Computers and Chemical Engineering 207 (2026) 109540

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/cace

Short communication

End-to-end reinforcement learning of Koopman models for eNMPC of an air separation unit

Daniel Mayfrank [a,d], Kayra Dernek [a,b], Laura Lang [b], Alexander Mitsos [c,a,b], Manuel Dahmen [a,*]

[a] Forschungszentrum Jülich GmbH, Institute of Climate and Energy Systems, Energy Systems Engineering (ICE-1), Jülich 52425, Germany
[b] RWTH Aachen University, Process Systems Engineering (AVT.SVT), Aachen 52074, Germany
[c] JARA-ENERGY, Jülich 52425, Germany
[d] RWTH Aachen University, Aachen 52062, Germany

ARTICLE INFO

Keywords:
Economic model predictive control
Koopman
Demand response
Air separation unit
Reinforcement learning

ABSTRACT

With our recently proposed method based on reinforcement learning (Mayfrank et al., 2024), Koopman surrogate models can be trained for optimal performance in specific (economic) nonlinear model predictive control ((e)NMPC) applications. So far, our method has exclusively been demonstrated on a small-scale case study. Herein, we show that our method scales well to a more challenging demand response case study built on a large-scale model of a single-product (nitrogen) air separation unit. Across all numerical experiments, we assume observability of only a few realistically measurable plant variables. Compared to a purely system identification-based Koopman eNMPC, which generates small economic savings but frequently violates constraints, our method delivers similar economic performance while avoiding constraint violations.

1. Introduction

Data-driven dynamic models can be trained in an end-to-end fashion for optimal performance as part of (economic) (nonlinear) model predictive control ((e)(N)MPC) (e.g., Gros and Zanon, 2019 and Amos et al., 2018). We recently introduced a method (Mayfrank et al., 2024) based on reinforcement learning (RL) for end-to-end learning of Koopman *surrogate* models (Korda and Mezić, 2018) for (e)NMPC applications. Such data-driven surrogate models can make eNMPC computationally tractable in case an accurate mechanistic process model is available but using it as part of a model predictive controller is too computationally expensive. Moreover, in scenarios where no reliable model is available, the same framework can learn directly from plant data. Alternative methods for end-to-end learning of data-driven models for control applications focus on linear models (Chen et al., 2019), optimize highly-structured models with few parameters requiring expert system knowledge (Brandner et al., 2023), cannot handle hard constraints on system states (Amos et al., 2018), or are only applicable to setpoint tracking problems (Iwata and Kawahara, 2022). Our method can optimize highly parameterized and thus flexible Koopman models for control problems with state constraints and arbitrary convex objective functions.

In Mayfrank et al. (2024), we demonstrated our method in two simulated case studies (NMPC and eNMPC) based on a small model of a continuous stirred-tank reactor (CSTR) (Flores-Tlacuahuac and Grossmann, 2006) comprised of just two ordinary differential equations. The resulting policies outperformed Koopman controllers employing models that were trained using the prevailing system identification (SI) approach by (i) achieving more accurate state tracking in the NMPC case study and (ii) substantially reducing the frequency of constraint violations in the eNMPC case study. In the present contribution, we demonstrate the scalability of our method (Mayfrank et al., 2024) using a large-scale differential–algebraic equations (DAE) model of an air separation unit (ASU) (Caspari et al., 2020).

The remainder of this short paper is organized as follows: First, the ASU demand response case study is introduced in Section 2. Then, Section 3 provides a brief explanation of our method, followed by a description of the adjustments to the Koopman model architecture we employed in this work. Section 4 presents the results of the numerical experiments. Finally, Section 5 discusses the conclusions and directions for future work.

2. Demand response of an air separation unit

We consider demand response of a single-product ASU for the production of purified nitrogen based on the benchmark process presented in Caspari et al. (2020), resulting in a mid-level complexity

* Corresponding author.
E-mail address: m.dahmen@fz-juelich.de (M. Dahmen).

https://doi.org/10.1016/j.compchemeng.2025.109540
Received 10 November 2025; Received in revised form 15 December 2025; Accepted 23 December 2025
Available online 24 December 2025

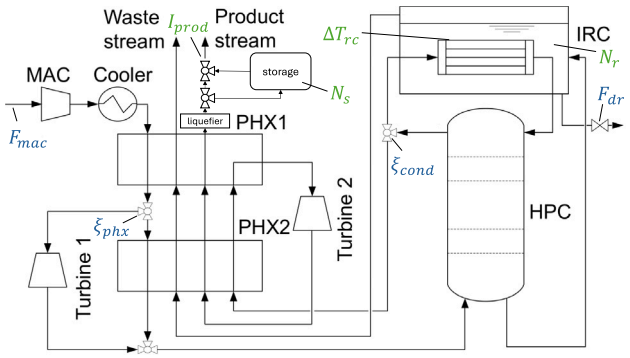

Fig. 1. Air separation process flowsheet. The following manipulated variables are shown in blue font: inlet air flow rate F_{mac}, air fraction passing through turbine 1 ξ_{phx}, distillation column reflux ratio ξ_{cond}, drain stream F_{dr}. The controlled variables are depicted in green font: product impurity I_{prod}, molar holdup in storage N_{s} and reboiler N_{r}, temperature difference between reboiler and condenser ΔT_{rc}.

control problem. The process flowsheet is shown in Fig. 1. Because RL approaches often need many policy–environment interactions to produce good results, wall-clock simulation speed is critical for simulation models that are to be used as part of the training environment. Because the full mechanistic model (Caspari et al., 2020) is computationally expensive, we construct the demand-response RL environment using the modified model of Schulze et al. (2023), which enables substantially faster simulation. The modified model is a nonlinear DAE system with 2327 algebraic and 119 differential states, implemented in Modelica and still computationally expensive, thus motivating the proposed end-to-end learning.

Ambient air is compressed in the main air compressor (MAC), pre-cooled, and then passes through a two-part multi-stream heat exchanger (MSHE), where it is cooled against returning process streams. After the first part of the MSHE (PHX1), a fraction of the air is used in turbine 1 for power generation, while the remainder is liquefied in the second part of the MSHE (PHX2). Both streams are recombined before entering the high-pressure distillation column (HPC). The oxygen-rich bottom product of the HPC is expanded and used in an integrated reboiler condenser (IRC) to cool the reflux stream. Liquid is withdrawn via a drain stream, and the exiting vapor leaves the process as waste after heat recovery in the MSHE. The nitrogen-rich top product passes through turbine 2 and a liquefier to yield liquid nitrogen, which can be stored in a product tank for flexible delivery.

The task of the eNMPC is to minimize operational cost by exploiting variations in the electricity price, while fulfilling a constant demand for liquid nitrogen and avoiding constraint violations. The operational cost is given by the overall power consumption E of the ASU multiplied by the electricity price. For the overall power consumption, the energy demand of the MAC and the liquefier, as well as the electricity generation from the turbines, are taken into account. Operational constraints and manipulated control inputs are shown in the ASU flowsheet in Fig. 1, and their respective lower and upper bounds are given in Table 1.

To use the Modelica model as part of an RL environment, we export it as a functional mock-up unit that can be simulated within Python code. At each control step in the environment, the policy receives the current state of the ASU and an electricity price prediction for the upcoming 9 h in hourly resolution. After receiving a control input from the policy, the state of the ASU is updated by simulating the model for a time step of 15 min. Furthermore, analogous to the reward calculation in Mayfrank et al. (2024), we calculate a reward based on constraint violations and electricity cost savings compared to steady-state production.

Table 1
Summary of lower (lb) and upper (ub) bounds of the operational and input variables.

Variable	lb	ub	Constraint type
I_{prod} [ppm]	0	1800	path
ΔT_{rc} [K]	2	5	path
N_r [kmol]	2	10	path
N_s [–]	0	6	path
F_{mac} [mol/s]	30	50	input
F_{dr} [mol/s]	0	2	input
ξ_{phx} [kmol]	0	0.1	input
ξ_{cond} [–]	0.51	0.54	input

3. End-to-end RL of Koopman model for eNMPC

In Mayfrank et al. (2024), we utilize Koopman models of the form proposed by Korda and Mezić (2018): (i) A nonlinear state observation function $\boldsymbol{\psi}_{\boldsymbol{\theta}} : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_z}$ that transforms the initial system state $\boldsymbol{x}_0 \in \mathbb{R}^{n_x}$ into the initial Koopman state $\boldsymbol{z}_0 \in \mathbb{R}^{n_z}$, where typically $n_z \gg n_x$: $\boldsymbol{z}_0 = \boldsymbol{\psi}_{\boldsymbol{\theta}}(\boldsymbol{x}_0)$. (ii) The $\boldsymbol{A}_{\boldsymbol{\theta}} \in \mathbb{R}^{n_z \times n_z}$ and $\boldsymbol{B}_{\boldsymbol{\theta}} \in \mathbb{R}^{n_z \times n_u}$ matrices, which linearly approximate the evolution of the Koopman state, driven by external control inputs $\boldsymbol{u}_t \in \mathbb{R}^{n_u}$: $\boldsymbol{z}_{t+1} = \boldsymbol{A}_{\boldsymbol{\theta}} \boldsymbol{z}_t + \boldsymbol{B}_{\boldsymbol{\theta}} \boldsymbol{u}_t$. (iii) The $\boldsymbol{C}_{\boldsymbol{\theta}} \in \mathbb{R}^{n_x \times n_z}$ matrix, which linearly transforms the Koopman state $\boldsymbol{z}_t$ into a predicted system state $\hat{\boldsymbol{x}}_t$: $\hat{\boldsymbol{x}}_t = \boldsymbol{C}_{\boldsymbol{\theta}} \boldsymbol{z}_t$. Such models can be trained by adjusting the parameters $\boldsymbol{\theta}$.

RL algorithms (e.g., Schulman et al., 2017) can be used to optimize parameterized policies $\boldsymbol{\pi}_{\boldsymbol{\theta}}(\boldsymbol{u}_t | \boldsymbol{x}_t) : \mathbb{R}^{n_x} \mapsto \mathbb{R}^{n_u}$, which map current system states $\boldsymbol{x}_t$ to control actions $\boldsymbol{u}_t$, by maximizing the expected cumulative reward of the policy. Therefore, to train a Koopman model in an end-to-end manner for a particular eNMPC application via RL, we construct an (automatically differentiable) eNMPC policy based on the Koopman model. When used as part of an eNMPC policy, Koopman models of the above stated form (Korda and Mezić, 2018) result in convex optimal control problems (OCPs) if the stage cost and all additional user-defined constraints are convex. By using the automatically differentiable solver *cvxpylayers* (Agrawal et al., 2019) for convex optimization problems, we can obtain $\partial \boldsymbol{u}_t^* / \partial \boldsymbol{\theta}$ via implicit differentiation of the KKT conditions. The solution map $\boldsymbol{\theta} \mapsto \boldsymbol{u}_t^*$ is only piecewise differentiable; nevertheless our numerical experience shows that the values returned from *cvxpylayers* as derivatives work well for the first-order training.

In our previous work Mayfrank et al. (2024), the path constraints for the controlled variables were formulated as inequality constraints. However, when applied to the ASU model, the end-to-end learning approach showed no improvement over the initial guess. We attribute this to poor gradient estimation caused by switching between active and inactive inequality constraints. To address this issue, we reformulate the OCP in the end-to-end RL by replacing the inequality constraints with equality constraints with slack variables $\boldsymbol{s}$, and penalize the slack variables through quadratically smoothed hinge loss (Zhang, 2004), defined as $L(s_i, \delta) = M \left[\max\left(0, |s_i| - \frac{1}{2}\Delta g_i + \delta\right)\right]^2$, where Δg_i represents the admissible range of the corresponding constraint (see supplementary material). The variable $M > 0$ is a penalty coefficient to balance the penalty and the objective performance. To more strongly penalize constraint violations, we introduce penalty scaling values δ that slightly tighten the bounds of the inequality constraints, leading to a more conservative control behavior.

Our workflow is shown in Fig. 2. Initial values for $\boldsymbol{\theta}$ are obtained through standard system identification (SI), i.e., we generate simulation data using the mechanistic model and fit $\boldsymbol{\theta}$ to that data. A more detailed description of the SI is provided in the supplementary materials. The model parameters $\boldsymbol{\theta}$ are then fine-tuned for optimal performance in a specific control task using Proximal Policy Optimization (PPO) (Schulman et al., 2017), an actor–critic RL algorithm.

The ASU model is a DAE system that features system outputs $\boldsymbol{y}_t$ that exhibit discontinuities when control inputs change non-continuously.

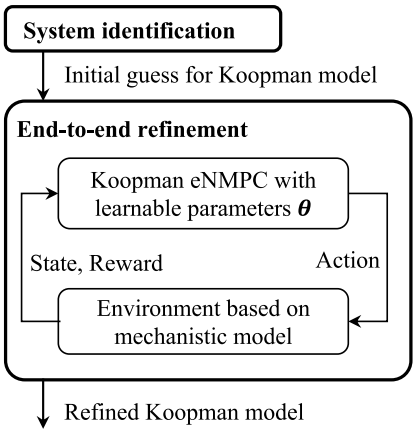

Fig. 2. Workflow for end-to-end learning of a Koopman model for eNMPC.

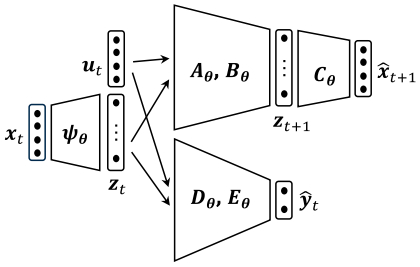

Fig. 3. Koopman model architecture.

The Koopman model architecture proposed by Korda and Mezić (2018), which we used in Mayfrank et al. (2024), cannot reflect such behavior since the control inputs enter directly into the predictor-part of the model, which produces the latent space vector one time step into the future. To enable the Koopman models to represent instantaneous output responses to input changes, we extend the framework of Korda and Mezić (2018) by adding a second decoder, i.e., $\boldsymbol{y}_t = \boldsymbol{D}_{\boldsymbol{\theta}}\boldsymbol{z}_t + \boldsymbol{E}_{\boldsymbol{\theta}}\boldsymbol{u}_t$, with learnable $\boldsymbol{D}_{\boldsymbol{\theta}} \in \mathbb{R}^{n_y \times n_z}$ and $\boldsymbol{E}_{\boldsymbol{\theta}} \in \mathbb{R}^{n_y \times n_u}$ matrices. Note that this second decoder was not needed in our previous work (Mayfrank et al., 2024) because the CSTR investigated there was described by an ordinary differential equation (ODE) system for which discontinuities in control inputs do not cause discontinuities in states/outputs. The overall architecture of the Koopman model is visualized in Fig. 3.

We do not assume full observability of the ASU. Instead, we use only the following states as inputs to the Koopman surrogate model: the product purity I_{prod}, the temperature difference between reboiler and condenser ΔT_{rc}, the reboiler holdup N_{r}, and the temperature of tray 20 in the distillation column $T_{\text{tray,20}}$. Based on this information, the model constructs a latent representation $\boldsymbol{z}_t$ of the state of the ASU. The vector of control inputs $\boldsymbol{u}_t$ consists of the four inputs listed in Table 1. The model predicts the evolution of the control-relevant entries of $\boldsymbol{x}_t$ through the $\boldsymbol{A}_{\boldsymbol{\theta}}$, $\boldsymbol{B}_{\boldsymbol{\theta}}$, and $\boldsymbol{C}_{\boldsymbol{\theta}}$ matrices. The resulting vector $\hat{\boldsymbol{x}}_t \in \mathbb{R}^3$ consists of I_{prod}, ΔT_{rc}, and N_r. The control-relevant, discontinuous outputs $\boldsymbol{y}_t \in \mathbb{R}^2$ are the energy demand E of the process and the production rate $\dot{n}_{product}$. The latter is used to calculate the molar holdup N_{s} of the storage tank through a mass balance, where the change in tank storage is given by the difference between the inflow and outflow rates, i.e., $\frac{d N_{\text{s}}}{dt} = \dot{n}_{\text{product}} - \dot{n}_{\text{demand}}$. The molar holdup N_{s} is upper and lower bounded to reflect the physical storage capacity limits of the tank.

We choose a latent space dimensionality of 10 for the Koopman model, i.e., $\boldsymbol{z}_t \in \mathbb{R}^{10}$. Altogether, the Koopman model thus consists of the matrices $\boldsymbol{A}_{\boldsymbol{\theta}} \in \mathbb{R}^{10\times10}$, $\boldsymbol{B}_{\boldsymbol{\theta}} \in \mathbb{R}^{10\times4}$, $\boldsymbol{C}_{\boldsymbol{\theta}} \in \mathbb{R}^{3\times10}$, $\boldsymbol{D}_{\boldsymbol{\theta}} \in \mathbb{R}^{2\times10}$, $\boldsymbol{E}_{\boldsymbol{\theta}} \in \mathbb{R}^{2\times4}$, and an encoder $\boldsymbol{\psi}_{\boldsymbol{\theta}} : \mathbb{R}^4 \to \mathbb{R}^{10}$. The encoder is a feedforward neural network with two hidden layers with 50 neurons each, and tanh activation functions.

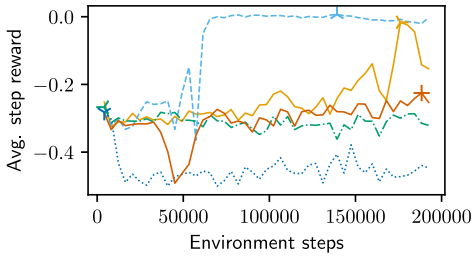

Fig. 4. Learning progress during end-to-end model refinement. Each line represents one training run. For each training run, a marker depicts the highest average reward achieved in one policy rollout.

4. Numerical experiments

All training code used in this work, including the implementation of the ASU demand response RL environment, is available online.[1]

We obtain an initial guess for the Koopman model via the iterative data sampling and system identification approach outlined in the supporting information. Then, we repeat the end-to-end refinement five times using different fixed seeds in every training run. We use the historic German day-ahead electricity prices from 2023-01-01 to 2023-12-31 (EPEX, 2023) for training.

Fig. 4 illustrates the evolution of the reward over 200,000 environment steps for each training run. Each training run takes approx. two days on a Windows 11 workstation with an Intel Core i7-14700 CPU. Three elements contribute to overall training time: (i) simulation of the mechanistic DAE environment, (ii) repeated solution of the convex eNMPC problem, and (iii) gradient-based policy optimization. The dominant contributor to the training time is the policy optimization step, which requires backpropagating through the Koopman eNMPC. In contrast, pure policy inference typically takes well under one second per control step, making the controller easily real-time capable. After training has completed, our final performance evaluation is based on the policy that attained the highest average reward during a policy rollout. Consequently, our primary metric of interest is the maximum average reward achieved by each training run, rather than the reward at the final training step. Three out of five training runs demonstrate a substantial improvement in the policy's average reward compared to the initial guess. After training, we test the performance of the policy that obtained the highest average reward in a three-day test episode. The test electricity price trajectory is generated using the method by Papadimitriou et al. (2024), which constructs representative price profiles from 2023 historical data while preserving key statistics (mean and variance). The resulting profiles are distinct from those used during training, ensuring a clear separation between training and testing data.

In the test episode, the eNMPC employing the Koopman model obtained solely via system identification (from hereon called *Koopman-SI* policy) saves 1 % of electricity cost compared to steady-state production while producing small constraint violations in 16.3 % of the time steps, resulting in an average reward of −0.26. The eNMPC employing the end-to-end refined Koopman model (*Koopman-PPO*) improves performance, and produces 2 % cost savings. Furthermore, it does so without violating any constraints in the test episode, thus yielding an average reward of 0.01.

Fig. 5 illustrates the behavior of the policies in the three-day test episode. We show the evolution of ΔT_{IRC} and N_R, since these are the

[1] https://jugit.fz-juelich.de/iek-10/public/optimization/koopmanenmpc4asu

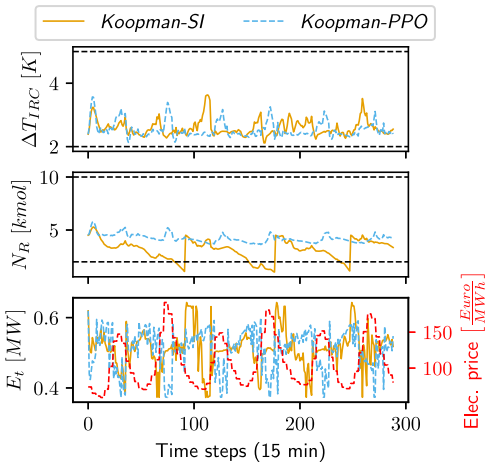

Fig. 5. Comparison of the control behavior of *Koopman-SI* and *Koopman-PPO*.

only states that operate close to their bounds, as well as the evolution of the energy demand E. Both policies exhibit an intuitive inverse relationship between the electricity price and the energy demand. It can be seen that the behavior of the policies differs for states ΔT_{IRC} and N_R. While *Koopman-SI* exhibits violations in N_R, *Koopman-PPO* maintains operation safely within the operation bounds. Both policies exhibit high-frequency oscillations with respect to E_t.

As stated above, the architecture of the Koopman models that we use (see Fig. 3) results in convex OCPs, which are relatively easy to solve. In the 72 h test episode, i.e., 288 control steps of 15 min length each, the inference time of the *Koopman-PPO* policy was between 0.1 s and 3 s with an average time of 0.5 s.

5. Conclusion

We apply our previously published method (Mayfrank et al., 2024) for end-to-end learning of Koopman surrogate models for (e)NMPC applications to a high-dimensional nonlinear demand response problem based on a mechanistic model of an ASU, demonstrating its applicability to complex, real-world systems.

Among the five independent end-to-end training runs, three significantly improve eNMPC performance over the SI baseline, reducing and even eliminating constraint violations while maintaining high economic efficiency. Compared to our earlier two-state CSTR case study (Mayfrank et al., 2024), where all ten training runs consistently improved performance and rewards saturated after roughly 180,000 steps, the present large-scale ASU problem poses a more challenging learning task, reflected in the lower fraction of successful runs. Note that while our method strongly incentivizes constraint satisfaction, it does *not* provide formal theoretical guarantees.

Our method produces data-driven predictive control policies that strike an exceptional balance between control performance (high economic performance and consistent constraint satisfaction) and computational efficiency (policy inference time $\ll$ sampling time). By demonstrating its scalability to an industrially-relevant process, we show that our method could be applicable to complex real-world control problems where mechanistic predictive control policies are not real-time capable and system identification-based data-driven controllers produce unsatisfactory performance.

Future work should test our method on a real-world process. Because the Koopman surrogate model is trained in an environment based on a mechanistic simulation model, it may overfit to systematic simulation errors, e.g., parameter mismatch or unmodeled dynamics. The resulting controller could perform suboptimally on the real process. This risk does not exist in the present work, since training and final evaluation use the same simulated environment. In the described real-world scenario, our method may be used to further refine the surrogate model via interactions with the real process. In such an approach, the sample efficiency of the learning procedure would become critical, which could motivate employing a model-based RL algorithm as done in Mayfrank et al. (2025).

CRediT authorship contribution statement

Daniel Mayfrank: Writing – review & editing, Writing – original draft, Visualization, Software, Methodology, Investigation, Conceptualization. **Kayra Dernek:** Writing – review & editing, Visualization, Software, Methodology, Investigation. **Laura Lang:** Writing – review & editing, Writing – original draft, Methodology. **Alexander Mitsos:** Writing – review & editing, Supervision, Funding acquisition, Conceptualization. **Manuel Dahmen:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Daniel Mayfrank reports financial support was provided by Helmholtz Association of German Research Centres. Laura Lang reports financial support was provided by Federal Ministry of Education and Research (BMBF). Given his role as editor of Computers & Chemical Engineering, Alexander Mitsos had no involvement in the peer review of this article and had no access to information regarding its peer review. Full responsibility for the editorial process for this article was delegated to another journal editor. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was performed as part of the Helmholtz School for Data Science in Life, Earth and Energy (HDS-LEE) and received funding from the Helmholtz Association of German Research Centres. Additionally, the authors gratefully acknowledge the financial support of the Kopernikus project SynErgie by the Federal Ministry of Education and Research (BMBF), and the project supervision by the project management organization Projektträger Jülich (PtJ).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.compchemeng.2025.109540.

Data availability

Data will be made available on request.

References

Agrawal, A., Amos, B., Barratt, S., Boyd, S., Diamond, S., Kolter, J.Z., 2019. Differentiable convex optimization layers. Adv. Neural Inf. Process. Syst. 32, 9558–9570.

Amos, B., Jimenez, I., Sacks, J., Boots, B., Kolter, J.Z., 2018. Differentiable MPC for end-to-end planning and control. Adv. Neural Inf. Process. Syst. 31, 8299–8310.

Brandner, D., Talis, T., Esche, E., Repke, J.-U., Lucia, S., 2023. Reinforcement learning combined with model predictive control to optimally operate a flash separation unit. In: Computer Aided Chemical Engineering, vol. 52, Elsevier, pp. 595–600.

Caspari, A., Tsay, C., Mhamdi, A., Baldea, M., Mitsos, A., 2020. The integration of scheduling and control: Top-down vs. bottom-up. J. Process Control 91, 50–62.

Chen, B., Cai, Z., Bergés, M., 2019. Gnu-RL: A precocial reinforcement learning solution for building HVAC control using a differentiable MPC policy. In: Proceedings of the 6th ACM International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation. pp. 316–325.

EPEX, 2023. EPEX SPOT market data. https://www.epexspot.com. (Accessed 02 January 2025).

Flores-Tlacuahuac, A., Grossmann, I.E., 2006. Simultaneous cyclic scheduling and control of a multiproduct CSTR. Ind. Eng. Chem. Res. 45 (20), 6698–6712.

Gros, S., Zanon, M., 2019. Data-driven economic NMPC using reinforcement learning. IEEE Trans. Autom. Control 65 (2), 636–648.

Iwata, T., Kawahara, Y., 2022. Data-driven end-to-end learning of pole placement control for nonlinear dynamics via Koopman invariant subspaces. arXiv preprint arXiv:2208.08883.

Korda, M., Mezić, I., 2018. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. Automatica 93, 149–160.

Mayfrank, D., Mitsos, A., Dahmen, M., 2024. End-to-end reinforcement learning of Koopman models for economic nonlinear model predictive control. Comput. Chem. Eng. 190, 108824.

Mayfrank, D., Velioglu, M., Mitsos, A., Dahmen, M., 2025. Sample-efficient reinforcement learning of Koopman eNMPC. arXiv preprint arXiv:2503.18787.

Papadimitriou, C., Schulze, J.C., Mitsos, A., 2024. Representative electricity price profiles for European day-ahead and intraday spot markets. arXiv preprint arXiv:2405.14403.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O., 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

Schulze, J.C., Doncevic, D.T., Erwes, N., Mitsos, A., 2023. Data-driven model reduction and nonlinear model predictive control of an air separation unit by applied Koopman theory. arXiv preprint arXiv:2309.05386.

Zhang, T., 2004. Solving large scale linear prediction problems using stochastic gradient descent algorithms. In: Twenty-First International Conference on Machine Learning - ICML'04. p. 116.