Heat Exchanger Surrogates for a Vapor Compression System

Nasrulloh Ratu Bagus Satrio Loka[1] Nicolás Ablanque Mejía[2] Santiago Torras Ortiz[2] Sriram Karthik Gurumurthy[3] Antonello Monti[3] Joaquim Rigola[2] Carles Oliet[2] Ivo Couckuyt[1] Tom Dhaene[1]

[1]IDLab, Ghent University - IMEC, Belgium, {nasrulloh.loka, ivo.couckuyt, tom.dhaene}@ugent.be
[2]Universitat Politècnica de Catalunya - Barcelona Tech (UPC), Heat and Mass Transfer Technological Center (CTTC), Spain, {nicolas.ablanque, santiago.torras, joaquim.rigola, carles.oliet}@upc.edu
[3]ACS, EONERC, RWTH Aachen University, Germany, {sgurumurthy,amonti}@eonerc.rwth-aachen.de

Abstract

Given the computationally intensive nature of heat exchanger simulators, utilizing a data-driven surrogate model for efficiently computing the heat exchanger outputs is desirable. This study focuses on developing integrated surrogate models of heat exchangers for a vapor compression system in Modelica. The surrogate models are designed to serve as steady-state equivalents based on an efficient physics-based model calibrated using reference data from a more advanced simulation model. Subsequently, the calibrated model was employed to generate the training and testing data for developing Gaussian Process (GP) and Multi-Layer Perceptron (MLP) surrogates. The findings indicate that GPs exhibit high accuracy when applied to the heat exchanger's outputs with smooth behavior. GPs also demonstrate excellent data efficiency compared to MLPs. In cases where the GP struggles to model specific outputs effectively, MLPs are able to capture the more complex behavior. Moreover, hyperparameter optimization is employed to identify optimal MLP topologies. Finally, the fast and compact surrogate model was integrated into the Modelica/Dymola environment. This adaptation allowed the surrogate models to be directly combined with the physical model of the heat exchanger.
Keywords: Heat Exchanger, Surrogate Model, Gaussian Process, Multi-Layer Perceptron, Hyperparameter Optimization

1 Introduction

In thermal systems, amid the most common components, heat exchangers are arguably the most challenging units to simulate from the numerical point of view due to the complex thermal and fluid-dynamic phenomena involved (e.g. fluid phase changes). Different approaches exist to model heat exchangers, but in general, the level of accuracy should increase with the level of detail considered in the model.

This work explores the development of heat exchanger surrogate models to be used within the Modelica environment. In this sense, a flexible heat exchanger model (Ablanque et al. 2022), adapted to simulate an air-to-refrigerant condenser, has been used to train and evaluate surrogate models. The specific condenser studied is part of a vapor compression system which, in turn, is included in an aircraft Environmental Control System (ECS).

Various neural networks and deep learning techniques have been implemented to model different aspects of heat exchangers. (Abbassi and Bahar 2005) use a shallow neural network to model the thermodynamics of an evaporative condenser. (Romero-Méndez et al. 2016) model convective heat transfer rate that occurs during the evaporation of a refrigerant flow using a neural network.

In this work, surrogate models have been developed for each target variable of the heat exchanger. Gaussian Process (GP) regression models have been constructed for target variables exhibiting smooth behavior due to GP's data efficiency and strong interpolation abilities for smooth functions. A Multi-Layer Perceptron (MLP) Neural Network Model has been employed for more challenging target variables with highly non-linear behavior. To enhance the performance of the MLP, hyperparameter optimization on the network architecture has been conducted. The performance of the proposed surrogate models has been demonstrated, and the advantages of hyperparameter tuning have been highlighted in the context of surrogate modeling.

Finally, the fast and compact surrogate model was successfully integrated into the Modelica/Dymola environment. This adaptation enabled the direct integration of the surrogate models with the physical model of the heat exchanger.

2 Heat Exchanger Model

2.1 Description

The structure implemented for the heat exchanger model is aimed to provide high flexibility in terms of geometries, participating fluids (i.e., liquids, gases, and two-phase refrigerants), and phenomenologies such as evaporation and condensation. The model layout consists of two specific sub-components for calculating fluid flows which are thermally linked via an additional sub-component that stands for the solid parts. Figure 1 shows the scheme for an air-to-refrigerant condenser.

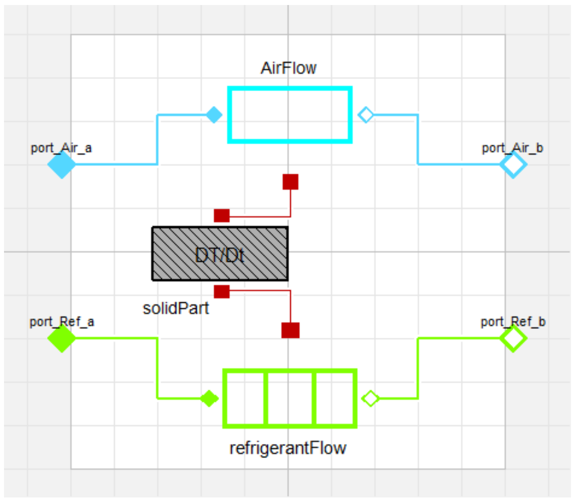

Figure 1. Heat exchanger main structure (air-to-refrigerant condenser).

The calculation of the fluid flow sub-component is based on a steady-state approach. The model discretization distinguishes three different zones, namely, superheated gas, two-phase, and sub-cooled liquid, as shown in Figure 2 for a condensation case. The aforementioned model zones can exist or not depending on the fluid inlet and outlet conditions so that the heat exchanger switches between different operating modes.

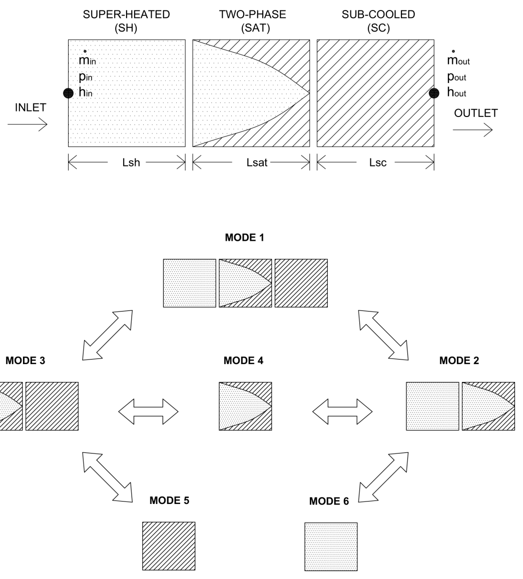

Figure 2. Condenser discretization and operating modes.

The pressure drop for the whole heat exchanger is calculated from a traditional approach (i.e., $\dot{m}$=KΔP$^{\alpha}$) where ΔP represents the pressure drop. The parameters K and α are previously determined from reference data.

The energy conservation equation is solved considering constant pressure and constant solid temperature. The calculation is conducted sequentially from zone to zone (if the heat calculated for the current zone is higher than the maximum heat allowed for this zone, the calculation will continue to the next zone. Otherwise, the calculation will terminate in the current zone). For single-phase zones, the heat flow rate between the fluid and the solid part $\dot{Q}_{single}$ is calculated based on an ε-NTU method (Incropera and DeWitt 1996) in order to optimize the calculation speed and to avoid unrealistic temperature values:

$$\dot{Q}_{single} = \varepsilon C(T_{solid} - T_{fluid,in}) \tag{1}$$

where C stands for the thermal capacity ratio and ε corresponds to the heat exchange effectiveness. For two-phase zones, the heat flow rate ($\dot{Q}_{two}$) is calculated from a standard approach based on a heat transfer coefficient (α), the temperature difference between the solid and the saturated fluid, and the heat transfer area (A):

$$\dot{Q}_{two} = \alpha(T_{solid} - T_{fluid,sat})A \tag{2}$$

The calculation of the solid sub-component is based on a transient approach. It is calculated considering a unique solid temperature (T), the solid mass (M), the solid mean specific heat capacity (c_p), and the heat rate transferred with the two fluids:

$$Mc_p\frac{dT}{dt} - \dot{Q}_{fluid,1} - \dot{Q}_{fluid,2} = 0 \tag{3}$$

The complete resolution is carried out by means of the default differential/algebraic system solver of Dymola. The heat exchanger's overall thermal response is dynamic as it combines the steady-state approach used for both flows with the dynamic approach considered for the solid part. Artificial relaxations can also be applied to the energy conservation equation of both flows to further overcome the negative impact of the absence of dynamic terms. The pressure drop equation is not only used to calculate the mass flow rate but also to approximate the phase saturation limits needed for the energy conservation equation.

2.2 Numerical Assessment and Tests

The current model has been developed to simulate different types of heat exchangers included in large thermal architectures consisting of multiple systems. Therefore, the need to meet demanding numerical requirements was crucial for its successful use in the aforementioned environments. The main requirements include robustness at initialization, robustness to any boundary conditions and/or signal types, robustness to conduct simulations at any particular simulation set-up parameter (e.g., interval length), the capacity of both fluids to handle null mass flow rate as well as reversed flows, and ability to handle changes of the expected heat flow direction. The model is a suitable platform to generate large quantities of training data and tests for the surrogate models due to its numerical characteristics.

2.2.1 Initialization and steady-state tests

A complete data set of cases has been generated to test the robustness of the model during initialization and the correct resolution for steady-state conditions. The data set has been built-up taking into account different values for all the boundary conditions (i.e., air and refrigerant inlet parameters) covering the whole physical range of possibilities and all its possible combinations. The data set also takes into consideration different fluid boundary condition types (see Figure 3) and different values for the interval length.

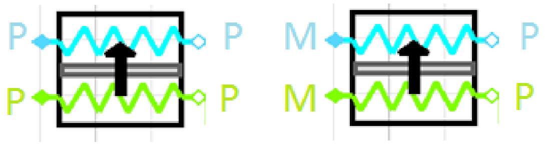

Figure 3. Boundary condition types: pressure - pressure (left) and Mass flow rate - pressure (right).

The data set consists of 1296 cases (216 cases for each combination of boundary condition type and interval length). The simulation stop time is set at 2000 seconds so that the steady-state condition can be reached. The results have shown that the model initialization is successful for all the cases without being affected by the combination of boundary values or the interval length value.

2.2.2 Transient tests

Similarly to the initialization studies, many tests have been conducted to assess the model's numerical robustness for other crucial transient conditions. Some illustrative examples are presented. Figure 4 shows the results for a test where both the null mass flow rate and the reversed flow capacities are tested. This particular case corresponds to a transient simulation where the refrigerant is operating at a particular mode and is forced to experience flow direction changes and null mass flow rate at different moments.

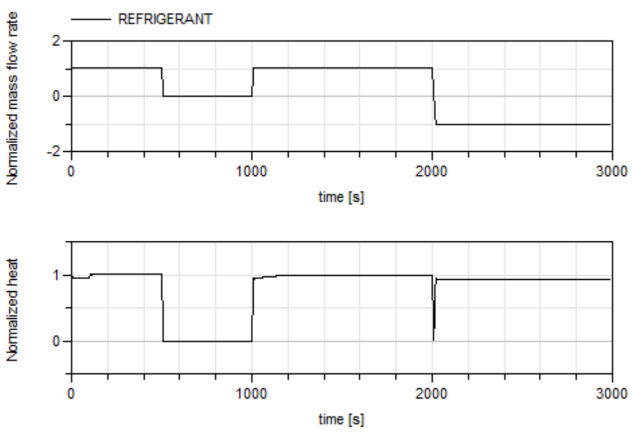

Figure 4. Null mass flow rate and reversed flow test example.

Figure 5 shows illustrative results for a test where many boundary conditions are provided as sine signals to force mode transitions in the heat exchanger (see Figure 1).

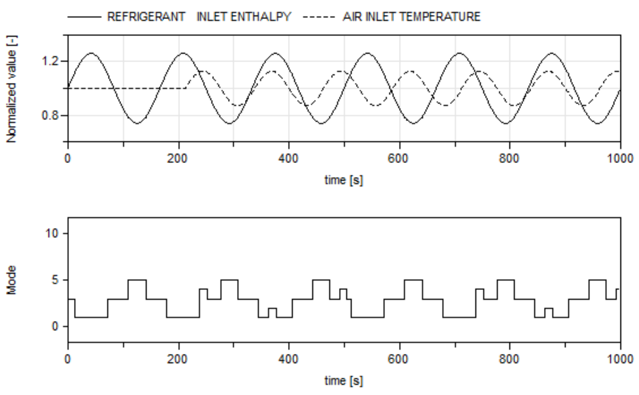

Figure 5. Sines signals test example.

2.2.3 Model validation

The accuracy of the condenser model has been comprehensively assessed. In this sense, a full set of reference data of 4766 cases has been provided from an advanced heat exchanger numerical model and used to compare the predictions. The parameter used for the comparisons is the so-called Prediction Error (PE) which characterizes the difference between the model-predicted value and the reference value of a particular variable. The local PE is a percentage value, and for the heat flow, it is evaluated as follows:

$$\mathrm{PE} = \frac{|\dot{Q}_{model} - \dot{Q}_{reference}|}{\dot{Q}_{reference}} \times 100 \tag{4}$$

To assess the accuracy regarding the whole data, an averaged PE is used, the so-called Mean Prediction Error (MPE), which is defined as follows:

$$\mathrm{MPE} = \frac{1}{N}\sum_{i=1}^{i=N} \mathrm{PE}_i \tag{5}$$

Table 1 shows the mean prediction error for the heat flow predicted by the condenser model. The results show good accuracy as the MPE for the whole data is 2.49 (this value decreases significantly as the less accurate results are not being considered).

3 Surrogate Modeling

Surrogate modeling aims to develop data-driven regression models that emulate complex systems or processes. This compact surrogate can then be used for real-time analysis, optimization, or prediction without the need for resource-intensive direct simulation of the system. In a regression problem, an input set is denoted as $X = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N]$, $\mathbf{x}_i \in \mathbb{R}^d$, where d is the dimensionality

Table 1. Condenser model accuracy assessment (heat flow prediction).

Data	%	MPE	MaxPE	std
4766	100	2.49	42.0	3.7
4671	98	2.12	15.6	2.6
4528	95	1.78	10.4	1.9
4289	90	1.43	5.8	1.2
4051	85	1.23	4.0	0.8
3575	75	0.99	2.2	0.6

of the input. A corresponding set of continuous observations is denoted as $Y = [f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_N)]$, with $f(\mathbf{x}_i) \in \mathbb{R}$. The goal is to construct a model that can predict the value of $y := f(x_*)$ for an unobserved point x_*. The prediction is denoted as $\hat{y}$. In this study, two techniques are employed to build predictive models of the heat exchanger system, namely Gaussian Processes and Multi-Layer Perceptrons.

3.1 Gaussian Process Regression

Gaussian Process (GP) (Rasmussen and Williams 2018) is a common data-efficient surrogate for regression problems. A GP is specified by a mean $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$. Given pair of input sets X and its evaluation on the system simulations, GP can be defined as: $f(\mathbf{x}) \sim GP(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$.

For the choice of the kernel function, the Matérn 5/2 kernel (Handcock and Stein 1993) is used as it does not put strong smoothness assumptions on the unknown function to be approximated (Genton 2002). The Matérn 5/2 kernel is defined as:

$$k\left(\mathbf{x}, \mathbf{x}'\right) = \gamma\left(1 + \sqrt{5}r + \frac{5}{3}r^2\right)\exp(-\sqrt{5}r), \tag{6}$$

$$r = \sqrt{\sum_{m=1}^{d} \frac{\left(x_m - x_m'\right)^2}{l_m^2}} \tag{7}$$

where γ is a scale parameter, and l is a lengthscale parameter for the kernel function.

Training the GP model involves estimating the hyperparameters $\hat{\theta}$. In this case, $\hat{\theta}$ contain the parameters of $k(\mathbf{x}, \mathbf{x}')$. Maximum Likelihood Estimation (MLE) is used to estimate the hyperparameters:

$$\hat{\theta} = \arg\max_{\theta} \log p(\mathbf{f} \mid X, \theta) \tag{8}$$

$$= \arg\max_{\theta} -\frac{1}{2}\left(\log|2\pi K_{\mathbf{xx}}| + \mathbf{f}^T K_{\mathbf{xx}}^{-1}\mathbf{f}\right) \tag{9}$$

The predictive mean $\mu\left(X_\star\right)$ and the predictive variance $\sigma^2\left(X_\star\right)$ of a new, untested data point X_* is calculated as:

$$\mu\left(X_\star\right) = K_{\star\mathrm{x}}K_{\mathrm{xx}}^{-1}Y \tag{10}$$

$$\sigma^2\left(X_\star\right) = K_{\star\star} - K_{\star\mathrm{x}}K_{\mathrm{xx}}^{-1}K_{\star\mathrm{x}}^T \tag{11}$$

where $K_{\mathrm{xx}} = k(\mathbf{x}_i, \mathbf{x}_j)$, $K_{\star\mathrm{x}} = k(\mathbf{x}_{\star i}, \mathbf{x}_j)$, and $K_{\star\star} = k(\mathbf{x}_{\star i}, \mathbf{x}_{\star j})$. The predictive mean of the GP is used as the surrogate model prediction $\hat{y}$. Moreover, the predictive variance could prove beneficial for quantifying uncertainty, which increases trust in the outcome and enables decision-making, optimization, or anomaly detection.

For Gaussian process regression, the GPFlow library (G. Matthews et al. 2017) is used, and the MLE is optimized using the Limited memory Broyden–Fletcher–Goldfarb–Shanno Bounded (LBFGS-B) optimizer (C. Zhu et al. 1997).

3.2 Multi Layer Perceptron

The *Multi Layer Perceptron* (MLP) (Hinton 1989) is a popular class of deep learning neural network architectures used for regression tasks. While it is not as data efficient as GP, it can capture more complex, non-linear relationships between input and output variables. It also scales well to the size of the data set compared to GP w.r.t. computational complexity. MLP is a versatile choice for surrogate modeling when a larger data set is available.

An MLP comprises an Input Layer, L Hidden Layers, and an Output Layer. The Input Layer matches the dimensionality of the input data, while the Output Layer matches the dimensionality of the target function. Generally, an MLP (Prince 2023) can be described as:

$$\begin{aligned}
\mathbf{h}_1 &= \mathbf{a}\left[b_0 + \mathbf{W}_0\mathbf{x}\right] \\
\mathbf{h}_2 &= \mathbf{a}\left[b_1 + \mathbf{W}_1\mathbf{h}_1\right] \\
\mathbf{h}_3 &= \mathbf{a}\left[b_2 + \mathbf{W}_2\mathbf{h}_2\right] \\
&\vdots \\
\mathbf{h}_L &= \mathbf{a}\left[b_{L-1} + \mathbf{W}_{L-1}\mathbf{h}_{L-1}\right] \\
\hat{\mathbf{y}} &= b_L + \mathbf{W}_L\mathbf{h}_L.
\end{aligned} \tag{12}$$

where b_l and $\mathbf{W}_l$ are the bias term and weight parameters of the network at l^{th} layer, and $\mathbf{a}$, is the activation function. Rectified Linear Unit (ReLU) (Fukushima 1969) is used for $\mathbf{a}$ and defined as $\mathbf{a}(x) = \max(0, x)$

To find the optimal parameters (b_l and $\mathbf{W}_\mathbf{l}$), the ADAM (Kingma and Ba 2015) optimizer is used. Specifically for this study, The base architecture of the MLP was set to 150, 100, and 50 neurons for each of the three hidden layers respectively, as illustrated in Figure 6. The Scikit-learn machine learning library (Pedregosa et al. 2011) is used to train the MLP surrogates.

4 Heat Exchanger Surrogate Models

Steady-state equivalent surrogate models for the heat exchanger have been developed and tested using data sets derived from the physical simulator. A heat exchanger model incorporating mass flow rate and pressure approach was used in this case. The input-output diagram of the surrogate models is illustrated in Figure 7.

Table 2. Heat exchanger surrogate model inputs domain.

Variable	Input	Lower Bound	Upper Bound	Unit
Pa_ref	Refrigerant pressure at refrigerant flow port A	150000	1800000	Pa
Pa_air	Air pressure at refrigerant flow air port A	20000	110000	Pa
m_ref	Refrigerant mass flow rate	-0.16	0.16	kg/s
m_air	Air mass flow rate	-1.8	1.8	kg/s
Tin_air	Air inlet temperature	-20	60	°C
Hin_ref	Ref. inlet enthalpy	210	490	kJ/kg

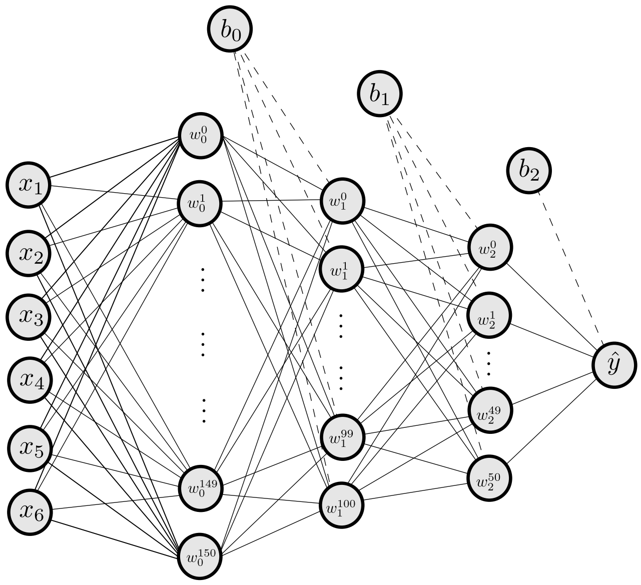

Figure 6. Architecture of the MLP used in this paper.

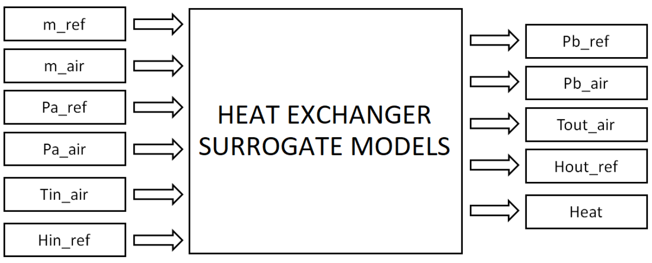

Figure 7. Heat exchanger surrogate model Inputs and Outputs.

Table 3. Specification of the heat exchanger surrogate.

Variable	Output	Unit
Pb_ref	Refrigerant pressure at port B	Pa
Pb_air	Air pressure at port B	Pa
Tout_air	Air outlet temperature	°C
Hout_ref	Ref. outlet specific enthalpy	kJ/kg
Heat	Heat transferred between fluids	W

4.1 Generating the data sets

Data collection for the heat exchanger surrogate models has been performed by evaluating the physical model developed in the Modelica framework. The data sets are described in Table 2 and 3 respectively. In total, four data sets were prepared. The first two data sets, consisting of 150 and 80,000 points, were drawn using the Halton random sequence (Owen 2017). The remaining two data sets consist of 40,000 and 100,000 random points for hyperparameter optimization and validation of the surrogate model, respectively (Gramacy 2020).

5 Surrogate Modeling Results

The performance of the surrogate models is evaluated in terms of their predictive accuracy. Furthermore, hyperparameter optimization was conducted for the Multi-Layer Perceptron (MLP) to improve the performance of the surrogate models.

5.1 Performance Comparison

Surrogate models have been developed and benchmarked for all of the outputs of the heat exchanger. In particular, four different surrogate model scenarios are executed to select the surrogate model with the best performance. The considered surrogate model scenarios are:

- MLP trained on 80,000 points (MLP-80K).
- MLP trained on 150 points (MLP-150).
- GP trained on 150 points (GP-150).
- Random Forest (RF) trained on 150 points (RF-150).

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{13}$$

The Root Mean Squared Error (RMSE) is used as a metric to evaluate the surrogate models on testing data sets of 100,000 points. The RMSE formula is presented in equation 13. The full result of the benchmark is shown in Table 4. The surrogate model with the lowest RMSE is used for each output as the final surrogate mode. Almost all MLP-80K models have the best RMSE compared to the other methods, except for the Pb_air output. In this case, GP is superior to MLP. Thus, all outputs are modeled using MLP with 80,000 data points except for Pb_air.

Table 4. Testing Root Mean Squared Error (RMSE) of all compared methods.

Output	RMSE			
	MLP-80K	MLP-150	GP-150	RF-150
Pb_ref	**8.54e+3**	4.55e+5	1.30e+5	5.06e+4
Pb_air	2.33e+2	7.36e+3	**4.24e+1**	4.14e+2
Tout_air	**1.51e+0**	1.26e+2	2.27e+1	3.49e+1
Hout_ref	**5.20e+3**	5.59e+5	8.02e+4	1.07e+5
Heat	**4.52e+2**	2.56e+4	8.98e+3	7.86e+3

5.2 Neural Network Hyperparameter Optimization

To explore the potential of MLP models in more detail, a hyperparameter optimization step was performed.

HyperParameter Optimization (HPO) (Yu and H. Zhu 2020; Morales-Hernández, Van Nieuwenhuyse, and Rojas Gonzalez 2022) for the MLP has been conducted using the Optuna framework (Akiba et al. 2019). The optimized hyperparameters are the learning rate and the architecture: the number of the hidden layer, the number of neurons of the hidden layers, and the activation function. Full specification of the MLP hyperparameter search space is presented in Table 5.

Table 5. Hyperparameter search space for the HPO.

Variable	Domain	Type
Number of hidden layer	[2, 6]	Integer
Number of neurons	[8, 1024]	Integer
Learning rate	[0.0001, 0.01]	Float
Activation function	{identity, tanh, logistic, ReLU}	Function

The HPO was conducted with a budget of 250 iterations. The training data consisted of 80,000 points, while the cost function was defined as the loss value on a randomly sampled validation data set of 40,000 points. The optimal hyperparameters identified through the HPO process are presented in Table 6.

Finally, the complete comparison of the Base-MLP and the optimal architecture found by the HPO (i.e., HPO-MLP) is presented in Table 7. It should be noted that the same hyperparameter setting from the previous section (MLP-80K) was used for the Base-MLP.

6 Modelica integration of the surrogate models

The resulting surrogate models have been integrated in the Modelica framework. To accomplish this, the optimal parameters obtained during training are stored as matrices on MATLAB ('*.mat*') files, as they are compatible with the Modelica framework. For the GP model, these files store the training data, kernel parameters, and precomputed terms that are independent of newly observed data (i.e., $K_{XX}^{-1}Y$). This speeds up the computation of predictions since the expensive matrix inverse operation does not need to be recalculated every time a new point needs to be predicted. Additionally, for the MLP model, the resulting $\mathbf{W}_l$ and b_l terms are saved, along with the activation functions at each layer for the HPO-MLP.

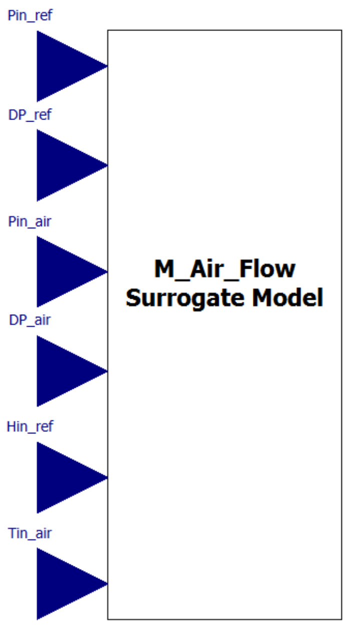

Figure 8. Surrogate model adaption in Open Modelica.

The integration process utilizes OpenModelica (Fritzson et al. 2005) and can also be adapted to Dymola. The command '*readRealMatrix*' imports the matrix, and '*readMatrixSize*' is used to retrieve the dimensions of the matrices. The surrogate model prediction routine was then implemented in Modelica using equations 10 and 12 for GP and MLP, respectively. The adapted Modelica surrogate prediction function block is shown in Fig. 8 where the six inputs required by the function are shown. The only input that this function block requires is the path where the '*.mat*' file can be found.

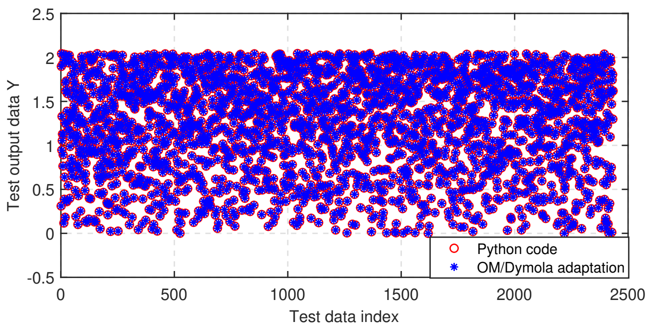

Figure 9. Testing the Open Modelica (OM)/Dymola adaptation.

2,500 test data points have been evaluated using the physical model for validating the Modelica implementation. We compared the resulting Modelica outputs with

Table 6. Multi-layer perceptron hyperparameter optimization result.

Variable	N hidden layer	Number of neurons	Learning rate	Activation function
Pb_ref	4	(1012, 252, 470, 392)	0.00119	ReLU
Tout_air	4	(410, 607, 906, 280)	0.00010	ReLU
Hout_ref	5	(752, 862, 122, 226, 221)	0.00098	ReLU
Heat	4	(895, 501, 667, 670)	0.00054	ReLU

Table 7. Root Mean Squared Error (RMSE) of Base-MLP and HPO-MLP. Ten repetitions have been conducted to validate the robustness of the optimized model.

Surrogate Name	RMSE ± Standard Deviation		Improvement (%)
	Base-MLP	HPO-MLP	
Pb_ref	9.380e+03 ± 5.608e+02	7.767e+03 ± 6.671e+02	17%
Tout_air	1.518e+00 ± 1.039e-01	1.266e+00 ± 1.016e-01	16%
Hout_ref	4.977e+03 ± 3.388e+02	4.374e+03 ± 5.778e+01	12%
Heat	4.685e+02 ± 4.408e+00	4.094E+02 ± 4.741E+01	13%

the original Python code for the surrogate model prediction. The results of this test are presented in Fig. 9, and it can be seen that the outputs are exactly matching. This generates trust to integrate the surrogate model in future applications where the heat exchanger will be used.

7 Conclusion

A heat exchanger model implemented in Modelica and adapted to simulate an air-to-refrigerant condenser has been validated and used to train and evaluate different surrogate models to mimic their steady-state behavior. The surrogate models are developed using Gaussian Process (GP) and Multi-Layer Perceptron (MLP) models. GPs are employed to capture the linear behavior of some heat exchanger outputs, while MLPs are utilized to handle other outputs with more complex, non-linear behavior. Additionally, hyperparameter optimization for the MLP architecture has been conducted, which led to significant improvements compared to the standard architecture. As a proof of concept, the surrogate models were also integrated in the Modelica/Dymola environment such that they can be directly augmented with physical models.

In this specific study, the surrogate model does not exhibit a substantial improvement in calculation time compared to the physical model. This limitation can be attributed to the fast nature of the physical model employed here. However, in cases with more complex physical models such as the distributed method, it will provide a large reduction in calculation time. Nonetheless, the surrogate model played a crucial role in ensuring calculation stability. Some of the factors that contribute to instability include operating modes transitions, empirical and corrector factor transitions, as well as thermophysical properties near the saturation dome, among others.

Future research will concentrate on constructing transient surrogate models to represent the transient behavior of the heat exchanger model explicitly. This can be accomplished by employing more suitable surrogate modeling techniques, such as non-stationary Gaussian Processes or Autoregressive Models like Long Short-Term Memory Neural Networks.

Acknowledgements

This project has received funding from the Clean Sky 2 Joint Undertaking (JU) under grant agreement No 886533. The JU receives support from the European Union's Horizon 2020 research and innovation program and the Clean Sky 2 JU members other than the Union.

Horizon 2020
European Union Funding
for Research & Innovation

Clean Sky 2
JOINT UNDERTAKING

This work has also been supported by the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" and the "Fonds Wetenschappelijk Onderzoek (FWO)" programs.

Carles Oliet, as a Serra Húnter Associate Professor, acknowledges the Catalan Government for the support through this Programme.

Disclaimer: The contents presented in this article reflect only the author's point of view: the authors and Clean Sky JU are not responsible for any use that may be made of the information it contains.

References

Abbassi, A. and L. Bahar (2005). "Application of neural network for the modeling and control of evaporative condenser cooling load". In: *Applied Thermal Engineering* 25.17, pp. 3176–3186. ISSN: 1359-4311. DOI: https://doi.org/10.1016/j.applthermaleng.2005.04.006.

Ablanque, Nicolás, Santiago Torras, Carles Oliet, and Joaquim Rigola (2022). “Thermal Systems Oriented Two-Phase Heat Exchanger Models. Focus on Numerical Robustness”. In: *19th International Refrigeration and Air Conditioning Conference at Purdue*, pp. 1–10. URL: https://docs.lib.purdue.edu/iracc/2405/.

Akiba, Takuya, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama (2019). “Optuna: A Next-Generation Hyperparameter Optimization Framework”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD ’19. Anchorage, AK, USA: Association for Computing Machinery, pp. 2623–2631. ISBN: 9781450362016. DOI: 10.1145/3292500.3330701.

Fritzson, Peter, Peter Aronsson, Håkan Lundvall, Kaj Nyström, Adrian Pop, Levon Saldamli, and David Broman (2005-01). “The OpenModelica Modeling, Simulation, and Development Environment”. In.

Fukushima, Kunihiko (1969). “Visual Feature Extraction by a Multilayered Network of Analog Threshold Elements”. In: *IEEE Transactions on Systems Science and Cybernetics* 5.4, pp. 322–333. DOI: 10.1109/TSSC.1969.300225.

G. Matthews, Alexander G. de, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman (2017). “GPflow: A Gaussian Process Library using TensorFlow”. In: *Journal of Machine Learning Research* 18.40, pp. 1–6. URL: http://jmlr.org/papers/v18/16-537.html.

Genton, Marc G. (2002-03). “Classes of Kernels for Machine Learning: A Statistics Perspective”. In: *J. Mach. Learn. Res.* 2, pp. 299–312. ISSN: 1532-4435. DOI: 10.5555/944790.944815.

Gramacy, Robert B. (2020). *Surrogates: Gaussian Process Modeling, Design and Optimization for the Applied Sciences*. Boca Raton, Florida: Chapman Hall/CRC. DOI: 10.1201/9780367815493.

Handcock, Mark S. and Michael L. Stein (1993). “A Bayesian Analysis of Kriging”. In: *Technometrics* 35.4, pp. 403–410. ISSN: 00401706. DOI: 10.2307/1270273.

Hinton, Geoffrey E. (1989). “Connectionist learning procedures”. In: *Artificial Intelligence* 40.1, pp. 185–234. ISSN: 0004-3702. DOI: https://doi.org/10.1016/0004-3702(89)90049-0.

Incropera, Frank P. and David P. DeWitt (1996). *Fundamentals of Heat and Mass Transfer*. Wiley. ISBN: 978-1-119-35388-1.

Kingma, Diederik P. and Jimmy Ba (2015). “Adam: A Method for Stochastic Optimization.” In: *ICLR (Poster)*. Ed. by Yoshua Bengio and Yann LeCun. URL: http://dblp.uni-trier.de/db/conf/iclr/iclr2015.html#KingmaB14.

Morales-Hernández, Alejandro, Inneke Van Nieuwenhuyse, and Sebastian Rojas Gonzalez (2022). “A survey on multi-objective hyperparameter optimization algorithms for machine learning”. In: *Artificial Intelligence Review*, pp. 1–51. DOI: 10.1007/s10462-022-10359-2.

Owen, Art B. (2017-06). “A randomized Halton algorithm in R”. In: URL: http://arxiv.org/abs/1706.02808.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay (2011). “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12, pp. 2825–2830. DOI: https://dl.acm.org/doi/10.5555/1953048.2078195.

Prince, Simon J.D. (2023). *Understanding Deep Learning*. MIT Press. URL: https://udlbook.github.io/udlbook/.

Rasmussen, Carl Edward and Christopher K. I. Williams (2018). *Gaussian Processes for Machine Learning*. Cambridge, Massachusetts: The MIT Press. DOI: 10.7551/mitpress/3206.001.0001.

Romero-Méndez, Ricardo, Patricia Lara-Vázquez, Francisco Oviedo-Tolentino, Héctor Martín Durán-García, Francisco Gerardo Pérez-Gutiérrez, and Arturo Pacheco-Vega (2016). “Use of Artificial Neural Networks for Prediction of the Convective Heat Transfer Coefficient in Evaporative Mini-Tubes”. In: *Ingeniería, Investigación y Tecnología* 17.1, pp. 23–34. ISSN: 1405-7743. DOI: https://doi.org/10.1016/j.riit.2016.01.003.

Yu, Tong and Hong Zhu (2020). “Hyper-Parameter Optimization: A Review of Algorithms and Applications”. In: *CoRR* abs/2003.05689. arXiv: 2003.05689. URL: https://arxiv.org/abs/2003.05689.

Zhu, Ciyou, Richard H. Byrd, Peihuang Lu, and Jorge Nocedal (1997-12). “Algorithm 778: L-BFGS-B: Fortran Subroutines for Large-Scale Bound-Constrained Optimization”. In: *ACM Trans. Math. Softw.* 23.4, pp. 550–560. ISSN: 0098-3500. DOI: 10.1145/279232.279236.