-frequency fluctuations before the deviation is applied to the FMU.

Reward function design. The reward combines a logarithmic penalty for NOx, a saturated linear penalty for steam-side tracking error, an RQ penalty with hard termination conditions and soft termination when $RQ <$ 0.78, three action regularization terms (L^2 magnitude, L^2 difference, L^1 residual), and an integral-NOx error term, which rewards steady-state convergence.

Network architecture and training strategy. The intelligent agent is implemented as a Recurrent PPO variant(Pleines et al., 2022). Both the actor (action generator) and the critic (value estimator) utilize an MLP-LSTM-MLP architecture, featuring an LSTM hidden layer size of 128 and two-layer MLP heads with dimensions [128, 64](Huang et al., 2025b, Yan et al., 2024). To mitigate value estimation bias under non-stationary policy weights, the critic employs a dedicated LSTM stream independent of the actor(Beck et al., 2024). The training follows a three-stage, step-by-step learning strategy(Liu et al., 2023). The initial safety-critical stage focuses on carrying out safety-critical tracking tasks and ensuring combustion stability. The emissions optimization stage gradually increases the penalty for nitrogen oxide emissions. This is followed by a fine-tuning stage to restore the accuracy of steam pressure tracking that was previously lost due to excessive emissions optimization.

Two non-learning controllers are utilized within the same observation-action interface: a pure FF controller, which reflects the plant intrinsic open-loop schedule, and a manually tuned PID controller that regulates steam temperature and NOx emissions via independent PI control loops. The framework is implemented in PyTorch, with the Modelica-based industrial plant model exported as an FMI 2.0 co-simulation FMU from Dymola and integrated based on FMPy 0.3.

The RL agent is implemented in Python using the fmpy FMI interface. This section presents three representative code listings. The observation vector set, the reward

function, and the two-stage course weights are presented in Figure 4-6. All the codes are directly excerpted from the training script.

```
def _build_obs(self, raw):
    load, t_ms, p_ms, nox, rq, ofa = raw
    t_tgt, p_tgt = get_steam_target(load)
    cur_err = np.array([t_ms - t_tgt,
                        p_ms - p_tgt,
                        nox - 35.0], dtype=np.float32)
    d_err = cur_err - self.prev_err
    obs = np.concatenate([
        raw,
        self.prev_raw,
        self.prev_action.copy(),
        self.err_int.copy(),
        d_err,
    ]).astype(np.float32)
    return obs
```

Figure 4. Observation vector set code

```
    nox_n    = np.clip(nox / 35.0, 0.1, 5.0)
    r_nox    = -w["nox"] * float(np.log(nox_n + 0.1))

    t_err_n  = np.clip(abs(t_steam - t_tgt) / 5.0, 0.0, 2.0)
    r_tsteam = -w["t_steam"] * float(t_err_n)

    p_err_n  = np.clip(abs(p_steam - p_tgt) / 2.0, 0.0, 2.0)
    r_psteam = -w["p_steam"] * float(p_err_n)

    rq_pen   = float(np.clip((0.86 - rq) / 0.04, 0.0, 1.5))
    r_rq     = -w["rq"] * rq_pen

    r_act      = -w["act"]      * float(np.mean(np.square(self.smoothed_action)))
    r_delta    = -w["delta"]    * float(np.mean(np.square(delta_action)))
    r_residual = -w["residual"] * float(np.mean(np.abs(self.smoothed_action)))

    nox_int_pen = float(np.clip(abs(self.err_int[2]) / 100.0, 0.0, 1.0))
    r_nox_int   = -w["nox"] * 0.3 * nox_int_pen

    reward = (r_nox + r_tsteam + r_psteam + r_rq +
              r_act + r_delta + r_residual + r_nox_int)
```

Figure 5. Reward function code

```
def get_reward_weights(global_steps):
    if global_steps < 1_000_000:
        return dict(t_steam=2.0, p_steam=0.5, nox=1.0, rq=1.0,
                    act=0.1, delta=0.5, residual=CONFIG["residual_coef"])
    else:
        return dict(t_steam=2.0, p_steam=0.5, nox=3.0, rq=0.8,
                    act=0.2, delta=0.8, residual=CONFIG["residual_coef"])
```

Figure 6. Two-stage course weight code

3 Results and discussion

The validation of the proposed framework is divided into three levels, corresponding to the three components contributed in this paper. 1. Assesses the agreement between the CFD-PINN-ROM model and the CFD. 2. Assesses the agreement between the DCS-driven adaptive correction layer and 5 days of plant operating DCS data. 3. Assesses the performance of the PPO agent against an expert-tuned FF and PID baseline across seven cases.

3.1 CFD-PINN-ROM accuracy on the CFD database

The CFD-PINN-ROM model is validated against CFD data from its source and evaluated across the entire training range. The accuracy of the flue gas temperature at the furnace outlet T_{exit}, the combustion ratio RQ, and the total furnace heat release Q_{tot} is summarized in Table 1. These variables will be used as inputs for the steam-side model and the reinforcement learning reward function.

Table 1. Accuracy of the CFD-PINN-ROM

Variable	Unit	MAPE/%	R^2
T_{exit}	K	0.7	0.9999
RQ	/	0.9	0.9995
Q_{tot}^{al}	MW	1.5	0.9980

The combustion variables were reproduced with an MAPE of less than 1% and R^2 close to 1, confirming that the PINN model maintains accuracy comparable to CFD within the deep peak shaving. This accuracy stems not only from the network capacity but also from the physics-based loss function. Energy balance and monotonicity penalty terms prevent the network from deviating in regions where CFD training samples are sparse, particularly in low-load boundary regions where deep peak-reduction runs are concentrated.

3.2 DCS-driven adaptive correction

The CFD-PINN-ROM is calibrated based on simulation results. However, it struggles to account for long-term drifts that occur only in actual plant operations, such as fouling of heat transfer surfaces and coal quality changes. Five days of plant DCS data were used to evaluate the two-mode DCS-driven correction layer. Each variable measures the residual between the ROM output and the DCS measurement, including samples taken before calibration (raw ROM) and after calibration (ROM + DCS layer), arranged in chronological order. Effect of the DCS adaptive correction is listed in Table 2.

Table 2. Effect of the DCS adaptive correction

Variable	ROM MAPE	Corrected MAPE	RMSE
T_{exit}	3.615%	0.153%	2.29 K
RQ	7.310%	1.452%	0.017

The magnitude of the residuals between the original ROM model and the real-time DCS data is greater than that between the residuals and the actual CFD values. This result confirms that the discrepancy between simulation and measured data cannot be overlooked in any practical deployment. Secondly, the inclusion of a correction term ensures that both types of residual are reduced, which is consistent with the inherent accuracy of thermocouple measurements.

3.2 Modelica dynamic validation

Following the integration of a calibrated combustion simulator and a DCS calibration table, the entire Modelica plant model was validated against the real-time DCS under two operating modes: a five-day steady-state operation and a dedicated ramp-up operation from 320 to 600 MW. The first was a steady-state operation, and the second was a dedicated ramp-up operation from 320 to 600 MW. A total of six observed variables were tracked, distributed across the combustion side (T_{exit}, RQ), the steam-water side (T_{ms}, p_{ms}, m_{ms}), and the flue gas path (temperature at the horizontal flue inlet, $T_{hor,}$ and at the

DOI
10.3384/ecp223217

economiser outlet, T_{eco}). The numerical accuracy is summarised in Table 3.

Table 3. Modelica dynamic model validation accuracy

Variable	Regime	MAPE%	RMSE
T_{exit}/K	5-day steady	0.107	1.81
RQ	5-day steady	0.979	0.01
T_{ms}/K	5-day steady	0.230	2.64
P_{ms}/bar	5-day steady	2.102	5.04
m_{ms}/kg·s^{-1}	5-day steady	1.509	7.64
T_{hor}/K	5-day steady	2.915	33.77
T_{eco}/K	5-day steady	1.738	14.13
T_{ms}/K	320 to 608 MW ramp	0.141	1.30
m_{ms}/kg·s^{-1}	320 to 608 MW ramp	1.600	7.68
P_{ms}/bar	320 to 608 MW ramp	2.575	6.28

The MAPE for the main steam temperature was 0.23% during steady-state and 0.14% ramp-up operation. Comparison of simulation results and measured data of main parameters under ramp up operation was shonw in Figure 6. The MAPE for the horizontal flue inlet temperature and the economiser outlet temperature were 2.9% and 1.7%, respectively. Flue gas temperatures depend on downstream convective heat exchangers, the surface conditions of which vary more rapidly over time than the time taken for the DCS correction layer to absorb these changes. Absolute deviations of 14–34 K remain within the operational tolerance limits for emissions control monitoring.

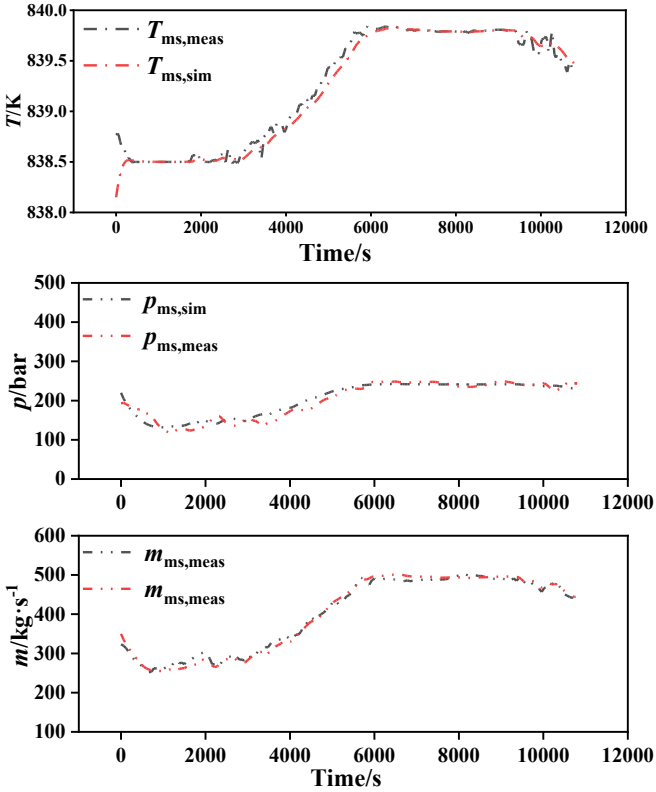


Figure 7. Comparison of simulation results and measured data of key parameters under ramp-up operation.

3.3 Evaluation of RL performance

The Recurrent PPO agent is compared against three benchmark controllers running on the same Modelica plant FMU. The expert-tuned feedforward schedule (PureFF), a manually tuned PID, and a gain-scheduled PID (GS-PID) interpolating between seven scenarios. The two PID variants share identical setpoints, anti-windup limits, and command saturation to ensure a fair comparison. The main loop for T_{ms} and p_{ms} remaining active throughout all experiments.

Seven scenarios are used to cover the 300–600 MW range obtained during training. Steady-state operation at 320 (steady low), 400 (steady mid) and 580 (steady high) MW for 600s. A 400 s gradual ramp-up/down process from 320 MW to 480 MW, followed by a 200 s steady-state period. A representative deep peak-shaving cycle, from 480 to 320 MW and back to 480 MW, with a total duration of 1050 s. A load change of 60 MW is applied every 120 s, covering the range of 340–460 MW. Each scenario is run only once in each controller, using a deterministic strategy with identical initial conditions and a fixed FMI warm-up time of 80 s. Statistical data on NOx and RQ for seven cases are summarised in Table 4.

Steady-state operation. Under steady-state operating conditions, all four controllers operated within the same range, with NOx mean concentrated within 5 mg/Nm³ and RQ mean converging to 0.82–0.87. At constant load, the plant gain remained constant, the open-loop FF mapping met design requirements and was accurate, and any feedback effects were minimal. The steady-state results are primarily used as regression tests to confirm that these controllers do not cause system instability at fixed operating points. PPO also achieved the highest RQ value at steady-state low load (0.87), indicating that its residual deviation design does not interfere with the carefully tuned open-loop response.

Dynamic operation. The four dynamic scenarios—Ramp-up, Ramp -down, Deep peak, and Staircase—were used to conduct practical tests of the deep peak controller. During transients, the NOx p95 is considered to determine compliance with ultra-low emission standards. Compared to manually tuned PID, PPO reduces NOx p95 by 19.2%–22.7% and Nox std by 32.7%–43.2%. Compared to GS_PID, the corresponding reductions were 21.0%–24.9% and 36.0%–46.9%, respectively. This gap was greatest in the staircase scenario. Rapid alternating load changes in the staircase scenario amplify the failure modes of scalar feedback control, whereas the PPO strategy remains unaffected.

The PPO and PureFF controllers performed comparably in terms of dynamic average NOx, confirming that the carefully designed open-loop response was not compromised by the residual deviation strategy. In all four dynamic scenarios, the NOx p95 of PPO agent was equal to or lower than that of PureFF, and it consistently delivered a higher RQ, indicating improved combustion efficiency under transient conditions. Among the four

controllers, PPO agent is the only one that achieves low average NOx, low NOx variance, low NOx peaks, and high combustion efficiency simultaneously under deep peak-shaving transients. While both PID variants can reduce average NOx relative to PureFF in certain cases, this comes at the cost of significantly increased variance and peaks—which runs counter to the requirements of ultra-low emission compliance. Although PureFF performs well in terms of average values, its closed-loop response capability is 0, making it unable to handle sudden events.

Table 4. NOx and *RQ* statistics across the seven evaluation scenarios

Scenario	Controller	NOx mean	NOx std	NOx p95	*RQ* mean	*RQ* min
Steady low	PureFF	27.34*	0.00*	27.34*	0.86	0.86
	GS_PID	26.83	0.7	27.01	0.43	4.23
	PID	29.80	1.30	30.16	0.86	0.86
	PPO	30.18	1.49	30.11	0.87*	0.83
Steady mid	PureFF	39.42	0.00*	39.42	0.82	0.82
	GS_PID	35.73	0.71	37.52	0.67	1.9
	PID	40.20	0.71	40.22	0.82	0.82
	PPO	35.73*	0.23	37.52*	0.83*	0.83
Steady high	PureFF	40.12*	0.00*	40.12*	0.83	0.83
	GS_PID	47.95	1.29	48.31	0.93	0.56
	PID	49.20	1.45	49.59	0.83	0.83
	PPO	41.39	0.14	41.39	0.83	0.83
Ramp up	PureFF	36.44	5.65	42.72	0.83	0.82
	GS_PID	40.12	9.54	54.06	0.66	3.28
	PID	41.01	9.01	53.41	0.84*	0.82
	PPO	35.25*	5.31*	42.33*	0.84*	0.83*
Ramp down	PureFF	35.81	5.81	42.72	0.84*	0.82
	GS_PID	38.75	9.89	54.05	0.76	1.99
	PID	39.41	9.25	52.85	0.84*	0.82
	PPO	34.39*	5.25*	42.33*	0.84*	0.83*
Deep peak	PureFF	36.83	5.42	42.77	0.83	0.82
	GS_PID	41.02	9.03	54.21	0.75	2.83
	PID	41.60	8.75	53.14	0.83	0.82
	PPO	35.81*	5.11*	42. 63*	0.84*	0.83*
Staircase 340 to 460 MW	PureFF	37.01	5.04	45.31	0.83	0.82
	GS_PID	40.47	7.87	57.30	0.83	0.82
	PID	40.24	7.49	55.71	0.83	0.82
	PPO	36.17*	4.38*	43.04*	0.84*	0.83*

Best value per row marked with an asterisk (*). Units: NOx in mg/Nm³; RQ is dimensionless. Nox p95: the 95th percentile of Nox.

3.4 Deep peak-shaving case study

Figure 8 shows the closed-loop NOx and *RQ* response under the deep peak-shaving scenario. In the deep peak shaving, a 160 MW down-ramp and a 160 MW up-ramp over 600 s and is the most demanding of the seven evaluation cases. This is the factor that most clearly distinguishes the operating modes of the four controllers. The ramp-down operation triggered a transient NOx excursion at approximately t ≈ 200s. The PID and GS_PID overshot to 57.3 and 58.1mg/Nm³ The PPO agent limited the same transient to 44.2 mg/Nm³, a 23% reduction relative to the better of the two PID variants. During the low-load (300 < t < 600 s), the two PID exhibited a sustained limit cycle of ±8 mg/Nm³ around its operating point, while the PPO converged to ±2 mg/Nm³. Deviation statistics confirm that during this brief process, the PPO utilized the full range of actuator authority—all three deviation channels were saturated at their control limits (±6 kg/s, ±80 kg/s, ±0.15)—while the PID utilized on average only 58% of the available range, consistent with its slower closed-loop response. During transient processes, PPO continuously improves two metrics that are critical to practical operation: NOx std and NOx p95.

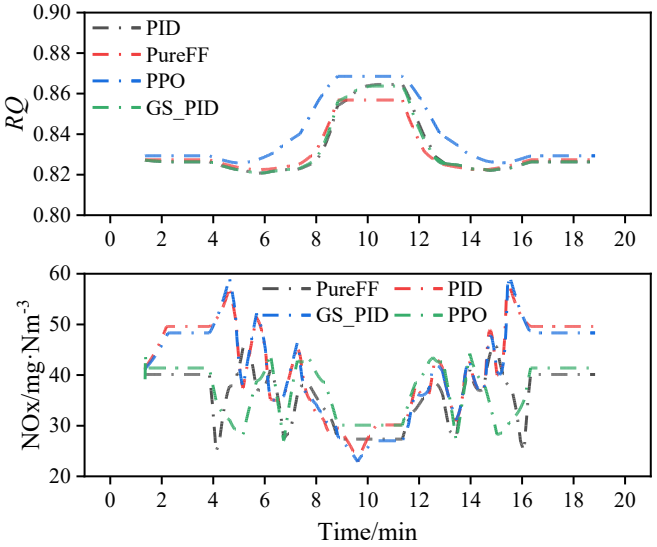


Figure 8. Closed-loop Nox and *RQ* response under different controllers during the deep peak shaving operation

DOI
10.3384/ecp223217

The strategy learned by PPO does not replace the existing feedforward control, but rather makes precise corrections in the areas where both the FF and PID structures are weakest. This situation can be regarded as a Pareto improvement rather than strict dominance. PPO is able to achieve the same average emissions as expert FF control, while reducing peak emissions and improving combustion efficiency. Under ultra-low emission standards, peak emissions determine the margin of compliance, and a 0.5% improvement in combustion efficiency translates directly into quantifiable economic benefits. Achieving simultaneous improvements in both these metrics is difficult to achieve through open-loop control design, and no scalar feedback rule can be implemented without the aid of specific multi-objective algorithms.

FMI provides a suitable standardised interface for coupling physics-based dynamic models with reinforcement learning controllers. Within the current framework, the FMI 2.0 co-simulation interface encapsulates the Modelica boiler model and its numerical solver within an FMU, whilst exposing only the necessary observations and residual control inputs to the external reinforcement learning agent. The sequence of interactions in FMI—setting inputs, advancing the FMU and obtaining outputs—maps naturally to the observation–action–transition cycle in reinforcement learning. The communication steps adopted in this study are also synchronised with the decision intervals of the recursive PPO agent. This separation allows the boiler model to be recalibrated or replaced without modifying the reinforcement learning implementation, thereby enhancing the model's portability and maintainability. However, FMI is a general-purpose standard for model interoperability and co-simulation, rather than an environment standard designed specifically for reinforcement learning. Consequently, RL-specific features—including the definition of observation and action spaces, reward functions, round termination, random initialisation, and vectorised parallel rollouts—are not standardised and must be implemented by the external RL environment. Although FMI 2.0 provides model reset and optional FMU state serialisation capabilities, efficient snapshot-driven round initialisation cannot be guaranteed across different FMU implementations. FMI 3.0 offers more advanced data exchange and synchronisation capabilities, but RL-specific episode and training semantics remain outside the scope of the standard. Consequently, combining FMI with an external RL wrapper provides a practical division of responsibilities for physics-based RL control.

4 Conclusion

This paper proposes a Modelica-based digital twin framework designed to reduce NOx emissions from a 600 MW coal-fired boiler operating in deep peak shaving. The core engineering contribution lies in a co-simulation architecture that integrates modeling and control strategy training into a single FMI 2.0 co-simulation FMU and interfaces with a cyclic PPO agent. This decoupling of boiler modeling from control policy training allows the FMU to be independently re-calibrated as plant conditions evolve, without retraining the agent.

A simplified combustion model based on CFD-PINN achieves high accuracy while remaining compatible with reinforcement learning training in terms of per-evaluation cost. Replacing the distributed thermo-hydraulic library (ClaRa, ThermoSysPro) with a three-state DAE submodel reduced the number of FMU states by one to two orders of magnitude, enabling the completion of over one million RL training steps. The Smooth(1,·) expression in the Modelica equations prevents the DASSL integrator from restarting at table interpolation breakpoints. The MAPE for the T_{ms} was 0.23% during steady-state and 0.14% ramp-up operation. The MAPE for the T_{hor} and the T_{eco} were 2.9% and 1.7%, respectively.

An intelligent RL agent utilizing the Recurrent PPO algorithm is employed for NOx emission optimization. Validated in seven scenarios spanning 300–600 MW, this strategy reduces the 95th-percentile NOx by 19%–23% and emission fluctuations by 32.7%–43.2% across four dynamic scenarios relative to both PID and GS_PID baselines, while maintaining combustion efficiency. Under the deep peak shaving, the strategy reduced the transient NOx peak under PID control from 57.3mg/Nm³ to 44.2 mg/Nm³, while successfully maintaining the restricted air ratio above the 0.80 safety threshold.

The broader contribution to the Modelica and FMI community lies in demonstrating a complete, reproducible workflow — from CFD database generation and PINN training, through Modelica/Dymola FMU compilation, to Python-based RL training via fmpy — for integrating scientific machine learning with equation-based plant models in a demanding industrial control application. The CFD-PINN model makes RL training feasible and the DCS layer ensures that the trained policy is robust to inherent fluctuations in actual operation.

Acknowledgements

This work has been supported by the Key R&D Program of Shanxi Province (grant numbers 202302090301005).

References

Beck, M., Poppel, K., Spanring, M., and Hochreiter, S.（2024). xLSTM: Extended Long Short-Term Memory. ArXiv, abs/2405.04517.

Cai, S., Mao, Z., Wang, Z., and Karniadakis, G. E. (2021). Physics-informed neural networks (PINNs) for fluid mechanics: a review. Acta Mechanica Sinica, 37, 1727-1738.

Chen, X., Zhong, W. and Li, T. (2023). Fast prediction of temperature and chemical species distributions in pulverized coal boiler using POD reduced-order modeling for CFD. Energy, 276, 127663.

Fu, L., Gao, Y., Cheng, S., and Liu, J. (2025). An optimal model predictive control based on Hammerstein model considering fatigue load reduction for wind turbines. International Journal of Electrical Power & Energy Systems, 172, 111132.

Gong, J., Qu, Z., Zhu, Z. and Xu, H. (2025). Parallel TimesNet-BiLSTM model for ultra-short-term photovoltaic power forecasting using STL decomposition and auto-tuning. Energy, 320, 135286.

Huang, C., Zheng, Y., Zhao, H., Zhu, J., and Peng, T. (2025). Intelligent Deep Learning Modeling and Multi-Objective Optimization of Boiler Combustion System in Power Plants. Processes, 13, 2340.

Huang, J., Zheng, P., Hu, X., Chen, W and Han, Y. (2025). Power generation efficiency and resources saving of the hydropower industry using the extended data based convolutional neural network. Renewable Energy, 243, 122541.

Jin, W., Si, F., Li, H. and Wang, Y. O. (2023). Numerical study on the effects of primary air ratio on ultra-low-load combustion characteristics of a 1050 MW coal-fired boiler considering high-temperature corrosion. Applied Thermal Engineering, 221, 119811.

Liu, K., Wang, C., Wang, L., Liu, B., and Che, D. (2023). Dynamic performance analysis and control strategy optimization for supercritical coal-fired boiler: A dynamic simulation. Energy, 282, 128712.

Liu, X. and Bansal, R. C. (2014). Integrating multi-objective optimization with computational fluid dynamics to optimize boiler combustion process of a coal fired power plant. Applied Energy, 130, 658-669.

Liu, X. and Kong, X. (2014). Nonlinear Model Predictive Control for DFIG-Based Wind Power Generation. IEEE Transactions on Automation Science and Engineering, 11, 1046-1055.

Pleines, M., and Preuss, M. (2022). Generalization, Mayhems and Limits in Recurrent Proximal Policy Optimization. ArXiv, abs/2205.11104.

Raissi, M., Perdikaris, P. and Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 378, 686-707.

Sage, M., and Zhao, Y. F. (2023). Optimal Economic Gas Turbine Dispatch with Deep Reinforcement Learning. IFAC-PapersOnLine, 56, 10039-10044.

Spielbrg, S., Tulsyan, A., and Bhunshan, R. (2019). Toward self-driving processes: A deep reinforcement learning approach to control. AIChE Journal, 65, e16689.

Wang, B., Sun, H., Liu, J., and Liu, Z. (2025). High-precision dynamic simulation of a 600 MWe supercritical boiler system for digital twin application. Energy Conversion and Management, 346, 120452.

Wang, C., Chen, X., Liu, B., and Che, D. (2025). Dynamic analysis for ultra-supercritical boiler water wall considering uneven heat flux distribution and combustion instability. International Journal of Heat and Mass Transfer, 236, 126269.

Wang, C., Xu, J., Xu, K., and Xiang, J. (2025). Real-time prediction and optimization of NOx emissions using artificial intelligence and online combustion data. Fuel, 391, 134836.

Wang, T., Chen, X. and Zhong, W. 2024. Air distribution and coal blending optimization to reduce slagging on coal-fired boiler water wall based on POD reduced order modeling for CFD. Fuel, 357, 129856.

Wei, T., Yan, W. and Zhu, Q. Deep reinforcement learning for building HVAC control. 2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC), 18-22 June 2017 2017. 1-6.

Wu, C., Wang, C., and WANG, Z. (2025). Flexible peak shaving in coal-fired power plants: A comprehensive review of current challenges, recent advances, and future perspectives. Energy, 327, 136446.

Wu, X., Shen, J., Li, Y. and Lee, K. Y. (2015). Steam power plant configuration, design, and control. WIREs Energy and Environment, 4, 537-563.

Yan, X., Ji, X., Meng, Q., and Lei, Y. (2024). A hybrid prediction model of improved bidirectional long short-term memory network for cooling load based on PCANet and attention mechanism. Energy, 292, 130388.

Zhou, D., Duan, Q., and BI, S. (2026). Fast prediction of combustion power field in 600 MW coal-fired boiler based on CFD-AI. Case Studies in Thermal Engineering, 77, 107598.

A Modelica-based Fluid Library for Small Modular Reactors: Design and Verification

Sixin Liu[1] Hao Hong[1] Hong Lyu[1]

[1]State Key Laboratory of Nuclear Power Safety Technology and Equipment, China Nuclear Power Engineering Co., Ltd., Shenzhen, Guangdong, 518116, China, sixin.liu@foxmail.com

Abstract

While Modelica has become a standard for multi-domain modeling in various complex engineering fields, professional and systematic model libraries specifically tailored for the nuclear power sector remain scarce. To bridge this gap, this paper presents the development of a specialized Modelica fluid component library for Small Modular Reactor (SMR) systems. Serving as a critical module within the broader CGN_NHR library, this framework bridges the gap between fundamental fluid mechanisms, equipment-level modeling, and system-level assembly. We detail the library's architecture, design philosophy, and the mathematical formulation of its core components. To verify its validity and reliability, a system-level model was constructed and benchmarked against the industry-standard RELAP5 model. Comparative results confirm that the proposed library offers high efficiency and accuracy, establishing a robust technical foundation for the rapid design, simulation, and verification of nuclear power systems.

Keywords: Modelica, Fluid Library, Small Modular Reactor

1 Introduction

Nuclear power systems are complex engineered systems that integrate physical processes across multiple domains, including fluid dynamics, thermal-hydraulics, reactor physics, control and protection, electrical engineering, and mechanical engineering(Zhang 2025). Modeling and simulation technologies play a vital role in the design, analysis, and optimization of these systems. They enable the reproduction and prediction of system behavior under a wide range of steady-state and transient operating conditions within a virtual environment, thereby substantially reducing the cost and risk of experimentation and research and development, while accelerating design iteration and validation processes. With the advancement of digital technologies, an increasing number of industries have begun to undergo digital transformation by introducing the concepts and methods of Model-Based Systems Engineering (MBSE). Guided by systems engineering methodologies, MBSE supports the forward design and development of new products through simulation models. However, current modeling and simulation practices for nuclear power systems still widely rely on a multi-tool co-simulation approach based on dedicated program interfaces. Although this method enables the integration of specialized models from different disciplines, its shortcomings are also pronounced: inconsistent interface standards lead to high integration complexity and debugging difficulties; strong coupling among subsystem solvers results in numerical instability during simulation; and poor model reusability makes it difficult to rapidly adapt to new system configurations. These limitations reveal a gap between current practice and the digital transformation goals advocated by MBSE, particularly the principles of a "single source of truth" and "models spanning the entire lifecycle."

Modelica, an equation-based and object-oriented unified modeling language for multidomain systems, offers a new approach to addressing the aforementioned challenges. Through its acausal modeling paradigm and physical-port connection mechanism, it enables the natural coupling of components from different domains at the equation level, thereby fundamentally avoiding issues related to interface protocols and solution sequencing. This technology has been successfully used to develop extensive standardized model libraries in complex equipment domains such as automotive and aerospace engineering, and has become a key enabling technology for implementing MBSE and constructing system-level digital twins in these industries(Jiang et al. 2025). However, in the nuclear power field, specialized and systematic Modelica model libraries, particularly those tailored to the characteristics of Small Modular Reactor (SMR), remain scarce. Existing general-purpose fluid libraries do not adequately cover the equipment types and working-fluid states unique to nuclear energy systems, resulting in a lack of reusable physical model components required for MBSE implementation in the nuclear domain. To fill this gap and respond proactively to the nuclear industry's demand for digitalization and MBSE-driven transformation, this paper proposes and develops a Modelica fluid component library for SMR systems. It should be emphasized that this fluid library was not developed as an isolated tool for a single application, but rather as an integral part of the CGN_NHR(Hong et al. 2026) modeling and simulation library for SMR systems, where it plays a central role in linking fundamental fluid mechanisms, equipment-level model development,

and system-level loop assembly. The main contribution of this work lies not only in the provision and validation of a series of fundamental fluid model components, but also in the establishment of a set of reusable, composable, and traceable fluid model assets consistent with MBSE principles, thereby providing a foundation for the subsequent unified integration of thermal-hydraulic, control, and equipment models.

2 Overall Design

The fluid library developed in this study is founded on the overarching principles of mechanism-based modeling, modular reuse, hierarchical decoupling, and engineering applicability. While ensuring consistency with the three fundamental conservation laws of mass, energy, and momentum, it also balances system-level simulation efficiency with the capability to represent component-level physical mechanisms. Within a unified multidisciplinary modeling framework, the fluid library primarily serves as a bridge for transferring fundamental fluid mechanisms to component-, equipment-, and system-level models. At the modeling level, it ensures consistency in conservation relationships and thermophysical property calls; at the software level, it realizes abstract base classes, generic interfaces, and component encapsulation; and at the engineering level, it supports the rapid assembly, parametric configuration, and simulation-based validation of typical system loops. Therefore, the fluid library is not merely a collection of components, but rather a foundational modeling platform that supports equipment modeling and system coupling within the unified model library for SMR systems, providing a common fluid-modeling basis for subsequent control analysis, scheme comparison, and library-wide extension.

2.1 Design Philosophy and Advantages

In terms of design philosophy, the fluid library utilizes unified conservation equations as its central theme. For instance, both pipe and volume models are constructed around mass, energy, and optional momentum balances, achieving cross-component coupling through unified port semantics. This approach ensures that models retain interpretable physical meanings even after system assembly, effectively avoiding the pitfall where local models are valid but fail at the system level.

The library also adopts a distinct hierarchical organization, characterized by clear and transparent logical structures between layers. Furthermore, it emphasizes parameterization and model family management. A wide range of components offer structural, initialization, and state selection parameters, allowing the same model to switch between tasks such as steady-state solving, transient disturbance analysis, and control analysis. Leveraging the object-oriented features of Modelica, the library achieves the design goal of "invariant governing equations with replaceable sub-models." Compared to the ad-hoc modeling approach often associated with generic fluid libraries, the advantages of the proposed fluid library are manifested in the following aspects:

- Strong Physical Consistency. The unified port variable transfer mechanism (mass flow, enthalpy flow, and component flow) combined with boundary legitimacy checks ensures that component connections satisfy conservation constraints, thereby reducing the risk of non-physical connections.
- High Efficiency in Reuse and Extension. Fundamental models can be inherited by various devices, while complex elements can be constructed via composition. Meanwhile, the replaceable mechanism reduces model coupling. This hierarchical and modular design significantly enhances maintainability, drastically reduces redundant modeling efforts, and facilitates collaborative development and future feature expansion.
- Prominent Engineering Applicability. The parameter organization, port configuration, and component granularity are closely aligned with the fluid analysis workflows of SMRs. This alignment facilitates rapid assembly, parameter tuning, and validation within system-level models.

2.2 Composition of the Fluid Library

In terms of architecture, the fluid library follows a development logic of “Fundamental Mechanism Layer – Component Implementation Layer – System Assembly Layer.” It comprises modules for interfaces, boundary conditions, pipelines, volumes, valves, mechanical equipment, connectors, and sensors. Specifically, interfaces, thermophysical properties, pressure-drop models, and heat-transfer models constitute the Fundamental Support Layer; pipelines, volumes, valves, and pumps form the Component Implementation Layer; and various typical loop models represent the System Assembly Layer. This organizational scheme is consistent with the hierarchical modeling philosophy of the CGN_NHR library and facilitates a continuous mapping from fundamental equations to system-level models. Specifically:

The interface module is primarily responsible for unifying connection semantics. Based on inheritance from standard Modelica connectors, it defines standardized fluid connectors and mechanical connectors, while consistently constraining the semantic definitions and flow conventions of variables such as pressure, specific enthalpy, mass flow rate, and species mass fraction at the ports. In this way, physical closure requirements can be ensured during component assembly. Unified interfaces are not only a means of implementing model connectivity, but also a key prerequisite for transmitting consistent physical semantics from fundamental models to component-, equipment-, and system-level models, thereby providing the basis for the coupled solution and scalable reuse of multidisciplinary models.

DOI
10.3384/ecp223227

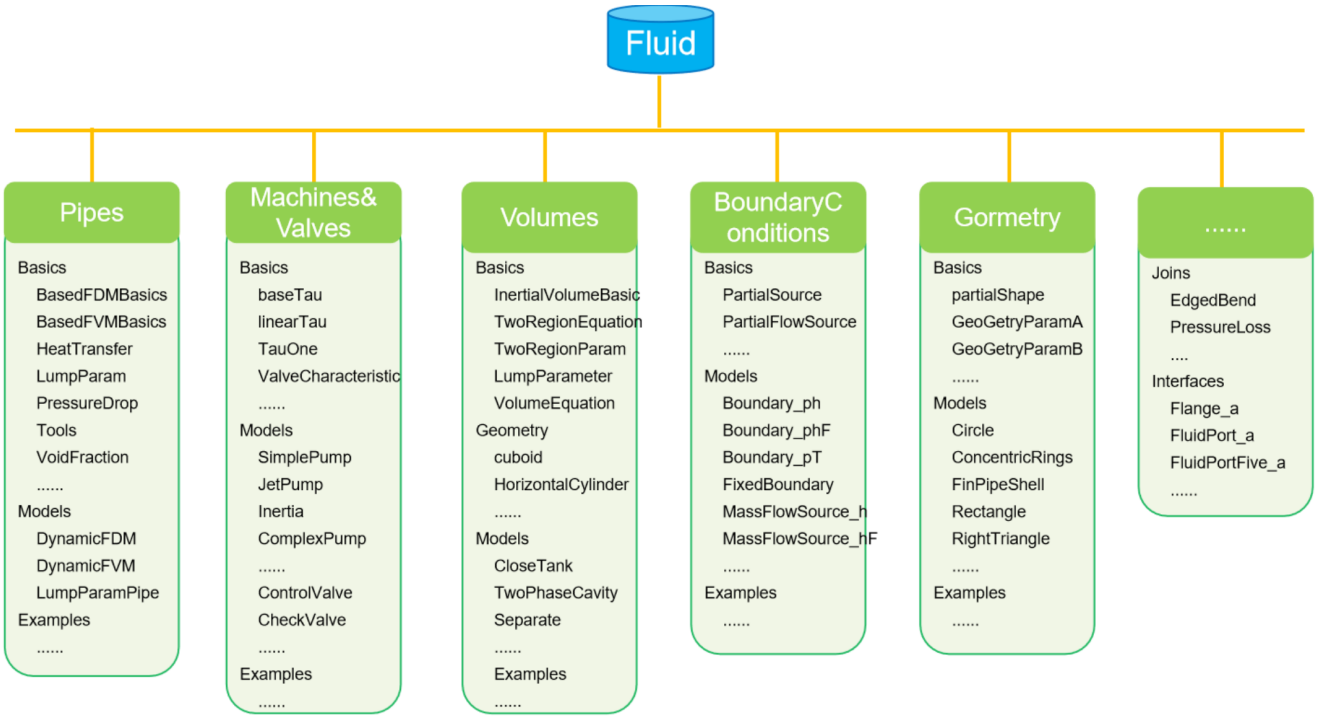


Figure 1. Architecture of the Fluid Library.

The boundary condition module encompasses models for fixed pressure, temperature, enthalpy, and mass flow rate. It supports dual input modes: parameter-based specification and signal-driven control.

The pipeline module employs a two-tier architecture comprising fundamental mechanism models and engineering components. The mechanism models rely on a unified two-port abstract framework that integrates geometric parameters, fluid states, and conservation laws, while offering modular sub-models for pressure drop, heat transfer, and void fraction. Building on this framework, engineering components are categorized into lumped-parameter models and distributed-parameter models utilizing the Finite Volume Method (FVM) and Finite Difference Method (FDM). Furthermore, pipeline models are integrated with geometric structures, pressure-drop correlations, and heat-transfer coefficient models rather than being constructed in isolation. This design allows for the flexible substitution of pipe types, boundary conditions, and resistance characteristics without modifying the governing equations, significantly enhancing the library's adaptability to the complex flow circuits of SMR systems.

The volume module consists of several parts, including fundamental equations, parameters, geometry, and components. The fundamental part is centered on the mass and energy conservation of volume control bodies, and uniformly provides dynamic/static equilibrium switching, working-fluid state definition, and coupling of heat-exchange terms. The parameter part specifies key parameters for vapor-liquid two-phase processes, including evaporation, condensation, initial liquid level, and heat capacity. The geometry part provides structural quantities such as total volume, liquid-level-related volume partitioning, and heat-transfer area. The component part encapsulates engineering models such as inertial volumes, two-phase volumes, dynamic steam drums, closed tanks, and steam-water separators, and achieves coupling with the overall system through multiple fluid ports, heat ports, and liquid-level outputs.

The valve and mechanical equipment module implements typical actuation and energy-conversion devices, including control valves, switching valves, check valves, pumps, and turbines. On the valve side, based on two-way and three-way topologies, pressure-drop-flow relationships are established through opening inputs, flow coefficients, or resistance coefficients, and functions such as regulation, shutoff, check, and flow splitting are realized under the constraints of mass, energy, and species transport. On the mechanical-equipment side, pumps, steam turbines, jet pumps, and shaft-inertia components are taken as the core elements, and the coupled conversion between fluid energy and mechanical work is described through head curves, isentropic efficiency, or torque characteristic functions, thereby completing processes such as pressurization, power generation, and jet entrainment.

The connector module comprises base classes for geometric parameters and typical connection elements. This module centrally defines geometric-hydraulic attributes, including discretized nodes for pipe segments, cross-sectional area, roughness, flow-path elevation, and pressure drop characteristics. Based on these definitions, models for straight pipes, local resistances, filter assemblies, and standard elbows are constructed.

The sensor module mainly includes sensors for mass

flow rate, volumetric flow rate, pressure, differential pressure, and two-port temperature, all of which are implemented by inheriting from the fluid sensor models in the Modelica Standard Library(Modelica Association 2023).

As shown in Figure 1, the fluid library is organized into a specific architecture comprising various functional parts.

2.3 Logical Architecture

The fluid library proposed in this paper employs a bottom-up, four-tier architecture, as illustrated in Figure 2. The first tier is the Unified Interface Layer, which serves as the syntactic foundation for cross-model interconnection by standardizing port definitions, variable naming conventions, and connection constraints. The second tier is the Fundamental Mechanism Layer. Acting as the mathematical core of the library, it centralizes the implementation of mass, energy, and momentum equations, while integrating thermophysical property functions, pressure drop correlations, and heat transfer sub-models. The third tier is the "Component Model Layer." By utilizing inheritance and composition, this layer maps the fundamental mechanisms onto specific physical devices, forming ready-to-use engineering components such as pipes, volumes, valves, and pumps. The fourth tier is the "System Assembly Layer." This layer facilitates the construction of loop networks, the application of boundary conditions and control signals, and the output of observable metrics.

This hierarchical structure achieves the design goals of stable interfaces, replaceable mechanisms, reconfigurable components, and scalable systems. Consequently, new devices or correlations can be incorporated on demand without reconstructing the existing framework. The logical flow is unidirectional and constraining: interfaces define mechanism boundaries, mechanisms govern component behavior, components aggregate into system models, and the system feeds back design parameters. This architecture establishes a seamless pathway from mathematical modeling to engineering application. It ensures that underlying equations are maintained centrally for maximum reuse, while allowing upper-level components to undergo rapid iteration with invariant interfaces—significantly reducing the cost of system model reconstruction.

2.4 Dependencies and Replaceability

Building on the aforementioned logical architecture, the dependency paradigm has shifted from "direct component coupling" to a strategy of "downward dependence on abstraction and upward exposure of unified interfaces." This transformation effectively prevents the structural complexity associated with cross-layer cross-referencing and preserves stable boundaries for future expansion.

In terms of implementation, the dependencies within the library are primarily governed by three mechanisms. First, Inheritance Dependencies are utilized to reuse equation skeletons and common parameter frameworks. For instance, pipe models inherit from fundamental two-port and fluid equation models, while volume models extend

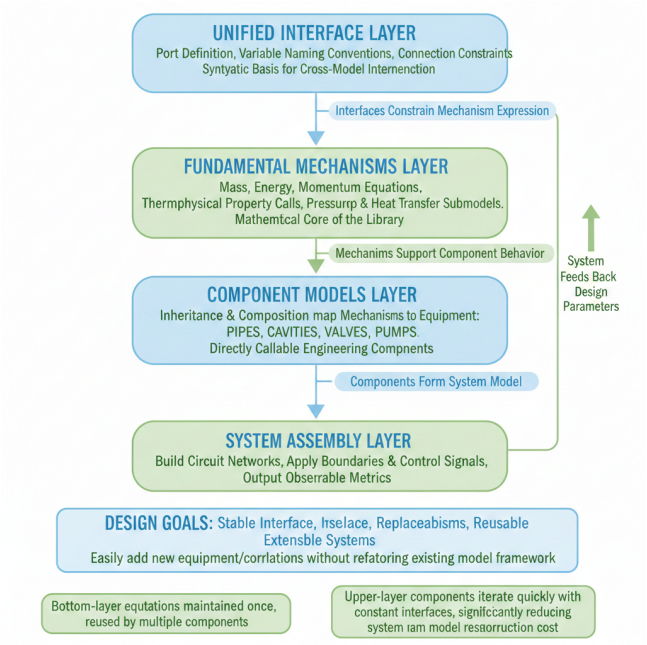


Figure 2. The Four-Tier Hierarchical Architecture.

fundamental volume and two-phase parameter equations. Second, Composition Dependencies are employed to construct device-level functional units. For example, the filter is assembled by combining an inertial volume with a local pressure loss module, and a three-way valve is formed by integrating two control valves with an internal volume unit. This approach realizes "functional assembly" rather than repetitive modeling. Finally, Parameter Redeclaration Dependencies are used to explicitly propagate working fluid and parameter semantics along the composition chain, ensuring strict consistency between sub-models and parent models regarding medium definitions and port variables.

The replaceability mechanism is central to the library's extensibility. By leveraging the Modelica *replaceable ... constrainedby ...* construct, the library enables the dynamic switching of specific sub-model implementations—such as medium packages, heat transfer correlations, pressure drop models, and mechanical characteristic functions—while strictly maintaining unified interfaces and the governing equation structure. This mechanism not only enhances component adaptability but also establishes a robust interface foundation for seamless integration with thermal, control, and equipment models within the broader multi-disciplinary framework for SMRs. Consequently, fluid models can be rapidly reconfigured to accommodate varying design phases, operating conditions, and equipment specifications.

Collectively, dependency management and replaceability mechanisms establish an architectural paradigm characterized by a "stable backbone and variable local elements." On one hand, conservation equations, port semantics, and initialization frameworks remain consistent, ensuring comparability across diverse components and mod-

DOI
10.3384/ecp223227

els. On the other hand, empirical correlations and equipment characteristics can be flexibly substituted based on operating conditions and data sources, thereby facilitating scheme comparison, sensitivity analysis, and model verification. As a quintessential application of object-oriented programming principles, this approach significantly reduces refactoring costs during library iteration and improves long-term maintainability. Ultimately, it provides unified technical support for small reactor fluid systems, ranging from rapid system simulations to in-depth mechanistic analyses.

3 Core Components

Building upon the overall architecture, this section presents the mathematical descriptions of the most representative core components within the fluid library(Hefni and Bouskela 2019).

3.1 Pipe

The pipe model is designed to simulate medium flow throughout the system and calculate the fluid flow status within piping equipment, taking into account influences such as heat transfer and friction. Formulated as a homogeneous flow model based on the three conservation equations, it is available in both lumped parameter and distributed parameter forms.

3.1.1 Lumped Parameter Pipe Equations

In the lumped parameter pipe equations, the mass conservation equation can be expressed as:

$$V\frac{d\rho}{dt} = W_{in} - W_{out} \tag{1}$$

for mixtures, the following condition must also be satisfied in addition to the above equation:

$$V\frac{d\rho \cdot x_n}{dt} = W_{in} \cdot x_{n,in} - W_{out} \cdot x_{n,out} \tag{2}$$

Neglecting density variations, the static mass conservation can be expressed as:

$$W_{in} - W_{out} = 0 \tag{3}$$

$$W_{in} \cdot x_{n,in} - W_{out} \cdot x_{n,out} = 0 \tag{4}$$

In the equations, V denotes the volume of the working fluid, m^3; ρ represents the density of the working fluid, $kg \cdot m^{-3}$; W indicates the mass flow rate, $kg \cdot s^{-1}$; x_n is the mass fraction of the n -th species in the mixture; and the subscripts *in* and *out* denote the inlet and outlet, respectively.

The energy conservation equation for the fluid can be expressed as:

$$V\frac{d\rho \cdot u}{dt} = W_{in} \cdot h_{in} - W_{out} \cdot h_{out} + Q \tag{5}$$

where u denotes the specific internal energy of the fluid within the control volume, $J \cdot kg^{-1}$; h represents the specific enthalpy, $J \cdot kg^{-1}$, and Q indicates the heat absorbed by the control volume, W.

The momentum conservation equation is given by:

$$\left(\frac{L}{A}\right) \cdot \frac{dW}{dt} = P_{in} - P_{out} - \Delta P_f - \Delta P_h \tag{6}$$

Here, ΔP_f denotes the frictional pressure drop, Pa; ΔP_h represents the gravitational pressure drop, Pa; L is the control volume length, m, and A stands for the pipe cross-sectional area, m^2. When flow rate variations are neglected, the momentum conservation equation can be simplified as:

$$0 = P_{in} - P_{out} - \Delta P_f - \Delta P_h \tag{7}$$

3.1.2 Distributed Parameter Pipe Equations

The fluid library developed in this study offers two distributed parameter pipe modeling approaches: the FDM and the FVM, allowing users to select the appropriate method based on specific modeling conditions. Here, the FDM is used as an example to illustrate the formulation of distributed parameter pipe equations. The pipe fluid equations based on FDM are designed to simulate the dynamic behavior of one-dimensional fluid flow, encompassing variations in mass, energy, and momentum. This approach is primarily applicable to incompressible fluids.

The mass conservation equation for the fluid can be expressed as:

$$V_{i-1}\frac{d\rho_i}{dt} = W_{i-1} - W_i, i = 2, \cdots, 6 \tag{8}$$

where the subscript i denotes the i -th control node. For mixtures, the following condition must also be satisfied:

$$V_{i-1}\frac{d\rho_i \cdot x_{n,i}}{dt} = W_{i-1} \cdot x_{n,i} - W_i \cdot x_{n,i}, i = 2, \cdots, 6 \tag{9}$$

Neglecting fluid density variations, the steady-state mass conservation equation can be written as:

$$0 = W_{i-1} - W_i, i = 2, \cdots, 6 \tag{10}$$

$$0 = W_{i-1} \cdot x_{n,i} - W_i \cdot x_{n,i}, i = 2, \cdots, 6 \tag{11}$$

The energy conservation equation for the fluid is given by:

$$\mathrm{V}_{i-1}\frac{d\rho_i \cdot u_i}{dt} = W_{i-1} \cdot h_{i-1} - W_i \cdot h_i + Q_{i-1} + W_{mech,i}, i = 2, \cdots, 6 \tag{12}$$

where $W_{mech,i}$ denotes the mechanical work done by the i -th control volume, W.

The momentum equation for the fluid can be written as:

$$\left(\frac{L_i}{A}\right) \cdot \frac{dW_i}{dt} = P_i - P_{i+1} - \Delta P_{fi} - \Delta P_{hi} \tag{13}$$

Simplified to the following when flow variations are neglected:

$$0 = P_i - P_{i+1} - \Delta P_{fi} - \Delta P_{hi} \tag{14}$$

The index i varies depending on the specific structure of the momentum conservation equations. These equations are established at the boundaries of control volumes based on the staggered grid method. The model structure dictates the configuration of inlet and outlet boundary conditions. The fluid library proposed in this study encompasses the following four structures:

- FF Model: A symmetric configuration comprising six momentum conservation equations, which are assigned to control volumes 1 through 6.
- VV Model: A symmetric configuration featuring four momentum conservation equations, located at control volumes 2 through 5. The inlet and outlet ports expose the thermodynamic states of the first and last control volumes, respectively.
- FV Model: An asymmetric configuration consisting of five momentum conservation equations, assigned to control volumes 1 through 5.
- VF Model: An asymmetric configuration consisting of five momentum conservation equations, located at control volumes 2 through 6.

3.2 Volume

The volume model represents the mixing of fluid within a defined space and is governed by mass and energy balance equations, while accounting for heat exchange between the fluid and other heat sources. The fluid library provides two variations: the inertial volume and the two-phase volume. This section focuses primarily on the two-phase volume model.

The mass conservation equations for the liquid and vapor phases within the two-phase volume can be expressed, respectively, as:

$$\begin{aligned}\frac{d\left(M_l\right)}{dt} =& \left(1 - x_{water,in}\right) \cdot W_{water,in} - W_{water,out} \\ &+ \left(1 - x_{steam,in}\right) \cdot W_{steam,in}\end{aligned} \tag{15}$$

$$\begin{aligned}\frac{d\left(M_v\right)}{dt} =& x_{water,in} \cdot W_{water,in} + x_{steam,in} \cdot W_{steam,in} - \\ & W_{steam,out} - W_{cond} - W_{cond,wall} + W_{evap}\end{aligned} \tag{16}$$

where M denotes the mass of the working fluid in kg; x is the vapor quality; W represents the mass flow rate of the working fluid in $\mathrm{kg \cdot s^{-1}}$. The subscripts *water* and *steam* denote water and steam, respectively; *in* and *out* refer to the inlet and outlet, respectively; *cond* and *evap* indicate condensation and evaporation, respectively; l and v represent the liquid and vapor phases, respectively; and *wall* denotes the volume wall.

The energy conservation equation for the liquid phase in the two-phase volume is given by:

$$\begin{aligned}\frac{d\left(V_v \rho_v u_v\right)}{dt} =& x_{water,in} \cdot W_{water,in} \cdot h^v_{water,in} + \\ & x_{steam,in} \cdot W_{steam,in} \cdot h^v_{steam,in} - \\ & W_{water,out} \cdot h_l - W_{cond,wall} \cdot h_l - \\ & W_{cond} \cdot h_{l,sat} + W_{evap} \cdot h_{v,sat} + \\ & Q_v - P \cdot \frac{dV_v}{dt}\end{aligned} \tag{17}$$

where $h^v_{water,in}$ and $h^v_{steam,in}$ are calculated using the following equations:

$$h^v_{water,in} = \begin{cases} h_{water,in}, \text{if } x_{water,in} = 1 \\ h_{v,sat}, \text{if } x_{water,in} < 1 \end{cases} \tag{18}$$

$$h^v_{steam,in} = \begin{cases} h_{steam,in}, \text{if } x_{steam,in} = 1 \\ h_{v,sat}, \text{if } x_{steam,in} < 1 \end{cases} \tag{19}$$

3.3 Valves and Mechanical Equipment

Valves and mechanical equipment play a pivotal role at the system level in fluid path distribution and pressure regulation. They are essential for supporting stable simulations of SMR loops under control adjustments and transient conditions. This section provides a detailed description using control valves, simple pumps, and turbine models as representative examples.

3.3.1 Control Valve

The fluid library developed in this study offers a variety of valve models; this section uses the control valve as a case study to exemplify the modeling process. First, the mass and energy conservation equations are formulated as follows:

$$W_{in} = W_{out} \tag{20}$$

$$h_{in} = h_{out} \tag{21}$$

where W denotes the mass flow rate in $\mathrm{kg \cdot s^{-1}}$; h represents the specific enthalpy in $\mathrm{J \cdot kg^{-1}}$; and the subscripts *in* and *out* denote the valve inlet and outlet, respectively. Considering the pressure loss across the valve:

$$\left(P_{in} - P_{out}\right) \cdot C_v \cdot |C_v| = 1.732189 \times 10^{12} \frac{W_{in} \cdot |W_{in}|}{\rho \cdot \rho_r} \tag{22}$$

where ρ denotes the average density of the working fluid in $\mathrm{kg \cdot m^{-3}}$; ρ_r represents the reference density of the working fluid in $\mathrm{kg \cdot m^{-3}}$; P is the valve pressure in Pa; and C_v is the flow coefficient in Imperial units, expressed in $\mathrm{gpm \cdot psi^{-0.5}}$. Here, C_v is a function of the valve opening Ω, defined as $C_v = f_v\left(\Omega\right)$, where f_v denotes the valve characteristic curve. The value of Ω ranges from 0 to 100%.

DOI
10.3384/ecp223227

The valve is fully closed when $\Omega = 0$ and fully open when $\Omega = 100\%$. The valve model also accounts for the valve travel time, which is expressed as:

$$\frac{d\Omega}{dt} = \begin{cases} \min\left(\frac{100\%}{\tau_{open}}, \frac{\Omega_d - \Omega}{\tau}\right), \text{if } \Omega_d > \Omega \\ \max\left(-\frac{100\%}{\tau_{close}}, \frac{\Omega_d - \Omega}{\tau}\right), \text{if } \Omega_d < \Omega \end{cases} \tag{23}$$

where Ω_d denotes the desired valve opening; τ_{open} represents the valve opening travel time, s; τ_{close} is the valve closing travel time, s; and τ denotes the valve actuation time, s.

3.3.2 Simple Pump

The centrifugal pump is a ubiquitous type of hydraulic machinery that operates on the principle of centrifugal force. In the simplified pump model, the volumetric flow rate is calculated based on the input driving power. The hydraulic power is derived from the pump mechanical efficiency as follows:

$$P_h = P_d \cdot \eta_{mech} \tag{24}$$

where P_h denotes the hydraulic power in W; P_d represents the input driving power, W; and η_{mech} is the mechanical efficiency of the hydraulic machinery. The volumetric flow rate is derived from the hydraulic power as follows:

$$Q = \frac{P_h}{\Delta p} \tag{25}$$

where Q denotes the volumetric flow rate in $\mathrm{m}^3 \cdot \mathrm{s}^{-1}$; and Δp represents the pressure difference between the pump inlet and outlet, Pa. The mass flow rate W can be calculated from the volumetric flow rate as follows:

$$W = \rho Q \tag{26}$$

The mass and energy conservation equations can be further expressed as:

$$W_{in} = W_{out} = W \tag{27}$$

$$h_{out} = h_{in} + \frac{P_h}{W} \tag{28}$$

where the subscripts *in* and *out* denote the pump inlet and outlet, respectively.

3.3.3 Turbine

The turbine model assumes the working fluid to be subsonic, the process to be quasi-steady, and the flow at the turbine inlet and outlet to be either supercritical or subcritical. According to Stodola's ellipse law, the mass flow rate within the turbine is given by:

$$W = \sqrt{\frac{P_{in}^2 - P_{out}^2}{C_s \cdot x_{in} \cdot T_{in}}} \tag{29}$$

where C_s denotes the Stodola ellipse coefficient; P represents the pressure in Pa; the subscripts *in* and *out* denote the turbine inlet and outlet, respectively; x is the steam quality; and T represents the temperature. The specific enthalpy of the outlet fluid can be expressed as:

$$h_{out} = h_{in} + \eta_{is} \cdot x_m \cdot (h_{is} - h_{in}) + \frac{(1 - \eta_{nz}) \cdot v_{out}^2}{2} \tag{30}$$

where η_{is} denotes the isentropic efficiency of dry steam; x_m represents the average steam quality; h_{is} is the specific enthalpy of the fluid after isentropic expansion; η_{nz} denotes the nozzle efficiency; and v represents the fluid velocity. The mechanical power generated by the turbine is given by:

$$P_{mech} = \eta_{sta} \cdot W \cdot (h_{in} - h_{out}) \cdot \left(1 - \frac{P_{fric}}{100}\right) \tag{31}$$

where P_{mech} denotes the mechanical power; η_{sta} represents the efficiency for calculating kinetic energy loss; and P_{fric} is the power loss caused by hydraulic friction.

4 Test Case

This section presents typical nuclear SMR system models constructed using the fluid library. Simulation results are provided and compared with a corresponding system modeled in RELAP5(U.S. Nuclear Regulatory Commission 2001) to validate the effectiveness of the fluid library in system-level assembly, dynamic behavior representation, and engineering applications. It is important to emphasize that this case study serves not only to demonstrate the correctness of the individual system model but also to indirectly verify the rationality of the fluid library regarding component organization, interface uniformity, and model reusability from an application perspective.

4.1 Main Feedwater System

The main feedwater system of the SMR is equipped with three motor-driven feedwater pumps, configured in a "two-in-service, one-standby" arrangement. This means that only two of the feedwater pumps are in operation during normal system functioning. The feedwater flow discharged from the pumps has two paths: one passes through the feedwater valve to the steam generator, while the other flows back to the deaerator via a bypass valve. On the branch leading to the steam generator, two low-load feedwater valves are connected in series and are arranged in parallel with the feedwater regulating valve. Specifically, the feedwater regulating valve is put into operation at high power levels, whereas the low-load feedwater valves are utilized under low-load conditions. The model structure is shown in Figure 3.

The main test content of the feedwater system includes: (1) comparing the simulation values of key parameters with reference values under rated operating conditions; and (2) under the designated control scheme, assuming a

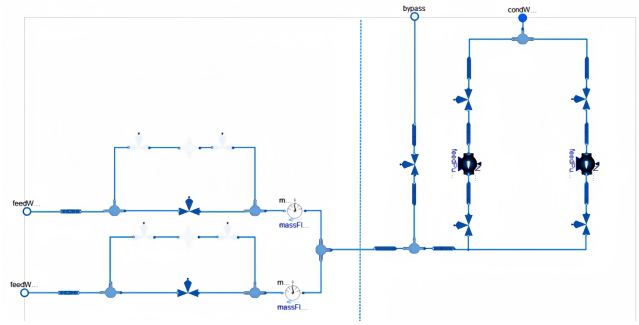


Figure 3. Configuration of the Main Feedwater System.

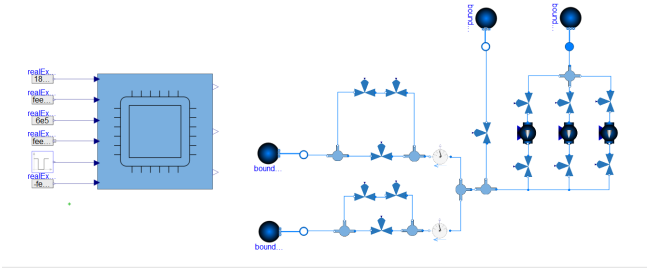


Figure 4. Transient Test Case of the Main Feedwater System.

Table 1. Comparison of Simulation and Reference Values for Main Feedwater System Parameters

System Parameter	Simulation Value	Reference Value	Unit	Relative Error
Deaerator Pressure	0.41	0.41	MPa	0.000%
Deaerator Temperature	144.53	144.5	$°C$	0.018%
Feedwater Pump Inlet Pressure	0.50	0.51	MPa	2.192%
Feedwater Pump Outlet Pressure	2.30	2.33	MPa	1.202%
Feedwater Pump Head	199.25	200	m	0.374%
Feedwater Flow Rate	161.81	162.26	t/h	0.278%

change in the feedwater flow setpoint, analyzing the variations in key system parameters such as feedwater flow, feedwater pump flow, and feedwater valve pressure difference, and comparing these results with Relap5 simulations.

4.1.1 Steady-State Test Scenario

The steady-state values of key system parameters under the rated operating conditions of the feedwater system are presented in Table 1. As indicated in the table, the maximum deviation between the simulated values and the design values of the key parameters is 2.192%. The simulation results of the main feedwater system under steady-state conditions are presented in Table 1.

4.1.2 Transient Test Scenario

The test cases for the feedwater system transient scenarios are illustrated in Figure 4. This test focuses exclusively on the feedwater system and was not conducted in parallel operation with other systems. Consequently, specific boundary conditions must be defined at the system boundary interfaces to ensure normal system operation. The detailed parameters for these boundary conditions are listed in Table 2.

The control scheme adopted in this test case is as follows: the differential pressure across the feedwater valve is maintained constant by regulating the feedwater pump speed; the feedwater flow rate is adjusted by modulating the feedwater valve opening; and the feedwater pump flow rate is kept constant by regulating the bypass valve opening. At $t = 50$, the feedwater flow rate setpoint undergoes a step decrease of 10 kg/s. Subsequently, at $t = 150$, the setpoint steps back to its original value. The simulation results are illustrated in Figure 5. When the feedwater flow rate setpoint decreases, the feedwater valve opening is reduced to decrease the flow passing through it. This reduction in valve opening leads to an increase in the differential pressure across the valve. Furthermore, the decreased opening increases the flow resistance from the deaerator to the feedwater valve outlet, resulting in a reduction of the flow through the feedwater pump. To maintain a constant pump flow rate, the bypass valve opening is increased to divert more flow back to the deaerator. Consequently, the bypass flow increases, and the feedwater pump flow rate returns to its setpoint after a brief dip. Conversely, when the feedwater flow rate setpoint increases, the trends are exactly opposite to those observed during the decrease.

Table 2. Boundary Conditions for Feedwater System Transient Test

Boundary Interface	Boundary Type	Boundary Value
Feedwater Pump Inlet	Pressure, Temperature	4.1 bar, 144.5 °C
#1 Feedwater Valve Outlet	Pressure	1.7 MPa
#2 Feedwater Valve Outlet	Pressure	1.7 MPa
Bypass Valve Outlet	Pressure	4.1 bar

Compared with the RELAP5 model, the system model constructed based on the fluid library shows good agreement in terms of key process variables and dynamic trends. This indicates that the fluid library developed in this paper not only supports the stable assembly and solution of typical SMR loops but also possesses robust dynamic response capabilities under control regulation and boundary disturbance conditions. This further validates the engineering applicability and numerical reliability of the fluid library modeling method at the system level.

A visible deviation occurs in the feedwater and pump flow rates immediately after the setpoint changes. The deviation is mainly concentrated in the short transition intervals and is small in absolute magnitude compared with the rated flow rate. It can be attributed to differences in component discretization, pump and valve characteristic curves, and controller implementation between the Modelica and RELAP5 models. After the control loops settle, the two models converge to comparable steady values. Therefore, the deviation does not affect the assessment of the overall dynamic trend or the system-level applicability of the library.

DOI
10.3384/ecp223227

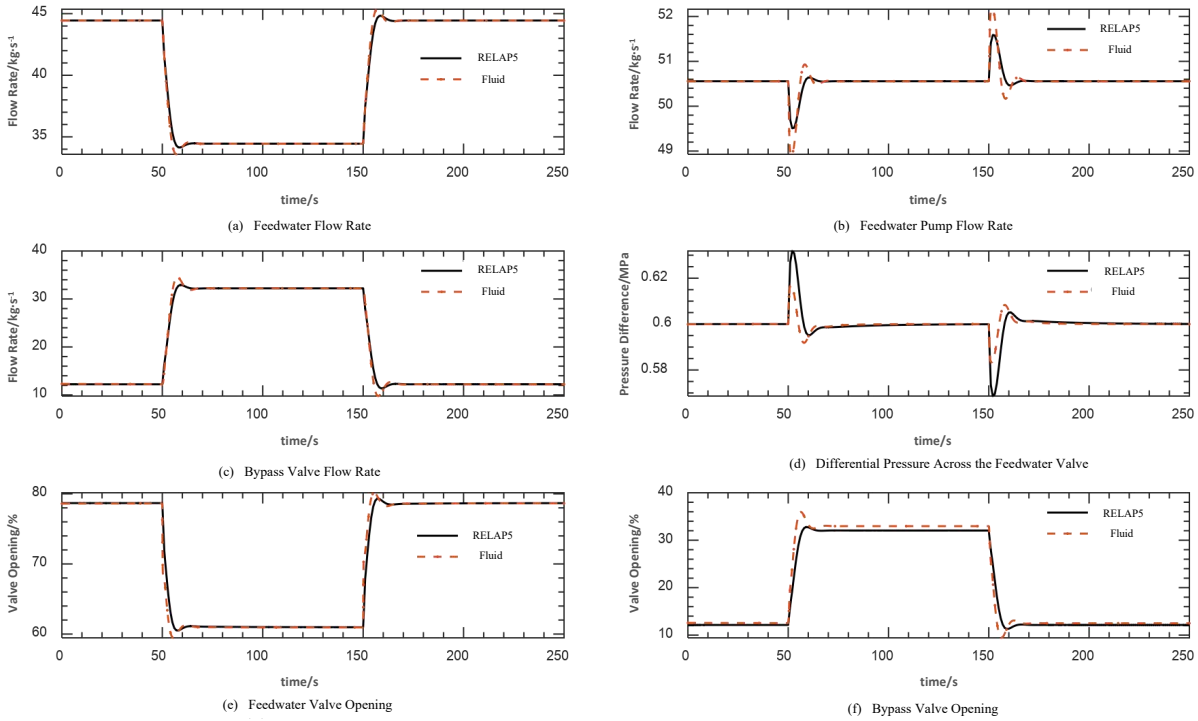


Figure 5. Transient Test Results of the Main Feedwater System.

4.2 Safety Relief System

The simulation model of the safety relief system established based on the fluid library presented in this paper is shown in the figure 6. The safety relief system is connected to the pressure vessel cavity. When the pressure within the cavity exceeds the safety limit, the relief valve opens to depressurize the cavity.

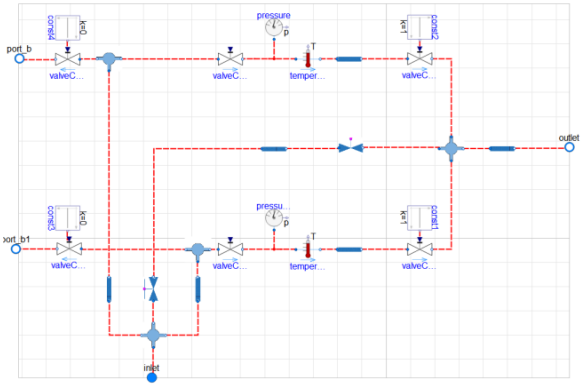


Figure 6. Configuration of the Safety Relief System.

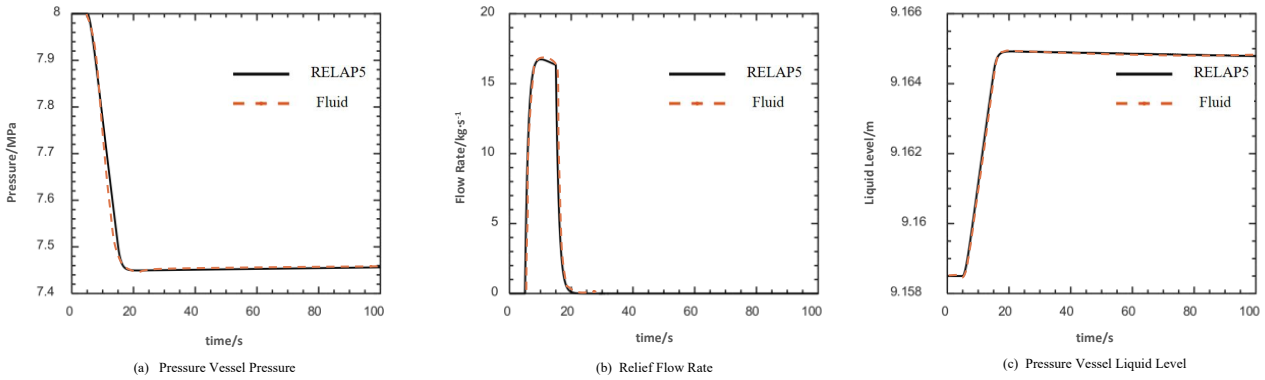
The test content for the safety relief system is as follows: the relief valve is opened and subsequently closed after a specific duration to analyze the transient behavior of parameters such as relief flow rate, pressure vessel pressure, and liquid level.

The test case for the safety relief system is illustrated in Figure 8. In this setup, the pressure vessel is represented by a closed tank, and the multi-purpose pool is simulated by a pressure boundary. The specific settings for the test boundary conditions are provided in Table 3.

Table 3. Boundary Conditions for Safety Relief System Test

Boundary Interface	Boundary Type	Boundary Value
Multi-purpose Pool Boundary	Pressure Boundary	1 MPa
Fluid Outlet Boundary #1	Pressure Boundary	7 MPa
Fluid Outlet Boundary #2	Pressure Boundary	7 MPa

The safety relief system is connected to the vapor space at the top of the pressure vessel. When the pressure in this space exceeds a certain threshold, the relief valve opens to depressurize the system. In this test scenario, the relief valve remains closed for the first $5s$. At $t = 5s$, the valve opens to initiate depressurization, and subsequently closes at $t = 15s$. Upon opening the relief valve at $t = 5s$, the relief flow rate rises rapidly, causing a sharp decrease in the pressure vessel's internal pressure. This pressure drop leads to a reduction in the density of the liquid phase within the vessel. Since the liquid mass remains constant, this decrease in density results in an expansion of volume, thereby causing the liquid level to rise. At $t = 15s$, when the relief valve closes, the relief flow rate drops quickly, and both the pressure and liquid level in the pressure vessel tend to stabilize. The simulation results are illustrated in Figure 7. Compared with the RELAP5 model, the relief system model constructed based on the fluid library demonstrates a rapid response at key transition points and shows good agreement in overall trends.



(a) Pressure Vessel Pressure (b) Relief Flow Rate (c) Pressure Vessel Liquid Level

Figure 7. Test Results of the Safety Relief System.

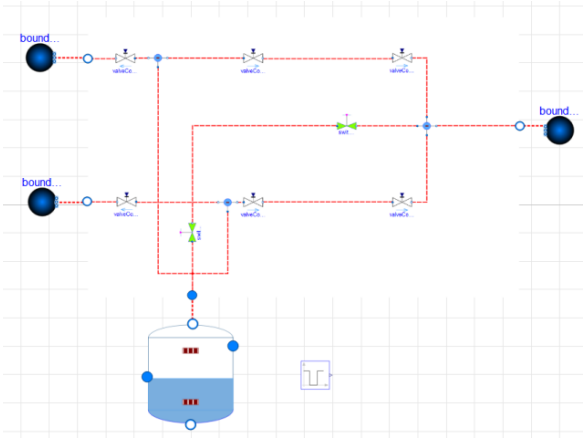


Figure 8. Test Case of the Safety Relief System.

5 Conclusion

To meet the thermal-hydraulic modeling and digital R&D requirements of SMR systems, this paper presents a fluid component library based on Modelica. The library is analyzed and validated from three perspectives: overall architecture, core models, and system-level cases. Adopting a hierarchical and modular design philosophy, the library establishes a comprehensive model organization system centered on unified interfaces, fundamental mechanisms, engineering components, and system assembly. It forms a collection of reusable components covering pipes, volumes, valves, mechanical machinery, connectors, and sensors. By modeling typical components within a unified conservation equation framework, this work achieves a unification of physical consistency, structural scalability, and engineering assemblability. Furthermore, as a fundamental part of the SMR library, the developed fluid library plays a critical role in bridging basic fluid mechanisms, device-level model development, and system-level loop assembly. Regarding validation, the main feedwater system and the safety relief system were selected as typical cases for simulation analysis. The simulation results show good agreement with the RELAP5 model, demonstrating that the developed fluid library possesses sound physical rationality and numerical stability.

In summary, this work not only provides a reusable modeling foundation for Modelica-based modeling of SMR fluid systems but also offers support for subsequent system scheme selection, control strategy analysis, and unified whole-system modeling oriented towards MBSE.

References

Hefni, Baligh El and Daniel Bouskela (2019). *Modeling and Simulation of Thermal Power Plants with ThermoSysPro. A Theoretical Introduction and a Practical Guide*. Cham: Springer. ISBN: 978-3-030-05104-4. DOI: 10.1007/978-3-030-05105-1.

Hong, Hao, Hong Lyu, Fuhong Luo, and Jingren Guo (2026). "Development and Verification of Simulation Model Library for Multi-Discipline Unified Modeling of Low Temperature Heat Reactor System". In: *Nuclear Power Engineering*, pp. 1–6. ISSN: 0258-0926.

Jiang, Qingfeng, Hao Hong, Hong Lyu, and Pengfei Wang (2025). "Research on Control Strategy of the Coolant System for Low-Temperature Heating Reactor". In: *Nuclear Power Engineering* 46.5, pp. 171–179. ISSN: 0258-0926. DOI: 10.13832/j.jnpe.2024.090030.

Modelica Association (2023). *Modelica Standard Library*. Online. Version 4.0.0. Accessed: 2024-04-13. URL: https://github.com/modelica/Modelica.

U.S. Nuclear Regulatory Commission (2001). *RELAP5/MOD3.3*. Computer Software. Version 3.3, Accessed: 2024-04-13. Idaho Falls, ID, USA: Idaho National Laboratory. URL: https://www.nrc.gov/reading-rm/doc-collections/nuregs/contract/cr7004/index.html.

Zhang, Liqun (2025). "Exploration and Research on Automatic Modeling of Nuclear Power Plants Based on Modelica". In: *Nuclear Science and Engineering* 45.01, pp. 90–97. ISSN: 0258-0918. DOI: 10.20190/j.cnki.02580918.202501012.

Research on Dynamic Simulation of Nuclear Power Steam Generator Based on Modelica

Hao Hong[1] Sixin Liu[1] Hong Lyu[1]

[1]State Key Laboratory of Nuclear Power Safety Technology and Equipment, China Nuclear Power Engineering Co., Ltd., Shenzhen, Guangdong, 518116, China, 1330636650@qq.com

Abstract

As the pivotal equipment connecting the primary and secondary loops of pressurized water reactor nuclear power plants, the dynamic thermo-hydraulic characteristics of the nuclear steam generator are directly related to the safe and stable operation of the unit. However, traditional modeling approaches suffer from poor versatility and low efficiency, making it difficult to meet the R&D demands for diverse application scenarios and iterative optimization. Addressing these bottlenecks, this paper proposes a modular modeling and dynamic simulation method for nuclear steam generators based on the Modelica language. First, based on a hierarchical modeling approach, component-level models covering plena, pipes, heat structures, steam-water separators, and feedwater plena are constructed to achieve free configuration and visual simulation modeling. Second, according to the operating principles and structure of the nuclear steam generator, a complete dynamic simulation model is assembled using these component-level models. Finally, the established model is validated against the RELAP5 code to verify the effectiveness and reliability of the proposed method. The results indicate that the Modelica-based steam generator simulation model combines high accuracy with high efficiency, providing significant technical support for the rapid design and verification of nuclear power systems.

Keywords: Modelica, Steam Generator,Nuclear Energy

1 Introduction

The long development cycles, high risks, and substantial costs inherent to nuclear energy pose severe challenges to its R&D paradigms. Digital transformation is not merely a critical pathway to resolving these dilemmas but also an intrinsic requirement and core support for the nuclear industry's transition toward high-quality development. Given the complex, strongly coupled, and multidisciplinary nature of nuclear power systems—spanning neutron physics, thermal-hydraulics, structural mechanics, and automatic control—traditional simulation approaches based on isolated disciplines or simplified models are increasingly inadequate for efficiently supporting the precise analysis of their integrated dynamic characteristics. With its distinct advantage in unified multidisciplinary modeling, the Modelica language employs a hierarchical modeling approach akin to "building blocks."(Huang et al. 2021; Xu et al. 2025; Zhang 2025; Hao et al. 2024) This allows for the intuitive connection and integration of heterogeneous models—including the reactor core, steam generator, turbine-generator set, and control systems—to construct a complete, simulatable virtual nuclear power system, perfectly aligning with the highly integrated design requirements of modern nuclear power plants.

Building on this, this paper introduces the concept of modular modeling, focusing on the nuclear steam generator to establish a simulation model framework that is modular, parameterized, and portable. Specifically, the study first analyzes the component models of the U-tube steam generator based on the system topology and physical processes, employing a top-down decomposition strategy. In terms of implementation, the modeling follows a bottom-up construction workflow: starting with base-level models describing fundamental physical processes, component-level models are constructed via multi-layer inheritance mechanisms. Finally, the dynamic simulation model of the nuclear steam generator is rapidly assembled through the drag-and-drop connection and combinatorial encapsulation of these models. To verify the validity and reliability of this modular modeling approach, the constructed dynamic simulation model is validated against a RELAP5 model.

2 Working Principle

The U-tube steam generator is a widely adopted intermediate heat exchanger in pressurized water reactors. It primarily comprises the shell assembly, lower head, tubesheet, U-tube bundle, and steam-water separators, with its structure illustrated in Figure 1. Primary coolant enters the steam generator's inlet plenum via the lower head nozzle, flows through the interior of the inverted U-shaped heat transfer tubes, and transfers heat to the secondary working fluid during this process. After heat exchange, the coolant flows into the outlet plenum and is discharged from the steam generator through the lower head outlet nozzle. Secondary feedwater is introduced through the upper shell nozzle and distributed by the feedwater ring into the annular downcomer between the tube bundle shroud and the steam generator shell. Here, it merges with recirculated water separated by the steam-water separators and flows downward. The fluid then enters the

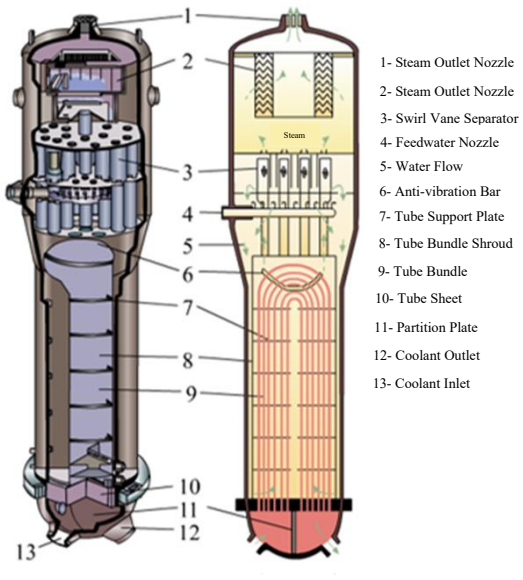


Figure 1. U-Tube Steam Generator Structural Diagram.(Qiu et al. 2025)

tube bundle region through the flow clearance between the tubesheet's secondary side surface and the tube bundle shroud. It transversely scours the outer surface of the heat transfer tubes before being constrained by the flow channel to transition into an upward longitudinal flow along the tube bundle. During this process, the secondary working fluid continuously absorbs heat from the primary side, causing its temperature to gradually rise to saturation and undergo a phase change, forming a steam-water mixture. This mixture ascends along the tube bundle and enters the steam-water separators after exiting the tube bundle's curved section. The separators employ centrifugal force to remove the majority of the liquid phase from the mixture; the separated water is then returned to the annular downcomer, thereby closing the secondary recirculation loop.

3 Development of the Steam Generator Dynamic Model

Based on the heat transfer mechanisms and operational principles of the vertical U-tube steam generator(Wu et al. 2025), the overall process is abstracted into the modeling schematic shown in Figure 2. The dynamic model is decomposed into several component-level sub-models, including pipes, volumes, heat structures, steam-water separators, and the feedwater plenum. The following sections detail the mathematical formulations adopted for each component, culminating in the assembly of the integrated dynamic simulation model for the steam generator.

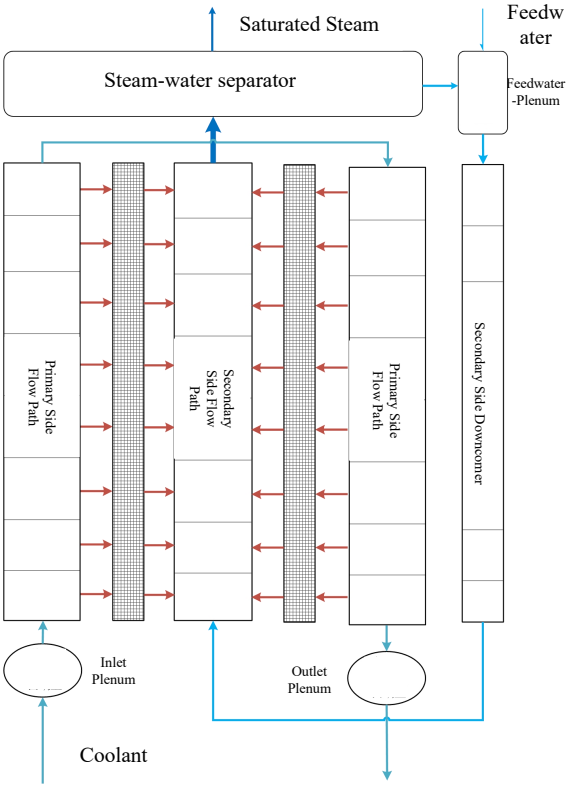


Figure 2. Vertical U-Tube Steam Generator Modeling Schematic Diagram.

3.1 Component-Level Models

3.1.1 Pipe Model

In the U-tube steam generator, the primary coolant remains in a single-phase state, whereas the secondary feedwater absorbs heat transferred from the primary side in the riser section, transitioning from subcooled liquid to saturated water. Consequently, this paper develops distinct pipe models tailored for single-phase and two-phase flow regimes, respectively.

Single-Phase Pipe Model Taking the division into five control volumes as an illustrative example, the nodalization scheme for a single-phase pipe is depicted in Figure 3. The governing equations for mass, energy, and momentum conservation within the internal flow are formulated as follows(Q. F. Jiang et al. 2025):

$$V_{i-1}\frac{d\rho_i}{dt} = W_{i-1} - W_i \tag{1}$$

$$V_{i-1}\frac{d\rho_i \cdot u_i}{dt} = W_{i-1}\cdot h_{i-1} - W_i \cdot h_i + Q_{i-1} \tag{2}$$

$$\left(\frac{L_i}{A}\right)\cdot\frac{dW_i}{dt} = P_i - P_{i+1} - \Delta P_{fi} - \Delta P_{hi} \tag{3}$$

where V denotes the volume (m^3); ρ is the density ($kg{\cdot}m^{-3}$); W represents the mass flow rate ($kg{\cdot}s^{-1}$); u and h are the specific internal energy ($J{\cdot}kg^{-1}$) and specific enthalpy ($J{\cdot}kg^{-1}$), respectively; Q is the heat transfer

DOI
10.3384/ecp223237

rate (W), where for the fluid flowing inside the primary side channel, the heat transfer coefficient can be calculated using the Dittus-Boelter correlation(NSA Division 2001); ΔP_f and ΔP_h denote the frictional and gravitational pressure drops (Pa), respectively; L represents the length (m); and A is the cross-sectional area of the pipe (m^2).The subscript i denotes the i-th control volume.

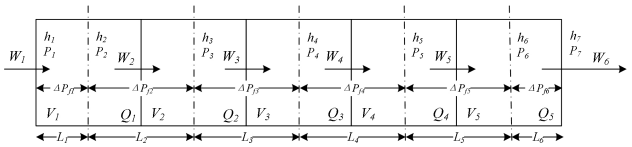


Figure 3. Schematic Diagram of Single-Phase Pipe Node Division.

Two-Phase Pipe Model When phase transition occurs within the pipe, fluid properties (e.g., density) undergo abrupt changes; consequently, applying a single-phase formulation would lead to numerical instability. In this study, a two-phase pipe model is established based on the Finite Volume Method (FVM), subject to the following assumptions: (1) the homogeneous equilibrium model is adopted, assuming equal velocities for both phases; (2) the velocity profile is uniform across the cross-section; (3) a lumped pressure approach is employed, where the pressure throughout the pipe is assumed to be equal to the inlet or outlet pressure; and (4) specific enthalpy is linearly distributed within the control volume. Taking the division into five control volumes as an example, the nodalization scheme for the two-phase pipe is depicted in Figure 4. Adopting the outlet pressure as the lumped pressure for the entire pipe, the governing conservation equations for the two-phase flow are formulated as follows(Casella 2006):

$$V_j\left(\frac{\partial\bar{\rho}_j}{\partial h_j}\frac{dh_j}{dt}+\frac{\partial\bar{\rho}_j}{\partial h_{j+1}}\frac{dh_{j+1}}{dt}+\frac{\partial\bar{\rho}_j}{\partial P_{out}}\frac{dP_{out}}{dt}\right)=W_j-W_{j+1} \tag{4}$$

$$V_j\bar{\rho}_j\frac{dh_{j+1}}{dt}-V_j\frac{dP_{out}}{dt}=W_j^{bar}\left(h_j-h_{j+1}\right)+Q_j \tag{5}$$

$$\frac{L}{A}\frac{dW_{in}}{dt}=P_{in}-P_{out}-\Delta P_h-\Delta P_f \tag{6}$$

where $\bar{\rho}_j$ denotes the average density of the j-th control volume ($kg{\cdot}m^{-3}$); h_j and h_{j+1} represent the specific enthalpy at the inlet and outlet of the j-th control volume ($J{\cdot}kg^{-1}$), respectively; Q_j is the heat exchange rate with the surroundings for the j-th control volume (W); W_{in} indicates the mass flow rate at the pipe inlet ($kg{\cdot}s^{-1}$); and P_{in} and P_{out} correspond to the pressure at the pipe inlet and outlet (Pa), respectively.

The calculation formulas for $\bar{\rho}_j$, $\frac{\partial\bar{\rho}_j}{\partial h_j}$, $\frac{\partial\bar{\rho}_j}{\partial h_{j+1}}$, and $\frac{\partial\bar{\rho}_j}{\partial P_{\text{out}}}$ vary depending on the thermodynamic state of the fluid within the control volume; specific details can be found in Ref(Casella 2006). Due to the phase change process occurring in the secondary side fluid, the calculation of the heat transfer coefficient involves numerous conditions. Therefore, the correlations adopted in this study are based on Ref(NSA Division 2001) and are not elaborated upon here.

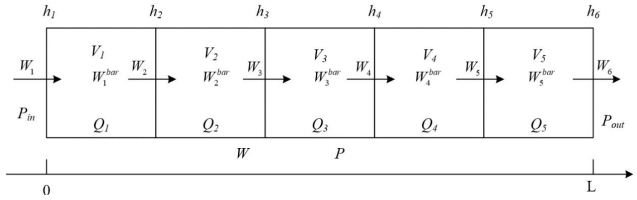


Figure 4. Schematic Diagram of Two-Phase Pipe Node Division.

3.1.2 Volume Model

The plenum model assumes that the fluid is perfectly mixed within a specific volume. It is governed by the mass and energy balance equations:

$$V\frac{d\rho}{dt}=\sum W_i-\sum W_o \tag{7}$$

$$V\frac{d\rho\cdot u}{dt}=\sum W_i\cdot h_i-\sum W_o\cdot h+Q \tag{8}$$

where V denotes the volume of the control volume (m^3); W_i and h_i represent the mass flow rate ($kg{\cdot}s^{-1}$) and specific enthalpy ($J{\cdot}kg^{-1}$) of the fluid entering the control volume from the i-th inlet; W_o is the mass flow rate exiting from the outlet ($kg{\cdot}s^{-1}$); and h corresponds to the specific enthalpy within the plenum control volume ($J{\cdot}kg^{-1}$).

3.1.3 Heat Structure Model

Heat structures serve primarily to transfer heat. In nuclear reactors, cylindrical heat structures are widely used, such as heat exchanger tube walls and fuel rods. The cylinder is discretized into control volumes in both the axial and radial directions, as shown in Figure 5. In the radial direction, r denotes the node position and i represents the radial node index; in the axial direction, z denotes the node position and j represents the axial node index. $\Delta\varphi$ represents the angular span of the control volume. Since the heat structure is modeled in two dimensions, $\Delta\varphi$ is set to 2π for each control volume, implying that heat transfer in the azimuthal direction is neglected. For the mathematical model of the tube wall, please refer to Ref(Q. F. Jiang et al. 2025), which will not be elaborated upon here.

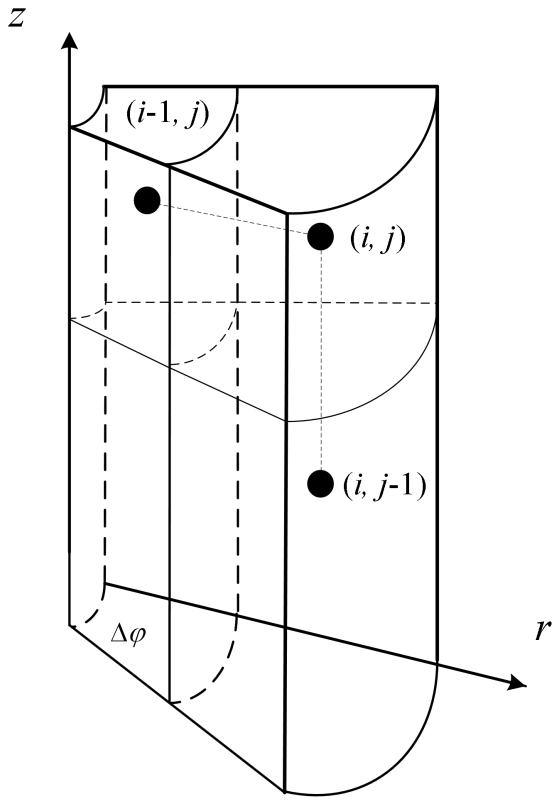


Figure 5. Schematic Diagram of Heat Structure Node Division.

3.1.4 Steam-Water Separator Model

The steam-water separator functions to separate the heated secondary side steam-water mixture. By reducing the steam humidity, it prevents erosion of turbine blades and ensures the safe and normal operation of the steam turbine. The separator divides the incoming mixture into water and steam. Assuming the dryness fraction of the separated steam is x_s (where $x_s \leq 1$), the specific enthalpy of the outlet steam is given by:

$$h_{g,out} = \begin{cases} x_s h_{sat,\mathrm{v}} + (1-x_s) h_{sat,l} & x < x_s \\ h_{s,out} & x \geq x_s \end{cases} \tag{9}$$

where $h_{s,out}$ denotes the specific enthalpy of the steam-water mixture entering the separation chamber ($\mathrm{J{\cdot}kg^{-1}}$); $h_{sat,l}$ and $h_{sat,v}$ represent the saturated specific enthalpy of the liquid and vapor phases ($\mathrm{J{\cdot}kg^{-1}}$), respectively; and x is the void fraction, which is calculated as follows:

$$x = \begin{cases} 1 & h_{s,out} > h_{sat,v} \\ \frac{h_{s,out} - h_{sat,l}}{h_{sat,v} - h_{sat,l}} & h_{sat,l} < h_{s,out} < h_{sat,v} \\ 0 & h_{s,out} < h_{sat,l} \end{cases} \tag{10}$$

The specific enthalpy at the liquid outlet is given by:

$$h_{f,out} = \begin{cases} h_{sat,l} & x \geq 0 \\ h_{s,out} & x < 0 \end{cases} \tag{11}$$

Based on the conservation of mass and energy, the outlet flow rates of water and steam are calculated as:

$$\begin{cases} W_{s,out} = W_{f,out} + W_{g,out} \\ W_{s,out} \cdot h_{s,out} = W_{f,out} \cdot h_{f,out} + W_{g,out} \cdot h_{g,out} \end{cases} \tag{12}$$

where $W_{s,out}$ denotes the mass flow rate of the steam-water mixture entering the separation chamber ($\mathrm{kg{\cdot}s^{-1}}$); $W_{f,out}$ and $W_{g,out}$ represent the mass flow rates ($\mathrm{kg{\cdot}s^{-1}}$) of the separated liquid and vapor phases, respectively.

3.1.5 Feedwater Plenum Model

The secondary side feedwater coalesces with the liquid phase separated by the steam-water separator in the feedwater plenum, subsequently flowing downward through the downcomer. The feedwater plenum is modeled as a two-phase plenum, where the working fluid within the control volume is divided into vapor and liquid phases, with mass and energy conservation equations established for each phase(El Hejni and Bouskela 2019). The mass conservation equation for the liquid phase is given by:

$$\frac{d(M_l)}{dt} = W_{c,in} - W_{out} + W_{f,out} + W_{cond} - W_{evap} \tag{13}$$

where M denotes the mass (kg); $W_{c,in}$, W_{out}, W_{cond}, and W_{evap} represent the feedwater flow rate ($\mathrm{kg{\cdot}s^{-1}}$), downcomer inlet flow rate ($\mathrm{kg{\cdot}s^{-1}}$), condensation flow rate ($\mathrm{kg{\cdot}s^{-1}}$), and evaporation flow rate ($\mathrm{kg{\cdot}s^{-1}}$), respectively; and the subscript l denotes the liquid phase. The mass conservation equation for the vapor phase is given by:

$$\frac{d(M_v)}{dt} = W_{evap} - W_{cond} + W_{steam,in} - W_{steam,out} \tag{14}$$

where the subscripts v, $steam$, in, and out denote the vapor phase, steam, inlet, and outlet, respectively. The energy conservation equation for the liquid phase is given by:

$$\begin{aligned} \frac{d(V_l \rho_l u_l)}{dt} =& W_{c,in} \cdot h_{c,in} - W_{out} \cdot h_l + W_{f,out} \cdot h_{f,out} + \\ & W_{cond} \cdot h_{l,sat} - W_{evap} \cdot h_{v,sat} - P \cdot \frac{dV_l}{dt} \end{aligned} \tag{15}$$

The energy conservation equation for the vapor phase is given by:

$$\begin{aligned} \frac{d(V_v \rho_v u_v)}{dt} =& W_{evap} \cdot h_{v,sat} - W_{cond} \cdot h_{l,sat} + W_{steam,in} \\ & \cdot h_{steam,in} - W_{steam,out} \cdot h_v - P \cdot \frac{dV_v}{dt} \end{aligned} \tag{16}$$

where $h_{l,sat}$ and $h_{v,sat}$ denote the specific enthalpy ($\mathrm{J{\cdot}kg^{-1}}$) of the saturated liquid and vapor phases, respectively.

DOI
10.3384/ecp223237

3.2 Equipment-Level Model

Based on the modeling schematic shown in Figure 2, the vertical U-tube steam generator is assembled from component-level models, including pipes, plena, heat structures, steam-water separators, and feedwater plenums, as illustrated in Figure 6. The primary coolant flows upward through the left-side pipes, transferring heat to the secondary side via the heat structure model, and subsequently flows downward through the right-side pipes to exchange heat with the secondary side in a similar manner. On the secondary side, the working fluid flows downward through the rightmost straight pipe, enters the intermediate pipe (two-phase pipe) to exchange heat with the heat structure, and evaporates into steam before exiting into the steam separation chamber. The liquid phase separated by the steam-water separator then coalesces with the feedwater and flows into the downcomer.

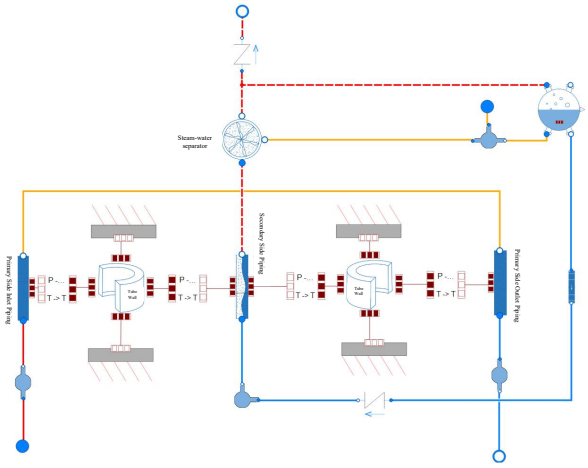


Figure 6. Vertical U-Tube Steam Generator Schematic Diagram.

4 Validation of the Steam Generator Dynamic Model

To investigate the dynamic characteristics of the U-tube steam generator model, the primary coolant inlet temperature was perturbed to analyze the variation trends of key parameters, and the results were validated against the RELAP5 model. The simulation test case is illustrated in Figure 7, where the primary side is defined by a flow boundary at the inlet and a pressure boundary at the outlet, while the secondary side is imposed with a flow boundary at the inlet and a velocity boundary at the outlet. The geometric and physical parameters of the simulation model are based on the data from the Hualong One nuclear power plant, as detailed in Table 1

At t=100s, a step increase of 2°C was imposed on the inlet temperature, and the resulting simulation results are depicted in Figure 8. As the primary inlet temperature rises, the primary outlet temperature follows suit.

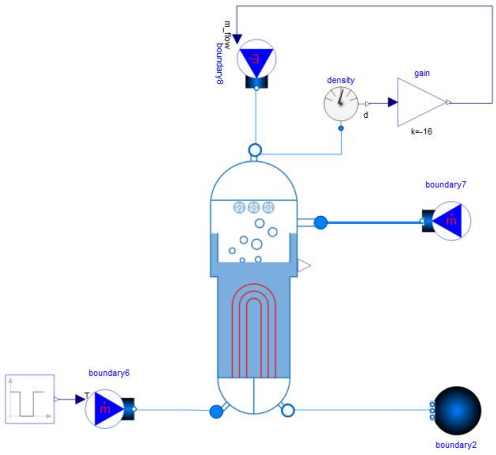


Figure 7. U-Tube Steam Generator Test Model.

Table 1. Geometric Parameters of the U-tube Steam Generator

Parameter Name	Value	Unit
Inner Diameter of U-tube	16.457	mm
Outer Diameter of U-tube	17.463	mm
Number of U-tubes	7997	
Equivalent Height of U-tubes	10.971	m
Length of Annular Downcomer	11.751	m
Inner Diameter of Annular Downcomer	3.39084	m
Outer Diameter of Annular Downcomer	3.7387	m
Volume of Inlet/Outlet Plenum	10.988	m^3
Secondary Side Flow Area in Bundle Region	5.0934	m^2

This temperature increment enlarges the temperature difference between the primary and secondary sides, thereby enhancing the heat transfer rate. Consequently, the secondary fluid temperature rises, leading to a reduction in fluid density. Given that a velocity boundary is prescribed at the secondary outlet, the constant flow velocity combined with the decreased density results in an increased steam mass flow rate. Compared with the RELAP5 model,the relative deviation of primary-side inlet temperature is 0.0153%, the relative deviation of of primary-side outlet temperature is 0.017%, the relative deviation of of primary-side flow rate is 0.13%, and the relative deviation of of steam flow rate is 0.086% ,the dynamic trends of the Modelica-based U-tube steam generator model show good agreement with the reference, with errors falling within acceptable limits.

5 Conclusions

Based on the multi-domain physical unified modeling language Modelica, this study aims to establish a modular, parameterized, and portable simulation modeling framework. Focusing on the vertical U-tube steam generator, research on modular modeling and dynamic simulation of nuclear power equipment was conducted. The main conclusions are as follows:

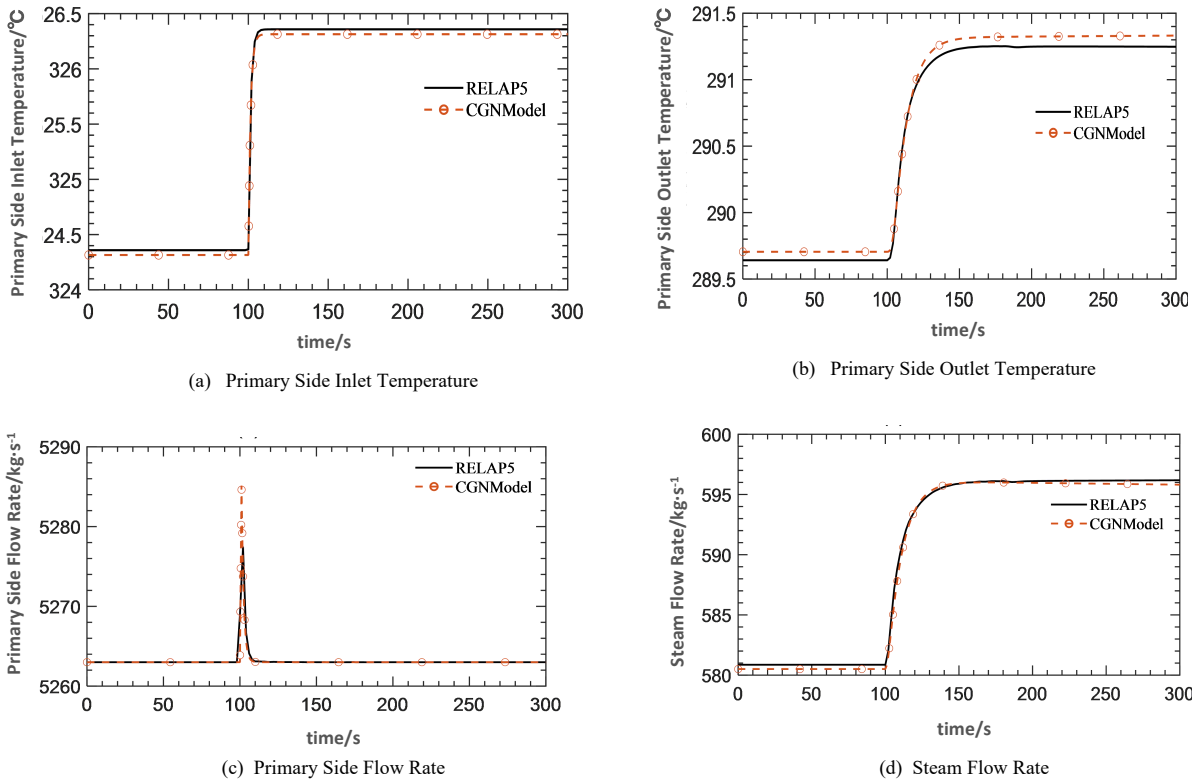


Figure 8. Test Results of U-Tube Steam Generator.

- Modelica technology achieves the decoupling of physical-mathematical equations from the numerical solution process. Its built-in solvers are capable of automatically discretizing and numerically solving given algebraic-differential equations, which significantly reduces the complexity of developing numerical calculation programs and enhances the construction efficiency and maintainability of simulation models.
- Using the Modelica language, component-level models—including plena, pipes, heat structures, steam-water separators, and feedwater plena—were developed. Based on a hierarchical modeling approach, these components were integrated to construct a complete dynamic simulation model of the nuclear steam generator.
- Through comparative validation against the model established by the nuclear energy system analysis code RELAP5, the dynamic response trends of the U-tube steam generator simulation model developed in this study show good agreement with the RELAP5 model. With errors in key parameters falling within acceptable limits, the effectiveness of the proposed modeling method and the reliability of the simulation results are verified.

References

Casella, F. (2006). "Object-Oriented Modelling of Two-Phase Fluid Flows by the Finite Volume Method". In: *Proceedings 5th Mathmod Vienna*.

El Hejni, B. and D. Bouskela (2019). *Modeling and simulation of thermal power plants with ThermoSysPro*. Cham: Springer.

Hao, Chengming, Lizhi Liu, Yifu Han, et al. (2024). "Global Parameter Optimization Analysis of Nuclear Power Plant Based on Modelica". Chinese. In: *Nuclear Power Engineering* 45.S2, pp. 28–34.

Huang, Yanping, Xiaokang Zeng, and Ji Ding (2021). "Architecture and Proof of Concept of Simulation Model for Two-Phase Thermal-Hydraulic Characteristics Based on Modelica". Chinese. In: *Nuclear Power Engineering* 42.1, pp. 1–7.

Jiang, Q. F., H. Hong, H. Lv, et al. (2025). "A generalized predictive control method for the coolant system of nuclear heating reactor". In: *Energy* 337, p. 138432.

NSA Division (2001). *RELAP5/MOD3.3 code manual, Volume 4: Models and correlations*. Technical Report. USA: Information Systems Laboratories, Inc.

Qiu, Guihui, Feng Jiang, Yuangang Duan, et al. (2025). "Research and Validation of Hualong One HL-T67 Steam Generator". Chinese. In: *Nuclear Science and Engineering* 45.6, pp. 1110–1119.

Wu, S. F., Y. Z. Chai, J. S. Wan, et al. (2025). "A modular, parametric and visual modeling & simulation method for pressurized water reactor". In: *Energy* 326, p. 136324.

DOI
10.3384/ecp223237

Xu, Zhitao, Aijun Niu, Yufan Lyu, et al. (2025). “Simulation of Dynamic Response to Step Reactivity of TRIGA Research Reactor Based on Modelica”. Chinese. In: *Nuclear Techniques* 48.12, pp. 97–104.

Zhang, Liqun (2025). “Exploration Research on Automatic Modeling of Nuclear Power Plant Based on Modelica”. Chinese. In: *Nuclear Science and Engineering* 45.1, pp. 90–97.

DOI
10.3384/ecp223237

Modelica-based Modeling and Simulation of an SPWR Nuclear Steam Supply System

Qingfeng Jiang[1] Pengfei Wang[1*]

[1]School of Nuclear Science and Technology, Xi'an Jiaotong University, Xi'an, 710049, China,
wangpengfei@xjtu.edu.cn

Abstract

Small pressurized water reactors (SPWRs) have emerged as one of the most promising nuclear energy technologies, as they can simultaneously meet current decarbonization goals and the demands of flexible load operation. The nuclear steam supply system (NSSS) is the heart of an SPWR. Traditional text-based modeling approaches lack sufficient modularity, depth, and visualization capabilities. Modelica, which features equation-based, non-causal, object-oriented, and hierarchical modeling, enables rapid, visual, and reusable simulation of nuclear energy systems. Leveraging these advantages, this paper developed a modular, hierarchical dynamic simulation model library to support the rapid modeling and optimal design of SPWRs. Based on the system topology, an SPWR NSSS simulation model was quickly constructed using the developed library, and a corresponding control system was also developed. The accuracy of the modeling method is verified through typical load-change transients.

Keywords: SPWR NSSS; Modelica model library develop; Control system; Simulation verification

1 Introduction

The International Atomic Energy Agency (IAEA) defines nuclear reactors with an electrical generating capacity of 300 MWe or less as small reactors (Liu and Fan, 2014). Against the backdrop of global decarbonization and accelerating energy transition, small modular reactors (SMRs) have emerged as one of the most promising technological pathways for expanding nuclear energy applications beyond traditional large-scale baseload power plants (Didi, 2025). Among these, small pressurized water reactors (SPWRs) are currently one of the most mature and widely adopted reactor types in the SMR sector. In addition to electricity generation, SPWRs can supply high-temperature heat for industrial processes, district heating, hydrogen production, and seawater desalination (Wang et al., 2022a). Meanwhile, the increasing share of variable renewable energy sources such as wind and solar power is shifting nuclear power plant operating modes from steady-state baseload operation to more flexible load-following modes (Hou and Wei, 2018), which imposes new demands on performance assessment and control system design.

The nuclear steam supply system (NSSS) is the heart of an SPWR. The SPWR NSSS comprises the reactor core, steam generators, pressurizers, main coolant pumps, and associated pipes, valves, and control systems (Wang et al., 2022b). The traditional modeling methods for nuclear thermal-hydraulic systems rely primarily on text-based programming languages, such as RELAP5 and CENTs. Although these methods are well validated and widely adopted, they suffer from limited modularity and insufficient visualization capabilities. Modelica, an equation-based, non-causal, object-oriented modeling language, enables the construction of reusable, modular component libraries (Zhao et al, 2006). These components can be assembled into complex system models with minimal modification to the underlying equations, offering significant convenience for rapid modeling of nuclear power plants. Modelica's inherent hierarchical modeling paradigm allows users to extract generic base classes from the fundamental conservation equations of mass, momentum, and energy, and to build specific equipment models through multi-level in-heritance and graphical connections, thereby achieving a balance between modeling precision and visualization. Furthermore, extensive research has demonstrated the feasibility of using Modelica for the nuclear reactor modeling tasks. Examples include Modelica-based simulations of a nuclear heating reactor coolant system (Jiang et al., 2025), simulations of two-phase flow instabilities in parallel multi-channel systems (Tong et al., 2024), and the TRANSFORM simulation library specifically developed by Oak Ridge National Laboratory (ORNL) for modeling nuclear energy systems (Greenwood et al., 2017).

Although the feasibility of Modelica for thermal-hydraulic simulation of nuclear power systems has been preliminarily demonstrated, existing research has largely focused on standalone modeling of individual components, specific physical processes, or particular reactor types. Consequently, a standardized dynamic simulation model library tailored specifically for the SPWR NSSS has yet to be developed. To address this gap, this paper leverages the unique advantages of Modelica in rapid modeling and

multidisciplinary unified simulation to develop a modular, hierarchical dynamic simulation model library for the SPWR NSSS. Implemented in the Modelica language, this model library supports rapid assembly and parameter configuration for target reactor types. Using the developed library, a simulation model of the SPWR NSSS and its corresponding control system is rapidly constructed, and the accuracy and effectiveness of the proposed modeling method are verified through typical operating conditions.

2 Model Library Development

2.1 System Overview

The SPWR NSSS features a compact layout, comprising a reactor pressure vessel, two once-through steam generators (OTSGs), two coolant pumps, a pressurizer, and other auxiliary equipment. The reactor core is housed in the lower section of the reactor pressure vessel, with the two OTSGs arranged symmetrically on the outside of the vessel. Each OTSG is connected to a main coolant pump, which drives the circulation of the coolant. The system layout is shown in Figure 1 (Wang et al., 2022b).

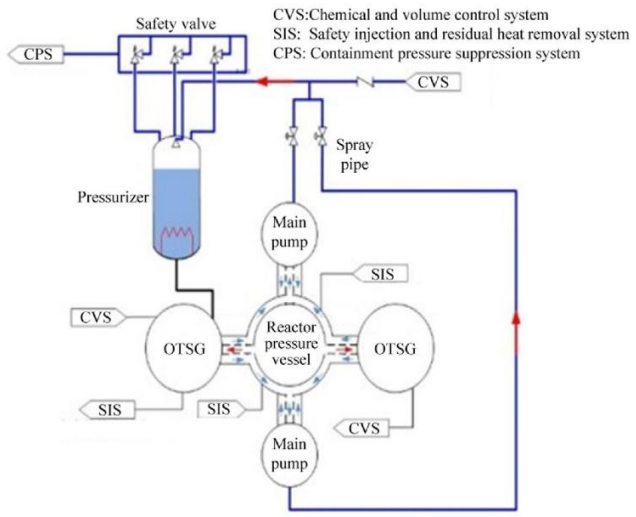


Figure 1. Schematic Diagram of the SPWR NSSS Structure.

2.2 Model Library Development

A top-down hierarchical decomposition method was adopted to establish the modeling library for the nuclear reactor system. The hierarchical decomposition process from the system level down to the equipment level, component level, and basic level is illustrated in Figure 2. First, the system topology was decomposed to break down the SPWR NSSS into equipment-level and component-level models, with the interconnections between them determined. The equipment-level models have the reactor core, pressurizer, and OTSG, while the component-level models include pipes, vessels, pumps, valves, and so forth. Next, a structural unit decomposition of the equipment-level models was performed, breaking them down into component-level and basic-level models. For example, the OTSG model can be represented as two dynamic pipes exchanging heat through their tube walls. Finally, the component-level and basic-level models obtained in the previous steps were further decomposed. The component-level models are broken down into models for pipes, volumes, pumps/valves, heat structures, sensors, and so on. The underlying fundamental physical laws and basic models were then distilled to form a basic-level model that encompasses fundamental equations, functions, material properties, interfaces, icons, and so on.

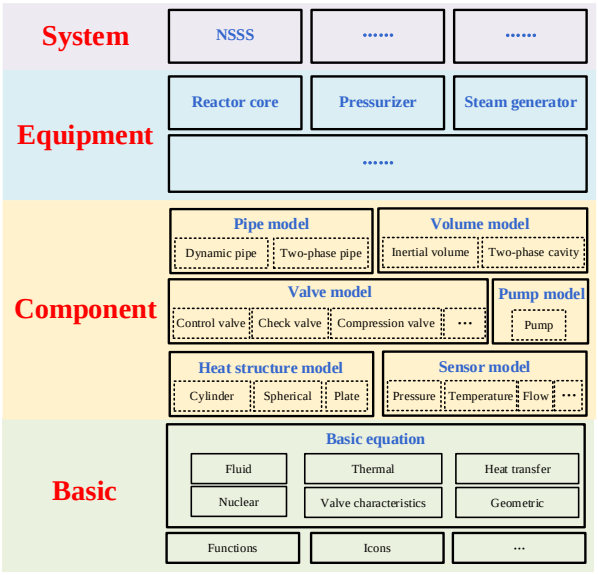


Figure 2. Dynamic Model Library Hierarchy.

Due to space constraints, the mathematical models used in the SPWR-NSSS dynamic model library are briefly described below:

1) The physical model of reactor core employs a point-reactor neutron dynamic model with six groups of delayed neutrons (Hetrick and Jarvis, 1971).
2) The cylindrical heat structure, which simulates heat transfer through the tube wall, is a two-dimensional heat-transfer model accounting for both axial and radial heat transfer (Jiang et al., 2025). Based on this, a fuel rod model is constructed that considers three material layers: the fuel core, gas gap, and cladding (Jiang et al., 2025).
3) The coolant, feedwater, and steam channels are all simulated using dynamic pipes based on the three-equation model for homogeneous flow (Jiang et al., 2025). Due to phase change on the secondary side of the OTSG, a two-phase dynamic pipe model is established using the finite volume method to ensure computational stability. For the specific model, please refer to the Ref (Casella, 2006).

DOI
10.3384/ecp223245

4) The working fluid in feedwater valves and spray valves is liquid water, and the control valve model described in the Ref (El Hejni and Bouskela, 2019) is adopted. Since steam valves must account for the compressibility of the working fluid, they are modeled as compressible valves (Lin et al., 2018).
5) The space within the pressurizer is divided into a fluctuating liquid phase region, a main liquid phase region, and a steam region. A three-zone non-equilibrium model of the pressurizer is established by considering mass and energy conservation in each region (Wang et al., 2020). The upper and lower chamber of the reactor core and the steam generator inlet and outlet headers are modeled using inertial volume models that account for mass and energy conservation (Jiang et al., 2025).
6) The main coolant pumps and feedwater pumps are simulated using the "PrescribedPump" model from the Modelica Standard Library.
7) The reactor core model is composed of a neutron dynamics model, a power distribution curve, a fuel rod, and a dynamic pipe with drag connections. The steam generator model is composed of a dynamic pipe (simulating the primary side), a heat structure, and a two-phase dynamic pipe with drag connections.

2.3 SPWR NSSS Simulation Model

Based on the SPWR-NSSS model library described in Section 2.2 and the SPWR system topology illustrated in Figure 1, a dynamic simulation model of the SPWR NSSS was developed, as shown in Figure 3. The system model comprises a reactor core, two OTSGs, two main pumps, a pressurizer, and several pipe, pump, and valve models.

3 Control System development

For SPWRs employing OTSGs, the most common steady-state operating mode is the dual-constant mode, in which both the average coolant temperature and the main steam pressure are maintained at constant values. The control systems involved in the SPWR NSSS include the reactor power control system, the steam generator feedwater control system, and the pressurizer pressure and level control systems (Wang et al., 2022b).

3.1 Reactor Power Control System

The reactor power control system regulates the reactivity introduced by control rods to maintain the reactor power and the average coolant temperature at their setpoints. The reactor power control system used in this study employs a double feedback control structure for both reactor power and average coolant temperature; specifically, the reactor power and average coolant temperature are controlled via power and temperature feedback loops, respectively. The principle of this control system is illustrated in Figure 4 (Wang et al., 2022b).

3.2 OTSG Feedwater Control System

The OTSG feedwater control system was designed using a feedwater flow control mode, configured as a cascade control system. The outer loop controls the main steam pressure, while the inner loop regulates the feedwater flow by adjusting the feedwater valve opening. Feedwater flow is influenced by both the valve opening and the pressure difference across the feedwater valve. To ensure that the feedwater flow is determined solely by the valve opening, the pressure difference across the feedwater valve is maintained at a constant level by adjusting the feedwater pump speed. When the turbine load changes, the system

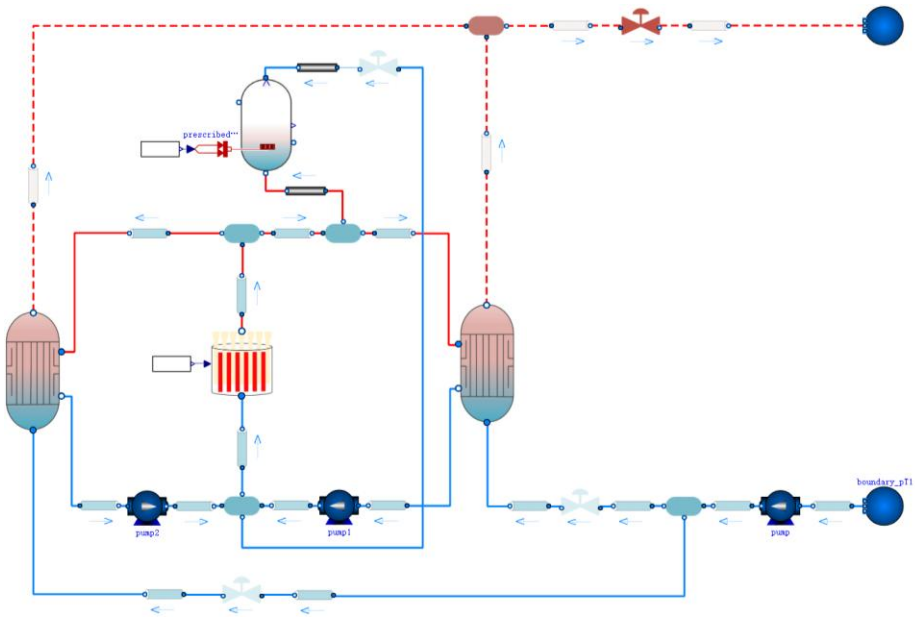


Figure 3. Dynamic Simulation Model of an SPWR NSSS.

adjusts the main steam valve, causing a disturbance in steam pressure. Then the system adjusts the feedwater valve to maintain a constant steam pressure. The control principle of the OTSG feedwater control system is shown in Figure 5 (Wang et al., 2022b). It is worth noting that when steam pressure becomes excessively high, the bypass valve is opened to allow a portion of the steam to quickly enter the condenser, thereby achieving rapid pressure relief of the main steam.

3.3 Pressurizer Pressure Control System

The pressure control system maintains or restores the pressurizer pressure to a preset target value. Its control logic is shown in Figure 6 (Wang et al., 2022b). The difference between the measured pressurizer pressure and the setpoint is processed through PID control to generate a compensation pressure differential signal. This signal is sent to a function generator and a threshold relay, which is used to control the pressurizer's electric heaters and spray valve, thereby regulating the pressurizer pressure. The pressurizer's electric heaters consist of variable heaters and backup heaters. The variable heater power and the spray valve opening are linearly adjusted based on the compensation pressure differential signal, while the activation of the backup heater is determined by the threshold relay. When the pressurizer pressure deviation falls below a certain threshold, the backup heater activates.

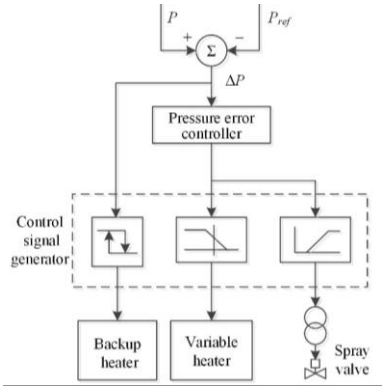


Figure 6. Schematic Diagram of the Pressurizer Pressure Control System

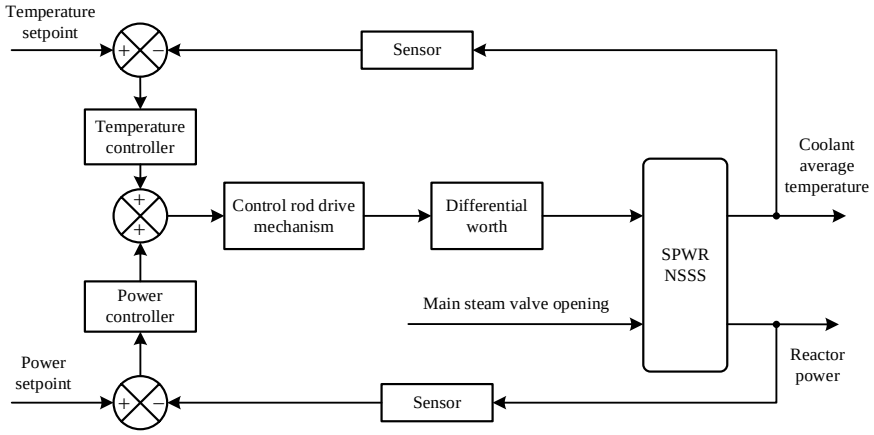


Figure 4. Schematic Diagram of the Reactor Power Control System.

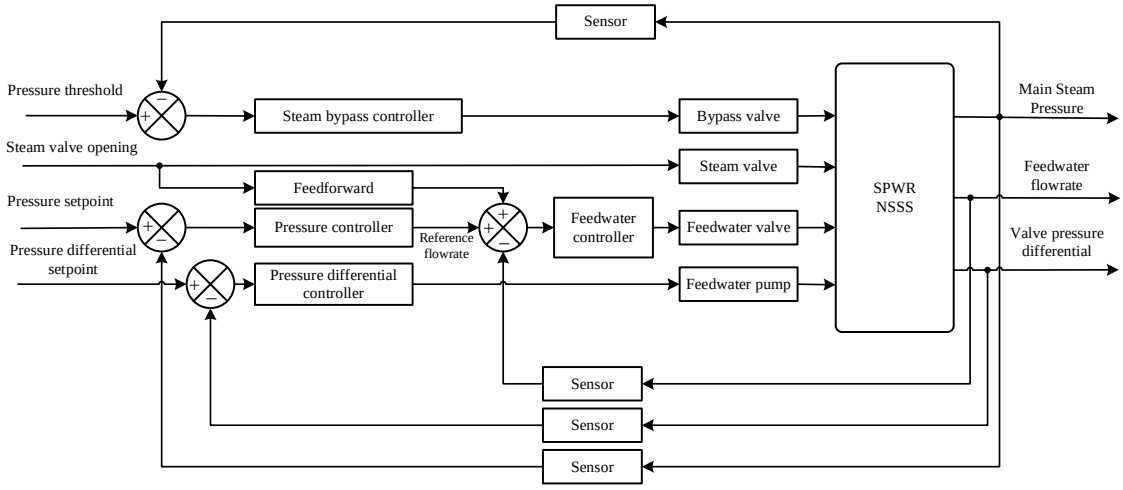


Figure 5. Schematic Diagram of the OTSG Feedwater Control System.

DOI
10.3384/ecp223245

3.4 Pressurizer Level Control System

The purpose of the pressurizer level control system is to maintain the pressurizer level at a constant value, thereby preventing safety issues caused by excessively high or low pressurizer levels during operation. The pressurizer level control principle is shown in Figure 7 (Wang et al., 2022b). The basic method of pressurizer level control is to fix the letdown flow rate and adjust the pressurizer level by changing the charging flow rate.

4 Simulation Validation

To verify the accuracy and reliability of the Modelica-based SPWR NSSS dynamic simulation model, transient simulations were conducted under typical load change transients to analyze the validity of the dynamic responses of the SPWR NSSS key parameters.

4.1 Load Step Decrease

At 100 s, the turbine load drops abruptly from 100% full power (FP) to 90% FP and remains at 90% FP for a period. The dynamic responses of system's key parameters under this load step decrease transient are shown in Figure 8.

When the demand load decreases, the main steam valve opening is reduced, resulting in a decreased steam flow. This causes the heat transfer rate of the steam generator to drop rapidly, leading to a gradual increase in the average coolant temperature. The increase in coolant temperature causes the coolant to expand, generating a positive flow fluctuation into the pressurizer; consequently, the pressure and level in the pressurizer increase as the average coolant temperature rises. At this point, the reactor power setpoint is reduced, generating a control rod insertion signal, and the reactor power begins to decrease, with the average

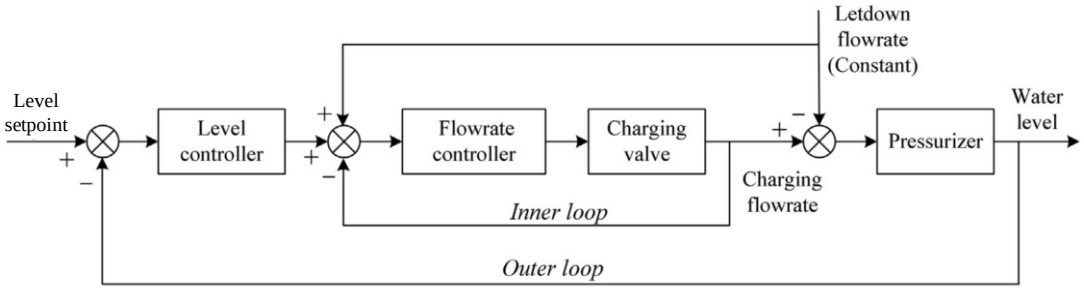


Figure 7. Schematic Diagram of the Pressurizer Level Control System.

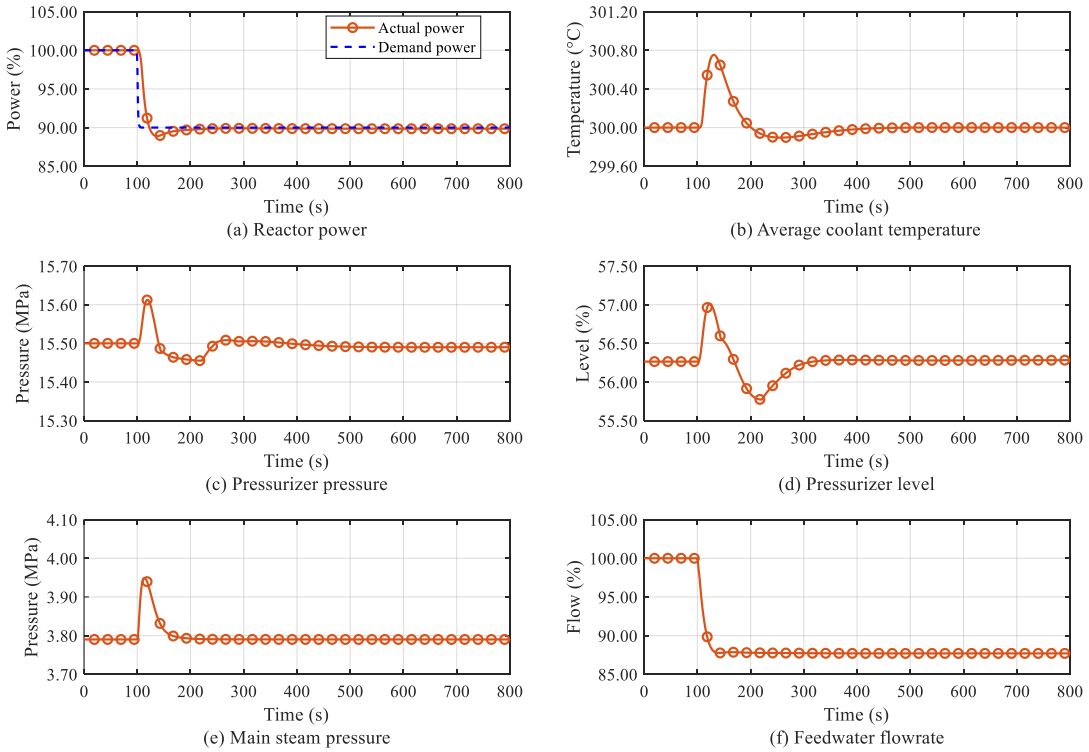


Figure 8. Dynamic Response Results for Load Step Decrease.

coolant temperature eventually returning to its setpoint. Additionally, the pressurizer pressure and level return to their setpoints under the action of pressurizer pressure and level control systems.

During the load step decrease transient, the main steam valve opening suddenly decreases, causing a reduction in main steam flow rate. Since the feedwater flow has not yet changed at this moment, and the outlet steam flow rate suddenly drops, a large amount of steam accumulates on the secondary side, causing the secondary steam pressure to rise abruptly. Subsequently, the steam pressure control system activates, reducing the feedwater flow rate at the secondary inlet. This gradually brings the feedwater flow rate closer to the outlet steam flow rate. At this point, the steam pressure rises to a maximum value before gradually decreasing and eventually stabilizing at the setpoint.

4.2 Load Linear Decrease

The turbine load decreases linearly from 100% FP to 70% FP at 100 s, at a constant rate of 10% FP per minute, and remains at 70% FP for a period. The dynamic responses of the system's key parameters under the load linear decrease transient are shown in Figure 9.

When the secondary loop load decreases linearly, the power control system generates a continuous control rod insertion signal, enabling the reactor power to closely follow the demand load changes. During the load decrease, the secondary-side steam flow rate of the steam generator decreases, leading to a reduction in the steam generator's heat transfer rate. Consequently, the average coolant temperature continues to rise until the demanded power reaches the setpoint, after which the average coolant temperature begins to decrease and eventually stabilizes. The pressurizer pressure and level rise with the increase in average coolant temperature, and then stabilize under the action of the their respective control system.

At the onset of the linear load decrease, the steam flow rate decreases, but the feedwater flow has not yet changed; therefore, the steam pressure rises rapidly. Subsequently, the main steam pressure control system engages, and the feedwater flow gradually decreases in tandem with the steam flow. However, because the feedwater flow remains greater than the steam flow, the steam pressure continues to rise slowly until the main steam valve stops closing, at which point the feedwater flow gradually equals the steam flow. Since the steam flow changes gradually throughout the entire operating condition, the feedwater flow rate can effectively follow the steam flow rate changes. Ultimately, the steam flow equals the feedwater flow, and the steam pressure stabilizes at the setpoint.

5 Conclusion

This study proposes and validates a Modelica-based dynamic simulation model library for the nuclear steam supply system (NSSS) of small pressurized water reactors (SPWRs). The library is implemented using a modular,

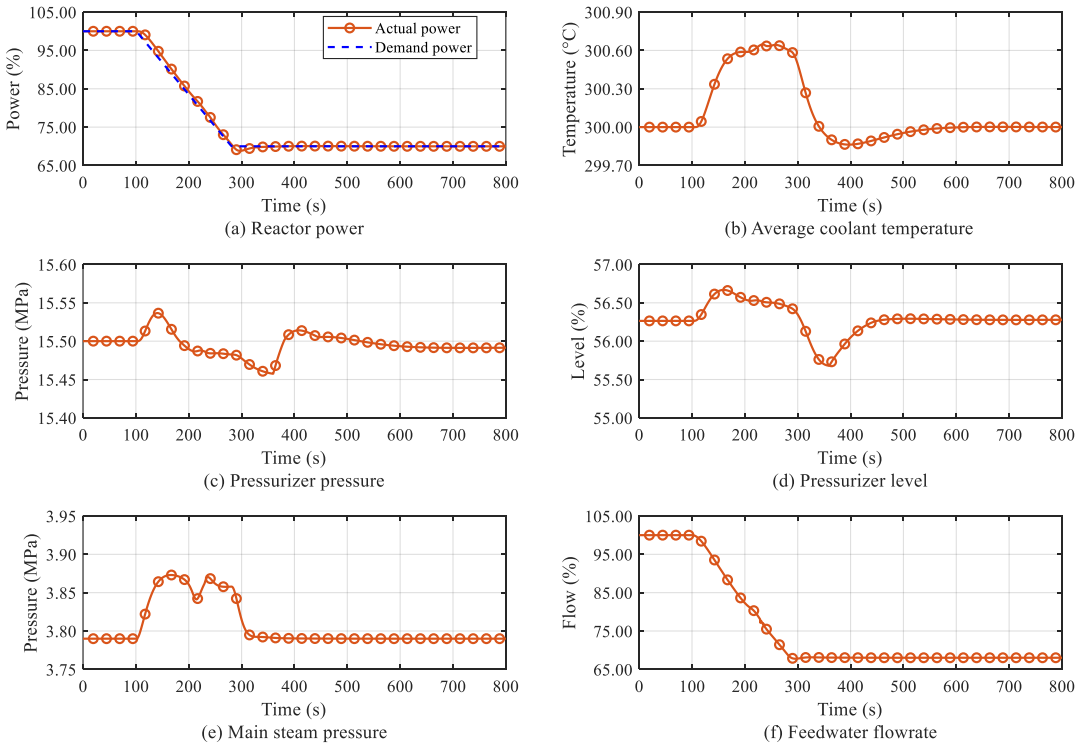


Figure 9. Dynamic Response Results for Load Linear Decrease.

DOI
10.3384/ecp223245

hierarchical architecture. First, starting from the basic physical laws governing each physical process, we construct the universal base classes, such as the three-equation model for homogeneous flow describing fluid dynamics and the point-reactor neutron dynamic equation. Subsequently, specific component-level and equipment-level models are constructed through multi-level inheritance and drag-and-drop connections. Finally, based on the system topology, a dynamic simulation model of the SPWR NSSS is rapidly assembled, and its corresponding control system is developed. To verify the effectiveness of the modeling approach, several simulation validations were conducted under typical load change transients. The simulation results shown that the dynamic response trends of the key parameters of the SPWR NSSS are reasonable and consistent with theoretical analyses, demonstrating the reliability of the proposed modeling method. However, the simulation model of the nuclear steam supply system for SPWR developed in this study is applicable only to the dynamic simulation of normal operating transients; it cannot simulate accident conditions that may occur in a nuclear power plant. In future work, more detailed component/equipment models will be developed to enable the dynamic simulation of accident conditions.

References

Liu, Z.T., Fan, J.H. (2014). “Technology readiness assessment of Small Modular Reactor (SMR) designs”. In: Progress in Nuclear Energy 70, p. 20-28.

Didi, A. (2025). “Emerging Small Modular Reactors: Comparative Insights into Technologies, Passive Safety, and Future Energy Applications”. In: Neutron News 36(2), p. 11–17.

Wang, P.F., Jiang, Q.F., Wan, J.S., et al. (2022a). “Robust controller design for small pressurized water reactors using non-smooth optimization”. In: Annals of Nuclear Energy, 165, p. 108775.

Hou, W.T., Wei, H. (2018). “A multi-source coordinated short-term dispatch model considering the dispatchability of nuclear power plants”. Chinese. In: Transactions of China Electrotechnical Society 33(12), p. 2873-2882.

Wang, P.F., Jiang, Q.F., Zhang, J.X., et al. (2022b). “A fuzzy fault accommodation method for nuclear power plants under actuator stuck faults”. In: Annals of Nuclear Energy 165, p 108674.

Zhao, J.J., Ding, J.W., Zhou, F.L., et al. (2006) “Modelica and Its Mechanism of Multi-domain Unified Modeling and Simulation”. Chinese. In: Journal of System Simulation S2, p. 570-573..

Jiang, Q. F., H. Hong, H. Lv, et al. (2025). “A generalized predictive control method for the coolant system of nuclear heating reactor”. In: Energy 337, p. 138432.

Tong, Q.S., Deng, K.J., Lyu, X., et al. (2024). “Simulation research on flow instability of parallel multi-channel system based on modelica”. Chinese. In: Software Guide 23(01), p. 21-25.

Greenwood, M., Cetiner, M.S., Fugate, D., et al. (2017). “TRANSFORM-TRANsient simulation framework of reconfigurable models”.

Hetrick, D.L., Jarvis, R.G. (1971). “Dynamics of nuclear reactors”. American Nuclear Society.

Casella, F. (2006). “Object-Oriented Modelling of Two-Phase Fluid Flows by the Finite Volume Method”. In: Proceedings 5th Mathmod Vienna.

El Hejni, B. and Bouskela, D. (2019). “Modeling and simulation of thermal power plants with ThermoSysPro”. Cham:Springer.

Lin, J.G., Zeng, X.L., Li, J.L. (2018). “Numerical calculation method of flow rate of the steam valve underhigh differential pressure and large flow”. Chinese. In: Mechanical and Electrical Equipment 35(2), p. 24-26, 30.

Wang, P.F., Wan, J.S., Wu, S.F., et al. (2020). “A fuzzy-logic-based pressure setpoint modification method for pressurized water reactor pressurizers”. In: Annals of Nuclear Energy, 135. p 106954.

DOI
10.3384/ecp223245

Modelica-Based Parametric Analysis of Steady-State and Transient Thermal-Hydraulic Characteristics of PCHEs in S-CO_2 Brayton Cycles

Hao Xu[1,2] Yifan Xu[1,2*] Yuandong Zhang[1,2] Minjun Peng[1,2] Genglei Xia[1,2]

[1]College of Nuclear Science and Technology, Harbin Engineering University, Harbin 150001, China;

[2]State Key Laboratory of Marine Thermal Energy and Power, Harbin Engineering University, Harbin 150001, China

Abstract

Printed circuit heat exchangers (PCHEs) are key components in supercritical carbon dioxide (S-CO_2) Brayton cycles because of their compact structure and high heat transfer capability. However, existing studies have mainly focused on steady-state structural optimization or isolated component analysis, while the influence of geometric parameters on thermal-hydraulic responses under system-level transient disturbances remains insufficiently understood. In this study, a Modelica-based dynamic simulation framework is developed for the system-level analysis of recuperators in an S-CO_2 Brayton cycle. A one-dimensional dynamic PCHE model is established and validated against benchmark data, with a maximum relative error of 0.87%. A continuously varying heat sink temperature ranging from 22 ℃ to 26 ℃ is introduced as a representative off-design disturbance. The effects of channel diameter and channel length on heat transfer coefficient and pressure drop are investigated under both design reference and transient operating conditions. The results show that smaller channels and longer flow paths enhance heat transfer performance but introduce larger pressure-drop penalties under the design reference condition. Under transient heat sink temperature disturbances, small-diameter and long-channel configurations exhibit stronger thermal-hydraulic response variations than under steady-state conditions. These results indicate that parameter trends obtained under steady-state conditions may not directly represent transient thermal-hydraulic behavior under off-design operation. The proposed framework provides an efficient approach for preliminary parametric assessment and system-level transient analysis of recuperators in S-CO_2 Brayton cycles.

Keywords: PCHE, S-CO2, Brayton cycle, Modelica, Thermal-hydraulic characteristics

1 Introduction

Small modular reactor (SMR) systems have attracted extensive attention because of their advantages in compactness, modular deployment, operational flexibility, and long-term economic potential (Mignacca et al., 2020; Pioro et al., 2020). Among various advanced power conversion technologies, the supercritical carbon dioxide (S-CO_2) Brayton cycle is considered a promising option for SMR applications due to its compact layout, high thermal efficiency, and reduced compressor work near the critical point. However, the strong thermophysical property variations of S-CO_2 in the near-critical region also introduce significant challenges for the thermal-hydraulic stability and dynamic response of cycle components under off-design operating conditions.

As a key heat transfer component in the S-CO_2 Brayton cycle, the printed circuit heat exchanger (PCHE) plays an important role in cycle heat recovery and strongly affects the overall system performance. Owing to its compact structure, high pressure resistance, and excellent heat transfer capability, the PCHE is considered particularly suitable for high-temperature and high-pressure S-CO_2 applications (Dostal et al., 2006). Consequently, the thermal-hydraulic characteristics of PCHEs have been extensively investigated through numerical simulation, experimental analysis, and structural optimization.

Existing studies have mainly focused on steady-state thermal-hydraulic analysis and geometric optimization of PCHE channels. Regarding modeling approaches, Sun et al. (2025) developed a one-dimensional model incorporating ambient heat-loss correction, although a predictive deviation of approximately 5% remained. Ming et al. (2024) established a dynamic one-dimensional discretization model capable of capturing thermal inertia effects, with a maximum relative error of 3.16% compared with CFD results. In terms of structural optimization, NSGA-II-based multi-objective optimization methods have been widely applied to improve PCHE performance (Kalyanmoy et al., 2002). Ding et al. (2024) combined

CFD, artificial neural networks, and NSGA-II to reduce frictional losses while maintaining thermal performance. Kuo et al. (2024) optimized airfoil fin geometries to decrease pressure drop, while Zhang et al. (2020) employed neural networks and genetic algorithms to reduce pressure loss in circular-channel exchangers. In addition, Jin et al. (2022) and Yang et al. (2020) demonstrated that optimized zigzag-channel structures could reduce equipment volume and hydraulic resistance under equivalent thermal conditions.

Although these studies have improved the understanding of PCHE thermal-hydraulic behavior, most existing research remains limited to steady-state operating conditions, local CFD flow analysis, or isolated component optimization. Under realistic operating conditions, however, fluctuations in heat sink temperature and variations in system boundary conditions may induce strong transient thermal-hydraulic responses in S-CO_2 Brayton cycles, particularly near the critical region where fluid properties vary significantly. Under such conditions, geometric parameter trends obtained under steady-state conditions may not directly represent transient operating behavior. In particular, channel configurations with favorable steady-state heat transfer performance may simultaneously exhibit amplified thermal-hydraulic fluctuations during transient operation.

To address this issue, the present study establishes a Modelica-based system-level dynamic simulation framework for investigating the thermal-hydraulic response of a recuperator under both design reference and transient off-design conditions. A one-dimensional dynamic PCHE model is developed and integrated into the S-CO_2 Brayton cycle system. By introducing heat sink temperature fluctuations as representative transient disturbances, the effects of channel diameter and channel length on heat transfer coefficient and pressure drop are systematically analyzed. The objective of this work is not to identify a unique optimal geometry, but to clarify how geometric parameters influence the coupled thermal and hydraulic responses of recuperators under transient operating conditions. The results provide theoretical support for the system-level thermal-hydraulic evaluation and preliminary parameter design of PCHEs in S-CO_2 Brayton cycle systems.

2 Models and Methods

2.1 System Description

A typical S-CO_2 Brayton cycle system is shown in Figure 1. The system mainly consists of a heat source, turbine, compressor, recuperator, and cooler. Among these components, the recuperator plays a central role in internal heat recovery and strongly affects the thermal efficiency and dynamic behavior of the overall cycle. Therefore, this study focuses on the thermal-hydraulic response of the recuperator under both design reference and off-design transient conditions.

In this work, a dynamic model of the recuperator is developed using the Modelica language and integrated into the S-CO_2 Brayton cycle system. The coupled differential-algebraic equations are solved using the DASSL solver with variable-step numerical integration. This modeling framework allows the recuperator to be analyzed under system-level boundary variations, rather than as an isolated component with fixed inlet conditions. Specifically, the recuperator investigated in this model is configured as a PCHE with semicircular channels.

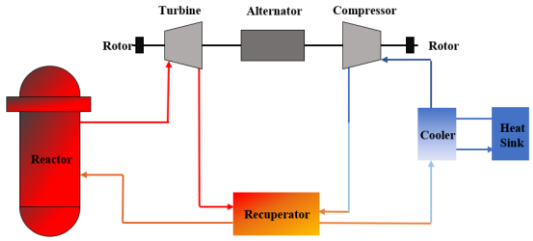


Figure 1. The S-CO_2 Brayton energy cycle system.

To investigate the influence of geometric parameters on the thermal-hydraulic performance of the recuperator, the most commonly used semicircular straight-channel recuperator in the Brayton cycle is selected as the reference geometry, as shown in Figure 2. The channel diameter and channel length are then varied to evaluate their effects on heat transfer coefficient and pressure drop. This configuration provides a representative basis for examining the competing effects of heat transfer enhancement and hydraulic resistance in recuperators.

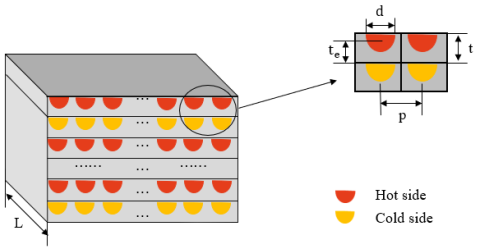


Figure 2. Geometry of the semicircular straight channels.

2.2 Modeling and Validation

Due to the large length-to-diameter ratio of PCHE channels, the dominant variations in temperature, pressure, and fluid properties occur along the axial direction. A lumped-parameter model is therefore insufficient to describe the spatial evolution of the working fluid inside the channels. In this study, a one-dimensional discretized modeling approach is adopted to resolve the axial distribution of thermal-hydraulic parameters while maintaining acceptable computational efficiency for system-level transient simulations.

The model is developed based on the following assumptions.

(1) One-dimensional axial flow assumption. The radial variations in temperature and pressure are neglected, and the main variations of thermal-hydraulic parameters are assumed to occur along the flow direction (Johnston et al.,

DOI
10.3384/ecp223253

2001). Under the investigated operating conditions, the flow is predominantly turbulent, and the one-dimensional approximation is considered suitable for capturing the overall axial transport behavior of the PCHE.

(2) Negligible heat loss to the environment. The external surface area of the recuperator is much smaller than its internal heat transfer area. In practical applications, PCHEs are also usually equipped with thermal insulation (Carstens et al., 2006). Therefore, heat loss from the external shell to the ambient environment is neglected.

Figure 3 depicts the discretization scheme of the PCHE channel. The hot and cold S-CO_2 streams are arranged in a counterflow configuration. A pair of adjacent hot-side and cold-side channels is defined as a basic heat transfer unit and discretized into N control volumes along the axial direction. For each control volume, mass, momentum, and energy conservation equations are established. The outlet state of one node is transferred to the inlet of the next node, allowing the axial evolution of temperature, pressure, enthalpy, and heat transfer rate to be calculated.

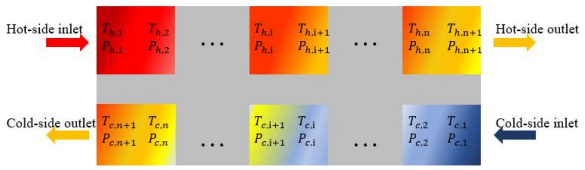


Figure 3. Discrete node division of heat transfer channels.

For a semicircular cross-section channel, the hydraulic diameter (d_e, m) and the equivalent thickness of the metal wall (t_e, m) are calculated as follows:

$$d_e = \frac{4\pi d^2}{8\left(\pi\frac{d}{2}+d\right)} \tag{1}$$

$$t_e = t - \frac{\pi}{8}d \tag{2}$$

These parameters are employed in the evaluation of the Reynolds number, Nusselt number, and friction factor along the flow direction.Where: d is the semicircular diameter, m. t is the metal plate thickness, m.

For each discrete node, the boundary variables include the inlet and outlet pressure, temperature, and mass flow rate of the working fluid. The governing equations are established based on mass, momentum, and energy conservation:

$$\frac{dh_{out}}{d\tau} = \frac{m\cdot h - m_{out}\cdot h_{out} - q}{M} \tag{3}$$

$$M = \frac{\pi d^2 L\rho}{8} \tag{4}$$

$$\rho = f(h_i,\ P_i) \tag{5}$$

Where, h and h_{out} are the inlet and outlet enthalpies of the fluid, J/kg. m and m_{out} are the inlet and outlet fluid mass flow rate per channel, kg/s. q is the convective heat-transfer rates between the fluid and wall, W. M is the fluid mass within a single channel, kg. And the local fluid density ρ is evaluated using REFPROP based on the local pressure (P_i) and specific enthalpy (h_i).

The energy conservation equation for the metal wall is given by:

$$\frac{dT_w}{d\tau} = \frac{q_h - q_c}{M_w C_w} \tag{6}$$

$$M_w = (p\cdot t - \frac{\pi d^2}{8})\cdot L\cdot \rho_w \tag{7}$$

Where: T_w is the wall temperature at each node, K. M_w is the mass of the metal wall, kg. C_w represents the specific heat capacity, J/(kg·K). and ρ_w denotes the density of the metal wall, kg/m^3.

The heat transfer between the hot and cold fluids is represented by three thermal resistances in series: convective heat transfer on the hot side, thermal conduction through the metal wall, and convective heat transfer on the cold side. The overall heat transfer coefficient (U, W/(m^2·K)) and the nodal heat transfer rates are calculated as:

$$\frac{1}{U} = \frac{1}{K_h} + \frac{t_e}{\lambda_w} + \frac{1}{K_c} \tag{8}$$

$$q_h = K_h\cdot A\cdot (T_h - T_w) \tag{9}$$

$$q_c = K_c\cdot A\cdot (T_w - T_c) \tag{10}$$

$$A = L(\frac{\pi}{2}+1)d \tag{11}$$

$$T = f(h_i,\ P_i) \tag{12}$$

Where: K_h and K_c are the surface heat transfer coefficients of the hot and cold sides, W/(m^2·K). λ_w is the thermal conductivity of the metal wall, W/(m·K). q_h and q_c are the heat transfer rates of the hot and cold side working fluids, W. A is the heat transfer area, m^2. T is the node temperature, K.

The convective heat transfer coefficient is determined by:

$$h = \frac{Nu\cdot\lambda}{d_e} \tag{13}$$

$$\lambda = f(h_i,\ P_i) \tag{14}$$

Where: h is the specific enthalpy of the fluid, J/kg. λ is the thermal conductivity at different nodes, W/(m·K).

For the calculation of flow and heat transfer in S-CO_2, Liu et al. (2022) experimentally verified that the Gnielinski model is applicable to S-CO_2 as a working fluid. Therefore, this study uses the Gnielinski model (Gnielinski et al., 1976) to calculate the Nusselt number. The Gnielinski correlation is applicable for turbulent internal flow within the ranges of $2300 < Re < 5\times 10^6, 0.5 < Pr < 2000$. The Reynolds number and Prandtl number investigated in the present study remain within these applicability limits.

$$Nu = \frac{(f/8)(Re-1000)Pr}{1+12.7\sqrt{f/8}(Pr^{2/3}-1)} \tag{15}$$

$$f = (1.82\lg Re - 1.64)^{-2} \tag{16}$$

$$Pr = \frac{\mu \cdot C_p}{\lambda} \quad (17)$$

$$Re = \frac{8m \cdot d_e}{\pi d^2 \mu} \quad (18)$$

Where: f is the local friction factor, Re is the Reynolds number, and Pr is the Prandtl number.

$$\Delta p = p_n - p_{n+1} = \frac{fLm^2}{(\frac{\pi d^2}{8})^2 \cdot \rho \cdot 2 \cdot d_e} \quad (19)$$

Based on the above mathematical model and computational procedure, a dynamic PCHE model was established in Modelica. To evaluate the reliability of the model, reference data from a 5 MW small modular reactor S-CO_2 Brayton cycle system (Ming et al., 2022) were used for comparison. The validation results are listed in Table 2.

Table 1. Heat exchanger design parameters.

Parameters	*Units*	*Reference*
Channel length	mm	1400
Equivalent diameter	mm	2
Channel spacing	mm	3
Plate thickness	mm	1.5
Number of channels	-	18800/18800
Hot side inlet temperature	K	693.6
Hot side inlet pressure	MPa	8.64
Cold side inlet temperature	K	332.8
Cold side inlet pressure	MPa	15.61
Hot side mass flow	kg/s	26.3
Cold side mass flow	kg/s	26.3

2.3 Simulation Conditions and Parameter Design

A 5 MW small modular reactor S-CO_2 Brayton cycle is used as the reference system in this study. The recuperator design parameters and boundary conditions are listed in Table 1. Under the design reference condition, the system heat sink temperature is set to 10 ℃, which is used to establish the baseline thermal-hydraulic performance of the recuperator, the simulation model is shown in Figure 4.

In practical operation, the heat sink temperature may vary due to changes in external environmental conditions or fluctuations in cooling system performance. Such variations can affect the cooler outlet state and further influence the operating boundary of the recuperator through system coupling. To investigate the transient response of the recuperator under off-design conditions, a time-varying heat sink temperature disturbance ranging from 22 ℃ to 26 ℃ is introduced into the Modelica system simulation, as shown in Figure 5.

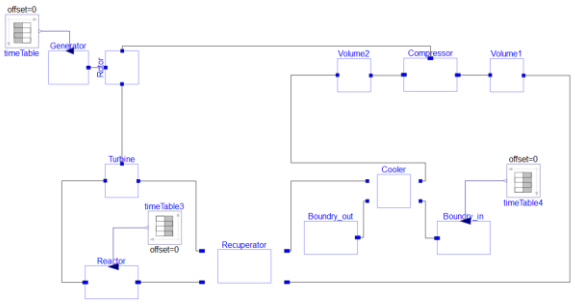


Figure 4. S-CO_2 Brayton cycle dynamic simulation model.

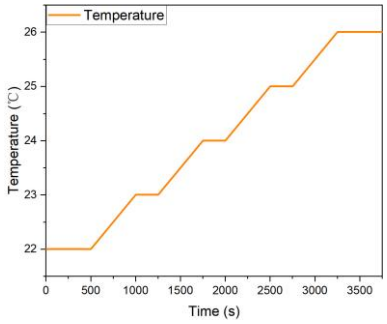


Figure 5. Variation of heat sink temperature over time.

The selected temperature range represents a relatively high heat sink temperature condition and is used to examine the dynamic sensitivity of the recuperator when the S-CO_2 properties in the low-temperature part of the cycle approach the near-critical region. Under this condition, variations in density, viscosity, specific heat capacity, and thermal conductivity may affect the Reynolds number, heat transfer coefficient, and pressure drop. Therefore, this transient boundary condition is suitable for analyzing how channel diameter and channel length influence the coupled thermal and hydraulic responses of the recuperator.

In this study, the channel diameter and channel length are varied independently while the remaining geometric and operating parameters are kept at their reference values. This design allows the individual influence of each geometric parameter to be identified under both the design reference condition and the transient heat sink temperature disturbance.

Table 2. Heat exchanger model validation.

Parameters	*Units*	*Symbol*	*Reference*	*Value*	*Error*
Hot side outlet temperature	K	$T_{h,out}$	335.8	338.2	0.71%
Hot side outlet pressure	MPa	$P_{h,out}$	8.31	8.28	0.36%
Cold side outlet temperature	K	$T_{c,out}$	611.8	606.5	0.87%
Cold side outlet pressure	MPa	$P_{c,out}$	15.52	15.47	0.32%

DOI
10.3384/ecp223253

3 Results and Discussion

3.1 The Effect of Recuperator Geometry on Thermal-Hydraulic Performance

This section investigates the effects of PCHE geometric parameters on the thermal-hydraulic performance of the recuperator under the design reference condition. Channel diameters of 1.5 mm, 2 mm, and 2.5 mm, together with channel lengths of 1000 mm, 1400 mm, and 1800 mm, were selected for analysis, while all remaining structural parameters were maintained at their reference values. The corresponding thermal-hydraulic results are presented in Table 3.

Table 3. Pressure drop and heat transfer coefficient for different structural parameters

Parameter	*Hot-side pressure drop* (MPa)	*Cold-side pressure drop* (MPa)	*Heat transfer coefficient* (W/(m²·K))
Channel diameter			
1.5 mm	1.31	0.48	86792.39
2 mm	0.36	0.14	53681.15
2.5 mm	0.13	0.05	36363.09
Channel length			
1000 mm	0.28	0.11	51638.57
1400 mm	0.36	0.14	53681.15
1800 mm	0.41	0.17	56095.38

As shown in Table 3, the channel diameter exerts a dominant influence on both heat transfer and flow resistance. Reducing the channel diameter increases the flow velocity and Reynolds number under the same mass flow condition, thereby enhancing convective heat transfer. When the channel diameter decreases from 2.5 mm to 1.5 mm, the heat transfer coefficient increases from 36363.09 W/(m²·K) to 86792.39 W/(m²·K). However, the corresponding pressure drop rises significantly because the reduced hydraulic diameter intensifies wall friction and local flow resistance. In contrast, larger channels suppress hydraulic losses but weaken the heat transfer capability because of the lower flow velocity and weaker convective transport capability.

The influence of channel length differs from that of channel diameter. Increasing the channel length mainly improves the total heat transfer capacity by extending the effective heat transfer area and thermal residence time. However, the additional flow distance also increases frictional losses along the channel. As the channel length increases from 1000 mm to 1800 mm, the total pressure drop rises from 0.39 MPa to 0.58 MPa. Compared with the influence of channel diameter, the effect of channel length on heat transfer enhancement is relatively moderate, whereas its contribution to pressure-drop accumulation becomes increasingly significant under long-channel conditions.

These results indicate that the thermal-hydraulic performance of the PCHE is governed by the competing effects of heat transfer enhancement and hydraulic resistance. Smaller channels and longer flow paths improve thermal performance but introduce a considerable pressure-drop penalty. Similar trends have also been reported in previous CFD-based studies of PCHE thermal-hydraulic characteristics (Tang, 2019), indicating that the present one-dimensional dynamic model can reproduce the main parameter dependence of the recuperator under the design reference condition.

3.2 Transient Response Characteristics of the Recuperator under Heat Sink Temperature Fluctuations

This section analyzes the thermal performance of the recuperator under transient conditions involving heat sink temperature fluctuations. As in the previous sections, design parameters are used as a reference. Based on the actual sea surface temperature at the nuclear power plant's water intake, as described in Section 2.3, the fluctuation range is set from 22℃ to 26℃. The time-varying curve is shown in Figure 5.

3.2.1 Dynamic Response of the Total Heat Transfer Coefficient

Figure 7 shows the transient response of the heat transfer coefficient under heat sink temperature fluctuations. The results show that the geometric parameters significantly affect the sensitivity of the recuperator to external thermal disturbances.

Among the investigated channel diameters, the smallest channel (1.5 mm) exhibits the largest fluctuation amplitude in heat transfer coefficient, reaching approximately 997.74 W/(m²·K), whereas the largest channel (2.5 mm) shows the smoothest response. The medium-diameter channel exhibits an intermediate behavior. This trend is primarily associated with the higher flow velocity and stronger thermal-hydraulic sensitivity in small channels. Under near-critical S-CO_2 conditions, small variations in temperature can induce substantial changes in fluid properties, including density, viscosity, and specific heat capacity. In smaller channels, these property variations lead to more pronounced changes in Reynolds number(see Figure 6.) and convective heat transfer behavior, thereby amplifying the fluctuation of the heat transfer coefficient.

The influence of channel length on transient thermal response exhibits a different characteristic. Short channels (1000 mm) show relatively weak fluctuations, with a variation amplitude of approximately 270.37 W/(m²·K), whereas long channels (1800 mm) exhibit significantly stronger dynamic responses. This phenomenon is related to the thermal residence time of the working fluid. In short

channels, the fluid passes through the heat exchanger rapidly, limiting the accumulation of changes in physical properties along the flow direction. By contrast, long channels provide a longer heat exchange path, allowing the cumulative effect of property variation to become more pronounced. Consequently, the outlet heat transfer coefficient becomes more sensitive to heat sink temperature disturbances.

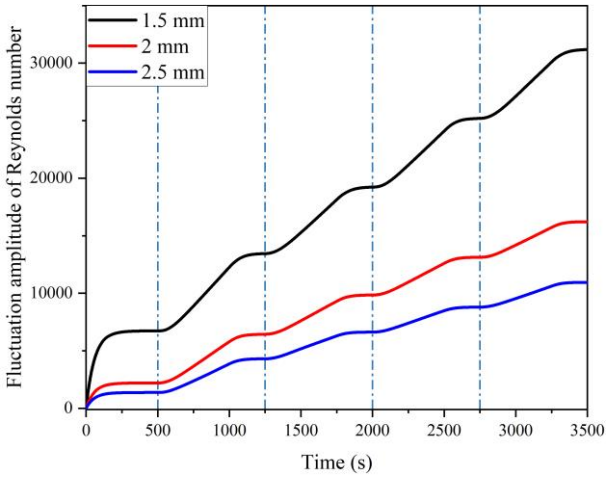


Figure 6. Reynolds number fluctuation amplitude at different diameters.

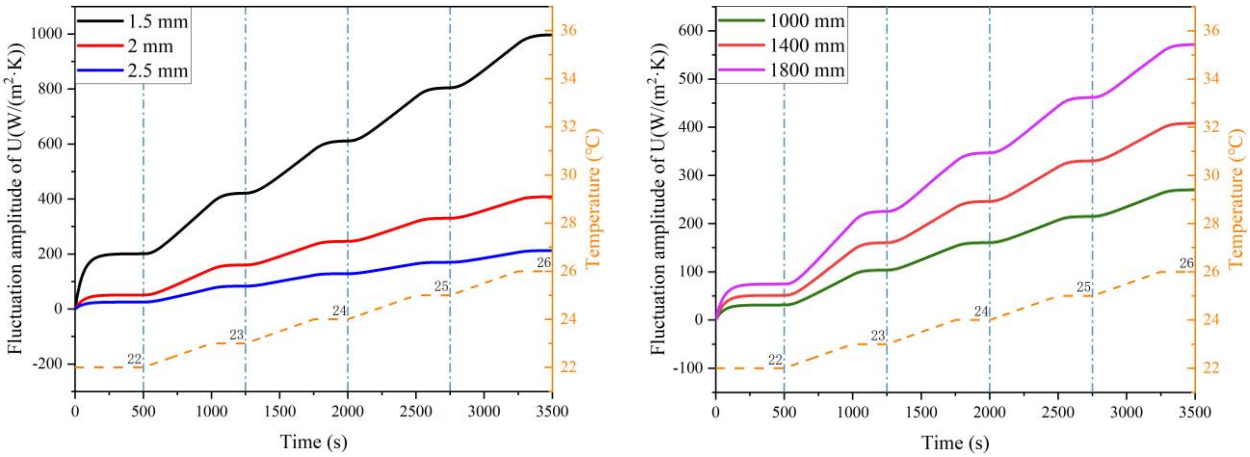
3.2.2 Dynamic Response of the Pressure Drop

Figure 8 illustrates the transient response of pressure-drop under fluctuating heat sink temperatures. Compared with the heat transfer coefficient, the pressure drop exhibits stronger sensitivity to geometric variations, particularly for small-diameter and long-channel configurations.

Among the investigated channel diameters, the 1.5 mm channel exhibits the largest pressure-drop fluctuation amplitude and the highest absolute pressure-drop level. By contrast, the 2.5 mm channel shows significantly weaker fluctuations and improved hydraulic stability. This behavior originates from the strong dependence of flow resistance on hydraulic diameter and flow velocity. Under the same mass flow condition, smaller channels produce higher velocities and larger frictional losses, making the pressure drop more sensitive to variations in fluid properties caused by heat sink temperature fluctuations.

The influence of channel length accumulates gradually. Pressure-drop fluctuations remain relatively moderate in the short channel (1000 mm), whereas the fluctuation amplitude increases substantially in the long channel (1800 mm). Since frictional losses accumulate continuously along the flow direction, local fluctuations caused by changes in physical properties are gradually amplified in longer channels, resulting in more pronounced fluctuations in the outlet pressure drop.

Combining the results of Figure 7 and Figure 8 indicates that the thermal-hydraulic responses of the recuperator are influenced differently by geometric parameters under transient operating conditions. Small-diameter and long-channel configurations exhibit stronger fluctuations in both heat transfer coefficient and pressure drop because S-CO_2 property variations become more sensitive under these conditions. By contrast, larger diameters and shorter channels produce relatively smoother thermal-hydraulic responses.These results suggest that the parameter observed under steady-state conditions may change under transient heat sink temperature disturbances. Therefore, system-level transient analysis can provide additional information for evaluating the thermal-hydraulic response characteristics of recuperators under off-design operating conditions.

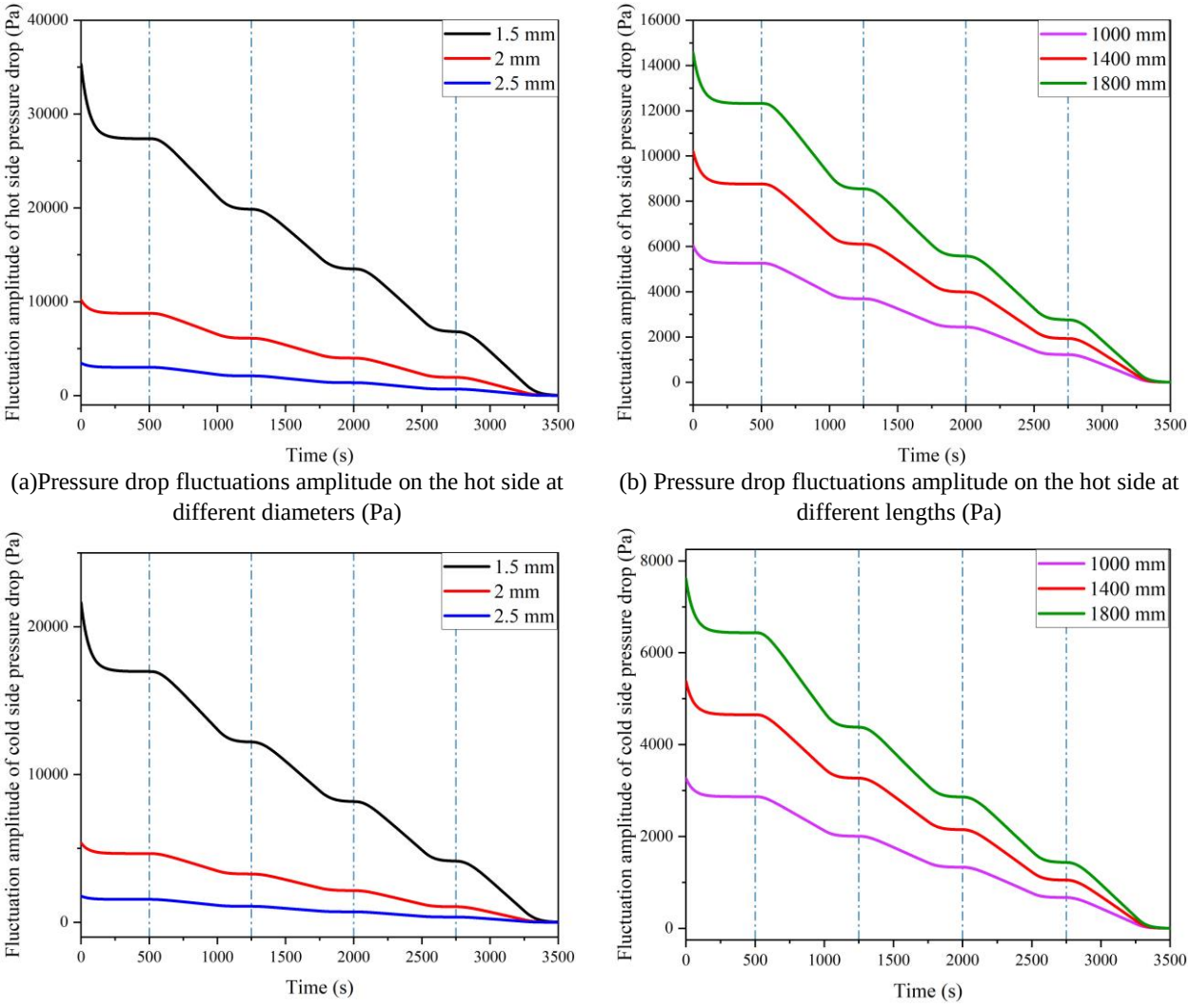


(a) Total heat transfer coefficient fluctuation amplitude at different diameters (W/(m^2·K))

(b) Total heat transfer coefficient fluctuation amplitude at different lengths (W/(m^2·K))

Figure 7. Effect of heat sink temperature fluctuations on the heat transfer coefficient.

DOI
10.3384/ecp223253



(a)Pressure drop fluctuations amplitude on the hot side at different diameters (Pa)

(b) Pressure drop fluctuations amplitude on the hot side at different lengths (Pa)

(c)Pressure drop fluctuations amplitude on the cold side at different diameters (Pa)

(d) Pressure drop fluctuations amplitude on the cold side at different lengths (Pa)

Figure 8. Effect of heat sink temperature fluctuations on pressure drop.

4 Conclusions

This study developed a Modelica-based system-level dynamic simulation framework to investigate the thermal-hydraulic characteristics of a recuperator in an S-CO_2 Brayton cycle under design reference and transient off-design conditions. The main conclusions are as follows.

(1) A one-dimensional dynamic PCHE model was developed and validated using benchmark data. The maximum relative error was controlled within 0.87%, indicating that the model can capture the main thermal-hydraulic characteristics of the recuperator and is suitable for system-level transient analysis.

(2) Under design conditions, reducing the channel diameter enhances heat transfer performance but significantly increases pressure drop, whereas increasing the channel length improves heat exchange while introducing additional frictional losses. The thermal-hydraulic performance of the recuperator is therefore governed by the competing effects of heat transfer enhancement and hydraulic resistance.

(3) Under heat sink temperature fluctuations, small-diameter and long-channel configurations exhibit stronger thermal-hydraulic response variations than under steady-state conditions. These results indicate that the parameter observed under steady-state operation may change under transient off-design conditions, particularly near the critical region of S-CO_2.

(4) The proposed Modelica-based framework provides an efficient approach for preliminary parametric assessment and system-level transient thermal-hydraulic analysis of recuperators. However, detailed engineering design still requires further CFD investigations and experimental validation.

Dynamic simulation results indicate that an excessive pursuit of a compact geometric structure significantly amplifies the thermal-hydraulic fluctuations of the

recuperator under off-design operating disturbances. Future research can further couple multi-objective optimization algorithms with system-level simulation frameworks to establish recuperator selection criteria incorporating dynamic constraints, thereby achieving an optimal balance between steady-state thermal efficiency and transient sensitivity in Brayton cycle design.

Acknowledgements

This work is supported by National Natural Science Foundation of China (Grant No. 12305179).

References

Carstens, N. A., Hejzlar, P., & Driscoll, M. J. (2006). Control system strategies and dynamic response for supercritical CO_2 power conversion cycles. Center for Advanced Nuclear Energy Systems.

Dostal, V., Hejzlar, P., & Driscoll, M. J. (2006). The supercritical carbon dioxide power cycle: comparison to other advanced power cycles. Nuclear technology, 154(3), 283-301

Kalyanmoy, D. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation, 6(2), 182-197.

Ding, J., Liu, Q., Ke, J., Deng, M., Yu, G., & Liang, Y. (2024). Development of a hybrid CFD-ANN method with multi-objective optimization for airfoil-finned PCHE used in Gen-IV nuclear systems. Progress in Nuclear Energy, 175, 105346.

Gkountas, A. A., Stamatelos, A. M., & Kalfas, A. I. (2017). Recuperators investigation for high temperature supercritical carbon dioxide power generation cycles. Applied Thermal Engineering, 125, 1094-1102.

Gnielinski, V. (1976). New equations for heat and mass transfer in turbulent pipe and channel flow. International chemical engineering, 16(2), 359-367.

Jin, F., Chen, D., Hu, L., Huang, Y., & Bu, S. (2022). Optimization of zigzag parameters in printed circuit heat exchanger for supercritical CO_2 Brayton cycle based on multi-objective genetic algorithm. Energy Conversion and Management, 270, 116243

Johnston, T., Levy, W., & Rumbold, S. (2001). Application of printed circuit heat exchanger technology within heterogeneous catalytic reactors. American Institute of Chemical Engineers.

Kuo, G. C., Xie, J. Y., & Chueh, C. C. (2024). Numerical thermal–hydraulic analysis and multiobjective design optimization of a printed circuit heat exchanger with airfoil overlap fin channels. Engineering Reports, 6(2), e12719.

Liu Y F, Shen T W, Sun W, et al. Heat transfer performance of a CO_2 gas cooler with internal threaded tube[J]. Chemical Engineering, 2022, 50(03): 20-24+46.

Mignacca, B., Locatelli, G., & Sainati, T. (2020). Deeds not words: Barriers and remedies for Small Modular nuclear Reactors. Energy, 206, 118137.

Ming, Y., Liu, K., Zhao, F., Fang, H., Tan, S. & Tian, R. (2022). Dynamic modeling and validation of the 5 MW small modular supercritical CO_2 Brayton-Cycle reactor system. Energy Conversion and Management, 253, 115184.

Ming, Y., Shen, H., Liu, K., Hong, J., Zhao, F., & Tan, S. (2024). Dynamic analysis of the PCHE in 5MWth small modular SCO_2 Brayton-Cycle reactor system. Nuclear Engineering and Design, 427, 113443.

Ngo, T. L., Kato, Y., Nikitin, K., & Ishizuka, T. (2007). Heat transfer and pressure drop correlations of microchannel heat exchangers with S-shaped and zigzag fins for carbon dioxide cycles. Experimental Thermal and Fluid Science, 32(2), 560-570.

Pioro, I., Duffey, R. B., Kirillov, P. L., & Dort-Goltz, N. (2020). Current status of reactors deployment and small modular reactors development in the world. Journal of Nuclear Engineering and Radiation Science, 6(4), 044001.

Sun, T., Zhang, J., & Wei, W. (2025). Analysis of the heat transfer characteristics of supercritical CO_2 microchannels considering heat transfer with the surroundings. International Journal of Heat and Fluid Flow, 116, 109918.

Tang, S., (2019) Numerical Simulation of Supercritical Fluid Flow and Heat Transfer Characteristics In a Straight PCHE Channel [D] Harbin Institute of Technology, Harbin, China.

Yang, Y., Li, H., Yao, M., Zhang, Y., Zhang, C., Zhang, L., & Wu, S. (2020). Optimizing the size of a printed circuit heat exchanger by multi-objective genetic algorithm. Applied Thermal Engineering, 167, 114811.

Zhang, Y., Peng, M., Xia, G., & Lv, L. (2020). Transient thermal–hydraulic analysis of SCO_2-cooled reactor system with modified RELAP5. Annals of Nuclear Energy, 147, 107681.

Modelica-Based Dynamic Analysis of an HTGR Combined Cycle System under Negative Reactivity Disturbance: Comparison of Constant-Pressure and Sliding-Pressure Operation

Haojie Qu[1], Xinhe Qu[2], Ming Ding[1]

[1]Heilongjiang Provincial Key Laboratory of Nuclear Power System and Equipment, College of Nuclear Science and Technology, Harbin Engineering University, Harbin 150001, Heilongjiang, China; heuqhj@hrbeu.edu.cn

[2]Institute of Nuclear and New Energy Technology, Tsinghua University, Collaborative Innovation Center of Advanced Nuclear Energy Technology, Key Laboratory of Advanced Reactor Engineering and Safety, Ministry of Education, Beijing 100084, China

Abstract

Based on cascade energy utilization, a high-temperature gas-cooled reactor (HTGR) combined cycle offers an efficient energy conversion scheme, but its complex system configuration, strong parameter coupling, and dynamic response mechanisms require systematic analysis. In this paper, a dynamic model of an HTGR combined cycle system is developed in Modelica and validated by comparison with reference results. A negative reactivity disturbance of -300 pcm is then introduced to investigate the system transient response, with further comparison between constant-pressure and sliding-pressure operation. The results show that the steam generator heat-transfer rate decreases by 10.8% under sliding-pressure operation, less than the 15.1% decrease under constant-pressure operation, indicating better heat-transfer matching. However, steam turbine power decreases by 15.8%, compared with 3.8% under constant-pressure operation. Therefore, constant-pressure operation is more suitable for maintaining main steam pressure and steam turbine power, whereas sliding-pressure operation is more appropriate for partial-load heat-transfer matching and load following.

Keywords: Modelica; HTGR; combined cycle; dynamic simulation; negative reactivity disturbance; constant-pressure operation; sliding-pressure operation

1 Introduction

High-temperature gas-cooled reactors are Generation IV advanced nuclear energy systems with high outlet temperature, inherent safety characteristics, and modular features. Combining a high-temperature gas-cooled reactor with a combined cycle can improve the overall energy utilization efficiency of the system through cascade energy utilization. Correspondingly, compared with a single steam cycle or a single Brayton cycle, the coupling among the reactor, helium Brayton cycle, and steam Rankine cycle in a combined cycle is also more complex. Foreign studies on combined cycles are represented by Professor McDonald, who proposed a nuclear combined cycle scheme with an overall efficiency of 51.5%, providing a reference for the power conversion arrangement of this type of system (McDonald 2014). In addition, Qu Xinhe et al. from Tsinghua University studied cycle optimization and control strategies for high-temperature gas-cooled reactor combined cycles (Qu, Yang, and Wang 2019a; Qu, Yang, and Wang 2019b; Qu, Yang, and Wang 2020); Wang Jie from Harbin Engineering University also carried out comparative studies on different arrangements of combined cycles (Wang, Ding, and Yang 2016).

Reactivity regulation is an important method for changing reactor power. For high-temperature gas-cooled reactor systems, reactivity feedback and core thermal inertia have important effects on the power change process (Takamatsu et al. 2014; Wu et al. 2023). When a reactivity disturbance acts on the reactor, the influence chain is transmitted from the helium side to the steam side. Therefore, it is necessary to dynamically analyze the reactivity regulation process at the system level. In addition to heat-source-side disturbances, the operation mode of the Rankine cycle also affects the dynamic response characteristics of the combined cycle. Under constant-pressure operation, the main steam pressure is maintained near the set value, and the system mainly adapts to heat-source-side changes through changes in steam flow rate and steam turbine power. Under sliding-pressure operation, the steam pressure changes with the system power level, and the steam-side parameters can adapt more flexibly to heat-source-side disturbances. Existing studies have used sliding-pressure

regulation in off-design analysis of high-temperature gas-cooled reactor combined cycles to improve the coordinated operation between the topping cycle and the bottoming cycle under partial-load conditions (Qu, Yang, and Wang 2019b).

For complex thermal-fluid systems such as high-temperature gas-cooled reactor combined cycles, which consist of reactors, rotating machinery, heat exchangers, pipelines, and other equipment, the Modelica language is naturally suitable because it is equation-based, object-oriented, and supports unified multi-domain modeling. In this paper, a dynamic model of a high-temperature gas-cooled reactor combined cycle system is established based on Modelica. On the basis of steady-state validation, simulations under negative reactivity insertion are carried out. The dynamic response characteristics of the main operating parameters are analyzed, and the differences in system response characteristics under constant-pressure operation and sliding-pressure operation are compared.

2 System Description and Modelica Model

2.1 System Configuration

The research object of this paper is a high-temperature gas-cooled reactor combined cycle system, which is composed of a coupled topping helium Brayton cycle and bottoming steam Rankine cycle. The main equipment of the topping cycle includes the reactor, gas turbine, compressor, generator, and connecting pipelines. The bottoming cycle consists of the steam generator, steam turbine, condenser, feedwater pump, deaerator, regenerative heater, and related pipelines. The topping and bottoming cycles are coupled through the steam generator. During system operation, the heat generated by the reactor is carried out by helium. Part of the heat is converted into mechanical work through the gas turbine, and the remaining part is transferred to the bottoming cycle through the steam generator to generate steam and drive the steam turbine to output work.

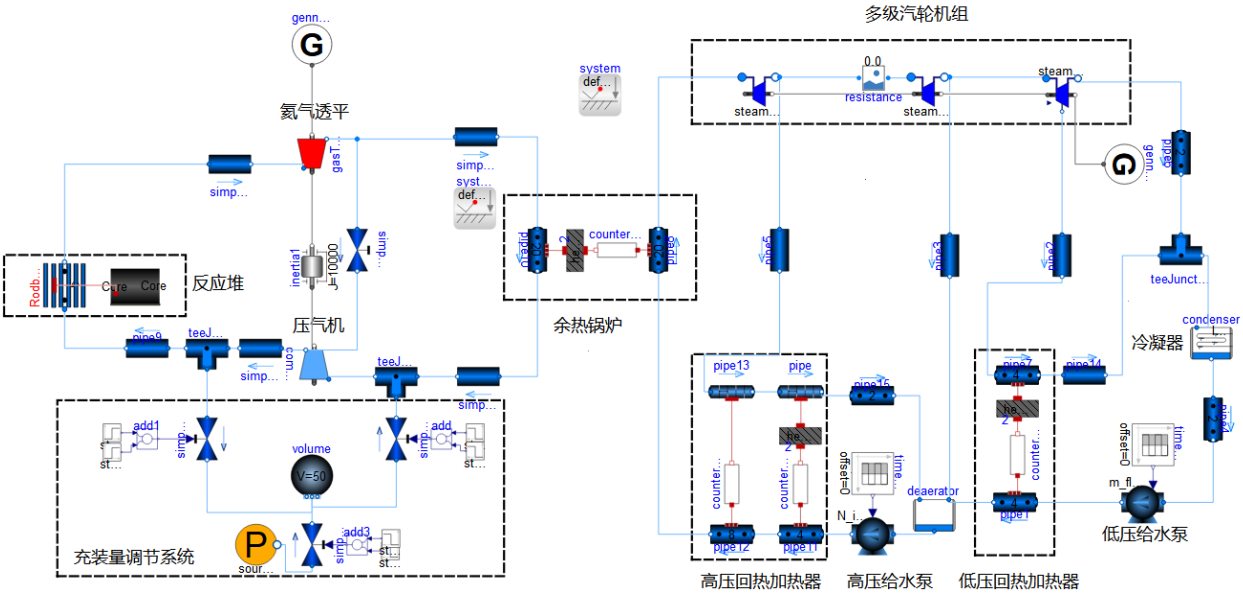


Figure 1. Configuration of the HTGR combined cycle system

2.2 Modelica Modeling Approach

Based on the Modelica language, this paper adopts a modular modeling method to build a dynamic model library for the high-temperature gas-cooled reactor combined cycle system. The library includes modules such as reactor model, gas turbine model, compressor model, heat exchanger model, generator model, gas storage tank model, valve model, and pipeline model. By dragging the relevant models, setting parameters, and connecting fluid interfaces, thermal interfaces, and mechanical rotational interfaces, a dynamic analysis model of the high-temperature gas-cooled reactor combined cycle is developed. The model can describe the coupling among the reactor, helium cycle, water-steam cycle, and shaft-system power transmission.

All equipment in the system uses mass conservation, momentum conservation, and energy conservation as the basic governing equations. The reactor model uses point reactor neutron kinetics equations considering the effects of the fuel temperature coefficient, moderator temperature coefficient, and other factors. The delayed-neutron-related parameters refer to HTR-PM and related heat-transfer equations, and are used to describe the dynamic relationship among core power, fuel temperature, coolant temperature, and reactivity feedback. The helium turbine includes the Flügel formula, and the expansion ratio is calculated through the pressure ratio and loop resistance. The compressor includes characteristic curves to calculate pressure ratio and efficiency. The coaxial module is balanced through the rotor dynamics equation. The steam turbine describes equipment characteristics through

DOI
10.3384/ecp223261

Stodola's law. The gas storage tank and valve models are used to realize regulation of the helium inventory. By controlling the valve opening time and opening degree, the total helium inventory in the system is changed.

3 Model Validation and Simulation Cases

3.1 Rated Steady-State Validation

In this paper, the calculated values under rated conditions are compared with the design values from the quasi-steady-state program, and the results are listed in Table 1. It can be seen from the table that, except for certain deviations in the primary-side outlet temperature of the steam generator and compressor-related temperatures, the errors of the other parameters are small. The errors are mainly caused by the relatively low actual heat transfer rate under the normal steam generator size. The results prove that the model can reflect the thermal state of the main equipment under rated conditions and can be used as the basis for subsequent dynamic analysis.

Table 1. Rated steady-state validation of the HTGR combined cycle model.

Component name	Parameter name	Unit	Design value	Program value	Relative error
Helium turbine	Mass flow rate	kg/s	116.67	116.68	0.009%
	Inlet temperature	℃	850	853.9	0.459%
	Outlet temperature	℃	632.86	644.04	1.767%
	Inlet pressure	MPa	7	6.95	-0.714%
	Outlet pressure	MPa	3.85	3.83	-0.519%
Steam generator primary side	Mass flow rate	kg/s	116.67	116.68	0.009%
	Inlet temperature	℃	632.86	644.04	1.767%
	Outlet temperature	℃	254.9	260.89	2.350%
	Inlet pressure	MPa	3.85	3.83	-0.519%
	Outlet pressure	MPa	3.55	3.52	-0.845%
Compressor	Mass flow rate	kg/s	116.67	116.68	0.009%
	Inlet temperature	℃	254.9	260.89	2.350%
	Outlet temperature	℃	453.53	459.3	1.272%
	Inlet pressure	MPa	3.55	3.52	-0.845%
	Outlet pressure	MPa	7.1	7.05	-0.704%
Reactor	Mass flow rate	kg/s	116.67	116.68	0.009%
	Inlet temperature	℃	453.53	459.3	1.272%
	Outlet temperature	℃	850	853.9	0.459%
	Inlet pressure	MPa	7.1	7.05	-0.704%
	Outlet pressure	MPa	7	6.95	-0.714%
Steam generator secondary side	Mass flow rate	kg/s	88.87	90.15	1.440%
	Inlet pressure	MPa	15.9	16.3	2.516%
	Outlet pressure	MPa	13.9	14.3	2.878%
	Inlet enthalpy	kJ/kg	940.66	924.26	-1.743%
	Outlet enthalpy	kJ/kg	3517.47	3499.47	-0.512%

3.2 Simulation Cases

Two dynamic cases are set here: negative reactivity input under constant-pressure and sliding-pressure conditions. The related information and case comparison are listed in the following table.

- Case 1: negative reactivity disturbance under constant-pressure Rankine-cycle operation;
- Case 2: negative reactivity disturbance under sliding-pressure Rankine-cycle operation.

Table 2. Negative reactivity disturbance simulation cases.

Case	Disturbance input	Start time	Duration
case1	-300pcm	2000s	1s
case2	-300pcm	2000s	1s

4 Dynamic Response under Negative Reactivity Disturbance

For a highly complex and strongly coupled system such as the high-temperature gas-cooled reactor combined cycle, under different boundary conditions, namely the constant-pressure operation and sliding-pressure operation discussed in this paper, the change direction and change amplitude of the related parameters of each component may be inconsistent under the same disturbance. Therefore, this chapter first discusses the parts with consistent change directions but different degrees, namely the transmission mechanism and physical logic of reactivity regulation in the topping Brayton cycle.

4.1 Reactivity Disturbance and Core Response

Sliding-pressure operation is taken as an example for analysis here. The external reactivity starts to be input at 2000s and changes from 0 to -300pcm within 1s. By reducing the neutron multiplication capability of the core, the reactor power decreases rapidly. At the initial stage of the disturbance, the power decreases significantly, with the minimum value dropping to 92.7MW. Then, under the effect of temperature feedback, it recovers, fluctuates, and finally tends to stabilize, decreasing from 239MW to 213MW, with a stable decrease of 10.9%. The core temperature response is smoother than the power change. The maximum fuel temperature decreases from 879℃ to 823℃, the average fuel temperature decreases from 720℃ to 670℃, the reactor outlet helium temperature decreases from 855℃ to 800℃, and the inlet helium temperature decreases from 537 ℃ to 494 ℃. The maximum temperature decrease is 8.0%, which is smaller than the power decrease and has a certain lag. This is mainly because the core structure itself has a large heat capacity, which buffers the temperature change.

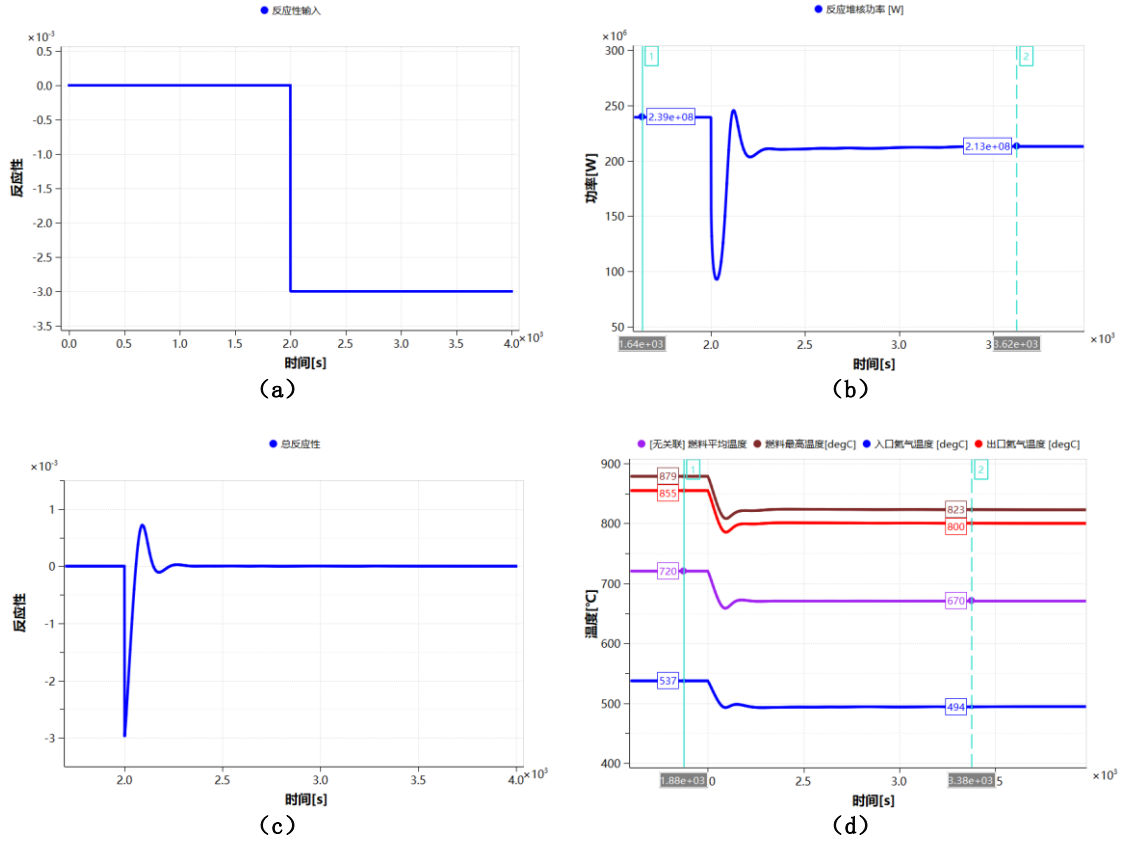


Figure 2. Reactor power, reactivity feedback, and core temperature response under negative reactivity disturbance.

The total reactivity curve reflects this process. After the negative reactivity is introduced, the total reactivity decreases rapidly, and the power decreases sharply accordingly. The fuel and coolant temperatures decrease correspondingly, and positive reactivity is introduced under temperature feedback, causing the total reactivity to recover. Because the temperature change has a lag, the total reactivity shows the characteristic of tending to equilibrium after fluctuation, and the reactor power also reaches a new equilibrium value. It can be seen that the power response under negative reactivity disturbance is not a simple step decrease, but the result of the coupling between neutron kinetics and thermal feedback. This also indicates that the reactor has a certain self-stabilizing and self-regulating capability.

4.2 Brayton-Cycle Response

After the core power decreases, the heat removed from the reactor by the working fluid decreases, the helium temperature decreases, and both the high-pressure-side pressure and low-pressure-side pressure decrease accordingly. The compressor outlet pressure and turbine inlet pressure decrease by about 9%, while the turbine outlet pressure and compressor inlet pressure decrease by about 6.8%. This reflects the integrity of the closed gas cycle, that is, the change in reactor outlet temperature is transmitted to the whole pressure field through turbine expansion, compressor compression, and pipeline resistance.

The change in mass flow rate further reflects the migration of the cycle operating point. After the negative reactivity disturbance, the helium mass flow rate decreases from 116 kg/s to 107 kg/s, with a decrease of

DOI
10.3384/ecp223261

7.8%. This is because the decrease in core outlet helium temperature weakens the turbine work capability, and the shaft-system power balance is broken. Then the compressor, turbine, and loop resistance are rematched, and the flow rate stabilizes at a new equilibrium value. At the same time, the flow rate does not simply change synchronously with power. Affected by the flow inertia of the closed loop and the rotational inertia of the rotating machinery, the change trend shows a process of first decreasing rapidly and then recovering slowly.

The change in rotating machinery power is the direct result of the adjustment of thermal-hydraulic parameters. The helium turbine power decreases from 126 MW to 107MW, and the compressor power decreases from 119M W to 101MW, with both decreases being about 15%. The decrease in turbine power comes from the decrease in inlet temperature, pressure, and mass flow rate. The decrease in compressor power comes from the decrease in flow rate and pressure ratio. The synchronous decrease of the two indicates that the negative reactivity disturbance not only weakens the output-end power, but also reduces the load on the compression end. The final change in Brayton-cycle net power depends on the difference between turbine output and compressor power consumption, rather than the power change of a single piece of equipment.

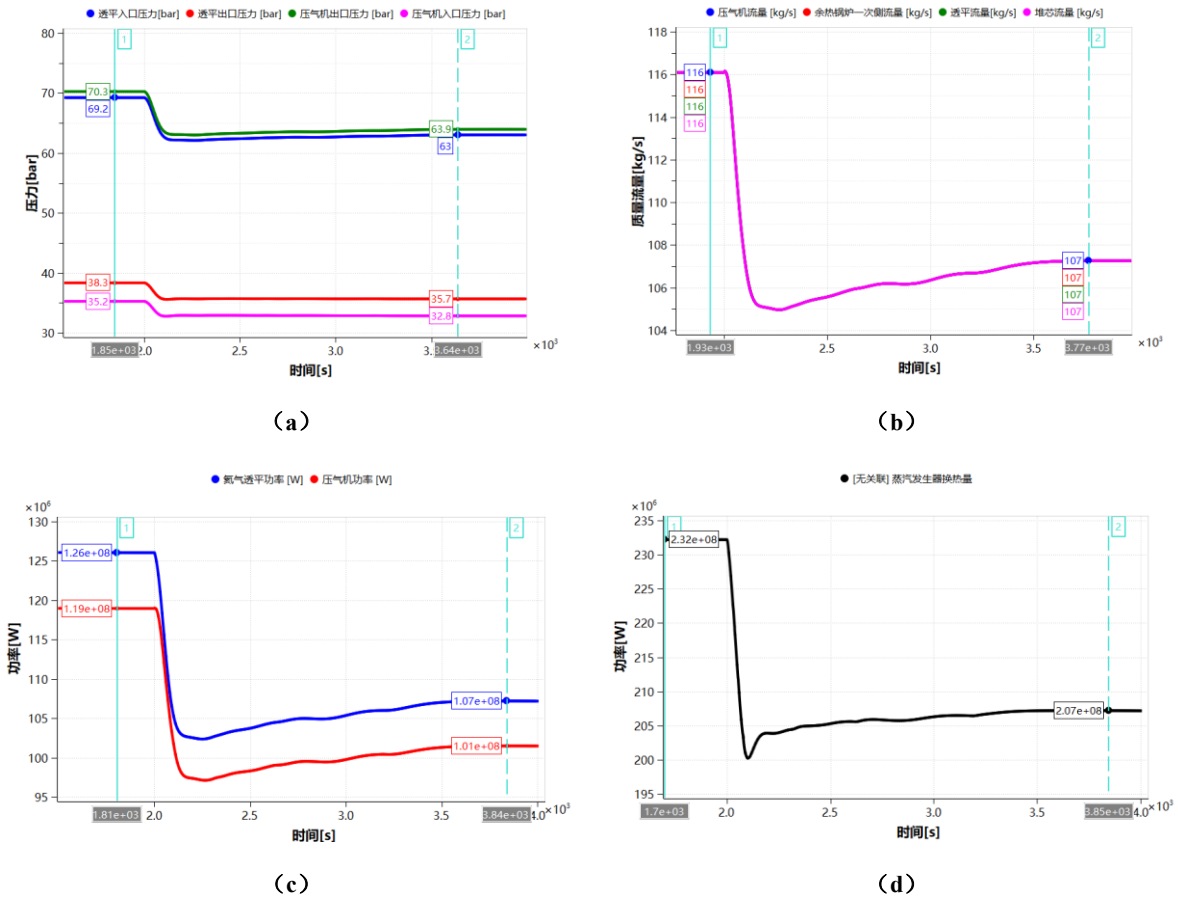


Figure 3. Brayton-cycle power response during the negative reactivity disturbance

4.3 Heat Transfer to Rankine Cycle

After the negative reactivity disturbance is transmitted through the reactor and the closed Brayton cycle, it is finally reflected in the change in heat input to the steam generator. The steam generator heat transfer rate decreases from 232MW to 207MW, with a decrease of 10.8%, which is close to the decrease in reactor power. This indicates that the heat-source-side power decrease is finally transmitted to the secondary loop through the reduction of the steam generator heat transfer rate. However, the steam generator is not a simple one-way heat-transfer component. For the combined cycle system, on the one hand, it transfers heat from the helium side to the water-steam side and is the channel through which the disturbance enters the Rankine cycle from the Brayton cycle. On the other hand, the pressure, flow rate, and temperature on the steam side change the heat-transfer temperature difference, phase-change position, and heat absorption capability of the superheating section, thereby reversely affecting the cooling effect on the helium side. Therefore, the steam generator is both the disturbance transmission interface and the coupling boundary for mutual feedback between the topping and bottoming cycles.

From the dynamic response, the steam generator heat transfer rate first decreases rapidly after the disturbance and then slowly recovers until it stabilizes. This process is not caused by a single factor, but by the combined effects of core power decrease, helium temperature

decrease, flow-rate reduction, and heat storage in the heat exchanger. The core power determines the upper limit of heat, while the heat storage of the heat exchanger wall gives the change in heat transfer rate a certain lag and buffering effect. Therefore, the system response under negative reactivity disturbance is strongly coupled among multiple parameters. Negative reactivity changes the core power, the core thermal state changes the operating point of the helium cycle, the helium-side state changes the steam generator heat transfer rate, and the steam-side operation mode reversely affects the helium-side cooling boundary through the steam generator. Based on this strong coupling characteristic, the next chapter further compares the influence of constant-pressure and sliding-pressure operation modes on the system response.

5 Comparison between Constant-Pressure and Sliding-Pressure Operation

5.1 Comparison and Influence Analysis of Steam-Side Pressure, Flow Rate,and Temperature

Under constant-pressure operation, the main steam pressure boundary is constrained, and the inlet and outlet pressures of the steam generator basically remain at their initial values. Under sliding-pressure operation, the steam-side pressure decreases synchronously with load reduction. The inlet pressure decreases by 8.0%, and the outlet pressure decreases by 10.5%. The pressure difference determines different heat-transfer responses inside the steam generator.

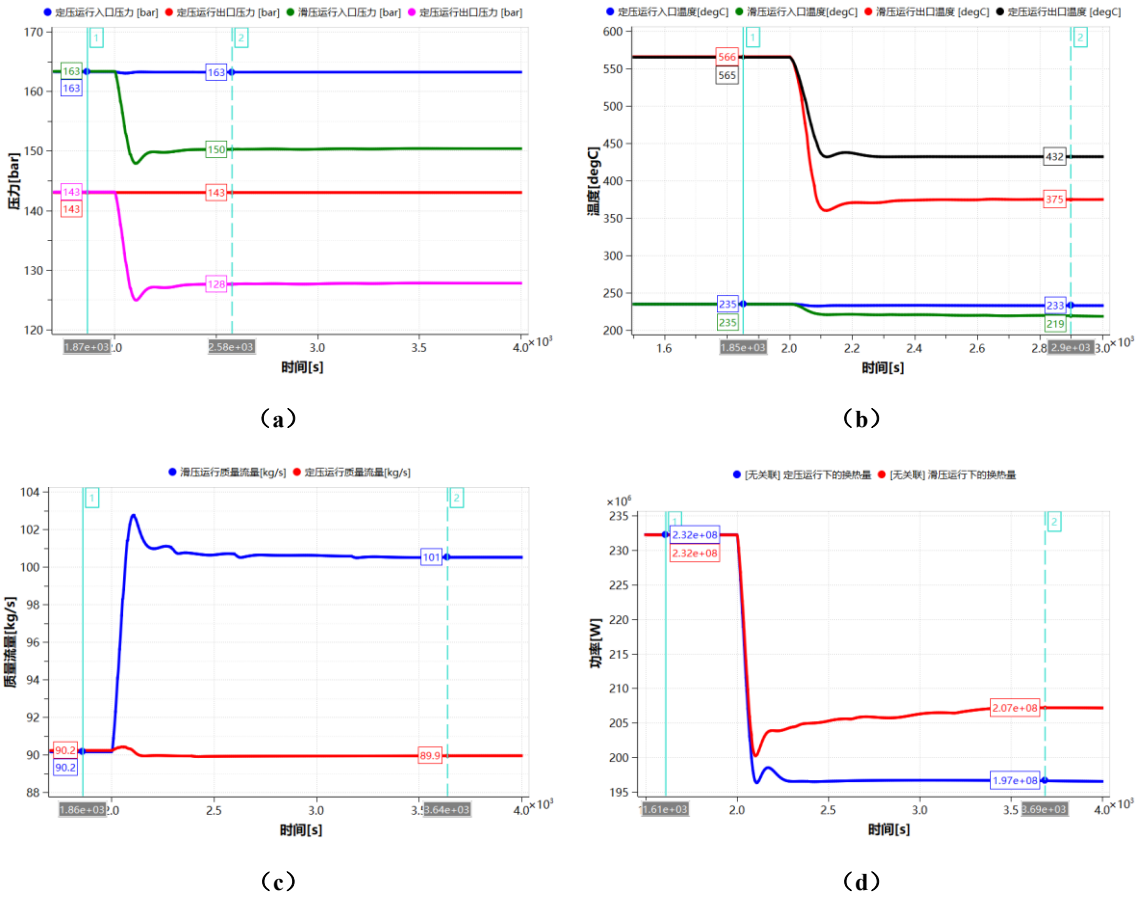


Figure 4. Comparison of main steam pressure and temperature under constant-pressure and sliding-pressure operation.

When facing the reduction of helium-side input heat caused by the negative reactivity input disturbance, under the constant-pressure mode, the steam pressure remains at a relatively high level, the saturation temperature is relatively high, and the heat-transfer temperature difference on the two sides of the steam generator is limited, especially the superheating capability in the rear section. Therefore, under constant-pressure operation, the steam generator heat transfer rate decreases more, reaching 15.1%. Under sliding-pressure operation, the decrease in main steam pressure causes the saturation temperature to decrease synchronously, the heat-transfer temperature difference decreases less, and the steam generator can still maintain a relatively strong heat-transfer capability under low heat input. The decrease in heat transfer rate is relatively small, at 10.8%. The comparison shows that sliding-pressure operation

DOI
10.3384/ecp223261

improves heat-transfer matching during load reduction by changing the steam-side pressure boundary.

Under constant-pressure operation, the steam flow rate is basically unchanged, and the stable value decreases by only 0.3%. Under sliding-pressure operation, the steam flow rate increases by 12.0%. At the same time, the steam generator outlet temperature decreases by 23.5% under constant-pressure operation, while the outlet temperature decreases by 33.7% under sliding-pressure operation. By comparison, it can be seen that although sliding-pressure operation improves the heat-transfer adaptability under low load, because the steam flow rate increases, the heat obtained by unit mass of steam decreases, and the steam quality decreases significantly. Constant-pressure operation ensures steam quality by maintaining the pressure level.

It can be seen that the two operation modes reflect different energy distribution logics. Constant-pressure operation emphasizes steam pressure stability and steam quality, but after the heat source decreases, the heat transfer rate decays relatively significantly. Sliding-pressure operation emphasizes the adjustment of the heat-transfer boundary with load, and the steam generator heat transfer rate is maintained better, but the main steam pressure and outlet temperature decrease more. For the combined cycle system, this difference further changes the available energy of steam at the steam turbine inlet and affects the helium-side cooling process through the steam generator.

5.2 Steam Turbine Power Response

After the negative reactivity disturbance, the steam turbine power decreases under both operation modes, but the decrease amplitudes are clearly different. Under constant-pressure operation, the steam turbine power decreases by 3.8%. Under sliding-pressure operation, the steam turbine power decreases by 15.8%. It can be seen that sliding-pressure operation makes the Rankine-cycle power more consistent with the heat-source-side power reduction process, while constant-pressure operation has a stronger maintaining effect on steam turbine output power.

From the perspective of energy conversion, steam turbine power depends on the steam mass flow rate and the work capability of unit mass of steam. Under sliding-pressure operation, the steam flow rate increases, but the main steam pressure and outlet temperature decrease significantly, resulting in a decrease in the available energy of inlet steam and a weakening of the expansion work capability of unit mass of steam. Therefore, the increase in flow rate is not converted into an increase in power. Instead, the steam turbine power decreases significantly due to the decrease in enthalpy drop. Under constant-pressure operation, the main steam pressure remains stable, the steam flow rate changes little, and the outlet temperature is also higher than that under sliding-pressure operation. Therefore, the work capability of unit mass of steam is maintained better, and the decrease in steam turbine power is smaller.

Overall, sliding-pressure operation is more suitable for conditions in which the load decreases synchronously with the heat source. It can make the secondary-loop power response maintain strong consistency with the reactor power reduction process. Constant-pressure operation is more suitable for conditions requiring stable main steam pressure and maintained steam turbine power, but its heat-transfer matching capability is relatively weaker than that of sliding-pressure operation.

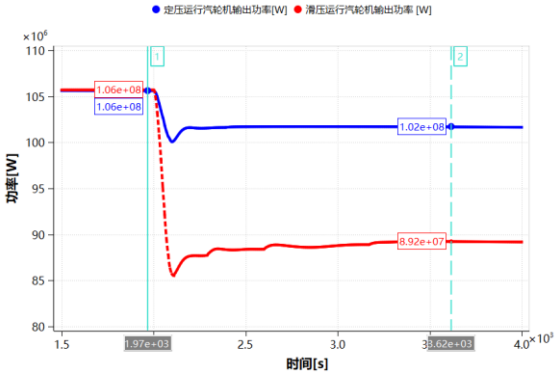


Figure 5. Steam-side power and heat-transfer matching under the two operation modes.

5.3 Evaluation of Coupling Characteristics of the Two Operation Modes

Constant-pressure operation and sliding-pressure operation are not simply superior or inferior to each other, but reflect different regulation objectives. Constant-pressure operation focuses more on maintaining the stability of main steam pressure and steam turbine power. Sliding-pressure operation improves heat-transfer matching under partial-load conditions by reducing steam pressure, but it causes larger changes in steam temperature and steam turbine power. Therefore, the selection of the operation mode should be comprehensively determined by considering the load regulation target, steam generator outlet superheat degree, steam turbine power demand, and system control margin.

Table 3. Comparison of constant-pressure and sliding-pressure operation characteristics.

Item	Constant-pressure operation	Sliding-pressure operation
Main steam pressure	The main steam pressure basically remains unchanged, and the pressure stability is relatively good.	The main steam pressure decreases as the load decreases.

Steam turbine power	The steam turbine power decreases by 3.8%. The higher and stable main steam pressure is beneficial for maintaining the work capability of the steam turbine.	The steam turbine power decreases by 15.8%. The decreases in main steam pressure and outlet temperature reduce the available enthalpy drop, leading to a more obvious power decrease.
Heat-transfer matching	The steam generator heat transfer rate decreases by 15.1%. Under constant-pressure conditions, after the heat-source power decreases, the effective heat-transfer temperature difference is limited.	The steam generator heat transfer rate decreases by 10.8%. Sliding-pressure conditions improve the heat-transfer temperature-difference distribution under partial load, and the heat-transfer matching capability is better.
Dynamic stability	The fluctuations of steam-side pressure and steam turbine power are smaller, and the parameter stability is better, but the adaptability of the steam generator to heat-source reduction is relatively limited.	The variation amplitudes of steam pressure, outlet temperature, and steam turbine power are larger, and parameter fluctuations are more obvious, but the partial-load adaptability is stronger.
Control complexity	The control target is relatively clear, mainly maintaining the stability of main steam pressure, and the operation strategy is relatively simple.	It is necessary to coordinate changes in pressure, flow rate, and temperature. The operation mode is more flexible, but the control complexity is higher.
Suitable operating condition	It is suitable for conditions requiring stable main steam pressure and small changes in steam turbine output, such as small load changes or operating scenarios with strong pressure constraints.	It is suitable for partial-load operation or coordinated load reduction processes, and for conditions that emphasize heat-transfer matching and load-following capability.

6 Conclusions

1. A dynamic model of the HTGR combined cycle system was developed in Modelica. The model includes the main components, such as the reactor, helium Brayton cycle, steam generator, and steam Rankine cycle. The rated steady-state validation shows that the model can reasonably reflect the main thermodynamic parameters of the system and can be used for subsequent dynamic analysis.

2. Under a -300 pcm negative reactivity disturbance, the reactor power decreases rapidly and then gradually reaches a new stable state under the effect of temperature feedback. The reactor power decreases from 239 MW to 213 MW, with a decrease of 10.9%. The helium mass flow rate decreases by 7.8%, and the turbine power and compressor power both decrease by about 15%. This indicates that the disturbance is transmitted from the core to the downstream system through the closed Brayton cycle.

3. Constant-pressure operation and sliding-pressure operation show different response characteristics. Under sliding-pressure operation, the steam generator heat-transfer rate decreases by 10.8%, which is smaller than the 15.1% decrease under constant-pressure operation, indicating better heat-transfer matching. However, the steam turbine power decreases by 15.8%, clearly larger than the 3.8% decrease under constant-pressure operation. Therefore, constant-pressure operation is more suitable for maintaining main steam pressure and steam turbine power, while sliding-pressure operation is more suitable for heat-transfer matching and load following under partial-load conditions.

Acknowledgements

This work has been supported by the CNNC Leading-Innovation Research Project. The authors would also like to thank the team members for their support with model development and simulation implementation.

References

McDonald, C. F. (2014). “Power conversion system considerations for a high efficiency small modular nuclear gas turbine combined cycle power plant concept (NGTCC).” In: Applied Thermal Engineering 73.1, pp. 82–103.

Qu, X. H., X. Y. Yang, and J. Wang (2019a). “Combined cycle-coupled high-temperature and very high-temperature gas-cooled

reactors: Part I—Cycle optimization." In: Annals of Nuclear Energy 134, pp. 193–204.
Qu, X. H., X. Y. Yang, and J. Wang (2019b). "Off-design performance and power-control strategy for combined cycle coupled with high-temperature gas-cooled reactor." In: Annals of Nuclear Energy 130, pp. 338–346.
Qu, X. H., X. Y. Yang, and J. Wang (2020). "Combined cycles-coupled high-temperature and very high-temperature gas-cooled reactors: Part II—Engineering design." In: Annals of Nuclear Energy 140, 106953.
Takamatsu, K., S. Nakagawa, N. Sakaba, et al. (2014). "Spontaneous stabilization of HTGRs without reactor scram and core cooling—Safety demonstration tests using the HTTR: Loss of reactivity control and core cooling." In: Nuclear Engineering and Design 271, pp. 379–387.
Wang, J., M. Ding, and X. Yang (2016). "Performance comparison and optimization of two configurations of (Very) high temperature gas-cooled reactors combined cycles." In: Annals of Nuclear Energy 94, pp. 279–287.
Wu, S., X. Ma, J. Liu, et al. (2023). "A load following control strategy for Chinese Modular High-Temperature Gas-Cooled Reactor HTR-PM." In: Energy 263, 125459.

DOI
10.3384/ecp223261

Dynamic Modelling of a PEM Electrolyser and Metal Hydride Hydrogen Storage System

Michele Bolognese[1] Paolo Piras[1] Giulia Bonamigo[1] Matteo Testi[1] Erika Michela Dematteis[2] Paola Rizzi[2] Marcello Baricco[2]

[1]Center for Sustainable Energy, Fondazione Bruno Kessler, Italy,
{mbolognese,ppiras,gbonamigo,testi}@fbk.eu
[2]Department of Chemistry, NIS and INSTM, University of Turin, Italy,
{erikamichela.dematteis,paola.rizzi,marcello.baricco}@unito.it

Abstract

Renewable deployment requires hydrogen storage decoupling production from demand, and metal hydride systems offer a safe, low-pressure, high-density option. Within the EU REMEDHYS project, a solid-state storage system based on an intermetallic alloy produced from recycled European metals is coupled to a 2 MW proton exchange membrane electrolyser. This paper presents a modular, equation-based Modelica model of the balance of plant of both subsystems, comprising eleven control loops, measured pressure-composition isotherms and a pseudo-two-dimensional discretisation of the hydride bed. A 2 MW to 1 MW load step, a full absorption-desorption cycle, a sensitivity analysis and a coupled scenario feeding twenty cassettes are simulated. The hydrogen delivered per cycle falls from 4281 to 3292 g as the delivery pressure rises from 1 to 5 bar and doubles between 50 and 70 °C. Coupled to the electrolyser, a cassette charges in 6.6 h and stops discharging at a state of charge of 0.24, limited by the alloy equilibrium pressure.

Keywords: PEM electrolyser, metal hydride, solid-state hydrogen storage, balance of plant, dynamic simulation, Modelica

1 Introduction

The transition towards carbon-neutral energy systems places hydrogen at the centre of long-term decarbonisation strategies, serving both as an energy carrier and a medium-term storage buffer for surplus renewable electricity. However, a critical prerequisite for the widespread adoption of hydrogen in stationary and industrial applications is the availability of storage technologies that are simultaneously safe, compact, cost-effective, and compatible with renewable power profiles. The most relevant physical and chemical hydrogen storage methods can be classified into compressed hydrogen, liquid hydrogen, adsorption-based methods, liquid organic hydrogen carriers and solid-state storage using metal hydrides. Among these, compressed and cryogenic hydrogen are mature technologies widely used in several applications, but they still present challenges related to pressure levels, energy demand, safety requirements, and infrastructure complexity. High-pressure storage solutions, typically operating at 350–700 bar, carry significant safety risks, including potential vessel failure, hydrogen embrittlement, and rapid uncontrolled H_2 release. Cryogenic liquid storage introduces additional challenges, such as boil-off losses and the energy penalty associated with liquefaction (Wang et al. 2025; Usman 2022). In this context, solid-state storage in metal hydride (MH) materials has the potential to mitigate these limitations by reversibly binding hydrogen within an intermetallic crystal lattice at pressures typically below 50 bar and at near-ambient temperatures, achieving a volumetric hydrogen density that exceeds that of 700 bar compressed gas storage under considerably milder conditions. The inherent thermodynamic stability of the hydride phase significantly reduces the risk of rapid pressure release, making MH systems intrinsically safer for deployment in populated or port environments (Colbe et al. 2019). Furthermore, the exothermic nature of the absorption reaction creates the opportunity to recover low-grade waste heat from co-located electrolysers, improving overall system efficiency. These attributes make MH storage particularly attractive as a stationary intermediate buffer co-located with electrolytic hydrogen production sites, where the tight thermal and hydraulic coupling between the MH tank and the upstream proton exchange membrane electrolyser (PEMEL) requires accurate dynamic modelling to support control strategy development and safe system integration (Alleori et al. 2025; Benmehel et al. 2024; Olivier et al. 2017). The REMEDHYS project (REMEDHYS 2026) addresses these challenges through the development of a 100 kgH_2 solid-state MH storage system, coupled to a 2 MW PEMEL at the Cuxhaven port facility in Germany. $TiFe_{0.85}Mn_{0.05}$ was selected as the MH storage alloy, following previous studies that identified TiFe-based compounds with Mn addition as promising candidates for stationary hydrogen storage (Barale and Baricco 2022). In REMEDHYS, this intermetallic alloy is produced from recycled metals sourced entirely within Europe, thereby embedding circular-economy principles into the hydrogen value chain. Moreover, the alloy enables operation below 50 bar at moderate temperatures

(34–70 °C) and is compatible with the potential partial recovery of waste heat from the electrolyser cooling loop during desorption, further improving round-trip system efficiency (Barale and Baricco 2022; Dematteis and Baricco 2021). Equation-based modelling languages such as Modelica are well suited to this multi-domain problem, owing to their acausal, object-oriented formulation, which facilitates the interconnection of component-level sub-models across heterogeneous physical domains (Fritzson 2015). While recent studies have demonstrated Modelica-based BoP models for PEM stacks (Pfennig et al. 2025) and MH storage systems independently (Cavo et al. 2021), integrated models that couple both subsystems through shared thermal and hydrogen mass boundaries and that capture the dynamic interplay between PEMEL load transients and MH bed thermal management remain scarce in the open literature. This paper addresses that gap by developing and demonstrating such an integrated model within the Modelon Impact environment (Modelon AB 2026) using the Process Systems Library (PSL) (TLK Energy 2026). Simulation results are provided for a PEMEL load transition, a complete absorption-desorption cycle of a single MH cassette, a sensitivity analysis of the discharge to the delivery pressure and to the desorption temperature, and a coupled scenario in which a variable electrolyser output feeds twenty cassettes in parallel, together with their engineering implications for the REMEDHYS project in terms of system sizing, control architecture, and instrumentation requirements.

2 Methodology

The BoP model of REMEDHYS system was developed in the Modelica language and implemented in the Modelon Impact environment using the Process Systems Library developed by TLK (TLK Energy 2026). A modular, equation-based dynamic modelling approach was adopted to simulate the PEMEL and the MH storage, which are the main process units, functioning as interconnected components exchanging mass, energy and control signals. This structure is particularly suitable for the REMEDHYS system, where electrochemical hydrogen production, solid-state hydrogen storage, thermal management, and process control are strongly linked. The PEMEL model includes the electro-chemical stack, the liquid/gas separators for hydrogen and oxygen, the water supply circuit, and the downstream cooling/demisting unit. The MH storage model includes the single-cassette volume, the hydrogen inlet and outlet lines, and the heat transfer fluid circuit, required to control hydrogen absorption and desorption. The preliminary engineering layout was used to define the main process interfaces, component connections, and control loops implemented in the model. Emphasis was also placed on the implementation of preliminary control loops using proportional-integral (PI) controllers to replicate the BoP functional operation and assess the dynamic response of the system under controlled conditions. The model development and integration were guided by established technical literature on hydrogen systems, PEMEL modelling and dynamic BoP simulation, highlighting the importance of a modular approach and robust control interfaces for advanced system-level simulation (Benmehel et al. 2024; Zhang et al. 2022; Polo-Molina et al. 2025; Riera et al. 2023).

2.1 PEMEL model

The water electrolysis model describes the electrochemical behaviour of the PEM electrolyser stack by explicitly resolving the electrochemical half-reactions occurring at each electrode. At the anode, water undergoes oxidative decomposition, yielding oxygen gas, protons, and electrons (Equation 1), while at the cathode, the protons transported through the membrane recombine with the electrons from the external circuit to form hydrogen gas (Equation 2). Together, these two half-reactions represent the overall water-splitting process:

$$H_2O \rightarrow \tfrac{1}{2}O_2 + 2H^+ + 2e^- \tag{1}$$

$$2H^+ + 2e^- \rightarrow H_2 \tag{2}$$

A key modelling assumption is that if the anode channel becomes depleted of liquid water, all electrochemical reactions cease instantaneously, as water is the sole reactant. The total cell voltage (Equation 3) is decomposed into an activation component and an ohmic component, where the latter is expressed as the product of the total ohmic resistance and the current density (Equation 4):

$$U_{cell} = U_{activation} + \Delta U_{ohm} \tag{3}$$

$$\Delta U_{ohm} = R\,i \tag{4}$$

The activation voltage (Equation 5) is further expressed as the sum of the reversible voltage and the activation overpotential, where the reversible voltage (Equation 6) is itself composed of a standard term and a Nernst correction:

$$U_{activation} = U_{rev} + \Delta U_{activation} \tag{5}$$

$$U_{rev} = U_{rev0} + \Delta U_{nernst} \tag{6}$$

The standard reversible voltage (Equation 7) accounts for temperature dependence, while the Nernst correction term (Equation 8) captures the influence of the species partial pressures on the equilibrium potential (Falcão and Pinto 2020):

$$U_{rev0} = 1.2285168 - 0.000846\,(T - 298.15) \tag{7}$$

$$\Delta U_{nernst} = \frac{RT}{2F}\left[\ln\left(\frac{p_{H_2}\,p_{O_2}^{0.5}}{p_{H_2O}}\right)\right] \tag{8}$$

where R is the universal gas constant, T the operating temperature, F the Faraday constant, and p_i the partial

DOI
10.3384/ecp223271

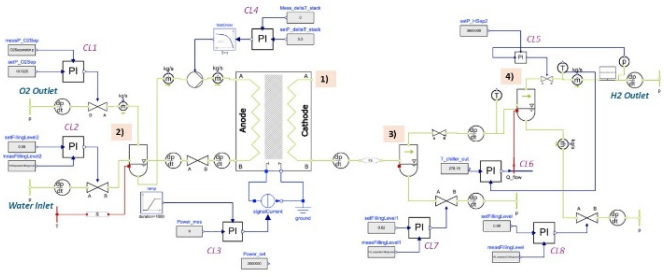


Figure 1. Scheme of the PEMEL BoP model built with the PSL library (TLK Energy 2026)

pressures of the respective species. The activation overpotential (Equation 9) is computed using a linearised (simplified) Butler–Volmer expression (Schalenbach et al. 2013):

$$\Delta U_{activation} = \frac{RT}{\alpha z F}\left(\frac{j}{j_0}\right) \tag{9}$$

This formulation relies on two electrode-specific kinetic parameters: j is the current density and j_0 is the exchange current density, which represents the intrinsic electrochemical activity of the catalyst at equilibrium. It also depends on the number of electrons exchanged, z, which is equal to 2, and the charge transfer coefficient α, which describes the asymmetry of the energy barrier for the electrochemical reaction. While both j_0 and α are physically rooted in the electrode catalytic characteristics, they are commonly treated as fitting parameters during model validation against experimental polarization curves. The model parameters and empirical correlations are based on experimental data from the literature, specifically from Schalenbach et al. (2013), ensuring accurate representation of PEM water electrolysis efficiency and gas crossover under various operating conditions. Figure 1 presents the scheme of the PEM BoP model developed for REMEDHYS project. The main components of the PEMEL BoP model are listed below:

- PEM electrolyser
- O_2 liquid/gas separator
- H_2 liquid/gas separator
- Chiller/Demister

The preliminary control loops implemented in the model are listed in Table 1, together with the tuning parameters of each PI block, namely the proportional gain k, the integral time T_i, the saturation limits of the controller output and its initialisation value. Each loop is based on a proportional-integral (PI) block, since the derivative action was found to be unnecessary for the time scales of interest and would amplify the noise introduced by the two-phase separator level signals. The eight loops cover the stack power set point, the thermal management of the anode recirculation, the liquid level and pressure of both separators, and the outlet temperature of the hydrogen chiller.

Table 1. Control loops of the PEMEL functional zone

CL	Controlled variable, actuator and PI tuning
1	**O_2 separator pressure.** Oxygen vent valve opening; set point 1.013 bar. $k = 1$; $T_i = 10$ s; $y \in [0.4, 10]$ %; $y_0 = 2$
2	**O_2 separator liquid level.** Make-up water valve opening; set point 0.55 [m] filling level. $k = 1\times10^5$; $T_i = 10$ s; $y \in [1, 10\,000]$; $y_0 = 10$
3	**Stack electrical power.** Stack current set by the power profile. $k = 10$; $T_i = 10$ s; $y \in [1, 25\,000]$ A; $y_0 = 40$
4	**Stack thermal management.** Circulation pump speed acting on the stack-to-separator temperature difference; set point ΔT = 5.5 °C. $k = 0.01$; $T_i = 10$ s; $y \in [1, 100]$ %; $y_0 = 40$
5	**H_2 delivery pressure.** Top outlet valve opening; set point 36 bar. $k = 1\times10^{-4}$; $T_i = 1$ s; $y \in [0.01, 100]$ %; $y_0 = 5$
6	**H_2 chiller outlet temperature.** Chiller heat flow rate on the knock-out drum; set point 5 °C. $k = 100$; $T_i = 10$ s; $y \in [-5\,000, 5\,000]$ W; $y_0 = 35$
7	**H_2 separator liquid level.** Bottom drain valve opening; set point 0.82 [m] filling level. $k = 1\times10^4$; $T_i = 10$ s; $y \in [10, 100]$ %; $y_0 = 35$
8	**Demister liquid level.** Condensate drain valve opening; set point 0.11 [m] filling level. $k = 500$; $T_i = 10$ s; $y \in [0, 100]$ %; $y_0 = 35$

2.2 MH storage model

The MH storage system is modelled considering a TiFe-based intermetallic alloy as the hydrogen storage medium. At the material level, pressure-composition isotherm (PCI) data from the literature are used to characterise the thermodynamic properties of the alloy under the operating conditions relevant to the REMEDHYS project. At the component level, a single-cassette Modelica model is developed in which the reactive hydride bed is thermally coupled to the heat transfer fluid (HTF) circuit through a lumped-parameter representation of the heat exchanger. At the system level, multiple cassettes are integrated within a functional zone that includes a closed-loop glycol-water thermal management circuit and a set of supervisory control loops governing the transition between absorption and desorption phases.

2.2.1 PCI and thermodynamic parameters

The thermodynamic behaviour of the selected intermetallic alloy is characterised through PCI, which describe the equilibrium relationship between the hydrogen partial pressure and the storage capacity (expressed in wt%) at a fixed temperature. The PCI dataset adopted in this work is taken from Barale and Baricco (2022), who provide the comprehensive sorption characterisation of an in-

dustrially produced $TiFe_{0.85}Mn_{0.05}$ alloy developed within the HyCARE project. This dataset serves as a representative reference for TiFe-based intermetallic alloys, on the assumption that its thermodynamic behaviour is potentially comparable to the alloy currently under development by REMEDHYS. Under the nominal absorption operating conditions of 36 bar and 55 °C, established by the project, the alloy exhibits a maximum reversible hydrogen capacity of 1.0 wt%, which is adopted as a preliminary design assumption for the single-cassette model. The temperature dependence of the plateau pressure p_{eq} is described by the Van 't Hoff relation (Colbe et al. 2019; Dematteis and Baricco 2021):

$$\ln\left(\frac{p_{eq}}{p_0}\right) = -\frac{\Delta H}{RT} + \frac{\Delta S}{R} \tag{10}$$

where p_{eq} is the equilibrium hydrogen pressure, p_0 is atmospheric pressure, R is the universal gas constant, and T is the temperature expressed in K. The enthalpy ΔH [J mol^{-1}] and entropy ΔS [J mol^{-1} K^{-1}] of the sorption reactions are taken from Barale and Baricco (2022). The implementation of the PCI model in the Modelica environment requires the equilibrium absorption $p_{eq_{ABS}}$ and desorption $p_{eq_{DES}}$ pressures to be expressed as functions of both temperature and hydrogen content over a common set of uniformly spaced hydrogen capacity levels. Since the experimental PCI data of Barale et al. are not reported at common wt% increments across all isotherms, the equilibrium pressures were pre-processed by linear interpolation between adjacent experimental data points. This interpolation was conducted at capacity levels ranging from 0.2 to 1.0 wt% in steps of 0.05 wt% for each of the four available isotherms at 34, 40, 55, and 70 °C. The resulting processed dataset, illustrated in Figure 2, overlays the original experimental curves with the interpolated equilibrium pressures at the selected common capacity levels.

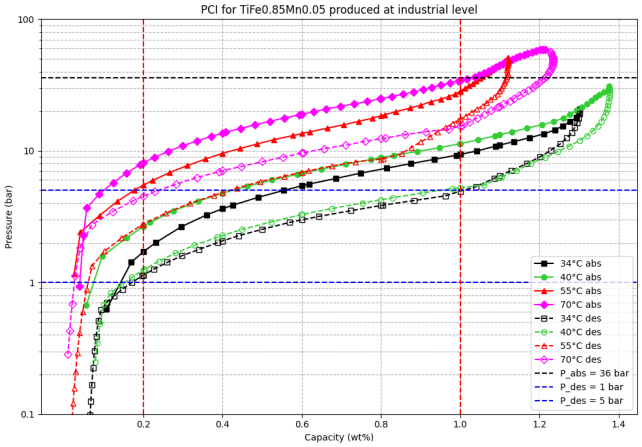


Figure 2. Interpolated absorption (full symbols) and desorption (empty symbols) PCI curves for $TiFe_{0.85}Mn_{0.05}$ (Barale and Baricco 2022)

The PCI exhibits a characteristic sloped plateau region corresponding to the phase transition from the solid solution (α-phase) toward the monohydride (β-phase) and the fully hydrogenated dihydride phase (γ-phase) (Shen et al. 2025). This progressive variation of the equilibrium pressure with H_2 content is of particular relevance for the dynamic model, as it causes the absorption driving force to diminish continuously as the material approaches its upper capacity limit. In the Modelica model, the pre-processed PCI dataset is stored in two separate two-dimensional look-up table arrays of type `CombiTable2Ds`, one for absorption (ABS) and one for desorption (DES), each with n_{MH} instances corresponding to the radial discretisation layer of the metal hydride bed. Linear segment interpolation (*LinearSegments*) is applied to evaluate the equilibrium pressure as a function of the local hydrogen content $w[j] = MH_{ratio}[j] \times 100$ [wt%] and the local layer temperature $T_{MH}[j]$. The resulting p_{eq} obtained from the look-up tables is subsequently normalised with respect to the atmospheric pressure p_0.

2.2.2 Metal Hydride bed

The metal hydride volume is represented by a radially discretised Modelica component, in which the cylindrical MH bed is divided into n_{MH} concentric annular layers to resolve the radial temperature and concentration gradients arising from the low thermal conductivity of the packed bed. The model integrates PCI equilibrium, first-order sorption kinetics, cylindrical Fourier conduction across n_{MH} radial layers, and a gas-phase species and energy balance, all expressed for n_{pipe} identical tubes in parallel. A key feature of the model is that the entire cylindrical geometry is derived analytically from the declared powder mass m_{powder}, the bulk density ρ_{MH}, the number of parallel pipes n_{pipe}, and the axial pipe length L ensuring full mass consistency without any independently prescribed radius. The derivation process initiates with the calculation of the MH powder volume for a single pipe, which can be expressed as:

$$V_{MH\,pipe} = \frac{m_{powder}}{\rho_{MH}\, n_{pipe}} \tag{11}$$

Assuming the powder fills a solid cylinder volume of radius R_{MH} and length L, the outer radius, which aligns with the HTF contact surface, can be derived as follows:

$$R_{MH} = \sqrt{\frac{V_{MH\,pipe}}{\pi L}} \tag{12}$$

Subsequently, the cylinder is divided into n_{MH} concentric annular layers Δr of equal radial width ($r_{n_{MH}} = R_{MH}$):

$$\Delta r = \frac{R_{MH}}{n_{MH}}, \quad r_j = j\Delta r, \quad j = 1, \ldots, n_{MH} \tag{13}$$

where layer $j = 1$ occupies the innermost disk (axis to r_1) and layer $j = n_{MH}$ the outermost annulus, whose outer surface is in contact with the HTF. The MH mass contained in each layer per single pipe is computed from the cylindrical annulus cross-section:

DOI
10.3384/ecp223271

$$m_{MH,j} = \begin{cases} \rho_{MH}\,\pi\, r_1^2 L & j = 1 \\ \rho_{MH}\,\pi \left(r_j^2 - r_{j-1}^2\right) L & j > 1 \end{cases} \tag{14}$$

where, by construction, $\sum_{j=1}^{n_{MH}} m_{MH,j} \cdot n_{pipe} = \rho_{MH}\,\pi R_{MH}^2 L n_{pipe} = m_{powder}$, so the mass closure is exact. The HTF contact area and the inner boundary area of each layer per single pipe are computed as follows:

$$A_{Ex} = 2\pi R_{MH} L \tag{15}$$

$$A_{MH,j} = \begin{cases} 0 & j = 1 \\ 2\pi\, r_{j-1} L & j > 1 \end{cases} \tag{16}$$

The MH_volume block solves the following coupled mass and energy balances. The hydrogen uptake of each radial layer is described by a first order (linear driving force) kinetic law:

$$\frac{d\,Cap_j}{dt} = rate_{ABS/DES} \tag{17}$$

where Cap_j [kg] is the hydrogen mass stored in layer j across all n_{pipe} tubes, and $rate_{ABS/DES}$ is the net sorption rate, positive in absorption and negative in desorption, obtained by a smooth switch between the two contributions at the equilibrium pressure crossing. The two contributions are expressed by a first-order law in the available capacity:

$$rate_{ABS} = K_{ABS}\,(Cap_0 - Cap)\,(DF_{ABS}) \tag{18}$$

$$rate_{DES} = K_{DES}\,(Cap)\,(DF_{DES}) \tag{19}$$

where $Cap_{0,j} = m_{powder,j} Rev_cap\, n_{pipe}$ is the maximum hydrogen mass storable in layer j. The driving forces $DF_{ABS/DES}$ are defined as:

$$DF_{ABS} = \text{smooth}\left\{ \left(\frac{p_{H2}}{P_{eq_{ABS}}} - 1 \right), 0,\, p_{H2} - P_{eq_{ABS}},\, dp_{eq} \right\} \tag{20}$$

$$DF_{DES} = \text{smooth}\left\{ \left(\frac{p_{H2}}{P_{eq_{DES}}} - 1 \right), 0,\, p_{H2} - P_{eq_{DES}},\, dp_{eq} \right\} \tag{21}$$

where the smooth function enforces continuity at the threshold pressure, and $P_{eq_{ABS}}$ and $P_{eq_{DES}}$ are the equilibrium pressures interpolated from the PCI isotherms described above. All the kinetic and thermophysical parameters of the bed are collected in Table 2. The kinetic constants were taken from the HyCARE project characterisation of the same alloy family (HyCARE-Project 2026) and were not re-fitted; they are therefore preliminary. In this study the kinetic constants were taken as temperature independent, so no activation energy appears in the formulation: the temperature dependence of the sorption rate is carried by the equilibrium pressures interpolated from the experimental PCI isotherms. A temperature-dependent kinetic law will be introduced once time-resolved sorption curves become available. Moreover, the conductive heat transfer between adjacent layers follows the cylindrical Fourier law. The heat flux at the outer boundary of layer j, denoted $\dot{Q}_{MH,j}$ (positive inward, i.e. from HTF toward the axis), is:

Table 2. Kinetic and thermophysical parameters of the MH bed model

Parameter	Symbol	Value
Absorption kinetic constant	K_{ABS} [s^{-1}]	0.0814 ± 0.005
Desorption kinetic constant	K_{DES} [s^{-1}]	0.01422 ± 0.005
Reversible capacity	*Rev_Cap* [-]	0.01
Smoothing tolerance	*dp_eq* [Pa]	100
Reference pressure	*P_0* [Pa]	1.0×10^5
Reaction enthalpy	ΔH [J/mol]	$31\,800 \pm 100$
Bulk density	ρ_{MH} [kg/m^3]	4500 ± 300
Specific heat	$c_{p,MH}$ [J/(kg·K)]	450 ± 50
Thermal conductivity	λ_{MH} [W/(m·K)]	0.18 ± 0.02
Gas–solid conductance	UA_{gas-MH} [W/K]	50 ± 50

$$\dot{Q}_{MH,j} = \frac{2\pi\,\lambda_{MH} L n_{pipe}}{\ln\left(r_{j+1}/r_j\right)} \left(T_{MH,j+1} - T_{MH,j}\right),\ j < n_{MH} \tag{22}$$

$$\dot{Q}_{MH,n_{MH}} = \frac{2\pi\,\lambda_{MH} L n_{pipe}}{\ln\left(R_{MH}/r_{n_{MH}-1}\right)} \left(T_{HTF} - T_{MH,n_{MH}}\right) \tag{23}$$

where λ_{MH} is the effective thermal conductivity of the packed bed. For the special case $n_{MH} = 1$, the logarithm singularity at $r = 0$ is avoided by replacing r_0 with an effective inner radius $r_{inner} = f_r R_{MH}$, with $f_r \ll 1$. The energy balance of each layer, expressed for all n_{pipe} tubes in parallel, reads:

$$m_{MH,j}\, n_{pipe}\, c_{p,MH} \frac{d\,T_{MH,j}}{dt} = \begin{cases} \dot{Q}_{MH,1} + \dot{H}_{MH,1} + \dot{E}_{MH,1} & j = 1 \\ \dot{Q}_{MH,j} - \dot{Q}_{MH,j-1} + \dot{H}_{MH,j} + \dot{E}_{MH,j} & j > 1 \end{cases} \tag{24}$$

where $\dot{H}_{MH,j} = (\Delta H / M_{H_2})\,\dot{m}_j$ is the sorption enthalpy source term, ΔH is the reaction enthalpy of the absorption and desorption reactions, and $\dot{E}_{MH,j} = UA_{gas-MH}\,(T - T_{MH,j})$ accounts for convective exchange between the H_2 gas phase (at bulk temperature T) and the solid. The zero-Neumann boundary condition at the axis ($\partial T/\partial r|_{r=0} = 0$) is enforced implicitly by the absence of an inward flux term for $j = 1$ (Hasnain et al. 2026). The heat exchanged with the HTF is coupled to the external thermal port through $\dot{Q}_{port} = \dot{Q}_{MH,n_{MH}}$, consistent with the Modelica heat port sign convention. The overall state of charge is finally defined as:

$$SoC = \frac{\sum_{j=1}^{n_{MH}} Cap_j}{\max\left(\varepsilon,\ \sum_{j=1}^{n_{MH}} Cap_{0,j}\right)} \tag{25}$$

2.2.3 Metal Hydride Module

As shown in Figure 3, the model of the single MH module integrates four sub-models: the MH reactive bed (MH_volume), the HTF serpentine circuit for heating and cooling, the thermal dissipation block with the environment, and the hydrogen mass ports for the absorption and desorption flows.

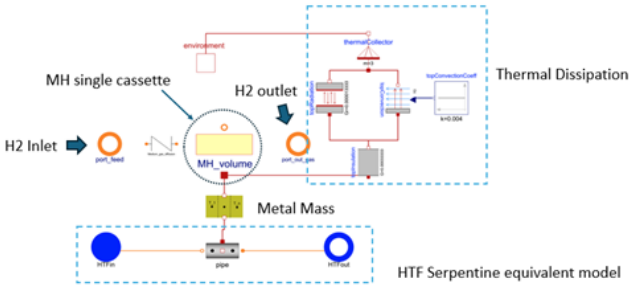


Figure 3. Modelica scheme of REMEDHYS single MH module

The MH module wraps the MH bed described above, adding a second discretisation level along the axial direction, yielding a pseudo-two-dimensional representation of the MH bed as shown in Figure 4. The tank is divided into n equal segments along the flow axis z. Within each segment the radial discretisation into n_{MH} concentric layers (from $j = 1$ at the axis to $j = n_{MH}$ at the HTF contact surface) is handled internally by the volume model, producing an $n \times n_{MH}$ grid of computational cells.

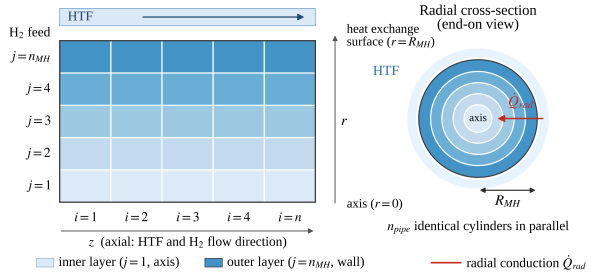


Figure 4. Pseudo-two-dimensional discretisation of the MH volume

Adjacent axial segments are coupled in series through gas flow resistances that resolve H_2 pressure drop and species transport, while the HTF circuit is discretised into the same n segments, so that each fluid cell exchanges heat with the corresponding radial stack of the bed.

2.2.4 Metal Hydride Functional zone

In this section, the model of the metal hydride functional zone for REMEDHYS is presented. The single-cassette model described above was first assembled, with an average behaviour assumed for the thermal conductivity of the metal frame, the low conductivity of the hydride bed being a severe bottleneck in thermal management (Atalmis et al. 2024; Pentimalli et al. 2017).

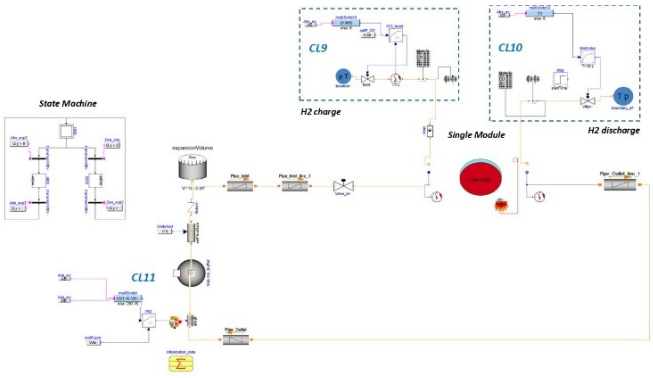


Figure 5. Layout of the MH functional zone model

Subsequently, the closed-loop cooling/heating circuit using glycol water was modelled by implementing the heater/chiller, a water buffer volume, the pump, and the expansion vessel, as presented in Figure 5.

The control loops implemented in the metal hydride storage model are listed in Table 3, together with the set point of each loop and the tuning parameters of the corresponding controller. Two loops, CL9 and CL11, are based on proportional-integral blocks acting respectively on the feed valve and on the thermal power supplied to the heat transfer fluid, whereas CL10 is an ON/OFF actuation of the discharge valve smoothed by a first-order lag representing the actuator dynamics. The two flow loops are driven by a state machine, developed with the Modelica StateGraph library (Otter et al. 2005), that governs the absorption (ABS) and desorption (DES) phases and prevents the feed and discharge lines from being active at the same time.

Table 3. List of control loops dedicated to MH functional zone

CL	Set point, description and controller tuning
9	**Inlet H_2 mass flow rate.** The feed valve, which is the integration point with the PEMEL BoP, is modulated to reach the mass flow rate required by the MH system. *PI; $k = 1\times10^4$; $T_i = 100$ s; $y \in [0, 1]$ (valve opening); set point switched by the ABS state*
10	**Outlet H_2 mass flow rate.** A valve dedicated to the discharging of H_2 is actuated with an ON/OFF logic. *ON/OFF; opening $\in \{0, 1\}$; first-order actuator lag $T = 10$ s; switched by the DES state*
11	**Inlet HTF operating temperature.** Modulation of the thermal power supplied to the heat transfer fluid according to the operating temperature required by the absorption and desorption phases, which are about 48 °C and 70 °C respectively. *PI; $k = 1\times10^3$; $T_i = 0.5$ s; $y \in [-5\,000, 5\,000]$ W; set point switched by the ABS/DES states*

DOI
10.3384/ecp223271

3 Results

3.1 PEMEL load transition

The PEMEL BoP model was simulated for 30 000 s, capturing the transition from 2 MW full operation to 1 MW partial operation. As shown in Figure 6 (right panel), the stack voltage (in red) drops from ∼135 V to ∼128 V and the current (in black) from ∼13 500 A to ∼8 000 A; the polarisation curve, Figure 6 (left panel), confirms operation on the ohmic-dominated branch, where voltage is a well-behaved, monotonic function of current density.

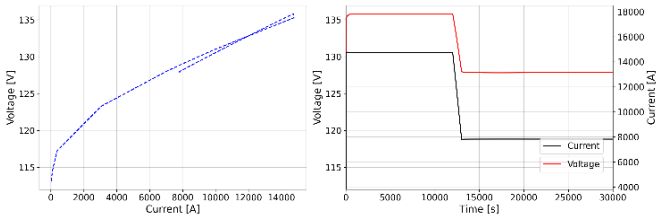


Figure 6. Polarisation curve (left) and voltage (red) and current (black) trend from full load (2 MW) to partial load (1 MW)

Figure 7 shows the system-level response to the nominal to partial load transition at $t \approx 13\,000$ s.

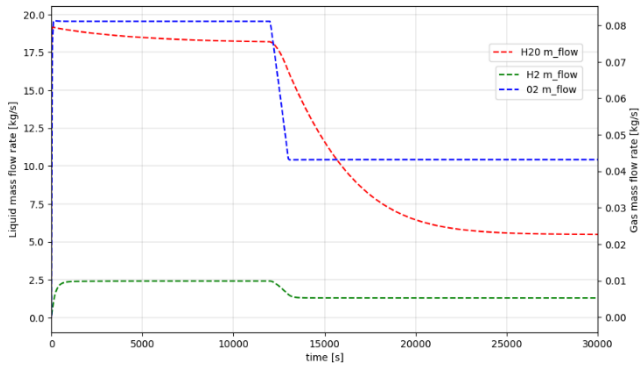


Figure 7. Water, H_2 and O_2 mass flow rates of the PEM electrolysis system.

H_2 and O_2 outlet mass flow rates halve proportionally, confirming Faraday's law, while the variation is smoothed because the water enters the liquid oxygen separator before reaching the stack, introducing a dampening effect that buffers the system's transient response. Figure 8 shows the response of the two thermal control loops to the load step.

Figure 8(a) reports CL4, which holds the temperature difference between the stack and the anodic separator at its set point of 5.5 °C by modulating the recirculation pump. At the load transition, imposed at approximately 12 000 s, the measured difference falls to about 1.8 °C, corresponding to a deviation of 3.7 °C, since the heat released by the stack decreases faster than the circulating flow is adapted. The set point is recovered within roughly 10 000 s as the pump command is reduced from about 40 % to 21 %. The same panel reports the stack thermal

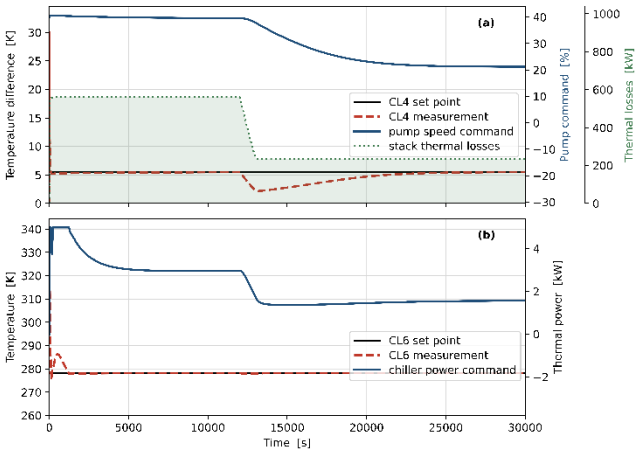


Figure 8. CL4 stack temperature difference, pump speed command and stack thermal losses (a); CL6 hydrogen outlet temperature and chiller power command (b).

losses, which fall from approximately 560 kW to 235 kW across the transition, consistent with the reduced electrochemical dissipation at partial current density. The step in the losses is essentially completed within 600 s, whereas the pump command follows with the lag imposed by the thermal inertia of the circuit. Figure 8(b) reports CL6, which holds the hydrogen outlet temperature at 5 °C. After the start-up transient, during which the measurement descends from ambient conditions and settles on the set point within about 1500 s, the loop tracks the reference throughout; the chiller duty decreases from about 2.8 kW at nominal load to 1.4 kW at partial load, then rises slightly as the outlet stream approaches its new equilibrium.

3.2 MH module absorption–desorption cycle

A 10 h cycle was simulated with a constant HTF flow rate and a constant H_2 mass flow rate from the PEMEL. By considering the boundaries of the electricity availability on the specific demo-site and a trade-off of the optimal charging profile of the end-user, a nominal absorption mass flow rate of 18 kg/h was considered for the MH tank, which acts as an H_2 buffer between the PEMEL and the specific H_2 end use. Figure 9 reports the HTF inlet and outlet temperatures at the MH module over a full simulation of 10 hours. During the absorption period, the HTF is supplied at the cooling setpoint of $T_{HTF,in} \approx 48\,^{\circ}\mathrm{C}$ to remove the exothermic sorption heat from the bed, while during the desorption period (5–8 h) the setpoint is raised to 70 °C to supply the endothermic heat of desorption. During the absorption phase $T_{HTF,out}$ rises above $T_{HTF,in}$, confirming that the HTF is effectively extracting heat from the exothermic MH bed; the maximum temperature difference reaches $\Delta T_{max} = 10.4\,^{\circ}\mathrm{C}$ at $t \approx 1.2$ h, corresponding to the onset of the absorption transient when the sorption rate, and hence the heat release, is at its highest value. During the desorption phase, the $T_{HTF,out}$ drops below $T_{HTF,in}$ as the fluid delivers heat to the endothermic bed, with a maximum difference of $\Delta T_{max} = 4\,^{\circ}\mathrm{C}$ at the start of

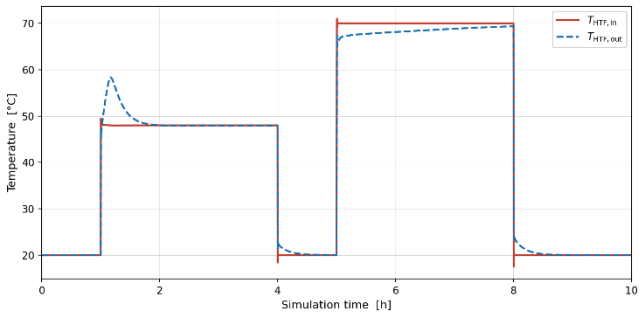


Figure 9. Inlet (red) and outlet (blue) HTF temperature

the discharge transient.

Figure 10 shows the MH temperature field across the 5×5 axial–radial grid. Each colour family represents one axial segment ($i = 1 \ldots 5$), with shades ranging from dark ($j = 1$, axis) to light ($j = 5$, HTF surface); the black dashed line is $T_{HTF,in}$.

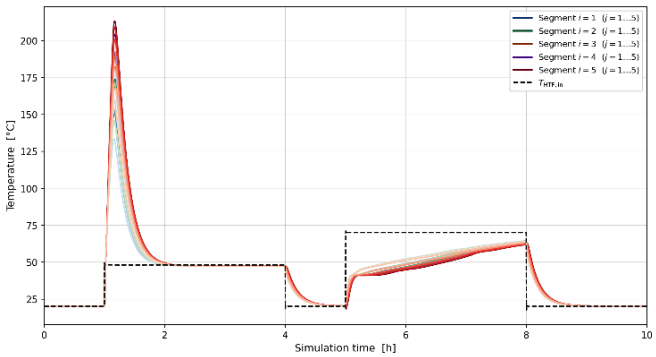


Figure 10. MH internal temperature profile along the pseudo-two-dimensional discretisation, compared with the inlet HTF profile

During the absorption phase, the exothermic reaction drives temperatures well above the HTF cooling setpoint, with peak values increasing from segment $i = 1$ to $i = 5$ due to progressive bed saturation along the flow direction, and a pronounced radial gradient reflecting the low thermal conductivity of the packed bed. During desorption, the endothermic heat supplied by the fluid brings all layers toward thermal equilibrium with it. A sensitivity analysis was finally carried out on a single cassette to characterise the time scales and the modalities of the charge and discharge of the stored hydrogen. The desorption phase is governed by two conceptually distinct parameters. The demand pressure p_{demand} is a boundary condition imposed at the cassette outlet by the end use served, and is therefore not a free design variable, whereas the HTF desorption temperature T_{desorb} is an operational choice, tracked by the control loop CL11 and constrained by the physical limits of the components and by the PCI characteristics of the alloy. Each parameter was varied in turn, all other conditions being unchanged. Every run reproduces a complete cycle: the cassette is charged from an initial SoC of 0.30 at the maximum electrolyser feed rate, 2.50×10^{-4} kg s^{-1} per cassette or about 18 kg h^{-1} over the twenty parallel lines, up to $\mathrm{SoC_{max}} = 0.95$, and is then discharged down to $\mathrm{SoC_{min}} = 0.05$ or to the end of the simulation horizon. Both thresholds are imposed operationally, to avoid complete charging and discharging of the storage. Figure 11 reports the outlet pressure on the left axis and the state of charge on the right axis for a desorption temperature fixed at 70 °C and a demand pressure varied between 1 and 5 bar, the total hydrogen delivered being given in the legend. The back pressure imposed downstream reduces the difference between the plateau pressure of the alloy and the outlet pressure, which is the driving force of desorption, so that the discharge rate falls as p_{demand} is raised. The penalty is markedly non-linear: the lower threshold is reached only at 1 bar, after approximately 2.1 h and with 4281 g delivered, whereas at 5 bar the discharge stalls at SoC ≈ 0.25 and 3292 g, the cumulative loss growing from 6 % at 3 bar to 23 %.

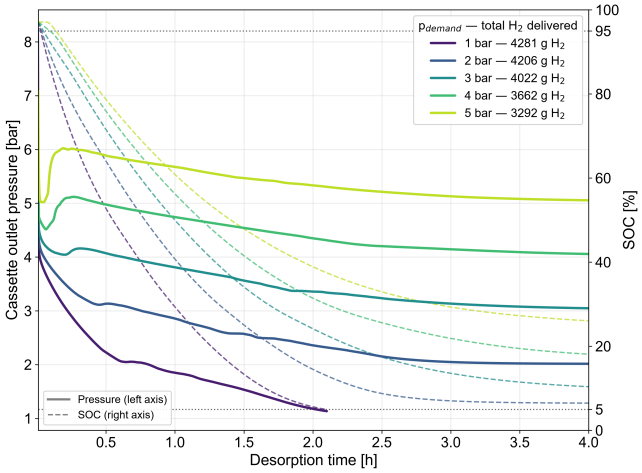


Figure 11. Sensitivity Analysis on p_{demand} from 1 to 5 bar ($T_{desorb} = 70\,°\mathrm{C}$ fixed)

Figure 12 shows the complete cycle at 5 bar, with T_{desorb} varied between 50 and 70 °C, the upper value being the temperature limit adopted operationally for the system. As expected, the absorption phase is identical in all five runs, the alloy charging from $\mathrm{SoC_0} = 0.30$ and reaching $\mathrm{SoC_{max}} = 0.95$ after 3.48 h, with 3.12 kg of hydrogen absorbed. The curves diverge only once the state machine triggers the transition to desorption. A higher temperature shifts the PCI curve of the alloy upwards, so that a given hydrogen content corresponds to a higher plateau pressure, while also increasing the thermal driving force sustaining the endothermic reaction. The peak outlet pressure consequently increases from 5.40 to 6.02 bar and the delivered mass is approximately doubled, from 1671 to 3292 g, the residual state of charge falling from 0.56 at 50 °C to 0.25 at 70 °C. Both analyses show that the demand pressure is at least as influential as the desorption temperature in determining the usable capacity of the cassette. Since it is dictated by the end use rather than by design, operation at 5 bar must be assumed, notwithstand-

DOI
10.3384/ecp223271

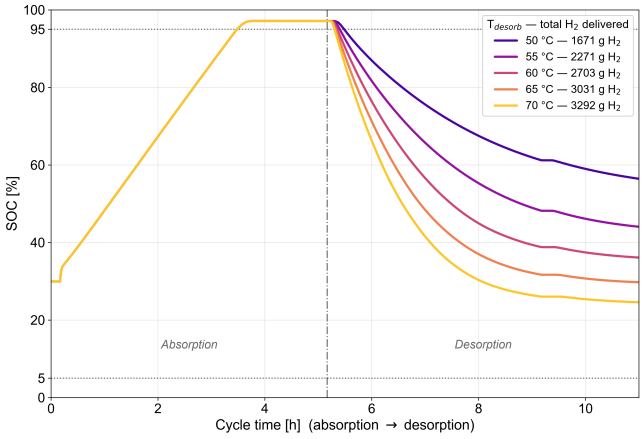


Figure 12. Sensitivity Analysis on T_{desorb} from 50 to 70 °C (5 bar fixed)

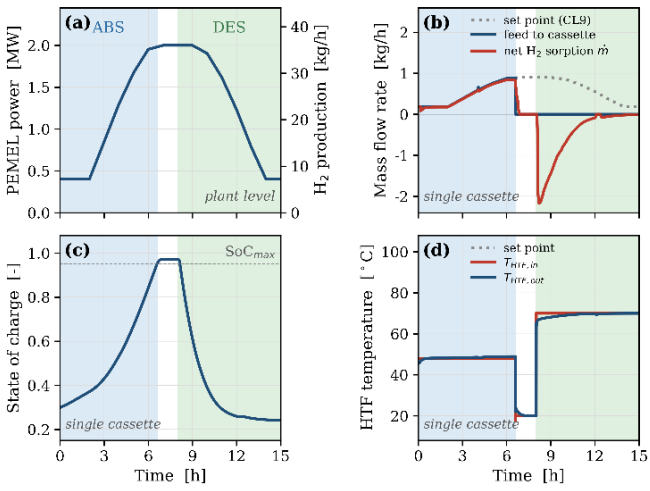


Figure 13. Coupled scenario over a 15 h dispatch cycle. (a) Plant level: electrolyser power and corresponding hydrogen production; (b)–(d) feed and net H_2 sorption rate, SoC and HTF temperatures for a single cassette of twenty in parallel.

ing the 23 % penalty with respect to atmospheric delivery. The desorption temperature is then the only remaining degree of freedom, and the upper operational limit of 70 °C is retained, as it maximises the recoverable capacity under that constraint.

3.3 Coupled operation scenario

Finally, to assess the interaction of the two functional zones, a coupled scenario was simulated. In this scenario the hydrogen supplied to the MH storage is derived from the electrolyser production under a variable power input, rather than being imposed as a constant boundary condition. The power profile, ranging from 20 % partial load to the nominal 2 MW, determines the stack current and hence the hydrogen production, which varies between 0.002 and 0.010 kg/s. Up to 50 % of the nominal production is directed to the MH storage and distributed over 20 cassettes in parallel, so that the feed of the single cassette ranges from 0.18 to 0.90 kg/h and constitutes the set point of loop CL9. The results reported below refer to one of the twenty cassettes. The transition between charging and discharging is governed by the state machine, which enables the opening of the hydrogen valves only once the HTF loop has reached the set point of the phase, and which terminates the phase when the state of charge exceeds SoC_{max} or when the dispatch signal indicates a reversal. The HTF is accordingly held at 48 °C during absorption and at 70 °C during desorption, with an intermediate step at 20 °C imposed while the loop is reconditioned between the two phases. No thermal integration has been considered, since the electrolyser waste heat is available at 50 to 60 °C, which is insufficient to sustain desorption at 70 °C, while absorption requires heat removal at 48 °C. Figure 13 reports the results.

During the charging window, the cassette absorbs 3.09 kg of hydrogen out of the 3.13 kg fed, resulting in an increase of the state of charge from 0.30 up to the upper threshold SoC_max, set at 0.95. The saturation point is reached at 6.6 h, coinciding with the end of the window imposed by the dispatch profile, which indicates that the twenty-cassette configuration is consistent with the daily surplus of the plant. The sorption rate follows the feed with the delay imposed by the heat transport through the bed, and the HTF temperature rise across the cassette reaches 3.4 °C in absorption. Discharge is not flow controlled, consequently the initial desorption rate reaches 2.2 kg/h, approximately 2.5 times the average feed rate during charging, and decays as the equilibrium pressure falls with the state of charge. Desorption releases 3.41 kg and stops at a state of charge of 0.24, well above the corresponding lower threshold. At 70 °C, the equilibrium pressure falls below the 5 bar back-pressure imposed at the outlet, resulting in discharge being limited by the thermodynamics of the alloy rather than by the control logic, and the residual capacity could only be recovered by raising the HTF set point or lowering the delivery pressure.

4 Conclusions

This paper presents a modular, equation-based dynamic model of a 2 MW PEM electrolyser and of a $TiFe_{0.85}Mn_{0.05}$ metal hydride storage cassette, developed in Modelon Impact within the EU REMEDHYS project. First-principles electrochemistry, PCI data of an industrially produced alloy and a closed-loop glycol-water HTF circuit are integrated within a single simulation environment.

Three findings emerge. Firstly, the eight-loop PI architecture holds all PEMEL variables within their set points during the 2 MW to 1 MW load step, with stack voltage and current falling from about 135 V and 13 500 A to 128 V and 8 000 A and the hydrogen and oxygen flow rates halving in accordance with Faraday's law. Secondly, the pseudo-two-dimensional cassette model resolves the radial and axial temperature gradients that govern the

sorption rate, and the sensitivity analysis shows that the usable capacity is dictated primarily by the delivery pressure imposed by the end use: at 70 °C the hydrogen delivered per cycle falls from 4281 g at atmospheric delivery to 3292 g at 5 bar, whereas raising the desorption temperature from 50 to 70 °C at 5 bar approximately doubles the recovered mass. Lastly, the coupled scenario shows that a cassette fed by a variable electrolyser output reaches the upper state-of-charge threshold at 6.6 h, against 3.48 h at the maximum feed rate, so that the twenty-cassette configuration is consistent with the surplus available at the demonstration site, while discharge stops at a state of charge of 0.24, limited by the equilibrium pressure of the alloy rather than by the control logic. The results constitute a verification of the numerical implementation and a preliminary assessment rather than an experimental validation. Three developments are foreseen. The single module will be validated against the data collected within REMEDHYS, allowing the kinetic constants and the PCI dataset to be refined for the specific recycled alloy batch. The twenty cassettes will then be simulated simultaneously under a supervisory state machine staggering their operating windows, so that a subset is always available for uptake or delivery while the thermodynamically unfavourable ends of the plateau are avoided. Finally, thermal integration with the electrolyser will be assessed: although the PEM cooling circuit is too cold to sustain desorption, recovery of its waste heat during the start-up of the absorption phase, and coupling with higher-temperature electrolyser technologies, remain open opportunities that the shared HTF boundary of the present model allows to be explored.

Acknowledgements

This work received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement No. 101192503 (REMEDHYS), within the Clean Hydrogen Partnership. Views expressed are those of the authors only.

EMD, PR and MB acknowledge support from the Project CH4.0 under the MUR program "Dipartimenti di Eccellenza 2023–2027" (CUP: D13C22003520001).

References

Alleori, Riccardo et al. (2025-05). "Experimental analysis on a Power-to-Hydrogen system based on PEM electrolysis and metal hydrides: Empirical correlations for optimal system design and performance improvement". In: *International Journal of Hydrogen Energy* 128, pp. 306–318. DOI: 10.1016/j.ijhydene.2025.04.201.

Atalmis, Gokay, Kamran Sattarkhanov, Recai Nuri Kaplan, Metin Demiralp, and Yuksel Kaplan (2024-07). "The effect of powder and pellet forms of added metal hydride materials on reaction kinetics and storage". In: *International Journal of Hydrogen Energy* 75, pp. 98–105. DOI: 10.1016/j.ijhydene.2023.12.262.

Barale, Jussara and Marcello Baricco (2022-08). "TiFe0.85Mn0.05 alloy produced at industrial level for a hydrogen storage plant". In: *International Journal of Hydrogen Energy* 47.69, pp. 29866–29880. DOI: 10.1016/j.ijhydene.2022.06.295.

Benmehel, Abdellah, Sara Chabab, Ana Laura Do Nascimento Rocha, Manuel Chepy, and Tahar Kousksou (2024-10). "PEM water electrolyzer modeling: Issues and reflections". In: *Energy Conversion and Management: X* 24, p. 100738. DOI: 10.1016/j.ecmx.2024.100738.

Cavo, Massimo, Gioele Gadducci, Davide Rattazzi, Massimo Rivarolo, and Loredana Magistri (2021-09). "Dynamic analysis of PEM fuel cells and metal hydrides on a zero-emission ship". In: *International Journal of Hydrogen Energy* 46.64, pp. 32630–32644. DOI: 10.1016/j.ijhydene.2021.07.104.

Colbe, Jose Bellosta Von et al. (2019-03). "Application of hydrides in hydrogen storage and compression: Achievements, outlook and perspectives". In: *International Journal of Hydrogen Energy* 44.15, pp. 7780–7808. DOI: 10.1016/j.ijhydene.2019.01.104.

Dematteis, Erika Michela and Marcello Baricco (2021-09). "Fundamental hydrogen storage properties of TiFe-alloy with partial substitution of Fe by Ti and Mn". In: *Journal of Alloys and Compounds* 874, p. 159925. DOI: 10.1016/j.jallcom.2021.159925.

Falcão, D. S. and A. M. F. R. Pinto (2020-07). "A review on PEM electrolyzer modelling: Guidelines for beginners". In: *Journal of Cleaner Production* 261, p. 121184. DOI: 10.1016/j.jclepro.2020.121184.

Fritzson, Peter (2015). *Principles of Object-Oriented Modeling and Simulation with Modelica 3.3*. IEEE Press/Wiley.

Hasnain, Muhammad, Haluk Sezer, and Jack H. Mason (2026-01). "Mathematical modeling of heat and mass transfer in metal hydride hydrogen storage systems: A comprehensive review". In: *Renewable and Sustainable Energy Reviews* 226, p. 116294. DOI: 10.1016/j.rser.2025.116294.

HyCARE-Project (2026). *HyCARE-Project – HyCARE-Project*. URL: https://hycare-project.eu/ (visited on 2026-05-14).

Modelon AB (2026). *Modelon*. URL: https://www.modelon.com/ (visited on 2026-08-01).

Olivier, Pauline, Chakib Bourasseau, and Belkacem Bouamama (2017-10). "Low-temperature electrolysis system modelling: A review". In: *Renewable and Sustainable Energy Reviews* 78, pp. 280–300. DOI: 10.1016/j.rser.2017.03.099.

Otter, Martin, Karl-Erik Årzén, and Isolde Dressler (2005-03). "StateGraph – A Modelica Library for Hierarchical State Machines". In: *4th International Modelica Conference*. Hamburg, Germany, pp. 569–578.

Pentimalli, Mariangela, Emanuela Imperi, Annalisa Zaccagnini, and Franco Padella (2017-09). "Nanostructured metal hydride – Polymer composite as fixed bed for sorption technologies. Advantages of an innovative combined approach by high-energy ball milling and extrusion techniques". In: *Renewable Energy* 110, pp. 69–78. DOI: 10.1016/j.renene.2016.07.074.

Pfennig, Marc, Bernd Schiffer, and Tanja Clees (2025). "Thermodynamical and electrochemical model of a PEM electrolyzer plant in the megawatt range". In: *International Journal of Hydrogen Energy* 104, pp. 567–583. DOI: 10.1016/j.ijhydene.2024.04.335.

Polo-Molina, Alejandro, Jose Portela, Luis Alberto Herrero Rozas, and Ricardo Correia González (2025-06). *Modeling Membrane Degradation in PEM Electrolyzers with Physics-*

Informed Neural Networks. DOI: 10.48550/arXiv.2507.02887. arXiv: 2507.02887.

REMEDHYS (2026). *REMEDHYS*. URL: https://www.remedhys.eu/ (visited on 2026-04-20).

Riera, Juan A., Ricardo M. Lima, and Omar M. Knio (2023-04). "A review of hydrogen production and supply chain modeling and optimization". In: *International Journal of Hydrogen Energy* 48.37, pp. 13731–13755. DOI: 10.1016/j.ijhydene.2022.12.242.

Schalenbach, Maximilian, Marcelo Carmo, Daniel L. Fritz, Jurgen Mergel, and Detlef Stolten (2013-11). "Pressurized PEM water electrolysis: Efficiency and gas crossover". In: *International Journal of Hydrogen Energy* 38.35, pp. 14921–14933. DOI: 10.1016/j.ijhydene.2013.09.013.

Shen, Shaobo, Yang Li, Liuzhang Ouyang, Lei Zhang, Min Zhu, and Zhao Liu (2025-12). "V–Ti-Based Solid Solution Alloys for Solid-State Hydrogen Storage". In: *Nano-Micro Letters* 17.1, p. 175. DOI: 10.1007/s40820-025-01672-w.

TLK Energy (2026). *Process simulation with PSL I TLK Energy*. URL: https://tlk-energy.de/en/software/process-systems-library (visited on 2026-05-14).

Usman, M. R. (2022-10). "Hydrogen storage methods: Review and current status". In: *Renewable and Sustainable Energy Reviews* 167, p. 112743. DOI: 10.1016/j.rser.2022.112743.

Wang, Xin et al. (2025-09). "Challenges and opportunities in hydrogen storage and transportation: A comprehensive review". In: *Renewable and Sustainable Energy Reviews* 219, p. 115881. DOI: 10.1016/j.rser.2025.115881.

Zhang, Kun et al. (2022-12). "Status and perspectives of key materials for PEM electrolyzer". In: *Nano Research Energy* 1, e9120032. DOI: 10.26599/NRE.2022.9120032.

DOI
10.3384/ecp223271

Waste to Hydrogen System Dynamic Modelling

Michele Bolognese[1] Emanuele Martinelli[1] Michele Urbani[1] Matteo Testi[1] Annarita Salladini[2]
Alessia Borgogna[2] Maria Rosaria Pepe[2]

[1]Center for Sustainable Energy, Fondazione Bruno Kessler, Italy,
{mbolognese,emartinelli,murbani,testi}@fbk.eu
[2]MyRechemical, NextChem, Viale Castello della Magliana 75, Rome, Italy,
{a.salladini,a.borgogna,m.pepe}@nextchem.com

Abstract

Waste gasification is a promising thermochemical route for converting municipal solid waste and biomass into syngas for clean hydrogen production. This paper presents a Modelica-based dynamic model of the complete Balance of Plant (BoP) of the Hy2Rome Important Project of Common European Interest (IPCEI), focusing on three key contributions: (i) the integration of the gasifier Reduced Order Model (ROM), exported as a Functional Mock-up Unit (FMU), into the Modelica environment; (ii) the dynamic model of the Temperature Swing Adsorption (TSA) unit; and (iii) the dynamic model of the Pressure Swing Adsorption (PSA) system for final hydrogen purification. All BoP components were calibrated against Heat and Material (H&M) balance data, achieving acceptable relative errors. The PSA model implements a four-vessel, eight-step cycle delivering hydrogen at a nominal purity of 99.99 %, maintained during the adsorption steps at cyclic steady state. The TSA model operates a two-vessel thermal swing cycle between 40°C and 270°C with a PID-controlled state machine.

Keywords: Modelica, FMU, Gasification, Syngas Purification, PSA, TSA, Hydrogen, Balance of Plant

1 Introduction

The global energy transition requires effective strategies for converting waste and biomass into clean fuels. Gasification thermochemically converts carbonaceous feedstocks into synthesis gas (syngas) through partial oxidation at elevated temperatures (Arena 2012; Bashir et al. 2025). High-purity hydrogen production from syngas enables fuel cell operation and chemical synthesis with near-zero carbon footprint when the feedstock is waste-derived (Blasi 2004; Mustapha et al. 2025). Waste gasification plants present unique engineering challenges. The heterogeneous composition of municipal solid waste (MSW) and biomass introduces syngas quality variability, requiring robust downstream purification trains for a wide spectrum of contaminants including HCl, H_2S, COS, HCN and Hg (Abdoulmoumine et al. 2015). The purification section typically comprises a Temperature Swing Adsorption (TSA) (Morales-Ospino et al. 2020; Tlili et al. 2009) followed by a Water Gas Shift (WGS) reactor and Pressure Swing Adsorption (PSA) for hydrogen purification (J. Yang et al. 1997). Dynamic modelling is a crucial tool for design, optimisation and control of complex energy systems. The Modelica language, with its equation-based acausal formulation and multi-domain capability, is particularly well suited to BoP modelling that integrates a multi-domain approach involving thermal, electrical, chemical and control phenomena (Mattsson and Elmqvist 1997; Qiu et al. 2024). Reduced Order Modelling (ROM) techniques address the high computational cost of high-fidelity gasifier models built from CFD results, while the FMI standard enables export of ROMs as FMUs for seamless integration into system-level simulation environments (Sahraei et al. 2015; Monaghan and Ghoniem 2012; Blochwitz et al. 2012). This work focuses on the gasifier FMU integration methodology, the TSA and PSA models with state machine control, and the calibration of all components against process H&M balance data using Modelon Impact.

2 System Description

The process flow diagram (PFD) is subdivided into three main sections: (i) Gasification, Primary Gas Cleaning and Syngas Stabilisation – evaporative quench, sedimentation tank, acid scrubber, alkaline scrubber, subcooling tower, gasometer; (ii) Gas Compression and Purification – compressor, metal removal beds, halide removal beds, hydrolysis reactor, mercury removal bed, H_2S removal system, TSA unit, ZnO desulphurisation beds; and (iii) Hydrogen Production – WGS reactor, PSA unit. Figure 1 shows the BoP block diagram.

Representative industrial syngas compositions and operating conditions were selected to ensure model realism and alignment with typical entrained-flow gasification processes. Specifically, three parallel gasifier lines were modelled, each producing syngas that converges and combines at the inlet of the downstream alkaline scrubber. This configuration reflects standard industrial layouts, promoting operational redundancy, load balancing, and efficient syngas pretreatment (e.g., for H_2S and CO_2 removal) prior to downstream applications such as fuel cells or chemical synthesis.

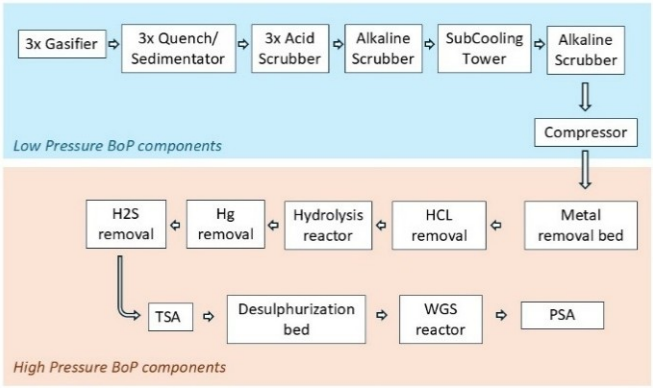


Figure 1. Block schematic of the process flow diagram.

3 Methodology

The modelling methodology adopts a bottom-up approach, in which fundamental component models – including thermodynamic medium properties, lumped pipe representations with trace gas reduction, condenser dynamics, and friction correlations – are developed independently and subsequently integrated into the low-pressure and high-pressure balance-of-plant subsections. All component parameters are calibrated against heat-and-mass balance data corresponding to nominal full-load operating conditions.

3.1 Gasifier Reduced Order Model (ROM)

The high-fidelity reference is a CFD model of the gasifier that resolves one-dimensional multiphase reacting flow. Its computational cost – typically hours per second of simulated time – makes direct use in system-level simulation impractical. Model Order Reduction (MOR) methodology was therefore applied to construct a data-driven surrogate that captures the essential input–output behaviour of the full-order model (Sahraei et al. 2015; Monaghan and Ghoniem 2012).

The training dataset was generated through a two-stage simulation campaign on the CFD model. Static simulations mapped the steady-state input–output relationship over a grid of operating conditions, spanning the relevant ranges of refuse-derived fuel (RDF) mass flow rate, gasification agent flow rates, C/H and C/O ratios. Dynamic simulations captured transient responses to step changes in these inputs, providing the temporal information necessary to train the ROM to reproduce time-dependent behaviour. Input variables are: RDF mass flow rate ($\dot{m}_{RDF}$), gasification agent flows ($\dot{m}_{fgb}$, $\dot{m}_{fgh}$), C/H ratio and C/O ratio, and the gasifier internal state variables. Output variables are gas-phase outlet temperature T_g, solid-phase outlet temperature T_s, outlet gas pressure p_g, molar fraction profiles X_i for H_2, CO, CO_2, H_2O, CH_4, and outlet mass flow rates $\dot{m}_{syn}$ and $\dot{m}_{slag}$.

Data pre-processing included normalisation of all input and output variables to the range $[0,1]$ based on the training dataset extremes, and removal of outliers identified through residual analysis. A recurrent layer (LSTM) followed by a feed-forward neural network architecture was selected for the regression task. The mean squared error was used as the loss function and the network was trained using the Adam optimiser (Kingma and Ba 2014). Hyperparameter selection (number of layers, neurons per layer, learning rate) was performed by cross-validation on a held-out validation subset. The average training and validation errors over 10 training runs are depicted in Figure Figure 3. The ROM was tested on a separate test set not used during training, achieving an average MSE of 1.5×10^{-2}, confirming generalisation capability beyond the training operating points.

3.2 FMU Architecture and Integration

The trained ROM was exported as a Functional Mock-up Unit (FMU) following the FMI 2.0 standard. The ROM was encapsulated in an FMU using ONNX2FMU (Urbani et al. 2025) as a compiled shared library with a standardised XML interface description, enabling tool-independent integration.

In Modelon Impact, the FMU is instantiated as a block whose input and output ports correspond to the defined model interface variables. Figure 2 shows the FMU test model assembled in Modelon Impact. A key integration challenge is the mismatch between the FMU output interface – which characterises the gasifier outlet gas in terms of five major species (H_2, CO, CO_2, H_2O, CH_4) – and the Modelica medium model, which additionally includes N_2 and five trace components (HCl, H_2S, COS, HCN, Hg). The dedicated adapter model `X_adapter_nS` bridges this gap. The block performs a purely algebraic, stateless mapping between the species vector returned by the FMU and the composition vector required by the downstream medium, so that no state variable is introduced and no additional dynamics are superimposed on the gasifier response, which is entirely contained within the ROM. The mapping is evaluated at every communication point and consists of two operations. First, the residual molar fraction not resolved by the ROM is assigned to N_2, so that the composition vector sums to unity. Second, the trace species concentrations are computed from the chlorine and sulphur content of the feedstock under the assumption of complete conversion, according to

$$\begin{aligned} \mathrm{Cl_2} + \mathrm{H_2} &\rightarrow 2\,\mathrm{HCl} \\ \mathrm{S} + \mathrm{H_2} &\rightarrow \mathrm{H_2S} \\ \mathrm{S} + \mathrm{CO} &\rightarrow \mathrm{COS} \end{aligned} \tag{1}$$

with the H_2 and CO consumed in these reactions subtracted from the respective FMU output streams so that the elemental balance is preserved across the interface. The only approximation introduced by the adapter is therefore the assumption of complete conversion in Equation 1, which is conservative with respect to the duty imposed on the downstream cleaning units, since it maximises the trace contaminant load reaching them. No equilibrium or

DOI
10.3384/ecp223283

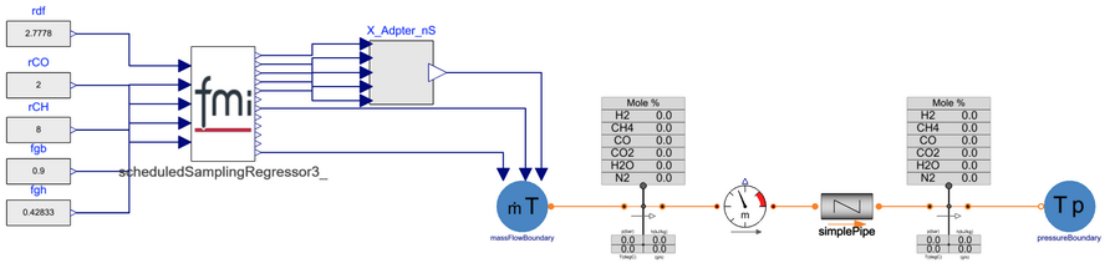


Figure 2. FMU test model assembled in Modelon Impact.

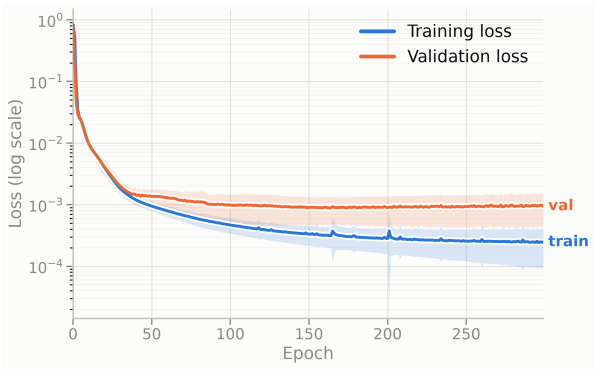


Figure 3. Train and validation error as a function of training epochs.

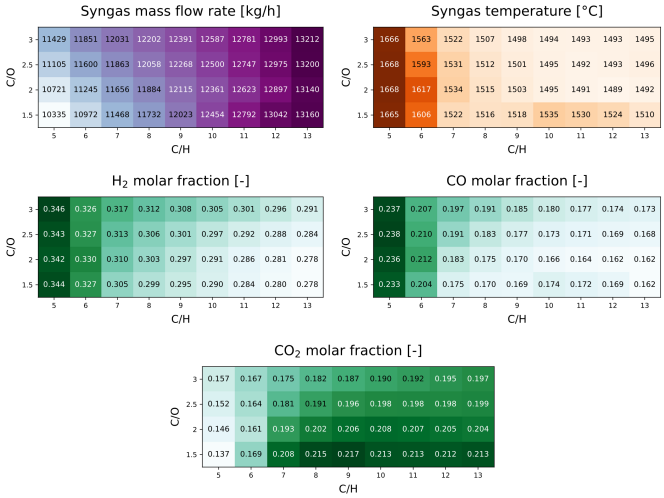


Figure 4. Mass fraction syngas composition as a function of C/O and C/H variability.

kinetic limitation is resolved at this interface; the relevant reduction kinetics are modelled explicitly within the acid gas removal components.

Prior to integration into the full BoP model, the FMU was verified through a sensitivity analysis exploring the response of the syngas composition to variations in the C/H and C/O ratios of the feed, these being representative of the variability of the inlet RDF, which is a mixture of biomass and polyolefins. Figure 4 shows the results at a nominal RDF and gasification agents mass flow rate.

The FMU correctly reproduces the expected trends: increasing C/O ratio leads to higher CO content and lower CO_2 content in the syngas, reflecting the shift from complete to partial oxidation; the C/H ratio governs the relative proportions of H_2 and H_2O. These results confirm that the ROM captures the fundamental gasification chemistry over the explored operating range, providing confidence in its use within the complete BoP model.

3.3 Functional Models for Low-Pressure Components

The medium package defines thermodynamic properties for $n_S = 6$ species $\{H_2, CH_4, CO, CO_2, H_2O, N_2\}$ and $n_C = 5$ traces $\{HCl, H_2S, COS, HCN, Hg\}$. The functional component for all BoP elements is a lumped pipe model incorporating Arrhenius-type trace reduction kinetics:

$$k_{red}[n_C] = A_{red}[n_C] \exp\left(-\frac{E_{a,red}}{R\,T_{gas}}\right) \qquad (2)$$

$$\dot{m}_{C,flow,out}[n_C] = \dot{m}_{C,flow,in}[n_C]\, e^{-k_{red}[n_C]\, t_{res}} \qquad (3)$$

Parameters A_{red} and $E_{a,red}$ are calibrated to reproduce the nominal trace reduction ΔC_s according to the available data from the H&M balance.

4 Low-Pressure Components Section

The low-pressure components section cools raw syngas from 1100°C to 20°C while removing particulates and water-soluble contaminants: Table 1 presents a description of the working principle of the low-pressure components. The cooling of the syngas stream was modelled by taking into account two heat transfer mechanisms: sensible heat transfer, which is already accounted for by the medium ports of the model, and the absorption of latent heat flow Q_{evap} [kW] during the evaporation of the aqueous solution:

$$Q_{evap} = L(T = T_{sol})\, \dot{m}_{flow,sol} \quad [\text{kW}] \qquad (4)$$

where $L(T) = -2.43\,(T - 273.15) + 2499.70$ kJ/kg is a linear fit of the water latent heat of vaporisation. Once the solution temperature T_{sol} [°C] is given to the model as a parameter, a PI controller adjusts the value of the mass flow rate $\dot{m}_{flow,sol}$ [kg/s] sprayed in the quench to maintain the nominal setpoint temperature of 86°C at the outlet section of the quench component.

Table 1. Low-pressure BoP components description.

Component	*Action*	*Op. T [°C]*
Evap. Quench	Liquid water is sprayed into the hot syngas stream to cool the gas down rapidly to its nominal operating temperature of 86°C	86
Acid Scrubber	The syngas is mixed with an acidic scrubbing solution to promote the chemical absorption of basic gas traces, cooling the gas further to 50°C	50
Alkaline Scrubber	The syngas is mixed with a basic scrubbing solution to promote the absorption of acid gas traces, cooling the gas to 40°C	40
Subcooling Tower	The syngas is cooled further to 20°C, the operating temperature of the gasometer in which it is stored	20

5 High-Pressure Components Section

After compression to ∼17 barg, syngas passes through metal removal beds, halide removal beds (HCl slip < 0.1 ppmv), a hydrolysis reactor (COS slip < 6 ppmv, HCN slip < 1 ppmv), a mercury removal bed (Hg slip < 10 ng/Nm^3) and H_2S removal (H_2S slip < 1 ppmv). All components use the pipe model with Arrhenius trace reduction, and Table 2 summarises the calibration results; here, Δ_i represents the trace removal efficiency of the component for the i-th trace:

$$\Delta_i = \frac{C_{in,i} - C_{out,i}}{C_{in,i}} \tag{5}$$

The parameters to be calibrated are A_{red} and E_a, but as the available data covered only one operating point, the calibration was performed by changing only one parameter. A more precise calibration of these parameters can be performed when data becomes available for partial load.

5.1 Water Gas Shift Reactor and Steam-to-Dry Ratio Control

The WGS reactor converts CO and steam into H_2 and CO_2 via the following reversible exothermic reaction ($\Delta H = -41$ kJ/mol):

$$CO + H_2O \leftrightarrow CO_2 + H_2 \tag{6}$$

Approximately 10 % of the syngas from the desulphurisation beds is directed to the WGS reactor; the remaining 90 % proceeds to other production sections. The syngas stream is then mixed with steam and pre-heated before entering the reactor at $T = 320$°C in nominal operating conditions. Steam temperature and flow rate are derived from mass and energy balances:

$$\dot{m}_{syn,in}\, h_{syn,in} + \dot{m}_{steam,in}\, h_{steam,in} = \dot{m}_{mix,out}\, h_{mix,out} \tag{7}$$

Reactor performance is quantified by the steam-to-dry ratio (S/dry), defined as the molar flow rate of steam over the molar flow rate of syngas in dry conditions:

$$\mathrm{S/dry} = \frac{\dot{n}_{H_2O}}{\dot{n}_C} \tag{8}$$
$$\dot{n}_C = \dot{n}_{CO} + \dot{n}_{CO_2} + \dot{n}_{CH_4} + \dot{n}_{H_2}$$

WGS reactors are typically operated with S/dry > 1 to ensure complete CO conversion, maintain stability during load changes, and suppress side reactions (methane and solid carbon formation). Based on technical specification data, a set point of S/dry $= 1.8$ is adopted for the control loop that modulates the steam required.

Table 2. High-pressure purification calibration summary.

Component	*Trace*	A_{red}	E_a	Δ
Halide beds	HCl	1	7510	97.6 %
Hydrolysis reactor	COS	1	15922	86.7 %
Hydrolysis reactor	HCN	1	15470	89.8 %
Mercury bed	Hg	1	6750	90.0 %
H_2S removal	H_2S	1	6335	99.90 %
ZnO desulph. beds	COS	1	1000	100.0 %
ZnO desulph. beds	H_2S	1	1000	100.0 %

5.2 TSA/PSA Purification Items

Syngas produced by thermo-chemical gasification of carbonaceous feedstocks (coal, biomass, or waste) has a raw composition that varies widely with the gasification agent and reactor type, typically containing 25–55 mol% H_2, 30–60 mol% CO, 5–20 mol% CO_2, up to 15 mol% CH_4, and minor amounts of N_2 and H_2O (Natesakhawat et al. 2025). After WGS, the resulting shifted syngas – now predominantly a binary H_2/CO_2 mixture with residual CH_4, CO, N_2 and H_2O impurities – must be purified to meet end-use specifications: 99.99 mol% H_2 for fuel-cell-grade hydrogen (Sircar and Golden 2000). This is a separation step for which Pressure Swing Adsorption (PSA), and for thermal-regeneration applications Temperature Swing Adsorption (TSA), are the established industrial technologies. Yang et al. demonstrated experimentally and theoretically that a two-bed, six-step PSA process using zeolite 5A, with the pressure cycled between 1 and 11 atm

DOI
10.3384/ecp223283

at ambient temperature, can concentrate a 70 mol% H_2 feed to 99.99 % product purity with a recovery of 75.87 % (H_2/CO binary) and 80.38 % (H_2/CH_4 binary) (J. Yang et al. 1997). Adsorption equilibrium in such processes is commonly described by the Loading Ratio Correlation (LRC) multicomponent isotherm (Doong and R. T. Yang 1986), which captures competitive adsorption through a shared denominator, while intraparticle mass transfer is approximated by the Linear Driving Force (LDF) model (Sircar and Hufton 2000). Both the LRC/LDF PSA model and its Langmuir-based TSA counterpart accurately reproduce column dynamics and product purity; however, their spatially discretised implementations in process simulators such as Aspen Adsorption or gPROMS generate on the order of 10^3 equations per bed, preventing direct embedding of the adsorption unit into a full-plant dynamic model as a single system of equations (Son et al. 2018). For system-level studies that prioritise cycle dynamics, thermal loads and energy integration rather than sub-percent spatial composition detail, a lumped well-mixed kinetic model offers a computationally tractable alternative and is sufficient to verify the control strategy and the Finite State Machine-based cycle sequencing (Vareda 2023).

In this work, each adsorption vessel is represented as a well-mixed fluid volume of fixed geometry, obtained by extending the `MultiportVolume` base class of the Modelon library, which solves the dynamic energy and species mass balances for a gas mixture of n_S components tracked by mass fractions (Modelon AB 2026). A kinetic solid–gas adsorption layer is superimposed on this base class and parameterised for zeolite 5A pellets. The model rests on four assumptions: (i) the gas phase is spatially homogeneous within each vessel; (ii) the vessel volume is constant; (iii) kinetic energy is negligible in the energy balance; and (iv) the sorbent occupies the same control volume as the gas phase, its mass represented implicitly through the per-species maximum capacity Cap_0. Hydrogen is the non-adsorbing product species ($K_{ads,H_2} = 0$). In contrast to the LRC isotherm, which accounts for competitive adsorption through a shared denominator, the present formulation uses a single-component van't Hoff equilibrium for each species (Grande 2012):

$$p_{eq,j} = A_{VH,j} \exp\left(\frac{\Delta H_{ads,j}}{RT}\right) \tag{9}$$

where $\Delta H_{ads,j} = -Q_{st,j}$ is the adsorption enthalpy and $A_{VH,j}$ is calibrated at $T_{ref} = 300$ K. Selectivity is therefore encoded parametrically through the magnitude of $Q_{st,j}$: species with large isosteric heats, such as CO_2 and H_2O, exhibit low equilibrium pressures and adsorb preferentially relative to species with smaller heats, such as N_2 and H_2.

Unlike the LDF model, which is linear in the solid-phase loading departure from equilibrium, the present rate law acts on the gas-phase partial-pressure driving force and distinguishes between an adsorption branch and a desorption branch. The two branches are blended continuously over a pressure band of half-width Δp_{lim} to eliminate the discrete zero-crossing event that caused numerical chattering in the original hard-switching formulation. In the PSA model, the net mass sorption rate $\dot{m}_{rate,j}$ [kg s^{-1}] couples an adsorption branch $\dot{m}_{ads,j}$ with a desorption branch $\dot{m}_{des,j}$ through the Modelica `spliceFunction`, which transitions continuously over a half-band of width Δp_{lim} (0.0005 bar) centred on the switching threshold p_{lim}:

$$\dot{m}_{rate,j} = \texttt{spliceFunction}(\dot{m}_{ads,j}, \dot{m}_{des,j}, p - p_{lim}, \Delta p_{lim}) \tag{10}$$

while in the TSA variant the same structure is preserved but the switching variable is replaced by the vessel temperature T [K], with threshold T_{lim} (343.15 K) and half-band ΔT_{lim}:

$$\dot{m}_{rate,j} = \texttt{spliceFunction}(\dot{m}_{ads,j}, \dot{m}_{des,j}, T - T_{lim}, \Delta T_{lim}) \tag{11}$$

where $\dot{m}_{ads,j}$ and $\dot{m}_{des,j}$ are the adsorption and desorption rates, calculated as follows:

$$\dot{m}_{ads,j} = k_{ads,j} \frac{Cap_{0,j} - Cap_j}{Cap_{0,j}} DF_{ads,j} \tag{12}$$

$$\dot{m}_{des,j} = k_{des,j} \frac{Cap_j}{Cap_{0,j}} DF_{des,j} \tag{13}$$

The capacity factors $(Cap_{0,j} - Cap_j)/Cap_{0,j}$ and $Cap_j/Cap_{0,j}$ ensure that the adsorption rate vanishes as the bed saturates and the desorption rate vanishes as the bed empties, preventing integrator wind-up past the physical bounds without explicit saturation guards that would otherwise introduce discontinuities. $Cap_{0,j}$ is the maximum storage capacity per species, while $k_{ads,j}$ and $k_{des,j}$ [s^{-1}] are first-order rate coefficients. $DF_{ads,j}$ and $DF_{des,j}$ are the driving forces for the different species, defined as functions of Δp_j, the difference between the partial pressure and the equilibrium pressure of the specific species (Cruz et al. 2003; Pigorini and LeVan 1997; Krauklis 2015). In scientific literature, driving force contributions assume different mathematical forms. Chou and Xu (Chou and Xu 2007) identified the main $DF(\Delta p_j)$ functions, summarised in Table 3 (Fernández and Sánchez 2002; Merabti et al. 2025; Ron 1999).

The choice of driving force (DF) expression determines the qualitative shape of the sorption rate curve and must be consistent with the dominant mass-transfer mechanism governing adsorption and desorption at the vessel scale. In this model, for the adsorption branch, a Hill-type expression is adopted rather than the plain linear relation, while for the desorption branch a linear form is retained:

$$DF_{ads,j} = \frac{[\max(0, \Delta p_j)]^{n_{Hill}}}{(dp_{half})^{n_{Hill}} + (\Delta p_j)^{n_{Hill}}} \tag{14}$$

$$DF_{des,j} = \max(0, \Delta p_j) \tag{15}$$

Table 3. Driving force relations from literature data (Fernández and Sánchez 2002; Merabti et al. 2025; Ron 1999).

Mechanism	*Adsorption*	*Desorption*
Linear	$P - P_{eq}$	$P_{eq} - P$
Parabolic	$\sqrt{P} - \sqrt{P_{eq}}$	$\sqrt{P_{eq}} - \sqrt{P}$
Normalized 1	$\frac{P}{P_{eq}} - 1$	$1 - \frac{P}{P_{eq}}$
Normalized 2	$1 - \sqrt{\frac{P_{eq}}{P}}$	$1 - \sqrt{\frac{P}{P_{eq}}}$
Chemical Reaction	$\ln\left(\frac{P}{P_{eq}}\right)$	$\ln\left(\frac{P_{eq}}{P}\right)$

where $\Delta p_j = pp_j - p_{eq,j}$ is the driving force (Pa) and dp_{half} [Pa] is the half-saturation pressure. This form is preferred over the linear relation for two distinct reasons. First, it incorporates an intrinsic saturation: as $\Delta p_j \to \infty$ the expression tends to unity regardless of the kinetic coefficient, so the rate cannot grow unboundedly even before the capacity factor $(Cap_{0,j} - Cap_j)/Cap_{0,j}$ becomes active. Second, for $n_{Hill} > 1$ the expression is sigmoidal, with an inflection near $\Delta p_j = dp_{half}$, reproducing the cooperative adsorption character of zeolite 5A: the Sips (Langmuir–Freundlich) isotherm – whose equilibrium structure $q = q_m(bp)^n/(1+(bp)^n)$ is mathematically identical to the Hill expression at the saturation limit – consistently outperforms the simple Langmuir model for CO_2 and CH_4 on faujasite-type zeolite pellets (Streb and Mazzotti 2021). The exponent n_{Hill} in the present model acts on the driving force $\Delta p_j = pp_j - p_{eq,j}$ and is set equal to 3. The sorbent loading Cap_j for each species then evolves as a first-order ODE driven by the net sorption rate:

$$\frac{dCap_j}{dt} = \dot{m}_{rate,j} \tag{16}$$

In this study, the TSA is the purification unit for the compressed syngas stream and is placed after the H_2S removal package, while the PSA is set after the WGS, as shown in Figure 1. Two parallel vessels (VT1, VT2) are packed with activated carbon adsorbent. The unit operates between an adsorption temperature $T_{ads} = 40°C$ and a desorption temperature $T_{des} = 270°C$, the two vessels alternating between the two conditions so that one bed is always available for treatment. The two-vessel configuration for the TSA is presented in Figure 5. A communication buffer is added, and the different valves dedicated to filling, purge and equilibration are controlled via ON/OFF controllers.

The alternation between adsorption and desorption in VT1 and VT2 is governed by a double StateGraph state machine, presented in Figure 6. Moreover, two PID controllers regulate the vessel temperatures: one targets 40°C (adsorption) and one targets 270°C (desorption). The respective temperature set points are updated when the ABS and DES operating states of the state machine, related to adsorption and desorption, are active. A buffer set point of 80°C is maintained between active phases. The `multiSwitch` component updates the PID set points dynamically from the active state.

Table 4 summarises the seven states of a single-vessel TSA cycle, together with the transition condition that triggers each step and the physical action performed.

Table 4. TSA state machine cycle step sequence for a single vessel.

State	*Transition condition*	*Action*
filling	$p_{vessel} > p_{out}$ (13.6 bar)	Inlet valve opens; syngas enters from the bottom at $T_{setdown} = 313$ K; impurities adsorb; H_2-rich gas exits the top valve to the product header.
ABS	250 s	Inlet closes; static adsorption soak; PID holds the bed at $T_{setdown}$; Cap_j grows toward capacity.
wait-cold	600 s & $T < T_{setdown} + \Delta T$	PID continues cooling; a minimum cold dwell prevents premature equalisation before the bed reaches $T_{setdown} + \Delta T$.
equilibration	250 s	Upper equalisation valve opens; the cold vessel exchanges gas co-currently with the hot partner; partial pressure recovery.
DES	600 s & $T > T_{setup} - \Delta T$	PID heats the bed to $T_{setup} = 543$ K; thermal desorption releases adsorbed impurities; desorbed gas exits the top valve, triggering blowdown in the adjacent vessel.
wait-hot	250 s	Synchronisation hold at peak temperature; top valve remains open; phase-alignment buffer before purge.
purge	250 s	Bottom purge valve opens; counter-current thermal desorption drives Cap_j toward zero; the regenerated bed is ready for the next filling step.

For the PSA section, a four-vessel configuration, presented in Figure 7, was considered. Each vessel operates five valves: (A) Feed/Pressurisation, (B) Purge/Dump via vacuum sink, (C) Pressure equalisation via buffer volume, (D) H_2 production via check valve (opens at $p > 9.1$ barg), (E) Repressurisation with pure H_2 back-fill.

Each bed is governed by an independent Modelica

DOI
10.3384/ecp223283

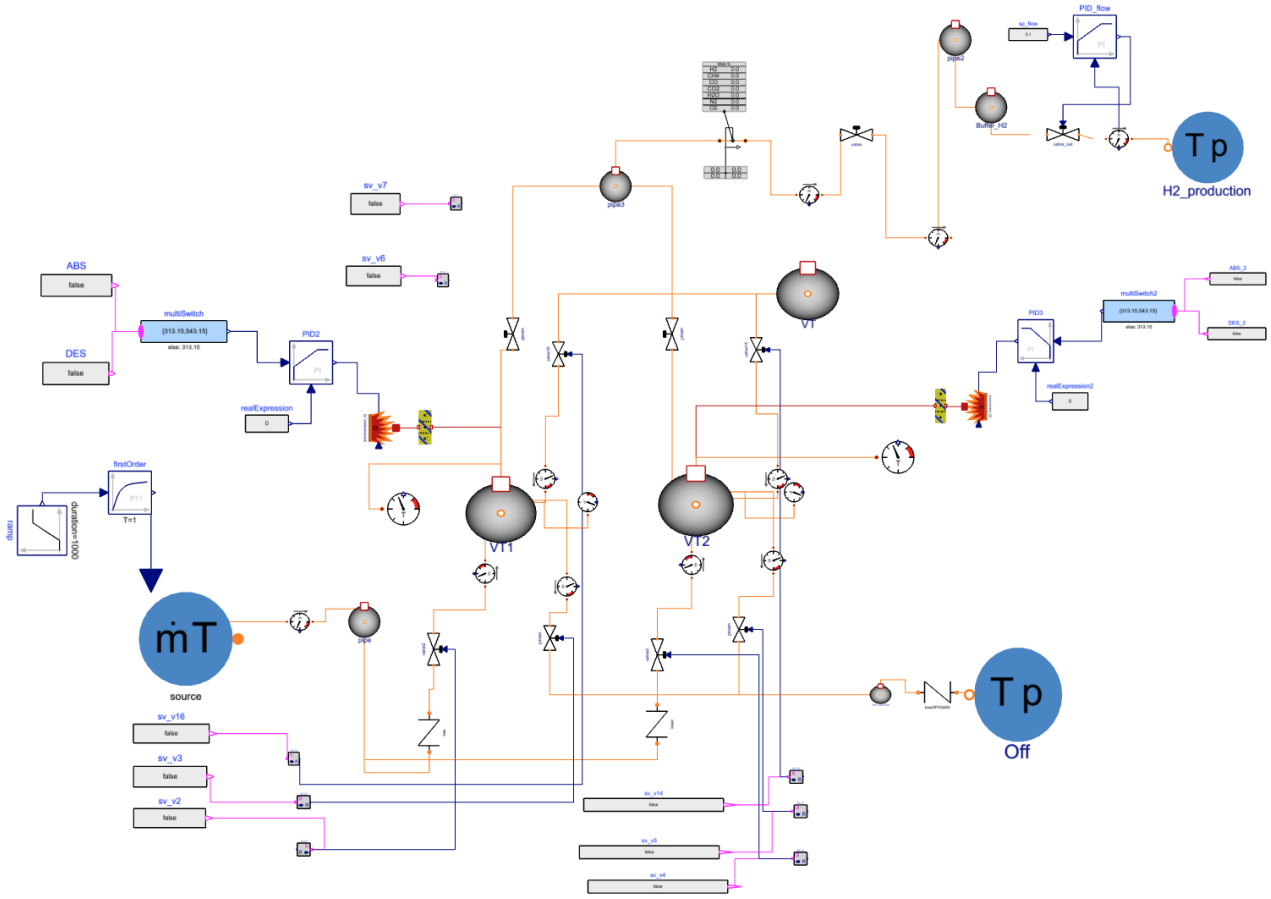


Figure 5. TSA model section.

StateGraph. The four state machines, presented in Figure 8, are structurally identical but operate in staggered phase, coordinated by cross-referenced transition conditions.

Table 5 summarises the eight states of a single-vessel cycle, together with the transition condition that triggers each step and the physical action performed. Duration is either time-based (T_{abs}, a fixed adsorption period) or pressure-based, where the transition fires as soon as the monitored vessel pressure crosses the stated threshold. The *wait* and *wait_align* states carry no process action; their purpose is to maintain phase coherence across the four state machines by absorbing timing mismatches that would otherwise accumulate over successive cycles.

6 Results

This section presents and discusses simulation results for selected subcomponents of the integrated Waste-to-H_2 purification system, with the aim of verifying the dynamic consistency of the most significant process variables under both transient and quasi-periodic steady-state conditions. Figure 9 shows the temperature trends of the two TSA vessels over a 12 000 s simulation. The 40°C adsorption set point is reached in ∼250 s; the 270°C desorption target in ∼750 s. The longer heating transient reflects the greater thermal mass and heat duty. The 80°C buffer temperature is correctly maintained between phases, confirming stable state machine control throughout all transitions.

The analysis focuses in particular on the behaviour of the dual-bed TSA subsystem and its interaction with the downstream WGS reactor, examining the mass balance and thermal energy consistency across the switching phases of the adsorption cycle. Figure 10 shows the cyclic pressure profiles. After alignment, each vessel reaches its maximum pressure consecutively, providing a quasi-continuous H_2 product stream. A vessel pressure of 3 bar separates the adsorption and desorption regimes during the purge step.

An alignment phase synchronises the four vessels at simulation start. Transition to the working cycle triggers when the upstream pressure reaches 9.5 barg. Figure 11 reports the time-domain evolution of the principal process variables over 12 000 s of simulation, from cold start through the establishment of quasi-periodic steady operation. The three panels show, respectively, the TSA inlet and outlet mass flow rates, the instantaneous thermal power of the two adsorption vessels, and the syngas and steam feed flows at the WGS reactor inlet.

Following an initial ramp of approximately 1500 s, the TSA inlet mass flow stabilises near 8 kg/s, with the out-

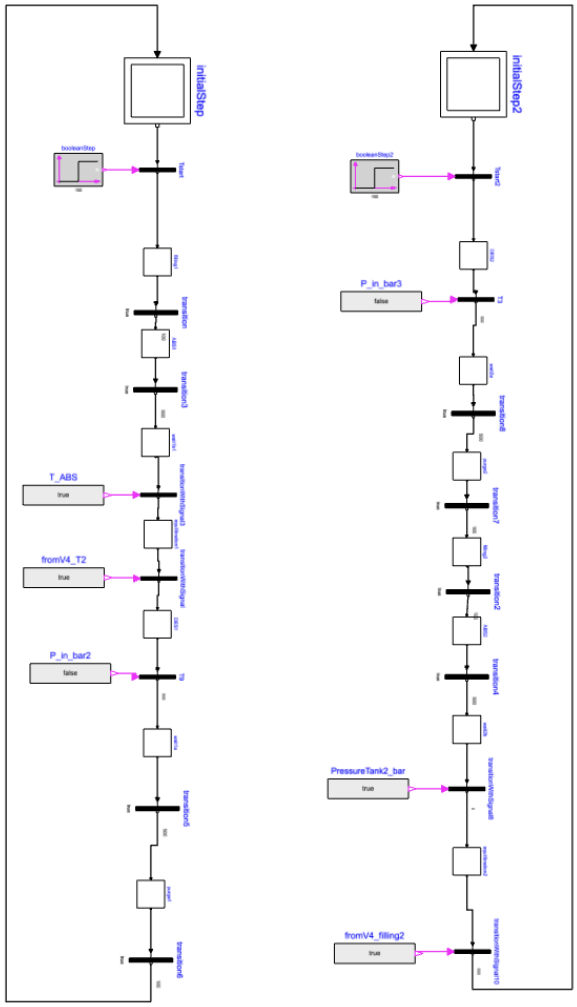


Figure 6. StateGraph state machine for the TSA cycle (ADS / DES phases).

Table 5. PSA state machine cycle step sequence for a single vessel.

State	*Transition condition*	*Action*
filling	300 s	Inlet valve opens; impurities adsorb along the bed. H_2-rich gas exits through the top check valve to the product header.
rinse	30 s	Feed inlet remains open, displacing the residual impurity mass-transfer front toward the bed bottom before equalisation, preventing a purity spike at the moment of co-current pressure exchange.
equilibration	T_{abs}	Inlet closes; the high-pressure bed exchanges gas co-currently with a low-pressure bed through the upper equalisation valves.
blowdown	$p_{vessel} < 8$ bar	Outlet valve opens to the off-gas sink at 9000 Pa; desorption begins.
wait	triggers blowdown in the adjacent vessel	Short synchronisation hold across all four state machines.
purge	T_{abs}	Counter-current purge; desorption with k_{des} drives Cap_j toward zero, regenerating the bed.
repressurisation	$p_{vessel} > 3$ bar	Inlet reopens; feed repressurises the bed from the bottom.
wait_align	T_{abs}	Phase-alignment buffer absorbs inter-state-machine timing mismatch.

let tracking it closely throughout steady operation except for brief discontinuities recurring at each vessel-switching event. The thermal power profiles of VT1 and VT2 display the expected antisymmetric, phase-shifted alternation between adsorption and regeneration, with the active regeneration vessel drawing on the order of 1000–1200 kW while the adsorbing vessel dissipates modestly, confirming correct sequential interlocking of the two-bed cycle. Downstream, the WGS syngas inlet flow mirrors the TSA outlet closely, reflecting the tight hydraulic coupling between the two subsystems, while the steam feed controller maintains an approximately constant S/dry throughout the cycle, with only minor transient deviations during switching attributable to the finite bandwidth of the control loop.

A fundamental component for the high-pressure section of the BoP is the WGS reactor, whose purpose is to increase the amount of hydrogen in the syngas stream that then goes to the PSA. Table 6 presents a sensitivity analysis on the gas mixture and hydrogen mass flow rates and temperatures for the syngas and steam streams before the mixing point, and for the WGS reactor's inlet and outlet; the results show that an increasing amount of steam at the WGS reactor inlet increases the amount of hydrogen produced.

Finally, Figure 12 shows the dynamic response of the four-vessel PSA unit over a simulation run of 17 500 s, reporting the hydrogen product mass flow rate (top) and molar purity (bottom) at the outlet of the H_2 storage volume.

The system undergoes a start-up transient during which an initial flow overshoot of approximately 0.0495 kg/s is observed, with the product purity reaching its operating level only at $t \approx 1500$ s, once the first adsorption and regeneration cycle has been completed and the beds are loaded. Cyclic steady state is established after roughly six to seven cycles, after which the product flow stabilises around a mean of 0.040 kg/s with periodic oscillations of

DOI
10.3384/ecp223283

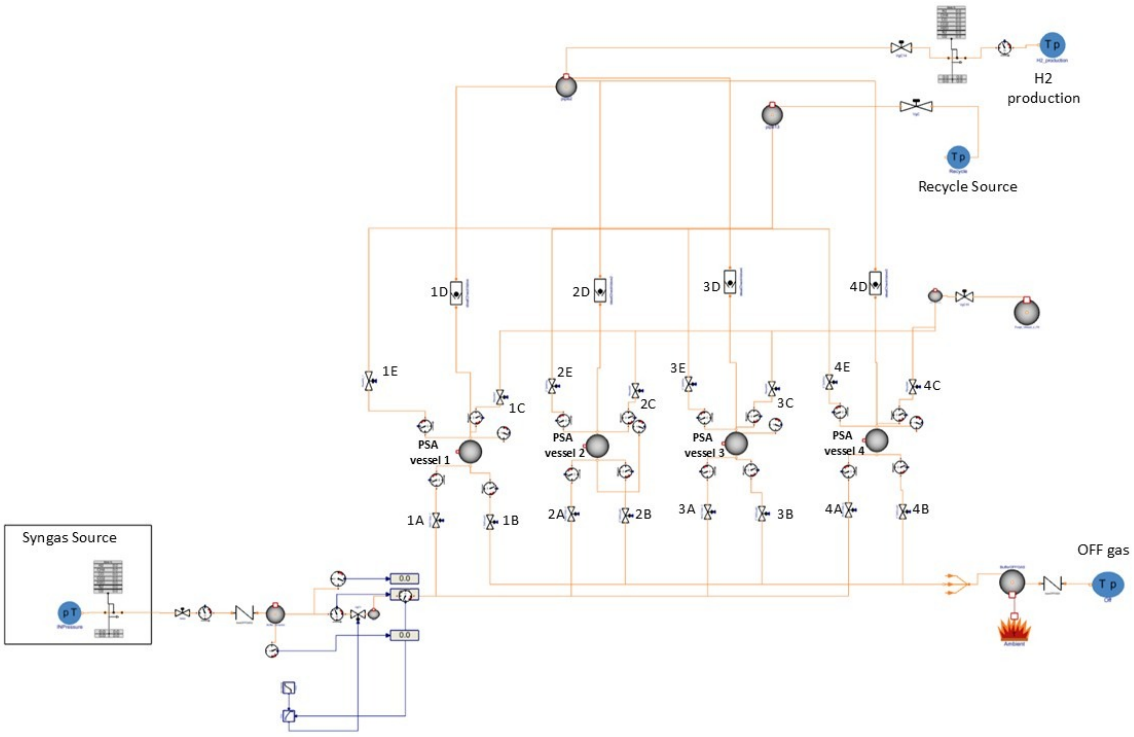


Figure 7. Four-vessel PSA model layout in Modelon Impact.

Table 6. S/dry sensitivity analysis.

S/dry		*Syngas*	*Steam*	*WGS in*	*WGS out*
1.7	T [°C]	197	394	324.8	466.2
	$\dot{m}_{syngas}$ [kg/h]	2925	4261.7	7186.7	7186.7
	$\dot{m}_{H_2}$ [kg/h]	117.0	–	117.1	**228.5**
1.85	T [°C]	197	394	328.6	464.3
	$\dot{m}_{syngas}$ [kg/h]	2925	4639.3	7564.3	7564.3
	$\dot{m}_{H_2}$ [kg/h]	117.0	–	117.2	**230.0**
2	T [°C]	197	394	328.6	464.3
	$\dot{m}_{syngas}$ [kg/h]	2925	5016.9	7941.9	7941.9
	$\dot{m}_{H_2}$ [kg/h]	117.0	–	116.7	**231.1**

$\pm 7.5\,\%$, inherent to the staggered four-bed scheduling at $T_{abs} = 300$ s. The nominal purity of 99.99 % is maintained throughout the adsorption steps, with brief reductions to approximately 99.5 % occurring at each vessel-switching instant because of the transient pressure equalisation across the inter-bed void volumes.

The isolated deeper excursion to approximately 98.9 % at $t \approx 14\,100$ s is due to the timing of the valve transitions and not to the adsorption process, with two adjacent vessels switching simultaneously when they reach their transition pressure at the same instant; the cyclic steady state is not affected. A minimum residence time in each state and a hysteresis band on the pressure conditions were introduced in the supervisory logic coordinating the four vessel state machines in order to prevent this, while a master supervisory logic extending the coordination to the whole BoP, and in particular to the interaction between the TSA and PSA sections, is identified as future work.

7 Conclusion

This paper presented a Modelica-based dynamic model of the complete Balance of Plant of a waste gasification system for hydrogen production, developed in the Modelon Impact environment. The work covered four main contributions: (i) the gasifier ROM construction and FMU integration; (ii) FMU sensitivity analysis; (iii) the low-pressure component chain for trace purification; and (iv) the TSA and PSA purification models.

The gasifier ROM was trained on static and dynamic CFD simulation data using a neural network with the Adam optimiser. It was then exported as an FMI 2.0-compliant FMU and integrated into Modelica via a dedicated adapter that generates trace species (HCl, H_2S, COS) from the feedstock chlorine and sulphur content. Sensitivity analysis of the FMU confirmed correct reproduction of gasification chemistry trends: increasing C/O ratio drives higher CO content and lower CO_2, while the C/H ratio governs the H_2/H_2O balance, validating the ROM across the full operating range explored.

All low-pressure and high-pressure BoP components

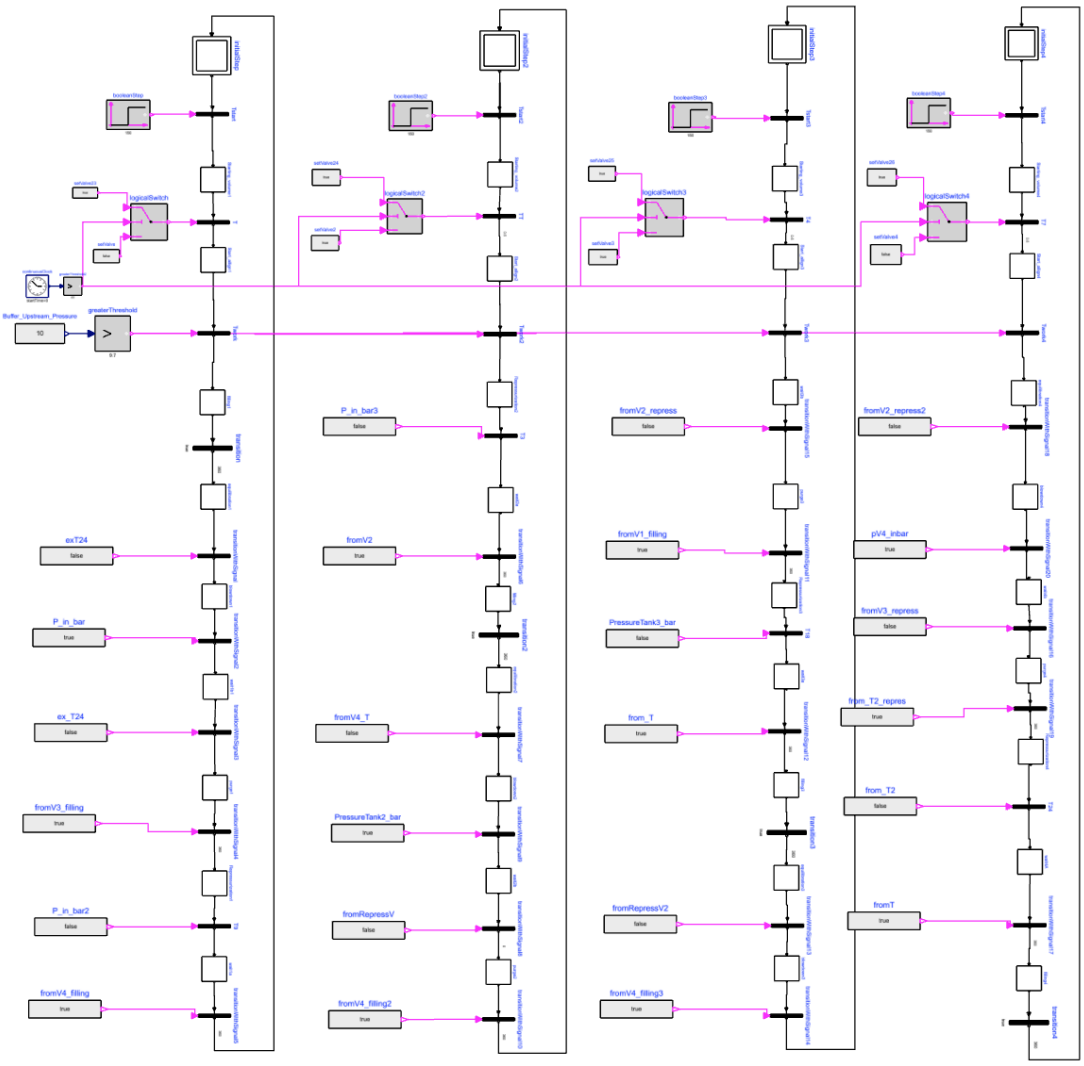


Figure 8. StateGraph state machine for the four-vessel PSA cycle.

were calibrated against H&M balance data, achieving minimal relative errors. The WGS reactor PI controller (with setpoint S/dry = 1.8) correctly modulates steam flow to maintain optimal conversion conditions at both full and partial load (80 %), confirming robust control performance. The TSA model implements a four-step thermal swing cycle on a two-vessel configuration with StateGraph control; PID controllers achieve the 40°C adsorption and 270°C desorption set points in approximately 250 s and 750 s, respectively. The PSA model implements a four-vessel, eight-step cycle producing hydrogen at 99.99 % purity, with the simulated product flow matching the H&M reference within 2 %. The model comprises three parallel gasifier lines, whose syngas streams are combined at the inlet of the alkaline scrubber, together with the shared downstream purification and hydrogen production sections. It therefore establishes the foundation for the next project phase, which concerns the advanced control of the gasification section rather than its assembly: the RDF and gasifying agent flow rates, as well as the C/H and C/O ratios of the feedstock, will be modulated in order to optimise hydrogen production.

The complete BoP was assembled in Modelon Impact (2.1.32), with the Modelica Standard Library 4.0.0 providing the control blocks and the StateGraph formalism, and the Modelon base libraries providing the hydraulic and thermal network. The modelling effort was concentrated on the components for which no library equivalent exists: the syngas medium including the trace species HCl, H_2S and COS; the adapter at the ROM interface; the components of the low-pressure purification chain, namely the alkaline scrubber and the acid gas removal units with their feed-specific reduction kinetics and the WGS reactor calibration; and the components of the high-pressure chain, namely the adsorption vessel employed in both the TSA and the PSA units, formulated with multicomponent isotherms and linear driving force kinetics, together with the associated cycle supervision logic. These components, rather than the plant assembly, represent the reusable modelling contribution of this work.

In terms of size, the coupled low- and high-pressure chains model comprises 60 equations after alias elim-

DOI
10.3384/ecp223283

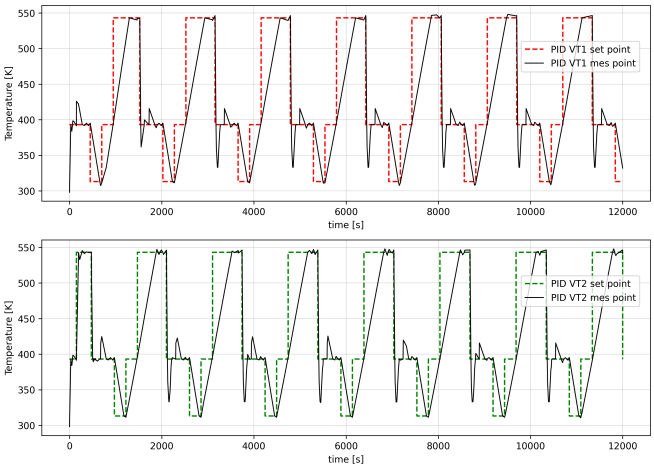


Figure 9. PID temperature trends for the two TSA vessels (12 000 s).

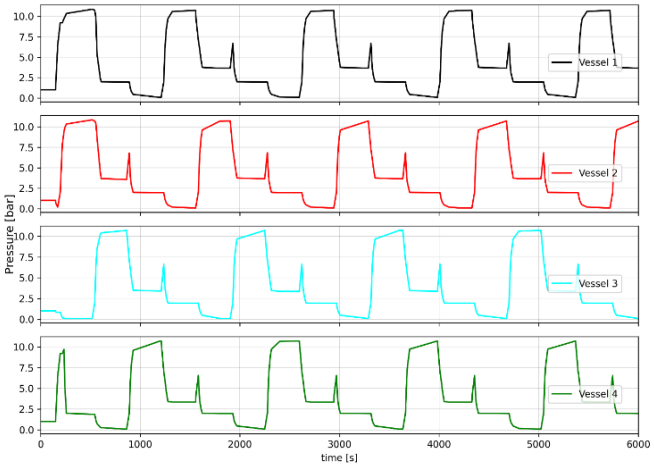


Figure 10. Time-pressure profiles of the four PSA vessels.

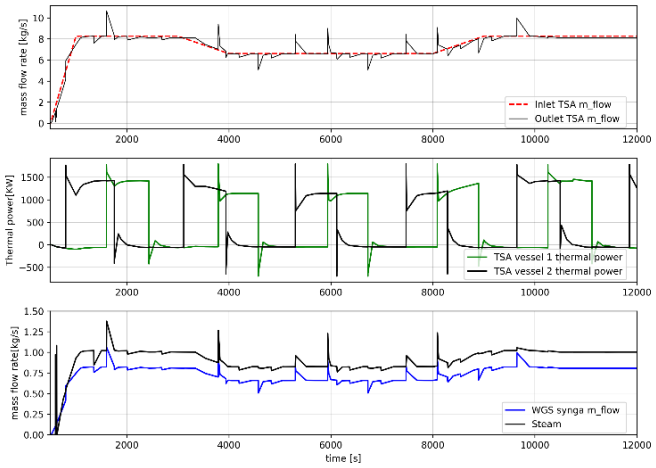


Figure 11. Simulation of the coupled TSA–WGS subsystem during the transition from nominal to partial load: TSA inlet and outlet mass flow rates (top), thermal power cycles of the two TSA vessels (centre), and the WGS syngas and steam feed mass flow rate (bottom).

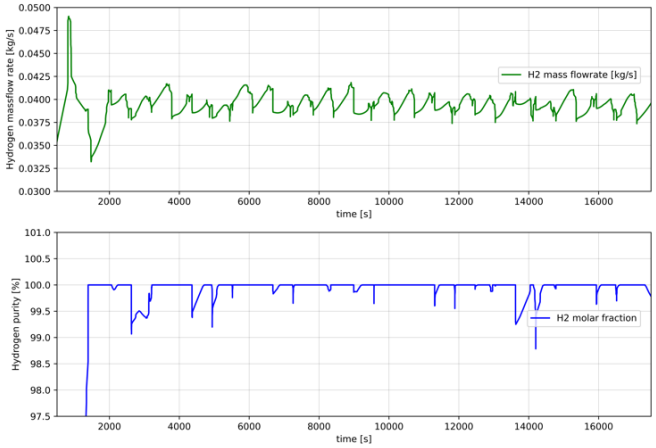


Figure 12. Dynamic response of the four-vessel PSA unit during a full simulation run of 17 500 s: hydrogen product mass flow rate (top) and hydrogen molar purity at the product stream (bottom).

ination and 78 continuous states, while the coupled TSA/WGS and PSA model comprises 160 equations after alias elimination and 221 continuous states. All simulations were performed with the CVode solver at a relative tolerance of 10^{-6}. The growth of the discrete variable count with the number of adsorption vessels is the quantity that governs the event handling effort, and it is therefore the principal constraint to be considered in the plant-level extension described below.

Future work will integrate three FMU instances at the alkaline scrubber inlet, where their syngas streams combine, enabling dynamic simulation of partial gasifier outages, load imbalances between lines, and coordinated control of the shared downstream purification and hydrogen production sections. Additional planned activities include the implementation of a comprehensive plant control strategy using nested state machines, calibration of trace reduction parameters at partial-load conditions once data become available, and model validation against experimental data from plant commissioning.

Acknowledgements

This project is part of the IPCEI on Hydrogen – Hy2Use, with NextChem as project owner, leveraging the proprietary technologies of its subsidiary MyRechemical to support the development of sustainable hydrogen value chains in Italy and Europe.

References

Abdoulmoumine, Nourredine, Sushil Adhikari, Avanti Kulkarni, and Shyamsundar Chattanathan (2015-10). “A review on biomass gasification syngas cleanup”. In: *Applied Energy* 155, pp. 294–307. DOI: 10.1016/j.apenergy.2015.05.095.

Arena, Umberto (2012-04). “Process and technological aspects of municipal solid waste gasification. A review”. In: *Waste Management* 32.4, pp. 625–639. DOI: 10.1016/j.wasman.2011.09.025.

Bashir, M. A. et al. (2025). “Plastic waste gasification for low-carbon hydrogen production: a comprehensive review”. In: *Energy Advances* 4.3, pp. 330–363. DOI: 10.1039/D4YA00292J.

Blasi, Colomba Di (2004-09). “Modeling wood gasification in a countercurrent fixed-bed reactor”. In: *AIChE Journal* 50.9, pp. 2306–2319. DOI: 10.1002/aic.10189.

Blochwitz, Torsten et al. (2012-11). “Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models”. In: *Proceedings of the 9th International Modelica Conference*. Munich, Germany, pp. 173–184. DOI: 10.3384/ecp12076173.

Chou, Kuo-Chih and Kai Xu (2007-05). “A new model for hydriding and dehydriding reactions in intermetallics”. In: *Intermetallics* 15.5, pp. 767–777. DOI: 10.1016/j.intermet.2006.10.004.

Cruz, P., J. C. Santos, F. D. Magalhães, and A. Mendes (2003-07). “Cyclic adsorption separation processes: analysis strategy and optimization procedure”. In: *Chemical Engineering Science* 58.14, pp. 3143–3158. DOI: 10.1016/S0009-2509(03)00189-1.

Doong, S.-J. and Ralph T. Yang (1986-03). “Bulk separation of multicomponent gas mixtures by pressure swing adsorption: Pore/surface diffusion and equilibrium models”. In: *AIChE Journal* 32.3, pp. 397–410. DOI: 10.1002/aic.690320306.

Fernández, J. F. and C. R. Sánchez (2002-06). “Rate determining step in the absorption and desorption of hydrogen by magnesium”. In: *Journal of Alloys and Compounds* 340.1, pp. 189–198. DOI: 10.1016/S0925-8388(02)00120-2.

Grande, Carlos A. (2012-12). “Advances in Pressure Swing Adsorption for Gas Separation”. In: *ISRN Chemical Engineering* 2012, pp. 1–13. DOI: 10.5402/2012/982934.

Kingma, Diederik P. and Jimmy Ba (2014). *Adam: A Method for Stochastic Optimization*. DOI: 10.48550/ARXIV.1412.6980. arXiv: 1412.6980.

Krauklis, A. E. (2015). *Modeling of an Isothermal Pressure Swing Adsorption Process*. DOI: 10.13140/RG.2.2.18296.19208.

Mattsson, Sven Erik and Hilding Elmqvist (1997-04). “Modelica - An International Effort to Design the Next Generation Modeling Language”. In: *IFAC Proceedings Volumes* 30.4, pp. 151–155. DOI: 10.1016/S1474-6670(17)43628-7.

Merabti, R., O. Merdoud, Y. Roumila, A. Boudjemaa, and M. Trari (2025-11). “Zeolite Y used as a Novel support for a coupled Cr- and Fe-oxides for Hydrogen Evolution Reaction under visible light”. In: *International Journal of Hydrogen Energy* 192, p. 152332. DOI: 10.1016/j.ijhydene.2025.152332.

Modelon AB (2026). *Modelon*. URL: https://www.modelon.com/ (visited on 2026-08-01).

Monaghan, Rory F. D. and Ahmed F. Ghoniem (2012-02). “Simulation of a Commercial-Scale Entrained Flow Gasifier Using a Dynamic Reduced Order Model”. In: *Energy & Fuels* 26.2, pp. 1089–1106. DOI: 10.1021/ef201551m.

Morales-Ospino, Rodrigo, Ramon G. Santiago, Rafael M. Siqueira, Diana C. S. De Azevedo, and Moises Bastos-Neto (2020-07). “Assessment of CO2 desorption from 13X zeolite for a prospective TSA process”. In: *Adsorption* 26.5, pp. 813–824. DOI: 10.1007/s10450-019-00192-5.

Mustapha, S. I. et al. (2025-07). “Biomass conversion for sustainable hydrogen generation: A comprehensive review”. In: *Fuel Processing Technology* 272, p. 108210. DOI: 10.1016/j.fuproc.2025.108210.

Natesakhawat, S. et al. (2025-09). “Hydrogen-rich syngas production from the steam co-gasification of low-density polyethylene and coal refuse”. In: *Fuel* 395, p. 135254. DOI: 10.1016/j.fuel.2025.135254.

Pigorini, G. and M. D. LeVan (1997-06). “Equilibrium Theory for Pressure Swing Adsorption. 3. Separation and Purification in Two-Component Adsorption”. In: *Industrial & Engineering Chemistry Research* 36.6, pp. 2306–2319. DOI: 10.1021/ie960716w.

Qiu, Kai, Jie Yang, Zhengbao Gao, and Fen Xu (2024-04). “A review of Modelica language in building and energy: Development, applications, and future prospect”. In: *Energy and Buildings* 308, p. 113998. DOI: 10.1016/j.enbuild.2024.113998.

Ron, M. (1999-02). “The normalized pressure dependence method for the evaluation of kinetic rates of metal hydride formation/decomposition”. In: *Journal of Alloys and Compounds* 283.1-2, pp. 178–191. DOI: 10.1016/S0925-8388(98)00859-7.

Sahraei, M. Hossein, M. A. Duchesne, R. Yandon, A. Majeski, R. W. Hughes, and L. A. Ricardez-Sandoval (2015-12). “Reduced order modeling of a short-residence time gasifier”. In: *Fuel* 161, pp. 222–232. DOI: 10.1016/j.fuel.2015.07.096.

Sircar, Shivaji and Timothy C. Golden (2000-01). “Purification of Hydrogen by Pressure Swing Adsorption”. In: *Separation Science and Technology* 35.5, pp. 667–687. DOI: 10.1081/SS-100100183.

Sircar, Shivaji and J. R. Hufton (2000-06). “Why Does the Linear Driving Force Model for Adsorption Kinetics Work?” In: *Adsorption* 6.2, pp. 137–147. DOI: 10.1023/A:1008965317983.

Son, K. N., G. E. Cmarik, J. C. Knox, J. A. Weibel, and S. V. Garimella (2018-05). “Measurement and Prediction of the Heat of Adsorption and Equilibrium Concentration of CO2 on Zeolite 13X”. In: *Journal of Chemical & Engineering Data* 63.5, pp. 1663–1674. DOI: 10.1021/acs.jced.8b00019.

Streb, Andres and Marco Mazzotti (2021-05). “Adsorption for efficient low carbon hydrogen production: part 1—adsorption equilibrium and breakthrough studies for H2/CO2/CH4 on zeolite 13X”. In: *Adsorption* 27.4, pp. 541–558. DOI: 10.1007/s10450-021-00306-y.

Tlili, N., G. Grévillot, and C. Vallières (2009-09). “Carbon dioxide capture and recovery by means of TSA and/or VSA”. In: *International Journal of Greenhouse Gas Control* 3.5, pp. 519–527. DOI: 10.1016/j.ijggc.2009.04.005.

Urbani, Michele, Michele Bolognese, L. Pratticò, and Matteo Testi (2025-10). “A Tool for the Implementation of Open Neural Network Exchange Models in Functional Mockup Units”. In: *Proceedings of the 16th International Modelica & FMI Conference*. Lucerne University of Applied Sciences and Arts (HSLU), Lucerne, Switzerland. DOI: 10.3384/ecp218645.

Vareda, João P. (2023-04). “On validity, physical meaning, mechanism insights and regression of adsorption kinetic models”. In: *Journal of Molecular Liquids* 376, p. 121416. DOI: 10.1016/j.molliq.2023.121416.

Yang, Jinshi, Chang-Ha Lee, and Jong-Wook Chang (1997-07). “Separation of Hydrogen Mixtures by a Two-Bed Pressure Swing Adsorption Process Using Zeolite 5A”. In: *Industrial & Engineering Chemistry Research* 36.7, pp. 2789–2798. DOI: 10.1021/ie960728h.

Development of an FMI-Oriented One-Dimensional Transient Heat Transfer Model for Separated Heat Pipes

Peng Ju[1], Xinchen Wang, Yujie Wang, Lei Peng, Xinxin Li[2*]

[1]Harbin Institute of Technology (Shenzhen), Shenzhen, China

[2]Chongqing University, Chongqing, China

Abstract

This paper develops an FMI-oriented one-dimensional transient thermal-hydraulic model for separated heat pipes used in passive residual heat-removal systems. The heat pipe is represented by modular control volumes for the evaporator, vapor riser, condenser, downcomer, wick, wall, vapor chamber, and external water tank. The model combines a nodal thermal-resistance network with a Knudsen-number startup criterion, vapor pressure-drop closure, Clausius-Clapeyron temperature-pressure coupling, phase-change source terms, and capillary/gravity-assisted return-flow closure. The resulting ordinary differential equations are solved using a three-stage Gauss-Legendre implicit Runge-Kutta method. To address system-level reuse, the physics, solver, parameters, inputs, outputs, and state-update routines are organized according to FMI Co-Simulation semantics, enabling later FMU wrapping without changing the core solver. Validation against small-scale and large-scale heat-pipe data shows temperature and flow-rate deviations of approximately 10% or less, supporting fast integration in Modelica-based accident-transient simulations.

Keywords: separated heat pipe; Functional Mock-up Interface (FMI); FMU-ready model; co-simulation; Modelica; passive residual heat removal

1 Introduction

Nuclear energy is an important component of a low-carbon energy system. The Fukushima nuclear accident showed that conventional safety systems that rely heavily on external power and active equipment can be vulnerable under extreme natural hazards and station blackout conditions. Therefore, passive safety technologies that can operate reliably for a long time without external power have become an important direction for improving the inherent safety of advanced reactors[1-4]. The core task of a passive residual heat removal system is to continuously remove decay heat after reactor shutdown, thereby preventing core overheating, loss of safety margin, and potential radioactive release.

Heat pipes transfer heat efficiently through evaporation and condensation of a working fluid. They are pump-free, highly isothermal, fast in response, and compact in structure, making them promising candidates for residual heat removal and heat-pipe-based cooling in advanced reactors[5-8]. Compared with an integral heat pipe, a separated heat pipe spatially separates the evaporator and condenser and forms a closed loop through a vapor riser and a liquid return line. It therefore offers flexible layout, long-distance heat transport, and strong adaptability to different hot- and cold-source arrangements. Previous studies have indicated that the vapor and condensate in separated heat pipes can be organized in separate flow passages, reducing the possibility of entrainment limitation and making the configuration suitable for pool-type reactors, spent fuel pools, and passive containment cooling systems[9-11].

The engineering performance of a separated heat pipe depends not only on the overall heat-transfer temperature difference, but also on the filling ratio, vacuum degree, non-condensable gases, hot- and cold-source boundary conditions, and two-phase flow regime in the evaporator. Experiments on separated heat pipes for pool-type reactors have shown that the start-up process can be divided into start-up, transition, and stable-operation stages. The filling ratio and vacuum degree are key factors affecting the start-up time and start-up temperature; increasing the pool-water temperature and air velocity, or decreasing the air temperature, generally shortens the start-up time and enhances the heat removal capacity[11]. Studies on header-type separated gravity heat pipes further show that bubbly flow, slug flow, and churn flow may occur in the parallel evaporator branches. When the evaporator is inclined, bubbles tend to accumulate near the upper wall and may induce local heat-transfer deterioration. The flow distribution among parallel branches is strongly affected by the header-to-branch diameter ratio, inlet subcooling, inlet mass flux, pool temperature, and number of branches[12].

These studies provide an important basis for applying separated heat pipes to nuclear residual heat removal. However, fast system-level calculation for reactor accident transients and engineering parameter scans remains insufficient. On the one hand, conventional system-level codes usually cannot directly represent the start-up temperature front, rarefied-vapor-to-continuum transition, phase-change heat transfer, and return-flow recovery inside a separated heat pipe. On the other hand, high-fidelity CFD models can resolve local flow regimes and phase-interface behavior, but their computational cost is too high for long accident sequences, multi-parameter sensitivity analysis, and online calls from system-level programs. Previous one-dimensional transient models for loop heat pipes and two-phase loop thermosyphons have demonstrated that reduced-order thermal-hydraulic formulations can reproduce the main dynamic response with far lower computational cost than CFD[13,14].

While reduced-order models offer computational speed, their value for system-level engineering also depends on whether they can be reused across simulation tools. Many nuclear thermal-hydraulic models are implemented in proprietary or monolithic environments, which makes it difficult to couple newly developed component models with reactor-core, containment, balance-of-plant, or control-system models. The Functional Mock-up Interface (FMI), developed under the Modelica Association, provides a standardized route for model exchange and co-simulation across tools such as Dymola, OpenModelica, Simulink, and other FMI-supported platforms[15,16]. For passive residual heat-removal analysis, an FMI-oriented separated heat-pipe module would allow the component to be connected to larger Modelica-based system models while preserving a dedicated fast solver for startup and phase-change dynamics. However, available FMI-oriented libraries still lack a reduced-order separated heat-pipe component that simultaneously accounts for rarefied-vapor startup, phase-change heat transport, pressure-drop coupling, and capillary/gravity-assisted liquid return.

Therefore, the objective of this work is to formulate a fast separated-heat-pipe model whose internal organization is compatible with FMI-based co-simulation rather than to provide only a standalone MATLAB calculation program. The physical equations are separated from the numerical time-advancement routine, and the model is expressed through explicit parameter, input, output, state, initialization, and step-update interfaces. This organization follows the information flow of an FMU for Co-Simulation: a master algorithm supplies boundary conditions and a communication step size, the heat-pipe solver advances its internal states, and heat-removal rates and state variables are returned to the system model. The main contributions are: (1) a one-dimensional nodal thermal-resistance network for fast separated-heat-pipe calculation; (2) startup and phase-change closures covering Knudsen-number transition, Clausius-Clapeyron coupling, vapor pressure drop, and capillary/gravity-assisted return flow; (3) an implicit Runge-Kutta solver suitable for stiff startup transients; (4) validation using small-scale and large-scale heat-pipe cases; and (5) an FMI-oriented interface definition that prepares the model for integration into Modelica-based digital twins of passive residual heat-removal systems.

2 Physical System and One-Dimensional Model

2.1 Separated Heat Pipe and Modeling Scope

The object of this study is a separated heat pipe for passive residual heat removal. The basic configuration consists of an evaporator, a vapor riser, a condenser, and a downcomer, as shown in Fig. 1. The evaporator absorbs heat from the hot source, causing the liquid working fluid in the wick or evaporator section to heat up and evaporate. Driven by pressure and density differences, the vapor flows upward through the riser to the condenser, where it releases latent heat and condenses. The condensate then returns to the evaporator through the downcomer under the combined action of gravity assistance and capillary pressure, completing a closed loop. This configuration inherits the separated-heat-pipe feature of spatially separated evaporator and condenser sections for long-distance phase-change heat transport, while retaining the wick-assisted capillary return mechanism. It can therefore be regarded as an engineering configuration combining features of separated heat pipes and loop heat pipes.

The model equations are organized so that the continuous-time heat-transfer and fluid-flow physics are separated from the numerical step-update routine. This separation is consistent with Modelica's equation-oriented modeling philosophy and with FMI Co-Simulation, in which the heat-pipe component can be advanced as an independent slave model under a system-level master algorithm.

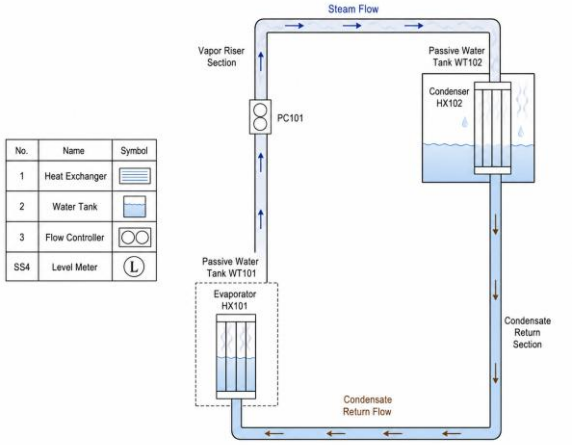


Fig. 1. Configuration of the separated heat pipe and one-dimensional nodal modeling object

To balance the dominant physical processes with engineering computational efficiency, the heat pipe is axially divided into four functional sections. In the radial direction of the evaporator and condenser, the vapor chamber, wick, wall, and external water tank are represented by separate control volumes. The model does not aim to resolve local two-dimensional flow-regime details; instead, it focuses on the effects of start-up temperature-front propagation, evaporation/condensation phase change, vapor pressure drop, and liquid return on the transient system response.

2.2 Nodal Thermal-Resistance Network and Energy Equation

The model adopts a hot-zone/cold-zone partitioning concept and a nodal thermal-resistance network. For each control volume, a temperature equation is established from energy conservation. Heat is transferred among the wall, wick, vapor chamber, and external water tank through conduction, convection, and phase-change heat transfer. For any control volume i, the energy equation can be written as

$$C_i \, dT_i/dt = \Sigma_j \, (T_j - T_i)/R_{ij} + Q_i, \quad (1)$$

DOI
10.3384/ecp223295

where C_i is the equivalent heat capacity of the control volume, T_i is the nodal temperature, R_{ij} is the equivalent thermal resistance between adjacent nodes, and Q_i is the external heat source or phase-change source term. The evaporation and condensation mass flow rates are determined from the interfacial heat flux and latent heat, and the local vapor flow rate is obtained by accumulating the axial evaporation and condensation rates. To describe the propagation of the start-up temperature front, the model assumes that effective heat transport in the heat pipe is mainly carried by vapor evaporation and condensation. Significant heat transport through the evaporation-condensation mechanism is activated only when the vapor temperature reaches the flow-transition temperature. The gas-liquid interface is assumed to be in thermodynamic equilibrium; that is, the interfacial temperature is equal to the local saturated vapor temperature. The one-dimensional nodal thermal-resistance network can be found in Fig. 2.

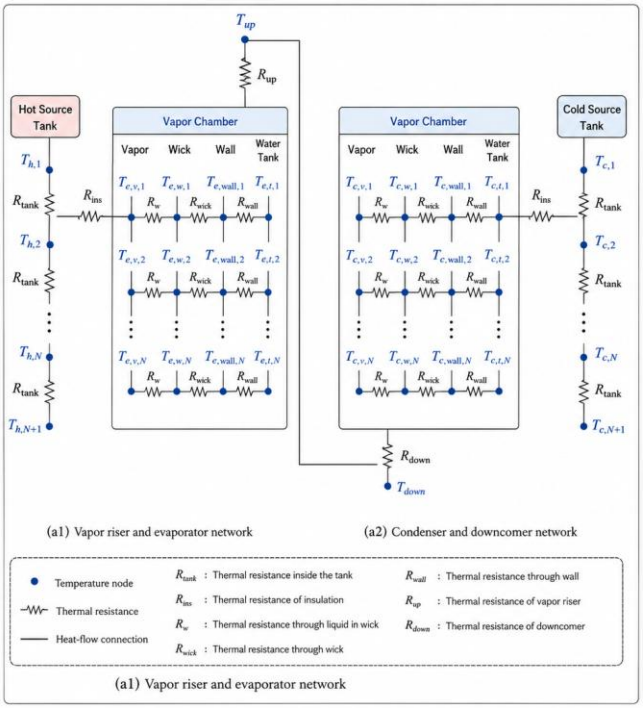


Fig. 2. One-dimensional nodal thermal-resistance network

2.3 Flow Transition, Pressure Drop, and Phase-Change Closure

At the early start-up stage, the vapor density is low, and the local vapor may be in a free-molecular or transition-flow regime. The model uses the Knudsen number to describe the flow state:

$$Kn = \lambda / D, \quad (2)$$

where λ is the mean free path of vapor molecules and D is the characteristic diameter. When Kn decreases below the continuum-flow criterion, the vapor flow can be treated as a continuum. Combining saturated-vapor properties with the pipe diameter gives the transition temperature from free-molecular flow to continuum flow, which is used as the hot-zone boundary temperature. The vapor pressure drop in the hot zone is calculated using the incompressible laminar flow pressure-drop model for circular tubes and is coupled with the Clausius-Clapeyron relation to obtain the axial vapor temperature distribution. The interfacial evaporation or condensation heat flux is determined by the temperature difference between the wick and vapor chamber, the effective interfacial thermal resistance, and the heat-transfer area. Liquid return is controlled by capillary pressure, gravity, and flow resistance. After full start-up and at steady state, the return mass flow rate approaches the total evaporation mass flow rate.

2.4 MATLAB Implementation and Fast Calculation Workflow

The overall model can be expressed as a coupled system of ordinary differential equations:

$$dY/dt = f(t, Y), \quad (3)$$

where the state vector Y includes nodal temperatures, local vapor pressures, vapor densities, evaporation/condensation rates, vapor-chamber flow rates, and liquid return flow rate. These variables are retained as internal states or algebraic variables of the heat-pipe component, while external boundary conditions are supplied through an explicit input vector.

To improve numerical stability during strongly coupled and nonlinear startup transients, a three-stage Gauss-Legendre implicit Runge-Kutta algorithm is used. For an FMI Co-Simulation wrapper, this routine corresponds to the internal advance performed during each communication step. The external master algorithm only exchanges boundary inputs and output variables at communication points, while the heat-pipe solver may use smaller internal nonlinear iterations to satisfy the coupled phase-change and pressure-drop equations. Compared with three-dimensional CFD, the present implementation sacrifices local flow-regime resolution in exchange for high computational speed and robustness in engineering parameter scans. The workflow of this MATLAB fast calculation is shown in Fig. 3.

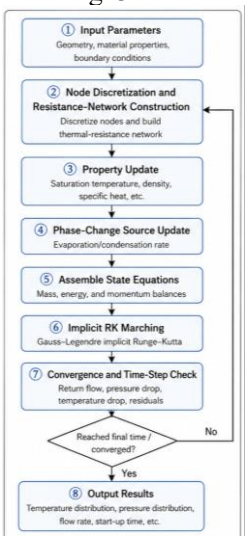


Fig. 3. MATLAB fast calculation workflow

3 Program Validation and Result Analysis

3.1 FMI-Oriented Model Architecture and Co-Simulation Interface

To make the heat-pipe model relevant to Modelica and FMI applications, the MATLAB prototype is structured as an FMU-ready Co-Simulation component. The component boundary is defined at the level of system variables rather than internal implementation details. A future Modelica or FMI-supported host model can therefore provide hot-source and condenser boundary conditions, call the heat-pipe solver over a communication step, and receive heat-removal and thermal-state outputs without resolving the internal phase-change closures.

The proposed FMI interface contains four groups of variables. First, fixed parameters describe the geometry, material properties, wick characteristics, filling condition, and initial vacuum condition. Second, time-varying inputs describe the hot-side boundary, condenser-side boundary, ambient or coolant condition, and optional system pressure constraints. Third, internal states store temperatures, vapor pressure, vapor density, and flow-rate variables advanced by the implicit solver. Fourth, outputs provide the quantities required by a system-level Modelica model, including heat-removal rate, evaporator/condenser temperatures, return-flow rate, and startup status. This variable grouping maps directly to FMI causality categories such as parameter, input, local, and output as shown in Table 1.

Table 1. Proposed FMI variable grouping for the separated heat-pipe module

FMI role	Representative variables	Use in co-simulation	Modelica-system connection
Parameter	Geometry, thermal properties, wick data, filling ratio, initial pressure	Set before initialization	Component record or parameter dialog
Input	Hot-source temperature/heat flux, condenser coolant temperature or HTC	Updated by the master at communication points	Connected to reactor, pool, containment, or cooling-loop models
Local state	Nodal temperatures, vapor pressure, density, phase-change rates	Advanced by the internal implicit Runge-Kutta solver	Hidden implementation variables of the FMU
Output	Heat-removal rate, outlet temperature, flow rate, startup flag	Returned to the master after each step	Used by system energy balances and safety margins

In this work, the interface definition is treated as part of the model contribution because it determines whether the reduced-order solver can be reused by Modelica/FMI users. The present manuscript reports the FMU-ready architecture and variable mapping; generation of a compiled FMU and cross-tool benchmark tests will be carried out as the next implementation step. This distinction avoids changing the physical solver while still aligning the model with the FMI-based workflow expected in system-level simulation.

3.2 Validation Cases and Main Parameters

The program is validated using two cases: a small stainless-steel/water loop heat pipe based on the parameters reported by Masahito Nishikawara et al. (2013)[13], and a large-scale standard loop heat pipe[9,10]. The small-scale case is used to examine the start-up temperature response and sectional heat-transfer behavior, whereas the standard case is used to evaluate the steady-state temperature, circulation flow rate, and energy balance under large-scale multi-tube conditions. Because the present model is positioned as a one-dimensional fast calculation tool, the validation focuses on three system-level indicators: temperature response, circulation flow rate, and energy balance. The main geometric and boundary parameters of the standard loop heat pipe are listed in Table 2.

Table 2. Main geometric and boundary parameters of the standard loop heat pipe

Parameter	Value
Total length	13434 mm
Section length (Evap/Riser/Cond/Downcomer)	1212/5500/1222/5500 mm
Parallel tubes	37
Evap./cond. OD/ID	57/50 mm
Vapor channel D	40 mm
Riser OD/ID	154/150 mm
Downcomer OD/ID	24/20 mm
Wick porosity	0.5
Condenser HTC	300 W/(m^2·K)

3.3 Start-Up Temperature Response Validation

The small-scale heat-pipe validation shows that the model can reproduce typical start-up features, including rapid temperature rise in the evaporator, temperature transition in the adiabatic section, and delayed response in the condenser as displayed in Fig. 4. The temperatures of the evaporator, adiabatic section, and condenser increase stepwise with time and eventually approach stable levels. The model predictions are consistent with the trends reported in the open literature and project validation data, with the main temperature deviations within approximately 10%. These results indicate that, after introducing the flow-transition temperature and nodal thermal-resistance network, the model can describe the hot-zone expansion and establishment of phase-change heat transport during start-up.

From the operating mechanism of separated heat pipes, the key to start-up is that the evaporator must generate

DOI 10.3384/ecp223295

sufficient saturated vapor and establish a stable condensation and return path at the condenser. The literature shows that an excessive filling ratio increases the thermal capacity of the heated liquid and prolongs the start-up time, whereas an insufficient filling ratio may cause liquid starvation or local dryout in the evaporator. A lower vacuum degree implies more non-condensable gas, which raises the start-up temperature and weakens condenser heat transfer[11]. Although the present model does not explicitly resolve the spatial distribution of non-condensable gas, it retains the dominant influence of start-up thermal-resistance variation on temperature response through the transition temperature, phase-change thermal resistance, and return-flow closure.

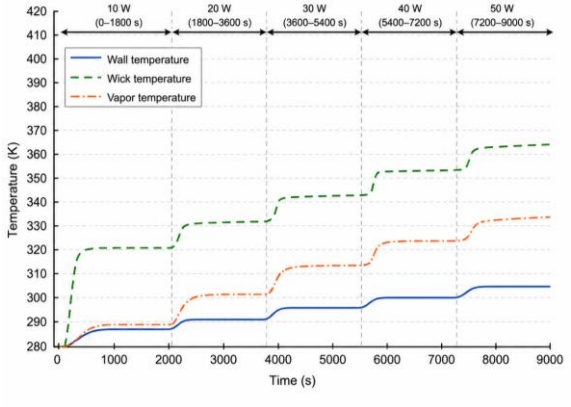


Fig. 4. Start-up temperature response validation for the small loop heat pipe

3.4 Steady-State Parameters and Energy-Balance Validation

The steady-state validation of the standard loop heat pipe shows that the evaporator outlet temperature error is less than 1%, the circulation flow-rate error is approximately 10%, and the maximum energy imbalance is approximately 10.9% as listed in Table 3 and Fig. 5. The transient temperature distribution shows clear sectional characteristics: a high-temperature plateau forms in the evaporator, a distinct axial temperature drop appears in the adiabatic section, and the condenser temperature remains lower and more uniform. These results demonstrate that the model can describe the coupled evaporation-driven, vapor-transport, and condensation-return processes during start-up and steady operation of a large-scale bent loop heat pipe.

Table 3. Summary of program validation results

Validation indicator	Result
Small-scale temp.	within about 10%
Evaporator outlet temp.	< 1%
Circulation flow rate	about 10%
Energy balance	max. 10.9%

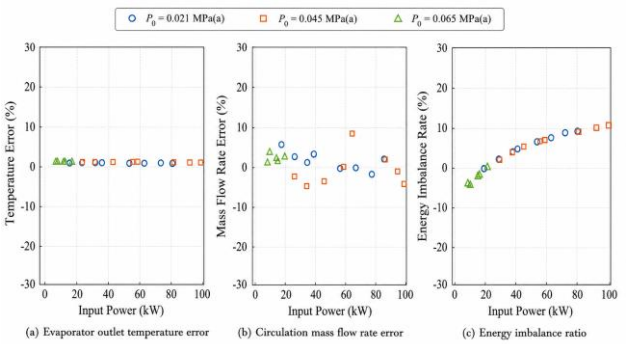


Fig. 5. Steady-state validation and error statistics of the standard loop heat pipe

It should be noted that flow distribution among parallel evaporator branches, local flow-regime transitions, and inclination-induced bubble accumulation in separated heat pipes can affect the local heat-transfer coefficient and dryout risk[12]. In the present model, these local effects are represented equivalently by sectional thermal resistances and phase-change source terms. Therefore, the model is more suitable for system-level fast calculation. If it is used for local evaporator enhancement or parallel-branch optimization, additional flow-regime criteria and branch flow-distribution submodels should be introduced.

4 Discussion

The model developed in this study is positioned as a one-dimensional fast calculation program for system-level applications. Compared with conventional system-level thermal-hydraulic programs, it divides the separated heat pipe into computable sectional nodes and retains the thermal-mass coupling among the evaporator, vapor riser, condenser, and downcomer at low computational cost. Compared with high-fidelity CFD, it does not pursue detailed resolution of local flow regimes and phase interfaces, but is more suitable for long-duration accident-transient integration, multiple geometry and operating-condition parameter scans, and heat-pipe submodule calls in system-level programs.

From a Modelica/FMI perspective, the value of the program lies in three aspects. First, the input-output boundary is explicit and compact, so the component can be connected to reactor, containment, coolant-loop, or control-system models without exposing internal heat-pipe implementation details. Second, the one-dimensional nodal formulation keeps the computational cost low enough for long-duration accident sequences, parameter scans, and repeated calls from a co-simulation master. Third, the model is not a purely empirical thermal-resistance block; it retains startup flow transition, phase-change source terms, vapor pressure drop, and liquid return, which are important for passive residual heat-removal transients. These features make the model a suitable candidate for an FMU-based heat-pipe component rather than only an offline engineering calculator.

Future work may introduce local flow-regime criteria, a nonuniform wick model, and more detailed condenser heat-transfer models. On the FMI side, the next step is to generate a Co-Simulation FMU from the MATLAB prototype, import it into a Modelica-based passive residual heat-removal system model, and compare standalone and coupled simulations under identical boundary conditions.

5 Conclusions

(1) An FMI-oriented one-dimensional transient thermal-hydraulic model is developed for separated heat pipes. The model uses a nodal thermal-resistance network to describe the evaporator, vapor riser, condenser, downcomer, wick, wall, vapor chamber, and external water tank, and captures evaporative heat absorption, vapor transport, condensation heat release, and liquid return at low computational cost.

(2) The model considers the transition from free-molecular flow to continuum flow, vapor pressure drop in the hot zone, Clausius-Clapeyron temperature-pressure coupling, evaporation/condensation mass flow rates, and capillary/gravity-assisted return flow. It is solved using a third-order Gauss-Legendre implicit Runge-Kutta algorithm and is suitable for strongly coupled transient calculations during heat-pipe start-up and accident load-reduction stages.

(3) Validation shows that the small-scale heat-pipe temperature response is consistent with literature trends, with the main temperature deviations within approximately 10%. For the standard loop heat pipe, the evaporator outlet temperature error is less than 1%, the circulation flow-rate error is approximately 10%, and the maximum energy imbalance is approximately 10.9%. These results indicate that the reduced-order solver is suitable for fast system-level calculations.

(4) The model architecture defines explicit FMI-style parameters, inputs, local states, and outputs, and maps the internal implicit time-advancement routine to the communication-step behavior of an FMU for Co-Simulation. This FMU-ready organization strengthens the model's relevance to Modelica/FMI-based system simulation and provides a basis for future cross-tool co-simulation of passive residual heat-removal systems.

Acknowledgements

The authors acknowledge the project team for their support in model development, program validation, and code implementation.

References

[1] Jeong J. J., Ha K. S., Chung B. D., et al. Development of a multi-dimensional thermal-hydraulic system code, MARS 1.3.1[J]. Annals of Nuclear Energy, 1999, 26(18): 1611-1642.

[2] Guillen D. P. Review of passive heat removal strategies for nuclear microreactor systems[J]. Nuclear Technology, 2023, 209(sup1): S21-S40.

[3] Jeong Y. S., Kim K. M., Kim I. G., et al. Feasibility study of hybrid heat pipe with control rod as passive in-core cooling system for advanced nuclear power plant[C]. NURETH-16, 2015.

[4] Kim K. M., Jeong Y. S., Kim I. G., et al. Development of passive in-core cooling system for nuclear safety using hybrid heat pipe[J]. Nuclear Technology, 2016, 196(3): 598-613.

[5] Faghri A. Heat Pipe Science and Technology[M]. Washington, DC: Taylor & Francis, 1995.

[6] Reay D. A., Kew P. A., McGlen R. J. Heat Pipes: Theory, Design and Applications[M]. 6th ed. Oxford: Butterworth-Heinemann, 2013.

[7] Wahlquist S., Hansel J., Sabharwall P., Ali A. A critical review of heat pipe experiments in nuclear energy applications[J]. Nuclear Science and Engineering, 2023, 197(5): 719-752.

[8] Jung J. H., Kim J. Y., Lee D. H., Bang I. C. Heat pipe type double-containment vessel as passive safety system for small modular reactors[J]. Nuclear Engineering and Technology, 2025, 57(3): 103236.

[9] Cheng C., Ju P., Wen Q., et al. Experimental study on heat transfer characteristics of separated type heat pipe[C]//Pacific Basin Nuclear Conference. Singapore: Springer, 2016: 339-350.

[10] Cheng C., Wang Q. L., Wang X. H., et al. Experimental study on transient characteristic of passive containment cooling system[J]. Energy Procedia, 2017, 127: 193-200.

[11] Zhou X. Study on the heat transfer characteristics of a residual heat removal system for a pool-type reactor based on separated heat pipes[D]. Harbin Engineering University, 2024. (in Chinese)

[12] Luo Y. Study on two-phase flow and heat transfer in a header-type separated gravity heat pipe[D]. Shanghai Jiao Tong University, 2020. (in Chinese)

[13] Nishikawara M., Nagano H., Kaya T. Transient thermo-fluid modeling of loop heat pipes and experimental validation[J]. Journal of Thermophysics and Heat Transfer, 2013, 27(4): 641-647.

[14] Zhang W., Hao W., Wu X. Development of transient thermal-hydraulic analysis model for the two-phase loop thermosyphon[J]. Applied Thermal Engineering, 2024, 243: 122764.

[15] Modelica Association. Functional Mock-up Interface Specification, Version 2.0.1. https://fmi-standard.org/, 2019.

[16] Blochwitz T., Otter M., Arnold M., et al. The Functional Mock-up Interface for tool independent exchange of simulation models[C]. Proceedings of the 8th International Modelica Conference, 2011: 105-114.

Research on Modeling and Simulation of Integrated Energy System Platform Based on OpenModelica

Yuan Yao[1] Xiaomeng Dong[1] Hong Lv[2] Hao Hong[2]
Wenji Li[2]

[1]College of Physics and Optoelectronic Engineering, Shenzhen University, China, 2510236008@mails.szu.edu.cn, dongxiaomeng@szu.edu.cn

[2]State Key Laboratory of Nuclear Power Safety Technology and Equipment,
China Nuclear Power Engineering Co., Ltd.,
Shenzhen, Guangdong 518116, China, lvhonglx@126.com,liwenji1986@163.com,1330636650@qq.com

Abstract

Integrated energy-system models become difficult to reuse when electrical signals, thermal-fluid states, supervisory logic, and evaluation variables are coupled through inconsistent interfaces. This paper presents an OpenModelica platform with unit-explicit cross-domain connectors, graphically encapsulated component models, and a reusable library for renewable generation, conversion, and storage. The thermal loop employs Modelica.Fluid ports and a temperature-dependent SolarSalt medium, while the wind and photovoltaic components expose available, delivered, and curtailed power to support source-level curtailment feedback from the energy-management system. A DASSL-based workflow is applied to a full-year Dunhuang case and complementary parameter-variation cases. The annual simulation maintains complete park electric-load supply, achieves a heat-load satisfaction rate of 90.16%, and yields an electric LCOE of 0.307 CNY/kWh. State-variable and sensitivity analyses indicate that renewable-source and battery sizing govern electrical absorption and economics, whereas winter thermal-storage availability and heat-side capacity constrain thermal reliability. The platform therefore links reusable Modelica implementation with long-horizon system evaluation without requiring topology-specific reformulation.

Keywords: OpenModelica, Integrated Energy System, Multi-Domain Modeling, Model Library, Annual Simulation

1 Introduction

Integrated energy systems coordinate renewable electricity, heat production, storage, conversion equipment, user demand, and grid interaction. Their behavior is governed by coupled power balances, fluid states, control decisions, and storage dynamics. As wind, photovoltaic, and solar-thermal capacities increase, a simulation environment must preserve these cross-domain interactions while allowing system configurations to be changed without rebuilding every component (Geidl and Andersson 2007; Hall and Buckley 2016).

Modelica's acausal, equation-based, and object-oriented formulation supports explicit multi-domain connections and reusable libraries, while OpenModelica provides an open translation and simulation environment (Fritzson et al. 2020; Modelica Association 2023). Existing energy libraries provide complementary coverage: Buildings and AixLib focus on building and district energy systems, TransiEnt addresses transient integrated energy networks, and ThermoSysPro provides thermo-fluid process models, including hydrogen chains (Wetter et al. 2014; Maier et al. 2024; Senkel et al. 2021; Vallejo Jimenez et al. 2023). The present platform uses the Buildings weather reader but adds a station-oriented wind-photovoltaic-tower-solar-thermal-storage configuration with custom SolarSalt properties, explicit source-level curtailment feedback, and unified annual and economic outputs.

This work makes three Modelica-oriented contributions. First, it defines unit-explicit electrical and control interfaces together with Modelica.Fluid thermal ports, enabling source, storage, load, and evaluation models to be assembled consistently across domains. Second, it encapsulates reusable OMEdit components, including a custom SolarSalt medium, and separates boundary data, physical equations, supervisory logic, and reported indicators. Third, it closes the supervisory loop between renewable-source models and the EMS by distinguishing available output, delivered output, device-level curtailment, and residual system curtailment, and

demonstrates stable full-year simulation and sensitivity analysis for the Dunhuang case.

2 Theoretical Models of the Integrated Energy System

This section summarizes only the governing relations needed to interpret the platform. Detailed constitutive equations are encapsulated inside reusable Modelica components, whereas system assembly uses unit-consistent power, heat-flow, fluid, control, and storage-state interfaces.

2.1 Energy Source Models

The source library contains wind, photovoltaic, and solar-thermal components. Wind output is evaluated from the turbine power characteristic (Carrillo et al. 2013):

$$P_m = \begin{cases} 0 & V < V_{\text{in}} \\ f_{\text{interp}}(V) & V_{\text{in}} \le V < V_{\text{rated}} \\ P_{\text{rated}} & V_{\text{rated}} \le V < V_{\text{out}} \\ 0 & V \ge V_{\text{out}} \end{cases} \quad (1)$$

where wind power is determined by wind speed, the cut-in, rated, and cut-out speeds, and the rated turbine power.

The photovoltaic component retains a single-diode equivalent relation to represent the effects of terminal voltage, irradiance, and temperature on array current (Villalva et al. 2009):

$$I = I_{ph} - I_0 \left[exp\left(\frac{V + IR_s}{aV_t}\right) - 1 \right] - \frac{V + IR_s}{R_p} \quad (2)$$

where module current depends on terminal voltage, reverse saturation current, the series and shunt resistances, and thermal voltage.

For the heliostat-receiver subsystem, the absorbed solar heat is represented by (Mehos et al. 2020)

$$Q_{helio} = DNI \cdot A_{helio} \cdot \eta_{field} \quad (3)$$

where absorbed heat depends on effective collection area, direct normal irradiance, and optical efficiency. Receiver inertia, fluid enthalpy transport, and pipeline heat loss remain inside the connected thermal-fluid components and are solved with the network equations.

2.2 Energy Conversion and Storage Models

The conversion and storage library contains battery storage, molten-salt thermal storage, an electric boiler, an electrolyzer, and a hydrogen-storage tank. Battery state of charge is governed by (Lu et al. 2013)

$$\frac{dSOC}{dt} = \frac{\eta_{\text{ch}} P_{\text{ch}}}{E_{\text{bat}}} - \frac{P_{\text{dis}}}{\eta_{\text{dis}} E_{\text{bat}}} \quad (4)$$

where battery state of charge evolves with charging and discharging power, rated energy, and the corresponding efficiencies. Power and state limits are enforced as component parameters.

The molten-salt tank exchanges heat with the solar loop and auxiliary boiler. Its energy balance is (Caraballo et al. 2021)

$$\frac{dE_{\text{tes}}}{dt} = Q_{\text{ch}} - Q_{\text{dis}} - Q_{\text{loss}} \quad (5)$$

where stored thermal energy changes with charging heat, discharging heat, and thermal loss. The reported thermal-storage state is normalized between the configured energy bounds.

The electric boiler provides controllable auxiliary heat according to

$$Q_{\text{eb}} = \eta_{\text{eb}} P_{\text{eb}} \quad (6)$$

where heat output equals electrical input multiplied by the conversion efficiency.

The electrolyzer, hydrogen-storage tank, and prescribed hydrogen load form an active flexible-conversion path. After higher-priority electrical uses are satisfied, the electrolyzer converts the remaining renewable electricity into hydrogen before residual curtailment is recorded.

$$\dot{m}_{H_2} = \frac{\eta_{el} P_{el}}{LHV_{H_2}} \quad (7)$$

$$\frac{dm_{H_2}}{dt} = \dot{m}_{in} - \dot{m}_{out} \quad (8)$$

$$P_{H_2,out} = \eta_{use} \dot{m}_{H_2,use} LHV_{H_2} \quad (9)$$

where the electrolyzer input power and conversion efficiency determine the hydrogen production rate, the stored hydrogen mass follows the inlet-outlet balance, and the hydrogen-use efficiency maps delivered mass flow to useful output.

2.3 Energy Management Model

The energy-management model coordinates renewable generation, storage, controllable conversion, user demand, and grid exchange. The station-oriented strategy first protects internal electric and heat demand while enforcing equipment capacities, temperature limits, and storage bounds.

Wind and photovoltaic power first supply the local electric load, while the battery compensates short deficits. Subject to equipment and storage limits, surplus power is then allocated to the electric boiler, grid export, battery charging, and hydrogen production. Only the remaining imbalance after these absorption paths is recorded as residual curtailed electricity.

The electric-side balance is

$$\begin{aligned} P_{\text{w}} + P_{\text{pv}} + P_{\text{bat,dis}} + P_{\text{def}} = {} & P_{\text{load}} + P_{\text{eb}} \\ & + P_{\text{grid,sell}} + P_{\text{bat,ch}} + P_{\text{el}} \\ & + P_{\text{curt}} \end{aligned} \quad (10)$$

where the source and storage terms balance local load, electric-boiler demand, grid export, electrolyzer consumption, curtailment, and unmet electric demand.

DOI 10.3384/ecp223301

On the thermal side, receiver heat and storage discharge serve the user load; the electric boiler supplies the remaining controllable heat. The balance is

$$Q_{\text{use}} + Q_{\text{tes,dis}} + Q_{\text{eb}} + Q_{def} = Q_{\text{load}} + Q_{\text{tes,ch}} + Q_{\text{curt}} \quad (11)$$

where the supplied solar, storage, and boiler heat equals user demand, storage charging, thermal curtailment, and unmet heat demand.

2.4 Economic and Comprehensive Evaluation Models

Economic evaluation reports electric LCOE, thermal LCOH, and simple payback. Electric and heat outputs are taken from the complete annual simulation. For carrier k in {e, h}, the allocated life-cycle cost includes construction, discounted operation and maintenance, replacement, and decommissioning costs; operating revenue is not deducted. The discounted life-cycle electricity and heat are

$$E_{life,k} = \sum_{y=1}^{N_k} \frac{E_{k,y}}{(1+r)^y}, k \in \{e, h\} \quad (12)$$

where E_k,y is the useful annual output of carrier k in year y and r is the discount rate. For electricity, the annual term combines wind and photovoltaic generation with their respective degradation rates; for heat, it is the annual heat delivered to users and no annual heat-output degradation is assumed.

The levelized costs are defined using the corresponding allocated life-cycle cost and discounted useful energy output:

$$LCOE = \frac{C_{life,e}}{E_{life,e}}, LCOH = \frac{C_{life,h}}{E_{life,h}} \quad (13)$$

Simple payback is evaluated separately from the levelized costs using construction cost and annual total operating revenue:

$$\mathrm{T_{pay}} = \frac{\mathrm{C_{cap}}}{\mathrm{R_{ann,total}}} \quad (14)$$

where annual total revenue is the sum of grid-electricity and user-heat revenue. Hydrogen revenue is zero under the present boundary, and annual operating cost is not subtracted in this simple-payback definition.

For electric and heat carriers k in {e, h}, the load-satisfaction rate is

$$\eta_{load,k} = 1 - \frac{E_{def,k}}{E_{load,k}}, k \in \{e, h\} \quad (15)$$

where accumulated deficit and demand are evaluated separately for the park electric and user heat loads.

System comprehensive energy efficiency is evaluated across the external system boundary as:

$$\eta_{sys} = \frac{E_{use}}{E_{wind} + E_{PV} + E_{CSP} + \Delta E_{sto}^{-}} \quad (16)$$

where the useful-output term combines park electricity actually supplied, grid export, useful heat delivered to users, and hydrogen delivered as useful output. The denominator contains actual wind and photovoltaic generation, incident CSP thermal input, and only positive net release from the battery, thermal, and hydrogen stores. Internal electric-boiler, electrolyzer, battery-charging, and storage-transfer flows are not counted again. Carbon-emission indicators are outside the present model boundary.

3 Platform Construction and Simulation Case Design

3.1 Platform Architecture and Model Organization

The platform is organized into input, model-library, management-and-evaluation, and output layers (Fig. 1). The separation is functional: external boundaries, physical equations, supervisory logic, and reported indicators remain replaceable while their interfaces stay explicit.

The input layer supplies Dunhuang weather data, prescribed electric and heat loads, equipment parameters, initial states, and economic assumptions. The library contains source, conversion, storage, load, and grid models. Management models coordinate component states and energy balances, and the output layer records typical-day, annual, economic, and parameter-variation indicators.

The diagram describes model organization rather than execution order. OpenModelica assembles the acausal equations of all connected components and solves them simultaneously, allowing verified components to be reused in different integrated energy-system configurations.

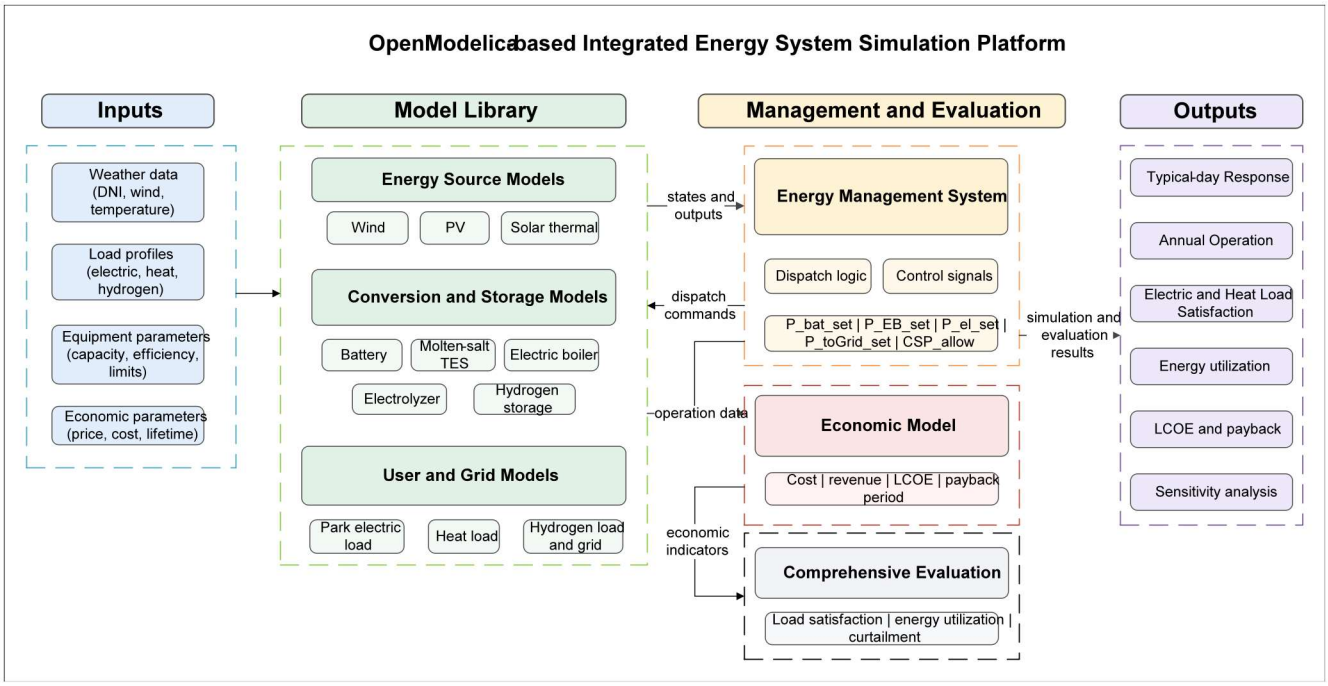


Figure 1. Architecture of the OpenModelica-based integrated energy system simulation platform.

3.2 Unified Interfaces and Graphical Model Encapsulation

Electrical, control, and evaluation interfaces use Modelica.Blocks.Interfaces.RealInput and RealOutput connectors with explicit units such as W, K, kg/s, and 1. The solar-thermal loop uses Modelica.Fluid.Interfaces.FluidPort_a and FluidPort_b, so pressure, mass flow, and stream enthalpy obey Modelica connection semantics. Redeclaring the working-medium package keeps the same thermal component compatible with different media.

Time-varying weather and load quantities are external inputs; capacity, efficiency, initial state, and operating limits are parameters. Generated power, supplied heat, storage state, curtailment, and deficits are outputs. This unit-explicit contract prevents ambiguous connections and supports automated equation checking during translation.

To support active renewable-power limitation, the wind and photovoltaic components expose available power, actual output, and curtailed power as separate variables. The EMS returns dimensionless source-specific power factors through RealOutput/RealInput connections, allowing source output to be reduced before the remaining electrical imbalance is evaluated. Device-level curtailment and residual curtailment are therefore recorded separately, preserving the physical meaning of both the source response and the system-level balance.

Each component is graphically encapsulated in OMEdit following Modelica's object-oriented and equation-based principles (Fritzson et al. 2020; Modelica Association 2023). Components are first checked independently and are then assembled progressively into source, storage, and system models without rewriting their internal equations.

3.3 Integrated Energy System Model Library

The reusable library contains energy-source, conversion-and-storage, and boundary-and-support models. Source components cover wind, photovoltaic, heliostat, receiver, and thermal-loop behavior; storage and conversion components include the battery, molten-salt tank, circulation pump, heat exchanger, electric boiler, electrolyzer, and hydrogen-storage tank.

Thermal-fluid components redeclare Energy.Media.SolarSalt, a project-specific medium package extending Modelica.Media.Interfaces.PartialMedium. It uses pressure and temperature as independent states and temperature-dependent correlations for density, heat capacity, viscosity, and thermal conductivity. This implementation preserves fluid-port compatibility while avoiding an external property-library dependency.

Boundary models include prescribed loads, the external grid, and the Buildings weather reader. Wind speed, direct and global solar irradiance, and ambient temperature are read from the Dunhuang file through Buildings.BoundaryConditions.WeatherData.ReaderTMY3 (Wetter et al. 2014).

The hydrogen path is active in the baseline dispatch as the final controllable sink before residual curtailment. The current annual configuration was translated with OpenModelica 1.25.7 using Modelica Standard Library 4.0.0 and Buildings 12.1.0. The translated model contains approximately 4,635 equation records, including 54 continuous states and 1,004 real algebraic

DOI 10.3384/ecp223301

variables. On an Intel Core i7-14700KF workstation with 32 GB RAM, DASSL at a tolerance of 1×10^{-4} completed the one-year simulation in approximately 55 s; total translation, compilation, and simulation time was approximately 62 s. This throughput made the 29 annual baseline and parameter-variation cases computationally practical on a desktop workstation.

3.4 Multi-module Coupling and Simulation Workflow

The verification case couples wind and photovoltaic generation, a battery, a heliostat-receiver loop, molten-salt storage, an electric boiler, an electrolyzer and hydrogen tank, prescribed loads, and the grid (Fig. 2). It contains the electrical, thermal, and hydrogen paths required to test cross-domain assembly and supervisory coordination.

Weather inputs drive the source models; the management model then assigns renewable power among local demand, auxiliary heating, grid export, electrical storage, hydrogen production, and source limitation. The thermal loop directs available receiver heat to the user or storage, discharges the tank when required, and records any remaining heat deficit.

The rule-based strategy tests coupled equations, state constraints, and energy balances rather than an optimal dispatch algorithm. Component implementations were checked against published response forms and operating ranges: the photovoltaic relation follows Villalva et al. (2009), and the SolarSalt properties and temperature bounds follow Caraballo et al. (2021). Annual simulations closed the implemented electric and heat balances while monitored SOC, temperature, pressure, and equipment-power states remained within configured limits. These checks establish numerical and physical consistency but not independent validation against plant measurements.

The same assembled model is used for a representative day and a complete year. The typical-day case exposes intraday state transitions, while the annual case reveals seasonal storage limits. Additional cases vary heliostat area, electric-boiler capacity, and molten-salt tank size with the remaining structure unchanged.

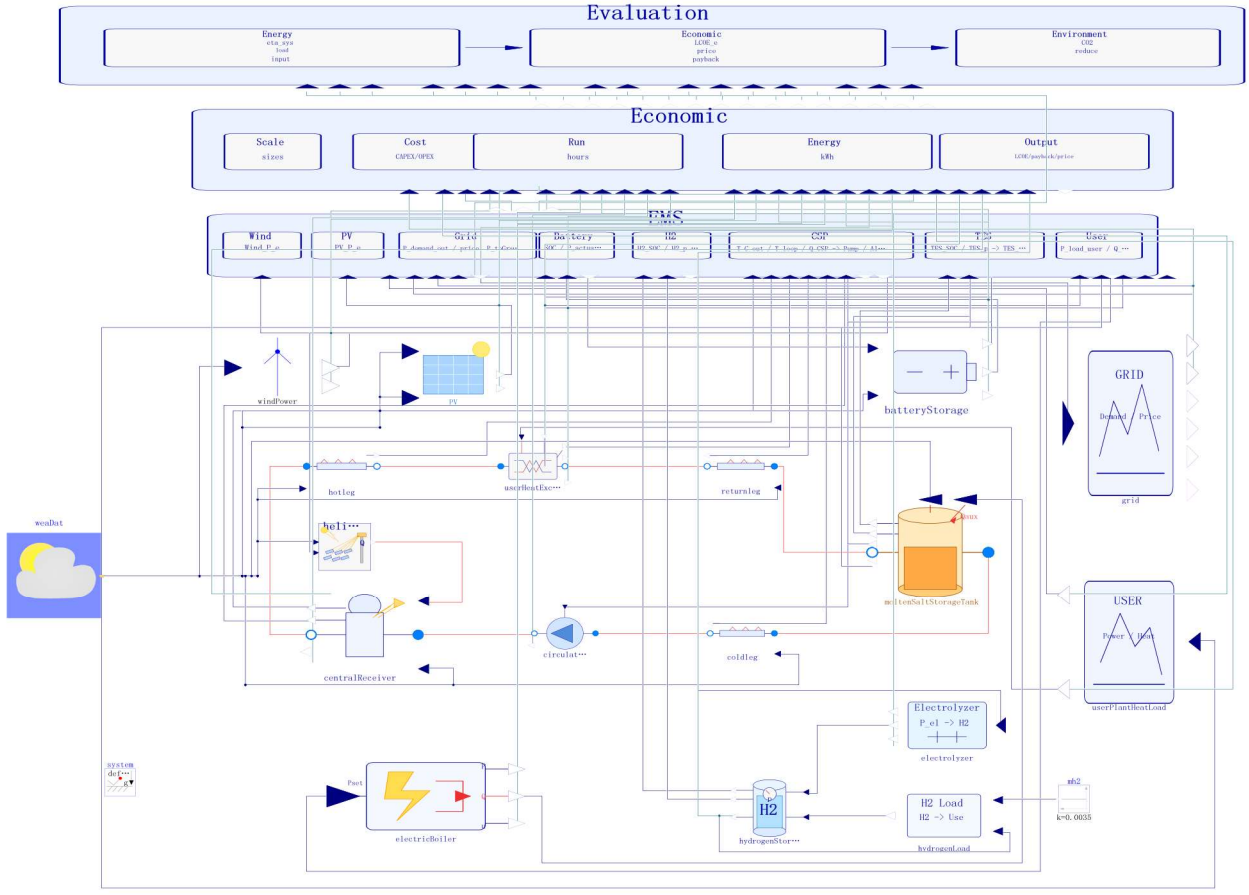


Figure 2. Topology of the integrated electricity-heat-hydrogen coupling case.

4 Simulation Results and Discussion

4.1 Simulation Conditions and Case Design

The simulation cases were designed to evaluate system-level coupling, energy-balance consistency, and state evolution under the Dunhuang weather sequence. The annual horizon was 31,536,000 s, with results recorded at 4 h intervals. DASSL was used with a tolerance of 1×10^{-4}. Day 346 was extracted as a representative winter day to examine intraday coordination, whereas the full-year case was used to quantify seasonal operation, supply reliability, energy utilization, and economic performance.

The baseline configuration comprised 45 wind turbines, 400,000 photovoltaic modules, a 250,000 m² heliostat

field, and a 1,000 MWh battery with maximum charging and discharging powers of 50 and 30 MW, respectively. The thermal and conversion equipment included an 18 MW electric boiler, an 8 MW electrolyzer, a 30 m × 20 m molten-salt tank, and a nominal solar-loop mass flow of 380 kg/s. For parameter screening, 14 engineering and control parameters were varied independently between low and high levels while the system topology and all unaffected settings were retained, resulting in 29 annual simulations including the baseline case.

4.2 Typical-Day Coordinated Operation

Figure 3 presents the electrical response on day 346. Photovoltaic output peaked near 83 MW at approximately 15:00, while wind output reached about 112 MW near 17:00. Their offset peaks reduced part of the source variability but also produced short surplus intervals. The battery operated bidirectionally between −30 and 50 MW, with positive power denoting charging, and the electrolyzer input briefly reached 7.31 MW. Device-level renewable curtailment peaked at 14.97 MW, whereas residual curtailment remained zero during the selected day. These trajectories show that the EMS coordinated storage, flexible conversion, grid exchange, and source-level limitation before the instantaneous surplus propagated to the final electrical balance.

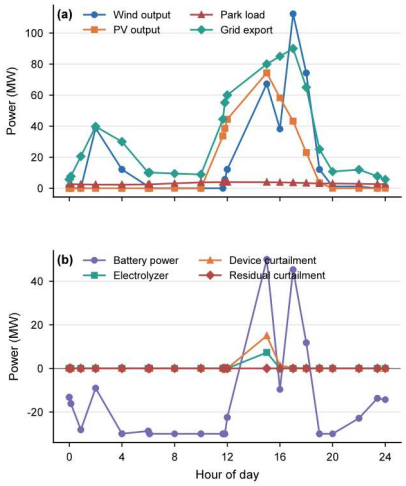


Figure 3. Typical-day electrical-side response on day 346: (a) wind and photovoltaic output, park load, and grid export; (b) battery and electrolyzer power and renewable curtailment.

The corresponding thermal response is shown in Figure 4. Heat demand varied from 11.88 to 15.90 MW, while delivered heat ranged from 10.26 to 14.65 MW. The remaining heat deficit was 0.17-4.49 MW and increased when solar-thermal input and stored energy were insufficient. Electric-boiler heat reached 16.48 MW, whereas net receiver output peaked at 2.94 MW. The molten-salt tank temperature remained between approximately 269 and 273°C, close to its lower reference temperature of 260°C. The trajectories therefore identify auxiliary heating and limited available stored heat, rather than electrical supply alone, as the active constraints on winter heat delivery.

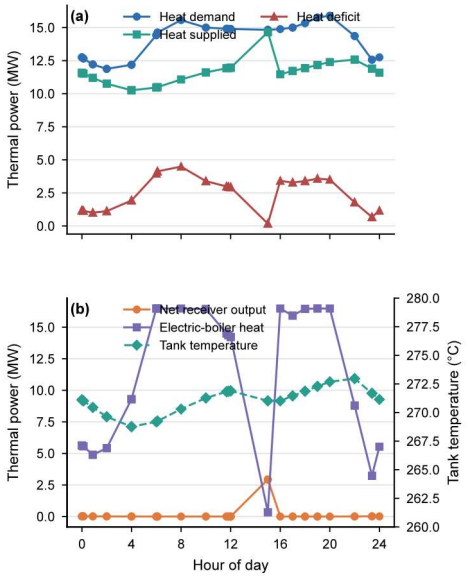


Figure 4. Typical-day thermal-side response on day 346: (a) heat demand, delivered heat, and heat deficit; (b) net receiver output, electric-boiler heat, and molten-salt tank temperature.

4.3 Annual Operating Characteristics

Figure 5 compares the daily mean available and actual wind-photovoltaic output over the annual simulation. Both trajectories exhibit pronounced weather-driven and seasonal variations, while the actual output generally follows the available resource. Integration of the complete annual trajectories gives 502.04 GWh of available wind and photovoltaic generation and 447.62 GWh of post-limitation generation. Their difference corresponds to 54.41 GWh of device-level curtailment.

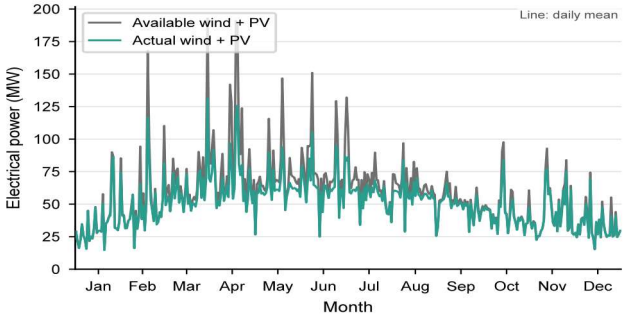


Figure 5. Annual daily-mean available and actual wind-photovoltaic output.

Figure 6 resolves the annual electrical output into monthly energy quantities. Actual wind and photovoltaic generation were 102.96 and 344.66 GWh, respectively. Device-level source curtailment totaled 54.41 GWh, while residual curtailment at the final system-balance stage was 29.37 GWh. Because the two records occur at successive locations in the control chain, they are additive when reporting total unrealized renewable resource: 83.78 GWh, or 16.7% of the 502.04 GWh available generation. Before the residual balance was curtailed, the electrolyzer absorbed 11.99 GWh of surplus electricity and supplied 2.94 GWh of useful hydrogen output.

DOI
10.3384/ecp223301

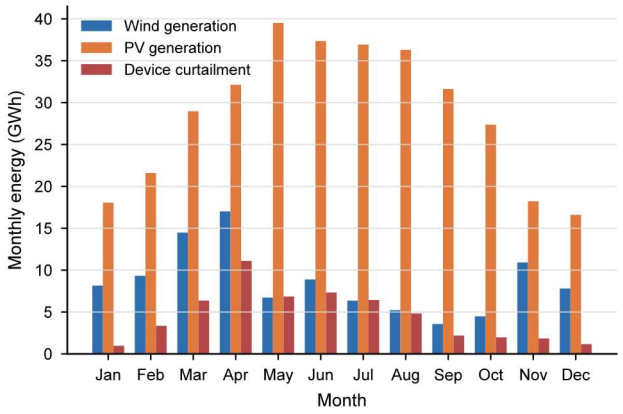


Figure 6. Monthly actual wind and photovoltaic generation and device-level renewable curtailment.

Figure 7 compares daily mean heat demand with delivered heat and highlights the remaining deficit. Heat demand was highest during the cold months, while the two trajectories nearly coincided through much of the warm season. The annual heat-load satisfaction rate was 90.16%; the average and maximum heat-deficit powers were 0.531 and 7.32 MW, respectively. Thermal curtailment nevertheless averaged 0.454 MW and reached 8.47 MW, showing that heat surplus and heat deficit occurred in different periods rather than canceling each other within the annual balance.

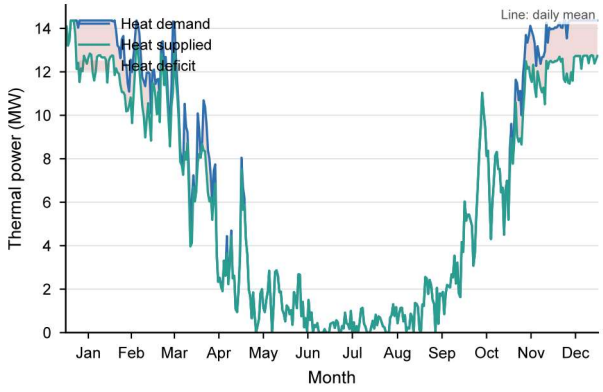


Figure 7. Daily mean heat demand and delivered heat over the annual simulation. Shading indicates the remaining heat deficit.

Figure 8 shows the corresponding seasonal behavior of the two controllable thermal sources. Electric-boiler heat was largest during the cold months and decreased to nearly zero during much of summer as heat demand declined. Net receiver output varied with solar availability and provided a larger contribution during spring and the transition seasons. The curves are shown independently rather than stacked because receiver output and boiler heat are measured at different positions in the thermal loop; storage exchange and thermal losses intervene before heat is delivered to the user. Together, Figures 7 and 8 show that the remaining winter deficit results from the coincidence of high heat demand, variable solar input, and finite auxiliary-heating and heat-transfer capability.

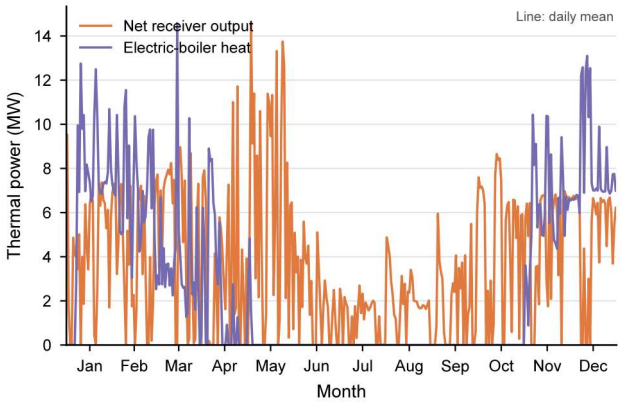


Figure 8. Daily mean net receiver output and electric-boiler heat over the annual simulation.

4.4 System-Level and Economic Performance

Table 1 summarizes the annual system-level and economic indicators. The park electric load was fully supplied, grid-demand satisfaction reached 95.44%, and heat-load satisfaction was 90.16%. Using the external-boundary definition in Eq. (16), the system comprehensive energy efficiency was 80.84%. This metric credits useful park electricity, grid export, delivered heat, and hydrogen output without double counting internal conversion and storage flows.

The full-year simulation produced 4.4762 x 10^8 kWh of post-limitation electricity. Applying wind and photovoltaic degradation and discount factors gives 8.5770 x 10^9 kWh of discounted life-cycle generation. With an electric-side life-cycle cost of CNY 2.6315 billion and without offsetting operating revenue, the electric LCOE is 0.307 CNY/kWh. The thermal LCOH is 0.372 CNY/kWh. Under Eq. (14), construction cost of CNY 2.581956 billion divided by annual grid-electricity and heat revenue of CNY 0.211242 billion gives a simple payback period of 12.22 years.

Table 1. Annual system-level and economic performance.

Indicator	*Value*
Actual annual electricity generation	4.4762×10^8 kWh
Discounted life-cycle electricity generation	8.5770×10^9 kWh
Electric-side life-cycle cost	CNY 2.6315 billion
Electric LCOE	0.307 CNY/kWh
Thermal LCOH	0.372 CNY/kWh
Investment payback period	12.22 years
Park electric-load satisfaction	100.0%
Grid-demand satisfaction	95.44%
Heat-load satisfaction	90.16%
Average heat-deficit power	0.531 MW
System comprehensive energy efficiency	80.84%

4.5 Parameter Sensitivity and Discussion

A local one-factor-at-a-time sensitivity screening was performed for 14 engineering and control parameters. Continuous parameters were varied by +/-10%, while the wind-turbine number was set to 40 and 50 around the baseline value of 45. Centered elasticities were calculated from each low-high response and normalized separately within each output column. Figure 9 therefore describes relative influence within one output metric; neither color intensity nor the printed values should be compared across different columns. The outputs are grid-demand satisfaction, heat-load satisfaction, system comprehensive energy efficiency, device-level curtailment, residual curtailment, electric LCOE, thermal LCOH, and simple payback.

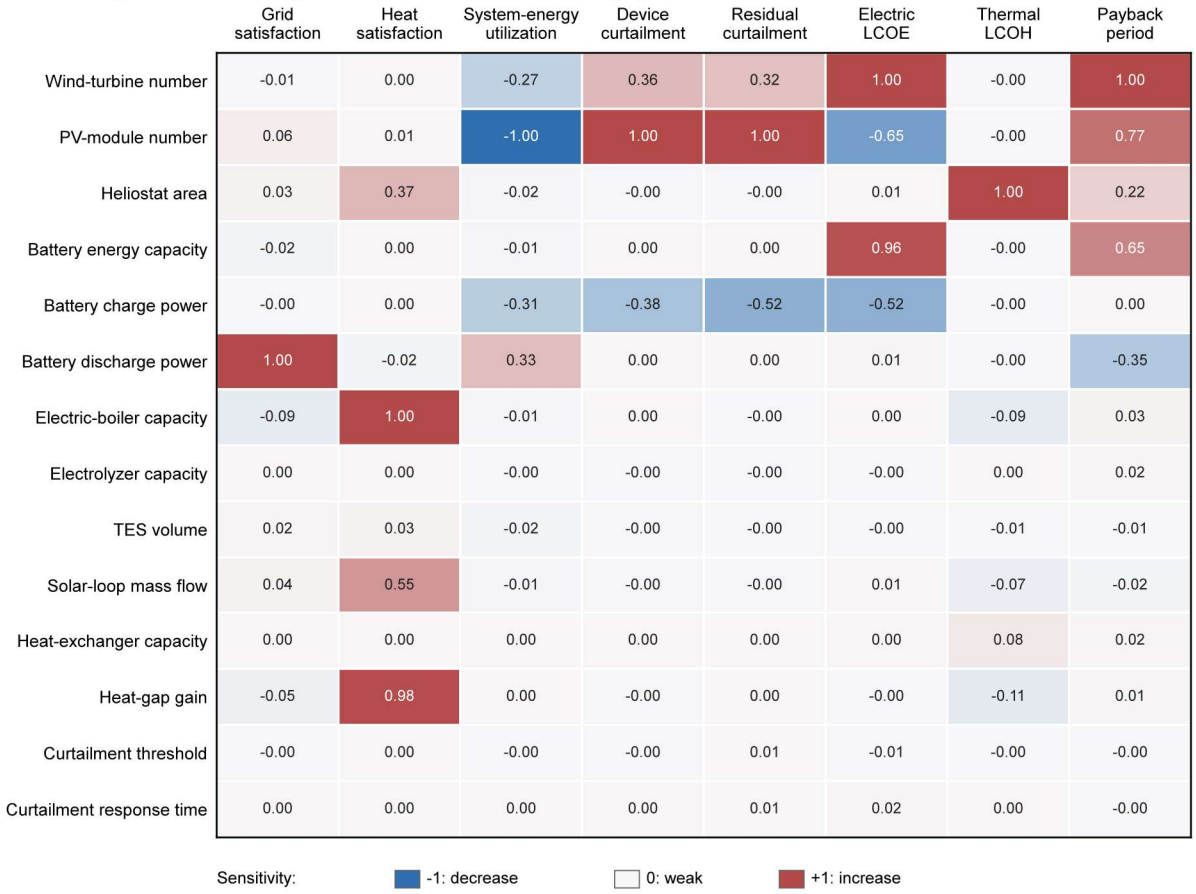


Figure 9. Column-normalized local sensitivities of eight annual outputs to low-high perturbations in 14 engineering and control parameters. Red and blue indicate positive and negative responses. Color intensity is comparable only within each output column.

The sensitivity matrix separates electrical-economic drivers from thermal-reliability drivers. Wind-turbine and photovoltaic-module numbers and battery sizing exert the strongest influence on electrical economics, system comprehensive energy efficiency, and curtailment. Additional photovoltaic capacity increases available generation but lowers comprehensive efficiency under the existing absorption limits, whereas greater battery charging power reduces curtailed energy by increasing short-term surplus absorption. Heat-load satisfaction responds most strongly to electric-boiler capacity and the heat-deficit compensation gain, followed by solar-loop mass flow and heliostat-field area.

Table 2 reports the corresponding physical changes for representative thermal-side parameters. Increasing electric-boiler capacity from 16.2 to 19.8 MW raises heat-load satisfaction by 1.13 percentage points, and increasing the heat-deficit compensation gain from 4.5 to 5.5 produces a similar 1.11-point change. Solar-loop mass flow and heliostat area produce smaller gains of 0.62 and 0.42 points. In contrast, a ±10% change in thermal-storage volume changes heat-load satisfaction by only 0.04 points. This weak local response indicates that storage volume alone is not the active bottleneck under the baseline control and charging conditions; it does not establish that thermal storage is unimportant outside the tested local range.

DOI
10.3384/ecp223301

Table 2. Two-level effects of representative thermal-side parameters on heat-load satisfaction.

Parameter	*Low level; ηh*	*High level; ηh*	*Δ (pp)*
Electric-boiler capacity	16.2 MW; 89.44%	19.8 MW; 90.57%	+1.13
Heat-deficit compensation gain	4.5; 89.55%	5.5; 90.66%	+1.11
Solar-loop mass flow	342 kg/s; 89.84%	418 kg/s; 90.46%	+0.62
Heliostat-field area	225,000 m²; 89.94%	275,000 m²; 90.36%	+0.42
TES volume	12,723 m³; 90.15%	15,551 m³; 90.19%	+0.04

The screening therefore supports two bounded conclusions. Source and battery parameters primarily govern electrical-economic performance, whereas heat-side control, transfer, and auxiliary-heating parameters govern thermal reliability. Because the analysis varies one parameter at a time over a local range, it does not quantify parameter interactions or global uncertainty; the results are used for engineering screening rather than universal parameter ranking.

5 Conclusion and Outlook

5.1 Conclusions

This study establishes a Modelica-oriented integrated energy-system platform based on unit-explicit cross-domain interfaces, reusable graphical component models, and stable full-year simulation of an approximately 4,635-equation system. The custom SolarSalt medium and Modelica.Fluid ports preserve thermal-fluid connection semantics, while source-level curtailment feedback distinguishes source response from residual system imbalance. For the Dunhuang case, park electric-load, grid-demand, and heat-load satisfaction reach 100.0%, 95.44%, and 90.16%, respectively, and system comprehensive energy efficiency reaches 80.84%. The annual electrolyzer consumption of 11.99 GWh confirms that hydrogen operates as a surplus-electricity sink before residual curtailment. The life-cycle calculation gives an electric LCOE of 0.307 CNY/kWh, a thermal LCOH of 0.372 CNY/kWh, and a simple payback period of 12.22 years. Annual trajectories and sensitivity results show that source and battery sizing govern electrical absorption and economics, whereas heat-side control, transfer, and auxiliary-heating capacity govern the remaining thermal deficit. These findings are bounded by the Dunhuang weather sequence, prescribed loads, rule-based dispatch, and the absence of plant-data validation.

5.2 Future Work

Future work will compare component and interface responses with measured plant data or established benchmark cases, introduce predictive or optimization-based supervisory control, and test additional climates and load profiles. Hydrogen dispatch, storage sizing, and revenue treatment will also be optimized so that its effect on curtailment, comprehensive efficiency, and project economics can be quantified consistently.

References

Caraballo, A., Galán-Casado, S., Caballero, A. and Serena, S. (2021). *"Molten Salts for Sensible Thermal Energy Storage: A Review and an Energy Performance Analysis"*. In: Energies 14.4, 1197. DOI: 10.3390/en14041197. URL: https://www.mdpi.com/1996-1073/14/4/1197.

Carrillo, C., Obando Montaño, A. F., Cidrás, J. and Díaz-Dorado, E. (2013). *"Review of power curve modelling for wind turbines"*. In: Renewable and Sustainable Energy Reviews 21, pp. 572–581. DOI: 10.1016/j.rser.2013.01.012. URL: https://www.sciencedirect.com/science/article/abs/pii/S1364032113000439.

Fritzson, P., Pop, A., Abdelhak, K., Asghar, A., Bachmann, B., Braun, W., Bouskela, D., Braun, R., Buffoni, L., Casella, F., Castro, R., Franke, R., Fritzson, D., Gebremedhin, M., Heuermann, A., Lie, B., Mengist, A., Mikelsons, L., Moudgalya, K., Ochel, L., Palanisamy, A., Ruge, V., Schamai, W., Sjölund, M., Thiele, B., Tinnerholm, J. and Östlund, P. (2020). *"The OpenModelica Integrated Environment for Modeling, Simulation, and Model-Based Development"*. In: *Modeling, Identification and Control* 41.4, pp. 241–295. DOI: 10.4173/mic.2020.4.1. URL: https://openmodelica.org/research/the-openmodelica-integrated-environment-for-modeling-simulation-and-model-based-development/.

Geidl, M. and Andersson, G. (2007). *"Optimal Power Flow of Multiple Energy Carriers*". In: IEEE Transactions on Power Systems 22.1, pp. 145–155. DOI: 10.1109/TPWRS.2006.888988. URL: https://ieeexplore.ieee.org/document/4077095.

Hall, L. M. H. and Buckley, A. R. (2016). "*A review of energy systems models in the UK: Prevalent usage and categorisation"*. In: Applied Energy 169, pp. 607–628. DOI: 10.1016/j.apenergy.2016.02.044. URL: https://www.sciencedirect.com/science/article/pii/S0306261916301672.

Lu, L., Han, X., Li, J., Hua, J. and Ouyang, M. (2013). "*A review on the key issues for lithium-ion battery management in electric vehicles*". In: Journal of Power Sources 226, pp. 272–288. DOI: 10.1016/j.jpowsour.2012.10.060.

Maier, L., Krutzke, A., Gorzalka, P., Blacha, T., Bargiacchi, E., Mehrfeld, P., Muller, D. and Hering, D. (2024). *"AixLib: an open-source Modelica library for compound building energy systems from component to district level with automated quality management".* In: Journal of Building Performance Simulation 17.2, pp. 196-219. DOI: 10.1080/19401493.2023.2250521.

Mehos, M., Price, H., Cable, R., Kearney, D., Kelly, B., Kolb, G. and Morse, F. (2020). "*Concentrating Solar Power Best Practices Study*". Tech. rep. NREL/TP-5500-75763. Golden, CO: National Renewable Energy Laboratory. URL: https://www.nrel.gov/docs/fy20osti/75763.pdf.

Modelica Association (2023). Modelica® – *A Unified Object-Oriented Language for Systems Modeling. Language Specification Version 3.6*. Tech. rep. Linköping: Modelica Association.

Senkel, A., Bode, G., Schmitz, G. and Ritschel, T. K. S. (2021). *"Status of the TransiEnt Library: Transient Simulation of Complex Integrated Energy Systems"*. In: Proceedings of the 14th International Modelica Conference, pp. 187-196. DOI: 10.3384/ecp21181187.

Vallejo Jimenez, J. A., Corona Mesa-Moles, M., Faille, D. and Irofti, D. (2023). *"Modeling and Simulation of the Hydrogen Value Chain with ThermoSysPro and Modelica".* In: Proceedings of the 15th International Modelica Conference. DOI: 10.3384/ecp204229.

Villalva, M. G., Gazoli, J. R. and Ruppert Filho, E. (2009). "*Comprehensive Approach to Modeling and Simulation of Photovoltaic Arrays*". In: IEEE Transactions on Power Electronics 24.5, pp. 1198–1208. DOI: 10.1109/TPEL.2009.2013862. URL: https://ieeexplore.ieee.org/document/4806084.

Wetter, M., Zuo, W., Nouidui, T. S. and Pang, X. (2014). "*Modelica Buildings library". In: Journal of Building Performance Simulation* 7.4, pp. 253–270. DOI: 10.1080/19401493.2013.765506. URL: https://www.tandfonline.com/doi/abs/10.1080/19401493.2013.765506.

Modelica-Based Performance Analysis of an Organic Rankine Cycle under Marine Hot- and Cold-Source Variations

Hongfei Zhu [1,2] Xiao Yan [2] Ming Ding [1]

[1] Heilongjiang Provincial Key Laboratory of Nuclear Power System & Equipment, Harbin Engineering University, Harbin, 150001, China;

[2] CNNC Key Laboratory on Nuclear Reactor Thermal Hydraulics Technology, Nuclear Power Institute of China, Chengdu, 610213, China

Abstract

Fluctuations in offshore low-temperature heat sources and cold seawater alter the heat input, heat rejection, and condensing pressure of organic Rankine cycle (ORC) systems, making reliable off-design assessment important for marine power conversion. This study investigates a closed R134a ORC subjected to hot- and cold-source temperature and flow variations. A variable-condensing-pressure model is developed in Modelica. The evaporator and condenser are discretized using the finite-volume method, and receiver pressure is determined from coupled mass and energy balances; the working-fluid mass flow is maintained at 0.50 kg/s. Steady-state matrices are used to quantify pressure, phase state, net power, and thermal efficiency. At 363.15 K and 2.50 kg/s on the hot side and 288.15 K and 1.53 kg/s on the cold side, the model gives high/low pressures of 2.496/0.793 MPa, net power of 4.557 kW, thermal efficiency of 4.66%, and receiver level of 0.501. Across the normal cold-source-temperature cases, condensing pressure and net power vary by 29.7 kPa and 0.185 kW, respectively; the corresponding ranges over the cold-source-flow matrix are 35.5 kPa and 0.224 kW. The 359.15 and 360.15 K hot-source cases lead to wet expansion, whereas the 278.15 and 283.15 K cold-source cases cross the pump-inlet phase boundary. Refined calculations over the 89-90 °C hot-source interval show increases of 0.128 kW in net power and 1.532 kW in evaporator heat input, with thermal efficiency increasing from 4.606% to 4.664%. These results quantify the distinct effects of heat-input and heat-rejection variations under variable condensing pressure and identify the associated phase boundaries.

Keywords: Organic Rankine cycle; marine energy; Modelica; steady-state performance; variable condensing pressure; R134a

1 Introduction

With the continuous expansion of global oceanographic exploration toward deep and remote sea areas, advanced marine equipment such as deep-sea integrated observation networks and offshore unattended platforms has created an increasingly urgent demand for long-term power supplies in the kilowatt to hundred-kilowatt range (Lin & Yang, 2020). Such equipment typically operates in environments characterized by high pressure, low temperature, seawater corrosion, and difficult maintenance. Supplying power solely with batteries is constrained by endurance and periodic replacement requirements, whereas shore-based power delivered through submarine cables involves engineering challenges such as long-distance transmission, subsea connection, pressure-resistant insulation of equipment, and difficult fault repair (Rajashekara, 2017). Therefore, it is of practical significance to develop small-scale energy conversion systems that can use in-situ marine energy resources to provide supplementary power for distributed marine equipment. Previous studies have shown that low-temperature fluids below approximately 100 °C exist in diffuse submarine hydrothermal flows, with heat derived from the mixing of magma-heated fluids with seawater and from conductive heating processes (Bemis, Lowell & Farough, 2012). For offshore geothermal resources, conceptual studies have also proposed arranging heat exchangers on the seabed to transfer heat from geothermal fluids to the working fluid of a closed binary cycle, thereby reducing pressure losses and two-phase flow problems caused by the long-distance transport of natural fluids (Hiriart, Prol, Alcocer & Espindola, 2010; Karason, Gudjonsdottir, Valdimarsson & Thorolfsson, 2013).

The organic Rankine cycle uses low-boiling-point organic working fluids and is suitable for compact energy conversion systems driven by medium- and low-temperature heat sources. In the field of marine energy utilization, closed ORC systems have been studied for ocean thermal energy conversion, in which deep cold seawater can serve as the cooling source on the condensation side and provide a reference for the cold-end configuration of low-temperature cycle systems in marine scenarios. For ORC systems, previous research has covered steady-state performance analysis, working-fluid selection, experimental test benches, dynamic modeling, and control strategies. Regarding dynamic modeling, relevant reviews indicate that heat exchangers are usually

the main components governing the dynamic response of ORC systems, and commonly used models include moving-boundary, finite-volume, and two-volume models; the time scales of pumps and expanders are generally shorter than the thermal response time scale of heat exchangers (Imran, Pili, Usman & Haglind, 2020). In terms of modeling tools, ThermoCycle is an open-source Modelica library developed for small-capacity thermodynamic cycles. It can handle the thermophysical properties of organic working fluids through external property interfaces and describe single-phase and two-phase flow processes using the finite-volume method (Quoilin, Desideri, Wronski, Bell & Lemort, 2014). For ORC systems using marine low-temperature heat-source loops and cold seawater as the hot and cold sources, the operating boundaries differ from those of conventional stable heat-source conditions. Variations in heat-source temperature and flow rate alter the heat supply capacity of the evaporator, the phase-change process, and the expander inlet state. Variations in cold-source temperature and flow rate affect the heat transfer capacity of the condenser, the condensation pressure, and the expander back pressure, thereby influencing system power output. It is therefore necessary to quantify how typical marine hot- and cold-source boundary variations affect heat transfer, phase state, pressure, net power, and thermal efficiency, and to identify the resulting operating limits.

This study establishes a parameterized R134a ORC model and imposes hot-source temperature, hot-source flow, cold-source temperature, and cold-source flow disturbances independently while holding the working-fluid mass flow at 0.50 kg/s. The heat exchangers are discretized using the finite-volume method, and condenser pressure evolves through the coupled inventory and energy balances of the receiver. The analysis distinguishes heat-input effects at the evaporator from heat-rejection effects at the condenser and compares their influences on pressure, phase state, net power, and thermal efficiency. Phase and operating-envelope constraints relevant to marine service are also identified.

2 System Description and Mathematical Model

2.1 System Configuration and Operating Principle

The object of this study is a closed organic Rankine cycle power generation system for offshore deep-sea energy supply scenarios, and its configuration is shown in Figure 1. After intermediate heat exchange, submarine hydrothermal fluid forms a low-temperature hot-water loop with a temperature range of 80-90 °C, which serves as the external heat source for the ORC evaporator. Cold seawater provides the cooling source for the condenser through a cooling loop. The internal cycle uses R134a as the working fluid and mainly consists of an evaporator, expander, condenser, and working-fluid pump.

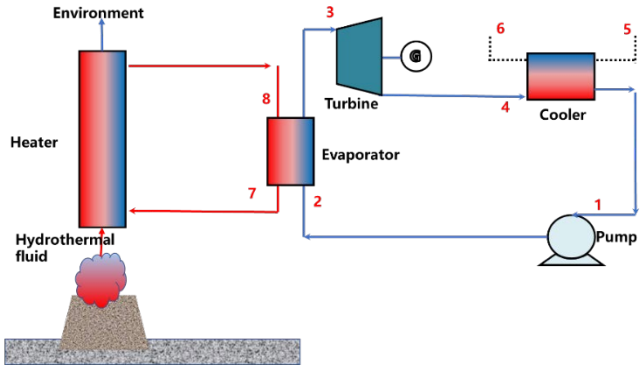


Figure 1 Schematic diagram of the closed organic Rankine cycle power generation system driven by submarine hydrothermal energy

During cycle operation, the organic working fluid is first pressurized by the working-fluid pump and then enters the evaporator, where it exchanges heat with the external low-temperature hot-water loop. The working fluid absorbs heat in the evaporator and undergoes preheating, evaporation, and necessary superheating. The high-temperature, high-pressure working fluid then enters the expander, expands to produce work, and drives the generator to output electrical power. The low-pressure working fluid at the expander outlet enters the condenser, exchanges heat with cold seawater, releases heat, and condenses into liquid. The liquid working fluid subsequently re-enters the working-fluid pump, completing the closed cycle.

2.2 Mathematical Model

Evaporator and Condenser Models

The evaporator and condenser are key components that enable energy exchange between the hot and cold sources and the circulating working fluid. In this study, a one-dimensional control-volume method is adopted to describe the dynamic processes in the heat exchangers, which are divided into several computational cells along the flow direction. The model assumes that fluid state parameters vary only along the main flow direction, while radial non-uniformity and heat loss from the heat exchanger to the surroundings are neglected. The working fluid in each control volume is assumed to be in local thermodynamic equilibrium.

The fluid mass conservation equation is expressed as:

$$V_{h,i}\frac{d\rho_{h,i}}{dt} = \dot{m}_{h,in} - \dot{m}_{h,out} \tag{1}$$

$$V_{c,i}\frac{d\rho_{c,i}}{dt} = \dot{m}_{c,in} - \dot{m}_{c,out} \tag{2}$$

where $V_{h,i}$ and $V_{c,i}$ are the volumes of the heat-transfer cells, generally taken as the cell volumes through which the working fluid flows; $\rho_{h,i}$ and $\rho_{c,i}$ are the working-fluid densities; and m is the working-fluid mass flow rate.

DOI
10.3384/ecp223311

The fluid energy conservation equation in the control volume is expressed as:

$$V_{h,i}\frac{d(\rho_{h,i}u_{h,i})}{dt}=\dot{m}_{h,in}h_{h,in}-\dot{m}_{h,out}h_{h,out}-Q_{h,i} \quad (3)$$

$$V_{c,i}\frac{d(\rho_{c,i}u_{c,i})}{dt}=\dot{m}_{c,in}h_{c,in}-\dot{m}_{c,out}h_{c,out}+Q_{c,i} \quad (4)$$

where $V_{h,i}$ and $V_{c,i}$ are the volumes of the hot-side and cold-side heat-transfer cells, respectively, and are equal to the product of the flow-channel cross-sectional areas $A_{h,i}$ and $A_{c,i}$ and the heat-transfer cell length δx; $\rho_{h,i}$ and $\rho_{c,i}$ are the working-fluid densities in the heat-transfer cells; and $h_{h,in}$, $h_{h,out}$, $h_{c,in}$, and $h_{c,out}$ are the specific enthalpies of the fluids at the hot-side and cold-side inlets and outlets of the heat-transfer cells, respectively.

The momentum equation connects adjacent nodes i and i+1. The unsteady momentum variation of the fluid in the flow channel is driven by the pressure difference between adjacent nodes and must overcome various resistance pressure drops:

$$\frac{1}{A_i}\cdot\frac{d\dot{m}_{h,i}}{dt}\cdot\Delta x_i=P_{h,in}-P_{h,out}-(\Delta P)^a_{h,i}-(\Delta P)^f_{h,i}-(\Delta P)^g_{h,i} \quad (5)$$

$$\frac{1}{A_i}\cdot\frac{d\dot{m}_{c,i}}{dt}\cdot\Delta x_i=P_{c,in}-P_{c,out}-(\Delta P)^a_{c,i}-(\Delta P)^f_{c,i}-(\Delta P)^g_{c,i} \quad (6)$$

The left-hand side of the equation is the unsteady inertial term, representing the rate of change of fluid momentum. The right-hand side is the resultant force acting on the control volume, including the inlet-outlet pressure difference and three main pressure-loss terms: the acceleration pressure drop caused by density variation, (ΔP)a; the frictional pressure drop caused by wall resistance, (ΔP)f; and the gravitational pressure drop caused by elevation difference, (ΔP)g.

Expander Model

The transient output shaft power Wexp of the expander depends on the enthalpy drop of the working fluid between the inlet and outlet and on the isentropic efficiency. Its dynamic balance equation is:

$$W_{\exp}=\dot{m}_{wf}(h_{in}-h_{out,s})\eta_t \quad (7)$$

where m is the instantaneous mass flow rate through the expander; hin is the actual specific enthalpy at the expander inlet; hout,s is the specific enthalpy of the working fluid after ideal isentropic expansion to the outlet pressure; and ηt is the isentropic efficiency of the expander.

To predict the operating characteristics of the system under off-design conditions caused by heat-source fluctuations, the Stodola ellipse law is introduced to constrain the expander flow characteristics [8]. The Stodola equation relates the turbine through-flow rate to the inlet and outlet pressures and the inlet state:

$$\dot{m}_{wf}=K\sqrt{\rho_{in}P_{in}\left(1-\frac{P_{out}}{P_{in}}\right)^2} \quad (8)$$

where Pin and Pout are the real-time inlet and outlet pressures of the expander, respectively, and ρin is the transient density of the working fluid at the inlet. The flow coefficient K is a constant determined from the expander parameters under the rated design condition and represents the equivalent flow area of the expander passage.

Pump Model

The nonlinear dynamic coupling among the actual pump head H, volumetric flow rate Qv, and real-time rotational speed n is converted equivalently using the affinity laws. Given the head-flow characteristic function fH(Qv) of the pump at the rated design speed n0, the dynamic head equation at an arbitrary variable-frequency speed n can be written as:

$$H=\left(\frac{n}{n_0}\right)^2\cdot f_H\left(Q_v\cdot\frac{n_0}{n}\right) \quad (9)$$

where H is the head under the current operating condition, Q_v is the volumetric flow rate, n is the actual pump speed, n0 is the rated pump speed, and $f_H(x)$ is the head-flow characteristic function at the rated speed.

The pump shaft power Wshaft is given by:

$$W_{shaft}=\frac{\Delta p\times Q_v}{\eta}=\frac{\rho\cdot g\cdot H\cdot Q_v}{\eta} \quad (10)$$

where η is the overall pump efficiency, $f\eta(x)$ is the efficiency-flow characteristic function at the rated speed, W_{shaft} is the pump shaft power, Δp is the pressure difference between the pump outlet and inlet, and ρ is the fluid density.

2.3 Modelica Implementation, Fluid Interfaces, and Phase Identification

The top-level implementation is shown in Figure 2. Modelica fluid connectors carry the across variable p, the flow variable m_flow, and the stream variable h_outflow. Transported enthalpy is evaluated with inStream or actualStream according to flow direction. R134a properties are calculated with FreeFluids. Its state record supplies phase and gas fraction gf; states are reconstructed from p and h and classified as subcooled liquid, two-phase fluid, or dry/superheated vapour. The evaporator and condenser use finite-volume balances, while the expander and pump retain lumped dynamic response assumptions.

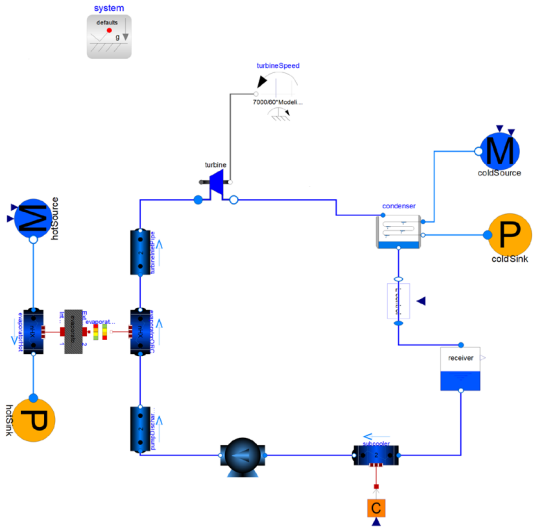


Figure 2 Modelica Diagram View of the parameterized ORC model with the dynamic receiver

Figure 3 plots four simulated state points on the R134a T-s diagram. The state temperatures and entropies are obtained from the baseline simulation. The saturation dome is calculated using CoolProp; before constructing the diagram, its saturation temperatures are compared with FreeFluids values at the simulated high and low pressures to maintain a consistent property basis.

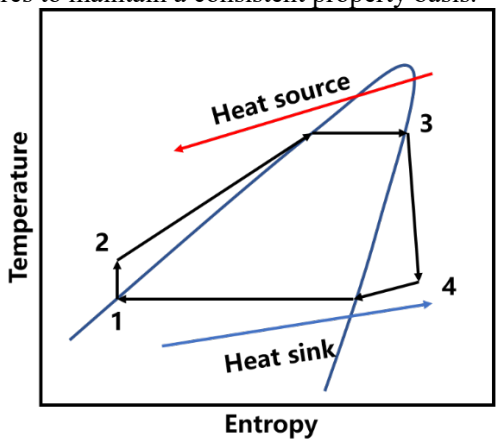


Figure 3 R134a T-s diagram based on baseline simulation state properties

3 Model Validation and Steady-State Performance Analysis

3.1 Validation Under the Baseline Operating Condition

To validate the computational accuracy of the established ORC system model under the rated operating condition, the rated operating data of an R134a organic Rankine cycle test rig reported in Ref. (Πάλλης & Pallis, 2020) are selected as the comparison basis. The experimental system uses low-temperature hot water at 90 °C as the heat source. In this study, the working-fluid inlet and outlet temperatures and pressures of the evaporator and condenser are extracted as validation parameters, and the key state-point data of the experimental system are compared with the steady-state calculation results of the Modelica model, as shown in Table 1. Table 1 indicates that the model predictions agree well with the literature data overall, with a maximum relative error of 1.43%. These results demonstrate that the established ORC system model can accurately describe evaporative heat absorption, expansion work production, and condensation heat rejection under rated operation, and can therefore be used for subsequent analysis of system dynamic characteristics under hot- and cold-source boundary fluctuations.

Table 1 Comparison between model calculation results and literature data under the rated operating condition

Parameter	Literature value	Modelica result	Relative error
Rated system flow rate (kg/s)	0.5	0.5	
Evaporator			
Inlet temperature (K)	308.15	303.75	1.43%
Inlet pressure (bar)	25.00	24.93	0.28%
Outlet temperature (K)	355.15	356.35	0.34%
Outlet pressure (bar)	25.00	24.93	0.28%
Condenser			
Inlet temperature (K)	313.15	314.35	0.38%
Inlet pressure (bar)	9.50	9.572	0.76%
Outlet temperature (K)	303.15	302.73	0.14%
Outlet pressure (bar)	9.50	9.51	0.11%

3.2 Steady-State Effects of Hot-Source Variations

Figures 4 and 5 show the hot-source steady-state results. The temperature range is 359.15-363.15 K, with an additional point at 362.40 K, and the flow-rate cases are 2.50, 2.60, 2.625, 2.65, and 2.70 kg/s. Increasing hot-source temperature or flow generally increases the evaporator heat input and system net power. At 359.15 and 360.15 K, the expander inlet enters the wet-vapour region; these cases therefore define phase boundaries rather than normal dry-expansion operating points.

DOI
10.3384/ecp223311

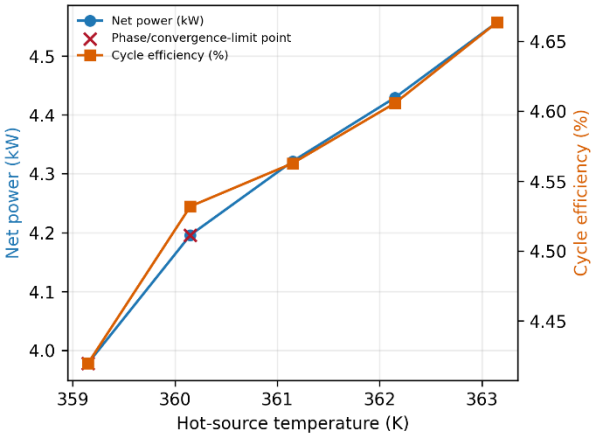


Figure 4 Effect of hot-source temperature; crosses identify phase-limit points

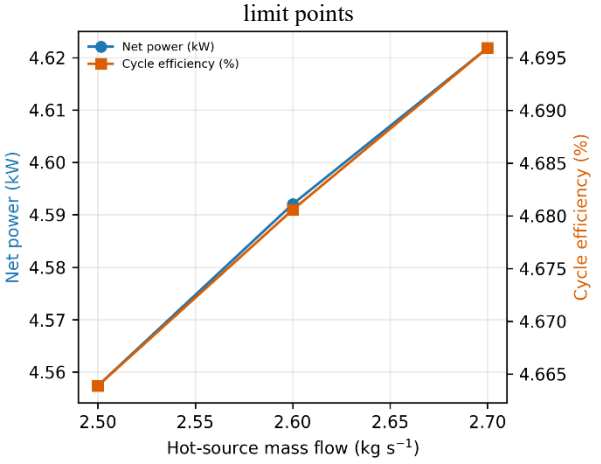


Figure 5 Effect of hot-source mass flow on net power and efficiency

To clarify the slope changes in Figures 4 and 5, refined calculations were performed over 89-90 °C and 2.60-2.70 kg/s. At 89 and 90 °C, expander-inlet quality is 1.0000/1.0000 and the superheat margin is 3.18/5.57 K. Both endpoints are dry or superheated; therefore, the slope change is not caused by a phase transition. High pressure increases from 2.474 to 2.496 MPa and low pressure from 0.792 to 0.793 MPa. Over the same interval, net power and evaporator heat input increase by 0.128 and 1.532 kW, respectively, while thermal efficiency increases from 4.606% to 4.664%. When hot-source flow increases from 2.60 to 2.70 kg/s, expander power, pump work, net power, evaporator heat input, and condenser heat rejection change by 0.032, 0.002, 0.030, 0.312, and 0.280 kW, respectively. Because thermal efficiency is defined as W_net/Q_evap, the non-proportional changes in power and heat transfer, together with the simultaneous movement of the high and low pressures, explain the slope variation.

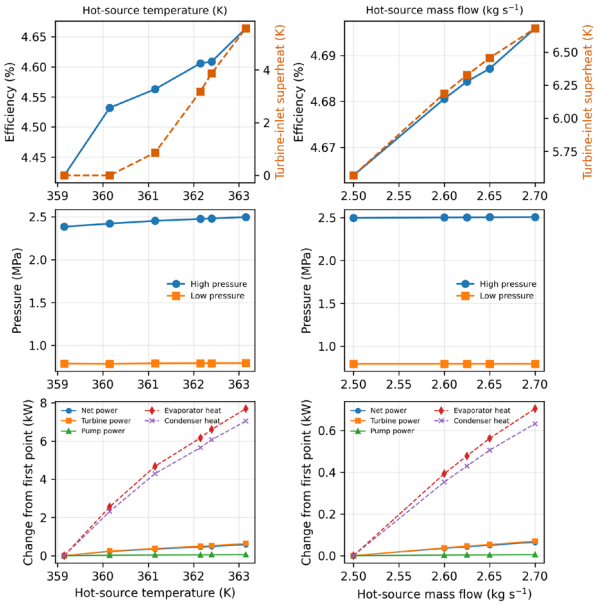


Figure 6 Refined scans near 89-90 °C and 2.60-2.70 kg/s

3.3 Steady-State Effects of Cold-Source Variations

Figures 7 and 8 show the cold-source steady-state results for four temperatures (278.15, 283.15, 288.15, and 293.15 K) and four comparable flow rates (1.23, 1.38, 1.53, and 1.83 kg/s). Within the normal temperature points, condensing pressure and net power span 29.7 kPa and 0.185 kW; the full flow matrix spans 35.5 kPa and 0.224 kW. The 278.15 and 283.15 K cases cross the pump-inlet phase boundary (two-phase quality approximately 0.14 and 0.13) and are excluded from normal performance comparisons; they are retained only to mark the operating limit. Warmer coolant or reduced coolant flow weakens heat rejection, raises receiver/condenser pressure and expander back pressure, and reduces net power.

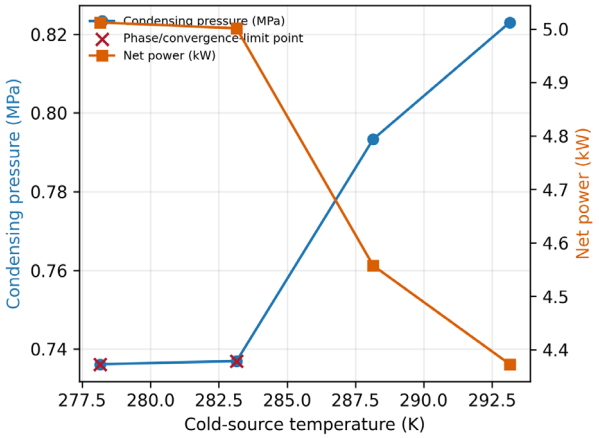


Figure 7 Effect of cold-source temperature on condensing pressure and net power

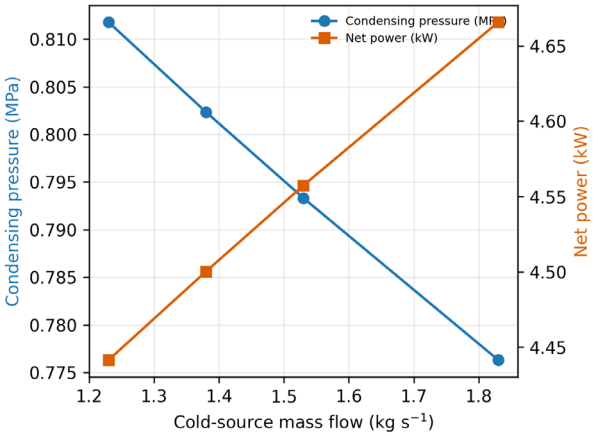


Figure 8 Effect of cold-source mass flow on condensing pressure and net power

3.4 Model Scope and Limitations

The receiver is treated as a well-mixed equilibrium two-phase volume; phase separation in vertical pipes and components, entrainment, and maldistribution are not resolved. Non-condensable-gas pressure is set to zero, and external seawater hydrostatic pressure, corrosion, fouling, and structural pressure compensation are outside the model. The source loops use prescribed boundary temperatures and flows. A 12-control-volume discretization reproduces the compilation and baseline trends, but grid independence within 2% across the full operating matrix has not been established. Accordingly, the quantitative analyses use the six-control-volume discretization. The model is intended for comparative off-design performance and operating-envelope analysis, rather than detailed prediction of separator hydrodynamics or long-term material degradation in marine environments.

4 Conclusions

At the baseline condition, the variable-condensing-pressure model predicts high/low pressures of 2.496/0.793 MPa, net power of 4.557 kW, thermal efficiency of 4.66%, and receiver level of 0.501. The expander-inlet superheat and pump-inlet subcooling are 5.57 and 1.51 K, respectively, while the mass and energy residuals remain below 1% of their corresponding fluxes. The cold-source-temperature matrix produces ranges of 29.7 kPa in condensing pressure and 0.185 kW in net power, while the cold-source-flow matrix produces corresponding ranges of 35.5 kPa and 0.224 kW. The 278.15 and 283.15 K cold-source cases cross the pump-inlet phase boundary, with two-phase qualities of approximately 0.14 and 0.13. In the 89-90 °C hot-source interval, net power increases by 0.128 kW, evaporator heat input increases by 1.532 kW, and thermal efficiency increases from 4.606% to 4.664%.

Acknowledgements

This work was supported by the project entitled “Dynamic Characteristics and Autonomous Control of an Organic Rankine Cycle System under Fluctuating Offshore Heat-Source Conditions”.

References

Bemis, K., Lowell, R. & Farough, A. 2012. Diffuse flow on and around hydrothermal vents at mid-ocean ridges. *Oceanography*. 25(1):182–191. DOI: 10.5670/oceanog.2012.16.

Hiriart, G., Prol, R.M., Alcocer, S. & Espindola, S. 2010. Submarine geothermics. Hydrothermal vents and electricity generation. 1–6.

Imran, M., Pili, R., Usman, M. & Haglind, F. 2020. Dynamic modeling and control strategies of organic rankine cycle systems: methods and challenges. *Applied Energy*. 276:115537. DOI: 10.1016/j.apenergy.2020.115537.

Karason, B., Gudjonsdottir, M.S., Valdimarsson, P. & Thorolfsson, G. 2013. Utilization of offshore geothermal resources for power production. 1–10.

Lin, M. & Yang, C. 2020. Ocean observation technologies: a review. *Chinese Journal of Mechanical Engineering*. 33(1):32–49. DOI: 10.1186/s10033-020-00449-z.

Quoilin, S., Desideri, A., Wronski, J., Bell, I. & Lemort, V. 2014. ThermoCycle: a modelica library for the simulation of thermodynamic systems, in. 683–692. DOI: 10.3384/ecp14096683.

Rajashekara, K. 2017. Electrification of subsea systems requirements and challenges in power distribution and conversion. *CPSS Transactions on Power Electronics and Applications*. 2(4):259–266. DOI: 10.24295/CPSSTPEA.2017.00024.

Pallis, P. 2020. Experimental investigation and economic assessment of a fully automated organic Rankine cycle for waste heat recovery from marine engine jacket cooling water. National Technical University of Athens, School of Mechanical Engineering. DOI: 10.12681/eadd/47576.

Modelica-Based Simulation of Hydrodynamics and Thermomechanical Stress in Water-Wall Tubes of a 1000 MW Ultra-Supercritical Boiler

Tianyu Chu[1] Can Fang[1] Jiali Huang[1] Wei Zhang[2]
Junhua Chen[3] Dong Guo[3] Yi Lu[4]

{chutianyu,fangcan,huangjiali}@rankyee.com
[2]Zhejiang Baimahu Laboratory Co., Ltd., Hangzhou, China,
zw_sim@163.com
[3]Huadian Electric Power Research Institute Co.{junhua-chen,dong-guo}@chder.com
[4] Shenzhen Rankyee Technology Co., Ltd.luyi@rankyee.com

Abstract

Under low-load conditions, the water wall of ultra-supercritical boilers often faces severe challenges such as flow distribution imbalance, phase change point migration, and hydrodynamic multiplicity, which can easily induce tube wall overheating and even tube rupture, directly threatening the safety of heating surfaces. To address this, this paper constructs a thermal-hydraulic-stress multi-physics coupled simulation model for a 1000 MW ultra-supercritical boiler water wall using the Modelica language. It analyzed three typical operating conditions, BMCR, 75% THA, and 50% THA, to investigate the coupled distribution of hydrodynamic characteristics and stress-strain. The results show that mechanical stress decreases linearly with load reduction, thermal stress peaks due to enhanced phase-change heat transfer, and strain distribution is dominated by thermal expansion. The equivalent stress is controlled by mechanical stress, exhibiting a trend of first decreasing and then increasing with height. The conclusions indicate that the essential causes of increased local failure risk are the coupling effect of flow distribution imbalance and non-uniform heat load under low load. This study provides a theoretical basis for optimization design.

Keywords: ultra-supercritical boiler,thermal-hydraulic-stress coupling, Modelica

1 Introduction

Ultra-supercritical circulating fluidized bed boilers have gradually replaced pulverized coal boilers as the mainstay of thermal power units, owing to their uniform heat flux distribution, low thermal stress, and operational safety (Cheng et al. 2025). Their wide-load and off-design operating characteristics are essential for ensuring power grid security and efficient energy utilization (Li et al. 2026). However, under low-load operation, the drastic changes in the thermophysical properties of the working fluid and intensified combustion instability can readily lead to flow and heat transfer instabilities in the water wall tubes, and even tube burst (Zhang et al. 2011). In existing thermal-hydraulic-mechanical coupled analyses, the application of the Modelica language for real-time coupling has not yet been fully exploited. This paper develops a thermal-hydraulic-mechanical multi-physics coupled simulation model of the water wall based on Modelica, employing flow and heat transfer as well as stress–strain correlations validated by experimental data. The aim is to provide a simulation foundation for the safety assessment of heating surfaces in ultra-supercritical boilers under various operating conditions.

Extensive research has been conducted on the above issues. Shao et al. developed a flow–combustion coupled model of water walls in Matlab and analyzed the flow and heat transfer characteristics of a 350 MW boiler under 75% supercritical and 50% subcritical peak-load conditions, revealing a higher overheating risk under the subcritical peak condition (Shao et al. 2026). Wang et al. established a segmented dynamic model of a 660 MW supercritical boiler and investigated the effects of feedwater temperature, heat load, and flow step changes on the outlet steam temperature at 50% THA, yet an in-depth analysis of full-load conditions was not performed (Wang et al. 2025). Li et al. focused on the impact of a sudden flow reduction in a single tube on peak temperature and stress under BMCR conditions, without extending to other loads (Li et al. 2025). Regarding heat transfer experiments, Wang et al. tested an inclined smooth tube under supercritical conditions, verified the safety of the metal temperature within the range of 30%–90% BMCR, and further demonstrated the significant influence of pressure and heat flux on the inner wall temperature of supercritical tubes; however, the effect of flow variation was not discussed (Wang et al. 2018).

Overall, research on multi-condition behavior and coupled hydrodynamic-stress analysis of ultra-supercritical boilers remains insufficient. Therefore, this paper develops a multi-physics coupled simulation model based on the Modelica language that

simultaneously solves the hydrodynamics, heat transfer, and tube wall stress–strain inside water wall tubes. The Modelica language is an equation-based, object-oriented modeling language that inherently supports multi-physics co-simulation. With its acausal modeling, declarative programming, and strong real-time simulation capabilities, it significantly improves the modeling efficiency and accuracy of complex coupled systems (Nabila et al. 2026; Dehghani et al. 2026). Taking a 1000 MW ultra-supercritical boiler as the study object, the flow, heat transfer, and stress distributions under three typical operating conditions-BMCR, 75% THA, and 50% THA-are analyzed. The potential failure risk zones and mechanisms under low-load conditions are revealed, and corresponding engineering optimization recommendations are proposed.

2 Overview of the Boiler and Water Wall

2.1 Boiler heat load distribution

The research object of this paper is a 1000 MW ultra-supercritical circulating fluidized bed boiler, which adopts a π-type layout. The furnace has a cross-section of 35.91 m × 35.91 m and a height of 120.45 m. The maximum continuous evaporation rate of the boiler is 2980 t/h, with a main steam outlet pressure of 35.42 MPa and a feedwater temperature of 634.15 K. The specific structural diagram of the boiler is shown in Figure 1.

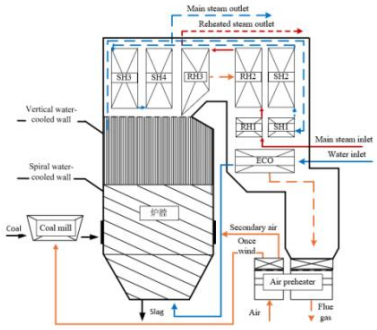


Figure 1. *Boiler structure schematic*

The main thermal parameters of this 1000MW ultra-supercritical boiler under BMCR, 75% THA and 30% THA operating conditions are shown in the table below.

Table 1. Main thermal parameters of the boiler

Parameters	*BMCR*	*75%THA*	*50%THA*
Flow rate/(t/h)	2981.02	2527.2	1776.99
Inlet temperature /°C	361	343.7	331.5
Inlet pressure /MPa	35.84	24.5	17.46
Outlet temperature /°C	493.5	441.6	357.3

Based on experimental data, the heat flux distribution along the furnace height and the heat flux non-uniformity coefficient of the ultra-supercritical boiler were obtained, as shown in **Figure 2** and **Figure 3**, respectively. The heat flux along the furnace height increases with the rise of the operating load, and the variation pattern is consistent with the actual boiler operating conditions. This heat load distribution serves as the heat flux boundary condition for the water wall model.

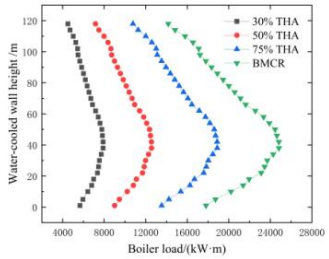


Figure 2. The heat load is distributed along the height of the boiler

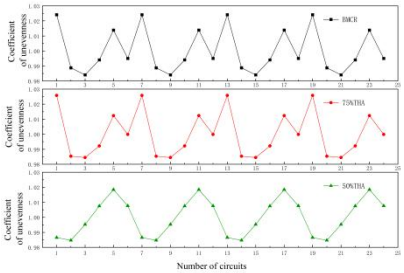


Figure 3. The curve of the lateral thermal load unevenness coefficient of the boiler

2.2 Introduction to Water-cooled Tube

The boiler water wall is constructed with a combination of internally rifled tubes and vertical tubes. The upper furnace adopts a vertical arrangement, while the lower furnace employs a spiral arrangement with a spiral inclination angle of 22.61°. A staged mixing header is arranged between the upper and lower furnaces. The spiral tube panels spiral upward from the inlet of the ash hopper to the intermediate mixing header of the water wall. The water wall comprises a total of 1200 tubes, with 300 tubes on each of the front, rear, left, and right walls of the furnace. All tubes are internally rifled tubes with dimensions of 46.7 mm × 7.3 mm and a tube spacing of 48 mm. The tube material is 12Cr1MoVG, which has an allowable temperature of 853 K, meeting the multi-condition operating requirements of the boiler. The specific thermophysical properties of this material are shown in the table below.

The feedwater of the water wall, after being heated by the economizer, enters the water wall inlet and flows upward along the furnace direction, passing sequentially through the ash hopper, the spiral tube water wall in the lower furnace, and the vertical tube water wall in the upper furnace. Since the number of water wall tube segments is large and their flow trends are similar, performing hydraulic calculations for all tube segments would be computationally too intensive. To ensure both

DOI
10.3384/ecp223317

efficiency and accuracy of the water wall model computation, this paper divides the water wall tubes into circuits: each of the four walls is divided into 6 circuits, with each circuit containing 50 tubes, resulting in a total of 24 circuits. Each circuit is discretized into 30 nodes along the flow direction of the working fluid. The hydrodynamic distributions of the tube segments at elevations of 20 m, 40 m, 60 m, and 80 m are selected as the research objects.

3 Water dynamic model of water-cooled tube

The hydrodynamic and heat transfer calculation within the water wall tubes is critical to the model computation. Based on the three conservation laws and the heat transfer equation for circular tube walls, the parameters such as working fluid properties and wall temperature distribution at each node of each circuit are calculated. Subsequently, the obtained thermal data at each node of the tube segment are input into the stress analysis equations, thereby yielding the stress distribution of the water wall tube segments.

3.1 Discrete method for water-cooled tube

The water wall tubes are discretized using the finite volume method (FVM) along the flow direction of the working fluid. The tubes are divided into multiple control volumes of equal length along the tube length direction, and a one-dimensional discretized equation is obtained by integrating over each control volume. For each control volume, the water wall tube model is established based on the conservation of mass, energy, and momentum, and is finally solved by an iterative method. The thermophysical properties of the working fluid within each control volume are processed using the central differencing scheme and are assigned to the center of the control volume. The discretization methods for single-phase and two-phase flow are identical within each control volume; therefore, only the conservation equations for the single-phase working fluid are given below.

3.2 Cold-wall tube mathematical model

The following assumptions are proposed in this paper to address the stability issue of the working fluid inside the water wall tubes during operation:

(1) Neglecting fluctuations under actual boiler operating conditions, it is assumed that the axial heat flux distribution of the tube remains stable under various working conditions.

(2) A one-dimensional heat transfer assumption and material homogeneity are adopted, and the water wall tube material is regarded as isotropic.

(3) Only the radial temperature variation along the furnace height is considered, while the axial heat conduction of the tube material is neglected.

(4) Based on the thermodynamic equilibrium assumption, the two-phase flow is treated as a fully mixed homogeneous medium.

Based on the above assumptions, the one-dimensional mathematical model inside the water wall tubes is formulated as follows:

Mass conservation equation:

$$\frac{d\rho_i}{dt}V_i = \dot{m}_{i-1} - \dot{m}_i \tag{1}$$

Where i is the control volume index;ρ is the density of the control volume, kg/m³; $\dot{m}$ is the mass flow rate of the control volume, kg/s;V s the volume of the control volume, m³.

Energy conservation equation:

$$\frac{dh_i}{dt}\dot{m}_i = \frac{dp_i}{dt}V_i + \dot{m}_{i-1}h_{i-1} - \dot{m}_i h_i + Q_i \tag{2}$$

Where h is the enthalpy of the working fluid in the control volume, kJ/kg;Q is the external heat input to the control volume, kJ.

Momentum conservation equation:

$$\frac{dh_i}{dt}\dot{m}_i = \frac{dp_i}{dt}V_i + \dot{m}_{i-1}h_{i-1} - \dot{m}_i h_i + Q_i \tag{3}$$

Where L is the length of the control volume, m;μ is the fluid velocity in the control volume, m/s;Δp_f is the frictional pressure drop, Pa; Δp_l is the local pressure drop, Pa; Δp_g is the gravitational pressure drop, Pa.

The thermophysical properties of the fluid in each control volume are calculated using the central differencing scheme. For the flow resistance and thermal calculation of fluids in different states, commonly used classical empirical correlations are adopted. The calculation methods for the frictional pressure drop, local pressure drop, and gravitational pressure drop are given below, respectively.

For single-phase water or supercritical water, the Colebrook-White equation is used for calculation:

$$\Delta p_f = \frac{1}{4\left[\lg\left(\frac{K_i}{3.7} + \frac{5.75}{Re_i^{0.9}}\right)\right]^2}\frac{L_i}{2d_i}\frac{\dot{m}_i^{\,2}}{A_i^{\,2}\rho_i} \tag{4}$$

where K is the relative roughness;d_i is the hydraulic diameter of the control volume, m;A_i is the cross-sectional area of the control volume, m²;ρ_i is the density of the fluid in the control volume, kg/m³.

$$\Delta P_l = \xi\frac{\rho_i v_i^2}{2} \tag{5}$$

where ζ is the flow resistance coefficient;v_i is the fluid velocity in the control volume, m/s.

For two-phase flow, the Müller-Steinhagen and Heck correlation is used for calculation:

$$\Delta p_{\mathrm{f}} = \frac{0.3164}{2d_i^{1.25}}\left[\left(\frac{\mu_{i,liq}^{0.25}}{\rho_{i,liq}} + 2x_i\left(\frac{\mu_{i,vap}^{0.25}}{\rho_{i,vap}} - \frac{\mu_{i,liq}^{0.25}}{\rho_{i,liq}}\right)\right)\right. \tag{6}$$

$$\left.\left(1-x_i\right)^{1/3} + \frac{\mu_{i,vap}^{0.25}}{\rho_{i,vap}}x_i^{\,3}\right]\dot{m}_i^{\,7/4}l_i$$

Where x_i is the quality of the working fluid in the control volume;$\rho_{i,vap}$ is the density of the vapor-phase working fluid in the control volume, kg/m³;$\rho_{i,liq}$ is the density of the liquid-phase working fluid in the control volume, kg/m³;$\mu_{i,vap}$ is the dynamic viscosity of the vapor-phase working fluid in the control volume, Pa·s; $\mu_{i,liq}$ is the dynamic viscosity of the liquid-phase working fluid in the control volume, Pa·s.

$$\Delta P_l = \xi \frac{\dot{m}_i^{\,2}}{2\rho_{i,liq}}\left[1 + x_i\left(\frac{\rho_{i,liq}}{\rho_{i,vap}} - 1\right)\right] \tag{7}$$

The gravitational pressure drop is determined by the density of the working fluid and the height difference between the inlet and outlet. Its calculation formula is given by:

$$\Delta P_g = (x_i\rho_{i,vap} + (1 - x_i)\rho_{i,liq})gL_i \sin\theta_i \tag{8}$$

where g is the fluid acceleration;L_i is the length of the control volume;θ is the angle between the tube segment and the horizontal plane. If the tube segment is vertically upward, sinθ_i is 1。

3.3 Stress-strain model

The variation of the working fluid pressure inside the water wall tubes has a significant influence on the mechanical stress of the tube wall, while the temperature difference between the inner and outer surfaces of the tube wall is the main cause of thermal stress. Therefore, after the hydrodynamic model of the water wall tubes reaches a steady state, the distribution of the working fluid pressure and temperature can be obtained. Subsequently, by combining the stress − strain relationship equations, the radial stress distribution and strain response along the water wall tube can be derived, providing a basis for the subsequent structural strength evaluation of the water wall tubes.

Circumferential stress equation:

$$\sigma_\theta = \frac{p_i}{K^2 - 1}\left(1 + \frac{R_0^2}{r^2}\right) \tag{9}$$

Axial stress calculation formula:

$$\sigma_z = p_i\left(\frac{1}{K^2 - 1}\right) \tag{10}$$

Radial stress calculation formula:

$$\sigma_r = \frac{p_i}{K^2 - 1}\left(1 - \frac{R_0^2}{r^2}\right) \tag{11}$$

σ_θ is circumferential stress, σ_z is axial stress, σ_r is radial stress, p_i is internal pressure, K is the radial ratio.

Circular axial thermal stress calculation formula:

$$\sigma_\theta^t = \frac{E\alpha_t\Delta t}{2(1-\mu)\ln\frac{r_o}{r_i}}\left[\begin{array}{c} 1 - \ln\frac{r_o}{r} - \\ \frac{r_i^2\ln\frac{r_o}{r_i}}{r_o^2 - r_i^2}\left(1 + \frac{r_o^2}{r^2}\right) \end{array}\right] \tag{12}$$

Axial thermal stress calculation formula:

$$\sigma_z^t = \frac{E\alpha_t\Delta t}{2(1-\mu)\ln\frac{r_o}{r_i}}\left[\begin{array}{c} 1 - 2\ln\frac{r_o}{r} - \\ \frac{2r_i^2\ln\frac{r_o}{r_i}}{r_o^2 - r_i^2} \end{array}\right] \tag{13}$$

Radial thermal stress calculation formula:

$$\sigma_r^t = \frac{\alpha E\Delta T}{2(1-\mu)\ln\left(\frac{r_o}{r_i}\right)}\left[\begin{array}{c} -\ln\frac{r_o}{r} - \\ \frac{r_i^2\ln\left(\frac{r_o}{r_i}\right)}{r_o^2 - r_i^2}\left(1 - \frac{r_o^2}{r^2}\right) \end{array}\right] \tag{14}$$

$\sigma^t\theta$ is circular thermal stress, $\sigma^t z$ *i*s axial thermal stress, $\sigma^t r$ is axial thermal stress, E is the elastic modulus, α_t is the radial of linear expansion.

Stress calculation formula:

$$\varepsilon_\theta = \frac{1}{E}\left[\sigma_\theta - \mu\left(\sigma_z + \sigma_r\right)\right] + \alpha_T\Delta T \tag{15}$$

$$\varepsilon_z = \frac{1}{E}\left[\sigma_z - \mu\left(\sigma_\theta + \sigma_r\right)\right] + \alpha_T\Delta T \tag{16}$$

$$\varepsilon_r = \frac{1}{E}\left[\sigma_r - \mu\left(\sigma_z + \sigma_\theta\right)\right] + \alpha_T\Delta T \tag{17}$$

ε_θ is circumferential strain, ε_z is axial strain, ε_r is radial strain, ΔT is the temperature difference between inside and outside the diameter.

The equivalent stress is calculated using the Von Mises stress criterion. This method can accurately compute the combined stress intensity of the material, converting the complex three-dimensional stress state of the water wall tube into a single equivalent stress value. It enables a unified quantification of the internal pressure stress and thermal stress, and can comprehensively describe the actual stress condition at a given point on the tube wall.

$$\sigma_{\text{eq}} = \sqrt{\frac{(\sigma_r - \sigma_\theta)^2 + (\sigma_\theta - \sigma_z)^2 + (\sigma_z - \sigma_r)^2}{2}} \tag{18}$$

3.4 Model validation

To verify the accuracy of the developed water wall tube model, the calculated values from the model are compared with the measured data. Under the same boiler heat load and tube structure conditions, the simulation results are compared with the measured data, as shown in Figure 4. The results indicate that the simulated and measured working fluid temperature distributions along the furnace height exhibit consistent trends, with a maximum relative error of 2.74%. This result verifies that the model developed in this paper possesses sufficiently high accuracy and is adequate for the subsequent hydrodynamic and stress analysis of the water wall.

DOI
10.3384/ecp223317

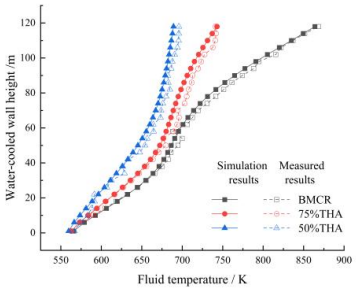


Figure 4. The temperature distribution of the fluid along the height of the furnace

4 Simulation results and analysis

4.1 Fluid temperature and pipe wall temperature

Figure 5 presents the mass flow distribution of each water wall circuit. The mass flow deviation among circuits is small under both BMCR and 75% THA conditions. Under the BMCR condition, the mass flow distribution ranges between 35.04 kg/s and 34.98 kg/s, with a maximum deviation of 1.2%; under the 75% THA condition, it ranges between 29.19 kg/s and 29.31 kg/s, with a maximum deviation of only 0.4%. Under the 50% THA condition, the mass flow distribution ranges from 20.43 kg/s to 19.01 kg/s, and the maximum deviation reaches 7.2%.

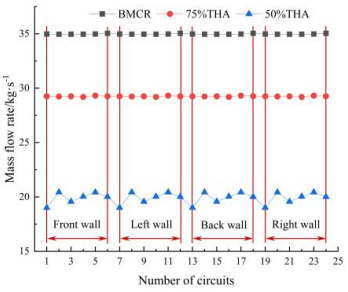


Figure 5. Fluid inlet mass flow curve

Figure 6 shows the quality distribution curves along the furnace height for Circuit 1 and Circuit 3. Under the BMCR condition, both circuits remain in the single-phase supercritical state throughout, with no two-phase region appearing, and the quality equals 1. Under the 75% THA condition, both circuits enter the two-phase region at an elevation of 38 m, with corresponding qualities of 0.123 and 0.059, respectively, and subsequently enter the superheated region simultaneously at 62 m. Under the 50% THA condition, the two circuits enter the two-phase region at 42 m, with qualities of 0.077 and 0.007, respectively. Thereafter, Circuit 1 achieves complete evaporation at 90 m, whereas Circuit 3 does so only at 98 m. The mass flow rate of Circuit 1 is approximately 6.96% lower than that of Circuit 3. Under the same heat load condition, the smaller flow rate causes a more rapid increase in the enthalpy of the working fluid along the flow path and a higher rate of quality increase; consequently, Circuit 1 reaches the fully evaporated state earlier.

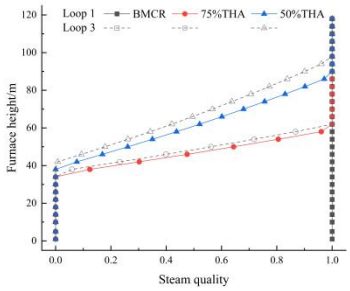


Figure 6. Quality distribution curves along the furnace height for Circuit 1 and Circuit 3.

Figure 7 shows the distribution curves of the fluid and tube wall temperatures along the furnace height under the BMCR condition. The working fluid is in the supercritical state, and both the fluid and tube wall temperatures increase with the rise of the heat flux. In the upper and lower water wall circuits, the fluid temperature and the metal temperature exhibit a nearly linear increase with the furnace height, ultimately reaching their maximum values at the outlet. In the lower water wall, the maximum fluid temperature is 679.99 K, and the maximum temperatures of the inner wall, mid-wall, and outer wall are 683.62 K, 689.82 K, and 696.01 K, respectively. At an elevation of 62 m, near the junction between the upper and lower water walls, both the fluid temperature and the tube wall temperature exhibit a distinct step change. This is because the lower furnace employs spiral tube panels, whereas the upper furnace adopts vertical tube panels, resulting in differences in the internal flow and heat transfer characteristics between the two. Moreover, a mixing header is installed at the junction; as the working fluid is mixed, the temperature distribution changes, leading to the abrupt temperature variation. In the upper water wall, the maximum fluid temperature is 718.95 K, and the maximum temperatures of the inner wall, mid-wall, and outer wall are 736.82 K, 727.04 K, and 746.59 K, respectively. Although the heat flux in the upper furnace gradually decreases along the height, the tube wall temperature still rises slowly because the fluid temperature is already at a high level and the heat transfer coefficient decreases.

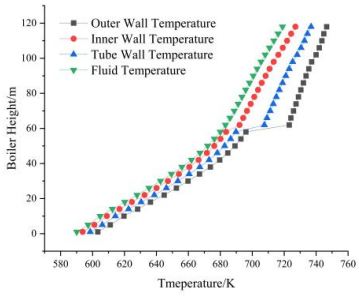


Figure 7. BMCR operating condition fluid and pipe wall temperature curves

Figure 8. shows the distribution curves of the fluid temperature and tube wall temperature along the furnace height under the 75% THA condition. In the lower water wall, the fluid temperature increases with the furnace height below 34 m, but the heating rate of the fluid decreases significantly near 34 m. This is because the working fluid enters the two-phase region at this point, where the specific heat capacity increases sharply, slowing the temperature rise after heat absorption. Meanwhile, the wall heat transfer coefficient increases notably, reducing the temperature difference between the wall and the fluid, leading to a slight decrease in the metal temperature. In the lower water wall, the maximum fluid temperature is 589.81 K, and the maximum temperatures of the inner wall, mid-wall, and outer wall are 593.41 K, 597.96 K, and 602.49 K, respectively. After the fluid enters the upper water wall, the heat transfer coefficient increases, causing the fluid temperature and tube wall temperature to rise rapidly and reach their maximum values at the outlet. In the upper water wall, the maximum fluid temperature is 681.22 K, and the maximum temperatures of the inner wall, mid-wall, and outer wall are 689.85 K, 704.81 K, and 719.76 K, respectively.

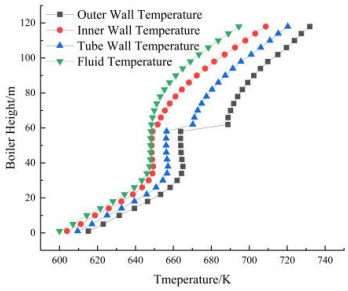


Figure 8. 75% operating condition fluid and pipe wall temperature curve

Figure 9 shows the distribution curves of the fluid temperature and tube wall temperature along the furnace height under the 50% THA condition. In the lower water wall, the working fluid enters the two-phase region at approximately 38 m. The temperature increases slowly and remains essentially at the saturation temperature, showing no significant further rise with height. The maximum fluid temperature in the lower water wall is 626.46 K, and the maximum temperatures of the inner wall, mid-wall, and outer wall are 627.79 K, 631.84 K, and 635.91 K, respectively. After the fluid enters the upper water wall, it evaporates completely at around 86 m. Subsequently, both the fluid temperature and the metal temperature increase monotonically with the furnace height. In the upper water wall, the maximum fluid temperature is 676.56 K, and the maximum temperatures of the inner wall, mid-wall, and outer wall are 701.66 K, 695.23 K, and 708.09 K, respectively.

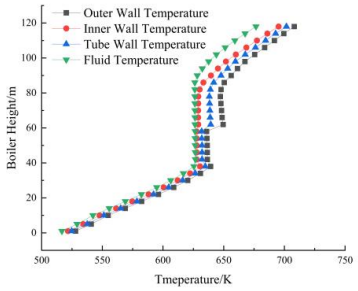


Figure 9. 50% operating condition fluid and pipe wall temperature curve

Figure 10 presents the temperature distribution of the working fluid at the outlet of the lower furnace circuits of the water wall. Under the BMCR condition, the maximum outlet temperature is 679.99 K, and the minimum outlet temperature is 676.17 K. As the operating load decreases, the outlet fluid temperature of the lower circuits shows an overall downward trend. Under the 75% THA condition, the outlet fluid temperature ranges between 648.15 K and 647.85 K. Under the 50% THA condition, the outlet fluid temperature ranges between 626.45 K and 625.51 K. This is because the water wall tubes in the lower furnace are arranged in an inclined spiral upward manner. Although the local heat load in the lower furnace varies, the combined effect of the heating area and the inclined flow enhances the heat transfer of the working fluid. The outlet temperature deviation under all operating conditions is small, indicating that the lower furnace water wall possesses good temperature uniformity.

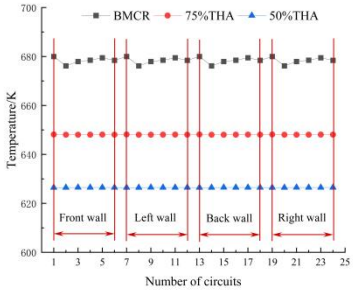


Figure 10. The outlet temperatures of the fluid in each loop of the lower furnace chamber

Figure 11 presents the temperature distribution of the working fluid at the outlet of the upper furnace circuits of the water wall. Compared with the lower furnace, the outlet temperature deviation among the circuits in the upper furnace is significantly larger. Under the BMCR and 75% THA conditions, the distribution trend of the temperature deviation is essentially consistent with the non-uniformity coefficient of the transverse heat load of the boiler. Under the BMCR condition, the outlet temperature ranges between 712.63 K and 718.95 K; under the 75% THA condition, it ranges between 685.04 K and 694.50 K. Under the 50% THA condition, the temperature deviation shows a

DOI
10.3384/ecp223317

negative correlation trend with the mass flow rate of the circuits, with the outlet temperature ranging between 660.72 K and 679.65 K. The maximum temperature deviation is 18.93 K, which falls within the allowable range for safe operation in engineering design.

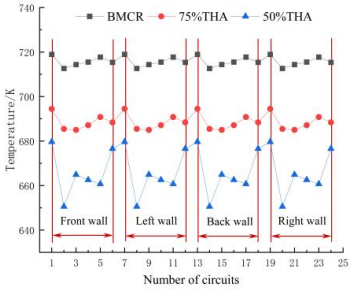


Figure 11. The outlet temperatures of the fluid in each loop of the upper furnace section

4.2 BMCR Operating Condition Stress Analysis

Because the water wall tube has a relatively thin wall and is subjected to high internal pressure over long-term steady-state operation, the stress gradient through the wall thickness is negligible. Therefore, the stress at the inner wall node can represent the critical stress level across the entire wall thickness (Rouse et al. 2013).

Figure 12 shows the distribution curves of the radial stress in each circuit under the BMCR condition. The radial stress is compressive, representing the compressive stress under the action of internal pressure. As the furnace height increases, the radial stress in each circuit gradually decreases. Taking Circuit 1 as an example, the radial stresses at elevations of 20 m, 40 m, 60 m, and 80 m are -35.81 MPa, -35.70 MPa, -35.58 MPa, and -35.51 MPa, respectively. From the perspective of the variation along the flow path, the radial stress decreases by 0.11 MPa (approximately 0.31%) from 20 m to 40 m, and by 0.07 MPa (approximately 0.18%) from 60 m to 80 m. The reason is that the lower water wall tube segments are long, the working fluid density is high, and both the frictional pressure drop and the gravitational pressure drop along the path are relatively large, leading to a more pronounced decrease in radial stress. The upper water wall tube segments are short, the density is low, and the overall pressure drop is relatively smaller, so the decrease slows down. Furthermore, with increasing height, the fluctuation of radial stress among the circuits is significantly reduced, thereby improving the pressure non-uniformity between circuits.

Figure 13 shows the distribution curves of the circumferential stress in each circuit under the BMCR condition. The distribution trend of the circumferential stress is similar to that of the radial stress, but the magnitudes are significantly larger. Taking Circuit 1 as an example, the circumferential stresses at elevations of 20 m, 40 m, 60 m, and 80 m are 242.90 MPa, 242.16 MPa, 241.33 MPa, and 240.89 MPa, respectively. The stress decreases by 0.74 MPa (approximately 0.31%) from 20 m to 40 m, and by 0.44 MPa (approximately 0.18%) from 60 m to 80 m. Compared with the yield strength of the material, these values remain within the safe range. The stress distribution among the circuits is uniform.

Figure 14 shows the distribution curves of the axial stress in each circuit under the BMCR condition. Taking Circuit 1 as an example, the axial stresses at elevations of 20 m, 40 m, 60 m, and 80 m are 103.55 MPa, 103.23 MPa, 102.88 MPa, and 102.69 MPa, respectively. The stress decreases by 0.32 MPa (approximately 0.31%) from 20 m to 40 m, and by 0.19 MPa (approximately 0.18%) from 60 m to 80 m. The magnitude of the axial stress lies between those of the radial and circumferential stresses, and its distribution characteristics are consistent with both.

In summary, under the BMCR condition, the radial, axial, and circumferential stresses of the water wall tubes in each circuit are uniformly distributed along the flow path, the variation trend along the path is consistent, and the overall stress state is stable, thus satisfying the strength design requirements for the water wall tubes.

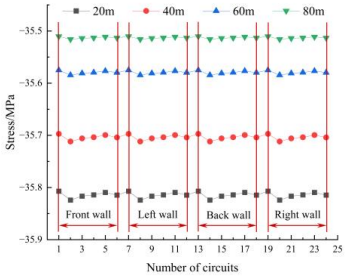


Figure 12. Radial stresses of different heights in each BMCR circuit

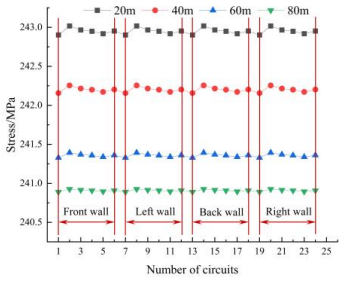


Figure 13. The circumferential stresses at different heights of each BMCR circuit

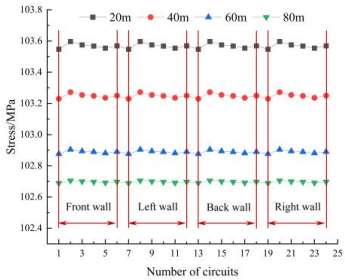


Figure 14. Axial stresses of different heights in each of the BMCR circuits

Figure 15 shows the distribution curves of the thermal stress in each circuit under the BMCR condition. The thermal stress, caused by the temperature difference between the inner and outer walls of the tube, exhibits distinct sectional characteristics along the furnace height. Taking Circuit 1 as an example, at elevations of 20 m and 40 m, the thermal stresses are 23.63 MPa and 25.38 MPa, respectively, representing an increase of approximately 7.41%. In the lower water wall section, the overall thermal stress level is relatively low and changes gently. After entering the upper water wall, the thermal stress rises rapidly, reaching 57.47 MPa at 60 m, indicating a significantly increased temperature difference between the inner and outer tube walls in this region. At 80 m, the thermal stress decreases to 50.23 MPa, a reduction of approximately 12.6% compared to the value at 60 m. This is because the heat load in the upper furnace decreases along the height, and the heat transfer coefficient between the working fluid and the tube wall increases, leading to a gradual reduction in the tube wall temperature difference. Overall, the variation of thermal stress along the height is basically consistent across all circuits. Although the deviation of the inner wall thermal stress at 60 m is slightly amplified, the overall distribution trend remains the same.

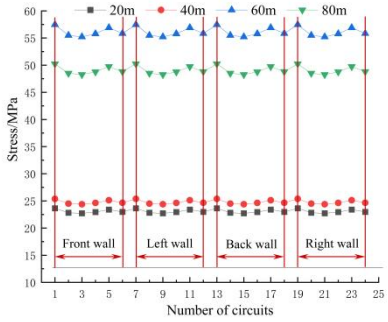


Figure 15. The different thermal stresses at various heights of each BMCR circuit

Figure 16 shows the distribution curves of the equivalent stress in each circuit under the BMCR condition. The equivalent stress comprehensively considers the mechanical stress caused by internal pressure and the thermal stress induced by the temperature difference across the tube wall. Taking Circuit 1 as an example, the equivalent stresses at elevations of 20 m, 40 m, 60 m, and 80 m are 221.21 MPa, 219.02 MPa, 192.19 MPa, and 197.46 MPa, respectively. From 20 m to 40 m, the equivalent stress decreases by approximately 0.99%, while from 60 m to 80 m it increases by approximately 2.74%, exhibiting a trend of first decreasing and then increasing. Overall, the equivalent stress level in the upper water wall is significantly lower than that in the lower water wall, and the stress distribution among the circuits is uniform with good consistency.

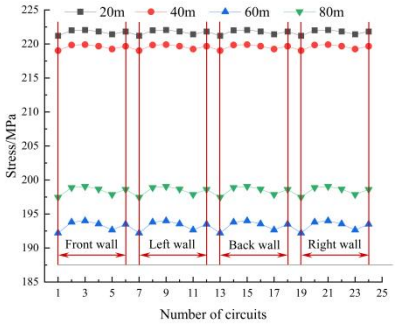


Figure 16. The equivalent stresses at different heights of each BMCR circuit

Figure 17 shows the strain distribution in the circuits at different heights under the BMCR condition. The strain represents the total equivalent strain value reflecting the geometric deformation of the tube wall, and its variation along the path is highly consistent with the thermal stress distribution. Taking Circuit 1 as an example, in the lower water wall, the strain at 20 m is 0.174, which increases to 0.187 at 40 m, an increase of approximately 7.47%. In the upper water wall, the strain is 0.423 at 60 m and decreases to 0.369 at 80 m, a reduction of approximately 12.77%.

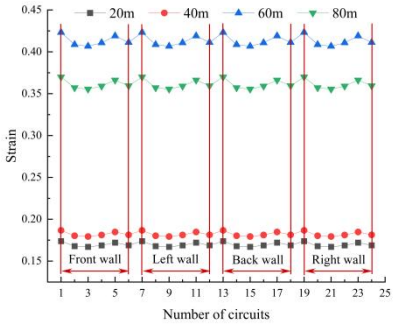


Figure 17. The different levels of strain in each of the BMCR circuits

4.3 Stress analysis of 75% THA

Among the three principal mechanical stress components of the water wall under the operating conditions in this study, the circumferential stress is significantly higher than the axial stress and the radial stress, and is thus the key component governing the strength of the water wall. Therefore, the following analysis focuses on the circumferential stress under various operating conditions.

Figure 18 shows the circumferential stress distribution of each circuit under the 75% THA condition. It can be seen that the circumferential stress of each circuit decreases in a nearly linear and monotonic manner with increasing height, and the degree of scatter of the circumferential stress among different circuits gradually reduces along the height. Taking the four characteristic heights in the figure as an example, for Circuit 1, the circumferential stresses at the lower water wall at elevations of 20 m, 40 m, 60 m, and 80 m are

DOI
10.3384/ecp223317

152.03 MPa, 151.42 MPa, 150.87 MPa, and 150.63 MPa, respectively; for Circuit 3, they are 152.07 MPa, 151.46 MPa, 150.89 MPa, and 150.64 MPa, respectively. For the two circuits, the circumferential stress decreases by an average of approximately 0.4% from 20 m to 40 m, and by approximately 0.17% from 60 m to 80 m. Compared with the BMCR condition, the above stresses are systematically reduced by approximately 37.4% under the 75% THA condition, with a uniform decrease along the height direction and a consistent stress variation trend among the circuits.

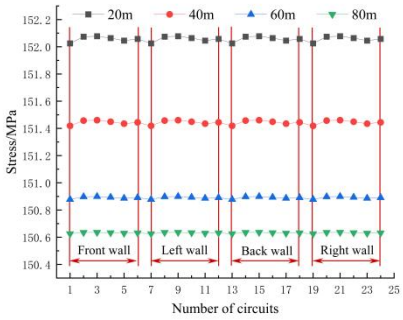


Figure 18. 75% THA - Radial stresses at different heights in each circuit

Figure 19 shows the distribution curves of the thermal stress in each circuit under the 75% THA condition. Compared with the BMCR condition, the 75% THA condition also exhibits distinct two-stage distribution characteristics. Taking the four characteristic heights in the figure as an example, for Circuit 1, the thermal stresses are 28.08 MPa, 30.15 MPa, 68.27 MPa, and 59.69 MPa, respectively; for Circuit 3, they are 26.95 MPa, 28.92 MPa, 65.52 MPa, and 57.27 MPa, respectively. From the data, it can be seen that for the two circuits, the thermal stress increases by an average of approximately 7.33% from 20 m to 40 m, and decreases by approximately 12.58% from 60 m to 80 m. In terms of the overall thermal stress level, compared with the BMCR condition, the thermal stresses at all characteristic heights increase under the 75% THA condition, with an increase of approximately 18.8% for each circuit.

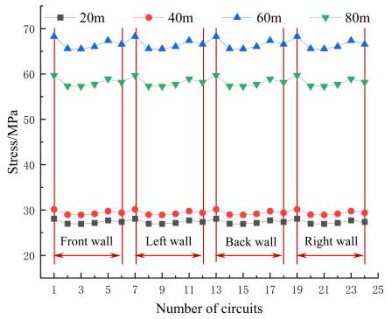


Figure 19. 75% THA - Different height thermal stresses in each circuit

Figure 20 shows the distribution curves of the equivalent stress in each circuit under the 75% THA condition. The variation trend along the circuits and the height under the 75% THA condition is consistent with that under the BMCR condition. Taking Circuit 1 as an example, from 20 m to 40 m, the equivalent stress decreases from 127.52 MPa to 125.27 MPa, a reduction of approximately 1.77%; from 60 m to 80 m, the equivalent stress increases from 97.00 MPa to 102.43 MPa, an increase of approximately 5.59%. For Circuit 3, from 20 m to 40 m, the equivalent stress decreases from 128.48 MPa to 126.28 MPa, a reduction of approximately 1.71%; from 60 m to 80 m, it increases from 98.79 MPa to 104.10 MPa, an increase of approximately 5.37%. The increasing ratios of the two circuits are consistent with those under the BMCR condition. Overall, the equivalent stress of each circuit under the 75% THA condition exhibits a decreasing trend, with an overall reduction of approximately 42.3% compared with the BMCR condition.

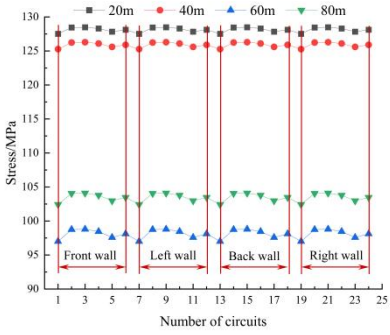


Figure 20. 75% THA - Different height equivalent stresses in each circuit

Figure 21 shows the distribution curves of the strain in each circuit under the 75% THA condition. The strain distribution trend along the circuits and the height is consistent with that under the BMCR condition, exhibiting a two-stage pattern with lower values in the lower water wall and higher values in the upper water wall. Taking Circuit 1 as an example, from 20 m to 40 m, the strain increases from 0.206 to 0.222, an increase of approximately 7.34%; from 60 m to 80 m, it decreases from 0.503 to 0.439, a reduction of approximately 12.59%. For Circuit 3, from 20 m to 40 m, the strain increases from 0.198 to 0.213, an increase of approximately 7.33%; from 60 m to 80 m, it decreases from 0.483 to 0.422, a reduction of approximately 12.6%. The variation patterns of the two circuits are basically consistent. The increasing ratio is consistent with that under the BMCR condition. The strain in each circuit under the 75% THA condition is overall approximately 15.9% higher than that under the BMCR condition.

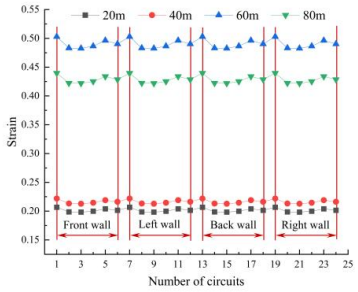


Figure 21. 75% THA - Strains at different heights in each circuit

4.4 Stress analysis of 50% THA

Figure 22 shows the distribution curves of the circumferential stress at different heights in each circuit under the 50% THA condition. The variation trend along the circuits and the height is consistent with that under the BMCR condition. Taking Circuit 1 as an example, from 20 m to 40 m, the circumferential stress decreases from 117.81 MPa to 117.07 MPa, a reduction of approximately 0.63%; from 60 m to 80 m, it decreases from 116.61 MPa to 116.42 MPa, a reduction of approximately 0.16%. For Circuit 3, from 20 m to 40 m, the circumferential stress decreases from 117.97 MPa to 117.20 MPa, a reduction of approximately 0.65%; from 60 m to 80 m, it decreases from 116.66 MPa to 115.46 MPa, a reduction of approximately 0.18%. The variation trend is consistent with that under the BMCR condition. Overall, the circumferential stress is approximately 47.75% lower than that under the BMCR condition.

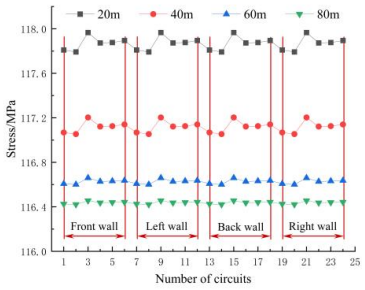


Figure 22. 50% THA - Different circumferential stresses at different heights in each circuit

Figure 23 shows the distribution of the thermal stress in each water wall circuit under the 50% THA condition. The thermal stress distribution under the 50% THA condition also exhibits distinct sectional characteristics along the furnace height. Taking Circuit 1 as an example, from 20 m to 40 m, the thermal stress increases from 15.56 MPa to 16.69 MPa, an increase of 7.33%; from 60 m to 80 m, it decreases from 37.82 MPa to 33.06 MPa, a reduction of approximately 12.59%. For Circuit 3, from 20 m to 40 m, it increases from 15.04 MPa to 16.14 MPa, an increase of approximately 1.10%; from 60 m to 80 m, it decreases from 36.57 MPa to 31.96 MPa, a reduction of approximately 12.59%. Compared with the BMCR condition, the overall thermal stress distribution under the 50% THA condition decreases by approximately 34.19%. The decrease in thermal stress from 60 m to 80 m is related to the two-phase state of the working fluid: in the two-phase region, nucleate boiling keeps the inner wall temperature close to the saturation temperature of the working fluid and remains stable, while the outer wall temperature varies with the furnace heat flux, resulting in an increased radial temperature difference and a correspondingly high level of thermal stress. Therefore, although the thermal stress shows a decreasing trend under this condition, the vicinity of this region remains a high-risk zone for water wall tube burst.

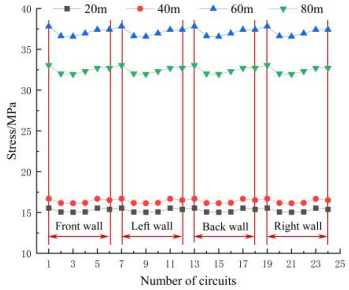


Figure 23. 50% THA - Different height thermal stresses in each circuit

Figure 24 shows the distribution of the equivalent stress in each water wall circuit under the 50% THA condition. It can be seen by comparison that, similar to the thermal stress under this condition, the equivalent stress distribution under the 50% THA condition also exhibits distinct sectional characteristics along the furnace height. Taking Circuit 1 as an example, from 20 m to 40 m, the equivalent stress decreases from 103.89 MPa to 102.21 MPa, a reduction of approximately 1.61%; from 60 m to 80 m, it increases from 85.24 MPa to 88.62 MPa, an increase of approximately 3.96%. For Circuit 3, from 20 m to 40 m, the equivalent stress decreases from 104.47 MPa to 102.80 MPa, a reduction of approximately 1.59%; from 60 m to 80 m, it increases from 86.22 MPa to 89.48 MPa, an increase of approximately 3.78%. Compared with the BMCR condition, the overall equivalent stress under the 50% THA condition decreases by approximately 54.29%.

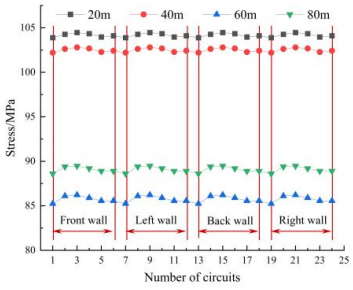


Figure 24. 50% THA - Different height equivalent stresses in each circuit

Figure 25 shows the distribution curves of the strain in each circuit under the 50% THA condition. The strain

DOI
10.3384/ecp223317

distribution under the 50% THA condition also exhibits distinct sectional characteristics along the furnace height. Taking Circuit 1 as an example, from 20 m to 40 m, the strain increases from 0.114 to 0.123, an increase of 7.34%; from 60 m to 80 m, it decreases from 0.279 to 0.243, a reduction of 12.59%. For Circuit 3, from 20 m to 40 m, the strain increases from 0.111 to 0.119; from 60 m to 80 m, it decreases from 0.269 to 0.235, an increase of 7.34%. Compared with the BMCR condition, the overall strain under the 50% THA condition decreases by approximately 34.18%.

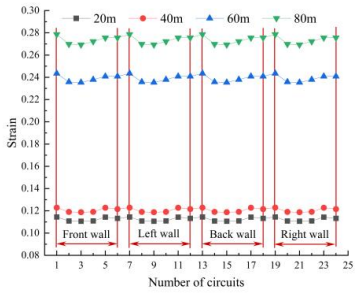


Figure 25. 50% THA - Strains at different heights of each circuit

5 Conclusion

This paper develops a coupled hydrodynamic–stress simulation model for the spiral tube water wall of a 1000 MW ultra-supercritical boiler based on the Modelica language. The flow and heat transfer characteristics as well as the stress and strain distributions of the water wall tubes under BMCR, 75% THA, and 50% THA operating conditions are investigated. The main conclusions are as follows:The uniformity of the mass flow distribution deteriorates as the load decreases. Under the BMCR and 75% THA conditions, the flow deviation among circuits is no greater than 1.2%; under the 50% THA condition, the deviation increases to 7.2%, and circuits with smaller mass flow rates enter the two-phase region and reach complete evaporation earlier.

In terms of the temperature distribution, under the BMCR condition, the fluid and tube wall temperatures increase nearly linearly along the furnace height, whereas under the 75% THA and 50% THA conditions, a saturation temperature plateau exists. A temperature step change occurs at the junction between the spiral and vertical tube panels under all three operating conditions. Owing to the enhanced heat transfer of the spiral tubes in the lower water wall, the outlet temperature difference is less than 4 K; the outlet temperature difference of the upper water wall increases with decreasing load, reaching a maximum of 18.93 K under the 50% THA condition.

The circumferential stress decreases in a nearly linear manner with increasing water wall height, and the decreasing gradient is basically consistent among circuits, reflecting the characteristic of uniform internal pressure dissipation along the flow path. The mechanical stress decreases significantly as the load decreases: compared with the BMCR condition, the circumferential stress decreases by 37.4% under the 75% THA condition and by 47.8% under the 50% THA condition, indicating a lower risk of mechanical failure at low loads.

The thermal stress distribution exhibits a two-stage pattern of lower values in the lower water wall and higher values in the upper water wall, corresponding to the furnace structure, which is mainly determined by the different heat flux and heat transfer boundary conditions of the upper and lower water walls. The thermal stress shows a non-monotonic variation with the load. Under the 75% THA condition, nucleate boiling in the two-phase region leads to an increased tube wall temperature difference, and the thermal stress increases by 18.8% compared with the BMCR condition, reaching the highest level among all operating conditions. Under the 50% THA condition, although the two-phase region is longer, the overall heat load is lower, and the thermal stress decreases by 34.2% compared with the BMCR condition.

The equivalent stress is dominated by the mechanical stress, whereas the total strain is dominated by the thermal expansion. The equivalent stress decreases systematically with decreasing load: compared with the BMCR condition, the equivalent stress and circumferential stress decrease by 42.3% and 37.4%, respectively, under the 75% THA condition, and by 54.3% and 47.75%, respectively, under the 50% THA condition. The total strain distribution is consistent with the thermal stress distribution: under the 75% THA condition, the thermal stress and strain increase by approximately 18.8% and 15.9%, respectively, whereas under the 50% THA condition, they decrease by 34.19% and 34.2%, respectively. This indicates that the thermal expansion caused by the temperature difference through the wall thickness is the decisive factor controlling the water wall strain.

Under low-load conditions, the coupling of flow imbalance and non-uniform heat load is the fundamental cause of local overheating and deformation failure. It is recommended to optimize the inlet header structure or install throttle rings to balance the mass flow distribution among water wall tubes, thereby reducing the flow deviation among circuits. Special attention should be paid to monitoring the changes in wall temperature, strain, and differential pressure in the two-phase zone and at the junction between tube panels, enabling real-time diagnosis of the dryout position. These measures can effectively identify and mitigate the coupled thermal–mechanical risks under low-load conditions.

References

Cheng, Leming, Kun Li, Qifeng Yu and Qingyu Zhang (2025). “The application of numerical methods in the design of large-size CFB boilers”. In: Applied Thermal Engineering 280.3, pp. 128243.DOI:10.1016/j.applthermaleng.2025.128243.

Dehghani, Fateme, Mohammad Reza Abbasi, Khashayar Sadeghi, Mahdi Aghaie, Ekaterina Sokolova and Majid Hosseinllu (2026). “Modular nuclear microreactors for water and power cogeneration: thermodynamic and economic analysis of sCO2-MED integration”. In: Annals of Nuclear Energy 236, pp. 112443. DOI: 10.1016/j.anucene.2026.112443.

Li, Bingying, Jiale Gao, Chenjiang Wang, Huan Han, Xingsheng Wang and Shiyuan Li (2025). “Fluid-Structure interaction analysis of a 1000 MW supercritical coal-fired boiler water-cooled wall based on multi-parameter simulation”. In: Applied Thermal Engineering 274.B, pp. 126668. DOI: 10.1016/j.applthermaleng.2025.126668.

Li, Yuanlu, Xuejiao Liu, Xi Chen, Guanwen Zhou, Hong Zhang, Yonghua Gu and Wenqi Zhong (2026). “Combustion reconstruction strategy for ultra-low-load stability in a 1000 MW ultra-supercritical tangentially fired boiler”. In: Applied Thermal Engineering 298.1, pp. 130877. DOI: 10.1016/j.applthermaleng.2026.130877.

Nabila, Umme Mahbuba, Paul Seurin, Linyu Lin and Majdi I. Radaideh (2026). “Physics-informed digital twin development of thermal energy distribution systems with active learning”. In: Applied Energy 416, pp. 127903. DOI: 10.1016/j.apenergy.2026.127903.

Rouse, J.P., M.Z. Leom, W. Sun, T.H. Hyde and A. Morris (2013). “Steady-state creep peak rupture stresses in 90° power plant pipe bends with manufacture induced cross-section dimension variations”. In: International Journal of Pressure Vessels and Piping 105-106, pp. 1-11. DOI: 10.1016/j.ijpvp.2013.02.002.

Shao, Huaishuang, Yibo Wang, Tao Shen, Guohua Wei, Chengfan Shi, Shifeng Deng and Qinxin Zhao (2026). “Comparative study of water-cooled wall supercritical and subcritical operation characteristics during deep peaking of 350 MW supercritical boiler”. In: Fuel 410, pp. 137950. DOI: 10.1016/j.fuel.2025.137950.

Wang, Chao, Xiangru Chen, Bin Liu, Kairui Liu, Yuan Han, Limin Wang and Defu Che (2025). “Dynamic analysis for ultra-supercritical boiler water wall considering uneven heat flux distribution and combustion instability”. In: International Journal of Heat and Mass Transfer 236.1, pp. 126269. DOI: 10.1016/j.ijheatmasstransfer.2024.126269.

Wang, Siyang, Dong Yang, Yunjie Zhao and Mofeng Qu (2018). “Heat transfer characteristics of spiral water wall tube in a 1000 MW ultra-supercritical boiler with wide operating load mode”. In: Applied Thermal Engineering 130, pp. 501-514. DOI: 10.1016/j.applthermaleng.2017.10.114.

Zhang, T.J., J.T. Wen, Y. Peles, J. Catano, R.L. Zhou and M.K. Jensen (2011). “Two-phase refrigerant flow instability analysis and active control in transient electronics cooling systems”. In: International Journal of Multiphase Flow 37.1, pp. 84-97.

Automobile Model Digital Engineering Practice

Zhong Yilin[1] Ou Haiying[1] Chen Jingke[1] Shi Maqian[1]
Chen Zhixu[1] Lu Guoguang[1]
[1]BYD Auto Industry Company Limited, China

Abstract

With the rapid growth of electrification, intelligence, and customization demands in the automotive industry, parallel development of multiple vehicle models imposes higher requirements on the re-usability, flexibility, and development efficiency of digital engineering models. Based on the practical application of automotive model digital engineering, this paper proposes an overall framework for automotive research and development. By integrating key technologies such as modular hierarchical architecture design, parametric configuration-driven modeling, cross-domain data collaboration management, and automated verification systems, the framework establishes an end-to-end solution spanning requirement analysis, model development, simulation validation, and iterative optimization. Based on a specific vehicle model system development case, this framework achieved a ~50% reduction in system development cycles, ~30% fewer quality issues, and 50% higher model development efficiency. It enables rapid model reuse and facilitates swift derivative design across various power trains (fuel, electric, hybrid) and configurations. The research provides methodological insights for efficient digital engineering systems in automotive industry.

Keywords: Automobile Model Digital Engineering, MBDE, automobile development, rapid reuse, quality improvement and cost reduction

1 Introduction

As the automotive industry accelerates its transition toward electrification, intelligence, and customization, concurrent multi-vehicle development has become a core strategy for enterprises to enhance market competitiveness. However, the compiled 'one vehicle, one design' workflow of traditional development models struggles to meet rapid iteration demands, resulting in protracted development cycles and persistently high costs. Industry statistics indicate that the R&D cycle for new-architecture vehicles averages 36-42 months, while model refresh cycles average approximately 24 months[1]. During development, data piles prevent real-time interoperability between design, simulation, and production phases, leading to rework rates as high as 31%[2]. In this context, Model-Based Digital Engineering (MBDE)[3] emerges as a critical solution: by establishing a digital thread spanning the entire life cycle from requirements to design and validation, it enables a model-driven closed-loop development system.

MBDE, defined as a methodology rooted in Model-Based Systems Engineering (MBSE) with digital system models as the core and the digital thread as the backbone, achieves automotive product, manufacturing, and industry digitization through digital delivery, full-system simulation validation, and digital twins[3].

MBDE, driven by requirement engineering, establishes a mapping chain across the 'Requirements-Function-Logic-Physical' (RFLP) framework. In traditional development processes, the disconnect between requirements and the design and verification phases often leads to 'requirements drift', with most vehicle development projects delayed due to design rework triggered by requirement changes. MBDE transforms natural language requirements into executable models using systems modeling languages such as SysML[4], establishing explicit causal relationships between requirements and design parameters. For example, a vehicle's range requirement can be directly mapped to physical model parameters such as motor efficiency and thermal management strategies, with automatic traceability of change impacts achieved through the digital thread. This causality-driven development paradigm not only enhances the accuracy of requirement implementation but also provides a logical foundation for multi-vehicle model reuse through modular interface definitions.

In recent years, digital engineering technologies have flourished both domestically and internationally. The U.S. DARPA's AVM (Advanced Virtual Manufacturing) experimental program has increased equipment development efficiency by fivefold[5]; China's aerospace sector has achieved first-time ignition success in rocket engine system testing and successfully established a digital space station. In the automotive industry, Tesla employs digital twin models to conduct full-process digital simulation for each vehicle, while Renault and Mercedes-Benz have established digital twin factories. Domestic automakers such as Changan and FAW Group have also integrated digital twin platforms. These cases have significantly shortened product delivery cycles and substantially reduced R&D expenditures[3][6].

However, existing MBDE practices still face two major challenges: First, insufficient process standardization restricts model reuse to localized domains, resulting in low collaboration efficiency across vehicle models and departments. Second, an inherent conflict exists between performance validation and model reuse — highly customized performance optimization often compromises module generality. Regarding the former, ASPICE (Automotive SPICE) [7] Level 3 certification mandates standardized capability for establishing and maintaining standard processes, yet current tool chains lack systematic support for multi-level model reuse. Regarding the latter, in performance-critical scenarios such as motor thermal simulation, traditional parametric methods struggle to balance accuracy and generalization capability, leading to significant performance degradation after model reuse.

To address these challenges, this paper proposes a rapid multi-vehicle model reuse framework for digital engineering based on a closed-loop 'demand-driven simulation-validation-causal traceability' paradigm. Its innovation is embodied in three aspects:

Demand-Driven Modular Architecture: Building causal links between requirements-function-logic via SysML, defining input-output constraints between modules through interfaces to ensure automatic propagation of requirement changes along the digital thread to physical model parameters.

Hierarchically Decoupled Model Library: Adopting a 'platform-derivative' hierarchical strategy, common vehicle requirements are abstracted into a platform-level model library, while differentiated requirements are encapsulated as reusable derivative modules. Parameterized configuration enables rapid iteration with 'platform-agnostic, derivative-flexible' adaptability.

Performance-Aware Automated Validation: Integrating multi-fidelity simulation technology to maintain critical performance accuracy (e.g., thermodynamics), while accelerating feasibility validation for multi-vehicle configurations through reduced-order models and response surface methodology.

To validate the framework's effectiveness, this paper conducts two engineering case studies:

Case Study 1:A SysML-based SCU development process (already ASPICE Level 3 certified) demonstrates the framework's advantages in process standardization and compliance.

Case Study 2:In a low-voltage motor thermal simulation case for a specific vehicle model, model development time was reduced by over 50% with errors kept within 10%, verifying the framework's capability to balance performance and reusability through its model library and automated validation.

This work provides a practical theoretical methodology for the automotive industry to achieve 'platform-based development with personalized derivatives'.

2 Overall Framework Design

2.1 Core Contradiction

The core contradiction in automotive digital engineering lies in how to break the path dependency of the 'one vehicle, one model' paradigm in traditional development while maintaining multi-disciplinary coupling accuracy. This paper proposes a framework oriented toward enhancing model reuse value, constructing a five-dimensional collaborative system spanning requirements-model-data-tools-verification to form a digital engineering ecosystem spanning the entire life cycle. Its innovation lies in the deep integration of systems engineering methodologies with industrial software tool chains, achieving a bidirectional closed-loop of 'requirement-driven modeling' and 'data feeding back to requirements'.

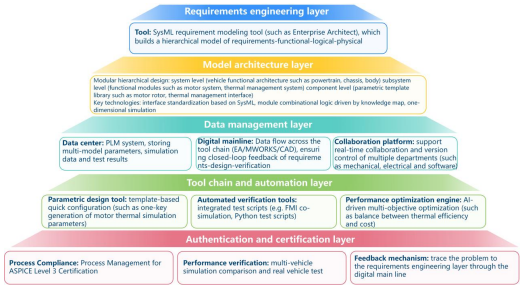


Figure 1. Overall Framework.

2.2 Evolution of the MBDE-Based Architectural Paradigm

Model-Based Automotive Digital Engineering (MBDE) adopts Model-Based Systems Engineering (MBSE) as its methodology, centers on digital system models, utilizes the digital thread as its backbone, and leverages digital twins as its implementation approach. By achieving digital delivery across the automotive supply chain, it realizes digital product development for vehicles, forming model-based digital vehicles that support multi-level system simulation and validation for the entire vehicle. Through the continuous evolution of digital system model systems across all life cycle phases — including design, testing, and operation — and model-based digital integration, it achieves full-process digitization of vehicle development. As the automotive digital system model evolves continuously, it gradually forms a comprehensive digital vehicle system model, which naturally establishes a digital twin of vehicle equipment post-production, enabling virtual testing and operational maintenance of vehicles.

In traditional V-model development processes, a significant semantic chasm exists between requirements engineering and physical implementation. This framework introduces Model-Based Requirements

DOI 10.3384/ecp223329

Engineering as the logical starting point of the digital thread, constructing mapping relationships between the requirement parameter space and design parameter space using SysML. During the requirements capture phase, 'scenario slicing' technology decomposes the whole vehicle level requirements into quantifiable functional feature vectors. For example, the requirement 'range ≥ 500 km' is transformed into a constraint equation system involving 23 coupled parameters, such as battery capacity, motor efficiency, and aerodynamic drag coefficient. This structured requirements approach not only achieves the bidirectional traceability mandated by ASPICE Level 3 certification but, more importantly, establishes mathematical represent ability for how requirement changes impact the design space—when new vehicle model requirements emerge from the market, the system can automatically identify the required model component sets for adjustment through parameter sensitivity analysis.

At the architectural design level, the framework breaks through the traditional 'functional domain-control domain' partitioning by proposing the concept of a 'dynamically re-configurable module library'. Each module encompasses a functional logic body defining input-output relationships, a physical implementation body storing manufacturing attributes (e.g., geometry, materials), and an efficiency assessment body embedding multi-fidelity simulation models. This three-dimensional representation enables modules to maintain interface consistency across different vehicle configurations while autonomously selecting implementation paths based on the target vehicle ' s positioning (e.g., economy-oriented/performance-oriented). For example, for the same power requirement in the motor system, multiple configured solutions (e.g., permanent magnet synchronous motors, induction motors, or switched reluctance motors) can be flexibly deployed to meet differentiated development needs.

2.3 Data-Knowledge Dual-Driven Modeling System

The bottleneck of model reuse efficiency often lies not in parametric design itself, but in the digital encapsulation of implicit engineering knowledge. This framework constructs a data hub for extensive engineering experience knowledge components, transforming designers' expertise into compute inference rules. In thermal management system development, the simulation platform automatically invokes cross-disciplinary knowledge units—such as flow boiling heat transfer correlations and metal fatigue life prediction models—based on input parameters from the data hub (e.g., coolant flow rate, ambient temperature). This knowledge-enhanced modeling approach not only improves simulation accuracy compared to traditional empirical formulas but also compresses multi-disciplinary coupling analysis time from week-level to hour-level.

2.4 Digital Thread Closed-Loop Verification Mechanism

In the construction of the digital thread, this framework introduces a SysML-based bidirectional requirement traceability tool as the intelligent core of the data closed-loop. This tool establishes a causal relationship network among modeling requirement parameters, design parameters, and validation metrics. When test site data indicates that a specific metric fails to meet specifications, the system can trace backward along the causal chain to identify abnormal confidence intervals in the corresponding parameters and automatically trigger design iterations. This causal traceability capability significantly enhances problem localization efficiency, eliminating resource waste inherent in traditional trial-and-error approaches.

Regarding process compliance assurance, a system design process compliant with ASPICE Level 3 has been developed. Each development activity automatically generates a comprehensive process evidence chain. Natural Language Processing (NLP) technology is employed to compare the coverage completeness between requirement specifications and test cases, thereby effectively enhancing model re-usability and subsequently shortening the R&D cycle.

2.5 Multi-Level Collaborative Development Platform

To enable model reuse across departmental boundaries, a model sharing platform based on a local area network has been developed. Each uploaded model component embeds a smart contract defining usage permissions, billing rules, and iterative feedback mechanisms, forming a virtuous cycle of 'increasing intelligence through continuous usage'. The platform adopts a federated learning architecture, enabling cross-organizational knowledge collaborative evolution while safeguarding enterprise data sovereignty, as illustrated in Figure 2.

In terms of tool chain integration, an adapter matrix covering multiple industrial software components has been constructed. By developing semantic middleware, high coupling between requirement tools, system modeling software, and simulation validation software is achieved, reducing the number of iterations and effectively shortening the validation cycle. This breakthrough in tool chain interoperability has brought multi-disciplinary optimization (MDO) into practical engineering application.



Figure 2. Collaborative Development Platform Interface Illustration.

3 Automotive Model Digital Engineering Practice Case

3.1 SCU System Design and Development

3.1.1 Case Background

The gear shift control system, also known as the SCU (Shift Control Unit) system, is a high-functional-safety-critical system. In a kind of EEA (Electrical/Electronic Architecture) framework, the SCU is integrated into the combination switch or shift panel product and developed by external suppliers, presenting pain points including lengthy development cycles, inability to achieve platform-based management, and high costs for change propagation. In the upgraded EEA architecture, the SCU adopts an in-house development approach, integrated into the body domain controller. Faced with the new requirements of shorter software change cycles and higher platform realization in the new architecture, the SCU must deliver high-reliability products. Consequently, the SCU system adopts a model-based digital R&D approach.

Throughout the software life cycle of the SCU system, process elements from the ASPICE standard framework are integrated into the MBDE methodology, achieving full coverage of the entire workflow across design, development, and verification phases. By linking original requirements, system requirements with design, software requirements with design, and corresponding test levels, a closed-loop structure of 'requirements-design-verification' is established. The MBDE & ASPICE process V-model is illustrated in Figure 3.

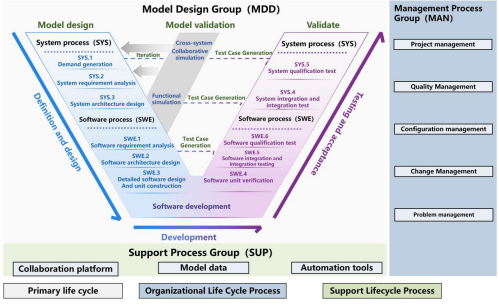


Figure 3. MBDE & ASPICE Process V-Model Diagram.

In the design and verification phases, the ASPICE-integrated MBDE methodology was applied, with SysML models providing clearer and more intuitive representation in Enterprise Architect.A bidirectional traceability relationship between upstream and downstream processes was established, ensuring 100% coverage of design verification.

3.1.2 Technical Solution

During the SCU system design process, ASPICE standards are strictly followed to establish a clear hierarchical structure of 'stakeholder requirements-system requirements-software requirements' through requirement diagrams (see Figure 4). In the SYS2 (System Requirements Analysis) phase, SysML activity diagrams are employed to model requirement scenarios, eliminating ambiguities inherent in textual descriptions. Logical symbols such as decision nodes comprehensively represent all branching paths, while subsystems are segmented to visually clarify inter-system interaction relationships. This approach avoids the cumbersome nature of traditional tabular signal matrices.

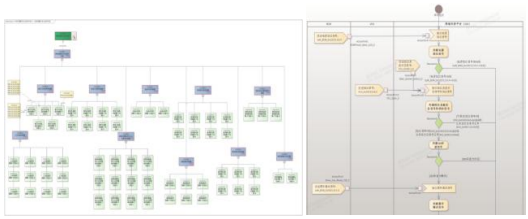


Figure 4. System Requirements Hierarchy and Requirement Scenario Diagram.

Leveraging the can be simulation nature of SysML behavioral diagrams, early validation of requirement logic or states is conducted. A SysML-based control panel and state machine achieve end-to-end functional design validation at the SCU system level (see Figure 5), enabling the identification and correction of potential errors during the modeling phase.

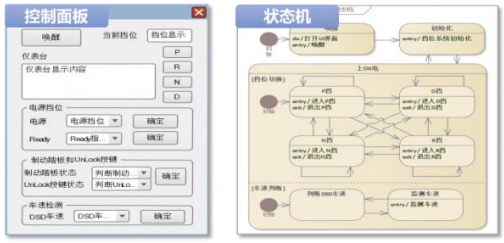


Figure 5. System Requirements State Machine Logical Closed-Loop Validation.

Compared to purely textual descriptions, model-based requirement representation provides more intuitive and structured logical expression, significantly reducing communication overhead between system design engineers and software developers and preventing

DOI 10.3384/ecp223329

misunderstandings. Simultaneously, visual preliminary validation enables rapid confirmation of requirement correctness, substantially enhancing overall design and development efficiency.

3.1.3 Case Outcomes

In application, the MBDE-based model development for SCU has enabled the electronic-electrical architecture platform, supporting 5 mass-produced vehicle models and further empowering additional R&D models. Compared to previous supplier collaborations, development cycles were shortened by 50%, and quality issues reduced by approximately 30%, achieving dual improvements in software quality and efficiency.

The SCU system has established a new paradigm for model-based R&D in the exploration process, overcoming the contradiction between quality and efficiency in full-stack in-house development systems. By leveraging a reliable tool chain to ensure product quality and enhance development efficiency, it has demonstrated the universality of the methodology, forming high-standard software development management specifications and guidance manuals. Ultimately, it achieved ASPICE CL3 capability certification and empowered 5 mass-produced vehicle models on the EEA4.0 platform, driving high-efficiency, high-quality development of SCU system software.

Leveraging complex functionality platforming and model-based R&D methodologies, the system-level functional model achieves high reuse rates.

3.2 Motor Thermal Simulation

3.2.1 Case Background

Taking a leading automaker as an example, among its 8 vehicle models under development, only 35% share core motor structures. Thermal models for motors require repeated adaptation for different platforms, power levels, and application scenarios, resulting in a model re-development rate exceeding 60%. This low reuse rate not only causes severe waste of development resources but also forces project timelines to be extended. Technically, traditional thermal management solutions rely on external sensor networks and over-engineering, which increase costs and integration complexity while failing to meet increasingly stringent light weighting requirements. More critically, according to survey data from the China Society of Automotive Engineers (CSAE), motor thermal failures accounted for 41.7% of vehicle malfunctions in 2023, making it the primary failure source during the new energy vehicle (NEV) warranty period.

3.2.2 Technical Solution Design Principles

This case adopts a standardized and modular modeling approach, leveraging the equation-oriented characteristics of Modelica[8] to establish a unified motor thermal modeling framework. During solution design, the following principles are prioritized:

(a)Unified Modeling Standards:Adopting thermal resistance network theory to establish standardized thermal modeling, ensuring model compatibility across different motor types.

(b)Parametric Design:Achieving flexible adaptation of a single foundational model to diverse motor types through parameterized configuration.

(c)Modular Architecture:Decomposing the motor thermal system into independent thermal modules using object-oriented modeling for enhanced maintainability and adaptability.

(d)Lightweight Validation:Establishing simplified validation frameworks to ensure core functionality reliability, avoiding overly complex simulation environments.

(e)Standardized Interfaces:All models support FMU (Functional Mockup Unit) packaging per Modelica standards, enabling seamless integration across development environments (e.g., Python, C/C++).

(f)Real-Time Interaction:Supporting real-time parameter adjustment via front-panel interfaces for visual monitoring and interactive control during simulation.

3.2.3 Implementation Path of the Technical Solution

The case practice adopts a phased development strategy, emphasizing practicality and scalability. Phase 1 focuses on foundational model standardization, establishing a unified motor thermal modeling framework, including:thermal modeling standards defining a unified thermal node partitioning method and establishing basic physical models for heat conduction and convection;development of over 200 standardized thermal components within the core component library, covering major motor heat sources and thermal pathways;basic model functionality enabling fundamental capabilities such as motor startup characteristics, steady-state thermal equilibrium, and transient response;FMU packaging where all foundational models are encapsulated as FMU files compliant with Modelica standards to support invocation across different development environments. Figure 6 and Figure 7 respectively illustrate the physical structure of a low-voltage automotive motor and its corresponding thermal network model.

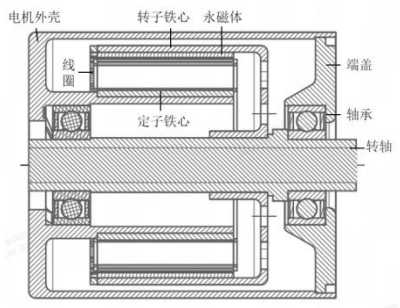


Figure 6. A low-voltage automotive motor structure.

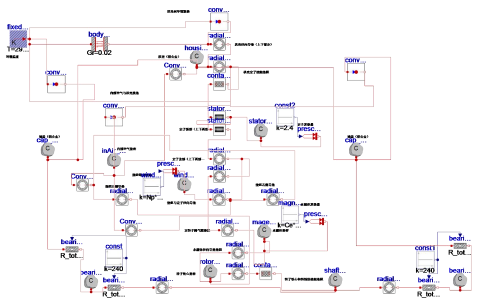


Figure 7. System Requirements State Machine Logical Closed-Loop Validation.

Phase 2 is parametric configuration, establishing parametric configuration rules to enable flexible model adaptation. Physical parameters of the model are accurately obtained from actual structural CAD data. First, parameter sensitivity analysis is conducted through experimental data calibration to determine the impact degree of key parameters; second, parameter constraint conditions are defined, specifying reasonable value ranges and coupling relationships for each parameter. Phase3 is automated assembly, achieving automated model integration. Figure 8 illustrates the process of converting the motor thermal model into an FMU and subsequently integrating it into a visualization panelvia Python.

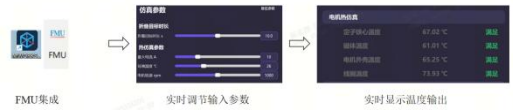


Figure 8. motor thermal model-FMU-visualization panel.

Phase 4 is optimization and improvement, that is, continuously refining the model based on application feedback. It includes: precision enhancement, continuously optimizing model parameters through experimental data feedback; function expansion, gradually adding model capabilities according to practical requirements;performance optimization, continuously improving computational efficiency and response speed; interface extension, supporting invocation methods for more programming languages (e.g., Python, C/C++).

3.2.4 Technical Features

This case demonstrates multiple technical features in motor thermal model reuse:(a) Equation-oriented modeling approach: Adopting Modelica's declarative modeling to better express the inherent physical system characteristics. (b) Application of thermal resistance network theory:Simplifying complex motor thermal transfer processes into thermodynamic network systems, quantifying characteristics of different thermal transfer paths through thermal resistance elements. (c) FMU standard packaging:All models are encapsulated as FMU files compliant with Modelica standards, ensuring seamless invocation across development environments (e.g., Dymola, JModelica, Python). (d) Real-time interaction capability:Supporting real-time parameter adjustment via front panels for visual monitoring and interactive control during simulation. (e) Parametric configuration system:Quantifying the impact of each thermal parameter on output temperature through parameter sensitivity analysis.

3.2.5 Case Outcomes

Through these technological innovations and implementation pathways, the case achieved substantial progress in related projects:First, development efficiency improvement—model development time reduced by over 50% compared to traditional development modes. Second, computational efficiency improvement — the thermal network model reduced computation time by over 95% compared to traditional 3D finite element simulation. Finally, precision assurance — model calculation results maintained errors within 5.9% of experimental data from a mass-produced motor. Figure 9 shows validation results using experimental data from a production motor.

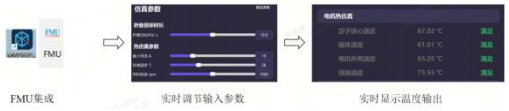


Figure 9. Validation results of a mass-produced motor.

4 Conclusion

The overall framework for rapid reuse of multi-vehicle model digital engineering proposed in this paper, driven by requirements engineering, achieves full life cycle closed-loop management of requirements-design-simulation-verification through the digital thread. It constructs an integrated technical system featuring modular architecture, parametric configuration, multi-disciplinary collaboration, and automated validation, effectively addressing the pain points of low reusability and insufficient collaboration efficiency in traditional automotive model development. The SysML-based SCU development process successfully achieved ASPICE CL3 certification, validating the significant improvement in quality control through process standardization. In the motor thermal simulation case, model reuse reduced development time by over 50% and maintained performance errors within 10%, highlighting the feasibility of technology implementation. Future work will focus on deep integration of multi-disciplinary knowledge graphs, exploring AI-driven automated model generation and validation, and further enhancing the framework's generalization capability in complex scenarios to provide a more flexible technical foundation for the automotive industry's digital transformation.

DOI
10.3384/ecp223329

Acknowledgements

-

References

iResearch (2022-3).“China Automotive Enterprise Digital Transformation Trend Series Research”.URL:https://m.thepaper.cn/baijiahao_17141978.

Sanpin PLM (2025-12). “PLM Industry Analysis Report: Technological Breakthroughs and Application Practices Under Digital Transformation[EB/OL]”.URL:https://weibo.com/ttarticle/p/show?id=230940523948240758396 5.

Ou Haiying, Zhong Yilin, Xu Xiaodong, et al (2023-10). “Model-Based Digital Engineering: An Essential Path for Automotive Innovation and Development”In: Proceedings of the 2023 Annual Conference of China Society of Automotive Engineers.CNKI,pp.361-367 .DOI:10.26914/c.cnkihy.2023.070505.

Deligiannis S (2015-11). “SysML Essentials [M]”. In：China Machine Press.CNKI, pp. 1-36. ISBN: 978-7-111-48458-5.

Walker J D, Chocron S, Anderson C et al (2013-1). “Survivability modeling in Darpa's Adaptive Vehicle Make (AVM) program”. In: International symposium on ballistics.pp. 967-979. ISBN: 978-1-60595-106-5.

Zhao Min, Zhou Fanli, Chen Liping, Zhang Ye (2021-10).” Developing Digital Engineering to achieve Digitalization of Aerospace Equipment”.In:Missile and Space Launch Technology.CNKI,pp.6 (2021) : 106-113.ISSN:2097-1974.

Mu Dayun, Zhu Weiwei, Yan Xiaodong, et al (2022-7).” Research on Automotive Software Development Process Based on Integration of ASPICE and ISO 26262 Standards”.In: Automotive Electrical Appliances.CNKI,pp.7(2022):24-26+29. DOI:10.13273/j.cnki.qcdq.2022.07.011.

Yuan Lin, Liu Yusheng (2018-3).”Conceptual Design Functional Modeling Method Based on SysML”.In:Computer Integrated Manufacturing Systems.CNKI,pp.24(1):164-177.DOI:10.13196/j.cims.2018.01.017.

DOI
10.3384/ecp223329

22-DOF Full-Coupled Vehicle Dynamics Modeling and Control Based on Modelica

Yufan Guo[1] Zehua Wu[1] Bingtao Ren*[1] Shichun Yang[1]

[1]School of Transportation Science and Engineering, Beihang University, Beijing, China

Abstract

A 22-DOF vehicle dynamics model is developed in Modelica and implemented in MWorks Sysplorer 2025a. The generalized coordinates comprise six sprung-body motions, twelve translational motions of the four unsprung masses, and four wheel-spin motions. The physical plant is formulated as a coupled differential-algebraic equation system: the body, suspension, unsprung mass, wheel, tire, and road components are connected through custom acausal physical connectors, whereas steering and wheel-torque commands are retained only as external causal boundaries. The model is assessed using acceleration, braking, and steering-step datasets from MWorks, a 22-DOF Simulink model, and road tests. The three-source comparison demonstrates consistent overall dynamic trends and close agreement in key performance indicators, particularly in braking-distance prediction and steady steering response, thereby validating the effectiveness of the proposed acausal 22-DOF modeling framework.

Keywords:Vehicle Dynamics, Modelica Language, 22-DOF Full-Coupled Model, Acausal Modeling,Mworks Platform

1 Introduction

With the evolution of autonomous driving technology to L3 and above, the system puts forward unprecedented requirements for the accuracy, real-time performance, and scenario coverage of virtual simulation. Researches by the European Commission show that autonomous driving systems need to complete 200 million kilometers of effective testing mileage to meet statistically significant safety requirements.However, real-vehicle testing is economically and practically limited: it incurs extremely high per-kilometer costs, rendering large-scale validation unfeasible, and cannot safely replicate extreme scenarios such as adverse weather and emergency obstacle avoidance (European Commission, 2019). Data from Baidu Apollo virtual simulation platform shows that its daily testing mileage can reach 18 million kilometers, equivalent to five years of real-vehicle testing. Virtual simulation has become the core support for the commercialization of autonomous driving technology,e.g.,(Baidu Apollo. (2023)). The ISO 21448 Safety of the Intended Functionality standard further clarifies that virtual simulation must cover at least 95% of dangerous scenarios, which imposes mandatory requirements on the full-condition accuracy of vehicle dynamics models,e.g.,(ISO 2021).

As the core link of autonomous driving simulation, the accuracy of vehicle dynamics models directly determines the verification effect of control algorithms. Current vehicle dynamics modeling is mainly divided into three categories: the first is high-precision models based on multi-body dynamics (such as Adams, RecurDyn), which can accurately describe the geometric characteristics and flexible deformation of components, but the single-step simulation time exceeds 10ms, making it difficult to meet real-time simulation requirements; the second is Simulink models based on causal modeling, with high computational efficiency, but the core modules are commercially encapsulated, making secondary development difficult and model reusability poor, making it difficult to migrate between different projects; the third is acausal modeling based on Modelica, which has both accuracy and efficiency advantages and supports unified modeling of multiple physical domains, and has received extensive attention from academia and industry in recent years ,e.g.,(Fritzson 2014;Otter 1998).

Unlike traditional causal modeling approaches (such as block-diagram based tools) that require manual definition of data flow and execution order, Modelica uses a non-causal declarative modeling paradigm. Engineers only need to describe the inherent physical laws of components through conservation equations, without specifying how to solve them. The compiler automatically performs symbolic analysis, equation sorting and code generation. This feature naturally supports multi-domain unified modeling of vehicle mechanics, powertrain and control systems, greatly improves model reusability and iteration efficiency, and ensures consistent simulation accuracy across all operating conditions

However, existing Modelica-based vehicle dynamics research still has obvious shortcomings: First, most models only consider the vertical DOF of unsprung mass (traditional low-DOF models), ignoring the influence of longitudinal and lateral motion on vehicle handling stability. For SUVs, unsprung mass accounts for 15%-20% of the total vehicle mass. The lateral motion of unsprung mass under limit steering conditions leads to a calculation error of tire load transfer exceeding 20%, thereby affecting the simulation accuracy of yaw rate and lateral accelera-

tion,e.g.,(Drogies 2002;Zhang 2023); Second, the lack of a systematic cross-platform verification method results in poor consistency of simulation results between different modeling platforms, making it difficult to achieve cross-platform migration and reuse of models. Existing studies mostly use qualitative comparison methods, which cannot quantitatively reveal error sources, leading to blind and inefficient calibration processes,e.g.,(Liu 2010;Wang 2024); Third, control algorithms are mostly integrated through the FMI(Functional Mock-up Interface)external interface, which has problems such as large data transmission delay and poor interface compatibility. For high-frequency control algorithms above 100Hz, delay will lead to degraded control performance or even instability,e.g.,(Janin 2007;Liu 2023). In addition, most existing studies are based on foreign Modelica platforms(such as Dymola), and the engineering applicability verification of domestic platforms is still lacking,e.g.,(Tongyuan 2024).

To address the above problems, this paper conducts research on Modelica-based 22-DOF full-coupled vehicle dynamics modeling and control verification. The main innovations of this paper are as follows:

1. **A connector-based fully coupled 22-DOF Modelica model:** A 22-DOF vehicle dynamics model is implemented in MWorks by including the six degrees of freedom of the sprung body, twelve translational degrees of freedom of the four unsprung masses, and four wheel-spin degrees of freedom. The physical subsystems are formulated through acausal connectors and conservation equations, enabling the coupled solution of body, suspension, wheel, tire, and road interactions.

2. **A modular physical-model and control-boundary architecture:** The proposed implementation uses `FramePort`, `SpinPort`, and `RoadPort` to establish the internal physical connections. External steering and wheel-torque commands are handled through causal boundary interfaces, while vehicle states are exported as read-only observations. This architecture separates the acausal vehicle plant from the causal control logic and improves the reusability of the Modelica model in MWorks.

2 22-DOF FULL-COUPLED VEHICLE DYNAMICS THEORETICAL MODELING

2.1 Model Assumptions and DOF Division

This paper models a certain SUV model. To balance model accuracy and computational efficiency, the following reasonable assumptions are introduced: 1. The vehicle body is a rigid body, ignoring its elastic deformation, and the products of inertia $I_{xy} = I_{xz} = I_{yz} = 0$; 2. The suspension system is a parallel connection of linear elastic and damping elements, ignoring internal friction and nonlinear hysteresis characteristics; 3. The steering system adopts the Ackermann steering model with a constant transmission ratio, ignoring steering clearance and elastic deformation; 4. Air resistance acts centrally on the body's center of mass, ignoring the effects of lift and side force; 5. The four wheels are always in contact with the ground without lifting.

Based on the above assumptions, the entire vehicle system is decomposed into 1 sprung mass (body) and 4 unsprung masses (wheels and support structures), with a total of 22 DOFs, specifically divided as follows:

Body module (6 DOFs): Longitudinal u, lateral v, vertical w translational DOFs; roll p, pitch q, yaw r rotational DOFs;

Wheel module (4 DOFs): Rotational DOF $\omega_{wi}(i = 1,2,3,4)$ of each wheel around the axle;

Unsprung mass module (12 DOFs): Longitudinal dx_i, lateral dy_i, vertical dz_i translational DOFs of each unsprung mass.

2.2 Coordinate System Definition and Transformation

To unify the motion description of each module, three right-hand Cartesian coordinate systems are defined: 1. Inertial coordinate system (geodetic coordinate system) $OXYZ$: The origin is fixed on the ground, the X-axis points to the initial forward direction of the vehicle, and the Z-axis is vertically upward from the ground; 2. Body coordinate system $oxyz$: The origin is at the body's center of mass and moves with the body, the x-axis is forward along the vehicle longitudinal direction, the y-axis points to the driver's left, and the z-axis is vertically upward from the body; 3. Tire coordinate system $o_t x_t y_t z_t$: The origin is at the tire-ground contact center, the x_t-axis is forward along the intersection line of the wheel plane and the ground, and the z_t-axis is vertically upward from the ground.

In any attitude, the transformation matrix from the inertial coordinate system to the body coordinate system is the product of three basic rotation matrices:

$$X_{gb} = X_\varphi X_\theta X_\psi \tag{1}$$

where X_φ, X_θ, X_ψ are the roll, pitch, and yaw rotation matrices, respectively, with specific forms:

DOI
10.3384/ecp223337

$$X_\psi = \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix},$$
$$X_\theta = \begin{bmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{bmatrix}, \quad (2)$$
$$X_\varphi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & \sin\varphi \\ 0 & -\sin\varphi & \cos\varphi \end{bmatrix}$$

where ψ, θ, φ are the yaw, pitch, and roll angles of the body, respectively. Furthermore, when converting the acceleration in the inertial coordinate system to the body coordinate system, the effects of Coriolis acceleration and centripetal acceleration must be considered:

$$\dot{v}_b = X_{gb}\ddot{r}_g - \omega_b \times v_b \quad (3)$$

where $\dot{v}_b$ is the acceleration in the body coordinate system, $\ddot{r}_g$ is the second derivative of position in the inertial coordinate system, ω_b is the angular velocity vector in the body coordinate system, and $\times$ denotes the cross product operation. This transformation relationship is the key to realizing the coupled modeling of the body and unsprung mass, and one of the main sources of error in traditional decoupled models ,e.g.,(Wei 2015).

2.3 Core Subsystem Dynamics Modeling

The proposed fully coupled 22-DOF vehicle dynamics model is developed using Modelicas object-oriented modeling approach. The vehicle body, suspension, unsprung masses, wheels, tires, and road are encapsulated as reusable physical components. These components are interconnected through acausal connectors that contain kinematic potential variables and mechanical flow variables. The connect() equations automatically impose kinematic compatibility and force/moment balance at each connection point. During compilation, the equations of all subsystems are assembled into a unified system of differential-algebraic equations and solved simultaneously. Therefore, the computational direction and solution sequence are determined by the Modelica compiler rather than being predefined by the model developer. Figure 1 presents the modular decomposition of the 22-DOF vehicle dynamics model . It is intended to illustrate the organization of the model components and their physical equations, rather than to represent a directed data-flow or physical-topology diagram.

2.3.1 Six-DOF Vehicle Body Dynamics Model

The vehicle body component is equipped with acausal physical ports at the four suspension mounting points. Each port contains kinematic potential variables, including the position, velocity, and heading of the mounting point, as well as flow variables representing the interaction forces and moments. The position and translational velocity of the center of gravity, together with the vehicle attitude angles and angular velocities, constitute the internal states of the body component. The longitudinal, lateral, and vertical accelerations are determined jointly from the NewtonEuler balance equations. Based on the Newton-Euler equations, a six-DOF dynamic differential equation system for the vehicle body is established in the body coordinate system.

Longitudinal motion equation:

$$m_b\left(\dot{v}_x - v_y\omega_z + v_z\omega_y\right) = \sum_{i=1}^{2} F_{x_i}\cos\delta_i - \sum_{i=1}^{2} F_{y_i}\sin\delta_i + \sum_{i=3}^{4} F_{x_i} + F_R \quad (4)$$

where m_b is the body mass, F_{x_i} and F_{y_i} are the longitudinal and lateral forces of the i-th tire, δ_i is the front wheel angle, and F_R is the driving resistance, including rolling resistance, air resistance, and grade resistance:

$$F_R = m_b g f\cos\alpha + \frac{1}{2}C_D A\rho v_x^2 + m_b g\sin\alpha \quad (5)$$

where f is the rolling resistance coefficient, C_D is the air resistance coefficient, A is the frontal area, ρ is the air density, and α is the road gradient.

Lateral motion equation:

$$m_b\left(\dot{v}_y - v_z\omega_x + v_x\omega_z\right) = \sum_{i=1}^{2} F_{x_i}\sin\delta_i + \sum_{i=1}^{2} F_{y_i}\cos\delta_i + \sum_{i=3}^{4} F_{y_i} \quad (6)$$

Vertical motion equation:

$$m_b\left(\dot{v}_z - v_x\omega_y + v_y\omega_x\right) = \sum_{i=1}^{4} F_{zs_i} - m_b g \quad (7)$$

where F_{zs_i} is the vertical force of the i-th suspension.

Yaw motion equation:

$$\begin{aligned} I_{zz}\dot{\omega}_z - (I_{xx} - I_{yy})\omega_x\omega_y = {} & \sum_{i=1}^{4} M_{zi} + \frac{B_f}{2}\left(F_{x1}\cos\delta_1 + F_{y1}\sin\delta_1 - F_{x2}\cos\delta_2 - F_{y2}\sin\delta_2\right) \\ & + \frac{B_r}{2}\left(F_{x4} - F_{x3}\right) \\ & + L_f\sum_{i=1}^{2}\left(F_{x_i}\sin\delta_i + F_{y_i}\cos\delta_i\right) \\ & - L_r\sum_{i=3}^{4} F_{y_i} \end{aligned} \quad (8)$$

where I_{xx}, I_{yy}, I_{zz} are the moments of inertia of the body around the x, y, z axes, M_{zi} is the tire self-aligning torque,

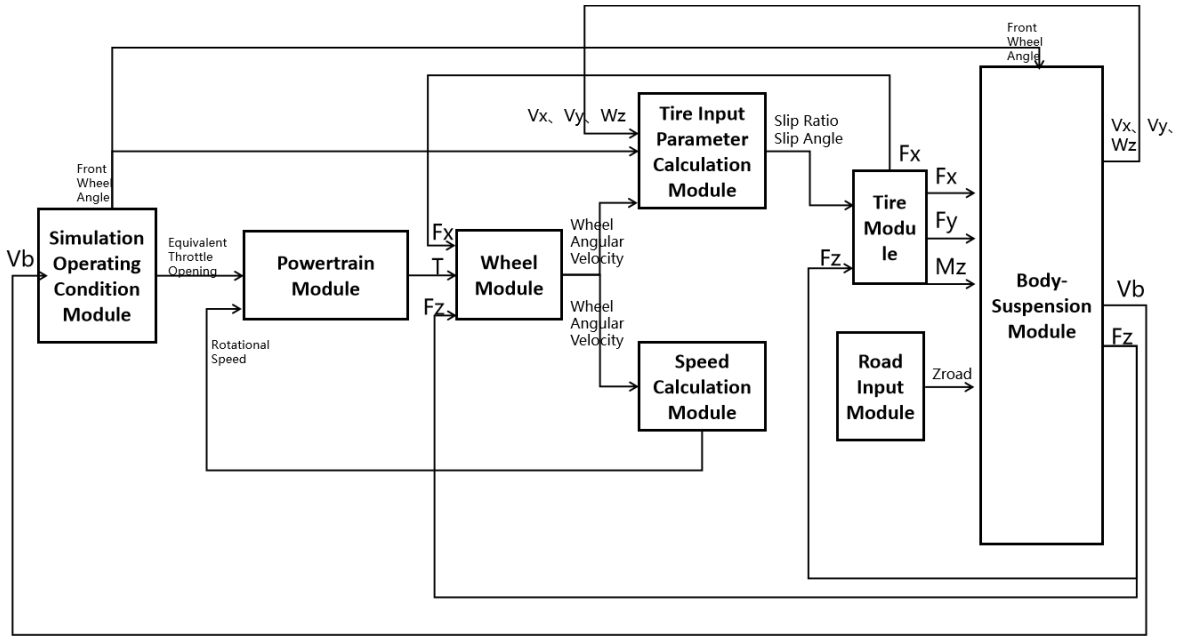


Figure 1. Modular decomposition of the 22-DOF vehicle dynamics model

B_f and B_r are the front and rear track widths, and L_f and L_r are the distances from the center of mass to the front and rear axles.

Roll motion equation:

$$\begin{aligned} I_{xx}\dot{\omega}_x - (I_{yy}-I_{zz})\omega_y\omega_z = {} & \frac{B_f}{2}(F_{zs1}-F_{zs2}) \\ & + \frac{B_r}{2}(F_{zs3}-F_{zs4}) \\ & + m_b g H_e \sin\varphi + m_b a_y H_e \cos\varphi \end{aligned} \tag{9}$$

where H_e is the height from the body's center of mass to the roll axis, and a_y is the body lateral acceleration.

Pitch motion equation:

$$\begin{aligned} I_{yy}\dot{\omega}_y - (I_{zz}-I_{xx})\omega_z\omega_x = {} & L_r\sum_{i=3}^{4}F_{zs_i} - L_f\sum_{i=1}^{2}F_{zs_i} \\ & + m_b g H_p \sin\theta + m_b a_x H_p \cos\theta \end{aligned} \tag{10}$$

where H_p is the height from the body's center of mass to the pitch axis, and a_x is the body longitudinal acceleration.

2.3.2 Suspension System Dynamics Model

This paper adopts a parameterized suspension model based on the roll center theory, considering the three-directional stiffness and damping characteristics of the suspension (longitudinal/lateral/vertical). Different from traditional suspension models that only consider vertical characteristics, the proposed model can accurately describe the influence of unsprung mass longitudinal and lateral motion on body attitude. The suspension force consists of elastic and damping forces:

$$\begin{cases} F_{xs_i} = k_{xi}dx_i + c_{xi}\dot{dx}_i \\ F_{ys_i} = k_{yi}dy_i + c_{yi}\dot{dy}_i \\ F_{zs_i} = k_{zi}(z_{w_i}-z_{s_i}) + c_{zi}(\dot{z}_{w_i}-\dot{z}_{s_i}) \end{cases} \tag{11}$$

where k_{xi}, k_{yi}, k_{zi} are the three-directional stiffness of the suspension, c_{xi}, c_{yi}, c_{zi} are the three-directional damping, dx_i and dy_i are the longitudinal and lateral displacements of the unsprung mass relative to the body, z_{w_i} is the wheel vertical displacement, and z_{s_i} is the vertical displacement of the body at the suspension connection point:

$$\begin{cases} z_{s_i} = z_b - L_f\theta - (-1)^i\frac{B_f}{2}\varphi, & i=1,2 \\ z_{s_i} = z_b + L_r\theta - (-1)^i\frac{B_r}{2}\varphi, & i=3,4 \end{cases} \tag{12}$$

where z_b is the vertical displacement of the body's center of mass.

The three-directional stiffness and damping parameters of the suspension are obtained through bench tests. The longitudinal stiffness is mainly used to describe suspension deformation during braking and acceleration, and the lateral stiffness is mainly used to describe suspension deformation during steering. For leaf spring suspensions , the longitudinal stiffness is approximately 1.5 to 2 times the vertical stiffness, and the lateral stiffness is approximately 0.8 to 1.2 times the vertical stiffness,e.g.,(Chen 2022).

2.3.3 Wheel Dynamics Model

Each wheel-rotor component is connected simultaneously to the drive/brake actuator and the tire component through a rotational physical port. The wheel rotation angle is treated as a potential variable, whereas the axle

DOI
10.3384/ecp223337

torque is treated as a flow variable. After connection, all components sharing the rotational port have the same angular position, and the algebraic sum of the torques at the connection is zero.

The wheel angular velocity is obtained from the time derivative of the wheel rotation angle. The driving or braking torque, the tire longitudinal reaction torque, the rolling-resistance torque, and the wheel inertial torque jointly satisfy the rotational dynamic balance:

$$I_w \dot{\omega}_{wi} = T_i - F_{xwi} R_w - F_{zwi} f_r R_w \tag{13}$$

where I_w is the wheel moment of inertia, T_i is the driving or braking torque, F_{xwi} is the tire longitudinal force, R_w is the wheel rolling radius, and f_r is the rolling resistance coefficient.

2.3.4 Tire Model

The tire component contains three types of physical interfaces: a translational hub port, a wheel-rotation port, and a road-contact port. The hub port transmits the position and velocity of the wheel center, together with the tire forces and moments. The rotational port describes the wheel rotation and the tire reaction torque. The road-contact port provides the road height, surface velocity, friction coefficient, and contact force.

The tire slip ratio, slip angle, vertical load, longitudinal force, lateral force, and self-aligning moment are algebraic variables within the tire component. These variables are determined jointly by the tire-contact kinematics, the vertical contact equations, the combined-slip Magic Formula, and the conservation equations imposed by the physical ports. The Magic Formula is used as the nonlinear constitutive relation that closes the relationship between tire forces and slip states. It does not operate as an independent signal-processing block with a predefined inputoutput direction.

The industry-standard MF-PAC2002 magic tire model is adopted to calculate the six-component tire force under combined longitudinal slip and cornering conditions. This model is fitted by test data and can accurately describe the nonlinear characteristics of tires under large slip ratio and large slip angle, suitable for SUV tire modeling. Compared with the latest MF-Tire 6.2 model, MF-PAC2002 has a smaller computational load while ensuring accuracy, making it more suitable for real-time simulation,e.g.,(Pacejka 2012).

The longitudinal force under pure longitudinal slip is:

$$F_{x0} = D_x \sin\left[C_x \arctan\left(B_x k_x + E_x\left(\arctan(B_x k_x) - B_x k_x\right)\right)\right] \tag{14}$$

where $k_x = \lambda + S_{Hx}$ is the corrected slip ratio, S_{Hx} is the horizontal offset, and B_x, C_x, D_x, E_x are the magic tire parameters.

The lateral force under pure cornering is:

$$\begin{aligned} F_{y0} = D_y \sin\big[C_y \arctan\big(B_y \alpha_y \\ + E_y\left(\arctan(B_y \alpha_y) - B_y \alpha_y\right)\big)\big] \end{aligned} \tag{15}$$

where $\alpha_y = \alpha + S_{Hy}$ is the corrected slip angle, and S_{Hy} is the horizontal offset.

Under combined working conditions, considering the mutual influence of cornering on longitudinal force and longitudinal slip on lateral force, correction coefficients $G_{x\alpha}$ and G_{yk} are introduced:

$$F_x = G_{x\alpha}(\alpha, \lambda, F_z) F_{x0}, \quad F_y = G_{yk}(\alpha, \lambda, F_z) F_{y0} \tag{16}$$

The calculation formulas for tire self-aligning torque, overturning torque, and rolling resistance torque are detailed in,e.g.,(Bakker 1989).

2.3.5 Driver and Control Boundary Module

The driver module adopts a hierarchical control structure to simulate human driving behavior. The upper layer uses a single-point preview following algorithm to calculate the desired front wheel steering angle based on the lateral deviation between the preview point and the reference path. The lower layer uses a PID controller to adjust the desired longitudinal force according to the error between the actual longitudinal velocity and the reference velocity, realizing accurate tracking of the target speed and path.

The driver and controller constitute the external signal-based control layer of the vehicle system. Therefore, this module intentionally uses directional control interfaces. It receives the reference vehicle speed, reference path, and feedback variables such as the current vehicle velocity, yaw angle, and position. It then generates the front-wheel steering command and the desired driving or braking torque.

The upper-level controller uses a single-point preview algorithm to determine the steering command, while the lower-level controller calculates the longitudinal control command from the vehicle-speed error. The causal signal interface is restricted to the external control boundary. It does not determine the force-transmission direction inside the vehicle plant. Once the control commands are applied at the vehicle boundary, the body, suspension, wheel, tire, and road components are still solved simultaneously through acausal physical connections and conservation equations.

2.3.6 Tire-Contact Kinematics and Slip-State Formulation

The tire-contact kinematics and slip-state formulation are encapsulated within the tire physical component rather than implemented as an independent one-way signal-conversion module. The relative velocity at the tire contact is determined from the velocity of the wheel hub and the road-surface velocity. The wheel angular velocity is obtained from the time derivative of the rotation angle associated with the rotational port.

The longitudinal and lateral velocity components in the local tire coordinate system, together with the wheel circumferential velocity, are used to formulate the algebraic constraints for the slip ratio and slip angle. These slip states and the vertical tire load participate directly

in the Magic Formula tire equations. The resulting tire forces then affect the wheel, unsprung mass, suspension, and vehicle-body motion through the connected physical ports.

Consequently, wheel speed, wheel-center velocity, slip ratio, tire force, and vehicle response form a closed coupled relationship. They are solved simultaneously by MWorks as part of the same system of differential-algebraic equations, rather than being passed sequentially through a causal calculation chain.

3 MODEL IMPLEMENTATION AND CROSS-PLATFORM VALIDATION BASED ON MWORKS

3.1 Model Implementation Based on Modelica

The proposed vehicle dynamics model was implemented on the MWorks Sysplorer 2025a platform using Modelica 4.0.0. Both textual and graphical modeling methods were used. Unlike a conventional block-diagram implementation, the vehicle physical plant is formulated as a differential–algebraic equation network. Each physical component describes its constitutive equations and conservation laws without prescribing a computational direction for the interaction variables.

Three types of physical connectors are defined for the vehicle plant. The FramePort connector carries position, velocity, and heading as potential variables, and force and yaw moment as flow variables. The SpinPort connector carries wheel rotation as a potential variable and axle torque as a flow variable. The RoadPort connector describes road height, road-surface velocity, adhesion, and tire–road contact quantities. The connect() equations impose kinematic compatibility and effort/flow conservation simultaneously. Consequently, the forces exchanged between the body, suspension, unsprung mass, wheel, tire, and road are solved as part of the same coupled equation system. The Modelica compiler determines the equation sorting and numerical solution procedure during translation.

To improve model reusability and facilitate the replacement of individual components, the model is organized into the following five layers:

1. **Basic component layer:** reusable Modelica library components and vehicle-specific spring, wheel, tire, and road components with parameterized physical connectors.

2. **Physical subsystem layer:** body, suspension, unsprung-mass, wheel, tire, and road components connected through FramePort, SpinPort, and RoadPort. Their interaction forces and torques are solved simultaneously rather than transmitted through one-way signals.

3. **Vehicle assembly layer:** integration of the sprung-body, unsprung-mass, and wheel-spin components into the complete 22-DOF vehicle model with unified parameters and initialization.

4. **Experiment-boundary layer:** definition of initial conditions, road adhesion, steering angle, and drive or braking torque. These externally imposed commands are causal test boundaries rather than internal physical signal paths.

5. **Control layer:** replaceable driver or control algorithms that generate boundary commands without modifying the acausal body–suspension–wheel–tire network.

The distinction between physical and signal interfaces is essential. Inside the Vehicle22DOF model, the interaction among the body, suspension, unsprung masses, wheels, tires, and road is established exclusively through physical connectors and connect() equations. The variables driveTorque and deltaFront represent external drive-torque and steering boundaries, respectively. Likewise, variables such as vehicle position, velocity, acceleration, yaw angle, and yaw rate are exposed only as observation aliases for result export and visualization; they do not form a feedback signal chain that determines the internal physical solution.

This architecture allows an individual tire, suspension, road, or control component to be replaced without manually redesigning the computational direction of the complete vehicle model. After a component is replaced, the Modelica compiler reassembles the connected equations and determines a new solution structure automatically. Therefore, Fig. 2 should distinguish the connector-based physical network inside the vehicle plant from the external command and observation interfaces used by the test conditions and control algorithms.

Each module is connected through standardized physical and signal interfaces, supporting independent testing and replacement of modules. For example, different tire characteristics can be simulated by replacing the tire model module; different control algorithms can be verified by replacing the control layer module. Fig. 2 shows the architecture diagram of the 22-DOF vehicle dynamics model built in Mworks.

3.2 Simulation Verification Scheme Design

To assess the dynamic responses of the proposed model, the Modelica outputs are compared with both a 22-DOF Simulink model and road-test measurements. The Simulink model is treated as a cross-platform reference rather than an error-free validation standard, and agreement with the road-test data is evaluated separately for each signal and metric.

According to GB/T 6323-2014 *Test Methods for Vehicle Handling Stability* and GB/T 12543-2009 *Test Methods for Vehicle Acceleration Performance*, three typical

DOI
10.3384/ecp223337

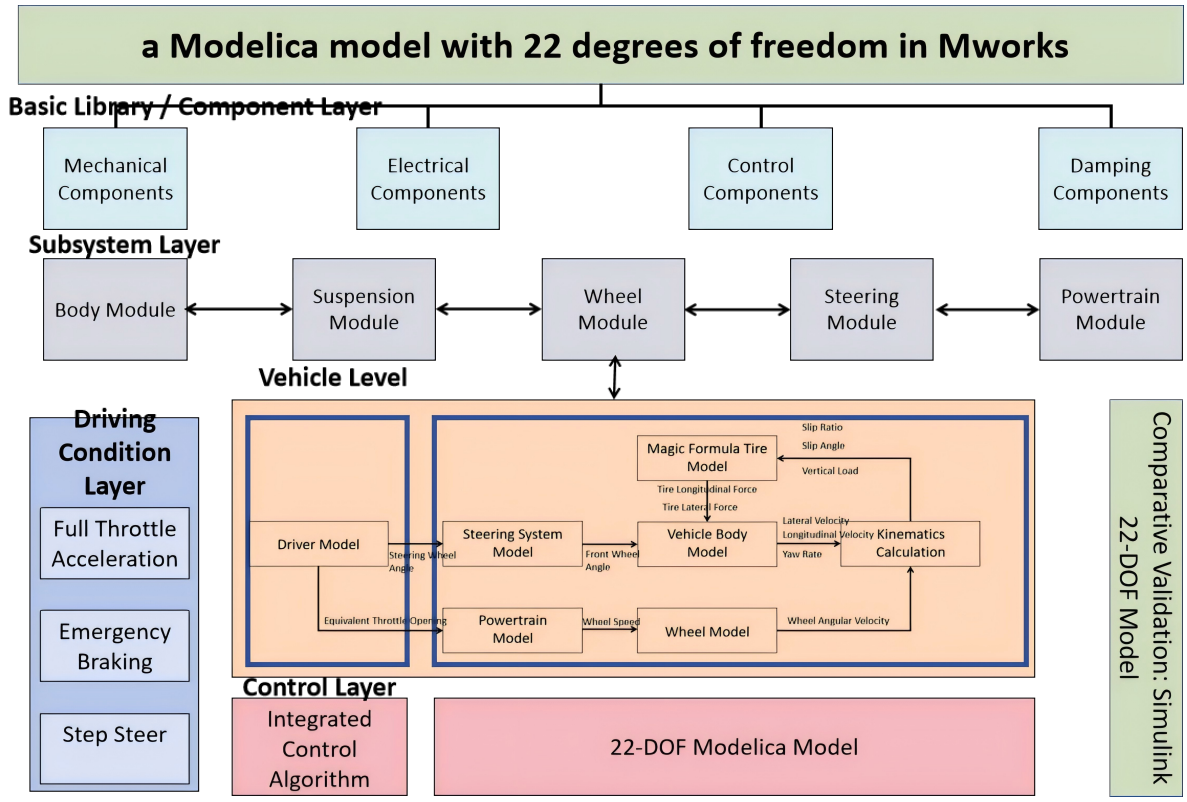


Figure 2. Top-level architecture of the 22-DOF vehicle dynamics model in Mworks

working conditions are designed to cover the core evaluation indicators of vehicle power performance and handling stability, as detailed in Table 1.

Table 1. Details of simulation working conditions

Test Item	*Details*
Full-throttle acceleration test	High-adhesion road ($\mu = 0.85$); Key parameters: longitudinal displacement x, velocity v_x, acceleration a_x (time history curves); Evaluation: 0-100km/h acceleration performance
Emergency braking test	60% brake pedal input (initial speed 102km/h); Key parameters: longitudinal motion parameters (x, v_x, a_x) over time t (decay characteristics); Evaluation: Emergency braking effectiveness
Transient steering characteristic test	Constant speed 60km/h, steering wheel step input 60°; Key parameters: lateral acceleration a_y, yaw rate r, yaw angle ψ (step response curves); Evaluation: Steering system dynamic characteristics

The common vehicle parameters were matched between the two models wherever traceable; however, exact equivalence of all initial conditions and input boundaries cannot be claimed because of differences in the available datasets. The MWorks model was solved using DASSL with a 1 ms output interval and a relative tolerance of 10^{-6}, while the Simulink benchmark model used the fixed-step ode4 solver with a step size of 1 ms. Therefore, the comparison reflects both physical-model and numerical-implementation differences, and the observed deviations cannot be attributed solely to the modeling language.

3.3 Simulation Result Analysis

To assess the dynamic responses of the proposed 22-DOF fully coupled vehicle model, acceleration, braking, and steering-step maneuvers are considered. The calibrated Modelica results are compared with the available Simulink and road-test datasets, as shown in Fig. 3. Because some initial conditions and actuator boundaries are not completely identical among the three sources, the results are interpreted as a calibration-stage response comparison rather than a strict same-input accuracy ranking.

3.3.1 Dynamic Performance Simulation Results

The longitudinal responses under the investigated acceleration and 60% braking maneuvers are compared in Fig. 3, while the steering-step responses are discussed separately in Section 3.3.2.The three data sources exhibit similar overall trends, although their agreement varies among acceleration, velocity, and displacement. The Modelica model reproduces the road-test braking distance and speed-decay trend well. Therefore, the results support the models ability to reproduce the main longitudinal dynamic characteristics under the investigated conditions.

Under the full-throttle acceleration condition, the

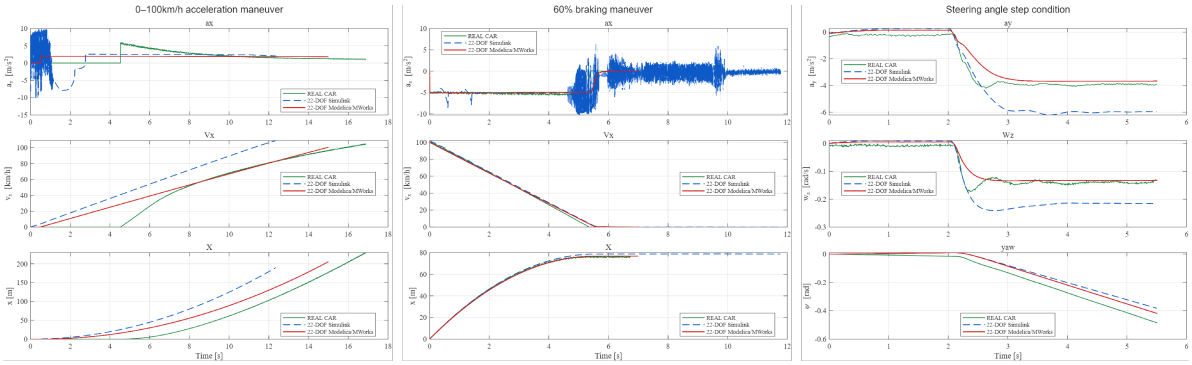


Figure 3. Comparison of Simulation Results Between Modelica Model, Simulink Benchmark Model and Real Vehicle Test Under Typical Working Conditions

model can accurately capture the full-range acceleration characteristics of the vehicle from middle speed to high speed, and the simulation curve matches with the measured acceleration speed trend of the real vehicle.

Full-throttle acceleration quantitative analysis:

Longitudinal acceleration: From vehicle launch to the first crossing of 100 km/h, the median longitudinal accelerations of the Modelica model, Simulink model, and road vehicle are 1.9427, 2.4729, and 1.9964 m/s^2, respectively. The Modelica deviation from the road-test value is 2.69%, whereas the Simulink deviation is 23.87%.

Speed response: With the first crossing of 0.5 km/h defined as vehicle launch, the 0.5–100 km/h times are 14.315 s for Modelica, 11.112 s for Simulink, and 11.132 s for the road vehicle. Modelica is slower in this metric, but its speed response remains monotonic, continuous, and consistent with its longitudinal acceleration.

Driving distance: At the first 100 km/h crossing of each source, the Modelica, Simulink, and road-test displacements are 202.50, 156.26, and 196.40 m, respectively. Modelica therefore provides a closer estimate of the distance required to reach the target speed.

The small response deviation in the low-speed section mainly comes from the cross-platform implementation difference of the torque output characteristics of the power system, and the characteristic deviation in the high-speed section mainly comes from the setting difference of air resistance-related parameters. The above deviations are within the engineering allowable error range , and the response consistency among the model, the benchmark model, and the real-vehicle measured data is significantly improved after directional calibration.

Under the emergency braking condition, the model can accurately reproduce the speed attenuation and deceleration change characteristics of the whole braking process, and the simulation change law is basically consistent with the measured data of the real-vehicle emergency braking.

Emergency braking quantitative analysis:

Braking deceleration: In the 20–80 km/h interval, the median accelerations of Modelica, Simulink, and the road vehicle are -4.9798, -5.0513, and -5.2342 m/s^2, respectively. Their deviations from the road-test magnitude are 4.86% and 3.49%.

Speed attenuation: Using the first decrease below 0.5 km/h as the stopping criterion, the stopping times are 5.587 s for Modelica, 5.663 s for Simulink, and 5.322 s for the road vehicle. The Modelica and Simulink deviations from the road test are 4.98% and 6.41%, respectively.

Braking distance: The corresponding stopping distances are 76.768 m for Modelica, 78.776 m for Simulink, and 76.626 m for the road vehicle. Modelica differs from the road test by only 0.142 m, or 0.19%, whereas Simulink differs by 2.150 m, or 2.81%. Stopping distance is therefore the clearest Modelica advantage in the braking maneuver.

The remaining differences are most evident near brake application and during the low-speed stopping stage. Because the initial speeds and actuator boundaries are not completely identical across the three sources, these differences cannot be attributed uniquely to brake hysteresis or tire-slip calculations without additional controlled sensitivity tests. The present Modelica calibration nevertheless provides close agreement with the road-test stopping distance.

3.3.2 Handling Stability Simulation Results

The response comparison of the angle-step steering condition (constant speed 60 km/h, 60° steering wheel step input) is shown in Fig.3. It can be seen from the curve comparison results that the 22-DOF full-coupled model built in this paper can accurately capture the transient response and steady-state characteristics of the vehicle during steering, and completely reproduce the whole process of rise, peak, and steady-state convergence of yaw rate and lateral acceleration; the simulation results are highly consistent with the Simulink benchmark model and the real-vehicle angle-step steering measured data in response trend, peak characteristics, and phase characteristics.

Angle-step steering quantitative analysis:

Yaw rate: The steady yaw rates of Modelica, Simulink, and the road vehicle are -0.13270, -0.21496, and -0.14071 rad/s, respectively, showing that Modelica better reproduces the steady yaw response.

Lateral acceleration: Over the steady interval from 3.5

DOI
10.3384/ecp223337

to 5.4 s, the Modelica, Simulink, and road-test lateral accelerations are -3.6767, -5.9864, and -3.9100 m/s^2, respectively. Modelica therefore reproduces the steady lateral-response gain more accurately.

Yaw angle:At $t = 5.4$ s, the Modelica, Simulink, and road-test yaw angles are -0.40472, -0.37035, and -0.47034 rad, respectively. The Modelica deviation is 13.95%, compared with 21.26% for Simulink. The remaining difference is mainly caused by the accumulation of transient yaw-rate errors.

The Modelica model shows its main advantages in state consistency, continuous numerical response, stopping-distance prediction, and steady steering response. The acceleration, speed, and displacement states satisfy their differential and integral relations, while the yaw rate is consistently related to the Euler-angle derivatives through the 3-2-1 rotational kinematic transformation. These properties arise from simultaneous solution of the acausal physical-connection equations rather than from post-processing of the plotted data. The current results do not imply that Modelica is superior in every metric; the low-speed powertrain boundary and transient steering parameters still require further identification.

3.3.3 ACC Control Integration Based on Modelica Boundary Interfaces

Following the open-loop dynamic assessment, an ACC cruise-speed controller is connected to the external command boundary of the proposed vehicle model. The controller and the vehicle plant form a closed-loop system without modifying the internal acausal connections among the body, suspension, wheels, tires, and road.

The controller integration framework contains three functional interfaces:

1. *Observation interface*: provides the vehicle speed and longitudinal acceleration calculated by the coupled vehicle model;
2. *Control interface*: calculates a normalized drive command according to the reference speed and the observed vehicle states;
3. *Actuation interface*: converts the normalized command into four wheel-torque boundary conditions.

The observed vehicle states are read-only result variables, whereas the wheel torques are external causal commands. Therefore, the use of a causal ACC controller does not change the acausal differential-algebraic formulation of the internal vehicle plant.

The speed-tracking error is defined as

$$e(t) = v_{\mathrm{ref}}(t) - v(t). \tag{17}$$

A proportional-integral controller with longitudinal-acceleration feedback is adopted:

$$u(t) = \mathrm{sat}\left[K_p e(t) + K_i \int_0^t e(\tau)\,\mathrm{d}\tau - K_d a_x(t), 0, 1\right], \tag{18}$$

where $u(t)$ is the normalized drive command, $a_x(t)$ is the observed longitudinal acceleration, and $\mathrm{sat}(\cdot)$ limits the command to the range $[0,1]$.

The commanded torque applied to each wheel is calculated as

$$T_i(t) = T_{\max} u(t), \qquad i = 1,2,3,4, \tag{19}$$

where $T_{\max}$ denotes the maximum torque boundary of each wheel.

The controller parameters are set to $K_p = 0.8$, $K_i = 0.02$, and $K_d = 0.02$. The simulation is conducted on a flat road with an initial speed of 57 km/h, a target speed of 100 km/h, a road-friction coefficient of $\mu = 0.85$.

As shown in Fig.4, the closed-loop vehicle speed approaches the target value smoothly, while the bounded drive command avoids abrupt torque changes. The results demonstrate that a causal speed controller can be integrated through replaceable boundary interfaces without changing the acausal formulation of the 22-DOF vehicle plant. This architecture provides a reusable basis for subsequent ACC, AEB, and vehicle-stability control applications.

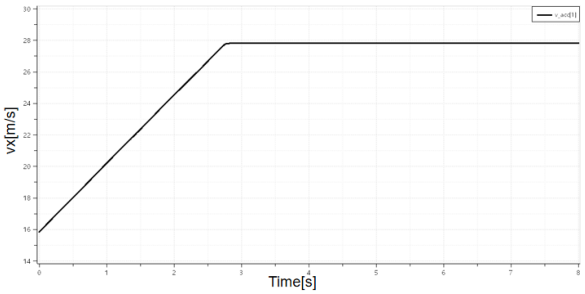


Figure 4. ACC cruise speed tracking curve

4 CONCLUSIONS AND FUTURE WORK

4.1 Conclusions

A connector-based 22-DOF vehicle dynamics model has been implemented in MWorks Sysplorer. The model comprises six sprung-body motions, twelve unsprung-mass translations, and four wheel-spin motions. Its physical subsystems exchange forces and torques through acausal connectors, while driver and controller commands remain at causal external boundaries. This structure separates physical conservation equations from control-signal direction and supports component replacement without manually redefining the internal computational order.

The three-source comparison demonstrates consistent overall dynamic trends and close agreement in key performance indicators. In particular, the proposed Modelica model reproduces the road-test braking distance with a relative difference of 0.18%, while the differences in steady lateral acceleration and yaw rate are 5.97% and 5.90%,

respectively. The model also maintains physically consistent relationships among acceleration, velocity, displacement, and vehicle-attitude states. These results demonstrate the effectiveness and engineering applicability of the proposed acausal 22-DOF vehicle dynamics model under the investigated operating conditions.

4.2 Future Work

Future work will be further deepened from three aspects: 1. Introduce the influence of tire temperature, wear, and road adhesion coefficient changes, establish a more accurate nonlinear tire model, and further improve the simulation accuracy under limit conditions; 2. Expand the control algorithm library, integrate advanced driving assistance systems such as ESP, AEB, and LCC, and develop hardware-in-the-loop simulation interfaces to realize real-time verification of control algorithms; 3. Carry out real-vehicle tests, further calibrate the model with real-vehicle test data, realize the precise mapping between the digital twin model and physical vehicles, and support the full life cycle management of vehicles.

References

Bakker, E., Pacejka, H. B. and Lidner, L. (1989). "A new tire model with an application in vehicle dynamics studies". SAE Technical Paper 890087, pp. 83-95.

Baidu Apollo. (2023). "Baidu Apollo Autonomous Driving Virtual Simulation Technology White Paper". Technical Report, Baidu (China) Co., Ltd., Beijing.

Chen, Z., Liu, Y. and Ma, B. (2022). "Parameter identification of leaf spring suspension for heavy trucks based on bench test". *Journal of Mechanical Engineering*, vol. 58, no. 12, pp. 156-164. (In Chinese)

Drogies, S. and Bauer, M. (2002). "Modeling road vehicle dynamics with Modelica". SAE Technical Paper 2002-01-1219.

European Commission. (2019). "Guidelines on the exemption procedure for the EU approval of automated vehicles". Technical Report, European Commission, Brussels. DOI: 10.2833/567890.

Fritzson, P. (2014). *Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Systems Approach*. John Wiley & Sons.

Guo, K. H. and Lu, D. (2007). "UniTire: unified tire model for vehicle dynamic simulation". *Vehicle System Dynamics*, vol. 45, suppl. 1, pp. 79-99.

ISO 21448:2021. (2021). "Road vehicles - Safety of the intended functionality". International Organization for Standardization, Geneva.

Janin, G. et al. (2007). "Modelica-Based Co-Simulation of Vehicle Dynamics and Driver Models". *Journal of Dynamic Systems, Measurement, and Control*, vol. 129, no. 4, pp. 529-536.

Li, K. Q., Wang, H. W., Li, S. B. et al. (2023). "Research on longitudinal and lateral collaborative control algorithm for distributed drive electric vehicles". *Automotive Engineering*, vol. 45, no. 6, pp. 789-797. (In Chinese)

Li, S., Wang, Y. and Li, K. (2023). "Adaptive PID control for heavy truck ACC system considering load variation". *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 11, pp. 11876-11887.

Liu, J. (2010). "Modelica-based modeling of multibody dynamic models". *China Mechanical Engineering*, vol. 21, no. 5, pp. 1088-1093. (In Chinese)

Liu, S., Zhao, J. and Sun, W. (2023). "Performance comparison of Modelica native interface and FMI interface for real-time simulation". In: 14th International Modelica Conference, pp. 457-464.

Modelica Association. (2021). "Modelicaő - A Unified Object-Oriented Language for Systems Modeling, Version 3.5". URL: https://www.modelica.org/documents/ModelicaSpec35.pdf (visited on 2026-05-13).

Otter, M., Elmqvist, H. and Mattsson, S. E. (1998). "Modelica - A language for system modeling and simulation". *Simulation Practice and Theory*, vol. 6, no. 4, pp. 309-322.

Pacejka, H. B. (2012). *Tire and Vehicle Dynamics*. 3rd ed. Elsevier.

Shi, Q., He, Z. J. and He, L. (2021). "Research progress on braking stability control of new energy vehicles". *Automotive Engineering*, vol. 43, no. 10, pp. 1289-1296. (In Chinese)

Tongyuan Soft Control. (2024). "Mworks System Modeling and Simulation Platform Technology White Paper". Technical Report, Suzhou Tongyuan Soft Control Information Technology Co., Ltd., Suzhou.

Wang, X., Chen, L. and Zhang, Q. (2024). "Quantitative error tracing method for cross-platform vehicle dynamics simulation". *Simulation Modelling Practice and Theory*, vol. 132, p. 102876.

Wang, Y. F., Li, H., Chen, H. B. et al. (2022). "Parameter identification method of vehicle lateral dynamics model based on physics-data fusion". *China Journal of Highway and Transport*, vol. 35, no. 5, pp. 189-201. (In Chinese)

Wei, T. J. (2015). "Research on 22-DOF vehicle dynamics modeling and simulation analysis". Doctoral Dissertation, Hunan University, Changsha. (In Chinese)

Yu, Z. S. (2009). *Automotive Theory*. 5th ed. China Machine Press, Beijing. (In Chinese)

Zhang, Y., Li, J. and Wang, H. (2023). "Influence of unsprung mass longitudinal and lateral motions on heavy truck handling stability". *International Journal of Vehicle Design*, vol. 91, nos. 2-3, pp. 187-206.

Modelica-Based Modeling of PMSM Control Methods

Yuang Yang[1] Hongyou Zhong[1] Yang Qi [1]
[1] Northwestern Polytechnical University, Xi'an, China
{firstname.surname}@mail.nwpu.edu.cn

Abstract

The strong coupling, nonlinearity, and multi-physical-domain nature of permanent magnet synchronous motor (PMSM) drive systems pose significant challenges to unified modeling and control-strategy verification. To address these challenges, this paper proposes a system-level modeling frame-work for PMSM control methods based on the Modelica language. By exploiting Modelica's object-oriented, acausal, and equation-based modeling features, the proposed framework provides modular encapsulation of the motor, inverter, modulation strategy, and thermal network under a unified physical-connection architecture. First, a basic component library is developed, including the d–q current controller, the Clark/Park transformations, and space-vector pulse-width modulation (SVPWM). On this basis, a PMSM vector control model is established, achieving the approximate decoupled control of the d- and q-axis currents. Furthermore, equivalent thermal models are constructed for different cooling configurations to characterize the temperature rise caused by copper, iron, and permanent-magnet losses. Finally, a maximum-torque-per-ampere (MTPA) control model is introduced to improve current utilization and overall system efficiency. Simulation results verify the modularity, scalability, and reusability of the proposed framework, and demonstrate its capability to serve as a unified platform for multi-physical-domain co-simulation and control-strategy verification of PMSM drive systems.

Keywords—Permanent magnet synchronous motor (PMSM), Modelica, vector control, maximum torque per ampere (MTPA), multi-physical-domain modeling.

1 Introduction

With the rapid development of electrified transportation, industrial automation, aerospace electric propulsion, and high-performance servo systems, the permanent magnet synchronous motor (PMSM) has become one of the key actuating components in modern electric-drive systems due to its high efficiency, high power density, high torque density, and excellent dynamic response. A PMSM drive system is essentially a strongly coupled, nonlinear, and multi-physical-domain coupled system, whose performance is determined not only by the motor parameters themselves, but also by the switching behavior of the inverter, the current regulator, coordinate transformations, modulation strategies, thermal losses, and the cooling structure. Consequently, the establishment of a unified, modular, and extensible system-level mode-ling methodology is of great significance for control-strategy design, dynamic-performance evaluation, and multi-physical-field collaborative optimization of PMSM drive systems.

Modelica is an object-oriented, acausal, and equation-based multi-physical-domain modeling language that is well suited to the modeling and simulation of complex heterogeneous systems (Modelica Association 2014) Compared with conventional signal-flow-based modeling approaches, Modelica places greater emphasis on physical-connection relationships and component reus-ability, and can describe inverters, motors, controllers, and thermal networks within a single unified framework (Fritzson et al.,2020). In recent years, Modelica has been increasingly applied to electric-drive systems, electric-vehicle powertrains, and multi-physical-domain digital twins, providing an effective tool for reconfigurable modeling of PMSM control systems (Fotias et al.,2021).

Among the various PMSM control methods, vector control is one of the most widely adopted approaches in high-performance PMSM drives. By means of the Clarke and Park transformations, the three-phase variables are mapped into the synchronously rotating reference frame, where approximate decoupled control of the d- and q- axis currents can be achieved (Lu et al.,2021). To deal with current-sampling errors, parameter perturbations, and dynamic-response requirements, improved current vector control schemes and predictive current control strategies have been proposed to enhance the robustness and control accuracy of PMSM drive systems (Yuan et al.,2019, W. Lee et al.,2019). However, as motor power density continues to increase, temperature rise caused by stator copper losses, iron losses, and permanent-magnet losses becomes more prominent. Excessive temperature can result in insulation aging, permanent-magnet demagnetization, and efficiency degradation. It is therefore necessary to incorporatecooling configurations and temperature-estimation models into the overall system modeling (Li et al.,2021, Wang et al.,2022).

In addition, to improve current utilization and overall efficiency, maximum-torque-per-ampere (MTPA) control has been widely adopted in interior PMSMs. By optimally allocating the d- and q- axis currents, MTPA achieves a larger electromagnetic torque for a given current magnitude, or equivalently, a smaller stator current and copper loss for a given torque demand (Han et al.,2020, Feng et al.,2022). However, few existing Modelica-based simulation studies on permanent magnet synchronous motors couple electrical and thermal fields to investigate the effects of different control strategies.

Motivated by the above background, this paper investigates Modelica-based modeling of PMSM control methods. First, basic modules including the d–q current controller, the coordinate transformations, and SVPWM are developed. A PMSM vector control model is then constructed on top of these basic modules. Subsequently, equivalent mod-

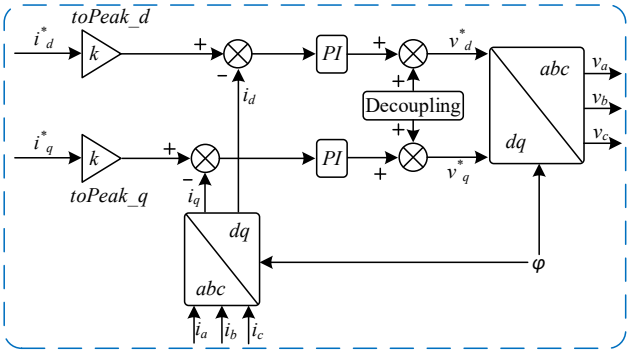


Fig. 1. Block diagram of the d–q current controller.

eling approaches are established for different cooling configurations. Finally, an MTPA control model is introduced, thereby providing a unified modeling framework for multi-physical-domain simulation and control-strategy verification of PMSM drive systems.

2 Basic Module Modeling

In this section, the basic modules used for PMSM control modeling are developed in the Modelica language on the Dymola platform.

2.1 DQ Current Controller Modeling

The DQ current controller is the core basic module of a PMSM vector control system. Its principal function is to regulate the d- and q- axis currents in the synchronously rotating reference frame and to generate the corresponding three-phase voltage references. The control block diagram is shown in Fig. 1.

In Fig. 1, i_d* and i_q* denote the d- and q- axis current references, φ is the rotor electrical angle, and i_{abc} denotes the measured three-phase currents. The output v_{abc}* represents the three-phase voltage references that are fed into the three-phase voltage source or the inverter modulation module.

The d–q current controller first transforms the three-phase measured currents i_{abc} into the synchronously rotating d–q reference frame through the toDQ module, yielding the actual feedback currents i_d and i_q. The rotor electrical angle φ is used as the synchronously rotating reference for the coordinate transformation. Meanwhile, the input d- and q- axis current references are converted from rms values to peak values so as to remain consistent with the peak d–q currents obtained from the instantaneous three-phase variables. The current errors are obtained as follows:

$$e_d = i_d^* - i_d \tag{1}$$

$$e_q = i_q^* - i_q \tag{2}$$

The errors ed and eq are fed into the PI regulators of the d- and q-axis, respectively. The PI controllers generate the d–q voltage control signals according to the current errors, and their parameters are related to the d- and q-axis inductances and the stator resistance:

$$v_{d,PI} = K_{pd}\left(e_d + \frac{1}{T_{id}}\int e_d dt\right) \tag{3}$$

$$v_{q,PI} = K_{pq}\left(e_q + \frac{1}{T_{iq}}\int e_q dt\right) \tag{4}$$

$$K_{pd} = \frac{0.5}{L_d 2\pi f_c}, T_{id} = \frac{R_s}{L_d 2\pi f_c} \tag{5}$$

$$K_{pq} = \frac{0.5}{L_q 2\pi f_c}, T_{iq} = \frac{R_s}{L_q 2\pi f_c} \tag{6}$$

where f_c is the characteristic frequency of the controller, R_s is the stator phase resistance, and L_d and L_q are the d- and q-axis inductances, respectively. By regulating the d- and q-axis currents independently, the controller realizes decoupled control of the excitation and torque components.

Furthermore, to address the cross-coupling terms in the d–q voltage equations, a feed-forward decoupling network is added to the model. The electrical angular speed and the approximate permanent-magnet flux linkage are defined as

$$\omega_e = p\frac{d\varphi}{dt} \tag{7}$$

$$\psi_M = \frac{\sqrt{2}V_{s,oc}}{2\pi f_s} \tag{8}$$

where p is the number of pole pairs, $V_{s,oc}$ is the rms open-circuit phase voltage at the rated frequency, and f_s is the rated frequency. The corresponding d- and q-axis feed-forward decoupling voltages can be expressed as

$$V_d = \sqrt{2}\left(R_s i_d^* - \omega_e L_q i_q^*\right) \tag{9}$$

$$V_q = \sqrt{2}\left(R_s i_q^* + \omega_e L_d i_d^*\right) + \omega_e \psi_M \tag{10}$$

These decoupling terms include the stator-resistance vol-tage drop, the cross-coupling terms, and the back-EMF compensation due to the permanent magnets. With this decoupling network, the dynamic coupling between the d- and q-axes is mitigated, and the response speed and accuracy of the current loop are improved.

Finally, the PI controller outputs are added to the decoupling voltages to form the d- and q-axis voltage references:

$$v_d^* = v_{d,PI} + V_d \tag{11}$$

$$v_q^* = v_{q,PI} + V_q \tag{12}$$

The d–q voltage references vd* and vq* are then transformed back to the three-phase stationary frame through the FromDQ module, yielding the three-phase voltage reference output that drives the three-phase voltage source or the inverter modulation module and closes the PMSM current loop.

2.2 Clarke Transformation Module Modeling

The Clarke transformation maps voltage or current variables from the three-phase stationary reference frame to the two-phase stationary α–β reference frame. In the Modelica model developed in this paper, the Clarke-transformation module takes the three-phase signals u_a, u_b, and u_c as inputs and provides u_α and u_β as outputs, which are subsequently fed into the SVPWM module to generate the gating signals for the inverter. The mathematical expression is given by

DOI
10.3384/ecp223347

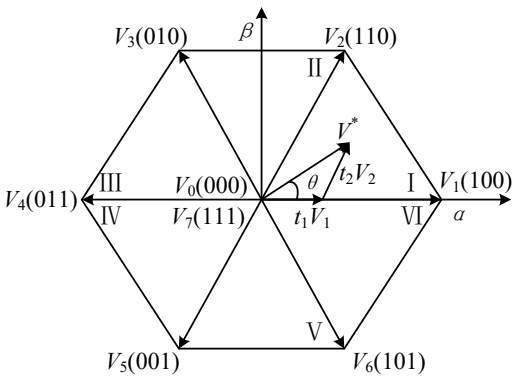


Fig. 2. Space-vector diagram of the SVPWM scheme.

$$\begin{bmatrix} u_\alpha \\ u_\beta \end{bmatrix} = \frac{2}{3}\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} & -\frac{\sqrt{3}}{2} \end{bmatrix}\begin{bmatrix} u_a \\ u_b \\ u_c \end{bmatrix} \tag{13}$$

Specifically, the α- and β-axis components can be written as

$$u_\alpha = \frac{2}{3}(u_a - \frac{1}{2}u_b - \frac{1}{2}u_c) \tag{14}$$

$$u_\beta = \frac{\sqrt{3}}{3}(u_b - u_c) \tag{15}$$

2.3 SVPWM Module Modeling

Space-vector pulse-width modulation (SVPWM) is a widely adopted modulation strategy for three-phase voltage source inverters. Compared with conventional sinusoidal pulse-width modulation, SVPWM is formulated from the standpoint of space voltage vectors of the three-phase inverter: the desired reference voltage vector is synthesized within each switching period by appropriately allocating the dwell times of two adjacent active vectors and the zero vectors.

The SVPWM module developed in this paper is implemented in pure Modelica code, and integrates reference-vector limiting, angle calculation, sector identification, dwell-time calculation, and the generation of the six gating signals.

The inputs of this module are the voltage reference vector expressed in the α–β reference frame, and the outputs are six Boolean gating signals corresponding to the upper and lower switching devices of the three legs of the inverter. Six active switching vectors are defined in the code, as shown in Figure 2. In addition to the six active vectors, there are two zero vectors, namely [0,0,0] and [1,1,1]. The fundamental idea of SVPWM is to synthesize the desired reference vector within each switching period by combining two adjacent active vectors and the zero vectors.

Prior to each sampling instant, the model first computes the magnitude of the reference space vector:

$$|u_{ref}| = \sqrt{u_\alpha^2 + u_\beta^2} \tag{16}$$

and then performs normalization and limiting. In the code, the magnitude is rewritten as

$$u_{Ref} = \min\left(\frac{\sqrt{u_1^2 + u_2^2}}{(2/3)u_{Max}}, \cos\frac{\pi}{6}\right) \tag{17}$$

where u_{Max} denotes the maximum allowable space-vector length. The factor $(2/3)u_{Max}$ scales the input vector to a per-unit representation, while $\cos(\pi/6)$ limits the modulation range and prevents the reference vector from exceeding the linear modulation region.

The angle of the reference vector is then calculated as

$$\phi_{ref} = \arctan 2(u_\beta, u_\alpha) \tag{18}$$

Since atan2 returns values in $(-\pi, \pi]$, the model further maps the result into the range $(0, 2\pi]$:

$$\phi_{pos} = \begin{cases} \phi_{ref} + 2\pi, & \phi_{ref} < 0 \\ \phi_{ref}, & \phi_{ref} \geq 0 \end{cases} \tag{19}$$

The space-vector plane of the three-phase inverter is divided into six sectors, each spanning π/3. The sector index is determined as

$$k_a = \text{integer}\left(\frac{\phi_{pos}}{\pi/3}\right) \tag{20}$$

with the left-boundary active vector index given by k itself and the right-boundary active vector index given by

$$k_b = \begin{cases} 0, & k_a \geq 5 \\ k_a + 1, & k_a < 5 \end{cases} \tag{21}$$

so that the cyclic transition from sector 6 to sector 1 is naturally handled. The relative angle of the reference vector within the current sector is

$$\phi_{sec} = \phi_{pos} - k_a\frac{\pi}{3} \tag{22}$$

After the sector is identified, SVPWM computes the relative dwell times of the two adjacent active vectors and the zero vectors within one switching period. Let t_a denote the dwell time of the left-boundary active vector, t_b the dwell time of the right-boundary active vector, and t_c the dwell time of the zero vectors, with

$$\begin{aligned} u_{ref} &= t_a u_a + t_b u_b + t_0 0 \\ t_a + t_b + t_0 &= 1 \end{aligned} \tag{23}$$

Based on the geometrical projection of the reference vector within the current sector, the dwell times are computed as

$$\begin{aligned} t_b &= min\left(\frac{u_{Ref}\sin\phi_{sec}}{\sin(\pi/3)}, 1\right) \\ t_a &= \min\left(u_{Ref}\cos\phi_{sec} - t_b\cos(\pi/3), 1\right) \\ t_0 &= \max(1 - t_a - t_b, 0) \end{aligned} \tag{24}$$

where the min and max functions clamp the dwell times within a valid range, avoiding negative values or values exceeding one switching period caused by an over-sized reference vector or numerical errors.

Once the dwell times have been determined, the model arranges the zero and active vectors within one switching period in a prescribed sequence. According to the relative position (time − T_0)/ T_s within the switching period, the zero vector, the left-boundary active vector, the right boundary active vector, and their reverse sequences are selected in turn, thereby generating the switching states of the three upper legs. Finally, the three upper-leg signals and their complementary signals are combined into six gating signals. The proposed Modelica-based SVPWM module,

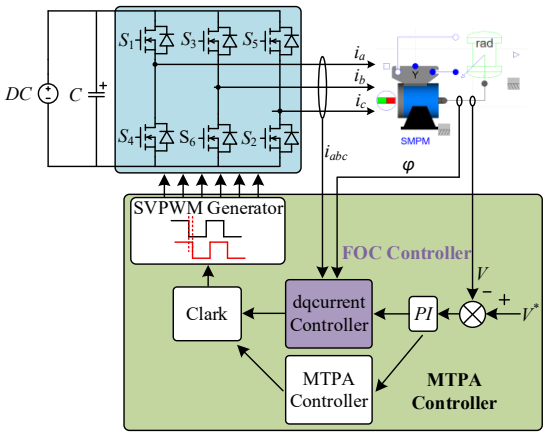


Fig. 3. Block diagram of the PMSM vector control system.

together with the upstream d–q current controller and Clarke transformation, as well as the downstream three-phase inverter model, constitutes a complete PMSM vector control system.

3 Proposed Control Scheme

3.1 PMSM Vector Control

Fig. 3 shows the block diagram of the proposed PMSM vector control system, which consists of a speed outer loop, a current inner loop, the coordinate-transformation modules, the SVPWM modulation module, a voltage-source inverter, the PMSM body, a mechanical load, and the signal-sensing modules. A typical double-loop vector control structure is employed. The speed loop generates the q-axis current reference through

a PI controller according to the deviation between the speed command and the measured speed, while the d-axis current reference is set to zero, implementing the $i_d = 0$ control strategy that is commonly used for surface-mounted PMSMs. The three-phase stator currents are sampled by current sensors and fed into the d–q current controller, and the rotor mechanical angle measured by an angle sensor is fed back to the controller to enable decoupled current control in the synchronously rotating reference frame. The d–q current controller outputs three-phase voltage references u_a*, u_b*, and u_c*, which are converted by the Clarke transformation module to two-phase stationary voltage references and subsequently fed into the SVPWM module to generate the six gating signals that drive the three-phase inverter.

The inverter is built as a custom three-phase full-bridge module, in which the conduction parameters of the IGBTs and diodes—namely $R_{on,IGBT}$, $R_{on.diode}$, V_{knee}, t_{on}, and t_{off}—are specified. With these parameters, the inverter model can not only represent the ideal switching behavior, but can also account for the influence of device on-state resistance, diode voltage drop, and switching delays on system performance.

The PMSM body is modeled with a component from the Modelica Standard Library, which captures the electromagnetic relations, mechanical-motion equations, and the energy conversion between the electrical and mechanical ports of the machine. The stator windings are configured

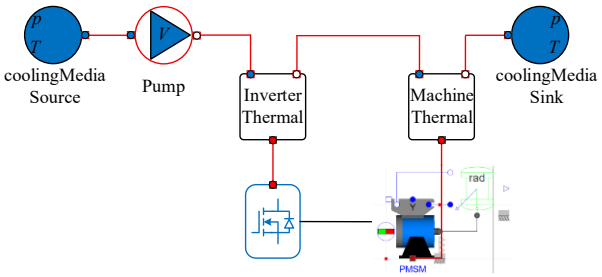


Fig. 4. Series liquid-cooling model of the PMSM control system.

as a Y connection via the Terminal Box. The three-phase outputs of the inverter are connected to the stator terminals, while the mechanical port is connected to a torque sensor, a load inertia, and a quadratic-speed-dependent load. This configuration allows both the electromagnetic dynamics and the mechanical-load response to be observed simultaneously.

The mechanical equation of motion is given by

$$J\frac{d\omega_m}{dt} = T_e - T_L - B\omega_m \qquad (25)$$
$$\omega_e = p\omega_m$$

where J is the equivalent moment of inertia, ω_m is the mechanical angular speed, T_e is the electromagnetic torque, T_L is the load torque, B is the viscous-friction coefficient, and p is the number of pole pairs. On the load side, the proposed model can be used to represent fan- or pump-type loads. Since the load torque of such applications is typically approximately proportional to the square of the speed, a quadratic-speed-dependent load is adopted in order to better reflect realistic operating conditions.

3.2 Liquid-Cooling Model of the PMSM Control System

During the operation of a PMSM drive system, both the inverter power devices and the motor itself generate considerable thermal losses. The IGBTs and antiparallel diodes in the inverter dissipate power during conduction and switching, while the stator windings, iron core, rotor, and permanent magnets within the machine also generate heat through electromagnetic interactions. If this heat is not dissipated in a timely manner, the device junction temperatures and the winding temperatures will rise, degrading system reliability and, in severe cases, leading to device failure or demagnetization. Consequently, it is necessary to introduce a thermal-management model into the PMSM drive system and to perform a cooling analysis for both the inverter and the machine.

The liquid-cooling model shown in Fig. 4 is built on top of the original motor vector control system and adds an inverter liquid-cooling model and a motor liquid-cooling model, in which both the inverter and the motor body are represented by commercial-library components with loss calculation. The inverter module adopts an electrothermal coupling model, in which turn-on loss, turn-off loss and conduction loss are calculated through real-time voltage, current and table lookup. The motor thermal loss module consists of a heat collector, temperature sensors, a fluid cooling system, stator and rotor thermal mass modules, and fixed boundary temperatures. By simulating heat generated

DOI
10.3384/ecp223347

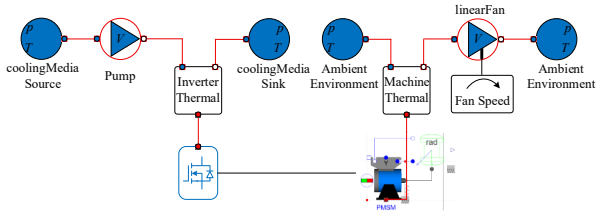


Fig. 5. Air-cooling model of the PMSM.

from copper loss in stator windings and core iron loss, as well as heat produced by rotor core loss and eddy current loss of permanent magnets, this module realizes the calculation of motor thermal loss and liquid cooling simulation via temperature monitoring and cooling control.

The liquid-cooling loop adopts a series configuration. The coolant first flows through the inverter thermal model, absorbing the heat generated by the conduction and switching losses of the inverter power devices; it then continues into the motor thermal model, where it cools the heat produced by the stator copper losses, iron losses, and other losses of the motor; finally, the coolant flows into the outlet boundary of the cooling medium. The overall liquid-cooling loop can be summarized as: coolant inlet → pump → inverter liquid-cooling module → motor liquid-cooling module → coolant outlet.

An ethylene-glycol/water solution, which exhibits good antifreeze and heat-transfer characteristics, is employed as the coolant. This type of coolant is commonly used in liquid-cooling loops for electric-drive systems, battery systems, and power-electronic apparatus. The coolant flow is driven by a pump module. Instead of using an external flow input, the pump in this work is configured to deliver a constant volumetric flow rate.

The temperature rise of the coolant after absorbing heat is related to the heat loss, the mass flow rate, and the specific heat capacity of the coolant. The fundamental thermal balance can be expressed as

$$Q = mc_p\Delta T \tag{26}$$

where Q is the heat absorbed by the coolant, m is the coolant mass flow rate, c_p is the specific heat capacity at constant pressure, and ΔT is the temperature rise between the inlet and outlet. The relationship between mass flow rate and volume-tric flow rate is

$$m = \rho q_v \tag{27}$$

where ρ is the coolant density. It follows that, for a given heat-loss level, a higher coolant flow rate removes more heat per unit time, resulting in a smaller coolant temperature rise and a better cooling performance.

3.3 Air-Cooling Model of the PMSM

In the series liquid-cooling model described above, the inverter and the PMSM share the same liquid-cooling loop, so that the coolant flows successively through the inverter and motor thermal models, and the inverter outlet temperature directly influences the motor inlet temperature. If the inverter losses are large, the coolant entering the motor is hotter, and the cooling capability of the motor is correspondingly reduced. In order to further compare the influence of different cooling configurations on the motor thermal behavior, the cooling structure is modified in this section: the inverter is still cooled by liquid, whereas the PMSM is cooled by forced air. The cooling loops of the inverter and the motor are thus made independent, eliminating the influence of the inverter coolant outlet temperatu-temperature on the motor coolant inlet temperature.

The motor air-cooling loop uses air as the cooling medium. A fan generates a certain air flow rate so that the air passes through the motor thermal model and carries away the heat produced during operation. The forced air-cooling loop of the motor can be summarized as: ambient air inlet → motor air-cooling thermal model → flow sensor → fan → ambient air outlet.

During the air-cooling process, the fan operates at a constant rotational speed and delivers a steady air flow rate to the loop. A larger air flow rate implies that more air mass passes over the motor surface per unit time, which strengthens the heat-removal capability. The fundamental thermal balance of the air-cooling process can be written as

$$Q_{air} = m_{air}c_{p,air}\Delta T_{air} \tag{28}$$

where Q_{air} is the heat absorbed by the air, m_{air} is the air mass, $c_{p,air}$ is the specific heat capacity at constant pressure, and ΔT_{air} is the air temperature rise across the motor. The relationship between the air mass flow rate and the volumetric flow rate is

$$m_{air} = \rho_{air}q_{v,air} \tag{29}$$

where ρ_{air} is the air density and $q_{v,air}$ is the air volumetric flow rate. For a given level of motor losses, a higher fan flow rate generally results in a smaller air temperature rise and a stronger heat-dissipation capability of the motor.

3.4 PMSM MTPA Control

In this section, an MTPA control module for the PMSM is developed in Modelica code. Taking the reference command as input and the d- and q-axis current references as outputs, this module computes the d-q current allocation based on the d-axis inductance, the q axis inductance, and the permanentmagnet flux linkage of the motor. In the synchronously rotating d-q frame, the electromagnetic torque equation of the PMSM is

$$T_e = \frac{3}{2}p\left[\psi_{PM}i_q + \left(L_d - L_q\right)i_d i_q\right] \tag{30}$$

where T_e is the electromagnetic torque, p is the number of pole pairs, φ_{PM} is the permanent-magnet flux linkage, i_d and i_q are the d- and q-axis currents, and L_d and $L_q$are the d- and q-axis inductances, respectively.

It can be seen that the electromagnetic torque consists of two components. The first is the permanent-magnet torque,

$$T_{PM} = \frac{3}{2}p\psi_{PM}i_q \tag{31}$$

and the second is the reluctance torque,

$$T_{rel} = \frac{3}{2}p\left(L_d - L_q\right)i_d i_q \tag{32}$$

When L_d=L_q, the reluctance torque vanishes and the electromagnetic torque is governed solely by the q-axis current; in this case, the simple i_d = 0 control strategy is

sufficient. When $L_d \neq L_q$, the motor exhibits saliency, and a properly chosen d-axis current allows the reluctance torque to be exploited, yielding a higher output torque for the same current magnitude.

The stator current magnitude satisfies

$$I_s = \sqrt{i_d^2 + i_q^2} \tag{33}$$

Therefore, the goal of MTPA control is to determine the optimal i_d and i_q, subject to a given stator-current magnitude, that maximize the electromagnetic torque.

Following the MTPA principle, for a given stator-current magnitude the optimal d-axis current can be expressed as

$$i_{d,MTPA}^{*} = \frac{\psi_{PM} - \sqrt{\psi_{PM}^2 + 8\left(L_q - L_d\right)^2 I_s^{*2}}}{4\left(L_q - L_d\right)} \tag{34}$$

For an interior PMSM, where $L_d < L_q$ is generally the case the MTPA-derived d-axis current is generally negative. A negative d-axis current weakens the d-axis flux linkage and, together with the q-axis current, produces a reluctance torque, so that a larger electromagnetic torque can be generated for the same current magnitude.

The q-axis current is then computed as

$$i_{q,MTPA}^{*} = \sqrt{T_{ref}^2 - i_{d,MTPA}^{*\,2}} \tag{35}$$

where the input signal T^2_{ref} of the MTPA module is treated, in the model code, as the stator-current magnitude reference.

In summary, when $L_d \neq L_q$, the MTPA module computes the negative d-axis current and the corresponding q-axis current according to the equations above. By exploiting the reluctance torque, the proposed module enhances the torque-producing capability of the motor, reduces the stator current magnitude required to deliver a given torque, and consequently decreases the stator copper losses and improves current utilization.

The MTPA control block diagram is similar in structure to the vector control diagram in Fig. 3, and likewise comprises a speed loop, a current loop, coordinate transformations, and the SVPWM module. The output of the speed-loop PI controller is fed into the MTPA module, which computes the optimal d- and q-axis current references and then forwards them to the d–q current controller.

To evaluate the MTPA control performance, several instantaneous and steady-state averaged variables are defined in the model. They are used to assess the current utilization, output torque, and copper-loss level under MTPA control.

The torque-to-current ratio is defined as

$$\frac{T_e}{I_s} \tag{36}$$

The stator copper loss based on the three-phase currents is calculated as

$$P_{cu} = R_s \sum_{j=1}^{3} i_j^2 \tag{37}$$

To avoid distortion of the performance evaluation caused by the start-up transient, the steady-state averages are evaluated starting from t =1s. The averaging interval is defined as

$$t_{avg} = \begin{cases} t-1, & t > 1 \\ 0, & t \leq 1 \end{cases} \tag{38}$$

Each performance index is averaged through integration. For example, the averaged electromagnetic torque is

$$\overline{T_e} = \frac{1}{t_{avg}} \int_1^t T_e(\tau) d\tau \tag{39}$$

The other averaged performance indices are obtained in the same manner and are not repeated here. These averaged indices are subsequently used to perform a quantitative comparison with the conventional vector control strategy.

3.5 Phase-Leg Loops Guide Solver Choice

We translated the top-level class SMPM-MTPA in Dymola 2023x Refresh 1 using Modelica Standard Library 4.0.0 and DymolaModels 1.6.0. The source package requests ElectrifiedPowertrains 1.5.0 and Dassault Systemes 1.8.0; The generated code also uses SDF 0.4.3. After parameter evaluation, the differential-algebraic equation system contains 4,224 scalar unknowns and equations and 28 continuous states. Dymola applies block lower triangular (BLT) sorting and identifies six linear algebraic systems with original sizes {3, 16, 16, 16, 14, 2}. Symbolic elimination and tearing reduce the corresponding numerical sizes to {0, 6, 6, 6, 4, 0}. Here, original size counts equations before symbolic elimination, while numerical size counts tearing variables retained for runtime solution; a zero value means that Dymola eliminates the block completely. The translation contains no nonlinear algebraic system.

The three size-16 mixed systems correspond to the three switching-inverter phase legs. Each system couples the upper and lower IGBT switches, reverse-blocking diodes, and antiparallel diodes. Dymola reports each block as mixed because continuous electrical variables couple to discrete switch states. Tearing retains six variables per phase leg. These blocks therefore account for 18 of the 22 post-tearing variables retained across all six linear systems.

The size-14 continuous system originates from the coupled stator, air-gap, and damper-cage equations of the PMSM. Tearing retains four derivative variables: the d-axis and q-axis stator currents and the two rotor-frame magnetizing currents. Dymola eliminates the remaining size-3 and size-2 systems before numerical integration. The generated equations map the size-2 block to the combined rotor-load inertia equation. During initialization, an additional 2 × 2 linear system enforces consistent d-q stator currents. The MTPA block evaluates its square-root expressions sequentially, so these expressions lie outside the algebraic loops.

We use DASSL, Dymola's variable-step implicit DAE solver, with a tolerance of 1 × 10−6 as the offline reference. Three repeated 20ms switching-level runs on an Intel Core i9-13900H workstation required 7.07 to 8.87 s of integration CPU time. The CPU-time-to-simulated-time ratio therefore ranged from 354 to 444, so the runs were 354 to 444 times slower than real time. Each run accepted 20,471

DOI 10.3384/ecp223347

steps and recorded 80 model time events and 1,481 state events. The minimum internal step was 1.2 × 10−12 s. The specified 1 × 10−4 s output interval sets 201 regular output-grid points; Dymola also records event points, while DASSL independently selects smaller internal steps around switching events.

The phase-leg blocks and the size-14 motor block account for all 22 post-tearing variables. The 1,481 state events and the internal steps down to 1.2 × 10−12 s expose event-dense, multi-timescale dynamics. We therefore use DASSL as the offline reference solver for this switching-level model. The benchmark covers offline execution only.

Dymola's real-time export path uses fixed-step inline integration, which embeds the integrator in the generated model code (Dassault Systèmes 2023). The phase-leg blocks provide the first reduction target because they contribute 18 of 22 post-tearing variables and drive switch-state events. An averaged inverter is a candidate reduction because it removes device-level switching events from the real-time plant model. Deployment requires two additional tests: accuracy against the DASSL reference and worst-case execution time on the target processor. These tests remain future work.

4 Simulation results

All results in this section use the Dymola 2023x translation environment specified in Section 3.5.

The parameters of the PMSM vector control model are listed in Table I.

The PMSM vector control adopts the double-loop scheme combining a speed loop and a current loop. Fig. 6 shows the simulated motor speed response, in which the motor speed converges to and remains stable at the commanded value.

The simulation results of the series liquid-cooling model are presented below. An ethylene-glycol/water solution at 25 °C is used as the coolant, with a mass of 0.1 kg and a volumetric flow rate of 10 L/min. The initial temperature of the inverter is 30 °C and that of the motor is 35 °C. The initial temperature of the motor thermal model is intentionally set higher than the coolant inlet temperature so as to emulate the operating condition after the motor has been preheated for some time.

TABLE I. SYSTEM PARAMETER VALUES OF THE PMSM VECTOR CONTROL MODEL.

Descriptions	Symbols	Values
DC-link voltage	V_{dc}	200V
Switching frequency	f_s	4 kHz
Inductance in d-axis	L_d	0.79 mH
Inductance in q-axis	L_q	0.79 mH
Number of pole pairs	p	2
Stator resistance	R_s	0.05 Ω
Moment of inertia	J_{Load}	0.29 kg m^2
Nominal torque	T_{Load}	181.4 N m

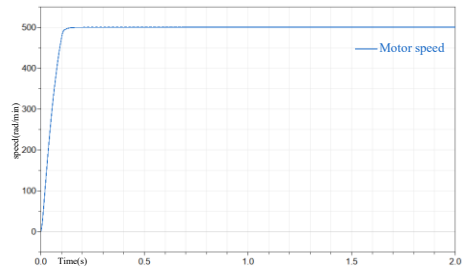


Fig. 6. Motor speed under vector control.

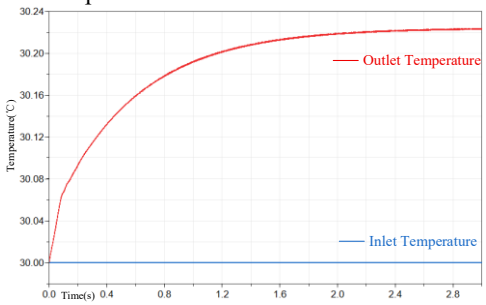


Fig. 7. Coolant temperature across the inverter.

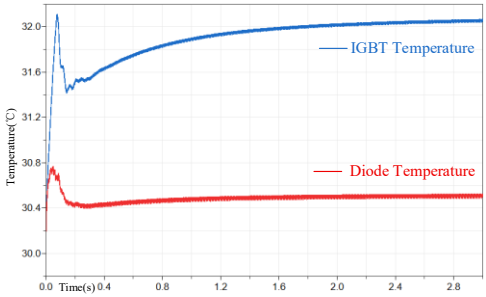


Fig. 8. Junction temperatures of the IGBT and the diode.

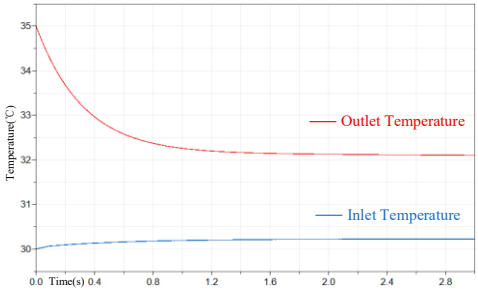


Fig. 9. Coolant temperature across the motor.

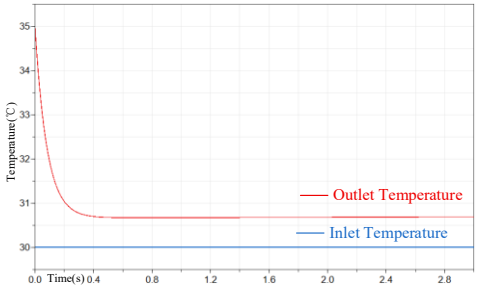


Fig. 10. Air temperature across the motor.

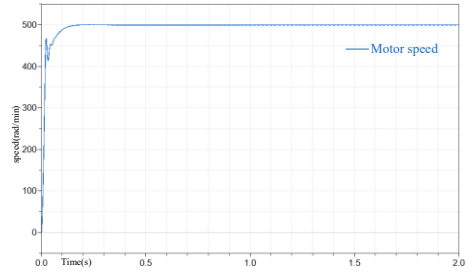
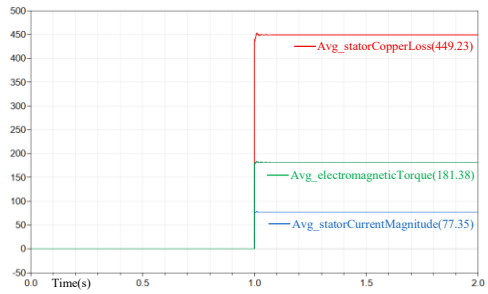
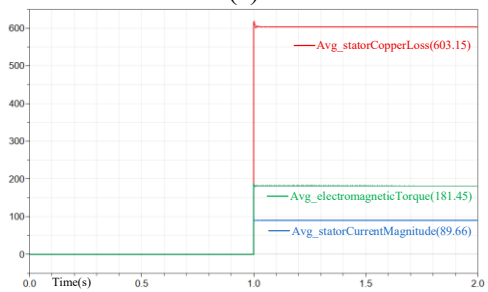


Fig. 11. Motor speed under MTPA control.



(a)



(b)

Fig. 12. Steady-state averages of motor operating performance indices: (a) MTPA control; (b) FOC ($i_d = 0$) control.

and the motor temperature decreases accordingly.

Fig. 7 shows the coolant inlet temperature together with the outlet temperature at the inverter. The outlet temperature increases over time, indicating that the coolant is absorbing the heat dissipated by the switching devices of the inverter.

Fig. 8 shows the junction temperatures of the IGBT and the diode in the inverter. Owing to the liquid cooling, the rate of temperature rise of both devices gradually decreases. Moreover, the diode temperature is lower than that of the IGBT, because during inverter operation the IGBT conduction and switching losses dominate, whereas the diode switches only during the freewheeling intervals.

Fig. 9 shows the coolant temperatures at the inlet and outlet of the motor. Because the liquid-cooling loop is configured in series, the coolant entering the motor is in fact that leaving the inverter. As can be seen from the figure, the coolant carries away the heat produced by motor operation,

For the air-cooling model, the medium is ambient air at 30 °C, with the air pressure boundary set to 101 325 Pa, the rated air volumetric flow rate set to 4 m³/min, and the rated fan speed set to 750 r/min. To clearly illustrate the heat-removal process of the forced air-cooling system, the initial temperature of the motor thermal model is set to 35 °C—higher than the inlet air temperature—again emulating a preheated motor.

Fig. 10 shows the air temperatures at the inlet and outlet of the motor under air cooling. The air flow successfully carries away the heat generated by motor operation, and the temperatures eventually reach a steady state. Moreover, when the mot-or is cooled by air alone, its cooling performance is superior to that of the motor in the series liquid-cooling configuration.

In the PMSM MTPA control model, L_d=0.79 mH and L_q=6.68 mH, while the remaining model parameters are the same as those in Table 1 and will not be listed here.

The MTPA control is designed for an interior PMSM and also adopts a double-loop structure based on a speed loop and a current loop. Fig. 11 shows the resulting motor speed response, which converges to and is well regulated around the commanded value.

The conventional i_d=0 vector control is typically used for surface-mounted PMSMs, but it can also serve as a baseline strategy for IPMSMs. Since an IPMSM exhibits saliency due to $L_d \neq L_q$, the conventional vector control cannot fully exploit the reluctance torque. To highlight this difference, the conventional vector control and the proposed MTPA control are compared under the same IPMSM parameters. Fig. 12(a) shows the steady-state averages of the key performance indices under MTPA control, where the torque-to-current ratio is about 2.34, while Fig. 12(b) shows the corresponding results under the conventional vector control, where the torque-to-current ratio is about 2.02. The comparison demonstrates that, for the same electromagnet-tic torque output, MTPA control reduces the stator-current magnitude, thereby decreasing the stator copper losses and improving current utilization.

5 Conclusions

This paper presents a Modelica-based system-level modeling framework for PMSM drive systems, integrating basic control modules, vector control, liquid/air cooling, and MTPA control models. Simulation results demonstrate the frame-work's modularity and extensibility, and confirm that MTPA control improves current utilization and reduces copper losses compared with conventional vector control, providing a unified platform for multi-physical-domain co-simulation and control-strategy verification.

References

Feng, G., C. Lai, Y. Han, and N. C. Kar, "Fast Maximum Torque per Ampere Angle Detection for Interior PMSMs Using Online Polynomial Curve Fitting," *IEEE Transactions on Power Electronics*, vol. 37, no. 2, pp. 2045–2056, 2022.

Fotias, N., et al., "A Modelica Library for Modelling of Electrified Powertrain Components," in Proc. 14th International Modelica Conference, 2021.

Fritzson, P., et al., "The OpenModelica Integrated Environment for Modeling, Simulation, and Model-Based Development," Modeling, Identification and Control, vol. 41, no. 4, pp. 241–295, 2020.

Han, Z., et al., "Improved Online Maximum-Torque-Per-Ampere Algorithm for Speed Controlled Interior Permanent Magnet Synchronous Machine," IEEE Transactions on Industrial Electronics, vol. 67, no. 5, pp. 3398–3408, 2020.

Lee, K.-W., and S.-I. Kim, "Dynamic Performance Improvement of a Current Offset Error Compensator in Current Vector-Controlled SPMSM Drives," IEEE Transactions on Industrial Electronics, vol. 66, no. 9, pp. 6727–6736, 2019.

DOI
10.3384/ecp223347

Li, Z., G. Feng, C. Lai, W. Li, and N. C. Kar, "Current Injection-Based Simultaneous Stator Winding and PM Temperature Estimation for Dual Three-Phase PMSMs," IEEE Transactions on Industry Applications, vol. 57, no. 5, pp. 4933–4945, 2021.

Lu, W., et al., "New Sensorless Vector Control System With High Load Capacity Based on Improved SMO and Improved FOO," IEEE Access, vol. 9, pp. 40716–40727, 2021.

Modelica Association, "Modelica Language Specification, Version 3.6," 2023.

Wang, X., et al., "A Critical Review on Thermal Management Technologies for Motors in Electric Cars," Applied Thermal Engineering, vol. 201, Art. no. 117758, 2022.

Yuan, X., S. Zhang, and C. Zhang, "Enhanced Robust Deadbeat Predictive Current Control for PMSM Drives," IEEE Access, vol. 7, pp. 148218–148230, 2019.

DOI
10.3384/ecp223347

Hardware-in-Loop Testing of an Induction Machine Drive System

Yang Qi[1] Hongyou Zhong[1] Yuang Yang[1]

[1]School of Automation, Northwestern Polytechnical University, China,
{qiyang@nwpu.edu.cn, zhonghonfyou@mail.nwpu.edu.cn,
yangyuang@mail.nwpu.edu.cn}

Abstract

Induction motors are highly preferred in electric vehicle (EV) applications due to robustness. This paper presents a comprehensive study on the vector control of an induction motor for EV applications. To achieve precise closed-loop speed and torque regulation, the mechanical angular speed of the rotor is obtained through a quadrature encoder. In addition to this, a magnetic flux estimator is implemented to provide the real-time flux information. The mathematical model and control for a motor drive system are established. To validate the effectiveness of the typical vector control, a real-time Hardware-in-the-Loop (HIL) test is conducted, where the power circuit and control strategies are both simulated by the PLECS RT Box. The HIL results verify the stability, dynamic response, and overall effectiveness of the proposed vector control scheme for EV drivetrains.

Keywords: Induction Motor, FOC, HIL Test.

1 Introduction

Driven by the global imperative to reduce greenhouse gas emissions, the trend of transportation electrification has rapidly accelerated. Within the electric vehicle (EV) drivetrain, the induction machine (IM) offers a promising solution apart from the permanent magnet synchronous machine (PMSM) [1]. While PMSMs dominate due to high power density, induction machines offer distinct, critical benefits, such as good power density, excellent high-speed cruising capabilities, and are free from rare-earth permanent magnets [2].

However, enabling an IM drive system to satisfy the stringent torque and speed requirements for complicated EV applications necessitates high-efficiency validation. Developing and deploying these algorithms directly onto physical high-power prototypes is inherently risky, time-consuming, and expensive. As an alternative, Hardware-in-the-Loop (HIL) testing has emerged as an approach in modern drive system development [3]. HIL simulations interface a physical controller with a real-time, high-fidelity virtual emulation of the power electronics and induction motor. This paradigm bridges the critical gap between offline simulation and full-scale dynamo-meter testing. It enables engineers to safely and quickly verify control strategies, validate embedded code, and test extreme fault conditions without risking catastrophic damage to physical hardware [4].

From the perspective of control, to achieve the precise electric torque or speed regulation for EV, the well-known Field-oriented Control (FOC), also known as vector control, serves as the industry-standard methodology [5]. Vector control functionally translates the complex, highly coupled AC dynamics of the induction motor into an independently controlled, DC-like machine. Specifically, accurate mechanical rotor angular speed must be acquired to close the outer speed control loop. This is practically achieved utilizing a high-resolution quadrature encoder mounted directly on the motor shaft [6], which generates pulse-train signals that the controller translates into precise mechanical speed and position. As the internal rotor magnetic flux cannot be directly measured, a robust magnetic flux estimator must be deployed. Utilizing the measured stator currents, phase voltages, and encoder-derived rotor speed, a digital flux observer continuously computes the magnitude and synchronous angle of the rotor flux linkage [7]. In addition, inner-loop AC current tracking is realized through the Proportional-Integral (PI) controllers under a rotating reference frame. The resulting voltage commands are then transformed back to the stationary AC frame. The Space Vector Pulse Width Modulation (SVPWM) is employed to synthesize the gating signals [8].

Despite the maturity of FOC, translating these complex mathematical frameworks into stable, hardware-ready HIL tests remains a significant challenge. Therefore, the primary contribution of this paper is the comprehensive design, discrete-time implementation, and rigorous real-time validation of a FoC system for EV induction machine drives. This work actively integrates quadrature encoder feedback and a dynamic rotor flux estimator within a digital control framework. To validate the proposed system, a high-fidelity HIL testing platform based on the PLECS RT Box is utilized to emulate the physical motor and inverter dynamics. The HIL testing results effectively validate the good transient dynamics as well as the precision of the electric torque tracking performance.

2 Modeling of an Induction Machine

To design and implement a high-performance vector control strategy, the mathematical model of the induction machine is required. The equivalent circuit of an induction

machine under the *dq* synchronous frame can typically be expressed as

$$\begin{cases} v_{ds} = R_s i_{ds} + p\lambda_{ds} - \omega\lambda_{qs} \\ v_{qs} = R_s i_{qs} + p\lambda_{qs} + \omega\lambda_{ds} \\ v_{dr} = R_r i_{dr} + p\lambda_{dr} - (\omega - \omega_r)\lambda_{qr} \\ v_{qr} = R_r i_{qr} + p\lambda_{qr} + (\omega - \omega_r)\lambda_{dr} \end{cases} \quad (1)$$

Besides, the flux linkages are linearly related to the currents through the machine inductances, as shown in the following expression.

$$\begin{cases} \lambda_{ds} = L_{ls} i_{ds} + L_m (i_{ds} + i_{dr}) \\ \lambda_{qs} = L_{ls} i_{qs} + L_m (i_{qs} + i_{qr}) \\ \lambda_{dr} = L_{lr} i_{dr} + L_m (i_{ds} + i_{dr}) \\ \lambda_{qr} = L_{lr} i_{qr} + L_m (i_{qs} + i_{qr}) \end{cases} \quad (2)$$

Fig. 1 demonstrates the *dq*-axis equivalent model of an induction machine. The electromagnetic torque produced by the induction motor can be calculated using the cross product of the stator flux linkages and stator currents. To align with rotor-flux-oriented control, the torque equation is most usefully expressed in terms of the rotor fluxes and stator currents, as given by

$$T_e = \frac{3PL_m}{2L_r}(i_{qs}\lambda_{dr} - i_{ds}\lambda_{qr}) \quad (3)$$

where P is the number of pole pairs. If the *d*-axis of the reference frame is perfectly aligned with the rotor flux vector, all rotor flux is concentrated on the *d*-axis, and the *q*-axis rotor flux is zero. In this case, the electric torque equation can be further simplified as

$$T_e = \frac{3PL_m}{2L_r} i_{qs}\lambda_r \quad (4)$$

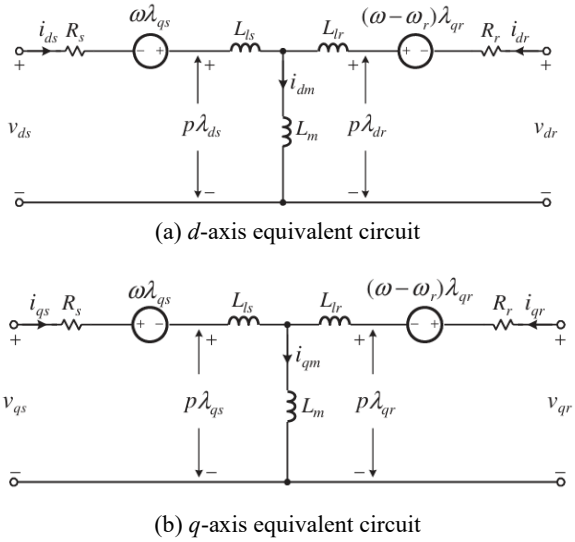
The above equation demonstrates that electromagnetic torque could be controlled linearly by the *q*-axis stator current, which is quite similar to excited DC motor control. Moreover, the mechanical motion of the motor is governed by the interaction between the generated electromagnetic torque and the mechanical load. The rotor dynamics are modeled as

$$T_e - T_l = J\frac{d\omega_m}{dt} + D\omega_m \quad (5)$$

where T_l is the load torque applied to the motor shaft; J is the inertia of the motor rotor; D is the friction coefficient; ω_m is the mechanical angular speed of the rotor. The mechanical speed is directly related to the rotor electrical speed by the number of pole pairs, as given by

$$\omega_r = P \cdot \omega_m \quad (6)$$

The above equation forms the mathematical model of an induction motor system.



(a) *d*-axis equivalent circuit

(b) *q*-axis equivalent circuit

Fig. 1. Induction motor *dq*-axis model.

3 FOC of Induction Machine

The primary objective of FOC is to achieve decoupled control of the electromagnetic torque and rotor magnetic flux in an induction machine. By employing the dynamic coordinate transformations, FOC allows the inherently coupled and non-linear AC machine to be controlled with the simple and high dynamic performance of an excited DC motor.

Fig. 2 shows the general control block diagram of FOC. The torque reference is generated by an outer-loop speed controller based on the reference speed and the detected or estimated rotor speed. The mechanical angular speed of the rotor is obtained from a Quadrature Encoder Interface.

The rotor flux reference is a function of reference speed. When the motor operates at or below its rated speed, the rotor flux reference is normally kept at its rated value. With the rated speed exceeded, the rotor flux reference should be weakened accordingly, such that the stator voltage and the output power of the motor would not exceed their ratings. The rotor flux and torque references are then sent to the flux/torque controller, and compared with the calculated rotor flux and torque for a closed-loop control. Based on the IM mathematical model, the stator flux vector can be expressed as:

$$\lambda_s = \int (v_s - R_s i_s) dt \quad (7)$$

The rotor flux vector can be found from the flux linkage equation

$$\lambda_r = L_r \frac{\lambda_s - L_s i_s}{L_m} + L_m i_s = \frac{L_r(\lambda_s - \sigma L_s i_s)}{L_m} \quad (8)$$

where σ is the total leakage factor, as given by

$$\sigma = 1 - L_m^2 / \left(L_s L_r\right) \quad (9)$$

DOI
10.3384/ecp223357

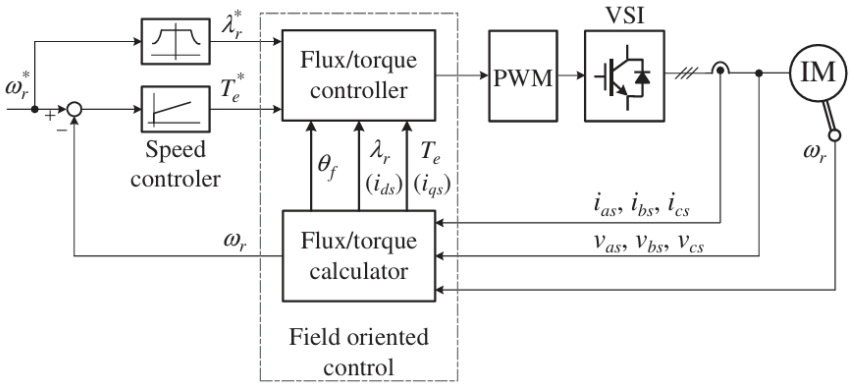


Fig. 2. General block diagram of rotor flux FOC [9].

The rotor flux can be decomposed into the *d*-axis and *q*-axis components, as given by

$$\lambda_{dr} = \frac{L_r}{L_m}(\lambda_{ds} - \sigma L_s i_{ds}) \tag{10}$$

$$\lambda_{qr} = \frac{L_r}{L_m}(\lambda_{qs} - \sigma L_s i_{qs}) \tag{11}$$

from which the magnitude and angle of the rotor flux can be calculated as

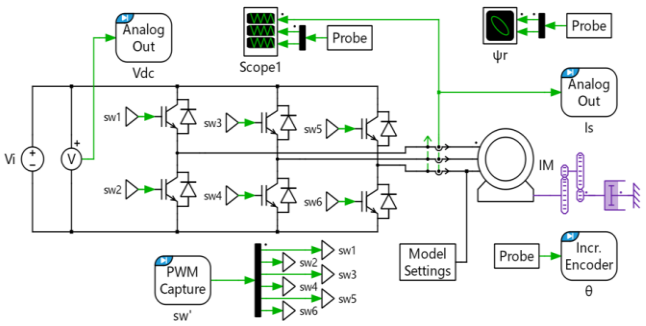
$$\begin{cases} \lambda_r = \sqrt{\lambda_{dr}^2 + \lambda_{qr}^2} \\ \theta_f = \tan^{-1}\dfrac{\lambda_{qr}}{\lambda_{dr}} \end{cases} \tag{12}$$

It should be mentioned that the rotor flux magnitude and angle can be obtained based on the measured stator current and voltage, as long as the motor parameters, such as L_s, L_r, L_m, and R_s are known.

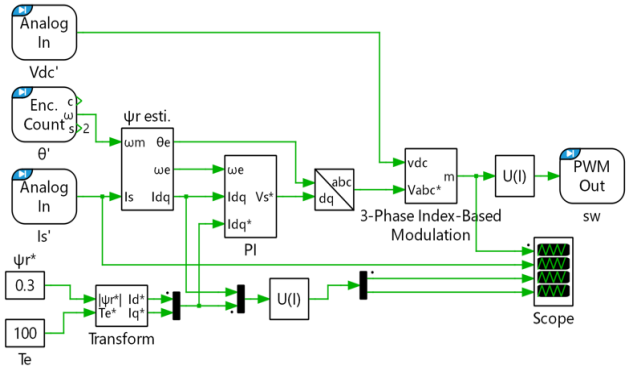
Once the rotor flux and torque references are confirmed, the *dq*-frame current references are generated. To realize fast current tracking, two inner PI current controllers regulate the measured stator currents to track the *dq*-frame current reference within the rotating frame. The outputs of these controllers are output voltage commands, which are subsequently transformed back to the stationary *abc* three-phase frame using the computed flux angle. Finally, Space Vector Pulse Width Modulation (SVPWM) is employed to translate these voltage references into switching signals for the inverter, thereby precisely forcing the AC motor currents to follow the desired torque and flux trajectories.

4 Induction Machine HIL Test

To validate the effectiveness of induction machine FOC, hardware-in-loop testing based on a PLECS RT Box was carried out. The HIL test is based on a demon model. The model consists of both the power circuit and control strategy parts. They are respectively shown in Fig. 3. The discrete simulation time step is selected as 5μs in order to make a good compromise between the simulation accuracy and the computational effort. The measured electrical variables, such as the stator current and the DC-link voltage, are transferred from the power circuit part to the FOC part through Analog OUT and Analog IN channels. The discrete signals, such as PWM sw1-sw6, are transferred from the FOC part to the power circuit part through Digital OUT and Digital IN channels. Key system parameters for the HIL simulation are listed in Table I.



(a) Power Circuit Part.



(b) FOC part.

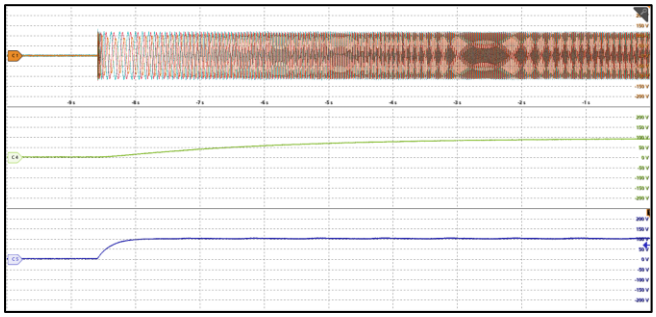
Fig. 3. Plant and controller model for HIL test.

TABLE I

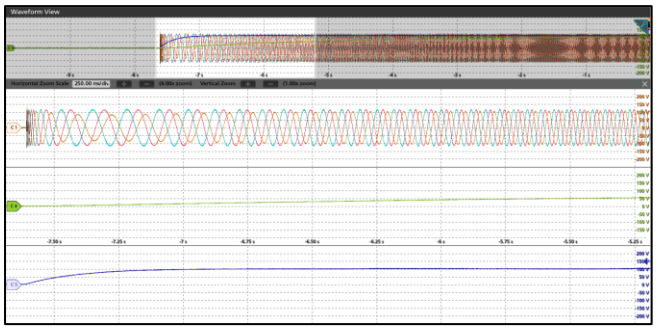
TABLE I SYSTEM AND CONTROL PARAMETERS

Parameters	Values
R_s	0.1062
R_r	0.0764
L_{ls}	0.000568
L_{lr}	0.000568
L_m	0.01547
L_r	0.016
L_s	0.016
J	2.8
D	0.1
P	2
f_s	10 kHz

Fig. 4 shows the HIL testing results during the IM startup process. All variables are measured through the Analog out channels and an oscilloscope. It can be observed that the IM rotation speed gradually increases from 0 to 100 rad/s in about 9 *s*, while the electric torque reaches the expected value of 100 N·m. The stator current waveforms also agree with the theoretical analysis of FOC.



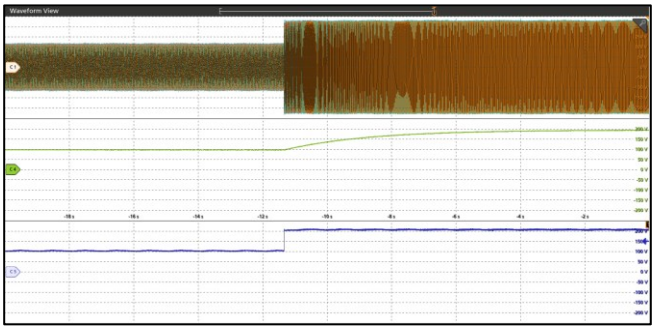
(a). IM startup process. C1: staor current waveform (1A per V), C4: rotational speed waveform (1 rad/s per V), C5: Electric torque wave from (1 N·m per V).



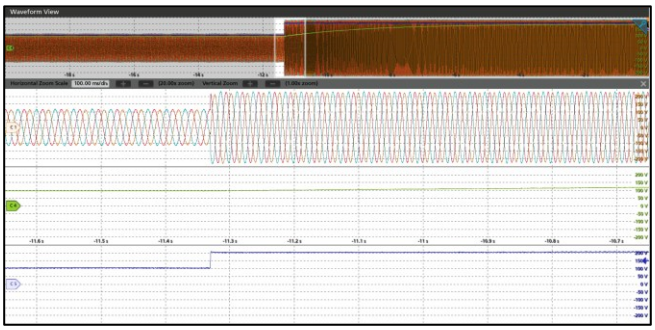
(b). Zoom view of IM startup process. C1: staor current waveform (1A per V), C4: rotational speed waveform (1 rad/s per V), C5: Electric torque wave from (1 N·m per V).

Fig. 4. IM startup waveforms.

Fig. 5 shows the HIL testing result when the IM reference torque suddenly changes from 100 N·m to 200 N·m. It is clear the measured IM electric can quickly and accurately track the reference changes, as realized by the FOC.



(a). IM startup process. C1: staor current waveform (1A per V), C4: rotational speed waveform (1 rad/s per V), C5: Electric torque wave from (1 N·m per V).



(b). Zoom view of IM startup process. C1: staor current waveform (1A per V), C4: rotational speed waveform (1 rad/s per V), C5: Electric torque wave from (1 N·m per V).

Fig. 5. IM torque reference step change.

Conclusions

This paper derived the equivalent mathematical model of an induction motor. Based on the well-known FOC, the flux estimation and torque tracking for an IM are realized. To bridge the gap between theoretical design and physical prototyping, the overall control framework was validated using a Hardware-in-the-Loop (HIL) simulation platform based on the PLECS RT Box. The HIL experimental results confirmed the accuracy of the flux estimator and validated the torque reference tracking capabilities of the controller.

References

[1] Ehsani, Mehrdad, et al. *Modern electric, hybrid electric, and fuel cell vehicles*. CRC press, 2018.

[2] Z Zhu, Z. Q., & Howe, D. (2007). Electrical machines and drives for electric, hybrid, and fuel cell vehicles. *Proceedings of the IEEE*, *95*(4), 746-765.

[3] Mohammed, O. A., Abed, N. Y., & Ganu, S. C. (2007, June). Real-time simulations of electrical machine drives with hardware-in-the-loop. In *2007 IEEE Power Engineering Society General Meeting* (pp. 1-6). IEEE.

DOI
10.3384/ecp223357

[4] Mohammadi, F., Bok, R., & Hajian, M. (2022, February). Real-time controller-hardware-in-the-loop testing of power electronics converters. In *2022 13th Power Electronics, Drive Systems, and Technologies Conference (PEDSTC)* (pp. 398-402). IEEE.

[5] Trzynadlowski, A. (1998). Modern power electronics. *IEEE Power Engineering Review*, *18*(7), 31-31.

[6] Zheng, D., Zhang, S., Wang, S., Hu, C., & Zhao, X. (2014). A capacitive rotary encoder based on quadrature modulation and demodulation. *IEEE Transactions on Instrumentation and Measurement*, *64*(1), 143-153.

[7] Yin, Z., Li, G., Zhang, Y., Liu, J., Sun, X., & Zhong, Y. (2016). A speed and flux observer of induction motor based on extended Kalman filter and Markov chain. *IEEE Transactions on Power Electronics*, *32*(9), 7096-7117.

[8] Li, S., Xiao, H., & Chen, H. (2005, September). The research of SVPWM control technique of double three-phase induction machine. In *2005 International Conference on Electrical Machines and Systems* (Vol. 1, pp. 109-114). IEEE.

[9] Wu, B., & Narimani, M. (2017). High-power converters and AC drives. John Wiley & Sons.

DOI
10.3384/ecp223357

Dynamic Load Calculation and Analysis of Multi-Zone RC Thermal Network Based on Modelica

Zhong Xuyang[1] Zhu Qiuyuan[1] Zhou Aimin[1] Shi Hongqi[1] Xu Xinhua[2]

[1]Wuhan Second Ship Design and Research Institute, China,

[2]Department of building environment and energy engineering, Huazhong University of Science and Technology, China, {bexhu@hust.edu.cn}

Abstract

Traditional air conditioning load calculation methods suffer from overestimated results in steady-state computation, excessive computational burden in unsteady-state calculation, and low efficiency in numerical analysis. Such defects make it difficult to quickly and accurately satisfy the control requirements of multi-zone air conditioning loads under complex dynamic boundaries. To solve these problems, this paper proposes applying the Modelica-based RC thermal network heat transfer model to air conditioning load calculation, and carries out dynamic load simulation and verification with the multi-zone structure in the control cabin of an underwater vehicle as the research object. The results show that the proposed method can realize multi-zone dynamic load solution under dynamic boundaries. The dynamic response accuracy is verified against measured loads under surface and underwater navigation conditions. The average relative error rates of the total load are only 3.17% and 2.90%, respectively, and the maximum error rate is less than 10%. The model has high accuracy and provides an effective approach for multi-zone dynamic load calculation.

Keywords: Modelica; RC thermal network model; air conditioning load; dynamic characteristics; underwater vehicle

1 Introduction

Air conditioning load calculation is the foundation of air conditioning system design (Gao 2005). Fast and accurate load calculation can efficiently regulate the thermal-humidity environment inside underwater vehicles, ensure safe operation of critical electronic equipment (Dong et al. 2012), and improve energy utilization efficiency(Huang 2019).

Traditional air conditioning load methods include steady-state calculation (State Bureau of Quality and Technical Supervision 1992), unsteady-state calculation (Yang 2010; Hu and Yao 2002; He and Wang 2000; He et al. 2002) and numerical analysis(Li et al. 2013). Steady-state calculation takes heat gain as the actual load and ignores heat storage, leading to overestimated results. Unsteady-state calculation needs to solve complex frequency-domain coefficients with unfixed orders in heat transfer functions, resulting in insufficient accuracy or excessive computation. Numerical analysis software such as Fluent requires massive grid generation for complex zones, leading to low solution efficiency. All these methods can hardly meet the demand for fast and accurate dynamic load solution.

The RC thermal network model, based on the thermoelectric analogy, simplifies the description of dynamic heat transfer in systems (Paschkis and Baker 1942). It abstracts key components in a thermal system into two elements: thermal resistance (impedance) and thermal capacitance (heat storage capacity). By constructing a thermal network containing these two elements, dynamic loads can be calculated according to boundary conditions. When used for air conditioning load calculation, the RC thermal network model only needs to build a topological network of the two elements based on the physical structure of the object, enabling fast load solution. The model features easy construction, convenient solution and high accuracy.

The RC thermal network model is often integrated into the energy consumption software TRNSYS (Fang 2024). Limited by professional scope, the RC model in TRNSYS has poor reusability and requires rebuilding basic components when the RC structure changes. Moreover, its external interface is complex, making cross-domain coupling difficult. This paper adopts Modelica (Chen 2013, Ding 2006), an object-oriented multi-domain unified modeling language, which supports object-oriented, acausal and declarative physical modeling. It can quickly build models based on the basic mathematical equations of the RC model while preserving the physical topology of the object. Existing literature has applied Modelica to build RC thermal network models for building envelopes. The established low-order RC models (Lauster et al. 2013) and general RC models (Felgner et al. 2006) can accurately calculate building dynamic thermal loads and provide basic thermal network components and thermodynamic interfaces, making the thermal network models easy to reuse, maintain and support cross-domain coupling. However, there is a lack of research on dynamic load calculation for underwater vehicles under complex working conditions.

This paper constructs a multi-zone RC thermal network model for the control cabin of an underwater vehicle based on Modelica, calculates the load variation of each zone under dynamic environmental boundaries, and verifies the dynamic characteristics of the model. Then, the model is compared with measured data under two working

conditions to validate its dynamic response accuracy and reliability.

2 Fundamentals Of Rc Thermal Network Model

2.1 RC Mathematical Model

The mathematical model of the RC thermal network is based on the physical principles of heat transfer and storage, serving as the basis for dynamic load calculation. Based on the thermoelectric analogy, the heat flow balance equation at the temperature node in the RC thermal network is established according to the effects of thermal capacitance and thermal resistance on heat flow:

$$C\frac{\mathrm{d}T}{\mathrm{d}\tau}=\frac{\Delta T_{\mathrm{in}}}{R_{\mathrm{in}}}-\frac{\Delta T_{\mathrm{out}}}{R_{\mathrm{out}}} \tag{1}$$

where $C\frac{\mathrm{d}T}{\mathrm{d}\tau}$ is the heat flow rate at the thermal capacitance; $\Delta T_{\mathrm{in}}/R_{\mathrm{in}}$ is the heat flow into the node through thermal resistance; $\Delta T_{\mathrm{out}}/R_{\mathrm{out}}$ is the heat flow out of the node through thermal resistance.

Equation (1) shows that the heat flow balance at a node includes heat storage by thermal capacitance and heat flow resistance by thermal resistance. The heat flow balance states that the heat flow entering a node equals the sum of the heat flow leaving the node and the heat stored in the thermal capacitance.

For a complex RC network, the heat flow balance equation at a specific node is obtained by subtracting the total outgoing heat flow from the total incoming heat flow of adjacent nodes:

$$C_p\frac{\mathrm{d}T_p}{\mathrm{d}\tau}=\sum_i\frac{\Delta T_{i\to p}}{R_{i\to p}}-\sum_o\frac{\Delta T_{p\to o}}{R_{p\to o}} \tag{2}$$
$$i,o=1,2,3,\ldots$$

where subscript i denotes nodes flowing into the target node; o denotes nodes flowing out of the target node; p denotes the target node.

2.2 Model Simplification Assumptions

When constructing the RC thermal network model, the number of thermal capacitances, thermal resistances and positions of temperature nodes are determined by the number of envelope layers. Based on the thermoelectric analogy, this paper constructs a **3R2C** model (Xu and Wang 2007) for three-layer envelopes and a **2R1C** model (Hassid 1985) for two-layer envelopes. Both models can accurately describe the heat transfer characteristics of envelopes under low-frequency (≥ 24 h) environmental boundaries. The 3R2C thermal network model of a three-layer envelope is shown in Figure. 1.

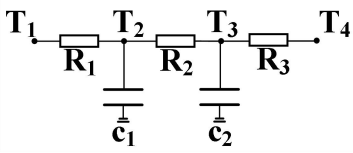


Figure 1. 3R2C model

The heat flow balance equations for temperature nodes T_2 and T_3 are as follows:

$$C_1\frac{\mathrm{d}T_2}{\mathrm{d}\tau}=\frac{T_1-T_2}{R_1}-\frac{T_2-T_3}{R_2} \tag{3}$$

$$C_2\frac{\mathrm{d}T_2}{\mathrm{d}\tau}=\frac{T_2-T_3}{R_2}-\frac{T_3-T_4}{R_3} \tag{4}$$

The 2R1C model of a two-layer envelope is shown in Figure. 2.

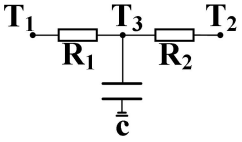


Figure 2. 2R1C model

The heat flow balance equation for temperature node T_3 is:

$$C\frac{\mathrm{d}T_3}{\mathrm{d}\tau}=\frac{T_1-T_3}{R_1}-\frac{T_3-T_2}{R_2} \tag{5}$$

Internal heat sources are described by a 2R2C model (Shi et al. 2014), as shown in Figure. 3.

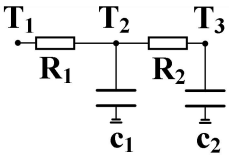


Figure 3. 2R2C model

The convective heat transfer model of the wall is described by convective thermal resistance, calculated as:

$$R=\frac{1}{\lambda A} \tag{6}$$

where λ is the convective heat transfer coefficient, W/(m²·K); A is the area, m².

The air inside the zone is described as air thermal capacitance C_a.

2.3 Parameter Calculation

Before solving the load for the established RC thermal network model, parameters R and C must be determined. For the envelopes studied in this paper, with uniformly distributed material layers, the lumped parameter method is used to directly determine the thermal capacitance and thermal resistance of the thermal network model based on

DOI
10.3384/ecp223363

Table 1. Physical parameters of the surrounding wall in zone 1

Region	*Material*	*Thermal conductivity W/(m·K)*	*Specific heat capacity J/(kg·K)*	*Density kg/m³*	*Specific heat capac J/(kg·K)*
top wall	steel	48.000	480	7850.0	480
	thermal insulation layer	0.047	2400	124.9	2400
side wall	steel	48.000	480	7850.0	480
	material layer	48.000	2400	124.9	2400
	thermal insulation layer	0.047	2400	124.9	2400

the geometric dimensions and physical properties of each material layer (Wang and Xia 2004).

The thermal resistance of a material layer equals the ratio of its thickness δ to thermal conductivity k:

$$R = \frac{\delta}{k} \tag{7}$$

Considering the uniform material distribution of two-layer and three-layer envelopes, this paper assumes equal thermal resistance for each layer, with the sum equal to the total thermal resistance. The thermal capacitance of a two-layer envelope is the sum of the thermal capacitances of the two material layers. For the three-layer envelope shown in Figure. 1, thermal capacitances C_1 and C_2 each account for half of the total thermal capacitance.

2.4 Multi-Zone RC Thermal Network Model of the Control Cabin

An underwater vehicle usually consists of a bow, control cabin, power cabin, payload cabin, etc. The control cabin is equipped with consoles, data communication devices and displays. When the vehicle navigates on the surface, the external disturbance of the upper envelope of the control cabin is air or seawater; when navigating underwater, it changes to seawater. The internal zones of the control cabin are disturbed by air from the bow and power cabins, and the lower envelope is always in contact with seawater. In view of these complex boundary conditions, the control cabin is divided into zones to build the RC thermal network model for load calculation. The geometric structure of the control cabin is shown in Figure. 4.

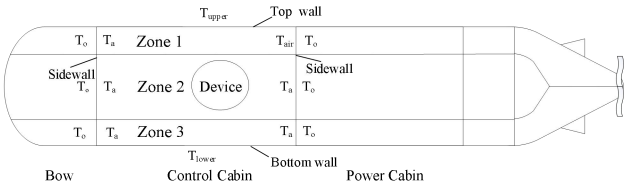


Figure 4. Control room geometric layout diagram

Based on the structural diagram and material composition of the control cabin envelope, the RC thermal network physical model of Zone 1 is shown in Figure. 5. The top wall is a two-layer envelope, simplified as a 2R1C model; the side walls are three-layer envelopes, simplified as a 3R2C model. The internal heat source is a 2R2C model. Thermal resistances Ru and Ro represent the external convective thermal resistances of the top and side walls, respectively; Ra is the internal convective thermal resistance of the zone. Ca is the air thermal capacitance inside the zone. Zone 1 is a **12R5C thermal network model**.

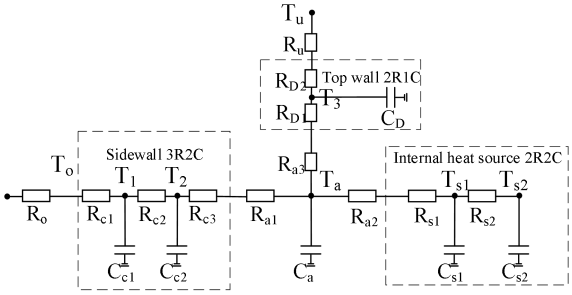


Figure 5. Physical model of zone 1 RC heating network

Following the same method, the simplified thermal network physical models of Zone 2 and Zone 3 are constructed as shown in Figures. 6 and 7.

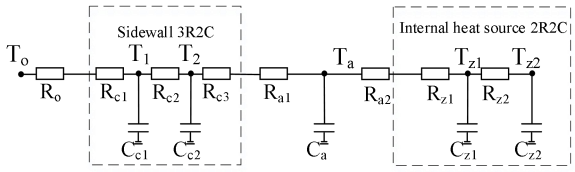


Figure 6. Physical model of zone 2 RC heating network

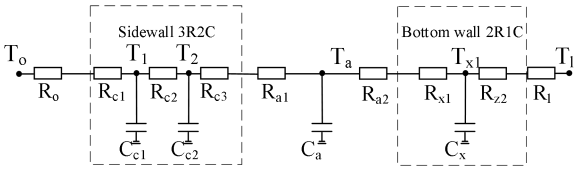


Figure 7. Physical model of zone 3 RC heating network

The physical parameters of each zone's envelope are determined based on actual test feedback of the studied underwater vehicle. Partial physical parameters of Zone 1 are listed in Table 1.

The physical parameters of Zone 2 and Zone 3 are also obtained from actual test feedback, and the corresponding thermal resistance and thermal capacitance are calculated accordingly.

3 Implementation Of Rc Thermal Network Model In Modelica

Based on the physical model and basic components of the thermal network, the RC thermal network model is built using Modelica.

3.1 Basic Components of the Thermal Network Model

The basic components of the RC thermal network model are provided by the Thermal.HeatTransfer standard library in Modelica, including HeatCapacitor and ThermalResistor, as shown in Figure. 8.



(a) heat capacity (b) heat resistance

Figure 8. Heat capacity, heat resistance component model

The constitutive equations of thermal capacitance and thermal resistance are built in Modelica to describe their behavior in heat transfer:

$$C * \mathrm{der}(T) = port.Q_flow \quad (8)$$

$$dT = R * Q_flow \quad (9)$$

where C is thermal capacitance; T is temperature; der(T) is the temperature rate; R is thermal resistance; Q_flow is heat flow rate.

Temperature nodes are described using "connector" classes in Modelica to implement the heat flow balance equation. Nodes are connected through interfaces, and equations of equal temperature and zero total heat flow are automatically generated at connections based on the generalized Kirchhoff's law and potential-flow theory. The thermodynamic interface between thermal capacitance and thermal resistance is shown in Figure. 9.

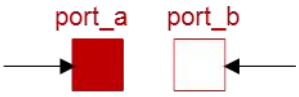


Figure 9. Thermodynamic interface

3.2 Modelica-Based 2R1C Thermal Network Model

Taking the two-layer envelope as an example, the thermal network model is built in Modelica as shown in Figure. 10.

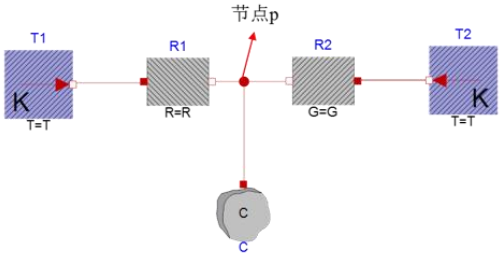


Figure 10. 2R1C heat net Modelica model

The Modelica model retains the physical topology of the 2R1C thermal network model and can be quickly constructed by replacing thermal capacitance and thermal resistance in the physical model with corresponding RC components. After connecting the components, the heat flow balance equation at node p is automatically generated in Modelica as:

$$connect(R1.port_b, C.port) \quad (10)$$

$$connect(R2.port_b, C.port) \quad (11)$$

According to Kirchhoff's law, the connection equation can be expressed as:

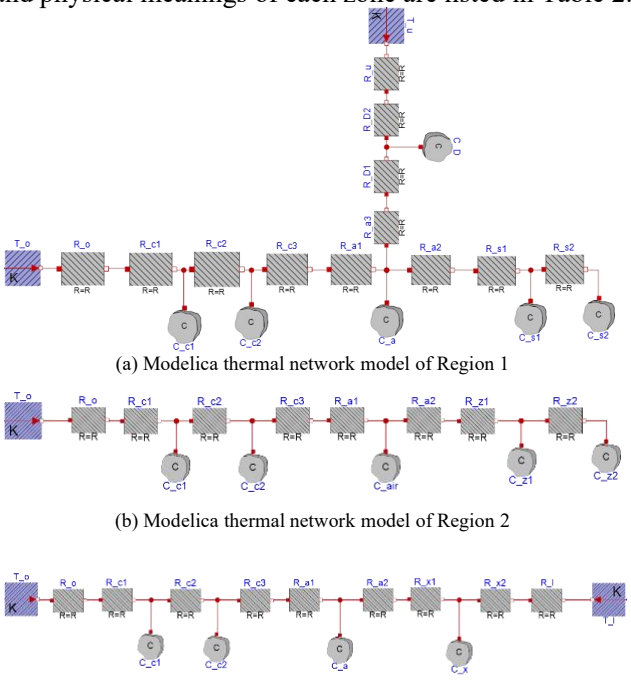
$$\begin{cases} R1.\mathrm{port_b}.T = C.\mathrm{port}.T \\ R1.\mathrm{port_b}.Q_flow + C.\mathrm{port}.Q_flow = 0 \end{cases} \quad (12)$$

where $R1$.port_b.T is the temperature at port b of thermal resistance $R1$; C.port.T is the temperature at the port of thermal capacitance; $R1$.port_b.Q_flow is the heat flow rate at port b of thermal resistance $R1$; C.port.Q_flow is the heat flow rate at the port of thermal capacitance. Equation (10) indicates that the potential variable "temperature" is equal and the flow variable "heat flow" sums to zero at the node.

The boundary conditions of the thermal network model are set by temperature boundary input blocks on both sides. The set boundary conditions are input into the blocks and connected thermally to complete boundary input.

3.3 Multi-Zone Modelica Thermal Network Model

According to the multi-zone RC thermal network physical models established in Section 2.4 and the modeling method in Section 3.2, the Modelica models of the three zones are obtained as shown in Figure. 11. The parameters and physical meanings of each zone are listed in Table 2.



(a) Modelica thermal network model of Region 1

(b) Modelica thermal network model of Region 2

(c) Modelica thermal network model of Region 3

Figure 11. Control Cabin Multi-Zone Modelica Thermal Network Model

DOI 10.3384/ecp223363

Table 2. Parameters and physical significance of regional RC model

Region 1		*Region 2*		*Region 3*	
parameter	physical significance	parameter	physical significance	parameter	physical significance
R_u	upside walls resistance	R_o	side walls resistance	R_o	side walls resistance
$R_{D1}=R_{D2}$	upside walls equivalent thermal resistance	$R_{c1}=R_{c2}=R_{c3}$	side walls equivalent thermal resistance	R_l	bottom wall thermal resistance
$R_{c1}=R_{c2}=R_{c3}$	side walls equivalent thermal resistance	R_{a1}	inner side walls thermal resistance	$R_{c1}=R_{c2}=R_{c3}$	side walls equivalent thermal resistance
R_{a1}	inner side walls resistance	R_{a2}	equipment surface thermal resistance	$R_{x1}=R_{x2}$	equipment equivalent thermal resistance
R_{a2}	equipment surface thermal resistance	R_{z1}	equipment thermal resistance	R_{a1}	inner side walls resistance
R_{a3}	inner upside walls thermal resistance	R_{z2}	equipment thermal resistance	R_{a2}	inner bottom wall thermal resistance
R_{s1}	equipment thermal resistance	$C_{c1}=C_{c2}$	side walls equivalent thermal capacitance	$C_{c1}=C_{c2}$	side walls equivalent thermal capacitance
R_{s2}	equipment thermal resistance	C_a	air thermal capacitance	C_x	bottom wal equivalent thermal capacitance
C_D	upside walls thermal capacitance	$C_{z1}=C_{z2}$	equipment equivalent thermal capacitance	L_x	region 3 length
$C_{s1}=C_{s2}$	equipment equivalent thermal capacitance	L_z	equipment length	A_x	area of the bottom wall
$C_{c1}=C_{c2}$	side walls equivalent thermal capacitance				
L_s	equipment length				
A	area of the top wall				

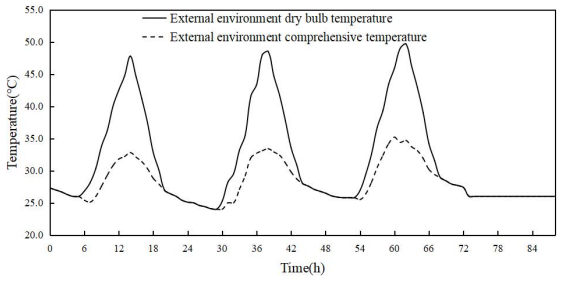
Thermal capacitance and thermal resistance are calculated using the lumped parameter method based on the geometric dimensions and physical properties of the control cabin envelope material layers. The calculation results are input as parameters into the model to calculate the load according to boundary conditions. The model can be modified by re-entering thermal capacitance and resistance values without rebuilding equations and components.

The modeling process shows that Modelica can quickly build a thermal network model that retains the physical topology of the RC network. Meanwhile, the basic components can be reused in different thermal networks by only modifying parameter values, making the model easy to reuse and maintain. Since thermal capacitance can store heat, the RC thermal network model built in Modelica has good dynamic characteristics.

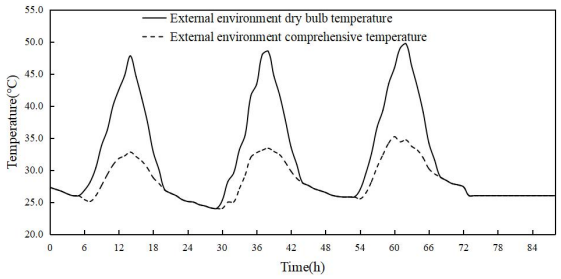
4 Dynamic Load Calculation And Simulation Analysis

4.1 Setting Complex Dynamic Boundary Conditions

To verify the effectiveness of the RC thermal network model in dynamic load response, dynamic external disturbance boundaries are set for the underwater vehicle navigation. The variation of external disturbance boundaries is shown in Figure. 12.



(a) Upper boundary conditions of Zone 1 enclosure



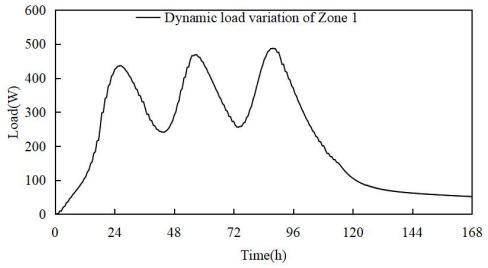
(b) Side and bottom wall boundary conditions

Figure 12. Complex dynamic boundary conditions

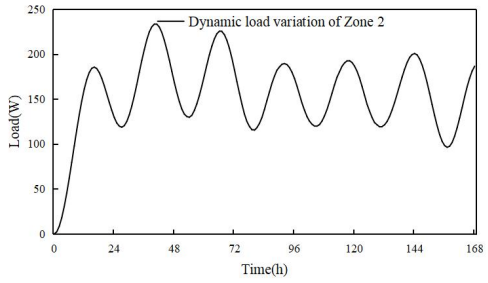
The vehicle navigates on the surface for the first 3 days and submerges after the 3rd day, so the upper envelope boundary of Zone 1 changes from air to seawater. During navigation, the air temperature inside the side wall compartments of the control cabin varies sinusoidally within 23±3℃, and the seawater temperature at the bottom wall is 26℃.

4.2 Multi-Zone Dynamic Load Simulation Analysis

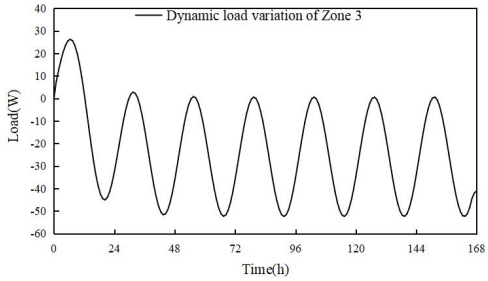
Based on the established RC thermal network model and input external disturbance boundaries, the simulation results of each zone's load and total load variation are shown in Figure. 13.



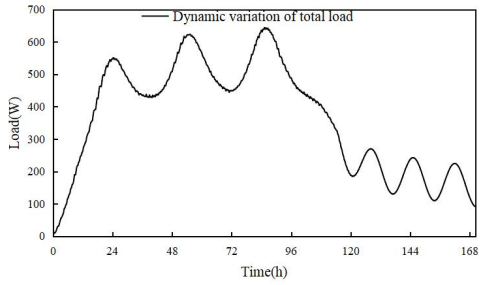
(a) Dynamic load variation of Zone 1



(b) Dynamic load variation of Zone 2



(c) Dynamic load variation of Zone 3



(d) Dynamic variation of total load

Figure 13. Dynamic change of load and total load in each region

The load variation trend of Zone 1 is consistent with the external disturbance temperature. The loads of Zone 2 and Zone 3 vary nearly sinusoidally, basically consistent with the external disturbance temperature.

The simulation results show that the multi-zone RC thermal network model built in Modelica can well simulate the dynamic load variation under complex dynamic boundaries.

4.3 Dynamic Response Accuracy Analysis

Dynamic response accuracy is analyzed by calculating the error between simulated and measured loads. Considering the complex practical application scenarios of underwater vehicles, the dynamic response accuracy is verified under two typical working conditions: surface navigation and underwater navigation.

The measured data are collected by the monitoring system in the control cabin, as shown in Figure. 14. Temperature sensors collect the air temperature at the air inlet and outlet of each zone. The signal processor transmits, amplifies and converts the sensor output signals, and the terminal data processor outputs the measured load of each zone.

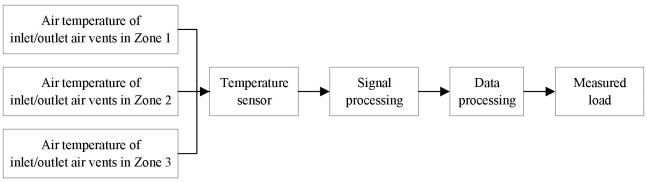
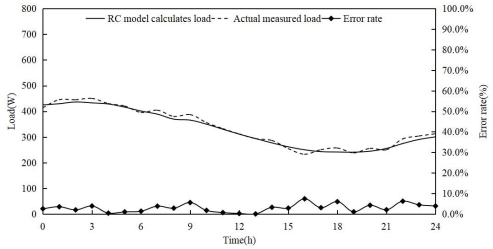
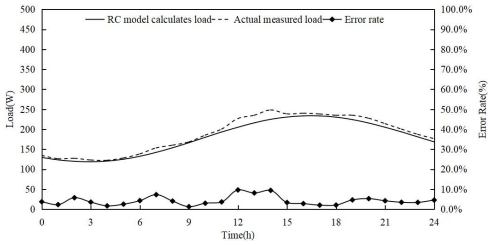


Figure 14. Flow chart of actual load collectionl

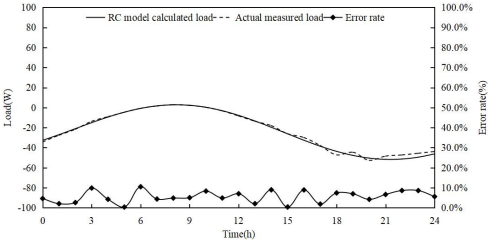
Based on the simulation results and measured loads, the dynamic response comparison of each zone under 24-hour surface navigation is shown in Figure. 15.



(a) Dynamic load response comparison of Zone 1

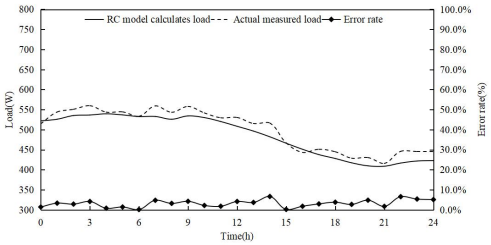


(b) Dynamic load response comparison of Zone 2



(c) Dynamic load response comparison of Zone 3

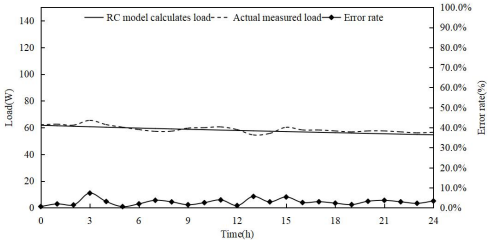
DOI
10.3384/ecp223363



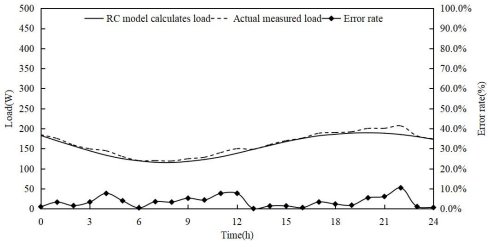
(d) Dynamic response comparison of total load

Figure 15. Surface condition RC model: dynamic response comparison of various regions

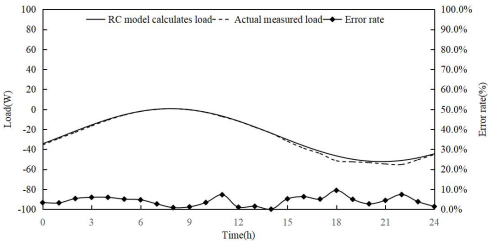
The dynamic response comparison under underwater navigation is shown in Figure. 16.



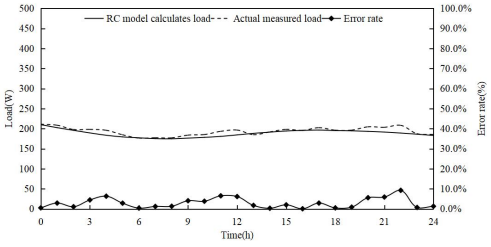
(a) Dynamic load response comparison of Zone 1 under underwater conditions



(b) Dynamic load response comparison of Zone 2 under underwater conditions



(c) Dynamic load response comparison of Zone 3 under underwater conditions



(d) Dynamic response comparison of total load under underwater conditions

Figure 16. Underwater Operating Conditions: Dynamic Response Comparison of RC Model Regions

The dynamic response error analysis results under surface navigation are listed in Table 3.

Table 3. Dynamic response error results of water surface conditions

Region	Maximum error rate /%	Average error rate /%
Region 1	7.38	2.96
Region 2	9.63	4.24
Region 3	10.51	5.57
Total heat load	6.64	3.17

The dynamic response error analysis results under underwater navigation are listed in Table 4.

Table 4. Dynamic response error results of underwater conditions

Region	Maximum error rate /%	Average error rate /%
Region 1	7.29	2.86
Region 2	10.45	3.52
Region 3	9.42	4.06
Total heat load	9.24	2.90

The error analysis shows that the maximum error rate of each zone and total load under both working conditions is within 11%, and the average error rate is less than 6%. Therefore, the multi-zone RC thermal network model built in Modelica has high dynamic response accuracy, good reliability and usability, and can be applied to multi-zone dynamic load calculation under complex working conditions.

To further verify the accuracy and effectiveness of the proposed Modelica-based RC thermal network model, the results are compared with those of other methods in Table 5.

Table 5. Comparative analysis of different load calculation methods

Computation method	Maximum error rate /%
Steady-state calculation (Li et al. 2013)	124.00
Numerical simulation (Li et al. 2013)	Structure 1：11.20 Structure 2：1.80
Response factor method (Zhou 2004)	9.12（≤120 items） 0.63（≥120 items）
z-transfer function model (Zhou 2004)	23.00
Modelica-based RC thermal network model	9.24

The comparison shows that the Modelica-based RC thermal network model outperforms steady-state calculation and z-transfer function method in calculation accuracy. For numerical simulation and reaction coefficient method, the maximum error rate varies greatly due to grid division and reaction coefficient terms, and their accuracy may be lower than the proposed model. This further validates the good accuracy and usability of the model.

5 Conclusion

This paper studies a multi-zone dynamic load calculation model based on the RC thermal network and builds its dynamic simulation model using Modelica. The model supports designers to quickly change parameters and boundary conditions, providing a convenient simulation

tool for dynamic load calculation. Through dynamic load analysis of each zone under complex dynamic boundaries and dynamic response verification under two working conditions, the established RC thermal network model is proven to have high dynamic response accuracy and reliability. It can be applied to various complex zone dynamic load calculation scenarios.

References

Chen, C. (2013). "Research on Thermal-Fluid System Modeling Method and Application with the Object-Oriented Technology". Thesis. Wuhan: Huazhong University of Science and Technology.

Ding, J. W. (2006). "Research on Methods for Consistency Analysis and Reduction of Declarative Simulation Models". Thesis. Wuhan: Huazhong University of Science and Technology.

Dong, P., Z. Y. Li, and Q. F. Jian (2012). "Large Temperature Difference Cold Air Distribution and VAV Air Conditioning Systems in Submarine". In: *Chinese Journal of Ship Research* 7.6, pp. 90–97.

Fang, Z. Q. (2024). "Building Room Temperature Prediction and Control Based on Equivalent RC Model". In: *Modern Architecture Electric* 4.1, pp. 23–35.

Felgner, F., R. Merz, and L. Litz (2006). "Modular Modelling of Thermal Building Behaviour Using Modelica". In: *Mathematical and Computer Modelling of Dynamical Systems* 12.1, pp. 35–49.

Fraisse, G., C. Viardot, O. Lafabrie, et al. (2002). "Development of a Simplified and Accurate Building Model Based on Electrical Analogy". In: *Energy and Buildings* 34.10, pp. 1017–1031.

Gao, S. (2005). "Dynamic Cooling Loads of Ship Air-condition and Calculation". Thesis. Dalian: Dalian University of Technology.

Hassid, S. (1985). "A Linear Model for Passive Solar Calculations: Evaluation of Performance". In: *Building and Environment* 20.1, pp. 53–59.

He, G. H., M. J. Zhong, and X. Gao (2002). "Transfer Function Method Used to Calculate the Air Conditioning Cooling Load in Submarine". In: *Journal of Naval University of Engineering* 14.3, pp. 45–58.

He, J. P. and D. F. Wang (2000). "The Application Study for Unsteady Heat Transmission of Wall-Method of Reaction Coefficient". In: *Journal of Engineering Thermophysics* 21.1, pp. 93–96.

Hu, Y. X. and Y. Yao (2002). "Research on the Method of Air Conditioning Cooling Metering Based on Frequency Response". In: *Heating Ventilating & Air Conditioning* 32.5, pp. 113–115.

Huang, Z. Y. (2019). "Research on Comprehensive Energy Efficiency Evaluation Technology for Ship Air Conditioning System". Thesis. Wuhan: Huazhong University of Science and Technology.

Lauster, M., J. Teichmann, M. Fuchs, et al. (2014). "Low Order Thermal Network Models for Dynamic Simulations of Buildings on City District Scale". In: *Building and Environment* 73, pp. 223–231.

Li, W. G., A. B. Li, X. H. Xu, et al. (2013). "Numerical Analysis of Heat Transfer for Complicated Ship Casing Walls". In: *Chinese Journal of Ship Research* 12.8, pp. 85–90.

Paschkis, V. and H. D. Baker (1942). "A Method for Determining Unsteady-State Heat Transfer by Means of an Electrical Analogy". In: *Transactions of the American Society of Mechanical Engineers* 64.4, pp. 105–112.

Shi, X., Q. Zhang, Y. Zhao, et al. (2014). "Review of RC Thermal Network Building Energy Consumption Forecasting Model". In: *Chinese Journal of Scientific Instrument* 35.12, pp. 59–65.

The State Bureau of Quality and Technical Supervision (1992). *GB/T 13409-92 Design parameters and calculation methods for air conditioning and ventilation in ship living quarters*. Standard. Beijing: Standards Press of China.

Wang, Z. J. and X. Y. Xia (2004). "One-Dimensional Dynamic Models for Radiant Floor Heating/Cooling". In: *Journal of Nanjing Normal University (Engineering and Technology Edition)* 4.1, pp. 1–4.

Xu, X. and S. Wang (2007). "Optimal Simplified Thermal Models of Building Envelope Based on Frequency Domain Regression Using Genetic Algorithm". In: *Energy and Buildings* 39.5, pp. 525–536.

Yang, W. (2010). "Study of Thermal Response Model and Algorithm of U-Vertical Ground Heat Exchanger". Thesis. Changsha: Hunan University.

Zhou, J. (2004). "The Research on Applying Novel but Simple Methods for the Space Load Calculations in HVAC". Thesis. Changsha: Hunan University.

Modelling of a Coupled PV and ITES Air Conditioning System Based on MWorks and its Operation Analysis for a Large High-speed Railway Station

Guo Zhang[1] Weihua Zhou[1] Xinhua Xu[1] Fei Guo[2] Liwei Tian[2]

[1]Department of building environment and energy engineering, Huazhong University of Science and Technology, China,{bexhu@hust.edu.cn}
[2]China Railway Siyuan Survey and Design Group Co., Ltd, China

Abstract

Large high-speed railway stations are major energy consumers with enormous cooling loads. Integrating distributed renewables such as photovoltaics (PV) offers new prospects for building decarbonization but brings challenges to system dispatch and economic optimization. This study evaluates operational modes of a coupled PV and ice thermal energy storage (ITES) air conditioning system to maximize economic benefits. A physical simulation model of the PV+ITES system including PV module, chiller module, pump module, ice storage tank module etc. was built in MWorks. Three operation modes are considered, and the yearly operation performances were simulated. Quantitative comparison shows Mode 3 (combining PV direct supply and nighttime off-peak ice storage via TOU tariffs) performs optimally, with an annual net operating cost of 2.852 million CNY and 27.53% cost saving vs. Mode 1 (independent PV grid-connection), providing practical references for large transportation hubs.

Keywords: PV, ITES system, Operation analysis, MWorks

1 Introduction

Large high-speed railway stations, characterized by massive floor areas and high occupant densities, exhibit immense air conditioning cooling loads (Chen et al. 2025). They are major energy consumers within the transportation infrastructure sector. The expansive roofs of these stations present significant opportunities for the large-scale deployment of rooftop PV systems in dense urban areas (Zhu et al. 2023). However, a spatio-temporal mismatch often exists between PV generation peaks and building cooling load peaks. Consequently, standalone grid-connected PV systems struggle to achieve complete self-consumption. In this context, integrating energy storage technologies has emerged as a crucial strategy to promote the effective application of renewable energy. ITES can convert surplus PV power or off-peak nocturnal electricity into ice for storage. During periods of peak power demand, the system discharges this cold energy to reduce the operating load of chillers (Dong et al. 2023). ITES not only smoothes PV output fluctuations and enhances the self-consumption rate, but also effectively facilitates peak shaving and load shifting for the power grid (Hillen et al. 2025).

The coordinated operation and dispatch of coupled renewable energy and storage systems face complexity challenges. These challenges primarily arise from strong multi-component coupling and nonlinear thermodynamic dynamics. To evaluate the operating strategies of such coupled systems, existing studies typically adopt two main approaches. The first approach relies on mathematical optimization algorithms to seek globally optimal dispatch commands. For example, Shi et al. (2022) developed a mixed-integer linear programming model for the coordinated optimization of storage and generation capacities in a coupled system. Onaolapo et al. (2021) employed an adaptive model predictive control algorithm for the optimal dispatch and economic evaluation of PV, wind power, and battery energy storage systems in microgrids. Cirocco et al. (2022) developed an advanced control and forecasting algorithm to cooperatively dispatch PV and latent heat thermal energy storage to respond to electricity prices and achieve peak shaving. However, these methods usually require highly linearized assumptions for key thermodynamic processes, such as the nonlinear heat transfer during ice phase change. This oversimplification frequently leads to significant performance deviations between theoretical optimums and actual engineering implementations (Hao et al. 2022).

The second approach emphasizes dynamic physical simulation. It focuses on the high-fidelity capture of the authentic, nonlinear transient behaviors of HVAC equipment and energy storage devices. Tang et al. (2023) constructed nonlinear physical models of ice coils and chillers based on heat transfer analysis, accurately reflecting dynamic phase-change processes and chiller degradation under varying conditions. Pinamonti et al. (2020) utilized TRNSYS to perform high-precision simulations of a coupled PV, heat pump, and thermal storage system to evaluate practical control strategies. Nonetheless, traditional physical simulation tools typically adopt a causal (signal-flow) modelling paradigm. When applied to multi-domain, dynamically changing energy systems,

these tools are often compelled to adopt idealized control assumptions to avoid convergence issues. Furthermore, they tend to neglect the dynamic hydraulic characteristics and reverse pressure feedback of the pipe network (Fan et al. 2021).

In contrast, the acausal modelling language Modelica provides a highly flexible and physically compliant virtual environment. By supporting object-oriented, modular modelling from the first principles of physics, Modelica significantly enhances the efficiency and intuitiveness of modelling complex systems (Lu et al. 2021). The Modelica Buildings library, developed by the International Building Performance Simulation Association , has established the authoritative position of acausal modelling in handling interdisciplinary building energy systems (Wetter et al. 2014). In recent years, numerous scholars have conducted pioneering explorations using Modelica in related fields. For instance, Hinkelman et al. (2022) achieved high-fidelity dynamic simulations of large-scale district cooling networks and their pipeline hydraulics using Modelica. Li et al. (2021) specifically developed and validated a high-precision physical model of an ice storage tank based on Modelica, accurately capturing the nonlinear transient phase-change heat transfer process. Furthermore, Chen et al. (2025) successfully applied such phase-change physical models to the dynamic performance evaluation of complex HVAC systems. Targeting broader application scenarios, recent studies have not only utilized Modelica to accomplish the reduced-order physical validation of large-scale thermal energy storage systems (Formhals et al. 2024), but also achieved rapid, high-fidelity transient simulations of phase-change storage materials at the building level (Menezes et al. 2026).

Therefore, taking a large high-speed railway station as the background, the primary objectives of this study are to: (1) construct a digital-physical model of the coupled PV-ITES air conditioning system based on the MWorks platform, aiming to simulate and predict its transient responses under various disturbances, including solar radiation, ambient temperature, and dynamic building loads; (2) quantitatively evaluate the actual operational performance and economic benefits of different operation modes through dynamic simulation analysis; and (3) quantify the economic response of each operational mode to dynamic grid electricity prices. This study can provide a reliable theoretical basis and robust data support for the subsequent formulation of system control strategies.

2 System Description

This study investigates a coupled PV and ITES air conditioning system in a high-speed railway station in a south china city. From an energy provision perspective, a 2.05 MWp distributed PV array is integrated with the 10 kV railway grid for self-consumption, effectively reducing grid reliance. From the cooling production and distribution perspective, the system employs a 25% ethylene glycol aqueous solution as the primary heat transfer medium.

The principle of the chilled water system is illustrated in Figure 1. The primary loop is hydraulically decoupled from the secondary chilled water network to the user terminal via a 6860 kW plate heat exchanger. This specific configuration ensures robust physical isolation while enabling efficient thermal exchange. The core equipment comprises a single dual-mode chiller (cooling capacity: 4219 kW in cooling mode, 2827 kW in ice-storage mode) alongside two base-load chillers (3164 kW each). Latent heat storage is accomplished through an array of 24 individual ice storage tanks, each boasting a capacity of 277 RTh. Furthermore, the fluid transport and heat rejection infrastructure is supported by two ethylene glycol pumps, four chilled water pumps, four cooling water pumps, and seven cooling towers.

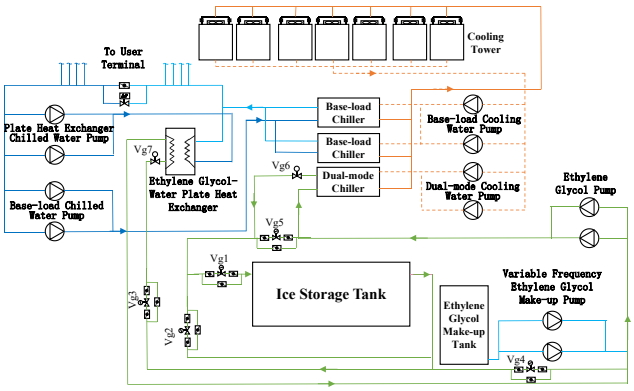


Figure 1. Principle of the chilled water system.

Furthermore, the ITES air conditioning system is configured with four primary operational modes: (1) ice-storage by the dual-mode chiller coupled with independent cooling by the base-load chillers; (2) combined direct cooling by both the dual-mode and base-load chillers; (3) joint cooling by all chillers combined with ice discharging; and (4) joint cooling by the base-load chillers and ice discharging. An automated control system seamlessly transitions among these modes based on real-time cooling loads, time-of-use (TOU) electricity tariffs, and predefined optimization strategies.

3 Modelling Based on MWorks

The coupled PV-ITES air conditioning system in a large high-speed railway station is an interdisciplinary engineering object that encompasses multiple physical domains, including thermodynamics, fluid dynamics, power electronics, and automated control. Modelica is a new-generation equation-based, acausal, and object-oriented modelling language. It can seamlessly bridge diverse physical fields, facilitating the convenient modelling of complex physical systems. MWorks serves as a comprehensive system-level design and simulation verification platform tailored for multi-domain industrial products. It fully supports the Modelica unified modelling specifica-

DOI
10.3384/ecp223371

tion and adopts a hierarchical modelling approach that mirrors real-world physical topologies thereby robustly underpinning model-based system engineering applications. Furthermore, MWorks provides a complete suite of capabilities, including system simulation modelling, compilation analysis, simulation solving and post-processing. It comprehensively covers the entire model-based system verification process.

To evaluate the dynamic operational performance of the aforementioned coupled PV and ITES air conditioning system, a system simulation model was developed in MWorks by using the Modelica language. Initially, physical models of core fundamental components, including the chiller model, the water pump model, the plate heat exchanger model, he cooling tower model, the PV system and the ice storage tank. They were established based on conservation laws and actual equipment performance parameters.

In the chiller model, a performance curve-based approach derived from the DOE-2.1 model is adopted. The dynamic available cooling capacity and the compressor power consumption are evaluated by applying correction functions for evaporator/condenser temperatures and part-load conditions. The core governing equations are expressed as the following formulas:

$$Q_{act} = Q_{ref} \times \mathrm{CAPFT}(T_{chw,o}, T_{cw,o}) \tag{1}$$

$$\begin{aligned} P_{act} = {} & P_{ref} \times \mathrm{CAPFT}(T_{chw,o}, T_{cw,o}) \\ & \times \mathrm{EIRFT}(T_{chw,o}, T_{cw,o}) \\ & \times \mathrm{EIRFPLR}(\mathrm{PLR}, T_{cw,o}) \end{aligned} \tag{2}$$

Where, Q_{act} and Q_{ref} denote the dynamic available and rated cooling capacities (W), respectively; P_{act} and P_{ref} are the actual and rated power consumptions (W); $T_{chw,o}$ is the leaving chilled water temperature from the evaporator (°C); and $T_{cw,o}$ is the leaving cooling water temperature from the condenser (°C). Both CAPFT and EIRFT are biquadratic polynomial functions of $T_{chw,o}$ and $T_{cw,o}$. Notably, the EIRFPLR is a biquadratic polynomial function of both the leaving condenser water temperature ($T_{cw,o}$) and the partial load ratio (PLR), effectively capturing the coupling degradation effects under off-design conditions.

In the water pump model, an empirical performance curve method is utilized. Based on the manufacturer's characteristic data at the nominal speed, the pump's transient pressure head and power consumption at any variable speed are dynamically calculated by using the affinity laws. The fundamental equations are defined as the following formulas:

$$H = r_N^2 \cdot s\left(\frac{\dot{V}}{r_N}, d\right) \tag{3}$$

$$P = r_N^3 \cdot s\left(\frac{\dot{V}}{r_N}, d\right) \tag{4}$$

Where, H is the pump head under specific operating conditions (Pa); P is the dynamic power consumption (W); r_N is the normalized speed (the ratio of actual speed to nominal speed); $\frac{\dot{V}}{r_N}$ represents the equivalent volumetric flow rate converted to the nominal speed condition; $s(...)$ represents the spline interpolation function based on the nominal performance curve data d.

In the plate heat exchanger model, the ε-NTU (effectiveness-NTU) method is implemented. Unlike simplified models with a constant overall heat transfer coefficient (UA), this model dynamically updates the UA value based on the real-time mass flow rates of the fluids to accurately reflect transient convective heat transfer behaviors. The core equations are expressed as the following formulas:

$$UA = \frac{1}{\frac{1}{hA_1} + \frac{1}{hA_2}} \tag{5}$$

$$Q_{flow} = \varepsilon \cdot Q_{max} \tag{6}$$

Where, UA is the dynamic overall heat transfer coefficient multiplied by the area (W/K); hA_1 and hA_2 are the products of the transient convective heat transfer coefficient and area for fluid 1 and fluid 2, respectively, which vary with real-time mass flow rates; Q_{flow} is the actual heat transfer rate (W); ε is the heat transfer effectiveness determined by NTU and flow regimes; Q_{max} is the theoretical maximum possible heat transfer rate (W).

In the cooling tower model, a physics-based grey-box approach based on Merkel's theory is adopted. The complex heat and mass transfer process is equivalently modeled as a counter-flow heat exchanger using the ε-NTU method, driven by the enthalpy difference between the saturated air at the water surface and the mainstream air. The actual heat transfer rate and the dynamic modification of the UA value under off-design conditions are expressed as the following formulas:

$$UA_e = UA_0 \cdot f_{wb} \cdot f_{air} \cdot f_{wat} \tag{7}$$

$$Q_t = \varepsilon \times C_{min} \times (T_{w,in} - T_{a,wb,i}) \tag{8}$$

Where, UA_e and UA_0 are the equivalent and nominal overall heat transfer coefficient-area products, respectively; f_{wb}, f_{air}, and f_{wat} represent the modification factors for current wet-bulb temperature, airflow, and water flow; Q_t is the actual total heat transfer rate of the cooling tower (W); ε is the effectiveness; C_{min} is the minimum heat capacity rate between air and water (W/K); $T_{w,in}$ and $T_{a,wb,i}$ are the entering water temperature and entering air wet-bulb temperature, respectively.

In the PV system model, the hourly power yield is dynamically calculated based on the global horizontal irradiance, installed capacity, and a comprehensive efficiency coefficient. The core governing equation is expressed as the following formula:

$$E_P = H_A \times \frac{P_{AZ}}{E_S} \times K \tag{9}$$

Where, E_P is the system power yield (kWh); H_A denotes the global horizontal irradiance (kWh/m^2); P_{AZ} is

the installed capacity (kWp); E_S is the standard test condition irradiance constant (defaulted to 1 kWh/m^2); K is the comprehensive efficiency coefficient, estimated between 81% and 85%.

In the ice storage tank model, a dynamic calculation method is employed based on the performance curves to reflect the transient thermodynamic characteristics during the charging and discharging processes. The rate of change of its core state parameter, the State of Charge (SOC), is governed by the following formula:

$$\frac{\mathrm{d}SOC}{\mathrm{d}t} = \frac{\dot{Q}}{H_f \cdot m_{ice,max}} \tag{10}$$

Where, SOC is a dimensionless number ranging from 0 to 1 (0 means completely melted, 1 means completely frozen); H_f is the latent heat of fusion of ice (J/kg); $m_{ice,max}$ is the maximum theoretical ice storage mass (kg); $\dot{Q}$ represents the transient heat transfer rate (W), which is positive during charging and negative during discharging.

Upon completing the construction of individual component models, these modules were organically coupled within the MWorks platform to establish the coupled system model, as depicted in Figure 2.

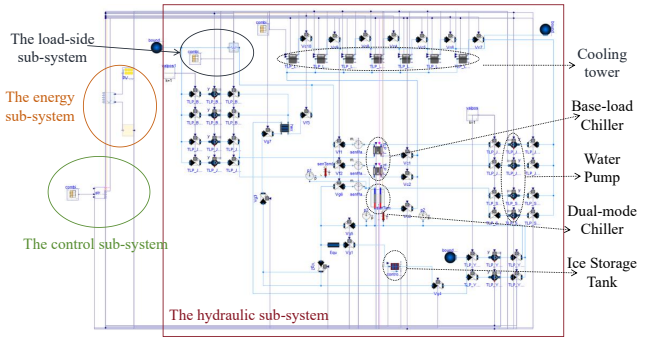


Figure 2. Simulation model of the coupled PV-ITES air conditioning system.

This coupled system model primarily comprises four sub-systems:

- The Hydraulic Sub-system: This constitutes the physical backbone of the entire model, mapping the actual piping topology and component interconnections. It encompasses all primary equipment, including the baseload chillers, dual-mode chillers, the ice storage tank, cooling towers, and the associated network of variable-speed pumps and valves.
- The Load Sub-system: It is represented by a simplified Air Handling Unit (AHU) model that calculates the required heat extraction based on building load profiles. This creates a closed-loop feedback mechanism, where the chilled water supply temperature from the hydraulic sub-system directly influences the AHU's heat exchange performance, ensuring a realistic representation of the interaction between the generation and demand sides.
- The Energy Sub-system: This module functions as the performance aggregator, responsible for quantifying the system's overall energy consumption and economic metrics. It dynamically calculates the total instantaneous power draw by summing the contributions from all major electrical components, including chillers, pumps, and cooling tower fans. Simultaneously, it accounts for the on-site PV power generation, computing the net grid electricity exchange that forms the basis for the subsequent economic analysis.
- The Control Sub-system: It receives dynamic boundary conditions as inputs, such as the building cooling load, PV generation forecast, and TOU electricity tariffs. Then it executes the predefined control strategies for each mode. Its outputs are the real-time control signals (e.g., on/off commands, temperature setpoints) that are dispatched to the actuators within the hydraulic sub-system.

4 Operation Modes

To evaluate economic performance and PV integration potential, this study compares three operational modes. Mode 1 is the baseline mode. In this mode, the ice storage system is disabled, building cooling loads are entirely met by conventional chillers, and all generated PV electricity is injected into the external grid. It is worth noting that while Mode 1 could theoretically utilize 100% pure water to improve hydraulic efficiency due to higher specific heat capacity, the simulation consistently maintained the 25% ethylene glycol properties across all modes to ensure a strictly unified physical baseline for economic comparison. To maximize hydraulic efficiency, the control logic strictly prioritizes the water-based base-load chillers. The dual-mode chiller remains on standby and is only activated as a supplementary backup during extreme peak load conditions. Thus, the majority of the cooling distribution benefits from the lower pump power consumption associated with pure water. Mode 2 utilizes direct cooling with PV self-consumption. The ice storage remains disabled, but the chillers prioritize local PV consumption, exporting surplus power to the grid and drawing from the grid only when encountering a power deficit. Mode 3 combines nighttime off-peak ice storage with daytime PV direct-supply. Deeply integrated with local TOU tariffs, this mode charges the ice storage using inexpensive grid electricity during nighttime off-peak valleys. During the daytime, it prioritizes PV output to drive base-load chillers in conjunction with ice discharging.

The 24-hour time-series dispatch logic of the system is illustrated in Figure 3. The different color blocks intuitively map the real-time operational boundaries of the system: blue for the idle state with no cooling load, red for direct cooling by conventional chillers, green for ice charging by the dual-mode chiller yellow for ice discharging from the storage tank.

DOI
10.3384/ecp223371

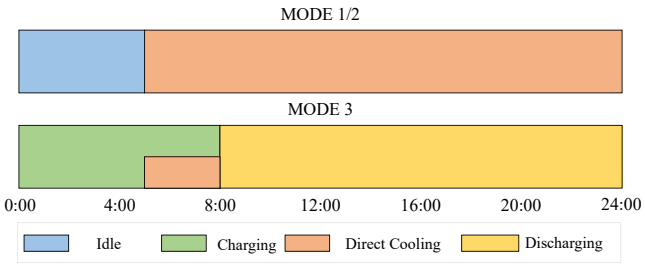


Figure 3. 24-hour time-series dispatch logic of the system.

During the direct cooling phase, a capacity allocation mechanism with hysteresis is embedded within the controller to ensure stable state transitions and prevent equipment oscillation caused by transient load fluctuations. The load threshold for activating the next sequential chiller is set at 95% of the total rated capacity of the currently running chillers, while the threshold for sequential shut-down is set at 80%. The specific staging sequence is as follows:

1. Base-load Chiller 1 operates as the primary active cooling source.
2. Base-load Chiller 2 is activated only when the real-time cooling load exceeds 95% of Chiller 1's rated capacity, and is shut down when the load drops below 80% of Chiller 1's rated capacity.
3. The dual-mode chiller (operating in cooling mode) is engaged only when the load exceeds 95% of the combined rated capacity of both base-load chillers, and is shut down when the load drops below 80% of this combined capacity.

During the ice discharging phase, the system prioritizes melting stored ice first, only activating regular chillers if the discharged cooling energy is insufficient to meet the instantaneous load.

In Mode 3, the periods from 5:00 to 8:00 AM operate as a parallel charging-and-cooling phase. The dual-mode chiller maintains ice-storage while the base-load chillers are sequentially activated to handle the initial building cooling loads. However, to guarantee cooling reliability, the system will take emergency action under extreme operating conditions. When the terminal load exceeds the maximum cooling capacity of the base-load chillers, it forcibly interrupts the ice-charging loop and smoothly switches the dual-mode chiller to the cooling state to respond to the peak load.

5 Results and Analysis

5.1 Evaluation Index

To systematically assess the operational characteristics, energy conversion efficiency, and grid interaction level of the PV system in the target scenario, the following key metrics are introduced to quantify the source-load matching performance.

The Self-Consumption Rate (SCR) denotes the proportion of PV-generated electricity that is directly consumed by the local load. It serves as a core indicator for evaluating the economic viability and source-load synergy of distributed PV systems. A higher SCR indicates that the majority of the PV generation is locally utilized, mitigating the economic degradation associated with high-tariff grid feed-in. The hourly self-consumption rate $SCR(t)$ is formulated as:

$$SCR(t) = \frac{E_{PV}(t) - E_{g,ex}(t)}{E_{PV}(t)} \times 100\% \tag{11}$$

Where, $E_{PV}(t)$ represents the PV power generation at time t (kWh); $E_{g,ex}(t)$ is the surplus PV power fed into the utility grid at time t (kWh).

Self-Sufficiency Rate (SSR) evaluates the system from the demand side. This metric intuitively reflects the local systems degree of independence from the main grid. A higher SSR implies enhanced system resilience against grid price fluctuations and potential supply instabilities. It represents the fraction of the local load demand directly fulfilled by the PV system:

$$SSR(t) = \frac{E_{load}(t) - E_{g,in}(t)}{E_{load}(t)} \times 100\% \tag{12}$$

Where, $E_{g,in}(t)$ is the power purchased from the external grid at time t due to insufficient PV output (kWh); $E_{load}(t)$ is the local load demand at time t (kWh).

The total self-consumption rate (SCR_{total}) and total self-sufficiency rate (SSR_{total}) can be derived by aggregating the aforementioned hourly metrics over a statistical period T (e.g., daily, monthly, or annually). Long-term evaluation metrics comprehensively reflect the overall energy efficiency of the system over the time dimension.

5.2 Dynamic Simulation Results and Analysis

Based on the annual dynamic simulation via the MWorks platform, the energy generation and consumption distributions under these three operation modes are presented in Table 1. In the baseline scenario (Mode 1), the annual grid electricity consumption reaches 4.90 GWh, while 2.33 GWh of PV generation is entirely exported to the grid, leading to zero local PV utilization. Following the introduction of the PV self-consumption mechanism (Mode 2), the system directly consumes 1.39 GWh of PV electricity, significantly reducing the grid electricity demand by 28.33%. Notably, the annual grid electricity consumption rebounds to 4.54 GWh in Mode 3, which is higher than in Mode 2. This rebound occurs because the load-shifting strategy transfers the heavy ice-storage load to the nighttime. During this period, the system not only relies entirely on the grid without any PV offset but also incurs the inherent thermodynamic efficiency degradation and auxiliary power consumption of the ice-storage cycle. Furthermore, this strategy results in a lower PV self-consumption of 1.16 GWh compared to Mode 2. This highlights a

critical trade-off: optimizing for price arbitrage in Mode 3 comes at the cost of reduced overall energy efficiency and lower direct PV utilization compared to a pure self-consumption strategy.

Table 1. Comparison of annual energy performance under 3 alternative modes.

Parameter	*Mode 1*	*Mode 2*	*Mode 3*
Grid electricity consumption (kWh)			
Annual total	4,902,029	3,513,142	4,536,961
Change vs. Mode 1	–	−1,388,888	−365,068
Reduction	–	28.33%	7.45%
PV self-consumption (kWh)			
Annual total	0	1,388,887	1,159,899
Increase vs. Mode 1	–	+1,388,887	+1,159,899
PV grid-export (kWh)			
Annual total	2,327,012	938,124	1,167,113
Change vs. Mode 1	–	−1,388,888	−1,159,899
Reduction	–	59.69%	49.84%

To deeply understand the thermodynamic behaviors and dispatch characteristics of different operation modes, a dynamic analysis on a typical day is indispensable. Although comprehensive dynamic simulations were conducted across various typical meteorological scenarios, including low, medium-high, and peak cooling load days. This study primarily presents the detailed transient results of the representative high-load day due to manuscript length constraints. The high-load scenario can effectively highlight the most rigorous operating conditions of the HVAC equipment, the maximum potential of PV energy utilization, and the critical thermodynamic scheduling conflicts among different operation modes.

To understand the thermodynamic characteristics that drive the energy distribution in different operation modes, Figure 4 provides a detailed breakdown of electricity consumption and the dynamic system Coefficient of Performance (COP). For Mode 1/2, the cooling demand is supplied directly by the chillers operating under conventional cooling conditions. The system COP remains relatively stable between 3.5 and 4.5 during the daytime operating hours (05:0023:00). In Mode 3, the integration of the ITES strategy alters the systems efficiency profile and load distribution. To accomplish the load-shifting objective, the dual-mode chiller operates extensively during the nocturnal hours (00:0007:00) to produce ice-storage. Due to the physical necessity of operating the evaporator at sub-zero temperatures during the ice-storage cycle, the compressor must overcome a larger temperature lift. Consequently, the system COP temporarily decreases to approximately 2.0 during the night. This inherent thermodynamic characteristic of latent heat storage explains the increased power consumption required to store cooling energy. During the morning ice-discharging phase (08:0010:00), the chillers are deactivated and the stored latent heat is released to meet the building's demand, with only the auxiliary pumps consuming power. During this period, the system achieves a high system COP (peaking at approximately 17.5).

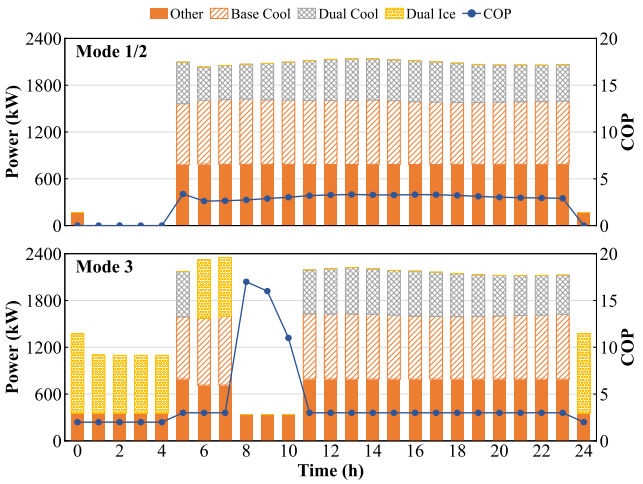


Figure 4. Detailed electricity consumption breakdown and the system COP on a typical high-load day.

Figure 5 illustrates the hourly electricity consumption breakdown alongside the SSR. In Mode 2, the system's dynamic electrical load follows a conventional bell-shaped curve, aligning seamlessly with the daytime cooling demand. Consequently, its SSR maintains a robust level during the daytime, peaking at 72% around 13:00, which effectively reduces peak-hour grid reliance. In contrast, Mode 3 reshapes the load profile to leverage off-peak electricity tariffs. Massive ice-storage loads are shifted to the nighttime. As depicted by the towering right bars in Figure 4, this period operates entirely on grid power (SSR = 0%), creating nighttime consumption peaks that significantly exceed the daytime loads of Mode 2. During the subsequent daytime (08:0011:00), the cooling demand is primarily satisfied by melting ice, causing the chillers to deactivate. With only minimal power load from auxiliary pumps remaining, the SSR hits 100%.

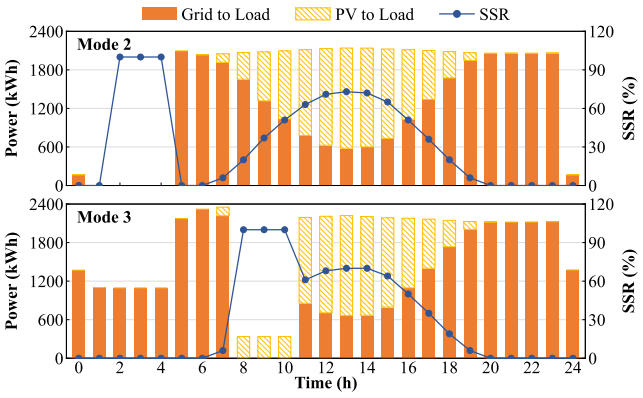


Figure 5. Dynamic electricity consumption breakdown and SSR on a typical high-load day.

Figure 6 compares the dynamic PV energy distribution and the SCR. Because the daytime electrical load perfectly

DOI
10.3384/ecp223371

matches the PV generation profile in Mode 2, almost all generated PV energy is consumpted by the active chillers, maintaining a 100% SCR from 07:00 to 19:00. Conversely, for Mode 3, the deactivation of chillers during the morning ice-discharging period results in a gap between the solar generation and the local load. The SCR plunges to a nadir of approximately 32% at 10:00, this mismatch causes substantial surplus PV energy to be exported to the grid, resulting in a waste of the PV self-consumption potential.

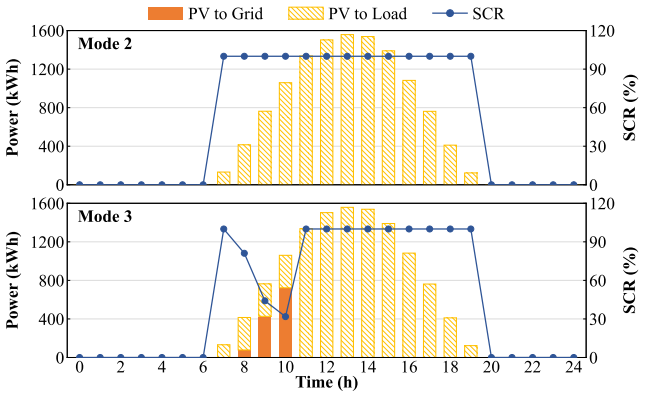


Figure 6. Dynamic PV utilization and SCR on a typical high-load day.

5.3 Economic Performance of the alternative operation modes

The economic performance of the three operation modes is detailed in Table 2.

Table 2. Comparison of annual economic performance under 3 modes.

Parameter	*Mode 1*	*Mode 2*	*Mode 3*
Grid electricity cost (CNY)			
Annual total	4,988,814	3,427,889	3,380,360
Change vs. Mode 1	–	−1,560,925	−1,608,454
Reduction	–	31.29%	32.24%
PV grid-export revenue (CNY)			
Annual total	1,054,136	424,970	528,702
Change vs. Mode 1	–	−629,166	−525,434
Reduction	–	59.69%	49.84%
Net operating cost (CNY)			
Annual total	3,934,678	3,002,919	2,851,658
Change vs. Mode 1	–	−931,759	−1,083,020
Reduction	–	23.68%	27.53%

The annual net operating cost is highest in Mode 1, at 3.9347 million CNY. By increasing the PV self-consumption rate, Mode 2 achieves a net cost reduction of 23.68%. However, Mode 3 exhibits the most superior economic benefits although it consumes more grid electricity than Mode 2. Its success stems from a strategy of electricity price arbitrage. By shifting the ice-storage energy to low-price nighttime, Mode 3 slashes the exorbitant cooling electricity expenses during daytime peaks. The effectiveness of this strategy is evident when comparing energy and cost reductions. The corresponding grid electricity cost plummets by 32.24% while grid electricity consumption is only reduced by 7.45% when compared to the baseline. While this mode accommodates a slightly lower 1.16 GWh of local PV self-consumption, the financial gains from load shifting outweigh the reduced PV utilization benefits. Ultimately, Mode 3 brings the annual net operating cost down to 2.8517 million CNY, achieving a maximized economic saving of 27.53% relative to the baseline. The results indicate that in a grid environment with significant peak-valley price spreads, the systems economic performance is not solely dependent on maximizing the PV self-consumption rate, but rather on an intelligent co-optimization of energy storage dispatch and TOU tariffs.

6 Conclusions

This study investigates the dispatch modes and economic response potential of a coupled PV and ITES air-conditioning system in a large high-speed railway station. The simulation model of the coupled system was constructed in MWorks by using the acausal Modelica language. The performance of three typical operation modes, including the independent PV grid-connection (Mode 1), PV self-consumption (Mode 2), and a dynamic mode combining PV direct supply with nighttime off-peak ice storage utilizing TOU tariffs (Mode 3), are analyzed and compared. The results demonstrate that Mode 3 is the optimal strategy for economic viability. Annual dynamic simulations confirm that, at the expense of a minor loss in physical energy efficiency, this mode utilizes deep tariff arbitrage to slash the baseline net operating cost from 3.9347 million CNY to 2.8517 million CNY, culminating in an impressive 27.53% economic saving. Additionally, this study verifies that it is effectiveness for evaluating the non-linear transient behaviors of complex multi-energy systems by using digital physical modelling combined with underlying hysteresis threshold controls and sequential dispatch logic. This study quantifies the economic response capacity of various dispatch modes to grid tariffs, and it is beneficial for the preliminary planning, flexible dispatch optimization, and investment decision-making of the coupled PV and ITES air-conditioning system in next-generation transportation hubs.

Acknowledgements

This work was supported by the Key Research and Development Project (Technology Innovation Program) of Hubei Provincial Science and Technology Plan (Grant number: 2025BCB033).

References

Chen, S., X. Zhou, Y. Yang, J. Li, S. Zhao, and J. Li (2025). "Modelica-based modeling and simulation of a novel dedi-

cated outdoor air-conditioning system with double heat recovery". In: *Journal of Building Engineering* 99, p. 111535. DOI: 10.1016/j.jobe.2024.111535.

Cirocco, L., P. Pudney, S. Riahi, R. Liddle, H. Semsarilar, J. Hudson, and F. Bruno (2022). "Thermal energy storage for industrial thermal loads and electricity demand side management". In: *Energy Conversion and Management* 270, p. 116190. DOI: 10.1016/j.enconman.2022.116190.

Dong, H., S. Vanage, and K. Cetin (2023). "Review of Ice Thermal Energy Storage (ITES) using Conventional Control Strategies in Commercial Buildings". In: *2023 ASHRAE Winter Conference*, p. 10. DOI: 10.63044/w23don61.

Fan, C. et al. (2021). "Open-source Modelica models for the control performance simulation of chiller plants with water-side economizer". In: *Applied Energy* 299, p. 117337. DOI: 10.1016/j.apenergy.2021.117337.

Formhals, J., X. Kirschstein, A. Dahash, L. Seib, and I. Sass (2024). "Development, validation and demonstration of a new Modelica pit thermal energy storage model for system simulation and optimization". In: *Geothermal Energy*. DOI: 10.1186/s40517-024-00302-9.

Hao, K., D. Kim, and J. E. Braun (2022). "Comparing the economic performance of ice storage and batteries for buildings with on-site PV through model predictive control and optimal sizing". In: *Journal of Building Performance Simulation* 15.5, pp. 691–715. DOI: 10.1080/19401493.2022.2084161.

Hillen, M., P. Schönfeldt, P. Groesdonk, and B. Hoffschmidt (2025). "Operational Optimization of Seasonal Ice-Storage Systems with Time-Series Aggregation". In: *Energies* 18.22, p. 5988. DOI: 10.3390/en18225988.

Hinkelman, K., J. Wang, W. Zuo, A. Gautier, M. Wetter, C. Fan, and N. Long (2022). "Modelica-based modeling and simulation of district cooling systems: A case study". In: *Applied Energy* 311, p. 118654. DOI: 10.1016/j.apenergy.2022.118654.

Li, G., Y. Fu, A. Pertzborn, J. Wen, and Z. O'Neill (2021). "An Ice Storage Tank Modelica Model: Implementation and Validation". In: pp. 177–185. DOI: 10.3384/ecp21181177.

Lu, X., Y. Fu, Z. O'Neill, and J. Wen (2021). "A holistic fault impact analysis of the high-performance sequences of operation for HVAC systems: Modelica-based case study in a medium-office building". In: *Energy and Buildings* 252, p. 111448. DOI: 10.1016/j.enbuild.2021.111448.

Menezes, M. V. P., A.-H. Othman, J. Tancabel, and V. C. Aute (2026). "An approach for fast and accurate simulation of phase change material based thermal energy storage in buildings". In: *Journal of Energy Storage* 152, p. 120384. DOI: 10.1016/j.est.2026.120384.

Onaolapo, A. K., R. P. Carpanen, D. G. Dorrell, and E. E. Ojo (2021). "Reliability Evaluation and Financial Viability of an Electricity Power Micro-Grid System With the Incorporation of Renewable Energy Sources and Energy Storage: Case Study of KwaZulu-Natal, South Africa". In: *IEEE Access* 9, pp. 159908–159924. DOI: 10.1109/ACCESS.2021.3129980.

Pinamonti, M., A. Prada, and P. Baggio (2020). "Rule-Based Control Strategy to Increase Photovoltaic Self-Consumption of a Modulating Heat Pump Using Water Storages and Building Mass Activation". In: *Energies* 13.23, p. 6282. DOI: 10.3390/en13236282.

Shi, Z., W. Wang, Y. Huang, P. Li, and L. Dong (2022). "Simultaneous optimization of renewable energy and energy storage capacity with hierarchical control". In: *CSEE Journal of Power and Energy Systems* 8.1, pp. 95–104. DOI: 10.17775/CSEEJPES.2019.01470.

Tang, H., J. Yu, Y. Geng, X. Liu, and B. Lin (2023). "Optimization of operational strategy for ice thermal energy storage in a district cooling system based on model predictive control". In: *Journal of Energy Storage* 62, p. 106872. DOI: 10.1016/j.est.2023.106872.

Wetter, M., W. Zuo, T. S. Nouidui, and X. Pang (2014). "Modelica Buildings library". In: *Journal of Building Performance Simulation* 7.4, pp. 253–270. DOI: 10.1080/19401493.2013.765506.

Zhu, H., H. Li, G. Liu, Y. Ge, J. Shi, H. Li, and N. Zhang (2023). "Energy storage in Variable high renewable penetration power systems: Technologies, applications, supporting policies and suggestions". In: *CSEE Journal of Power and Energy Systems* 9.6, pp. 2099–2108. DOI: 10.17775/CSEEJPES.2020.00090.

Control Strategy for a High-Efficiency Refrigeration Plant Based on Dymola Simulation: A Case Study of a Xiamen Airlines Support Facility

Haoran Wu, Yuhui Wang, Hejiang Sun*
School of Environmental Science and Engineering, Tianjin University, Tianjin 300072, China
*Corresponding author: sunhe@tju.edu.cn

Abstract

The operational control strategy of a refrigeration plant directly impacts its annual energy efficiency. Conventional control logic, based on static design conditions or time-scheduled fixed parameters, cannot effectively adapt to continuous fluctuations in cooling load and meteorological conditions. This paper employs a model-based Dymola environment to develop a full-system model of a high-efficiency refrigeration plant and proposes a dynamic control strategy driven by hourly cooling load and outdoor meteorological parameters. Based on the hourly cooling load profile of a public building in a hot-summer warm-winter climate zone, the proposed strategy is simulated and compared against a conventional baseline. The results indicate that the proposed strategy achieves an energy efficiency ratio (*EER*) of 6.02 over the cooling season, representing a 19.4% improvement compared to the baseline strategy. The proposed methodology offers a practical reference for control strategy optimization in chiller plants of large public facilities with fluctuating cooling loads, such as airports and shopping malls.

Keywords: *chiller plant optimization; dynamic simulation; control strategy; Modelica; energy efficiency ratio*

1 Introduction

In public buildings, the energy consumption of the refrigeration plant exceeds 70% of the total energy consumption of the air-conditioning system, and its operational efficiency plays a decisive role in the overall energy performance of the building (Xu, 2022). Although selecting high-efficiency chillers, variable-speed pumps, and other equipment can substantially enhance overall system efficiency, numerous field investigations indicate that the annual comprehensive operational energy efficiency ratio of refrigeration plants typically falls within the range of 3.0 to 4.0, significantly lower than the theoretical values under rated equipment conditions (Yang et al., 2024). The fundamental cause of this discrepancy is that once the equipment system is commissioned, the extent to which its actual performance is realized depends heavily on the appropriateness and adaptability of the control strategy. The control logic prevalent in current engineering practice—such as time-scheduled fixed water temperature setpoints and constant differential pressure variable-frequency control—is inherently discrete and static, governed by empirical rules, and fails to achieve optimal comprehensive system efficiency across the entire operating range (Wu et al., 2024).

To address this issue, pre-evaluating control strategy performance under all operating conditions during the design or retrofit phase has become a critical pathway toward improving actual plant operational efficiency. Dynamic simulation provides the principal means of achieving this goal. Among available tools, Modelica—an object-oriented, equation-based, multi-domain unified modeling language—supports integrated simulation of physical systems and control logic, offering distinct advantages for modeling complex energy systems (Mattsson and Elmqvist, 1997). Its open-source Buildings library provides high-fidelity component models for HVAC systems and has been

applied in building thermal system simulation and control strategy verification (Wetter et al., 2014). Jiang et al. employed the Dymola environment to simulate and analyze the energy-saving effects of different control schemes for water systems in subway stations, demonstrating the environment's suitability for evaluating and comparing control strategies (Jiang et al., 2024). Nevertheless, simulation research on dynamic regulation strategies that comprehensively incorporate multiple parameters for water systems in refrigeration plants remains relatively limited.

This paper employs Dymola environment to construct a full-system dynamic simulation model of a high-efficiency refrigeration plant and proposes a control strategy driven by real-time cooling load demand and outdoor meteorological parameters. To evaluate the performance of the proposed strategy, a baseline strategy reflecting current engineering practice for high-efficiency plants is established. Simulations are conducted under representative cooling season load and meteorological conditions in a hot-summer warm-winter climate zone to analyze the impact of the dynamic regulation strategy on system operating parameters and the mechanisms underlying its energy efficiency gains.

2 Research Methods

2.1 Simulation Model Development

This study focuses on a public building located in a hot-summer warm-winter climate zone (Xiamen). The hourly cooling load during the cooling season (April 1st to November 30th) was simulated using EnergyPlus, yielding a peak load of 3411 kW. Based on the load characteristics, one large-capacity and one small-capacity variable-speed chiller were selected to cover different load demand ranges. The main equipment specifications are presented in Table 1.

Table 1. Main equipment specifications and parameters of the refrigeration plant

Equipment Type	Model	Single Unit Parameters	N
Large Chiller	CVHG720 RT	Cooling: 2532 kW; COP: 6.11	1
Small Chiller	RTHDD4H 3G6V	Cooling: 985 kW; COP: 5.90	1
Cooling Tower	DNT350T	Water flow: 500 m^3/h; Power: 7.5 kW	2
Large Chiller Chilled Water Pump	Grundfos NBG 150/200	Flow: 211 m^3/h; Head: 25 m; Power: 18.5 kW	2
Small Chiller Chilled Water Pump	Grundfos NBG 125/140	Flow: 137 m^3/h; Head: 25 m; Power: 15 kW	1
Large Chiller Cooling Water Pump	Grundfos NK 200/250	Flow: 250 m^3/h; Head: 21 m; Power: 18 kW	2
Small Chiller Cooling Water Pump	Grundfos NK 150/200	Flow: 192 m^3/h; Head: 21 m; Power: 15 kW	1

Chilled water supply/return temperature is 7/13℃; cooling water supply/return temperature is 32/37℃ for both chillers.

In the Grundfos pump model designations, NBG/NK denotes the series; the first group of digits indicates the inlet/outlet diameter (mm) and impeller diameter (mm); and the second group indicates the motor power (kW) and number of poles. The equipment listed above includes only those components directly involved in the modeling; other plant equipment, such as standby pumps and auxiliary devices, is excluded.

The large chiller is configured with two chilled

DOI 10.3384/ecp223379

water pumps due to hydraulic and efficiency considerations. The rated flow rate of the large chiller is 352 m^3/h. If a single pump of equivalent capacity were used, its operating point would deviate substantially from the high-efficiency region under low-load conditions, even with variable-speed operation, leading to a marked drop in efficiency. By employing two smaller pumps operating in parallel, one pump can be deactivated during part-load conditions while the other remains operational, thereby maintaining higher pump efficiency. Moreover, the system curve with two parallel pumps is flatter, which facilitates a wider flow regulation range and improved energy performance under variable-speed control.

The dynamic simulation model of the refrigeration plant was developed using Dymola 2023x and Modelica Buildings Library 10.1.0. The modeling approach for each primary component is as follows.

Chillers are modeled with `Buildings.Fluid.Chillers.ElectricEIR`, which captures off-design performance based on rated cooling capacity, *COP*, and part-load performance curves corresponding to the selected equipment.

Variable-speed pumps are modeled with `Buildings.Fluid.Movers.SpeedControlled_y`, with flow-head-power characteristic curves input according to the selected equipment.

Cooling towers are modeled with `Buildings.Fluid.HeatExchangers.CoolingTowers.YorkCalc`, which simulates the direct-contact heat and mass transfer between air and water based on the NTU-ε method; the fan operates with variable speed.

The piping system employs components such as `Buildings.Fluid.FixedResistances.PressureDrop` to represent frictional and local resistances.

The load side is represented by `Buildings.Fluid.HeatExchangers.HeaterCooler_u` serving as an equivalent heat exchanger, with the heat flow rate extracted from the chilled water system prescribed by an external hourly load data file.

Controllers are constructed using the PID, mathematical operation, and logical switch blocks from the `Modelica.Blocks` library.

The architecture of the complete simulation model is illustrated in the figure below.

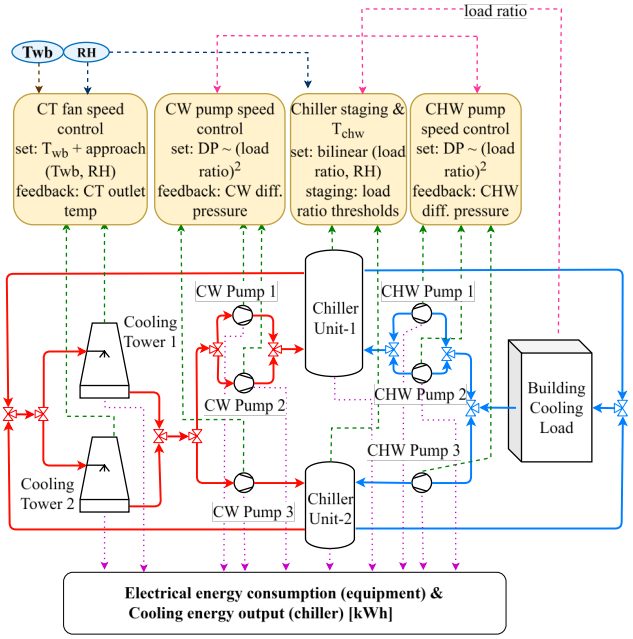


Figure 1. Schematic diagram of the proposed optimization-based control strategy and refrigeration plant model.

2.2 Control Strategy Design

The control strategy is designed to maintain high operational efficiency across the full range of load and meteorological conditions. To achieve this, the strategy is based on a set of continuous functions that dynamically determine the key control parameters—chilled water supply temperature, pump differential pressure setpoints, and cooling water approach temperature—based on real-time load and outdoor meteorological conditions. The functional forms are determined by system thermodynamic characteristics and piping resistance characteristics; however, the coefficients and thresholds within these functions cannot be readily derived through theoretical means alone. To overcome this limitation, the undetermined coefficients and thresholds are treated as adjustable parameters. With the objectives of maximizing the comprehensive energy efficiency ratio and minimizing total energy consumption over the cooling season, the LSHADE (Success History-based Adaptive Differential Evolution with Linear Population Size Reduction) algorithm is applied to identify the optimal combination of these

parameters. LSHADE is an improved differential evolution algorithm that adaptively adjusts its search strategy based on historical success information, rendering it suitable for global optimization problems in continuous parameter spaces. In the implementation, the Dymola model is exported as a Functional Mock-up Unit (FMU) in the Functional Mock-up Interface (FMI) format, which is then invoked and executed within a Python environment. Upon completion of the search, the resulting parameters are written back into the Dymola model for full-cooling-season simulation across all operating conditions. The specific details of the control strategy are presented below.

The load ratio l is defined as the ratio of the building's hourly cooling load to the total rated cooling capacity of the system:

$$l = \frac{Q_{\text{load}}}{Q_{\max}} \tag{1}$$

The load ratio ranges from 0 to 1 and represents the current cooling demand relative to the maximum capacity of the plant.

The number of possible operating condition combinations during the cooling season is substantial, and performing an individual optimization for each condition would incur prohibitive computational cost. Since the optimal control parameters vary continuously with load and meteorological conditions, the optimization results obtained at the boundaries of the operating domain can effectively constrain the parameter values for interior conditions. Consequently, precise optimization at boundary conditions, coupled with an interpolation algorithm, can encompass the entire operating range with a finite number of calculations (Liu et al., 2023).

Bilinear interpolation requires all input variables to be mapped to a common scale. The load ratio l is inherently a dimensionless quantity within the interval [0,1], whereas relative humidity and wet-bulb temperature are expressed as percentages and degrees Celsius, respectively—their numerical ranges differing markedly from that of the load ratio. To enable these quantities to function together within the interpolation formula, they must first be mapped onto the [0,1] interval via linear normalization (Zhou, 2024). The bounds of each normalization interval were determined from the distribution of meteorological parameters during the cooling season in Xiamen: the relative humidity interval for the chilled water side was taken as 60%–95%, corresponding to the predominant range of outdoor humidity variation during the cooling season; the relative humidity interval for the cooling water side was taken as 30%–90%, encompassing the sensitivity range over which evaporative cooling efficiency is influenced by humidity; and the wet-bulb temperature interval was taken as 19–28°C, corresponding to the typical range of wet-bulb temperature variation during the cooling season.

The chilled water humidity coefficient r is obtained by linear normalization of the outdoor relative humidity RH:

$$r = \min[\max(\frac{RH - 0.6}{0.35}, 0), 1] \tag{2}$$

The cooling water humidity coefficient y and the wet-bulb coefficient x are obtained by linear normalization of the outdoor relative humidity and wet-bulb temperature, respectively:

$$y = \min[\max(\frac{RH - 0.3}{0.60}, 0), 1] \tag{3}$$

$$x = \min[\max(\frac{T_{\text{wb}} - 19}{9}, 0), 1] \tag{4}$$

The four boundary conditions for the chilled water temperature correspond to the combinations of load ratio and humidity coefficient at their extreme values: low load with low humidity (l=0, r=0), low load with high humidity (l=0, r=1), high load with low humidity (l=1, r=0), and high load with high humidity (l=1, r=1). With the objectives of maximizing the comprehensive energy efficiency ratio and minimizing total energy consumption over the cooling season, the LSHADE algorithm was used to simultaneously determine the optimal chilled water supply temperatures at the four boundary conditions. The resulting values are 15.8°C, 12.2°C, 10.6°C, and 7.3°C, respectively. The supply water temperature for intermediate conditions is given by the following bilinear interpolation formula:

DOI
10.3384/ecp223379

$$T_{\mathrm{chw}} = 15.772(1-l)(1-r) + 12.247(1-l)r + 10.552l(1-r) + 7.320lr \quad (5)$$

The cooling water supply temperature setpoint is defined as the sum of the wet-bulb temperature and the approach temperature. The bilinear interpolation for the cooling water approach temperature takes the wet-bulb coefficient x and the cooling water humidity coefficient y as input variables. The four boundary conditions correspond to combinations of these two variables at their extreme values: low wet-bulb and low humidity (x=0, y=0), low wet-bulb and high humidity (x=0, y=1), high wet-bulb and low humidity (x=1, y=0), and high wet-bulb and high humidity (x=1, y=1). Following optimization with the LSHADE algorithm, the corresponding optimal approach temperatures were determined to be 1.9℃, 2.6℃, 3.2℃, and 3.6℃, respectively. The approach temperature for intermediate conditions is calculated using the following bilinear interpolation formula:

$$T_{\mathrm{app}} = 1.915(1-x)(1-y) + 2.551(1-x)y + 3.200x(1-y) + 3.637xy \quad (6)$$

The pump differential pressure setpoints are defined as quadratic functions of the load ratio l, based on the piping resistance characteristics. The coefficients correspond to the most unfavorable circuit resistance of the respective supply and return mains:

$$\Delta P_{\mathrm{chw_set}} = 86 \times l^2 \mathrm{kPa} \quad (7)$$

$$\Delta P_{\mathrm{cw_set}} = 139 \times l^2 \mathrm{kPa} \quad (8)$$

Equipment staging is governed by the load ratio thresholds obtained from the optimization, with start/stop decisions based on a comparison between the current load ratio and the threshold values. When both chillers operate, the load distribution is implemented by adjusting the split ratio of the three-way diverter valve according to the current load ratio. The cooling tower fan speed is modulated by a PID controller according to the deviation between the cooling tower outlet water temperature and its setpoint, whereas the pump speed is modulated by a PID controller according to the deviation between the measured differential pressure and its setpoint.

2.3 Baseline Strategy Configuration

To evaluate the performance of the proposed strategy, a baseline system was established. This baseline integrates the control approach of a high-efficiency plant in Tianjin (Wu et al., 2024) with conventional control methods commonly adopted in current engineering practice for high-efficiency plants. The integrated strategy is further adapted to the local meteorological conditions of Xiamen. The baseline system model employs exactly the same building hourly load profile, meteorological parameters, equipment parameters, and simulation duration as the proposed strategy model; the control logic constitutes the sole difference.

Table 2. Control parameter settings for the baseline strategy

Control Object	Control Strategy
Chilled Water Supply Temperature	Time-scheduled fixed values: Apr: 10℃; May–Jun: 9℃; Jul–Aug: 7℃; Sep: 9℃; Oct: 10℃; Nov: 11℃
Cooling Water Supply Temperature	Fixed approach temperature of 2.5℃
Chiller Operation	Small chiller operates alone under low load demand; large chiller starts when cooling load demand exceeds 837 kW, and stops when total cooling load demand falls below 542 kW during dual-chiller operation; load distribution during dual-chiller operation is proportional to rated capacity (large chiller 72%, small chiller 28%)

Table 2(Continued)

Control Object	Control Strategy
Cooling Tower Operation	PID variable-speed regulation of the fan based on the deviation between the cooling water supply temperature setpoint and the measured value
Chilled Water Pump Operation	Constant differential pressure of 30 kPa for the main, PID variable-speed regulation. Two chilled water pumps are paired with the large chiller and interlocked for synchronous start/stop; one chilled water pump is paired with the small chiller and interlocked for synchronous start/stop (5-minute delay for large unit and pump, 4-minute delay for small unit and pump)
Cooling Water Pump Operation	Constant differential pressure of 26 kPa for the main, PID variable-speed regulation. One cooling water pump starts when only the small chiller operates; two cooling water pumps start when both chillers operate; pumps start 2 minutes before the chiller and stop 2 minutes after the chiller

The baseline strategy emulates the equipment interlock control approach typical of current engineering practice, particularly the one-to-one interlocked start/stop of cooling water pumps with chillers. If independent equipment selection were performed according to this logic, pumps of different capacities would be specified for the large and small chillers. To ensure the fairness of the comparison, however, the baseline strategy in this study does not employ independent selection but instead adopts the identical equipment configuration as the proposed strategy, retaining only the interlock characteristics in the control logic. In contrast, the proposed strategy employs one dedicated cooling water pump for the small chiller and two parallel cooling water pumps for the large chiller, with staging control based on the load ratio threshold and no direct interlock with specific chillers. Consequently, the two strategies operate on the same equipment basis but differ in their control logic, forming the basis of the comparative study presented herein.

The architecture of the complete baseline simulation model is illustrated in the figure below.

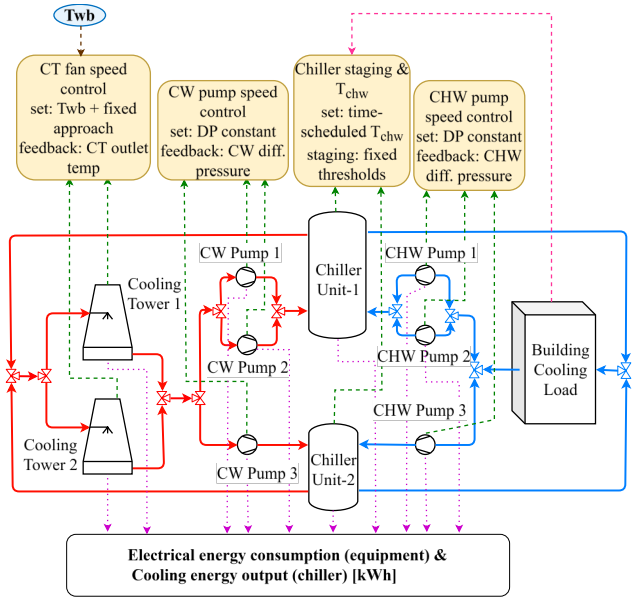


Figure 2. Schematic diagram of the baseline control strategy and refrigeration plant model.

3 Results

After completing the simulations in Dymola for both strategies, the resulting data were processed. To quantitatively evaluate the operational efficiency of the refrigeration plant, the comprehensive energy efficiency ratio (*EER*) and the chiller coefficient of performance (*COP*) served as the primary evaluation metrics.

The comprehensive energy efficiency ratio *EER* of the refrigeration plant is defined as:

$$EER = \frac{Q_{\text{total}}}{W_{\text{total}}} \tag{9}$$

The chiller coefficient of performance *COP* is defined as the ratio of the chiller's cooling capacity to its energy consumption:

DOI 10.3384/ecp223379

$$COP = \frac{Q_{\text{chiller}}}{W_{\text{chiller}}} \quad (10)$$

Following data processing, the key performance indicators for the refrigeration plant under the two strategies are summarized in Table 3.

Table 3. Comparison of simulation results: proposed strategy vs. baseline strategy

Comparison Item	Proposed Strategy	Baseline Strategy
EER	6.02	5.04
Total Energy Consumption / J	2.73×10^{12}	3.37×10^{12}
Total Cooling Capacity / J	1.65×10^{13}	1.70×10^{13}
Large Chiller Average COP	7.76	7.57
Small Chiller Average COP	5.16	4.89
Large Chiller Energy Consumption / J	1.59×10^{12}	1.44×10^{12}
Small Chiller Energy Consumption / J	8.04×10^{11}	1.24×10^{12}
Cooling Tower Energy Consumption / J	5.12×10^{10}	1.88×10^{11}
Cooling Water Pump Energy Consumption / J	1.73×10^{11}	2.61×10^{11}
Chilled Water Pump Energy Consumption / J	1.18×10^{11}	2.34×10^{11}

The comprehensive energy efficiency ratio of the proposed dynamic strategy is 6.02, compared to 5.04 for the baseline strategy, corresponding to a reduction in total energy consumption of approximately 19%. Although the baseline strategy already incorporates conventional high-efficiency measures—including approach temperature control for the cooling water and constant differential pressure variable-frequency control—the proposed strategy produces additional energy savings through the continuous adjustment of control parameters.

In terms of component-level energy consumption, the pumps constitute the largest share of the savings. Under the baseline strategy, both the chilled water and cooling water pumps operate with constant differential pressure control. At low loads, the piping resistance decreases while the differential pressure setpoint remains unchanged, requiring the pumps to maintain higher speeds than necessary to meet the setpoint, resulting in additional energy losses. In contrast, the proposed strategy links the differential pressure setpoint to the square of the load ratio. At low loads, the setpoint is reduced accordingly, and the pump speed decreases proportionally, leading to significant reductions in the energy consumption of both the chilled water and cooling water pumps. For the chillers, the average COP of the large chiller increased from 7.57 to 7.76 under the proposed strategy, while its energy consumption remained nearly unchanged. This indicates that the continuous variable water temperature control enables the large chiller to operate in a more efficient region for a comparable energy input. The average COP of the small chiller increased from 4.89 to 5.16, with a concurrent reduction in energy consumption, owing to the dynamic load distribution strategy, which reduces its operating time in less efficient operating ranges. The cooling tower energy consumption was generally reduced under the proposed strategy because the adaptive approach temperature control avoids unnecessary fan operation under favorable wet-bulb conditions. Meanwhile, the adjustment of approach temperature reflects the trade-off between improved chiller efficiency and additional fan power requirements under certain operating conditions. From a system-level perspective, the proposed strategy replaces discrete setpoints with functional relationships that vary continuously with operating conditions, allowing the cooling source and distribution equipment to operate near their optimal operating points throughout the cooling season. This results in a substantial reduction in system energy consumption and a marked improvement in efficiency.

4 Discussion

The optimal control parameters derived in this

study were calibrated for Xiamen, a representative city in the hot-summer warm-winter climate zone. When applying the proposed strategy to other climate zones—such as cold regions, where lower wet-bulb temperatures provide greater evaporative cooling potential, or hot-dry regions, where low humidity alters the energy–water trade-off—the boundary conditions of the optimization must be redefined and the parameter values re-calibrated. Nevertheless, the core methodology—coupling EnergyPlus-based load calculation, Dymola-based dynamic simulation, and LSHADE-based parameter optimization—remains applicable to any site by substituting local weather data and load profiles.

The evaporative cooling tower may introduce additional makeup water consumption due to extended operation under some transitional conditions. However, water consumption was not included in the current optimization objective because this study focuses on electrical energy performance. Future work will incorporate water-energy coupling into a multi-objective optimization framework.

Additional limitations include: the analysis is based on a single building case in Xiamen; the cooling load was pre-computed via EnergyPlus rather than coupled dynamically with the plant model; and only control parameters were optimized, excluding structural system modifications such as thermal storage integration or equipment resizing.

5 Conclusions

This paper developed a full-system dynamic simulation model of a high-efficiency refrigeration plant based on Dymola environment, and proposed a dynamic control strategy driven by hourly cooling load and outdoor meteorological parameters. The key control parameters—including chilled water supply temperature, cooling water approach temperature, pump differential pressure setpoints, and equipment staging thresholds—were optimized using the LSHADE algorithm to achieve coordinated operation across all conditions. The simulation results demonstrate that the dynamic control strategy significantly reduces pump energy consumption and improves chiller efficiency under part-load conditions, which increases the comprehensive energy efficiency ratio during the cooling season from 5.04 to 6.02. While the specific parameters are calibrated for Xiamen, the simulation-based optimization workflow is applicable to other locations and building types—including airports and shopping malls—by substituting local weather data and load profiles.

References

Jiang Yating, Zhu Han, Li Zhengrong. Study on Energy Saving Degree of Subway Environmental Control Strategies Based on Dymola [J]. *HV&AC*, 2024.

Liu Xuefeng, Ma Wenjing, Zeng Deqiang, He Juan. A Central Air-Conditioning Intelligent Control Method, System, and Storage Medium [P]. China: CN116972516A, 2023-10-31.

Mattsson S E, Elmqvist H. Modelica – An international effort to design the next generation modeling language [J]. *IFAC Proceedings Volumes*, 1997, 30(4): 151-155.

Wetter M, Zuo W, Nouidui T S, et al. Modelica buildings library [J]. *Journal of Building Performance Simulation*, 2014, 7(4): 253-270.

Wu Chen, Li Xiaoman, Wang Yunping. Design and Discussion of a High-Efficiency Refrigeration Plant in Tianjin [J]. *HV&AC*, 2024.

Xu Wei. *Research Report on the Development of High-Efficiency Air-Conditioning Refrigeration Plants in China (2021)* [M]. Beijing: China Architecture & Building Press, 2022.

Yang Muhe, Zeng Gang, Xu Huan. Design and Case Analysis of High-Efficiency Refrigeration Plant Systems [J]. *Shanghai Gas*, 2024(4): 1-5.

Zhou Peng. Research on Modeling and Control Strategy of HVAC Refrigeration Systems [J]. *China New Technologies and New Products*, 2024(07): 78-80.

DOI
10.3384/ecp223379

Coordinated Pressure Control Strategy Integrating Feedforward Compensation and Integral Freezing for Biosafety Laboratories under Transient Door Operation Disturbances

Yuhui Wang[1], Hejiang Sun[1]

[1]School of Environmental Science and Engineering, Tianjin University, Tianjin 300072, China

*Corresponding author: sunhe@tju.edu.cn

Abstract

Safe operation of biosafety laboratories requires directional airflow and stable pressure gradients to contain pathogenic aerosols. However, door operations between the core and buffer rooms frequently disrupt this balance, inducing transient pressure disturbances. Conventional PI control often responds inadequately to such rapid transients, which can lead to airflow reversal and compromise biosafety. This study develops a Modelica-based two-zone transient differential pressure regulation model for the core and buffer rooms and proposes a coordinated control strategy that integrates feedforward compensation with an integral-freezing mechanism. During door opening, the feedforward mechanism adjusts the exhaust airflow, rapidly increasing the doorway air mass flow rate to 0.12–0.14 kg/s. This establishes a directional airflow barrier (velocity ≥ 0.2 m/s), thereby reducing the risk of leakage during personnel transit. By freezing the PI integral term during the door-open phase and applying feedforward exhaust compensation upon closure, the proposed strategy restricts the maximum pressure undershoot during door closure to −17 Pa (against the −15 Pa design setpoint), compared to −24 Pa under conventional PI control. The pressure recovery time is also reduced from 48.3 s to 13.6 s. These findings provide a simulation-based basis for improving the dynamic containment of high-level biosafety laboratories under transient disturbances.

Keywords: *Biosafety laboratory; Differential pressure control; Feedforward compensation; Integral freezing*

1 Introduction

As core facilities for handling highly pathogenic microorganisms within the public health system, biosafety laboratories rely on maintained cascaded negative pressure gradients to prevent the outward leakage of pathogenic aerosols. While steady-state differential pressure control has become relatively mature, the door opening and closing operations during personnel entry and exit can trigger severe transient pressure disturbances. These disturbances can disrupt the directional airflow, constituting a significant vulnerability in dynamic containment. In practice, buffer rooms and airlocks are used as transitional spaces to maintain pressure separation and airflow direction between controlled and adjacent areas; however, door operation can cause a temporary loss of pressure differential and increase the risk of airborne cross-contamination (Sun, 2018).

Existing literature (Xu et al., 2016) has demonstrated the adverse effects of door operations on laboratory airflow patterns. When the door of an isolation room or a negative pressure ward is opened, the pressure difference across the doorway drops rapidly to 0 Pa within 1 to 2 seconds. Similarly, other studies (Villafruela et al., 2016) have shown that the opening of a door temporarily eliminates the pressure barrier, leading to a rapid air exchange between the two sides. Such transient fluctuations in differential pressure can easily induce containment failure, allowing contaminated air to leak through the doorway into adjacent clean zones or corridors.

To address this dynamic control challenge, recent studies (Tang et al., 2026) have employed Modelica-based multivariable control to manage spatial pressure coupling in BSL-3 laboratories. Nevertheless, traditional PI control, equivalent to a PID controller with the derivative term set to zero, remains predominant in engineering practice, despite facing two primary limitations under transient door disturbances. The first is response lag. PI passively adjusts the exhaust valve only after the pressure drops, providing insufficient initial airflow to prevent contaminants from leaking outward in the wake of personnel movement. The second limitation involves integral windup and door-closing pressure undershoot. Since the differential pressure cannot reach the setpoint while the door remains open, the integral term continuously accumulates error, causing the valve to over-open. Upon sudden closure, this excessive exhaust airflow instantly triggers a severe negative pressure undershoot, increasing the mechanical load on the laboratory envelope (e.g., ceilings and sandwich panels). Furthermore, from a control system perspective, traditional PI struggles with such strong disturbances, inducing coupled oscillations in adjacent rooms and substantially prolonging the recovery and stabilization time of the system's differential pressure (Jeremić et al., 2025).

To overcome these limitations, this study proposes a coordinated feedforward and PI integral-freezing strategy using a Modelica-based dual-room model. The strategy manages transient disturbances by: (1) adjusting exhaust flows to build an airflow barrier when the door opens; (2) freezing the PI integral to prevent windup while open; (3) using feedforward compensation to minimize pressure fluctuations during closure. Dynamic simulations indicate improved disturbance-rejection performance over traditional PI for dynamic containment. The novelty of this study lies in the coordination of disturbance anticipation through feedforward compensation and integral-state management through integral freezing, rather than applying either method independently.

2 Research Methods

The two-zone model (core room and buffer room) uses a constant supply and variable exhaust system. This system maintains a stepped negative pressure gradient to prevent aerosol leakage. During steady-state operation, the flow rate difference between the fixed supply and variable exhaust sustains this gradient. When door operations drop the differential pressure, a doorway velocity below 0.2 m/s significantly increases the risk of the escape of pathogen-laden droplet nuclei (Hang et al., 2015). Therefore, the controller must switch from regulating static differential pressure to controlling transient velocity—instantly creating an airflow barrier.

2.1 System Modeling

This study investigates a BSL-3 laboratory in Fuzhou, with all parameters set to their design values. The buffer room maintains a −40 Pa differential pressure against the atmosphere, while the core room maintains −15 Pa relative to the buffer room (an absolute −55 Pa). Using Dymola 2023x and the Modelica Buildings Library V12.1.0, this study constructs a PI-based dynamic differential pressure regulation baseline model to evaluate the coordinated differential pressure control strategy. The core modeling mechanisms are:

Multi-zone and boundary modeling: The core (55 m³) and buffer (11.5 m³) rooms utilize the lumped-parameter model (`Buildings.Fluid.MixingVolumes.MixingVolume`) to calculate transient pressure changes. The supply terminals employ ideal constant flow sources (`Buildings.Fluid.Movers.FlowControlled_m_flow`) to deliver constant air supplies of 0.36 kg/s and 0.09 kg/s, respectively.

Transient doorway disturbance modeling: The model applies an operable inner door component (`Buildings.Airflow.Multizone.DoorDiscretizedOperable`, 0.8 m × 2.1 m, flow coefficient = 0.65, flow exponent = 0.5) to compute dynamic infiltration and air jets. This study represents the door-opening disturbance as a trapezoidal-wave driven flow-area variation function (3 s opening, 2 s fully open, 3 s closing).

Ductwork resistance modeling: The simulation uses the `Buildings.Fluid.FixedResistances.PressureDrop` component to represent frictional and local resistance within the supply and exhaust ductwork.

Control and actuator response delay modeling: The control system utilizes a limiting controller (`Buildings.Controls.Continuous.LimPID`) with differential pressure feedback signals as inputs. For precise BSL-3 pressure control, engineers typically employ PI control, omitting the derivative term (Åström & Hägglund, 2006). Based on the Ziegler-Nichols (Z-N) ultimate gain method, this study tunes the PI parameters, as listed in Table 1:

Table 1. Parameter settings of the baseline PI controllers for the multi-zone system

Room	k	τ_i	τ_d
Core Room	0.0005	20	0
Buffer Room	0.001	30	0

High-level biosafety laboratories typically employ fast variable air volume (VAV) dampers to manage transient airflow disturbances. Following the critical control system response time specifications in the *ASHRAE Laboratory Design Guide* (McIntosh et al., 2001), this section accounts for damper delay by cascading a first-order inertia element with 2 s between the PI output and the exhaust airflow command. This setup simulates the combined response delay of the fast VAV damper and the ductwork system.

The complete simulation model architecture is as follows:

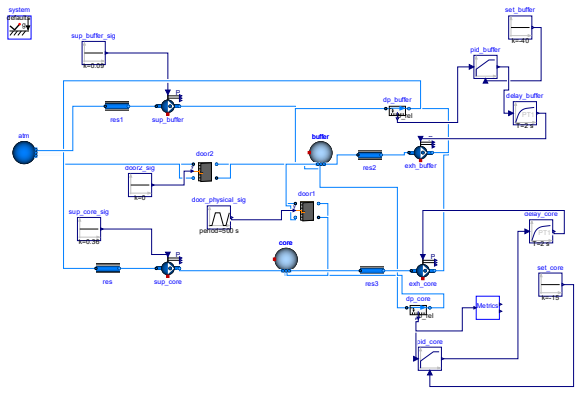


Figure 1. Schematic diagram of the simulation model architecture.

2.2 Control Strategy

In dynamic pressure control for biosafety laboratories, PI feedback control struggles to manage large door-operation disturbances due to physical door movement and VAV

DOI 10.3384/ecp223387

actuator response delay. To prevent contaminant leakage and pressure undershoot upon door closure, this study proposes a coordinated pressure control strategy combining staged feedforward compensation and state-switching-based integral freezing. The control logic consists of four steps: door-state detection, activation of feedforward compensation and integral freezing during door opening, compensation adjustment during door closing, and transition back to PI feedback control after complete closure.

The system formulates the theoretical exhaust airflow control command $u(t)$ as a linear superposition of the PI feedback output $u_{PI}(t)$ and the feedforward compensation output $u_{FF}(t)$:

$$u(t) = u_{PI}(t) + u_{FF}(t) \tag{1}$$

The physical limits of the VAV exhaust fan—specifically its maximum capacity and minimum safe airflow—constrain the final actuator command $\tilde{u}(t)$ via a saturation function:

$$\tilde{u}(t) = sat(u(t)) = \begin{cases} u_{\max}, & if\ u(t) > u_{\max} \\ u(t), & if\ u_{\min} \le u(t) \le u_{\max} \\ u_{\min}, & if\ u(t) < u_{\min} \end{cases} \tag{2}$$

where sat denotes the actuator saturation function, while $u_{\max}$ and $u_{\min}$ represent the maximum allowable and minimum safe exhaust airflows, respectively.

2.2.1 Coordinated Feedforward Strategy

Feedforward control reduces the effect of system response delay via anticipatory disturbance compensation. Given the deterministic sequence of door operations (opening–holding–closing), the strategy divides the feedforward compensation for both rooms into three phases.

Upon door opening, the core room increases exhaust while the buffer room synchronously reduces exhaust. This generates a coordinated push-pull airflow effect to rapidly establish a directional airflow at the doorway. Based on orifice flow principles, an integrated equation calculates the theoretical transient feedforward compensation airflow $u_{FF}(t)$ required to build this directional airflow:

$$u_{FF}(t) = \rho \cdot v_t \cdot (\alpha \cdot A_d - A_p) \tag{3}$$

where air density $\rho \approx 1.2\ kg/m^3$ and target velocity $v_t = 0.2\ m/s$. Incorporating the half-open door state during transit (opening coefficient $\alpha \approx 0.5$) and dynamic human blockage (frontal projection area $A_p \approx 0.45\ m^2$) alongside the doorway area A_d, the calculation yields a baseline airflow requirement of approximately 0.09 kg/s. Factoring in ductwork losses and safety margins, the implemented airflow transfer is set to approximately 0.13 kg/s.

During the fully open period, the controller maintains a high-level feedforward command, allowing open-loop feedforward to dominate the system flow field.

As the door enters its final closing stroke, the core room feedforward signal is switched to a negative compensation command to reduce exhaust, while the buffer room feedforward shifts positive. This action reduces pressure oscillations triggered by sudden flow variations. Upon complete door closure, the feedforward compensation $u_{FF}(t)$ drops to zero, and the system smoothly transitions back to closed-loop PI control.

2.2.2 Integral Freezing Algorithm

During the unsteady door-opening phase, the interconnected spaces cause the differential pressure to decay rapidly toward zero, driving the system away from the target setpoint. A conventional PI controller continuously accumulates this error within its integral term. Upon door closure, the accumulated integral action forces the exhaust damper wide open, triggering a large negative pressure undershoot (Åström & Hägglund, 2006). Drawing on conditional integral anti-windup mechanisms from control theory, the controller uses a magnetic door switch signal to assess the physical door state in real time. By outputting a discrete switching signal via magnetic induction, the magnetic switch rapidly and precisely captures door movements, bypassing the delays associated with complex physical opening calculations. The control logic operates as follows: when the door opens and connects the two air volumes, the magnetic switch disconnects. The controller then sets the state function $S(t)$ to 1, registering an open door. Upon complete closure, the magnetic switch engages, and the controller resets $S(t)$ to 0, registering a closed door.

The controller formulates the output $u_{PI}(t)$ incorporating integral freezing logic as:

$$u_{PI}(t) = K \cdot e(t) + \frac{K}{T_i}\int_0^t \mathrm{e}\,(\tau)d\tau \tag{4}$$

which introduces a controlled conditional error variable $e(t)$:

$$\mathrm{e}(t) = [1 - S(t)] \cdot [p_{set} - p_m(t)] \tag{5}$$

where K and T_i denote the proportional gain and integral time constant, respectively; p_{set} represents the target differential pressure setpoint, while $p_m(t)$ indicates the actual differential pressure continuously measured by the sensor.

Based on Equation (5), the PI controller executes standard feedback regulation during normal operation ($S(t) = 0$). Upon door opening ($S(t) = 1$), the logic switch disconnects the sensor feedback channel and substitutes the actual measurement with the target setpoint. Consequently, the controller receives a zero-error value. This mechanism effectively freezes the integral action and locks the controller output at the pre-disturbance steady-state value. Following door closure, the logic switch

reconnects the actual differential pressure signal. The system then resumes control from an unsaturated state, mitigating overshoot oscillations and shortening the recovery time.

The system modeling framework incorporating this coordinated differential pressure control strategy operates as follows:

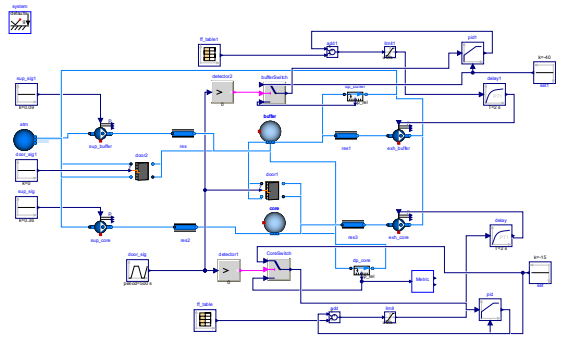


Figure 2. Schematic of the system modeling framework incorporating the coordinated control strategy.

2.2.3 Dynamic Performance Metrics

To quantitatively evaluate the transient disturbance rejection capabilities of the coordinated differential pressure control strategy, this study selects three dynamic performance metrics for systematic comparison:

(1) Maximum Pressure Excursion

This metric encompasses Least Negative Pressure (LNP) and Maximum Negative Pressure Undershoot (MNPU). High-level biosafety laboratories strictly prohibit transient positive pressures (> 0 Pa) to reduce contaminant leakage risks. Simultaneously, the control system must limit the negative pressure undershoot at the instant of door closure to reduce damage to the envelope structure and equipment.

(2) Recovery Time (T_{rec})

This metric measures the time required for the system differential pressure to recover from the moment of complete door closure and remain within the target setpoint error band of ±1 Pa. It evaluates the system's disturbance suppression speed.

(3) Comprehensive Error Metrics

To quantify the control deviation and attenuation capability throughout the door-closing disturbance cycle, this section introduces the Integral of Absolute Error (IAE) and the Integral of Time-weighted Absolute Error (ITAE) as comprehensive evaluation benchmarks. IAE quantifies the total accumulation of control deviation during the entire disturbance period. ITAE penalizes long-term residual oscillations during the late recovery phase via time weighting:

$$IAE = \int_0^T |\varepsilon(t)|dt \tag{6}$$

$$ITAE = \int_0^T t|\varepsilon(t)|dt \tag{7}$$

Where $\varepsilon(t)$ denotes the real-time deviation between the actual measured differential pressure and the target setpoint, while$T$represents the total disturbance regulation time. This study sets the exact start of the door-closing process as 0 s and limits the total duration T to 20 s.

3 Results

To evaluate the effectiveness of the proposed strategy, this paper compares the dynamic responses of traditional PI, PI with feedforward but without integral freezing (PI-FF), and the coordinated control strategy in Dymola during a door operation triggered at 400 s.

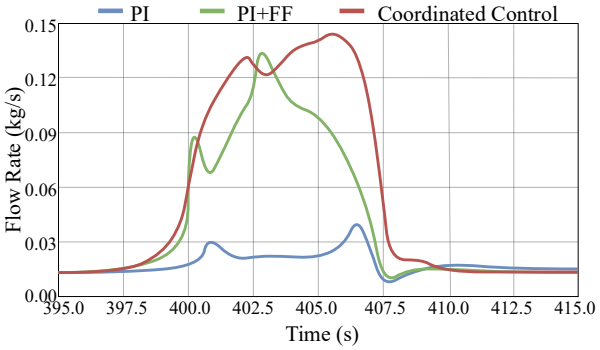


Figure 3. Transient responses of air mass flow rate at the doorway cross-section

Figure 3 illustrates the air mass flow rate variations at the doorway cross-section. During the initial door-opening phase, actuator response delay and passive feedback characteristics restrict the traditional PI flow peak below 0.04 kg/s, which is insufficient to establish an effective directional airflow barrier. Conversely, driven by anticipatory feedforward commands and the multi-zone "push-pull" mechanism, the PI-FF control and coordinated control strategies rapidly increase the mass flow rate to 0.12–0.14 kg/s during the opening phase. This establishes a directional airflow barrier before personnel transit.

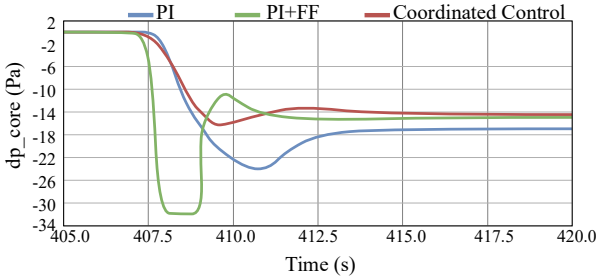


Figure 4. Transient responses of differential pressure in the core room

Figure 4 presents the core room differential pressure relative to the buffer room (setpoint: −15 Pa), demonstrating how integral freezing suppresses negative

DOI
10.3384/ecp223387

pressure undershoot. At the instant of door closing, traditional PI produces a large negative pressure undershoot of −24 Pa alongside a prolonged recovery period. Meanwhile, continuous error accumulation during the door-open phase causes substantial integral saturation of the PI-FF controller, triggering a −32 Pa undershoot upon closure. In contrast, the coordinated strategy utilizes the state-switching mechanism to freeze the integral action during the open phase. This mechanism successfully restricts the maximum negative undershoot to −17 Pa and drives the pressure back to the −15 Pa setpoint following door closure.

To precisely quantify the overall disturbance rejection capabilities of the three strategies, Table 2 details the core dynamic performance metrics for each condition.

Table 2. Performance metrics for various control methods

Method	LNP (Pa)	MNPU (Pa)	T_{rec} (s)	IAE	ITAE
PI	-17	-24	48.3	74.6	249.3
PI-FF	-11	-32	10.9	56.9	119.6
Proposed	-14	-17	13.6	26.6	63.3

Note: Proposed denotes the coordinated feedforward and integral-freezing control strategy.

As illustrated in Table 2, coordinated control (13.6 s) yields a slightly longer recovery time than the PI-FF control (10.9 s), but it substantially limits the negative pressure undershoot at the instant of door closure. Although PI-FF achieves a faster recovery, the absence of integral freezing results in excessive actuator adjustment after door closure. The proposed strategy therefore provides a better balance between recovery speed and pressure stability. This improvement is attributed to the complementary roles of the two coordinated mechanisms. The feedforward compensation provides rapid disturbance anticipation during door opening, while the integral freezing mechanism prevents accumulated integral error from causing excessive corrective actions during door closure. Therefore, the proposed strategy achieves faster disturbance rejection than conventional PI control while maintaining better pressure stability than PI-FF. This restricts the maximum negative undershoot to −17 Pa, balancing response speed with system safety. Substantially lower comprehensive error metrics (IAE and ITAE) demonstrate the strategy's effectiveness in suppressing system oscillations and minimizing residual error. The core room avoids positive pressure reversal throughout the disturbance cycle, satisfying the biosafety requirement to prevent aerosol escape.

4 Conclusions

This study develops a Modelica-based dynamic pressure regulation model and proposes a coordinated control strategy integrating feedforward compensation with integral freezing. Simulation results demonstrate that this strategy rapidly establishes a directional airflow barrier upon door opening to reduce leakage risk. Furthermore, the strategy limits pressure undershoot during door closure and accelerates a smooth recovery to the steady state, thereby enhancing the dynamic containment performance of the system. Compared with conventional PI, the proposed strategy reduced the recovery time from 48.3 s to 13.6 s and limited the closing undershoot to −17 Pa. This research provides a potential control solution for improving the dynamic safety of high-level biosafety facilities.

The current study focuses on a representative two-zone biosafety laboratory configuration with predefined door-operation patterns, constant supply airflow, and simplified airflow boundaries. Future work will consider variable door operations, multi-zone pressure interactions, and advanced control methods such as Model Predictive Control (MPC) to further improve the robustness and applicability of the proposed strategy.

Acknowledgements

The authors acknowledge the support provided by the School of Environmental Science and Engineering, Tianjin University, during the preparation of this work.

References

Åström, Karl Johan and Tore Hägglund (2006). *Advanced PID Control*. ISA - The Instrumentation, Systems and Automation Society. ISBN: 978-1-55617-942-6.

Hang, Jian, Yuguo Li, W. H. Ching, Jianjian Wei, Ruiqiu Jin, Li Liu, and Xiaojian Xie (2015). "Potential airborne transmission between two isolation cubicles through a shared anteroom". In: *Building and Environment* 89, pp. 264–278. DOI: 10.1016/j.buildenv.2015.03.004.

Jeremić, Branislav M. and Aleksandar Ž. Rakić (2025). "Multivariable Model Predictive Control of Cleanroom Pressure Cascades". In: *Electronics* 14.16, 3296. DOI: 10.3390/electronics14163296.

McIntosh, Ian B. D., Chad B. Dorgan, and Charles E. Dorgan (2001). *ASHRAE Laboratory Design Guide*. Atlanta: American Society of Heating, Refrigerating and Air-Conditioning Engineers. ISBN: 978-1-883413-97-2.

Sun, Wei (2018). "Cleanroom Airlock Performance and Beyond". In: *ASHRAE Journal* 60.2, pp. 64–69.

Tang, Jinhui, Hejiang Sun, Rong Wang, and Junjie Liu (2026). "Investigation of dynamic pressure differential control in BSL-3 laboratory based on Full-System modeling". In: *Energy and Buildings* 359, 117286. DOI: 10.1016/j.enbuild.2026.117286.

Villafruela, José Manuel, Julio Francisco San José, Francisco Castro, and Andrés Manuel Zarzuelo (2016). "Airflow patterns through a sliding door during opening and foot traffic in operating rooms". In: *Building and Environment* 109, pp. 190–198. DOI: 10.1016/j.buildenv.2016.09.025.

Xu, Zhonglin and Bin Zhou (2016-12). “Three Misunderstandings for Design of Negative Pressure Ward”. In: *Dynamic Isolation Technologies in Negative Pressure Isolation Wards*. Springer, Singapore, pp. 13–49. DOI: 10.1007/978-981-10-2923-3_2.

Simulation and Experimental Verification of Temperature Rise in an Oil-Cooled Motor Based on Amesim

Cunhao Zhang[1] Shuo Cheng[1*] Yanfang Liu[1] Peng Dong[1] Shuhan Wang[1] Xiangyang Xu[1]

[1]School of Transportation Science and Engineering , Beihang University, China

*Corresponding author: S. Cheng, cshuo@buaa.edu.cn

Abstract

With the development of electric drive systems for electric vehicles toward high power density and integration, electric drive assemblies integrating the drive motor, motor controller, and reducer have attracted lots of concerns. Under the combined effect of multiple heat sources within a compact space, the system heat dissipation of an electric drive system remains an open challenge, so it is necessary to analyze the temperature rise characteristics inside the motor. This paper examines in detail the distribution of internal losses in the motor and calculates the thermal resistances of the motor components. Then, a thermal–hydraulic coupled simulation platform on the Amesim platform for predicting the temperature rise of oil-cooled motors is developed based on the lumped-parameter thermal network method. A bench test is established to verify the accuracy of the simulation model. Experimental results show that the maximum error is 5.53%. This study provides a reference for the thermal analysis of motors in electric drive assemblies.

Keywords: Electric drive assembly, temperature rise characteristics, lumped parameter thermal network method, bench test

1 Introduction

With the continuous advancement of electric vehicle technology, enhancing the power density and integration level of electric drive systems has become a prominent development trend.[1] The electric drive assembly, which integrates the drive motor, motor controller, and reducer, enables higher power density and lightweight design in electric vehicles. However, it also introduces greater challenges for system heat dissipation. The high degree of integration concentrates heat generation from multiple power components, leading to severe cooling conditions. To prevent irreversible damage caused by excessive drive motor temperature, which would affect the operational stability and safety of the electric drive system, it is necessary to analyze the motor temperature rise characteristics and their influencing laws.

Unlike conventional water cooling, which only dissipates heat peripherally through water jackets, oil cooling possesses favorable insulation and thus can directly contact motor components. It enables targeted cooling of key parts and exhibits higher adaptability to highly-integrated electric drive assemblies. KH Lee[2] et al. adopted housing cooling channels and a hollow shaft to cool the rotor, with coolant circulating throughout the motor from the housing to the rotor, which can achieve superior cooling performance. Myeong Hyeon Park[3] et al. designed a dedicated internal oil jet cooling structure for in-wheel motors. Tanguy Davin[4] et al. investigated the external pipeline oil injection scheme, where cooling oil is sprayed onto the stator outer surface and winding end turns. The influences of flow rate, rotational speed, and oil temperature on cooling performance were analyzed. The results indicate that the cooling effect is most sensitive to flow rate and least affected by rotational speed, which only contributes to local temperature differences.

For high-power-density motors, thermal analysis is critical to the operational stability of the complete machine. South Korean scholars HA[5] et al. analyzed the oil flow field based on a self-developed 160 kW motor and obtained the distribution data of oil outflow. Furthermore, they investigated the heat transfer characteristics inside coils and the stator, and clarified the effects of oil temperature and flow rate on the motor temperature distribution. Fasquelle[6] et al. analyzed the correlation between the cooling system and multiple factors, including ambient conditions, cold plate layout distance, and coolant flow rate. Based on computational fluid dynamics and the finite element method, numerical simulations were conducted to investigate the temperature distribution and flow characteristics. Hu[7] et al. proposed a novel hybrid cooling system that integrates the advantages of traditional radial and axial cooling structures. This design simplifies the ventilation structure while achieving prominent cooling performance. Prototyping experiments based on finite element simulation verify the effectiveness of the proposed cooling system.

Most existing studies focus on the thermal simulation analysis of individual subsystems[8], while investigations into the temperature rise characteristics of oil-cooled motors within highly integrated electric drive assemblies remain relatively limited. This study takes an integrated electric drive system as the research object. A thermal network model, a loss distribution calculation model, and an oil cooling pipeline model for an oil-cooled motor are established on the Amesim platform. The lumped-parameter thermal network model is further modified based on bench test data.

2 Loss Analysis and Thermal Resistance Parameter Calculation

2.1 Basic Information of the Electric Drive Assembly

The research object of this paper is an electric drive assembly system for passenger cars. The main components, including the resolver, motor, and reducer, are drawn into a schematic diagram of the electric drive assembly as shown in Figure 1. The blue pipeline in the figure represents the cooling water circuit, which is responsible for cooling the resolver. The orange pipeline represents the cooling oil circuit, which is the main cooling method for the motor. After passing through the oil pump, oil filter, and heat exchanger, the oil is sprayed onto the motor stator surface and windings through three oil injection nozzles.

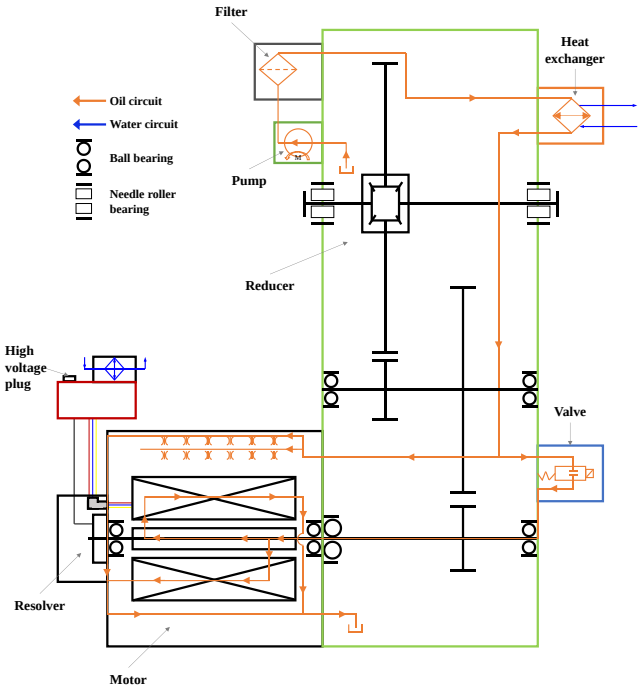


Figure 1. Assembly Structure Diagram.

The main parameters of the motor are listed in Table 1.

Table 1. Main Parameters of the experimental Motor.

Parameters	*Value*
Rated power / kW	70
Rated torque / N·m	150
Peak power / kW	150
Peak torque / N·m	310
Stator diameter/mm	105
Rotor diameter/mm	71
Number of stator slots	72
Number of rotor poles	8

2.2 Loss Analysis

During motor operation, losses occur in the conversion efficiency, and this portion of energy is released in the form of heat through heat exchange with the external environment or internal heat transfer. Such heat constitutes the fundamental cause of motor heating. To research motor thermal management, it is necessary to analyze the heat generation mechanism of each part of the motor. The main heat-generating components, namely the loss-generating parts of the motor, include the windings, stator, rotor, and permanent magnets. The primary loss sources consist of winding copper loss, stator iron loss, rotor permanent magnet eddy current loss, and bearing mechanical loss. Minor factors such as friction losses in other components and some stray losses are relatively small in magnitude and are not considered in this paper. The loss of power of the main heat-generating components is calculated based on theoretical formulas, laying a foundation for the overall temperature field simulation of the motor.

The calculation formula for winding copper loss P_{Cu} is as follows:

$$P_{Cu} = mI^2R = m\left[\frac{i_{\max}}{\sqrt{2}}\right]^2 R \tag{1}$$

Where m is the number of phases. As a three-phase AC machine, the permanent magnet synchronous motor takes $m = 3$; I is the RMS current; R is the resistance of the motor winding; $i_{\max}$ is the peak current.

During motor operation, the winding continuously generates heat, resulting in a temperature rise. Since the winding resistance is significantly affected by temperature, the temperature variation must be taken into account.

The formula for calculating the winding resistance considering temperature effects is as follows:

$$R = R_0[1+\alpha_0(T-T_0)] \tag{2}$$

Where R is the winding resistance at the target temperature T, R_0 is the winding resistance at ambient temperature T_0, and α_0 is the temperature coefficient of the winding, which is taken as 4.25×10^{-3}(1/℃) in this paper.

The stator core loss P_{Fe} mainly consists of hysteresis loss P_H, eddy current loss P_C, and additional loss P_E, and its calculation formula is as follows:

$$P_{Fe} = P_H + P_C + P_E \tag{3}$$

The eddy current loss in permanent magnets is caused by harmonics generated from the variation of magnetic permeability due to slotting in the stator or rotor of the motor. In accordance with the characteristics of permanent magnets, the expression for permanent magnet eddy current loss is summarized as follows:

$$P_{pm} = \int \frac{J^2}{\sigma} dV \tag{4}$$

Where J is the eddy current density of the permanent magnet, σ is the electrical conductivity of the permanent magnet, and V is the volume of the permanent magnet.

The calculation formula for the friction loss P_f of deep groove ball bearings is as follows:

$$P_f = 0.15 \frac{F}{d} v \times 10^{-5} \tag{5}$$

Where F is the bearing load, d is the distance from the center of the bearing roller to the bearing center, and v is the circumferential speed of the roller center.

2.3 Thermal Resistance Parameter Calculation

In thermal network modeling, the accurate calibration of thermal resistance directly affects the prediction accuracy of the temperature field. The thermal resistance value reflects the heat dissipation efficiency of components: the lower the thermal resistance, the better the heat dissipation performance. The thermal resistance relationship between nodes directly determines the local temperature rise. Therefore, accurate calculation of the thermal resistance of motor components is particularly important for establishing an equivalent thermal network model.

A hollow cylinder is a basic shape found in electric motors. Most motor components, such as the housing, stator, rotor, etc., can be regarded as or approximated to hollow cylinders. Therefore, it is necessary to derive the calculation formula for the thermal resistance of a hollow cylinder.

The geometric parameters are shown in Figure 2, and the calculation formula for the thermal resistance of a hollow cylinder is as follows:

$$R = \frac{\delta}{\lambda A} = \int_{r_i}^{r_o} \frac{1}{\lambda l \cdot 2\pi r} dr = \frac{1}{2\pi\lambda l} \ln\left(\frac{r_o}{r_i}\right) \tag{6}$$

Where r_i is the inner radius of the hollow cylinder, r_o is the outer radius of the hollow cylinder, l is the axial length of the hollow cylinder, and λ is the thermal conductivity.

For a partial hollow cylinder, the thermal resistance formula is modified as follows:

$$R = \frac{2\pi}{\phi} \frac{\ln\left(\frac{r_o}{r_i}\right)}{2\pi\lambda l} \tag{7}$$

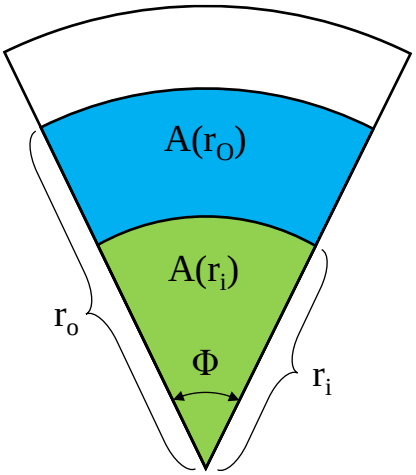


Figure 2. Radial schematic diagram of a hollow cylinder.

A contact gap exists between the stator surface and the motor housing, which is the cause of contact thermal resistance. The calculation formula for equivalent contact thermal resistance is as follows:

$$R_h^{lam} = \frac{\delta_h^{lam}}{\lambda_{air} A_h} \tag{8}$$

Where R_h^{lam} is the equivalent contact thermal resistance, δ_h^{lam} is the air gap thickness, λ_{air} is the thermal conductivity of air, and A_h is the contact area.

Calculation formula for the radial equivalent thermal resistance of the stator yoke:

$$R_y = \frac{1}{2\pi\lambda_r l_{Fe}} \ln\left(\frac{r_{sto}}{r_{sto} - \frac{r_{sto} - r_{sti}}{2}}\right) \tag{9}$$

Calculation formula for the axial equivalent thermal resistance of the stator yoke:

$$R_y^{axial} = \frac{l_{Fe}}{\pi\lambda_{axial}\left[r_{sto}^{\ 2} - r_{sti}^{\ 2}\right]} \tag{10}$$

Where R_y is the equivalent thermal resistance of the stator yoke in the radial direction of the cylinder, R_y^{axial} is the equivalent thermal resistance of the stator yoke in the axial direction of the cylinder, r_{sto} is the outer diameter of the stator core, r_{sti} is the inner diameter of the stator core, and l_{Fe} is the axial length of the stator core.

Since each tooth of the motor stator core has the same shape and size, the equivalent thermal resistance can be regarded as the parallel connection of the thermal resistances of individual teeth.

Calculation formula for the radial thermal resistance of the stator teeth:

$$R_{st} = \frac{1}{k_{st} \cdot 2\pi\lambda_r l_{Fe}} \ln\left(\frac{r_{syi}}{r_{si}}\right) \tag{11}$$

Where r_{syi} is the inner radius of the yoke, r_{si} is the inner radius of the tooth, and k_{st} is the ratio of the inner area to the outer area of the tooth.

Calculation formula for the axial thermal resistance of the stator teeth:

$$R_{st}^{axial} = \frac{l_{Fe}}{\lambda_y^{axial} h_s b_{ts} Z_s} \quad (12)$$

Where R_{st_axial} is the axial thermal resistance of the stator teeth, h_s is the tooth height, b_{ts} is the tooth width, and Z_s is the number of teeth.

The rotor region is divided into several sections, as shown in Figure 3, each of which can be regarded as a partial hollow cylinder. The specific values are listed in Table 2.

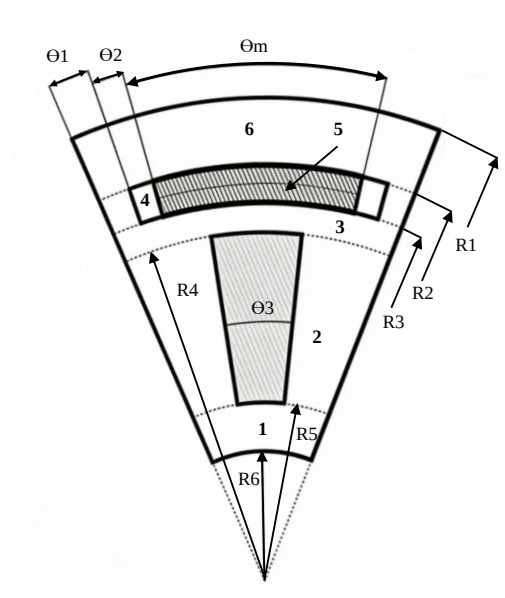


Figure 3. Schematic Diagram of Rotor Thermal Resistance.

Table 2. Rotor Thermal Resistance Parameters.

Thermal resistance	*Inner diameter*	*Outer diameter*	*Angle*
R_{r1}	R_6	R_5	2π
R_{r2}	R_5	R_4	2π-p*Θ_3
R_{r3}	R_4	R_3	2π
R_{r4}	R_3	R_2	Θ_2
R_{r5}	R_3	R_2	Θ_m
R_{r6}	R_2	R_1	2π

3 Thermal Modeling and Bench Test Verification

3.1 Thermal Modeling

As electric drive systems evolve toward higher power density, higher speed, and greater integration, research on motor thermal simulation carries significant engineering value. As an interdisciplinary simulation platform, Amesim enables the construction of coupled simulation models covering mechanical, electrical, hydraulic, and thermal physics based on its industrially validated standard component library.

The modeling principle of the equivalent thermal network method is to divide the motor into different sections. The number of nodes for each section can be set according to the required level of detail, and the various motor sections are connected along the direction of heat flow to form a thermal network model. Since the actual heat transfer in a motor is highly complex, in-depth analysis of the asymmetric heat transfer and the variability of heat capacity parameters during the process cannot be achieved using the Amesim platform alone. Therefore, the following assumptions are adopted in the motor thermal simulation in this paper:

(1) The non-uniformity of the heat source in the motor is neglected, and the heat source is assumed to be continuously and uniformly distributed.

(2) The heat transfer coefficient does not change with time.

(3) Parameters such as thermal conductivity and heat capacity are regarded as constant values.

In this paper, the motor is divided radially into the rotor, air gap, stator, housing, and other components. The rotor is subdivided into nodes, including the rotor shaft, permanent magnets, and rotor contact surface, while the stator is divided into nodes such as end windings, winding cores, teeth, and yoke.

The thermal network model consists of multiple heat sources, heat capacities, and thermal resistances. The heat sources, which serve as the system input, are derived from various losses in different motor parts. Heat sources are applied to the corresponding heating elements, heat capacity parameters are assigned inside each motor component, and thermal resistance parameters are set within heat exchange elements. These heat sources, heat capacities, and thermal resistances are connected according to the direction of heat flow to form the overall motor. Figure 4 presents the specific configuration where each thermal node is interconnected via thermal resistances. The thermal capacitance of each node depends on material properties, and the calculation of thermal resistance parameters is elaborated in Section 2.3 Thermal Resistance Parameter Calculation.

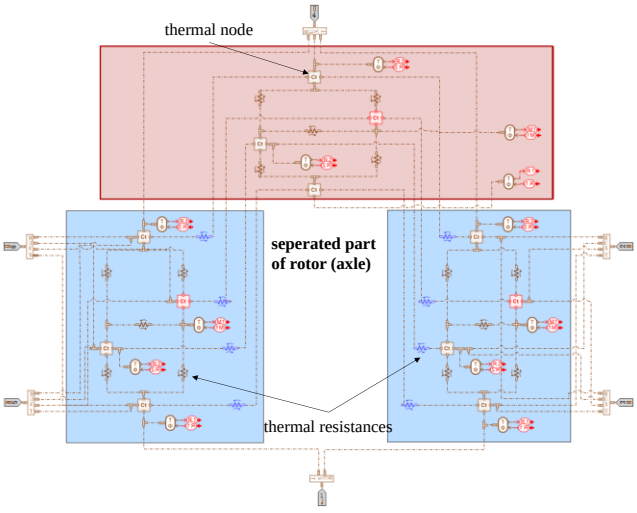


Figure 4. Schematic of Thermal Network Model.

DOI
10.3384/ecp223393

The thermal network model of the motor established on the Amesim platform is shown in Figure 5. The diagram illustrates the connection relationships between the motor model, the ambient environment, and loss inputs. The loss calculation method is detailed in Section 2.2 Loss Analysis. The thermal model, built in Simcenter Amesim, is fully compatible with the Modelica and FMI standards. It is exported as an FMU for co-simulation with MATLAB/Simulink, thereby fitting naturally into Modelica/FMI-oriented development processes.

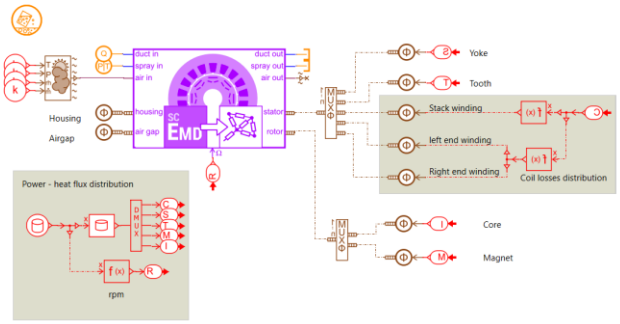


Figure 5. Thermal Model Diagram of the Motor.

3.2 Bench Test Verification

A DC power supply is used as the test power source, with a power analyzer connected between the power supply and the controller to measure the input power. The controller converts DC power into three‑phase AC power and is connected to the main bench controller, allowing real‑time variations of internal parameters of the controller to be displayed on the main control computer.

A dynamometer is connected to each side of the motor, and the rotational speed and torque sensors are used to monitor the real‑time output speed and torque. The output speed of the motor equals the measured speed (the left and right speeds are equal) multiplied by the reduction ratio of the gearbox. The output torque is the sum of the left and right torques divided by the reduction ratio. The schematic diagram is shown in Figure 6.

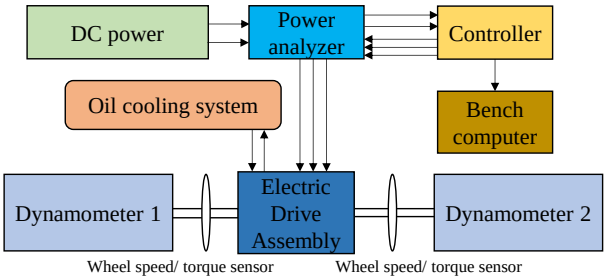


Figure 6. Schematic Diagram of the Test Bench.

The actual photograph of the test bench site is shown in Figure 7.

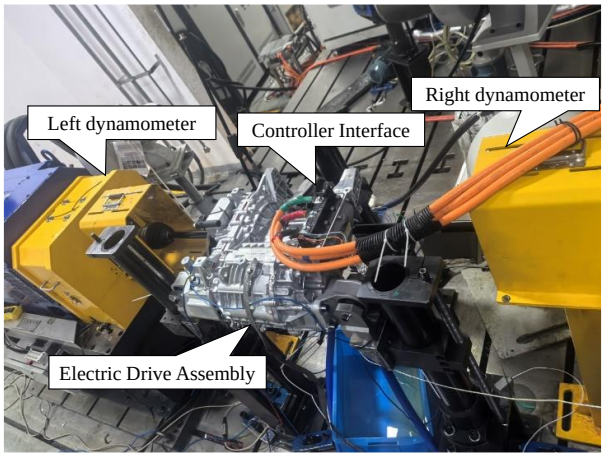


Figure 7. Demonstration of the Test Bench.

Multiple temperature sensors are attached to the yoke of the tested motor. Three operating conditions with an inlet oil temperature of 80 ℃ and an oil flow rate of 8 L/min were selected in the test, and the motor yoke temperatures under different working conditions were measured. The comparison between experimental data and simulation results is presented in Table 3.

Table 3 Measured and Simulation Error.

Conditions	*Test result*	*Model result*	*Error*
37N·m 15550rpm	114.7℃	109.2℃	4.79%
80N·m 8400rpm	104.9℃	99.1℃	5.53%
190N·m 1500rpm	102.2℃	102.6℃	0.39%

It can be seen from the table that the established simulation model can well reproduce the measured data under most operating conditions, with the maximum error controlled within 5.53%.

4 Conclusion

This paper focuses on the thermal behavior of an electric drive assembly for electric vehicles, systematically analyzes the heat source distribution, and calculates the internal thermal resistance of the motor. Combined with numerical modeling and experimental tests, the reliability of the thermal simulation model is verified. The research results can provide a reference for the thermal management design of electric drive assemblies and facilitate the high-efficiency and integrated development of electric vehicles.

References

[1] Alptekin, Ertan. “Emission, injection and combustion characteristics of biodiesel and oxygenated fuel blends in a common rail diesel engine.” Energy 119 (2017): 44-52.

[2] Lee K H, Cha H R, Kim Y B. Development of an interior permanent magnet motor through rotor cooling for electric vehicles[J]. Applied Thermal Engineering, 2016, 95: 348-356.

[3] Park M H, Kim S C. Thermal characteristics and effects of oil spray cooling on in-wheel motors in electric vehicles[J]. Applied Thermal Engineering, 2019, 152: 582-593.

[4] Davin T, Pellé J, Harmand S, et al. Experimental study of oil cooling systems for electric motors[J]. Applied Thermal Engineering, 2015, 75: 1-13.
[5] Ha T, Kang Y, Kim N S, et al. Cooling effect of oil cooling method on electric vehicle motors with hairpin winding[J]. Journal of Mechanical Science and Technology, 2021, 35: 407-415.
[6] Fasquelle A, Laloy D. Water cold plates cooling in a permanent magnet synchronous motor[J]. IEEE Transactions on Industry Applications, 2017, 53(5): 4406-4413.
[7] Hu Y , Chen B , Jia J ,et al. Design and Analysis of Cooling System for High Speed PM Synchronous Motor with Magnetic Bearings[C]//2018 21st International Conference on Electrical Machines and Systems (ICEMS).2018.
[8] Y. Zhang et al. Integrated Thermal Management of New Energy Vehicles[M]. Beijing: Huazhong University of Science and Technology Press, 2023.
[9] Anderssson B . Lumped Parameter Thermal Modelling of Electric Machines. 2013.

DOI 10.3384/ecp223393

Calibration workflow development using FMI-based reduced-order models for NEVs thermal management

Rui Gao[1] Yuhao Xu[1] Junjie Chen[1] Junyang Hou[1]
Weilin Li[2] Da Li[2] Yang Qi[2] Yue Tang[3]
[1]Nanjing Yuansi SimTek Co., Ltd, China
{rui.gao, yuhao.xu, junjie.chen, junyang.hou}@simtek.cc
[2]Northwestern Polytechnical University, Xi'an, China
{Wli907, lida, qiyang}@nwpu.edu.cn
[3]Harbin University of Science and Technology, China
tangyue5@hrbust.edu.cn

Abstract

New Energy Vehicles (NEVs) drive China's automotive market. However, calibrating thermal management for the battery, motor, and inverter is a major bottleneck, especially as development cycles shrink to 24 months. High-fidelity thermal-fluid models are too slow for repeated calibration and fixed-step real-time execution. This paper presents an end-to-end industrial workflow: 1) Build high-fidelity models using 3D data; 2) Train reduced-order models (ROMs), implemented here as non-intrusive data-driven surrogate models, from simulation data; 3) Export the trained surrogates as FMI-compliant co-simulation FMUs for NI VeriStand real-time testing. The ROM tool proves efficient for repeated calibration runs; in this paper, ROM refers to a non-intrusive data-driven surrogate model exported as an FMI-compliant co-simulation FMU, rather than a projection-based or equation-level model-order-reduction method. The reported speedup is interpreted as repeated-simulation acceleration, with data generation and training treated as one-time amortized costs.

For the reported battery validation cases, temperature RMSE was evaluated against experimental or CAE reference data and normalized by the initial temperature of the battery heat-generation operating condition. The calculation results show that the single-cell RMSE is within 5%, the battery-pack RMSE is within 10%, and the absolute temperature error is within 3 K. For a representative 10-module battery-pack cooling case, the 250 s closed-loop simulation time was reduced from 2200 s to 44 s after surrogate-FMU replacement, and the calibrated PI response met the control targets.

Keywords: *thermal management, 1D-3D modeling, data-driven ROM, NEV three-electric thermal management system.*

1 Introduction

The rapid growth of China's New Energy Vehicle (NEV) market has driven a significant compression of vehicle development cycles, now typically as short as 24 months. While thermal management calibration for traditional powertrains (e.g., engine and transmission) has been studied for decades, the calibration of thermal systems for the three core electric components in NEVs—battery, motor, and inverter—has emerged as a critical development bottleneck. Effective thermal management directly influences battery longevity and overall vehicle lifecycle, yet current calibration practices remain time-consuming and resource-intensive.

In conventional development workflows, engineers rely heavily on physical testing and high-fidelity 3D CAE simulations, which are computationally expensive and can take hours or even days to evaluate a single use case. Moreover, control system calibration—where plant and controller parameters are tuned against experimental data—is often decoupled from the CAE analysis process. Simulation teams working on 3D CAE and 1D control models seldom exchange data due to differences in model fidelity and scale. Although co-simulation between CAE and system simulation tools has been explored, traditional co-simulation approaches still fall short in computational efficiency and automation support, especially for calibration tasks that require frequent parameter iteration and extensive operating-condition testing.

To overcome these challenges, model reduction and surrogate modeling have gained increasing attention. Model-based reduction uses governing equations or system structure to preserve physical interpretability, whereas data-driven surrogate modeling learns input-output behavior directly from simulation or measurement data. In this paper, ROM is used for non-intrusive data-driven surrogates that replace high-fidelity models with lower-complexity representations for calibration. These ROMs may be static or dynamic, depending on whether time-varying behavior must be represented.

Related work has already addressed FMI-based neural-network and AI deployment. Urbani et al. proposed ONNX2FMU to wrap ONNX neural-network models as FMI-compliant FMUs[1]. Thummerer, Kircher, and Mikelsons developed NeuralFMU/FMIFlux.jl to integrate FMUs into neural-network training workflows[2]. Heuermann et al. demonstrated an OpenModelica compiler prototype that replaces selected non-linear algebraic loops with error-controlled machine-learning surrogates[3]. More recent work by Hofmann and Mikelsons on physics-enhanced Neural ODEs[4], Kamp et al. on Modelica/eFMI embedded hybrid simulation models[5], and Langenkamp et al. on efficient PeN-ODE training[6] further confirms that the AI-FMI interface is an active research direction.

Accordingly, the novelty claimed here is not a standalone neural-network-to-FMU conversion algorithm. The contribution is the calibration workflow that connects CAE-informed Modelica modeling, surrogate training, FMI co-simulation FMU export, and real-time HIL deployment for NEV thermal-management calibration.

For vehicle thermal-management context, prior Modelica and co-simulation work on integrated vehicle thermal management and battery-electric-vehicle thermal systems is also cited explicitly[7,8].

Surrogate-assisted calibration for complex system models and credibility reporting for Modelica models further motivate the validation and reporting requirements used in this workflow[9,10].

In this paper, we present an integrated workflow designed to accelerate the virtual calibration of NEV thermal management systems. Our contributions are threefold:

1. We propose a modeling best practice that systematically integrates 3D high-fidelity simulation data into 1D system-level models for batteries and motors.
2. We integrate a ROM generation and FMI export workflow that evaluates candidate machine-learning algorithms and exports validated FMI-compliant co-simulation FMUs for calibration use.
3. We construct and demonstrate a calibration workflow spanning offline parameter tuning and real-time hardware-in-the-loop (HIL) validation, and report its effect on repeated-simulation time in the evaluated cases.

The proposed workflow supports earlier-stage virtual calibration and provides a basis for reducing prototype-dependent iterations, subject to case-specific validation.

The remainder of this paper is organized as follows. Section 2 formulates the problem and outlines the proposed calibration workflow. Section 3 details the multiphysics modeling of the battery, motor, inverter, and liquid cooling system, with 3D simulation results integrated as tabular data or reference for 1D model calibration. Section 4 describes the training of neural-network-based surrogate models using simulation data from high-fidelity Modelica models. Section 5 presents the implementation and results of real-time simulation on the NI VeriStand platform. Finally, Section 6 provides concluding remarks and discusses future work.

2 Problem formulation

Traditionally, the calibration of New Energy Vehicle (NEV) thermal management systems relies heavily on physical testing conducted after the full system integration. System models are developed using tools such as MATLAB/Simulink, Modelica/Dymola, and AMESim. An integrated thermal management system model typically combines plant models from physical modeling tools and controller models from MATLAB/Simulink. The development process spans from Model-in-the-Loop (MIL) to Hardware-in-the-Loop (HIL) stages, where plant models must be simplified to achieve faster simulation speeds.

This conventional development paradigm faces significant challenges as vehicle development cycles continue to shorten. To accelerate calibration, one potential approach is to reduce the number of tunable parameters, thereby favoring precise models with fewer calibration variables. The integration of 3D CAE (Computer-Aided Engineering) analysis results is also considered to enhance model fidelity. In the automotive industry, for instance, 3D CAE tools like Maxwell are commonly used for electromagnetic analysis of motors, while tools like Star-CCM+ are employed for flow resistance and thermal conduction analysis in motor cooling circuit design.

However, a fundamental conflict arises: while fast simulation speed is crucial for efficient calibration, high model fidelity is equally important for accuracy. These two requirements often present contradictory conditions that are difficult to satisfy simultaneously.

The central methodological challenge is to reconcile the fidelity required for physically credible thermal-management models with the computational efficiency required for iterative calibration and HIL deployment. High-fidelity component models preserve the relevant multiphysics behavior and provide reference data for calibration, but their computational cost limits repeated evaluation over diverse operating conditions. To address this trade-off, the proposed workflow integrates model construction, calibration, surrogate-model generation, and deployment validation into a single process. CAE-informed high-fidelity component models first provide the physical reference and calibration data; their validated input-output behavior is then represented by non-intrusive, data-driven ROMs, which are exported as FMI-compliant co-simulation FMUs for repeated simulation and HIL validation. This organization assigns complementary roles to the two model classes: high-fidelity models preserve physical credibility, whereas ROM-FMUs provide computationally efficient execution. Specific

DOI
10.3384/ecp223399

operating conditions and validation targets are reported with the corresponding case studies.

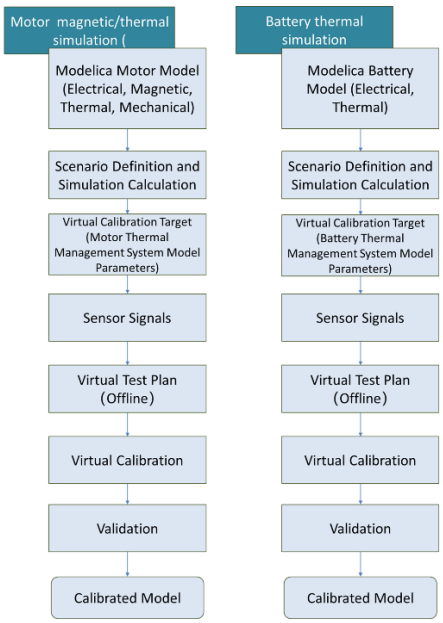


Figure 1 Virtual calibration workflow for physical models.

To address this challenge, we propose and implement a two-step virtual calibration workflow, comprising two main perspectives: one for physical models (Figure 1) and another for control system models (Figure 2).

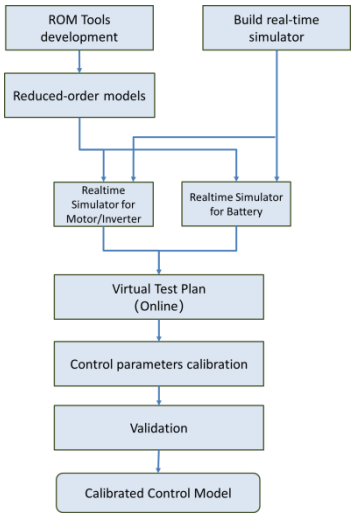


Figure 2 Virtual calibration workflow for control models.

Furthermore, to enable a left-shift in the calibration process—moving calibration activities earlier in the development cycle—we focus on calibrating the thermal management systems of core components (battery, motor, and inverter) individually, rather than attempting to calibrate the entire NEV thermal management system as a whole.

Regarding the thermal management technology considered, this study focuses on the widely adopted liquid cooling approach, optionally supplemented by air cooling. We do not explore direct refrigerant cooling technologies, as our primary goal is to establish a generalized workflow rather than to optimize a specific cooling method[11,12].

3 Modeling

This section details the development of multi-physics models for the core components of New Energy Vehicle (NEV) thermal management systems, covering the battery, motor, inverter, and associated cooling circuits. To ensure model fidelity, high-fidelity 3D simulation results from Star-CCM+ are integrated into the Modelica-based battery models, enabling direct comparison of simulated and predicted surface temperatures on the battery pack. For the motor models, electromagnetic loss data from Maxwell simulations are imported as tabular map data into the 1D Modelica motor models, while thermal-fluid results from Star-CCM+ serve as reference data for validating the temperature predictions of the corresponding Modelica models.

The thermal-management models were implemented in Modelica/Cautosim and organized around component-level plant models, cooling-loop elements, actuator paths, and ambient boundary conditions. High-fidelity CAE data were used as references for system-level model construction and calibration: Star-CCM+ provided flow-resistance and thermal-field information for battery and cooling-channel models, while Maxwell supplied electromagnetic loss maps for the motor model. Component-level comparisons with reference simulation results were used to define the validation boundary before the calibrated models were further used for ROM generation and HIL-oriented deployment studies. For the HIL-oriented workflow, a system-level communication frequency greater than 1 Hz was taken as the minimum requirement for real-time communication.

3.1 Battery

In the Modelica community, battery thermal management system modeling has been studied intensively[13-15]. In this study, a high-fidelity battery thermal management model is developed in two distinct, complementary steps. First, a battery electro-thermal coupling model is constructed based on fundamental physical principles, including Kirchhoff's voltage law, the law of conservation of energy, and Newton's law of cooling. In this coupled formulation, the electrical submodel receives the measured sensor current and the temperature output from the thermal submodel as inputs. It uses a Thévenin equivalent-circuit representation to predict output voltage. Conversely, the thermal submodel takes the sensor current and the voltage computed by the electrical submodel as inputs. Leveraging the battery's topological structure, it calculates heat generation, heat transfer, and heat dissipation using Bernardi's equation, Fourier's law, and Newton's law of cooling, respectively, to output the battery temperature.

Subsequently, a detailed 3D cell model is constructed using Star-CCM+, explicitly representing the positive electrode, negative electrode, separator, electrolyte, and terminals. This geometry is combined with the electro-thermal coupling model and the defined simulation environment, including temperature, pressure, and cooling parameters, to establish thermal boundary conditions for a 3D single-cell representation. The model is then extended to pack level by including additional components such as busbars, structural brackets, the cooling system, and the enclosure. The resulting Star-CCM+ pack model provides temperature-field data exported to CSV and used as reference data for calibration of the corresponding 1D Modelica system model. In this role, the 3D model supplies physically resolved boundary information for system-level calibration rather than serving as an independently optimized battery-pack thermal design.



Figure 3 3D battery and cooling plate model

Simulation studies are conducted to characterize the liquid cooling system's performance. This includes analyzing the temperature distribution within the cooling flow paths under various constant charge and discharge currents, as well as evaluating pressure drop variations across a range of mass flow rates. These results are summarized in Figure 4.

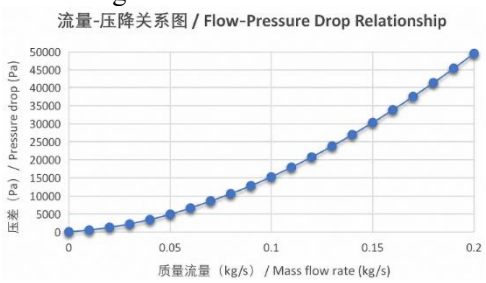


Figure 4 Pressure drop w.r.t. flow rate in the cooling plate

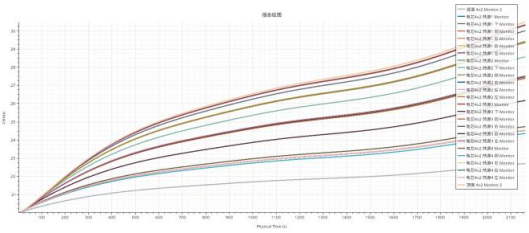


Figure 5 Monitoring temperature of the 4 x 2 cell

Complementary studies are performed to assess the battery pack's thermal behavior under different cooling strategies, specifically examining the surface temperature of both individual cells and the full battery pack under adiabatic (no cooling) and air-cooled conditions, as shown in Figure 5.

A corresponding 1D electric-thermal battery model is constructed in Modelica, encompassing the electrical behavior, internal heat generation, conductive heat transfer, and convective heat exchange with the liquid cooling system. Equivalent circuit models (ECMs) are widely employed for battery state estimation due to their low parameterization requirements and computational efficiency. ECMs are primarily categorized based on the number of RC branches: the Rint model, the Thévenin model, the second-order RC model, and the generalized N-order RC model. The Thévenin model—comprising an ideal voltage source, a series ohmic internal resistance, and a single RC parallel branch—is selected for this work. This choice is motivated by its favorable trade-off between simulation accuracy and speed, as well as the clear physical interpretability of its parameters, which facilitates linkage to measurable battery characteristics.

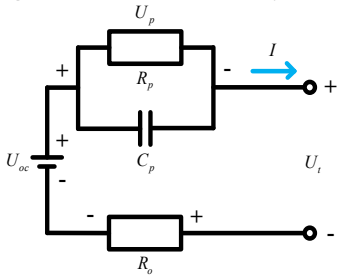


Figure 6 Thévenin equivalent model

The governing equations for the selected Thévenin model are given in (1) and (2). In these equations U_p, R_0, R_p, C_p, and I denote the polarization voltage, ohmic internal resistance, polarization resistance, polarization capacitance, and current, respectively. The corresponding implementation of this electric battery model in the Modelica environment is shown in Figure 7.

$$\begin{cases} \dfrac{dU_p}{dt} = -\dfrac{U_p}{R_p \cdot C_p} + \dfrac{1}{C_p} \cdot I & (1) \\ U = U_{oc} - R_0 \cdot I - U_p & (2) \end{cases}$$

The state of charge (SOC) is calculated by coulomb counting, as defined in equation (3), in which Q denotes the nominal cell capacity.

$$SOC(t) = SOC(0) - \frac{1}{Q}\int_0^t I(\tau)d\tau \qquad (3)$$

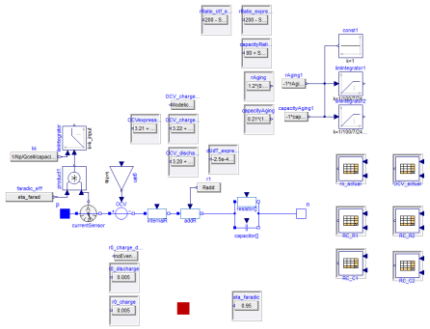


Figure 7 Electric battery model in Modelica

DOI
10.3384/ecp223399

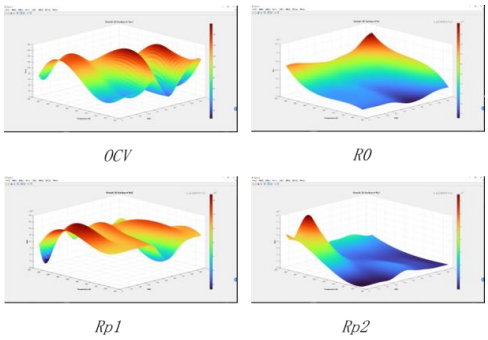


Figure 8 Electric coefficients calibration model for a single cell model

The parameters of the 1D Modelica battery model were calibrated against the 3D reference results using open-loop identification and closed-loop electro-thermal validation, as illustrated in Figures 8 and 10.

Validation was conducted for representative single-cell and cell-stack cases. The resulting temperature histories and error profiles are presented in Figures 9-11.

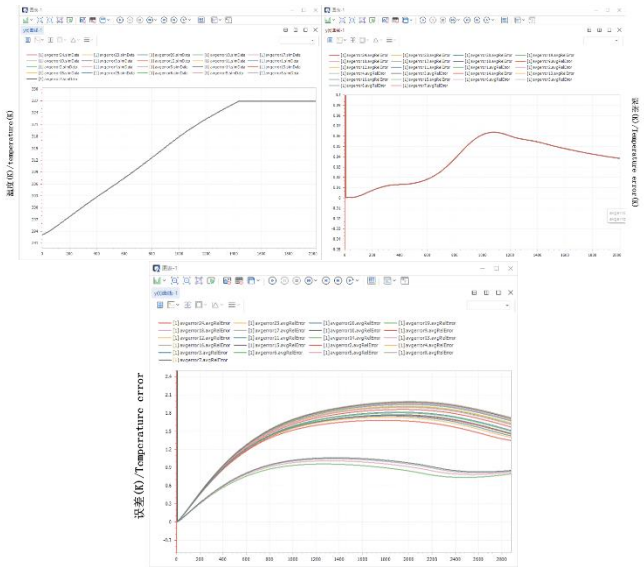


Figure 9 Calibration results for heat conduction coefficients of the single cell and battery pack *(Top-left: cell temperature; Top-right: cell temperature error; Bottom: battery temperature error.)*

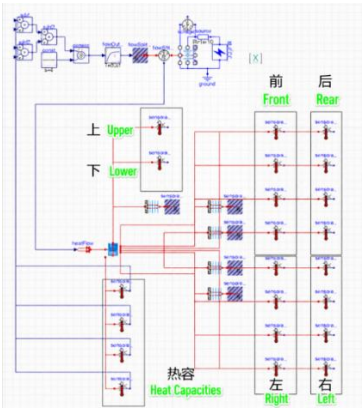


Figure 10 Closed-loop calibration model for thermal capacitors

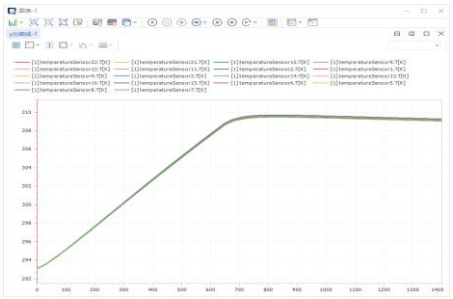


Figure 11 Temperature of the capacitors at different position in the closed-loop calibration

After parameter calibration, the temperature-error evaluation was performed using experimental measurements or benchmark CAE results as the reference. Consistent with the RMSE treatment used for other compared physical quantities, the temperature RMSE was normalized by the initial temperature of the battery heat-generation operating condition, because the battery temperature rise during operation is the primary response of interest. The calculation results show that the single-cell case remains within a 5% RMSE bound, the battery-pack case remains within a 10% RMSE bound, and the absolute temperature error remains within 3 K.

The open-loop thermal-parameter calibration model is shown in Figure 12.

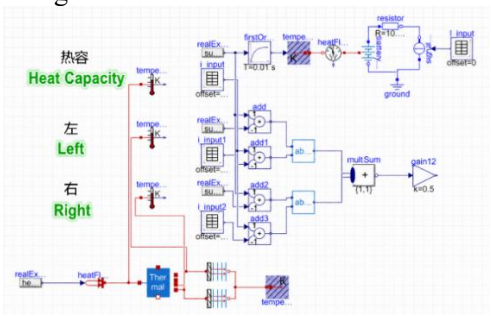


Figure 12 Heat conduction coefficients calibration model for a single cell model

Through the integration of 3D high-fidelity analysis results into the Modelica environment and subsequent parameter calibration, a calibrated 1D Modelica battery model was obtained for the validation and ROM-generation studies.

3.2 Motor

The Permanent Magnet Synchronous Machine (PMSM) is the predominant traction motor technology in New Energy Vehicles (NEVs). To enable high-fidelity system-level simulation, detailed electromagnetic and thermal analyses are performed. Electromagnetic characterization is conducted using Maxwell simulation software, which enables precise 3D analysis of electromagnetic forces and losses. The process begins with constructing a detailed three-dimensional electromagnetic model of the motor, incorporating accurate geometries and material properties (e.g., core magnetic permeability, winding conductivity) for the stator, rotor, and windings. By applying appropriate boundary conditions and excitation sources,

the software simulates the motor's operational state across various working conditions.

For loss analysis, Maxwell is used to estimate both copper and iron losses. Copper losses, resulting from current flow through winding resistance, are determined from the simulated current distribution and material resistance parameters. Iron losses, comprising hysteresis and eddy-current losses, are modeled using the core material characteristics, electrical conductivity, geometry, and magnetic-field frequency. The resulting electromagnetic loss maps are used as heat-source inputs for the motor thermal-management model and support analysis of motor temperature-field behavior. A simulated motor efficiency map is shown in Figure 13.

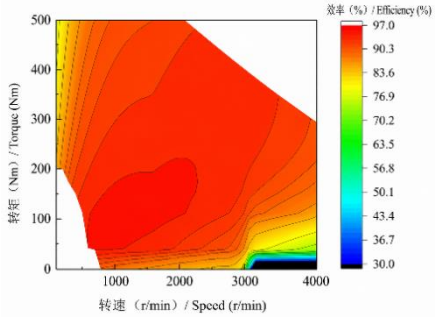


Figure 13 Motor efficiency map by simulation

Subsequently, a conjugate heat transfer analysis is performed using Star-CCM+ to characterize the cooling system's performance. This analysis establishes the relationship between the cooling flow rate and key thermal parameters, namely the surface temperature and the local heat transfer coefficient (HTC) on the motor housing. The surface HTC distribution for a representative flow condition is visualized as a contour plot in Figure 14.

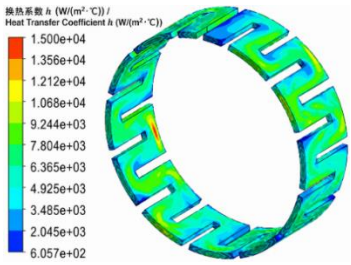


Figure 14 Contour plot of the surface heat transfer coefficient (HTC) on the motor housing

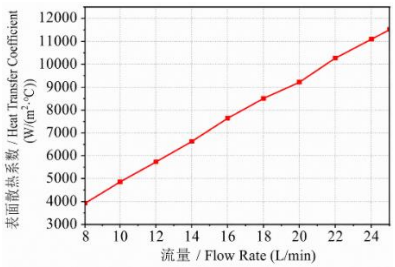


Figure 15 Flow rate versus area-averaged surface heat transfer coefficient

For a series of different inlet flow conditions, numerical simulations are used to obtain the local HTC distribution across the heat dissipation surface. To concisely represent the overall cooling performance under each condition, the local HTC values are area-weighted and averaged, yielding a single representative average HTC. The relationship between the coolant flow rate and this area-averaged surface HTC is plotted in Figure 15.

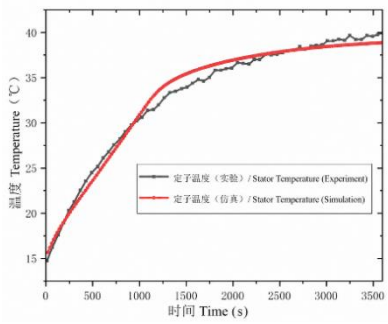


Figure 16 Stator temperature comparison: experiment (black) vs. Simulation (red).

The predicted stator temperature from the coupled electromagnetic-thermal simulation was compared with motor test-bench measurements, as shown in Figure 16. The agreement provides qualitative validation of the simulated loss maps and heat-transfer coefficients.

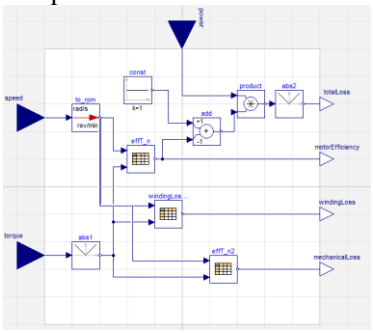


Figure 17 Motor losses modeled by tables

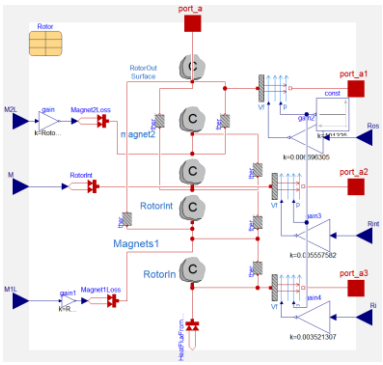


Figure 18 Rotor heat network model

The CAE-derived loss maps and thermal boundary conditions are then integrated into a 1D Modelica system-level motor model. This integration is typically implemented using parameterized lookup tables or empirical correlations within the Modelica environment, as conceptually depicted in Figure 17. In this way, the system model retains the relevant heat-source and

DOI
10.3384/ecp223399

boundary-condition dependencies while remaining suitable for full-vehicle simulation. Finally, analogous to the battery-model calibration, the thermal capacitances and heat-transfer coefficients in the 1D motor model are adjusted against detailed CAE analysis results, as illustrated in Figure 18.

3.3 Power Electronics

The power-electronics subsystem, including inverters, chargers, and controllers, integrates multiple electronic components and power modules. During operation, these devices generate heat due to current flow, switching losses, and signal-processing activity. For instance, inverter switching during DC-AC conversion produces losses that are converted into thermal energy. If this heat is not dissipated effectively, elevated temperatures may lead to performance degradation or reliability concerns.

The inverter is represented using a Modelica-based modeling approach, as shown in Figure 19. Separate 3D high-fidelity simulation (e.g., with CFD tools) was not performed in this study, primarily for two reasons: (1) there is an industry trend toward integrating the motor and inverter into a single physical unit, making a standalone 3D inverter analysis less aligned with current packaging practices; and (2) the present work focuses on demonstrating and validating the overall calibration workflow, rather than developing a complete, vehicle-specific thermal management model for every component. Therefore, the 1D Modelica representation provides a suitable balance between modeling fidelity and computational efficiency for the purposes of this workflow demonstration.

Because no standalone 3D inverter CFD model was used, the inverter case is presented mainly as a workflow demonstration. Its accuracy is bounded by the 1D assumptions and by any available component-level validation.

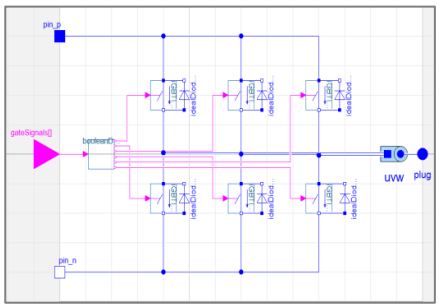


Figure 19 Inverter model

3.4 Thermal management systems

The developed battery, PMSM, and inverter models, together with their respective cooling system models, serve as the foundational building blocks for constructing various New Energy Vehicle (NEV) thermal management architectures. An example of a battery liquid cooling model, which forms part of such an architecture, is shown in Figure 20. For the validation discussion in this subsection, the battery liquid-cooling model in Figure 20 is selected as the representative high-fidelity object.

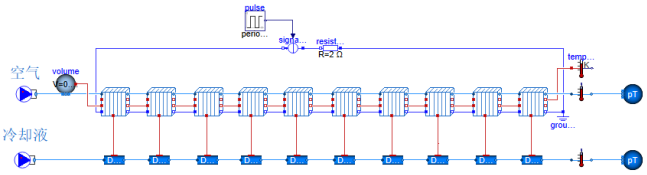


Figure 20 Battery liquid cooling model

When evaluating different thermal-management architectures under representative operating scenarios, the computational cost of high-fidelity physical models can limit repeated calibration studies. To make the validation boundary and numerical conditions explicit, the battery liquid-cooling model shown in Figure 20 was simulated for 250 s using an explicit Euler solver with a fixed step size of 0.01 s and a numerical accuracy setting of 1e-5. The validation was conducted by comparing the resulting temperature, flow-rate, and pressure responses with the corresponding Star-CCM+ three-dimensional CAE fields or component-level AMESim results, as applicable. For temperature responses, RMSE was computed relative to the experimental or CAE reference data and normalized by the initial temperature of the battery heat-generation operating condition. Under this evaluation, the battery-pack RMSE remains within 10%, and the absolute temperature error remains within 3 K. Under these conditions, the simulation required 36 min 40 s, as recorded in the simulation log in Figure 21. This runtime indicates that direct use of the high-fidelity model is inefficient for iterative calibration over multiple parameter settings.

```
INFO: Translation time: 00:00:22.595
INFO: Solver simulation started at: 2025/12/07 14:24:06
INFO: Simulation start time: 0.000000 seconds,
Algorithm: Euler
INFO: Initialization time: 00:00:00.011
INFO: Model simulation time: 00:36:40.023
INFO: Simulation completed at 250.000000 seconds (Stop
time = 250.000000 seconds)
INFO: State events: 2
INFO: Simulation steps: 25000
INFO: Simulation completed.
```

Figure 21 Simulation log for battery cooling simulation

Thermal-management system models typically consist of multiple interconnected cooling loops governed by control logic, which increases computational cost. These observations motivate the surrogate-model and FMI-deployment steps. At this stage, the subsection is restricted to defining the high-fidelity model, its numerical operating conditions, and the reference-based validation evidence; the ROM formulation and FMI deployment workflow are introduced in the subsequent sections.

4 ROM generation and FMI export

To address the computational bottleneck of high-fidelity Modelica thermal management models, a

dedicated reduced-order modeling (ROM) toolbox was developed. This toolbox implements several neural network architectures, categorized for different modeling needs: three are suited for static models, including Deep Neural Network (DNN), Gaussian Process (GP), and Response Surface Method (RSM); and three for dynamic models, including Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), and Nonlinear AutoRegressive with eXogenous inputs (NARX). The inclusion of multiple algorithms is motivated by the practical uncertainty in identifying a priori which method will best satisfy the accuracy and efficiency requirements for a specific ROM scenario. In this study, the candidate algorithms are used as an algorithm-selection interface, and no comprehensive performance ranking among DNN, GP, RSM, RNN, LSTM, and NARX is claimed.For the battery-pack case, the toolbox was configured as a time-series NARX model.

High-fidelity Modelica models are executed over a range of typical operating conditions to generate a comprehensive training dataset. This dataset includes key input variables such as current, voltage, torque, rotational speed, coolant flow rate, and inlet temperature, alongside target output variables like component temperatures, heat flow rates, and system pressure drops. A schematic of the ROM toolbox workflow is shown in Figure 22.

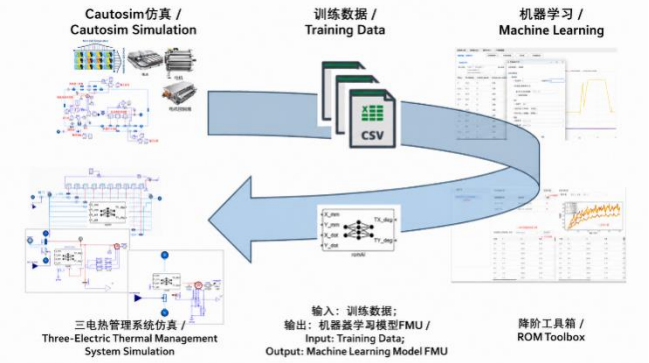


Figure 22 ROM toolbox schematic

The data preparation and training process was evaluated using alternative dataset sizes. Training with 25,000 data points per file required more than eight hours. A down-sampled dataset of 2,500 representative points per file, including key operating extremes, reduced training time to less than ten minutes on the tested laptop-class hardware while maintaining the reported accuracy. This result suggests that, for the evaluated thermal-system ROM case, a compact and representative dataset can be more efficient than a chronologically dense dataset.

The trained data-driven ROMs approximate the input-output relationships of the original physical models and are exported as FMI-compliant co-simulation FMUs for integration into larger system models.

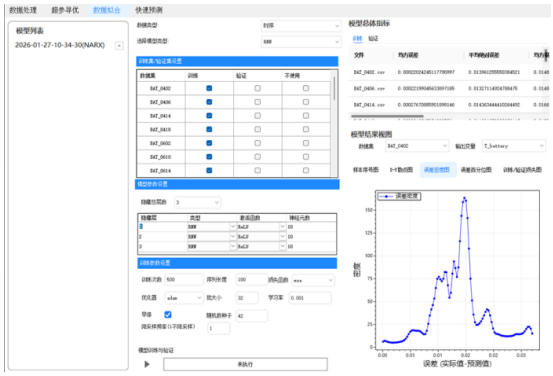
Dramatically Improved Simulation Speed: Enables rapid execution of system simulations.

Retention of Key Characteristics: Preserves essential thermal and hydraulic dynamics of the original system.

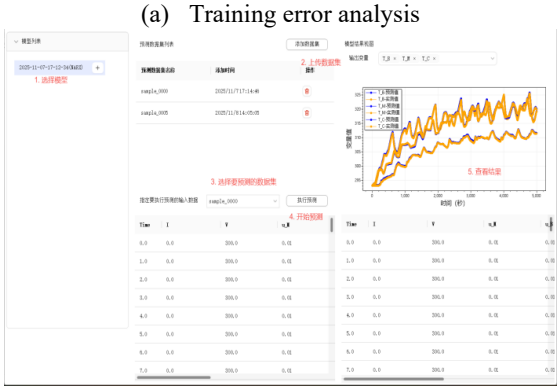
Standards-Based Compatibility: FMU export ensures interoperability with standard co-simulation environments.

Dual-Use Support: Facilitates both offline parameter calibration and real-time hardware-in-the-loop (HIL) applications.

The user interface of the ROM Toolbox, including model training and fast prediction modules, is shown in Figure 23.



(a) Training error analysis



(b) Fast prediction module

Figure 23 UI for ROM Toolbox

In the evaluated battery-cooling case, applying the trained ROM-FMU increased repeated-simulation speed while maintaining the reported temperature-error bounds relative to the high-fidelity reference model. For the 250 s scenario, the simulation time decreased from more than 36 min to 44 s after surrogate-FMU replacement, as shown in Figure 24. The calculation results indicate that the battery-pack RMSE remains within 10% and that the absolute temperature error remains within 3 K in the tested operating envelope. This result supports ROM-FMU substitution for iterative calibration in the evaluated case.

The timing comparison describes repeated simulation after the ROM surrogate has been trained. Data generation and model training are one-time costs that are amortized over repeated calibration runs.

The ROM-FMU acceleration evidence is summarized as follows:

- 5 Modules with PID controller: 20 train, 5 val.; simulation time decreased from 64 s to 17 s.

DOI
10.3384/ecp223399

- 5 Modules open loop: 20 train; 5 val.; simulation time decreased from 59 s to 16 s.
- 10 Modules with PID controller: 11 train, 5 val.; simulation time decreased from 2200 s to 44 s.
- 10 Modules open loop: 11 train, 5 val.; simulation time decreased from 2000 s to 40 s.

```
INFO: Translation time: 00:00:09.499
INFO: Solver simulation started at: 2025/12/08 15:26:55
INFO: Simulation start time: 0.000000 seconds,
Algorithm: Euler
INFO: Initialization time: 00:00:00.010
INFO: Model simulation time: 00:00:43.810
INFO: Simulation completed at 250.000000 seconds (Stop
time = 250.000000 seconds)
INFO: State events: 2
INFO: Simulation steps: 25000
INFO: Simulation completed.
```

Figure 24 Simulation log after the ROM handling

Using the same procedure, the thermal-management system models for the Permanent Magnet Synchronous Machine (PMSM) and the inverter were exported as co-simulation FMUs with the ROM Toolbox.

5 Real-time HIL test

To evaluate real-time execution, the generated reduced-order FMUs (Functional Mock-up Units) are deployed to the NI VeriStand real-time simulation environment. The hardware platform consists of a PXIe chassis equipped with a zero-slot controller (Model MT-X217). The controller uses an Intel i7-10510U quad-core processor (4 cores, 8 threads, maximum turbo frequency 4.9 GHz), a 1 TB M.2 NVMe solid-state drive, and peripheral I/O interfaces. It supports the Linux RT operating system. Key hardware specifications are summarized as follows:

- Intel® i7-10510U, quad-core processor, max turbo frequency 4.9 GHz;
- Single-channel 16 GB RAM;
- 1 TB M.2 NVMe SSD;
- Two Gigabit Ethernet ports;
- One USB 2.0 port, two USB 3.0 ports;
- Windows 10 / Linux RT.

The workflow for real-time testing comprises the following steps:

1. **Import High-Fidelity Models:** The high-fidelity Modelica thermal management system models are imported as the reference plant models.
2. **Generate and Train ROMs:** Training data is generated from the high-fidelity simulations, which is then used to train neural network-based reduced-order surrogate models (ROMs).
3. **Export ROMs as FMUs:** The trained ROMs are exported as FMI-compliant co-simulation FMUs, ready for real-time execution.
4. **Build Real-Time Simulation System:** A real-time simulation system is constructed within NI VeriStand, integrating the ROM-FMU (as the controlled plant) with the virtual controller model.
5. **Perform Online Calibration:** Online calibration and controller parameter tuning are conducted within this real-time environment.

Experimental results confirm that the ROM-FMU-based platform runs stably in real time, accurately reproduces the thermal dynamics of the original system, and supports hardware-in-the-loop (HIL) validation and virtual calibration tasks. This capability enables efficient verification of control strategies and contributes to shortening the development cycle for thermal management systems.

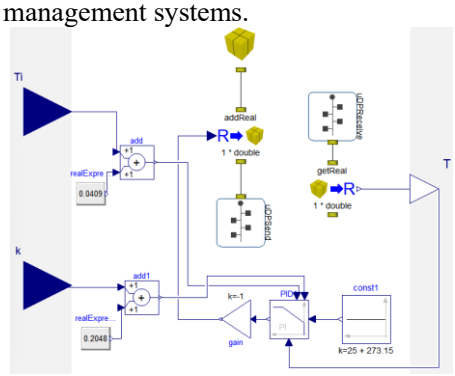


Figure 25 Device Driver model for UDP communication

Communication between the host PC and the real-time simulator is established using the User Datagram Protocol (UDP). The Modelica Device Driver library is employed to implement this communication interface, as depicted in Figure 25.

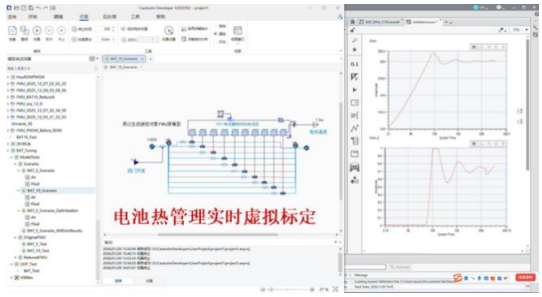


Figure 26 Realtime simulation interface in NI Veristand

Figure 26 shows a screenshot of the real-time simulation environment in NI VeriStand during the calibration of battery thermal-management controller parameters. In all tested cases, including the battery, PMSM, and inverter thermal-management subsystems, the communication/update frequency was set to 25 Hz. During NI VeriStand calibration, changes in the PID controller parameters were reflected in the monitored response curves in real time, indicating that the 25 Hz setting was adequate for interactive parameter tuning under the tested conditions.

6 Concluding remarks

This paper presents a model calibration workflow for New Energy Vehicle (NEV) thermal-management systems based on FMI-compliant data-driven reduced-order models (ROMs). In the evaluated cases, the workflow integrates 3D Computer-Aided Engineering (CAE) simulation results into 1D Modelica system models, trains neural-network-based surrogate models,

and deploys the resulting ROMs as co-simulation FMUs to real-time simulation platforms. The results demonstrate improved repeated-simulation speed, reported battery temperature-error bounds (single-cell RMSE within 5%, battery-pack RMSE within 10%, and absolute temperature error within 3 K), and FMI-based compatibility with real-time simulation.

The workflow also demonstrated deployment of the surrogate FMUs in a real-time Hardware-in-the-Loop (HIL) environment.

The method acknowledges certain limitations. Primarily, the accuracy of the ROMs remains dependent on the coverage and representativeness of the training data. Additionally, the risk of extrapolation error when applying the models to highly nonlinear or transient operating conditions outside the training domain cannot be fully mitigated by the current approach. For future work, we plan to investigate automated strategies for generating optimal training data to enhance model robustness and generalizability.

Acknowledgements

The authors extend their sincere gratitude to Shaoqing He, Bo Jin, Qi Wang, Jingwen Ding, Ben Hou, and numerous other colleagues at CATARC (China Automotive Technology & Research Center) for providing the Modelica simulation environment Cautosim and for their invaluable technical support throughout this research.

References

[1] Urbani, M., Bolognese, M., Pratticò, L., & Testi, M. (2025). A tool for the implementation of Open Neural Network Exchange models in Functional Mockup Units. In Proceedings of the 16th International Modelica & FMI Conference, Lucerne, Switzerland (pp. 645-651). DOI: 10.3384/ecp218645

[2] Thummerer, T., Kircher, J., & Mikelsons, L. (2021). NeuralFMU: Towards structural integration of FMUs into neural networks. In Proceedings of the 14th International Modelica Conference, Linköping, Sweden (pp. 297-306). DOI: 10.3384/ecp21181297

[3] Heuermann, A., Hannebohm, P., Schäfer, M., & Bachmann, B. (2023). Accelerating the simulation of equation-based models by replacing non-linear algebraic loops with error-controlled machine learning surrogates. In Proceedings of the 15th International Modelica Conference 2023, Aachen, Germany (pp. 275-284). DOI: 10.3384/ecp204275

[4] Hofmann, A., & Mikelsons, L. (2025). Towards integration of PeN-ODEs in a Modelica-based workflow. In Proceedings of the 16th International Modelica & FMI Conference, Lucerne, Switzerland (pp. 435-444). DOI: 10.3384/ecp218435

[5] Kamp, T., Bürger, C., Rein, J., & Brembeck, J. (2025). Hybrid simulation models for embedded applications: A Modelica and eFMI approach. In Proceedings of the 16th International Modelica & FMI Conference, Lucerne, Switzerland (pp. 545-555). DOI: 10.3384/ecp218555

[6] Langenkamp, L., Hannebohm, P., & Bachmann, B. (2025). Efficient training of physics-enhanced Neural ODEs via direct collocation and nonlinear programming. In Proceedings of the 16th International Modelica & FMI Conference, Lucerne, Switzerland (pp. 445-457). DOI: 10.3384/ecp218445

[7] Batteh, J., Gohl, J., & Chandrasekar, S. (2014). Integrated vehicle thermal management in Modelica: Overview and applications. In Proceedings of the 10th International Modelica Conference, Lund, Sweden (pp. 409-418). DOI: 10.3384/ecp14096409

[8] Shelly, T., Weibel, J. A., Ziviani, D., & Groll, E. A. (2022). A dynamic co-simulation framework for the analysis of battery electric vehicle thermal management systems. In 2022 21st IEEE Intersociety Conference on Thermal and Thermomechanical Phenomena in Electronic Systems (iTherm), San Diego, CA, USA (pp. 1-8). DOI: 10.1109/iTherm54085.2022.9899623

[9] Aka, J., Brunnemann, J., Freund, S., et al. (2023). Efficient global multi-parameter calibration for complex system models using machine-learning surrogates. In Proceedings of the 15th International Modelica Conference, Aachen, Germany (pp. 107-120). DOI: 10.3384/ecp204107

[10] Otter, M., Reiner, M., Tobolar, J., Leo, G., & Matthias, S. (2022). Towards Modelica models with credibility information. Electronics, 11(17), 2728.

[11] He, H., Saini, V., & Chen, X. (2022). Battery thermal management systems for EVs and its applications: A review. In Proceedings of the 8th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS) (pp. 57-68). DOI: 10.5220/0011030700003191

[12] Murugan, M., Prabhakar, S., Elumalai, P. V., Dhanasekaran, R., Sivaraman, R., Chakravarthy, B. S. V. R. K., ... & Prabhakar, S. (2025). A comprehensive review of thermal management methods and ideal system design for improved electric vehicle battery pack performance and safety. Energy Science & Engineering, 13(3), 1011-1036. DOI: 10.1002/ese3.2081

[13] Gerl, J., Janczyk, L., Krüger, I., & Modrow, N. (2014). A Modelica based lithium ion battery model. In Proceedings of the 10th International Modelica Conference, Lund, Sweden (pp. 335-341).

[14] Uddin, K., & Picarelli, A. (2014). Phenomenological Li-ion battery modelling in Dymola. In Proceedings of the 10th International Modelica Conference, Lund, Sweden (pp. 327-334).

[15] Ginneken, F., Busch, A., Varchmin, A., Tegethoff, W., & Koehler, J. (2025). Modelling, simulation and validation of thermal propagation for 3D discretized battery cells in Modelica. In Proceedings of the 16th International Modelica Conference, Lucerne, Switzerland (pp. 263-272). DOI: 10.3384/ecp218263

Thermal Simulation of a Low-Voltage Motor Based on a Thermal Resistance Network

Chen Zhixu[1] Yu Bing[1] Tan Han[1] Ou Haiying[1]
Xu Xiaodong[1]
[1]BYD Auto Industry Company Limited, China

Abstract

Low-voltage motors are increasingly used in steering, cooling, pumping, and cabin-control subsystems of new-energy vehicles, and excessive temperature can impair their reliability. This paper develops a compact Modelica thermal-network model for an external-rotor low-voltage permanent-magnet synchronous motor. The model represents the principal heat sources, lumped heat capacities, and conductive, convective, and radiative heat-transfer paths. For the tested mass-produced motor, an end-to-end simulation of 30 min of physical operation completed in approximately 2s. Comparison with measured housing temperature for one operating case yielded a maximum relative discrepancy of 5.9%, occurring at the housing-temperature measurement point. These results indicate that the model is a computationally efficient reduced-order tool for the motor and operating range studied. Because the experimental comparison is limited to one motor type and one measured temperature location, accuracy and parameter transferability must be reassessed before the model is applied to other motor topologies or cooling arrangements.

Keywords: Thermal resistance network, Modelica, Low-voltage motor, Permanent-magnet synchronous motor, TSemperature prediction

1 Introduction

The rapid expansion of electric vehicles has increased the number of low-voltage motors used in steering, thermal-management, pumping, and cabin-control subsystems. Gartner forecasts that the global in-use electric-vehicle fleet, including passenger cars, buses, vans, and heavy trucks, will reach 116 million in 2026 [1]. Because these motors often operate in compact spaces and under variable loads, their thermal behavior directly affects component reliability and vehicle safety.According to the research report of China Automotive Engineering Society, the proportion of vehicle failures caused by motor thermal failure reached 41.7% in 2023, which became the primary failure source of new energy vehicles during the warranty period [2].

Motor designers face two competing requirements. Increasing power density raises volumetric heat generation[3], whereas cost, mass, and packaging constraints limit the use of conservative structural margins[4]. Sensor-intensive protection and systematic overdesign can increase cost and integration complexity[5]. A fast thermal model that retains the dominant heat-transfer mechanisms is therefore useful during concept selection and design verification.

Reduced-order approaches address the computational cost of fully coupled three-dimensional analysis by retaining only the dominant thermal paths. For example, simplified multiphysics models of flat-wire motors omit secondary coupling paths to reduce computational burden[6]. Laboratory studies of low-voltage asynchronous motors also show that a single time constant may not adequately represent transient thermal behavior[7]. These studies motivate compact dynamic models whose assumptions and parameters are explicitly documented.

Other studies combine lumped thermal networks with finite-element or lower-dimensional parameter extraction to reduce run time while retaining engineering accuracy[8-10]. Parameter sensitivity and experimental perturbation have also been used to identify the thermal quantities that require careful calibration[11, 12]. Related Modelica work by Kral, Haumer, and co-authors developed experimentally compared thermal networks for surface-cooled and water-cooled squirrel-cage induction machines[13, 14]. Those studies provide an important reference for induction-machine cooling networks; the present work instead considers a compact external-rotor low-voltage PMSM and examines the effects of speed, current, emissivity, and pulse loading. Direct accuracy comparisons remain inappropriate because the machine topology, cooling configuration, and experimental measurements differ.

Three practical issues remain. First, detailed multiphysics models can be too slow for repeated design iterations. Second, reduced-order models must balance transient fidelity with computational efficiency. Third, uncertain contact, air-gap, and speed-dependent parameters can limit reuse unless their identification and calibration are reported transparently.

This paper develops a Modelica thermal-resistance-network model for an external-rotor PMSM. The contributions are: (1) a compact network that includes the principal solid-conduction, bearing, air-gap, internal-convection, and housing heat-rejection paths; (2) explicit

descriptions of the parameter sources and the empirical speed correction; (3) a housing-temperature comparison for one mass-produced motor; and (4) parametric studies of speed, current, emissivity, and pulse-current excitation. The results characterize the tested configuration and provide a basis for subsequent cross-motor validation rather than establishing universal accuracy.

2 Motor Model and Thermal Resistance Theory

2.1 Motor Model

External-rotor PMSMs are attractive for low-voltage vehicle auxiliaries because their geometry can provide high torque density in a compact package. This study uses one mass-produced external-rotor PMSM to illustrate the model structure, operating principle, and thermal-network formulation.

The motor in Figure 1 comprises a shaft, a double-sealed bearing system, an external rotor, an internal stator, surface-mounted permanent magnets, and concentrated windings. The permanent magnets are attached to the inner surface of the rotor, while the windings are distributed around the stator. The housing is die-cast aluminum alloy. These components define the heat-storage nodes and the principal radial and axial heat-transfer paths used in the reduced-order model.

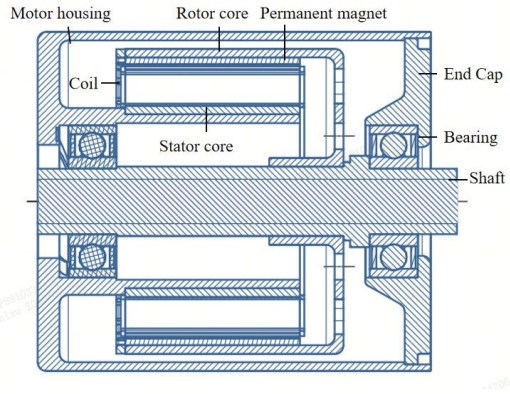


Figure 1. Structure of the tested low-voltage external-rotor PMSM.

2.2 Thermal Resistance Theory

A lumped thermal network represents a heat-transfer system by an electrical analogue: temperature corresponds to voltage, heat-flow rate to current, thermal resistance to electrical resistance, and heat capacity to capacitance. Fourier conduction, Newton convection, and surface radiation can therefore be expressed as network relations and solved together with the nodal energy balances.

Thermal resistance characterizes opposition to heat flow, and its reciprocal is thermal conductance. In the motor model, windings, cores, magnets, bearings, housing, and internal air are represented by lumped nodes connected through series or parallel resistance elements. The resulting network predicts nodal temperature histories for prescribed losses and boundary conditions.

For a lumped node, conservation of energy is written as:

$$C\frac{dT}{dt} = \dot{Q}_{in} - \dot{Q}_{out} \tag{1}$$

where C is the nodal heat capacity ($J \cdot K^{-1}$), T is the nodal absolute temperature (K), and $\dot{Q}_{in}$ and $\dot{Q}_{out}$ are the total heat-flow rates entering and leaving the node (W), respectively.

For a connection represented by a thermal resistance, the heat-flow rate is:

$$Q = \frac{\Delta T}{R_{th}} \tag{2}$$

where ΔT is the temperature difference between the connected nodes (K) and Rth is the thermal resistance ($K \cdot W^{-1}$).

For one-dimensional conduction, convection, and linearized radiation, the corresponding resistances are:

$$R_{th,cond} = \frac{L}{kA} \tag{3}$$

$$R_{th,conv} = \frac{1}{hA} \tag{4}$$

$$R_{th,rad} = \frac{(T - T_{ref})}{\varepsilon\sigma A(T^4 - {T_{ref}}^4)} \tag{5}$$

where $R_{th,cond}$, $R_{th,conv}$, and $R_{th,rad}$ are the conduction, convection, and radiation resistances ($K \cdot W^{-1}$); L is path length (m); k is thermal conductivity ($W \cdot m^{-1} \cdot K^{-1}$); A is area ($m^2$); h is the convection coefficient ($W \cdot m^{-2} \cdot K^{-1}$); ε is surface emissivity (dimensionless); σ is the Stefan-Boltzmann constant ($W \cdot m^{-2} \cdot K^{-4}$); and Tref is the radiative reference temperature (K).

Figure 2 maps the physical motor in Figure 1 to the thermal network. The principal heat sources are the winding, stator core, and permanent magnets. The five main solid nodes are the winding, stator core, rotor permanent magnet, bearing, and housing; internal air and the ambient environment provide additional thermal nodes and boundary conditions. Thermal resistances represent solid conduction, contact conduction, internal convection, air-gap heat transfer, and heat rejection from the housing and end cap.

DOI
10.3384/ecp223409

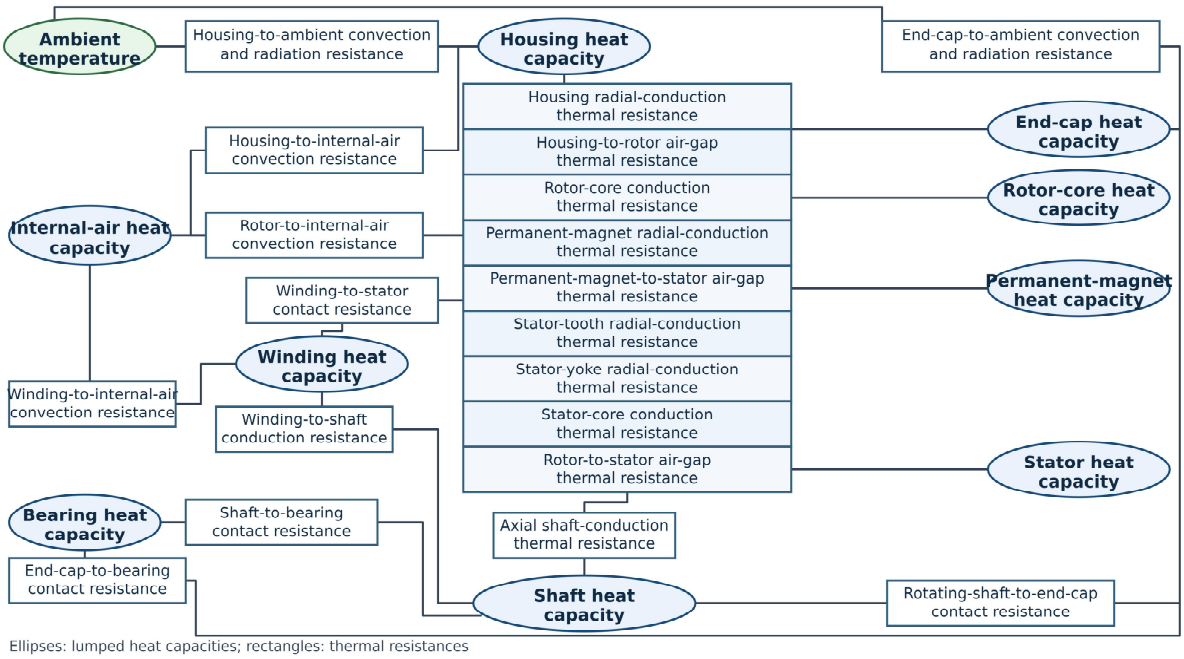


Figure 2. Equivalent thermal-resistance network of the motor.

3 Modelica Software Simulation

3.1 Software Introduction

Modelica is an equation-oriented, object-oriented language for multidomain physical modeling. Its declarative formulation is well suited to thermal networks because component energy balances and constitutive heat-transfer relations can be assembled without prescribing a manual computational sequence.

Modelica was selected for four reasons. First, its equation-based formulation directly represents conduction, convection, radiation, and nodal energy balances. Second, reusable custom components allow thermal resistances, heat capacities, and loss sources to be parameterized consistently. Third, automatic symbolic processing and code generation support fast repeated simulations. Fourth, the component structure can be extended later to include additional electrical, mechanical, or fluid-domain effects.

3.2 Thermal Network Model

Figure 3 shows the implemented Modelica network. The reduced-order model contains three principal heat sources (winding copper loss, permanent-magnet eddy-current loss, and stator-core loss) and the dominant heat-transfer paths through the shaft, bearing, internal air, stator, rotor, end cap, and housing. Heat rejection from the outer surfaces includes convection and radiation.

The network assigns a resistance and, where appropriate, a heat capacity to each retained component or interface. Particular attention is given to the bearing oil film and shaft contact, radial conduction through the laminated stator yoke, the rotating air gap, and the temperature-dependent winding loss. These terms are potentially influential and are therefore identified separately below.

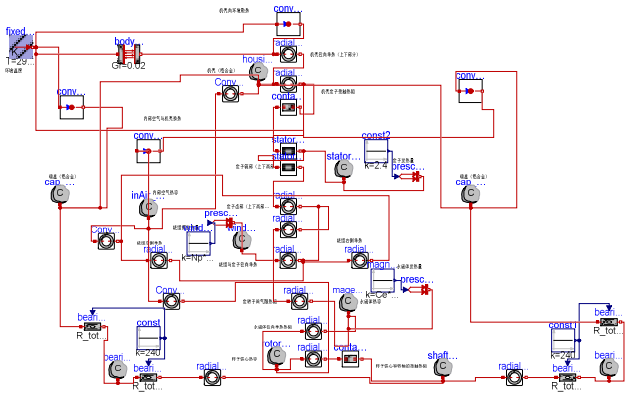


Figure 3. Modelica implementation of the motor thermal network.

The bearing-to-shaft path is represented by an oil-film resistance and an axial shaft-conduction resistance. Equation (6) is an empirical reduced-order correction, not a first-principles heat-transfer law. It follows from assuming that the oil-film thermal conductance increases linearly with the magnitude of rotational speed:

$$R_{oil} = R_{oil,base} \frac{1}{(1 + F_{speed} \times |\omega|/100)} \quad (6)$$

$$R_{shaft,axial} = \frac{L_{shaft}}{\pi {R_{shaft}}^2 k_{shaft}} \quad (7)$$

where R_{oil} and $R_{oil,base}$ are the operating and zero-speed oil-film resistances (K·W^{-1}); F_{speed} is a dimensionless empirical slope referenced to 100 rad·s^{-1}; ω is shaft angular speed (rad·s^{-1}); $R_{shaft,axial}$ is axial shaft resistance (K·W^{-1}); L_{shaft} and R_{shaft} are shaft length and radius (m); and k_{shaft} is shaft thermal conductivity (W·m^{-1}·K^{-1}). F_{speed} is motor- and lubricant-specific and is calibrated rather than transferred unchanged to another design.

Radial conduction through the stator yoke is represented by the cylindrical-wall relation:

$$R_{cyl,radial} = \frac{ln(r_{outer}/r_{inner})}{2\pi k_{eff} L} \quad (8)$$

$$k_{eff} = F_{stack} \cdot k_{Fe} + (1 - F_{stack}) \cdot k_{air} \quad (9)$$

where k_{eff} is the effective in-plane thermal conductivity of the laminated stack (W·m^{-1}·K^{-1}); r_{outer} and r_{inner} are the outer and inner yoke radius (m); L is active axial length (m); F_{stack} is the measured lamination stacking factor (dimensionless); and k_{Fe} and k_{air} are the thermal conductivities of electrical steel and interlaminar air (W·m^{-1}·K^{-1}). Equation (8) uses ln(r_{outer} / r_{inner}), as required by radial conduction through a cylindrical wall.

The air-gap convection coefficient is obtained from the Nusselt and Taylor relations:

$$Nu = \frac{2h_{air}\delta}{k_{air}} \quad (10)$$

$$Ta = \frac{\omega^2 r_g \delta^3}{v^2 F_g} \quad (11)$$

where Nu and Ta are dimensionless; h_{air} is the air-gap convection coefficient (W·m^{-2}·K^{-1}); δ is air-gap thickness (m); k_{air} is air thermal conductivity (W·m^{-1}·K^{-1}); v is air kinematic viscosity (m^2·s^{-1}); F_g is a dimensionless empirical correction factor; and r_g is the mean air-gap radius (m). Thus, hair is calculated from the selected Nu-Ta correlation and is not an independently fitted constant.

The principal internally generated heat rates considered here are winding copper loss and permanent-magnet eddy-current loss. The winding loss is:

$$W_{coil} = \frac{N_p i^2 \theta L_{av} N_s}{{N_b}^2 N_a A_{cu}} (1 + \alpha \Delta T_{coil}) \quad (12)$$

where W_{coil} is winding heat generation (W); θ is copper resistivity at the reference temperature (Ω·m); i is phase current (A); L_{av} is average turn length (m); A_{cu} is conductor cross-sectional area (m^2); N_p is the number of phases; N_s is the number of series turns per phase; N_a is the number of parallel windings; N_b is the number of parallel branches; α is the copper temperature coefficient (K^{-1}); and ΔT_{coil} is the winding-temperature rise from the resistivity reference temperature (K).

The permanent-magnet eddy-current heat generation is estimated as:

$$W_{magnet} = \frac{\pi^2}{6\beta}(dfB)^2 V_{magnet} \quad (13)$$

where W_{magnet} is permanent-magnet heat generation (W); f is magnetic-field frequency (Hz); V_{magnet} is total magnet volume (m3); β is magnet electrical resistivity (Ω m); B is magnetic-flux-density amplitude (T); and d is the effective magnet thickness normal to the induced-current path (m).

3.3 Parameter Identification and Calibration

The model parameters are identified in three groups. Geometric quantities (dimensions, areas, volumes, and F_{stack}) are obtained from the production-motor drawings or direct measurements. Material properties (density, heat capacity, thermal conductivity, electrical resistivity, and temperature coefficients) are taken from supplier data at the stated reference temperature, supplemented by standard literature values when supplier data are unavailable. Operating inputs (ambient temperature, current, and speed) are taken from the test record. The calculated quantities hair and R_{oil} then vary with speed through Eqs. (6), (10), and (11).

The uncertain empirical quantities—$R_{oil,base}$, F_{speed}, F_g, interfacial contact resistances, and outer-surface emissivity — are calibrated within physically plausible bounds by minimizing the sum of squared differences between measured and simulated housing temperature. Geometry and tabulated material properties remain fixed during this step. The same motor dataset is used for this parameter adjustment and for the agreement curve in Figure 4; consequently, the reported 5.9% value is an in-sample agreement metric, not independent cross-validation. For another motor, the empirical quantities must be re-identified before prediction accuracy is assessed.

4 Experimental Verification and Parameter Influence Analysis

4.1 Experimental Comparison for One Mass-Produced Motor

Housing temperature was measured on the mass-produced motor during a 60-min operating test, from start-up toward thermal equilibrium. A surface temperature sensor was placed at the housing location represented by the housing node, and the measured ambient temperature and operating inputs were imposed in the simulation. Only housing temperature was used for the quantitative comparison; internal winding, stator, and magnet temperatures were not measured in this test.

DOI 10.3384/ecp223409

Figure 4 compares the measured and simulated housing-temperature histories. Both increase rapidly during the first 30 min and approach a slower rate of change after approximately 50 min. The largest pointwise relative discrepancy is 5.9% at 40 min. This value characterizes the housing-temperature trace of the single tested motor and operating case; it is not a maximum error bound for internal temperatures, other operating conditions, or other motor types. Remaining differences can arise from ambient-temperature variation, nonuniform losses, contact-resistance uncertainty, and the lumped representation of spatial temperature gradients.

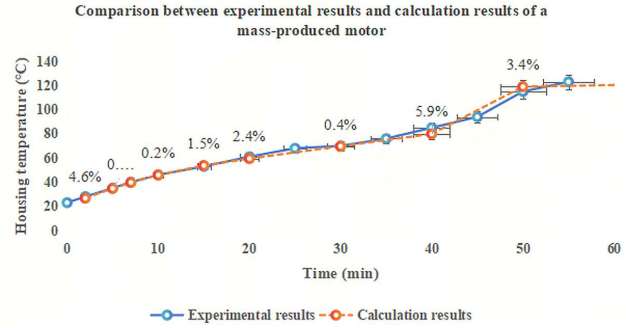


Figure 4. Measured and simulated housing temperature for the tested motor.

4.2 Influence of Simulation Parameters

This section examines how heat-transfer power, shaft speed, coil current, housing emissivity, and pulse-current excitation affect the predicted nodal temperatures. The results are sensitivity calculations for the calibrated model; they should not be interpreted as additional experimental validation cases.

Figure 5 summarizes the instantaneous heat-transfer-power distribution at 300 s. The stator-core path accounts for 26% of the reported total, followed by heat rejected from the housing to the environment (24%), heat transfer through the housing (18%), the end cap (10%), the winding (9%), the rotor core (7%), and the shaft (6%). These values describe the model's instantaneous energy-flow partition at the selected time and operating point.

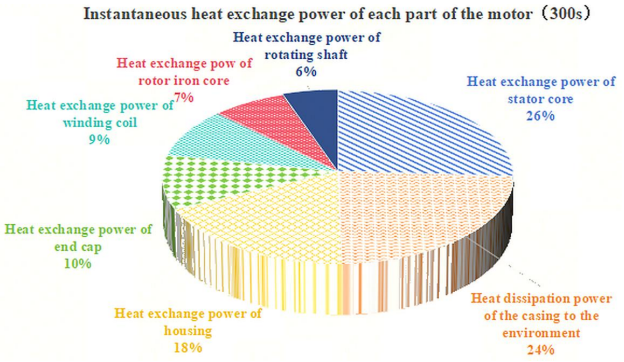


Figure 5. Instantaneous heat-transfer-power distribution at 300 s.

Figure 6 shows the instantaneous heat-transfer powers over 9000 s. The internal heat generation rises rapidly after start-up. Heat rejection from the housing increases more slowly because energy must first propagate through the thermal network; it then approaches the generated heat rate as the system tends toward thermal equilibrium. The other internal heat-flow rates decay toward their steady values as the nodal temperature differences stabilize.

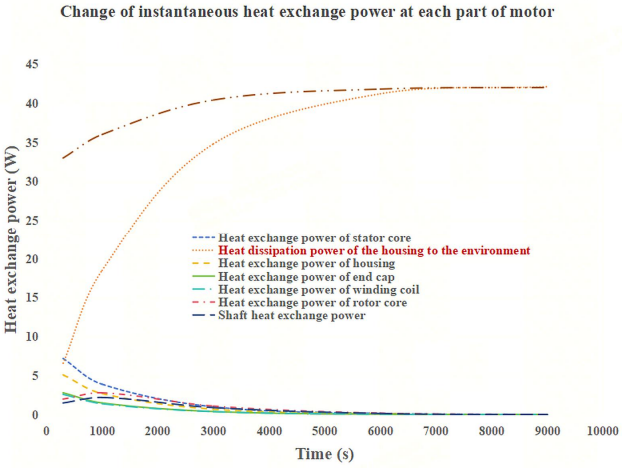


Figure 6. Transient heat-transfer powers of the modeled motor components.

Figure 7 shows component temperatures at several shaft speeds. In the model, permanent-magnet temperature increases slightly with speed because the speed-dependent magnet loss increases, whereas housing, stator-core, and winding temperatures decrease modestly as internal heat transfer is enhanced. This opposing behavior illustrates that speed affects both heat generation and heat rejection. The quantitative trend depends on the empirical speed and air-gap parameters and should be recalibrated for a different bearing, lubricant, air gap, or rotor configuration.

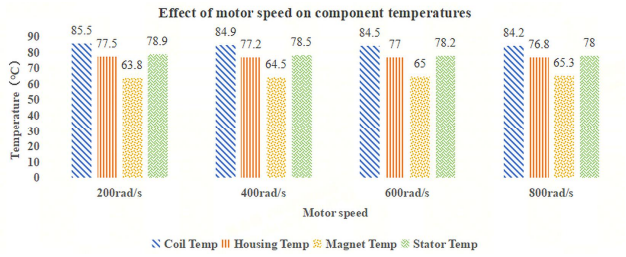


Figure 7. Predicted effect of shaft speed on component temperature.

Figure 8 shows the predicted temperatures at coil currents from 2 to 10 A. All four reported component temperatures increase with current, and the winding shows the largest increase because copper loss is proportional to i^2 in Eq. (12). Over each 2-A increment, the predicted temperature increase ranges from approximately 3 to 10 °C, depending on the component and operating point. The differences reflect the distinct heat capacities and thermal paths of the nodes.

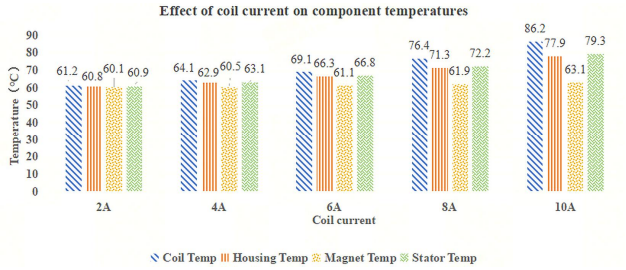


Figure 8. Predicted effect of coil current on component temperature.

Figure 9 shows the effect of housing and end-cap surface emissivity. Increasing emissivity increases radiative heat rejection and therefore reduces the predicted temperatures of all reported components. The winding remains the hottest node and exhibits the largest absolute decrease over the investigated range.

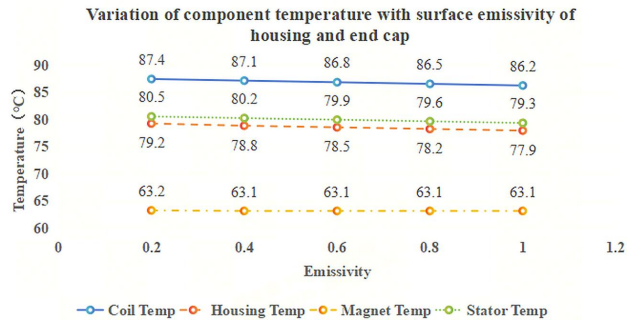


Figure 9. Predicted effect of surface emissivity on component temperature.

Figure 10 presents a simulated pulse-current case consisting of three 5-s, 10-A pulses separated by 5-s intervals. The winding responds most rapidly, increasing from approximately 60 °C to 70 °C during the first pulse and partially cooling after the current is removed. The permanent-magnet temperature changes little over 30 s because of its larger effective thermal time constant. The housing temperature increases from approximately 60 °C to 63 °C, while the stator-core temperature rises more gradually to about 65 °C. This calculation demonstrates the model's expected transient behavior; no experimental pulse-test data are available here, so it is not an independent validation.

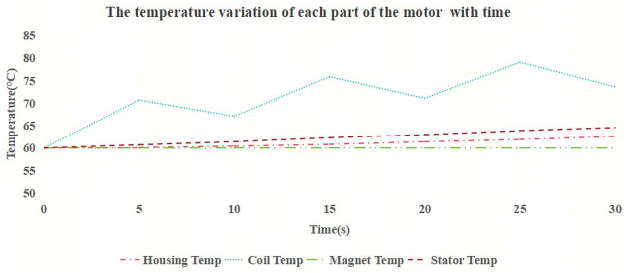


Figure 10. Predicted component temperatures under pulse-current excitation.

Table 1 reports five solver-timing runs. N is the run index, Ta is the prescribed physical simulation horizon, and Tcpu is the CPU time reported for numerical integration. The simulations were executed in Dymola with the DASSL integrator using one CPU core. Tcpu excludes model translation and compilation; an end-to-end simulation of 1800 s of physical time, including translation and result handling, completed in approximately 2 s on the test workstation. Because absolute timings depend on software version, solver tolerances, processor, and operating system, these values are intended to document the order of magnitude of the compact model's computational cost rather than provide a hardware-independent speed guarantee.

Table 1. Solver CPU time for the compact Modelica thermal model.

N	Physical time, Ta (s)	CPU time, Tcpu (s)
1	*1000*	*0.007*
2	*5000*	*0.007*
3	*10000*	*0.008*
4	*100000*	*0.008*

DOI 10.3384/ecp223409

N	Physical time, Ta (s)	CPU time, Tcpu (s)
5	*200000*	*0.008*

5 Conclusion

A compact Modelica thermal-network model was developed for the tested external-rotor low-voltage PMSM. The model represents the dominant heat sources, heat capacities, and conductive, convective, and radiative paths, including nontrivial bearing-oil and rotating-air-gap terms. The parameter-identification procedure separates geometry- and material-based inputs from empirically calibrated quantities.

For one mass-produced motor and operating case, the simulated housing-temperature trace differs from the measured trace by at most 5.9% at the sampled points. An end-to-end 30-min simulation completes in approximately 2 s under the reported setup. The sensitivity calculations show that current, speed, and surface emissivity affect the predicted temperature response through different heat-generation and heat-rejection mechanisms.

The present evidence supports use of the model as a fast reduced-order analysis tool for the tested PMSM after motor-specific parameter identification. It does not yet establish accuracy for other low-voltage motors, induction machines, cooling arrangements, or internal component temperatures. Future work will use separate calibration and validation datasets, add winding and magnet temperature measurements, and test multiple motor types and operating cycles before broader applicability is claimed.

Acknowledgements

-

References

Gartner. “Predicts that the number of electric vehicles in use worldwide will reach 116 million in 2026” (2025-12). URL: https://www.gartner.com/cn/newsroom/press-releases/2026-ev-forecast.

Kouchakipour N, Negari S, Mitra G, et al (2024-12). “White Paper-Electric Vehicle Charging Systems--Interoperability, Safety, and Testing Considerations”. In: *Electric Vehicle Charging Systems--Interoperability, Safety, and Testing Considerations.* IEEE, pp. 1-39. ISBN: 979-8-8557-1609-2.

Manalac E J, Park M H, Kim S C (2025-3). “Transient thermal network development and evaluation of rotational oil-spray cooled in-wheel module for electric vehicles under drive cycle conditions”. In:*Applied Thermal Engineering*. Elsevier, pp. 262:125211. DOI: 10.1016/j.applthermaleng.2024.125211.

Sen A, Singh B, Mahtani K, et al (2025-6). “Optimized design of a permanent magnet brushless DC motor for solar water-pumping applications”. In: *Results in Engineering*. Elsevier, 26: 104633. DOI: 10.1016/j.rineng.2025.104633.

Wallscheid O (2021-6). “Thermal monitoring of electric motors: State-of-the-art review and future challenges”. In: *IEEE Open Journal of Industry Applications*. IEEE, pp. 2: 204-223. DOI:10.1109/OJIA.2021.3091870.

Yan Y, Mao C, Chen L (2024-9). “Multi-physical field coupling analysis of flat wire motor”. In: *International Journal of Automotive Technology*. Springer, pp. 26(2): 475-489. DOI:10.1007/s12239-024-00155-y.

Nogal Ł, Magdziarz A, Rasolomampionona D D, et al (2021-8). “The laboratory analysis of the thermal processes occurring in low-voltage asynchronous electric motors”. In: *Energies*. Mdpi, pp. 14(8): 2056. DOI:10.3390/en14082056.

Nonneman J, T Jollyn I, Paepe M D (2021-8). “Accurate time-efficient thermal modeling and discretization methodology applied to electric machines”. In: *Heat Transfer Engineering*. Taylor & Francis, pp. 46(8): 693-712. DOI: 10.1080/01457632.2024.2337973.

Nonneman J, T Jollyn I, De Paepe M (2024). “Enhancing thermal modelling of electric machines: steady-state verification of a novel methodology”. In: *Journal of Physics: Conference Series*. IOP Publishing, pp. 2766(1): 012111. DOI: 10.1088/1742-6596/2766/1/012111.

Nonneman J, De Paepe M, T Jollyn I (2024). “Advancing electric machine lumped parameter thermal modeling: A novel spatial and temporal discretization methodology”. Ichmt Digital Library Online. Begel House Inc. DOI: 10.1615/ICHMT.2024.CHT-24.

Urbieta J G, Rodríguez B, Rodríguez A J, et al (2023-9). “Sensitivity analysis of lumped-parameter thermal networks for the experimental calibration of emotor models”. In: *IEEE Transactions on Transportation Electrification*. IEEE, pp. 10(3): 6210-6220. DOI: 10.1109/TTE.2023.3331097.

Assaad B, El kadri Benkara K, Friedrich G, et al (2017). “Reducing the complexity of thermal models for electric machines via sensitivity analyses”. 2017 IEEE Energy Conversion Congress and Exposition (ECCE). IEEE, pp. 4658-4665. DOI: 10.1109/ECCE.2017.8096795.

Kral C, Haumer A, Plainer M (2005). “Simulation of a thermal model of a surface cooled squirrel cage induction machine by means of the SimpleFlow-library”. Proceedings of the 4th International Modelica Conference, Hamburg, pp. 213-218.

Kral C, Haumer A, Bäuml T (2008). “ Thermal model and behavior of a totally-enclosed-water-cooled squirrel-cage induction machine for traction applications ” . IEEE Transactions on Industrial Electronics, 55(10): 3555-3565. DOI: 10.1109/TIE.2008.927242.

DOI
10.3384/ecp223409

Model-Based Thermal Management for Electric Vehicles: Modeling, Simulation, and Control

Chengen Li[1†] Zhixu Chen[1] Li Xie[1] Yuxi Liu[1] Xiaohu Wang[1] Yuanhao Piao[1]

[1]BYD Auto Industry Company Limited, China, {lastname.firstname}@byd.com
[†]Corresponding Author

Abstract

With the continuous growth in global sales and market penetration, electric vehicles are increasingly required to operate under extreme temperatures, high altitudes, and humid or saline conditions, posing substantial challenges to thermal management. To address this challenge, this paper proposes a 1D model-based framework for integrated thermal management system (ITMS), enabling the unified design of modeling, simulation, and control. A vehicle-level thermal model is developed in Modelica to capture the coupled dynamics of the refrigeration cycle, cabin, battery, and motor, and is driven by driving cycles to simulate diverse operating conditions. The functional mock-up interface (FMI) is adopted to bridge the physical simulation model and the co-simulation environment, establishing a control interface that supports multivariable coordination and facilitates the integration of data-driven methodologies, such as reinforcement learning (RL). Within this framework, a RL-based control model is implemented to coordinate the system's actuators. Under dynamic driving cycles, we evaluate the learned RL policy against a fixed-parameter baseline and a rule-based controller. The results validate the feasibility of the proposed integrated architecture in coupling physics-based level modeling with adaptive data-driven control, providing a systematic approach for thermal management design in electric vehicles.

Keywords: Modelica, Model-Based Systems Engineering, Electric Vehicle, Thermal Management, Reinforcement Learning

1 Introduction

The rising complexity of vehicle systems has prompted the automotive industry to transition from traditional document-driven development models to systems engineering approaches, thereby enabling more efficient and traceable development processes. In recent years, model-based systems engineering (MBSE) has gradually emerged as an important paradigm to replace traditional document-centric systems engineering. By using unified system models to describe requirements, design, and verification processes, MBSE enables engineering teams to manage increasingly complex system structures from a global perspective and perform design validation during the early stages of development, thereby significantly improving system development efficiency and reliability.

In electric vehicle development, the vehicle thermal management system involves multiple subsystems, including battery temperature control, drive motor cooling, passenger cabin air-conditioning load, and waste heat recovery. These subsystems exhibit complex thermal coupling relationships and span multiple engineering domains such as mechanical engineering, electrical engineering, and thermodynamics. Traditional thermal management development processes often lack systematic requirement management and model integration mechanisms. In addition, a unified structure linking thermal design requirements, simulation models, and experimental validation is usually absent. This problem is particularly prominent during the conceptual design and early development stages of vehicles, because the system architecture remains under development while the selection of thermal management solutions may have long-term impacts on overall vehicle performance.

In practical engineering scenarios, requirement management, architecture modeling, and 1D thermal simulation are typically maintained on different software platforms. Such fragmentation among tools makes it difficult to establish clear mapping relationships between requirements and models, thereby limiting system-level verification capabilities. To achieve a truly integrated systems engineering process, it is necessary to establish a complete data chain from system requirements to actuators and integrate modeling, simulation, and control development into a unified framework.

In this context, the combination of detailed 1D thermal modeling and data-driven control methods provides a new technological pathway for electric vehicle thermal management development. On the one hand, 1D physical modeling can rapidly evaluate the performance of different thermal management architectures under multiple operating conditions. On the other hand, data-driven methods such as reinforcement learning (RL) can learn optimal control strategies by utilizing large-scale data generated from simulations.

2 Motivation

The vehicle thermal management system plays a crucial role in the operation of electric vehicles. Its primary task is

to ensure that key components remain within safe temperature ranges while reducing system energy consumption and maintaining passenger comfort. With the continuous growth in global sales and market penetration, electric vehicles are increasingly required to operate under extreme temperatures, high altitudes, and humid or saline conditions. Different climatic conditions impose significantly different requirements on thermal management systems. For example, low-temperature environments require rapid battery heating, while high-temperature environments demand efficient heat dissipation capabilities. Therefore, the design of thermal management systems must be able to adapt to more diverse operating conditions.

As illustrated in Figure 1, the research objectives of the integrated thermal management system (ITMS) mainly focus on three core aspects, namely energy efficiency, thermal safety, and cabin comfort. These objectives significantly influence overall vehicle performance across diverse operating conditions. Energy efficiency optimization primarily aims to reduce overall energy consumption and extend driving range. Thermal safety focuses on maintaining stable operation of the battery, motor, and electronic control systems within safe temperature boundaries. Cabin comfort relies on precise temperature regulation and coordinated air conditioning control to ensure a stable thermal environment for occupants. To support electric vehicle operation across diverse global climate regions, the ITMS must demonstrate strong environmental adaptability, which requires systematic operating-condition calibration and comprehensive data collection for modeling and analysis. Real vehicle road tests and 1D simulation modeling serve as the primary data sources during this phase. Once representative operating conditions have been characterized and the system model has been validated, the design of ITMS control strategies plays a critical role in achieving optimal performance.

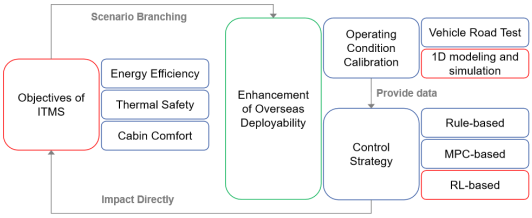


Figure 1. Research objectives and approach overview

Currently, thermal management control strategies widely adopted in industry are typically calibrated through standard driving cycles or real-vehicle road tests, which provide reliable validation for representative operating conditions. With the continuous evolution of vehicle usage patterns, environmental diversity, and electrification trends, new complex operating scenarios are increasingly emerging. In this context, real-vehicle validation exhibits relatively limited agility for iterative strategy refinement and involves considerable time and resource investments.

1D physical modeling based on Modelica provides an effective approach to address this issue (Bouvy et al. 2012; Batteh et al. 2014; Lee et al. 2025; Yadav et al. 2026). By uniformly describing the battery, motor, refrigeration cycle, and cabin thermal environment, the overall thermal behavior of the vehicle can be comprehensively represented within a single simulation model. Meanwhile, by parameterizing environmental temperature, altitude, and driving behavior, a large number of different operating scenarios can be reproduced in simulation, thereby compensating for the limited coverage while reducing the resource investments required for real-world testing.

The functional mock-up interface (FMI) can be adopted as a standardized co-simulation protocol to bridge the physical simulation environment and control strategies. By enabling structured and continuous data exchange, this interface naturally accommodates data-driven methodologies. Taking RL as a representative paradigm, continuous state-action exchange enables RL methods to be directly employed for closed-loop policy training. Conducting this exploration within a virtual environment inherently alleviates safety concerns, as exploration could cause thermal runaway or component damage if applied directly to real vehicles. In the thermal management control problem, objectives such as safe temperature limits, energy consumption minimization, and passenger comfort can be unified into reward functions, allowing the control strategy to approach optimal behavior during the learning process.

Building upon the proposed framework, this work mainly focuses on the following key aspects:

- Development of a 1D vehicle-level model for thermal dynamics simulation.
- Construction and integration of an FMI-based co-simulation interface for model-control coupling.
- Implementation and evaluation of an RL-based control model for actuator control, validated against a fixed-parameter baseline and a rule-based controller.

3 Modeling

Built on the MWorks.Sysplorer platform, the proposed framework integrates key components from customizable Modelica libraries to seamlessly couple electric vehicle driving cycles with the ITMS, thereby establishing a unified system-level model.

3.1 Key Subsystems

3.1.1 Cabin

The passenger cabin model describes the in-cabin thermal environment and the heat exchange process between the cabin and the external environment. The model comprehensively considers factors such as occupant metabolic heat, humidity dynamics, and air quality variations, and describes the thermal inertia of cabin air and interior materials through a thermal capacitance model. In terms of human thermal load modeling, the model introduces

DOI
10.3384/ecp223417

both sensible heat and latent heat generated by occupant metabolism and considers humidity and carbon dioxide emissions produced by occupant respiration, enabling a more realistic representation of the impact of occupants on the cabin environment, as shown in Figure 2.

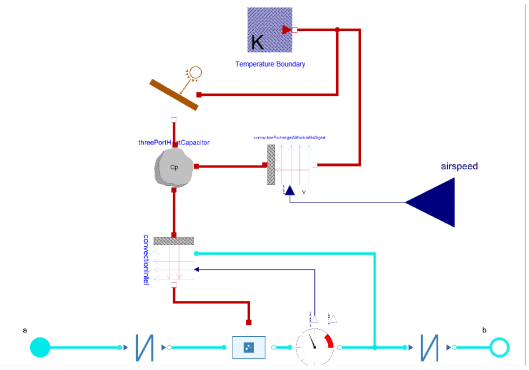


Figure 2. Architecture of the cabin model

In terms of environmental heat exchange, the model not only considers ambient air temperature but also introduces solar radiation heat input. Solar radiation first acts on the outer surface of the vehicle and is then conducted into the passenger cabin through the vehicle body structure, producing a heating effect on the cabin air. Through the above modeling approach, the cabin model can provide reliable simulation data for thermal comfort evaluation and air-conditioning energy consumption analysis.

3.1.2 Motor

The motor model describes the electro-mechanical energy conversion process in the electric vehicle drivetrain and calculates the thermal load generated during operation. The model first receives the demanded torque signal from the vehicle control layer and, combined with motor speed information, obtains the maximum available torque under the current operating condition through lookup tables. The demanded torque is subsequently converted into mechanical output torque through a torque source module to drive the vehicle, as shown in Figure 3.

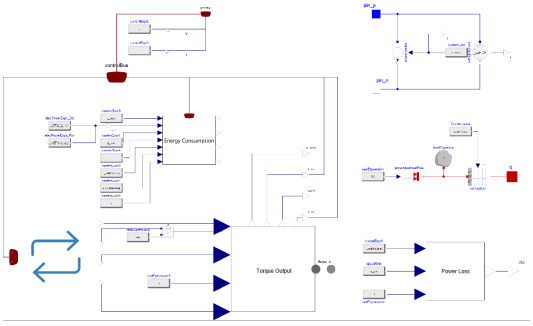


Figure 3. Architecture of the motor model

In terms of energy calculation, the model includes torque calculation, efficiency calculation, and energy loss calculation modules. The torque module performs interpolation in the torque-speed characteristic map to obtain actual output torque. The efficiency module calculates electro-mechanical conversion efficiency based on efficiency maps. The energy loss module computes system losses from the difference between input electrical power and output mechanical power and converts them into heat. This heat is then transferred to the ITMS to simulate motor cooling demand. The model outputs key parameters including torque, speed, electrical power, mechanical power, and thermal loss, which are used both for vehicle dynamics calculation and as thermal load inputs for ITMS.

3.1.3 Battery

The battery model mainly consists of a state-of-charge (SOC) calculation module, an open-circuit voltage (OCV) lookup model, and an internal resistance lookup model. The model reads battery current through an interface and computes SOC via integration. Then, the corresponding electromotive force is obtained from the OCV lookup table, and the internal resistance is determined from the resistance lookup model, as shown in Figure 4.

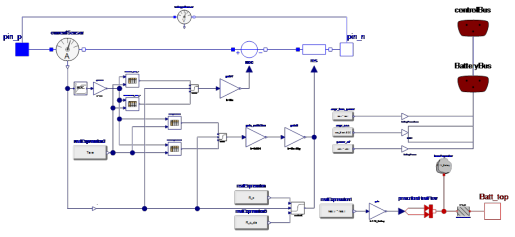


Figure 4. Architecture of the battery model

During charging and discharging operations, the electromotive force and internal resistance form an equivalent circuit to simulate battery electrical behavior. Electrical losses during operation are converted into heat generation and fed into the ITMS. Meanwhile, battery temperature affects OCV and internal resistance, forming a coupled relationship among temperature, current, and voltage.

3.1.4 Driving Cycle

The driving cycle model simulates vehicle operation under predefined driving cycles, as shown in Figure 5. It is based on a proportional-integral-derivative (PID) speed control system. Predefined cycles such as CLTC, WLTC, NEDC, and EPA are used as reference speed inputs.

The difference between actual and target speed is calculated and fed into a PID controller. The controller output is converted into acceleration commands after saturation and transmitted to the vehicle dynamics model. When speed exceeds thresholds, braking logic is triggered automatically. This enables accurate tracking of driving cycles and provides stable inputs for energy consumption and thermal behavior analysis.

3.2 Integrated Thermal Management System

In the full-vehicle thermal management simulation model, multiple heat sources, including the motor, battery, and

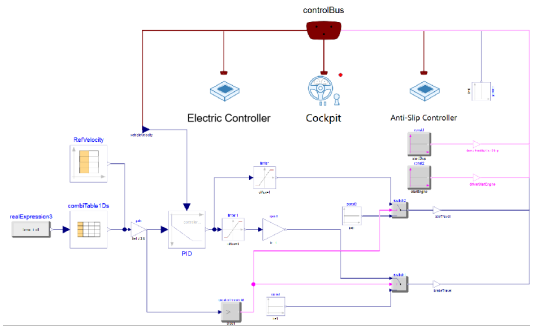


Figure 5. Architecture of the driving cycle model

cabin, are integrated into the ITMS. The system adopts a dual-loop architecture consisting of a refrigerant loop and a coolant loop to achieve heat distribution and regulation. The refrigerant loop is driven by compressor work to circulate refrigerant, utilizing refrigerant phase transitions for heat transfer, while the coolant loop flows through the battery and motor to perform heat transfer, temperature equalization, and waste heat recovery. The two loops exchange heat through a chiller, forming cross-loop coupling and establishing a complete heat transfer path. The condenser and radiator are integrated into the front-end module of the vehicle, where airflow and active cooling form the heat exchange channel with the external environment, as shown in Figure 6.

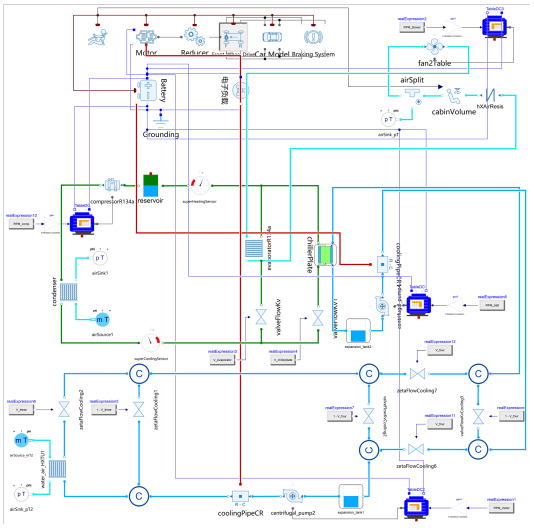


Figure 6. Architecture of the proposed ITMS

On this basis, the passenger cabin model is further refined by not only considering human metabolic heat and humidity loads and solar radiation effects but also introducing the influence of vehicle driving wind speed on system heat dissipation performance. The simulated airflow velocity signal is simultaneously applied to the cabin convective heat transfer calculation and the condenser heat dissipation model, enabling the simulation to reflect the influence of airflow on system heat exchange efficiency during vehicle driving.

Key components such as compressors, water pumps, and fans are modeled using lookup-table-driven models. Their operating power consumption is fed back to the battery model, thereby enabling energy consumption estimation. In addition, within the coolant loop, various flow configurations can be achieved through coordinated control of the expansion valves in the refrigerant loop, as well as the three-way and four-way valves in the coolant loop. This enables multiple refrigerant and coolant routing configurations, as illustrated in Figure 7. Based on this multi-loop coupled architecture, the system is capable of dynamically allocating thermal capacity to satisfy the temperature requirements of different components.

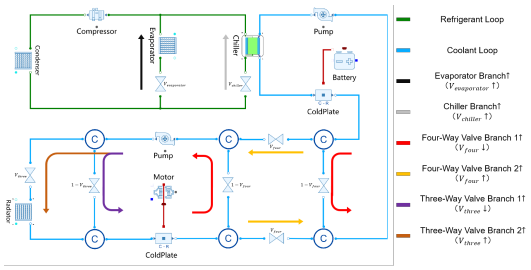


Figure 7. Operating modes of refrigerant and coolant loops

In summary, this 1D simulation model integrates vehicle dynamics, multi-heat-source coupling, fluid loop control, and environmental interaction, establishing a virtual verification foundation for electric vehicle ITMS.

4 Control

4.1 Control Interface

As shown in Figure 8, this paper presents a co-simulation and control framework. In this configuration, the control strategy interacts with the Modelica simulation model via the FMI standard. During each control cycle, the simulation environment provides system states to the external controller, which in turn generates control commands and feeds them back to the simulation model for execution. Additionally, the control model can adopt various strategies, such as rule-based and data-driven approaches, thereby enabling flexible comparative evaluation.

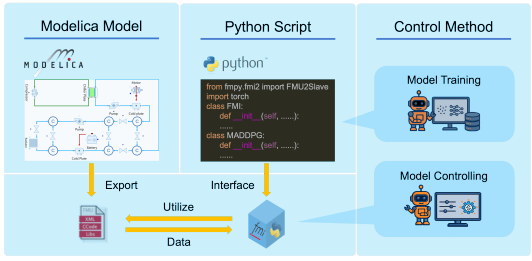


Figure 8. Co-simulation and control framework

A physical model is first developed in Modelica and exported as a functional mock-up unit (FMU) compliant with the FMI standard. A Python script then leverages

DOI
10.3384/ecp223417

the FMI standard to establish bidirectional communication with the FMU. The control algorithm utilizes this data stream for training, ultimately issuing control commands back to the simulation environment through the same standardized channel, enabling integration of model development, data exchange, and control execution.

This study implements an interface using the open-source `FMPy` library to instantiate and manage the FMU, effectively bridging the simulation environment with control models. The interaction pipeline strictly adheres to the FMU lifecycle. Initially, the FMU archive is unpacked and its model description is parsed to extract variable metadata and configure runtime parameters. During instantiation, a co-simulation instance is created, followed by an initialization phase where communication step sizes, tolerances, and initial conditions are applied. Once initialized, the simulation advances through discrete communication steps. At each step, control actions from the controller are mapped to FMU input ports, while system outputs are retrieved via standardized value references, ensuring synchronized bidirectional data exchange. To accommodate episodic RL training, the environment reset routine terminates the simulation, deallocates computational resources, clears temporary runtime artifacts, and re-executes the instantiation pipeline. This guarantees a deterministic initial state for each episode, as illustrated in Figure 9.

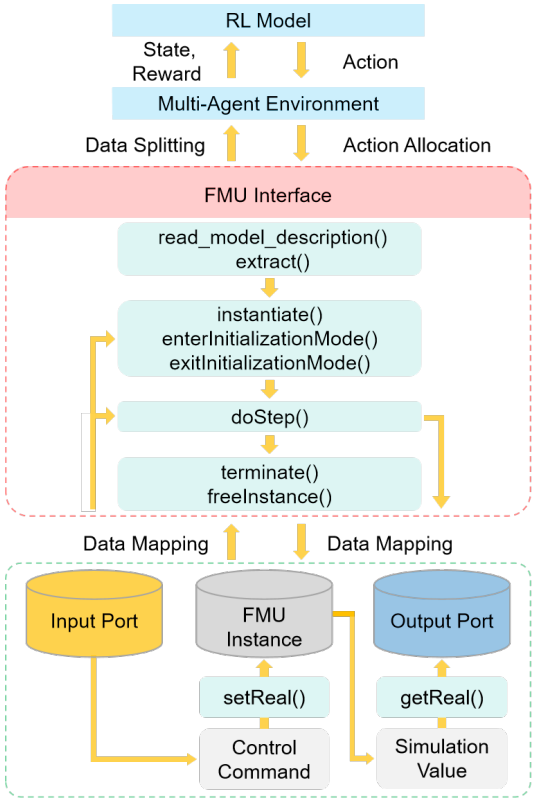


Figure 9. FMI-based co-simulation interface

The controllable parameters of the FMU span multiple actuation channels, including compressor speed, coolant pump speeds for battery and motor circuits, blower flow, and valve positions for various branches. The controllable parameters and their fixed values are listed in Table 1.

Table 1. Controllable parameters of the FMU in this work

Symbol	Description	Value
$N_{\text{compressor}}$	Compressor speed (rpm)	$[0, 4000]$
$N_{\text{pump}}^{\text{battery}}$	Battery-circuit coolant pump speed (rpm)	$[0, 3000]$
$N_{\text{pump}}^{\text{motor}}$	Motor-circuit coolant pump speed (rpm)	$[0, 3000]$
N_{blower}	Blower flow (m^3/s)	$[0, 0.1]$
V_{three}	Three-way valve position	$[0, 1]$
V_{four}	Four-way valve position	$[0, 1]$
$V_{\text{evaporator}}$	Evaporator-branch valve position	$[0, 1]$
V_{chiller}	Chiller-branch valve position	$[0, 1]$

4.2 Control Strategies Analysis

The development of vehicle ITMS control strategies has evolved from rule-based and optimization-based control to data-driven approaches. Early methods primarily rely on PID control, which is simple in structure and easy to implement but exhibits limited performance in handling highly coupled and nonlinear systems. Subsequently, model predictive control (MPC) is introduced into ITMS, achieving tracking error minimization by optimizing future control sequences. However, MPC heavily depends on model accuracy and involves high computational complexity, making it difficult to meet real-time control requirements under complex operating conditions.

In recent years, RL has been an important approach for solving complex control problems. RL does not rely on accurate system models but instead shifts the intensive computational burden to an offline training phase, where it learns control policies through direct environment interaction, ultimately demonstrating strong adaptability in complex nonlinear systems. Existing studies have shown that RL can significantly reduce vehicle energy consumption while maintaining temperature safety constraints for key components, and it exhibits superior adaptability under dynamic conditions compared to traditional methods (Tan et al. 2019; Guo, Peng, He, et al. 2025; Guo, Peng, J. Wu, et al. 2025; Pang et al. 2025; C. Wu et al. 2025). However, it should be noted that detailed FMU simulations incur substantial computational costs, imposing higher requirements on the sample efficiency of RL methods. Therefore, in practical applications, appropriate RL paradigms must be selected based on available simulation resources.

Under conditions where FMU simulation incurs high computational cost and computational resources are rel-

atively limited, model-based RL can be adopted to improve sample efficiency. Such methods learn the system dynamics model and utilize it for planning or generating synthetic data, thereby reducing reliance on interactions with the real environment. A representative method is PETS (Chua et al. 2018), which models environmental dynamics uncertainty using an ensemble of probabilistic neural networks and combines trajectory sampling to perform model-based predictive control, significantly improving sample efficiency while maintaining decision performance. Furthermore, the MBPO (Janner et al. 2019) method integrates model learning with policy optimization by generating high-quality synthetic samples through short-horizon rollouts and mixing them with real data for policy updates, thereby improving training stability and convergence speed while mitigating the impact of model bias. In engineering practice, model order reduction techniques can be further employed to approximate detailed simulations, thereby constructing lower-cost training environments. Proxy modeling combined with RL can significantly reduce computational cost while maintaining acceptable control performance. However, its effectiveness strongly depends on the generalization capability and accuracy of the surrogate model. When model errors are large, policy learning may deviate from the true optimal solution. Therefore, online correction mechanisms or hybrid data training strategies are usually required to improve robustness.

Since FMU simulations involve complex numerical integration processes, the cost of environment interaction is significantly higher than that of conventional RL benchmark environments. When computational resources are relatively sufficient, model-free RL can be directly applied to optimize control strategies. These methods do not require explicit modeling of system dynamics but instead learn policies through repeated interactions with the environment, guided by reward signals. Among them, DDPG (Lillicrap et al. 2015) is a representative algorithm for continuous action spaces. It employs an actor–critic architecture to implement deterministic policy gradient updates and has demonstrated good performance in control tasks.

It should be noted that while MPC provides a principled optimization-based control framework, it typically relies on an explicit system dynamics model for prediction. Applying MPC within this co-simulation setup would require system identification or a surrogate model to approximate the black-box FMU behavior, introducing additional modeling complexity and potential model mismatch. In contrast, model-free RL requires no prior dynamics identification and learns directly through interaction with the FMU, making it more naturally aligned with the adopted co-simulation framework.

4.3 Experiments and Results

In all control experiments, only valve opening positions are adjusted as control variables, while other parameters are held constant at fixed operating values, validating adaptive flow redistribution as a foundational step toward full multi-actuator coordination. The compressor, battery pump, and motor pump speeds were set to 2000 rpm, 1000 rpm, and 1000 rpm, respectively, as reasonable engineering operating conditions. The ambient temperature was fixed at 40 °C, with initial temperatures of the cabin, battery, and motor set to 30 °C, 30 °C, and 60 °C, respectively. For validation, all simulations were conducted on the WLTC driving cycle using a variable-step DASSL solver with a relative tolerance of 1×10^{-4}.

The fixed-parameter baseline serves as a reference case where all four valve positions are held constant at 1, representing a simple open-loop configuration without active feedback control. This baseline illustrates the inherent coupling and interference among the three temperature zones under varying thermal loads.

4.3.1 Implementation

Two representative strategies are employed for ITMS temperature regulation, with implementation details provided in the following paragraphs.

Rule-based Control. To benchmark the effectiveness of rule-based control paradigms, a representative fuzzy logic-driven gain-scheduled PID controller is employed for ITMS temperature regulation, featuring independent control loops for each actuation valve. The architecture employs the absolute temperature error e and the rate of error change de as fuzzy inputs, online computing adjustments to the proportional, integral, and derivative gains, ΔK_p, ΔK_i, and ΔK_d, respectively, thereby realizing adaptive tuning of parameters. The inference engine utilizes triangular membership functions to map e and de onto corresponding linguistic fuzzy subsets, which are subsequently employed to formulate the scheduling rules. Following classical control theory, the core strategy increases K_p under large error conditions to accelerate dynamic response, enhances K_i in the small-error steady-state regime to eliminate residual error, and appropriately raises K_d when the error derivative is large to provide system damping that effectively suppresses overshoot and oscillation.

RL-based Control. Building upon the control interface introduced previously, we implement the DDPG algorithm to enable reward-driven temperature regulation. The agent operates with a fixed simulation timestep to adequately capture the slow thermal dynamics. The observation space s includes temperatures of the cabin T_{cabin}, battery $T_{battery}$, and motor T_{motor}, alongside ambient conditions, current actuator states, and key thermodynamic variables such as refrigerant subcooling and superheat degrees, while the action space a comprises four continuous valve opening position commands. The reward function is designed to balance temperature tracking accuracy with operational safety:

DOI
10.3384/ecp223417

$$
\begin{aligned}
r = &-w_1|T_{cabin} - T^{set}_{cabin}| \\
&- w_2|T_{battery} - T^{set}_{battery}| \\
&- w_3|T_{motor} - T^{set}_{motor}| - r_{penalty}
\end{aligned} \tag{1}
$$

where w_1, w_2, and w_3 are normalized weighting coefficients, and $r_{penalty}$ is a composite penalty term accounting for episode timeouts, violations of safe temperature bounds, and FMU integration errors.

The DDPG algorithm updates the actor policy $\pi_\theta(s)$ and critic $Q_\phi(s,a)$ by sampling a minibatch of N transitions from the replay buffer. The critic is updated by minimizing the temporal difference (TD) error:

$$
\mathscr{L}_Q = \frac{1}{N}\sum_i \left(r_i + \gamma Q_{\phi'}(s'_i, \pi_{\theta'}(s'_i)) - Q_\phi(s_i, a_i)\right)^2 \tag{2}
$$

where r_i is the reward received at step i, s_i and a_i are the state and action at step i respectively, s'_i is the next state after executing a_i, ϕ' and θ' denote the target network parameters, and $\gamma = 0.99$ is the discount factor balancing immediate and future rewards.

The actor is updated by deterministic policy gradient:

$$
\nabla_\theta J \approx \frac{1}{N}\sum_i \nabla_a Q_\phi(s_i, a_i)\big|_{a_i=\pi_\theta(s_i)} \nabla_\theta \pi_\theta(s_i) \tag{3}
$$

where J is the expected discounted return, γ is the discount factor, r_i is the reward at step i, $\nabla_a Q_\phi(s_i, a_i)$ is the gradient of the Q-value with respect to the action, and $\nabla_\theta \pi_\theta(s_i)$ is the gradient of the policy with respect to its parameters. Both actor and critic networks adopt identical four-layer MLP architectures with hidden layer dimensions of 512, 256, 128, and 64, utilizing ReLU activation functions. The training configuration employs a learning rate of 1×10^{-3}, a soft update coefficient $\tau = 1 \times 10^{-3}$ for target networks, a replay buffer capacity of 1×10^4, a batch size of 128, and an exploration noise scale of 0.1.

Figure 10 illustrates the reward trajectory. During the early training phase, the policy frequently incurs substantial penalties due to temperature bound violations and FMU numerical integration failures, resulting in highly volatile reward signals with frequent sharp drops. As training progresses, the policy gradually learns to avoid these unsafe regions and improve temperature tracking accuracy. The policy eventually converges to a stable plateau with minor oscillations, reflecting the inherent exploration-exploitation trade-off. This overall convergence profile validates the stability and efficacy of the proposed training framework.

4.3.2 Performance Evaluation

Temperature regulation performance is quantified using the mean absolute error (MAE) for cabin, battery, and motor, as summarized in Table 2.

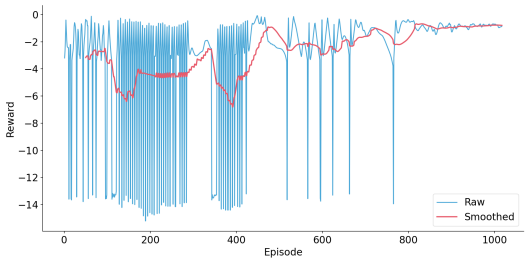


Figure 10. Training reward trajectory

Table 2. Comparison of temperature MAE (↓) across strategies

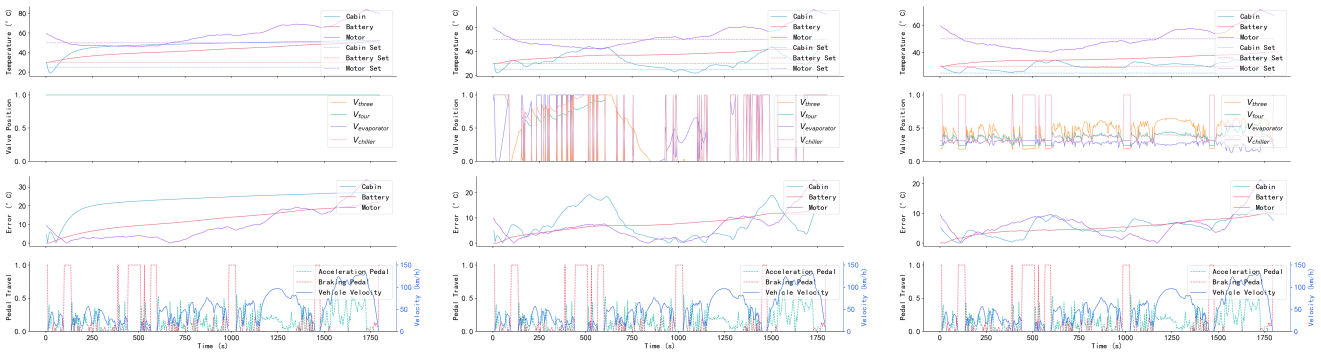
Strategy	Cabin	Battery	Motor
Fixed-parameter	22.87	12.69	10.34
Rule-based	7.88	7.75	6.85
RL-based	**5.63**	**5.51**	**6.45**

The fixed-parameter strategy (Figure 11a), where all four valve positions are held constant at 1, yields the worst performance across all three components. The cabin temperature exhibits sustained deviation from the setpoint, indicating that the single fixed configuration cannot adapt to the dynamically varying thermal loads imposed by the driving cycle. Battery and motor temperatures also diverge significantly from their respective setpoints.

The rule-based controller (Figure 11b) substantially improves temperature tracking across all three components. This result reflects the rule-based controller's effective feedback action in rejecting disturbances and maintaining temperatures near their setpoints. However, it should be noted that rule-based controllers typically require manual parameter tuning for specific operating conditions and may not adapt well to unseen scenarios without retuning.

The RL-based strategy (Figure 11c) achieves lower MAE values than the rule-based controller across all three subsystems. Specifically, DDPG achieves 28.6% lower cabin MAE, 28.9% lower battery MAE, and 5.8% lower motor MAE compared to the rule-based controller. This demonstrates effective multi-actuator coordination, reallocating thermal capacity among subsystems in response to thermal demands. The temperature response curves under DDPG control exhibit suppressed oscillations and stable convergence, demonstrating effective handling of multi-source coupling and external disturbances inherent in the driving cycle.

This performance advantage can be primarily attributed to the fundamental differences in control architecture. Specifically, the rule-based controller employed in this work adopts a decoupled architecture, wherein each temperature zone is regulated independently without explicit compensation for inter-zone thermal coupling. Consequently, this configuration fails to account for the complex thermal interactions, leading to suboptimal performance. In contrast, the RL policy implicitly captures the underlying coupling dynamics through reward-driven



(a) Fixed-parameter **(b)** Rule-based control **(c)** RL-based control

Figure 11. Temperature responses, valve positions, and tracking error under different control strategies: (a) fixed-parameter with constant valve positions; (b) rule-based control; (c) RL-based control.

training, thereby enabling inherently coordinated multi-variable control.

Overall, the experimental results demonstrate that the RL-based strategy achieves adaptive multi-actuator coordination, effectively addressing the rigidity of the fixed-parameter baseline and the tuning-dependent nature of the rule-based controller. By implicitly learning inter-zone thermal coupling and dynamically reallocating thermal capacity in response to varying driving conditions, the RL policy reduces temperature tracking errors, while ensuring improved stability. These findings suggest that data-driven control presents a more effective and self-adapting alternative for managing complex ITMS. Leveraging this adaptability, the learned policy naturally enables continuous refinement. By integrating online continual learning or meta-adaptation, it progressively updates its implicit coupling representation to accommodate component aging, long-tail scenarios, or unobserved driving regimes without full retraining. These extensions ultimately reinforce the shift from calibration-heavy paradigms to fully autonomous thermal management.

5 Conclusion

This paper addresses the thermal management challenge of electric vehicles under complex operating conditions by proposing a development framework that integrates thermal modeling with data-driven methods. A full-vehicle 1D ITMS model is constructed in Modelica, providing a unified description of multiple heat sources and their coupling relationships. Combined with typical driving cycles, simulations verify the physical consistency and dynamic consistency of the model responses. Leveraging the FMI standard, the model is seamlessly integrated into a co-simulation environment, enabling efficient bidirectional interaction and stable data exchange for strategy execution. Within this pipeline, rule-based and RL-based controllers are implemented and evaluated. Comparative results demonstrate that the learned RL policy achieves lower MAE than both the fixed-parameter baseline and the rule-based controller across all three temperature zones, facilitating balanced multi-variable coordination.These results suggest the natural compatibility of data-driven approaches with the proposed framework. Collectively, this architecture achieves tight integration of modeling, simulation, and control, demonstrating strong scalability and adaptability. This work establishes a reliable and extensible technical pathway, providing a scalable reference for subsequent studies in electric vehicle thermal management.

Limitations. This study primarily validates the feasibility of the proposed integrated framework for electric vehicle thermal management, rather than pursuing high-fidelity system modeling or globally optimal control algorithms. Consequently, the current evaluation is confined to a simulation environment utilizing a parameterized thermal model under predefined driving cycles. Within this scope, the implemented RL controller serves as a representative demonstrator to verify the pipeline's applicability, rather than a benchmark of algorithmic superiority.

Future work. Follow-up research can build upon this framework in several directions. The ITMS model can be further calibrated using real vehicle data collected from hardware-in-the-loop (HIL) validation or real-world vehicle road testing to obtain more realistic thermal parameters and boundary conditions, thereby reducing the model-reality gap. However, since any 1D model is inherently a simplified abstraction of the real world, a residual gap inevitably remains. To bridge this gap, sim-to-real transfer techniques can be explored, enabling the RL policy to continuously self-optimize under real-world operating conditions. In particular, domain randomization (Tobin et al. 2017) and online adaptive fine-tuning (Clavera et al. 2019) enable direct policy deployment on physical vehicles. Furthermore, multi-agent RL approaches, such as MADDPG (Lowe et al. 2017), could be explored to enable decentralized coordinated control across subsystems, potentially combined with multi-agent communication mechanisms (Ding et al. 2020; Zhu et al. 2024) and evolutionary algorithms to further enhance global state estimation and policy optimization performance.

DOI
10.3384/ecp223417

Acknowledgements

This work has been supported by BYD Auto Industry Company Limited.

References

Batteh, John, Jesse Gohl, and Sureshkumar Chandrasekar (2014). "Integrated vehicle thermal management in modelica: Overview and applications". In: *Proceedings of the 10th International Modelica Conference*, pp. 409–418.

Bouvy, Claude, Sidney Baltzer, Peter Jeck, Jörg Gißing, Thomas Lichius, and Lutz Eckstein (2012). "Holistic vehicle simulation using modelica–An application on thermal management and operation strategy for electrified vehicles". In: *Proceedings of the 9th International MODELICA Conference, Munich, Germany*, pp. 3–5.

Chua, Kurtland, Roberto Calandra, Rowan McAllister, and Sergey Levine (2018). "Deep reinforcement learning in a handful of trials using probabilistic dynamics models". In: *Advances in neural information processing systems* 31.

Clavera, Ignasi, Anusha Nagabandi, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn (2019). "Learning to Adapt in Dynamic, Real-World Environments through Meta-Reinforcement Learning". In: *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=HyztsoC5Y7.

Ding, Ziluo, Tiejun Huang, and Zongqing Lu (2020). "Learning individually inferred communication for multi-agent cooperation". In: *Advances in neural information processing systems* 33, pp. 22069–22079.

Guo, Xin, Jiankun Peng, Hongwen He, Changcheng Wu, Hailong Zhang, and Chunye Ma (2025). "Integrated thermal-energy management for electric vehicles in high-temperature conditions using hierarchical reinforcement learning". In: *Expert Systems with Applications* 276, p. 127221.

Guo, Xin, Jiankun Peng, Jingda Wu, Changcheng Wu, and Chunye Ma (2025). "Energy-Efficient integrated thermal management for electric vehicles using evolutionary deep reinforcement learning". In: *Expert Systems with Applications*, p. 130331.

Janner, Michael, Justin Fu, Marvin Zhang, and Sergey Levine (2019). "When to trust your model: Model-based policy optimization". In: *Advances in neural information processing systems* 32.

Lee, Junbeom, Kwonhee Suh, Juhyeong Park, Donkyu Joo, Junghun Yun, and Daeoh Kang (2025). "A Study on Model-Based Thermal Management Systems Architecture Modeling and Energy Efficiency Prediction of Fuel Cell Electric Vehicles". In: *Modelica Conferences*, pp. 103–108.

Lillicrap, Timothy P et al. (2015). "Continuous control with deep reinforcement learning". In: *arXiv preprint arXiv:1509.02971*.

Lowe, Ryan, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch (2017). "Multi-agent actor-critic for mixed cooperative-competitive environments". In: *Advances in neural information processing systems* 30.

Pang, Yuan, Kuining Li, Shuai Dai, Yi Xie, Dongpeng Zhao, and Chi Huang (2025). "Deep reinforcement learning for thermal comfort and energy-efficient vehicle thermal management". In: *Case Studies in Thermal Engineering*, p. 106808.

Tan, Huachun, Hailong Zhang, Jiankun Peng, Zhuxi Jiang, and Yuankai Wu (2019). "Energy management of hybrid electric bus based on deep reinforcement learning in continuous state and action space". In: *Energy Conversion and Management* 195, pp. 548–560.

Tobin, Josh, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel (2017). "Domain randomization for transferring deep neural networks from simulation to the real world". In: *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, pp. 23–30.

Wu, Changcheng, Jiankun Peng, Jiaming Zhou, Xin Guo, Hongqiang Guo, and Chunye Ma (2025). "Thermal management methodology based on a hybrid deep deterministic policy gradient with memory function for battery electric vehicles in hot weather conditions". In: *IEEE Transactions on Transportation Electrification* 11.3, pp. 7232–7242.

Yadav, Perla, Lakith Jinadasa, Alex Wray, Simon Petrovich, Marios Georgiou, and Kambiz Ebrahimi (2026). "Battery Electric Vehicle Thermal Management System Modelling and Validation". In: *Thermo* 6.1, p. 4.

Zhu, Changxi, Mehdi Dastani, and Shihan Wang (2024). "A survey of multi-agent deep reinforcement learning with communication". In: *Autonomous Agents and Multi-Agent Systems* 38.1, p. 4.