

Picard-KKT-hPINN: Enforcing Nonlinear Enthalpy Balances for Physically Consistent Neural Networks

Lastrucci, G.; Karia, T.; Gromotka, Z.; Schweidtmann, A.M.

DOI

[10.69997/sct.108423](https://doi.org/10.69997/sct.108423)

Publication date

2025

Document Version

Final published version

Citation (APA)

Lastrucci, G., Karia, T., Gromotka, Z., & Schweidtmann, A. M. (2025). *Picard-KKT-hPINN: Enforcing Nonlinear Enthalpy Balances for Physically Consistent Neural Networks*. Paper presented at European Symposium on Computer Aided Process Engineering, ESCAPE 35, Ghent, Belgium.
<https://doi.org/10.69997/sct.108423>

Important note

To cite this publication, please use the final published version (if applicable).
Please check the document version above.

Copyright

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

Takedown policy

Please contact us and provide details if you believe this document breaches copyrights.
We will remove access to the work immediately and investigate your claim.

Picard-KKT-hPINN: Enforcing Nonlinear Enthalpy Balances for Physically Consistent Neural Networks

Giacomo Lastrucci^a, Tanuj Karia^a, Zoë Gromotka^b, and Artur M. Schweidtmann^{a*}

^a Process Intelligence Research Team, Department of Chemical Engineering, Delft University of Technology, The Netherlands

^b Mathematical Physics Group, Delft Institute of Applied Mathematics, Delft University of Technology, The Netherlands

* Corresponding Author: a.schweidtmann@tudelft.nl.

ABSTRACT

Neural networks (NNs) are widely used as surrogate models but they do not guarantee physically consistent predictions thereby preventing adoption in various applications. We propose a method that can enforce NNs to satisfy physical laws that are nonlinear in nature such as enthalpy balances. Our approach, inspired by Picard's successive approximations method, aims to enforce multiplicatively separable constraints by sequentially *freezing* and projecting a set of the participating variables. We demonstrate our Picard-KKT-hPINN for surrogate modeling of a catalytic packed bed reactor for methanol synthesis. Our results show that the method efficiently enforces nonlinear enthalpy and linear atomic balances at machine-level precision. Additionally, we show that enforcing conservation laws can improve accuracy in data-scarce conditions compared to *vanilla* multilayer perceptron.

Keywords: Surrogate modeling, Hard-constrained neural networks, Physics-informed neural networks, Constrained learning

INTRODUCTION

Surrogate modeling plays a crucial role in simplifying and approximating complex physical models, making them suitable for large-scale simulations and optimization studies of industrial relevance. Machine learning models, such as neural networks (NNs), are particularly well-suited for this purpose due to their simplicity and strong regression capabilities [1]. However, despite exceptional advancements in machine learning, issues and skepticism regarding the black-box nature and physical inconsistency of these models hinder the adoption of machine learning-based tools (and, more broadly, artificial intelligence) in industrial applications [2,3]

To mitigate this limitation, significant research has been carried out to enforce known mechanistic relationships between inputs and predictions in NNs. Soft-constrained neural networks represent an approach in which physical equations are included as penalty terms in the loss function [4,5]. This method does not require any modification of the neural network architecture, but it does not guarantee the exact satisfaction of the constraints. When exact adherence to constraints such as closure of balances is required, hard-constrained neural

networks become relevant. Models belonging to this class aim to strictly satisfy given equations or symmetries underlying the original system (constrained learning).

Different methodologies have been proposed for constrained learning. For instance, Beucler et al. (2019) and Donti et al. (2019) apply a method known as prediction and completion, where an NN is trained to predict a subset of variables, while the remaining ones are determined by solving a (non)linear system during training [6,7]. Researchers in optimization proposed to learn solutions to constrained optimization problems leveraging Lagrangian duality [8]; however, aspects of this approach still rely on soft constraints. Chen et al. (2021) introduced a projection step to guide the learning process by enforcing adherence to discretized differential equations [9]. Based on a similar concept, Chen, et al. (2024) presented KKT-hPINN to adjust neural network predictions, ensuring compliance with linear algebraic constraints [10]. Recently, Mukherjee and Bhattacharyya (2024) proposed to train constrained neural network using an NLP solver such as IPOPT [11]. Likewise, Schweidtmann et al. (2019) have used deterministic global optimization to compute guaranteed worst-case accuracies of NNs [12]. However, the use of local or global NLP solvers can be memory and

time-intensive for NNs that have a large number of parameters and training data points. Overall, current approaches (1) rely on external solvers that increase the computational cost to train and evaluate the NN or (2) are limited to enforcing linear equality constraints.

To address these challenges, we introduce a computationally efficient method that enforces relevant physical laws in the form of nonlinear algebraic equations. We extend the KKT-hPINN framework by considering local projections and numerical techniques to ensure exact constraint satisfaction at machine-level precision. Drawing inspiration from Picard's successive approximation method for solving nonlinear PDEs, we name our model Picard-KKT-hPINN.

METHODS

We introduce Picard-KKT-hPINN, a method to enforce algebraic nonlinear constraints, such as energy balances, in NN predictions. It extends the KKT-hPINN approach with local projections and numerical techniques such as variable-freezing, while maintaining computational efficiency.

Problem statement

Given a dataset $(x_i, y_i)_{i=1, \dots, N}$, representing the input-output behavior of a physical system (e.g., a unit operation), our goal is to train an NN to approximate the underlying model governing this system. In some cases, algebraic equations describing some relationship between input and output are known beforehand. We aim to develop an NN f_θ such that the predicted output $\hat{y} = f_\theta(x)$ satisfies a set of algebraic constraints $c_k(x, y) = 0, k = 1, \dots, N_c$, which may represent conservation laws or design specifications. The number of constraints (N_c) must not exceed the dimensionality of the neural network's output (N_o). This ensures the system is underdetermined ($N_c < N_o$), allowing the network to learn from the data while adhering to these constraints.

Background: KKT-hPINN

Chen et al. (2024) proposed a model denoted as KKT-hPINN to enforce linear relationships in NNs [10]. In KKT-hPINN, the (physically inconsistent) prediction of a generic multilayer perceptron is corrected by an orthogonal projection onto the linear feasible region described by a system of linear equations and geometrically defining a hyperplane.

Specifically, the prediction \hat{y} is corrected so as to respect a linear system by solving the following optimization problem:

$$\hat{y} = \underset{y}{\operatorname{argmin}} \|y - \hat{y}\|^2 \quad \text{s.t.} \quad \mathbf{A}x + \mathbf{B}y = \mathbf{b} \quad (1)$$

The corrected, physically consistent, solution \tilde{y} is achieved by solving the Karush-Kuhn-Tucker (KKT)

conditions of this optimization problem (Eq. 1), thus the name KKT-hPINN:

$$\tilde{y} = \mathbf{A}^*x + \mathbf{B}^*\hat{y} + \mathbf{b}^* \quad (2)$$

Considering the identity matrix \mathbf{I} :

$$\begin{aligned} \mathbf{A}^* &= -\mathbf{B}(\mathbf{B}\mathbf{B}^T)^{-1}\mathbf{A} \\ \mathbf{B}^* &= \mathbf{I} - \mathbf{B}(\mathbf{B}\mathbf{B}^T)^{-1}\mathbf{B} \\ \mathbf{b}^* &= -\mathbf{B}^T(\mathbf{B}\mathbf{B}^T)^{-1}\mathbf{b} \end{aligned} \quad (3)$$

The projection matrices $(\mathbf{A}^*, \mathbf{B}^*, \mathbf{b}^*)$ are computed analytically owing to the quadratic objective function and linear constraints. This projection acts as a physics-informed activation layer on the network output, providing a computationally efficient approach to enforce linear constraints. However, its applicability is limited to linear systems, making it unsuitable for scenarios with nonlinearities. For instance, while effective for modeling mass balances in chemical engineering, it struggles with enthalpy balances, which often involve nonlinear relationships.

Local projection

We generalize the linear constraints formulation in Eq. 1 to any nonlinear function in the input vector x :

$$c(x, y) = \mathbf{B}y + g(x) - \mathbf{b} = \mathbf{0} \quad (4)$$

Since the input vector is known for every instance, we can consider it as a constant term and define $v(x) = \mathbf{b} - g(x)$. Moreover, the input vector does not participate as an optimization variable in the projection process and thus $g(x)$ can be any nonlinear. Hence, Eq. 4 can be written as:

$$c(x, y) = \mathbf{B}y - v(x) = \mathbf{0} \quad (5)$$

where $v(x)$ is a vector-valued function $v: x \subseteq \mathbb{R}^{N_I} \rightarrow \mathbb{R}^{N_c}$. In particular, v is a nonlinear function of the N_I -dimensional input to the NN, meaning that its value is instance-dependent (an *instance* is a given datapoint (x_i, y_i)). Thus, the value of the constraint must be updated depending on each datapoint instance and can not be defined offline. Additionally, we consider local (*instance-specific*) constraint definition by allowing the matrix \mathbf{B} to depend on the output instance y_i . To simplify the notation, we define $\mathbf{B}_i = \mathbf{B}(y_i)$ and $v_i = v(x_i)$ such that:

$$c(x_i, y_i) = \mathbf{B}_i y_i - v_i = \mathbf{0} \quad (6)$$

This formulation comprises a wider class of functions in respect to the linear case in Eq. 1 and is foundational to handle local nonlinearities.

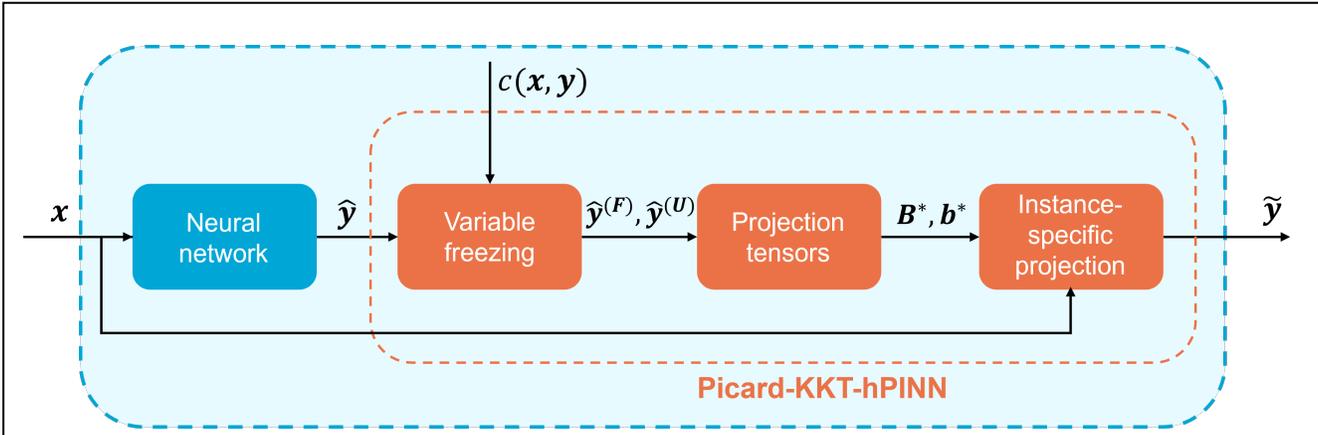


Figure 1: Picard-KKT-hPINN consists of a set of non-trainable layers that can be appended on every NN backbone (e.g., multilayer perceptrons, convolutional neural networks, transformers). Multiplicatively separable constraints (Eq. 8) are enforced exactly by partitioning the prediction vector (*variable freezing*). Then, the prediction of the NN is projected onto the feasible space defined by the constraints c . The layers composing Picard-KKT-hPINN are differentiable, hence, the NN is trained end-to-end and the architecture is equivalent at training and inference time.

The computational cost to achieve local projection can be reduced by exploiting parallel computing. We perform batch training (with BS being the number of elements in the batch) by building a *rank-3* tensor $\mathbf{B} \in \mathbb{R}^{BS \times N_c \times N_o}$ holding a number BS of local \mathbf{B}_i matrices. Similarly, we construct a *rank-2* tensor $\mathbf{V} \in \mathbb{R}^{BS \times N_c}$ to store the local \mathbf{v}_i vectors. The local projection matrices are computed in parallel (Eq. 7), hence reducing the apparent complexity of the most expensive computation, the matrix inversion, from $O(BS \times N^3)$ to $O(N^3)$.

$$\begin{aligned} \mathbf{B}^* &= \mathbf{I} - \mathbf{B}^T (\mathbf{B}\mathbf{B}^T)^{-1} \mathbf{B} \\ \mathbf{V}^* &= \mathbf{B}^T (\mathbf{B}\mathbf{B}^T)^{-1} \mathbf{V} \end{aligned} \quad (7)$$

More specifically, considering the batch inversion of the tensor $\mathbf{B}\mathbf{B}^T$, the dimension N is the number of constraints. Assuming to deal with tasks involving $N_c = N < 10^3$, the inversion is performed in a negligible amount of FLOPs in modern hardware. To the best of our knowledge, this dimensionality bound does not limit the applicability of the method for the majority of the surrogate modeling tasks in chemical engineering.

Picard-KKT-hPINN

Inspired by Picard's successive approximation method for nonlinear partial differential equations (PDEs), we develop a model to exactly enforce nonlinear constraints, including physical balances expressed as algebraic equations (See Figure 1).

Output partitioning and definition of the constraint class

Given the NN's output vector $\mathbf{y} = [y_1, \dots, y_{N_o}] \in \mathbb{R}^{N_o}$,

we partition its components as $\mathbf{y} = [\mathbf{y}^{(F)} \ \mathbf{y}^{(U)}]$. $\mathbf{y}^{(F)}$ is the subvector of *frozen* variables and $\mathbf{y}^{(U)}$ is the subvector of *unfrozen* variables, with $\mathbf{y}^{(F)} = [y_{j_1}, \dots, y_{j_{N_F}}] \in \mathbb{R}^{N_F}$ and $\mathbf{y}^{(U)} = [y_{j_1}, \dots, y_{j_{N_U}}] \in \mathbb{R}^{N_o - N_F}$, with $N_U = N_o - N_F$.

We aim to enforce multiplicatively separable equality constraints of the form:

$$c_k(\mathbf{x}, \mathbf{y}) = \mathbf{B}_k \mathbf{y} + F_k(\mathbf{y}^{(F)}) \cdot \mathbf{y}^{(U)} - v_k(\mathbf{x}) = 0 \quad k = 1, \dots, N_c \quad (8)$$

The constraint includes some linear function in the output \mathbf{y} , any nonlinear function v_k in the input \mathbf{x} , and a scalar multiplication between F_k , a nonlinear vector-valued function in the *frozen* variables $\mathbf{y}^{(F)}$, and the *unfrozen* variables $\mathbf{y}^{(U)}$. Note that $F_k: \mathbb{R}^{N_F} \rightarrow \mathbb{R}^{N_U}$.

Steady-state macroscopic conservation principles can be generally formulated as Eq. 8. Thus, the structure of the considered constraint is of interest for engineering and physics-based applications.

More in general, the constraints in Eq. 8 can be formulated in matrix notation as:

$$c(\mathbf{x}, \mathbf{y}) = \mathbf{B}\mathbf{y} + \mathbf{F}(\mathbf{y}^{(F)}) \cdot \mathbf{y}^{(U)} - \mathbf{v}(\mathbf{x}) = 0 \quad (9)$$

Here, $c(\mathbf{x}, \mathbf{y})$ is the vector of N_c constraints and $\mathbf{F}(\mathbf{y}^{(F)})$ is an $N_c \times N_U$ matrix. We develop a method to let the NN predictions exactly guarantee constraints belonging to the class in Eq. 9 by *freezing* a partition of the output variables, resembling Picard's iteration method.

Variables freezing

Given the output $\hat{\mathbf{y}}$ of an NN $f_{\theta}(\mathbf{x})$, we aim to guarantee the prediction to satisfy a constraint of the form as in Eq. 9 by projecting the latter onto the feasible space.

The projected prediction $\tilde{\mathbf{y}}$ can be computed as the solution of the following nonlinear program (NLP):

$$\begin{aligned} \tilde{\mathbf{y}} &= \underset{\mathbf{y}}{\operatorname{argmin}} \|\mathbf{y} - \hat{\mathbf{y}}\|^2 \\ \text{s.t. } c(\mathbf{x}, \mathbf{y}) &= \mathbf{B}\mathbf{y} + F(\mathbf{y}^{(F)}) \cdot \mathbf{y}^{(U)} - v(\mathbf{x}) = 0 \end{aligned} \quad (10)$$

Instead of relying on an external solver, we reformulate the NLP in Eq. 10 as a quadratic program (QP) solvable analytically. To achieve that, we assume the variables in $\mathbf{y}^{(F)}$ as *correct* and thus set them constant. This variable-freezing allows to reformulate Eq. 9 as a linear constraint by defining $\mathbf{B}' = \mathbf{B}\mathbf{y} + F(\mathbf{y}^{(F)})$:

$$c(\mathbf{x}, \mathbf{y}) = \mathbf{B}'\mathbf{y} - v(\mathbf{x}) = 0 \quad (11)$$

Since Eq. 11 is the matrix form equivalent of Eq. 6, the local projection is achieved according to Section 2.3 and results in $\tilde{\mathbf{y}} = \mathbf{B}^*\hat{\mathbf{y}} + \mathbf{v}^*$, where the projection tensors can be computed exploiting parallel computing during training.

The nonlinear term is handled by projecting only a partition of the output variables. We observe the method to be more efficient when $N_U < N_F$, that is, more variables are projected to guarantee the nonlinear constraints. However, in general, the whole output vector is still potentially projected since linear terms can be present. Such numerical technique resembles Picard's iteration method to solve nonlinear PDEs iteratively (hence, the name Picard-KKT-hPINN), where some variables are kept constant to the previous iteration's value to achieve a linear expression. Similarly, here we exploit the iterative training process to converge towards the ground truth data while guaranteeing the prediction obeys the nonlinear constraint.

RESULTS AND DISCUSSION

Case study and constraints

Our proposed method is illustrated on a tubular packed-bed reactor for methanol synthesis, using the same modeling assumptions as in Lastrucci et al. (2024) [13]. The surrogate models involve 10 inputs (inlet temperature, pressure, component flowrates, and coolant flowrate) and 10 outputs (outlet temperature, pressure, component flowrates, and hotspot temperature). The training and test datasets, consisting of 20,000 and 500 data points respectively, are generated by solving the mechanistic model of the reactor implemented in Python under diverse conditions bounded by operational limits derived from flowsheet simulation. Specifically, we simulate the simplified plant for methanol synthesis implemented in the IDEAS library [14] and evaluate extreme recycle conditions to determine those bounds. Hence, each datapoint (x_i, y_i) represents the input-output state variables corresponding to a specific operating scenario. The

generated data are intentionally noise-free, as the goal is to develop a surrogate model that faithfully replicates the original mechanistic model. The relevance of this study lies in enabling faster computations, for instance, by eliminating the need for dynamic optimization in flowsheet optimization problems.

We derive algebraic constraints underlying the steady state physical system that we aim to enforce in the surrogate model, such as atomic balance and total enthalpy balance. The system consists of four linear atomic balances (Eq. 12) and one nonlinear equation for the energy balance (Eq. 13):

$$\sum_{i=1}^{N_S} \dot{a}_{ij,in} - \sum_{i=1}^{N_S} \dot{a}_{ij,out} = 0, \quad j = C, H, O, N \quad (12)$$

$$\sum_{i=1}^{N_S} \dot{n}_{i,in} \tilde{h}_{i,in} - \sum_{i=1}^{N_S} \dot{n}_{i,out} \tilde{h}_{i,out} - \dot{Q} = 0 \quad (13)$$

In Eq. 12, $\dot{a}_{ij,in}$ and $\dot{a}_{ij,out}$ are the molar flowrates of atom j carried by species i , entering and exiting the reactor, respectively, and N_S is the number of chemical species involved (composed by four atoms, such as carbon, hydrogen, oxygen, and nitrogen). Being the atomic balances linear constraints, they can effectively be enforced using KKT-hPINN [10]. In Eq. 13, $\dot{n}_{i,in}$ and $\dot{n}_{i,out}$ are the molar flowrates of species i , $h_{i,in}$ and $h_{i,out}$ are the molar-specific enthalpies and \dot{Q} is the rate of heat transfer out of the system. In general, the specific enthalpy $\tilde{h}_{i,eval} = \tilde{h}_{i,eval}(T_{eval})$ is a nonlinear function of the temperature, depending on the modeling choice for the specific heat at constant pressure c_p . Since in the considered case study $\dot{Q} = \dot{n}_c \Delta \tilde{H}_{ev}(T_c)$ is a function of the input (coolant flowrate, \dot{n}_c) [13], the term $\sum_{i=1}^{N_S} \dot{n}_{i,in} \tilde{h}_{i,in} - \dot{Q}$ can be regarded as $v_k(\mathbf{x})$ in Eq. 8. More importantly, the term $-\sum_{i=1}^{N_S} \dot{n}_{i,out} \tilde{h}_{i,out}$ can be seen as the dot-product in Eq. 8, with $F_k(\mathbf{y}^{(F)}) = \tilde{h}_{i,out} = \tilde{h}_i(T_{out})$ and $\mathbf{y}^{(U)} = \dot{n}_{i,out}$. Thus, our Picard-KKT-hPINN model can exactly enforce both the atomic (Eq. 12) and the enthalpy balances (Eq. 13).

Experiments and comparison

We train a shallow Picard-KKT-hPINN consisting of 1 hidden layer with 64 neurons and ReLU activation function, and we compare against a KKT-hPINN and a multi-layer perceptron (MLP) with the same backbone architecture. Every model is trained for 50,000 epochs, using the Adam optimizer with a learning rate of 10^{-5} and a batch size of 2,000 samples. For reproducibility, the fully connected layers are deterministically optimized from a fixed Glorot initialization [15]. The training time of Picard KKT-hPINN on GPU hardware is comparable to that of MLP and KKT-hPINN, averaging only 15% longer, primarily due to the matrix inversion operation. At inference, the computational time of the three models is equivalent, even on CPU hardware (11th Gen Intel(R) Core(TM) i7-1185G7 @ 3.00GHz, 4 Core(s)).

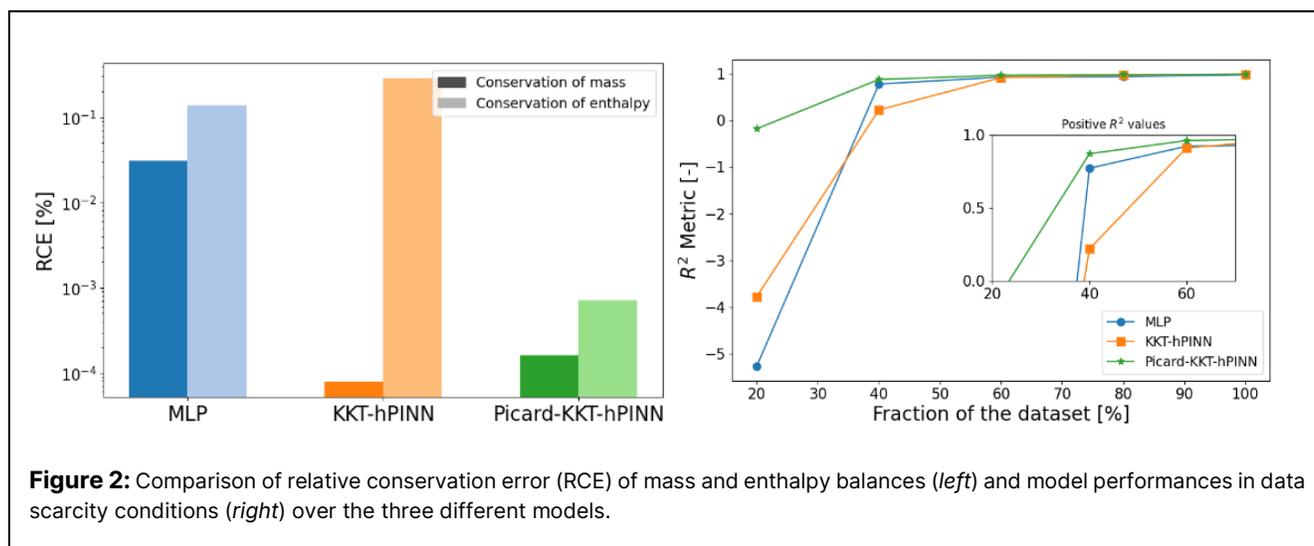


Table 1: Accuracy comparison over the three different models. We train all the models on a GPU hardware (NVIDIA GeForce RTX 3090).

	MLP	KKT-hPINN	Picard-KKT-hPINN
R^2 [-]	0.97	0.98	0.98
MAPE [%]	0.81	0.58	0.71

The accuracy of the models on the test set is comparable in terms of coefficient of determination (R^2) and mean absolute percentage error (MAPE). In Table 1, the metrics are averaged over all the predicted variables in the test set. We do not expect the constraints to increase the accuracy considerably when a traditional MLP is already very accurate.

The focus of the study is to demonstrate the effectiveness of Picard-KKT-hPINN in enforcing both linear and nonlinear constraints. Figure 2 (left) shows the relative conservation error of the total mass and enthalpy. The MLP, although very accurate, does not strictly satisfy the balances. As claimed by the authors [10], KKT-hPINN effectively enforces the linear mass balances at machine-level precision (note that the plot is in logarithmic scale), however, the predictions do not strictly satisfy the nonlinear enthalpy balance. Picard-KKT-hPINN, instead, enforces mass and enthalpy balances exactly, reducing the error below 0.001% for both.

In a real-world scenario, the amount of data is often a bottleneck when training *data-hungry* machine learning models such as NNs. We analyze the performance of the models in data scarcity conditions by training on uniformly sampled fractions of the original dataset. We demonstrate that constrained learning can improve model performance in data scarcity conditions (Figure 2, right). Specifically, enforcing atomic balance through KKT-hPINN outperforms the MLP when using between 20 and 30% of the original training dataset. When

enforcing both atomic and total enthalpy balance with Picard-KKT-hPINN, the model performance shows a positive coefficient of determination ($R^2 \approx 0.75$) when only 35% of the training dataset is used, while both MLP and KKT-hPINN show a negative R^2 in the same data regime.

CONCLUSION

We propose Picard-KKT-hPINN, a method to ensure that NN predictions adhere strictly to nonlinear algebraic constraints, such as enthalpy balances. Our work extends the recent KKT-hPINN model by Chen, et al. (2024) from linear to nonlinear constraints [10]. Thereby, we address a critical limitation of surrogate models in failing to satisfy fundamental physical principles such as mass and energy conservation.

In our proposed method, we project the NN predictions onto the feasible space defined by mass and enthalpy balances, ensuring the exact satisfaction of these laws. We tested Picard-KKT-hPINN in surrogate modeling tasks, specifically for a methanol synthesis packed bed reactor. The method guarantees atomic and enthalpy balances without adding computational overhead, while also outperforming the accuracy of standard MLPs in data-scarce regimes. These results highlight the potential of Picard-KKT-hPINN to advance the adoption of NNs in relevant chemical engineering applications, including large-scale process optimization and simulation, where developing physically consistent surrogate models is crucial for ensuring both computational efficiency and reliable outcomes.

ACKNOWLEDGEMENTS

We gratefully acknowledge the support provided by Shell Global Solutions International B.V.

REFERENCES

1. Kevin McBride and Kai Sundmacher. Overview of surrogate modeling in chemical process engineering. *Chemie Ingenieur Technik*, 91(3):228–239 (2019) <https://doi.org/10.1002/cite.201800091>
2. Patrick Bedué and Albrecht Fritzsche. Can we trust ai? An empirical investigation of trust requirements and guide to successful AI adoption. *Journal of Enterprise Information Management*, 35(2):530–549. (2021) <https://doi.org/10.1108/JEIM-06-2020-0233>
3. Artur M. Schweidtmann, Jana M. Weber, Christian Wende, Linus Netze, and Alexander Mitsos. Obey validity limits of data-driven models through topological data analysis and one-class classification. *Optimization and Engineering*, 23(2):855–876. (2021) <https://doi.org/10.1007/s11081-021-09608-0>
4. M. Raissi, P. Perdikaris, and G.E. Karniadakis. 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. (2019) <https://doi.org/10.1016/j.jcp.2018.10.045>
5. Zhiyong Wu, Huan Wang, Chang He, Bingjian Zhang, Tao Xu, and Qinglin Chen. The application of physics-informed machine learning in multiphysics modeling in chemical engineering. *Industrial and Engineering Chemistry Research*, 62(44):18178–18204. (2023) <https://doi.org/10.1021/acs.iecr.3c02383>
6. Tom Beucler, Michael Pritchard, Stephan Rasp, Jordan Ott, Pierre Baldi, and Pierre Gentine. Enforcing analytic constraints in neural-networks emulating physical systems. *Physical Review Letters*, 126(9):098302. (2019) <https://doi.org/10.1103/PhysRevLett.126.098302>
7. Priya L. Donti, David Rolnick, and J. Zico Kolter. Dc3: A learning method for optimization with hard constraints. *International Conference on Learning Representations*. (2021) <https://doi.org/10.48550/arXiv.2104.12225>
8. Ferdinando Fioretto, Pascal Van Hentenryck, Terrence WK Mak, Cuong Tran, Federico Baldo, and Michele Lombardi. Lagrangian duality for constrained deep learning. *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. (2021) https://doi.org/10.1007/978-3-030-67670-4_8
9. Yuntian Chen, Dou Huang, Dongxiao Zhang, Junsheng Zeng, Nanzhe Wang, Haoran Zhang, and Jinyue Yan. Theory-guided hard constraint projection (hcp): A knowledge-based data-driven scientific machine learning method. *Journal of Computational Physics*, 445:110624. (2021) <https://doi.org/10.1016/j.jcp.2021.110624>
10. Hao Chen, Gonzalo E. Constante Flores, and Can Li. Physics-informed neural networks with hard linear equality constraints. *Computers & Chemical Engineering*, 189:108764. (2024) <https://doi.org/10.1016/j.compchemeng.2024.108764>
11. Angan Mukherjee and Debangsu Bhattacharyya. On the development of steady-state and dynamic mass-constrained neural networks using noisy transient data. *Computers & Chemical Engineering*, 187:108722. (2024) <https://doi.org/10.1016/j.compchemeng.2024.108722>
12. Artur M. Schweidtmann, Dominik Bongartz, Wolfgang R. Huster, and Alexander Mitsos. Deterministic global process optimization: Flash calculations via artificial neural networks. pages 937–942. (2019) <https://doi.org/10.1016/B978-0-12-818634-3.50157-0>
13. Giacomo Lastrucci, Maximilian F. Theisen, and Artur M. Schweidtmann. Physics-informed neural networks and time-series transformer for modeling of chemical reactors. pages 571–576. (2024) <https://doi.org/10.1016/B978-0-443-28824-1.50096-X>
14. Lee, A., Ghose, J. H., Eslick, J.C., Laird, C.D., Siirola, J.D., Zamarripa, M.A., Gunter, D., Shinn, J. H., Dowling, A. W., Bhattacharyya, D., Biegler, L. T., Burgard, A. P., & Miller, D.C. (2021). The IDAES process modeling framework and model library—Flexibility for process simulation and optimization. *Journal of Advanced Manufacturing and Processing*, 3(3), 1–30. <https://doi.org/10.1002/amp2.10095>
15. Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Volume 9 of *Proceedings of Machine Learning Research*, pages 249–256. (2010) <https://proceedings.mlr.press/v9/glorot10a.html>

© 2025 by the authors. Licensed to PSEcommunity.org and PSE Press. This is an open access article under the creative commons CC-BY-SA licensing terms. Credit must be given to creator and adaptations must be shared under the same terms. See <https://creativecommons.org/licenses/by-sa/4.0/>

