# Deep Dictionary-Free Method for Identifying Linear Model of Nonlinear System with Input Delay

Patrik Valábek, Marek Wadinger, Michal Kvasnica, Martin Klaučo[†]

*Institute of Information Engineering, Automation, and Mathematics*
*Slovak University of Technology in Bratislava*
Bratislava, Slovakia
patrik.valabek@stuba.sk

*Abstract*—Nonlinear dynamical systems with input delays pose significant challenges for prediction, estimation, and control due to their inherent complexity and the impact of delays on system behavior. Traditional linear control techniques often fail in these contexts, necessitating innovative approaches. This paper introduces a novel approach to approximate the Koopman operator using an LSTM-enhanced Deep Koopman model, enabling linear representations of nonlinear systems with time delays. By incorporating Long Short-Term Memory (LSTM) layers, the proposed framework captures historical dependencies and efficiently encodes time-delayed system dynamics into a latent space. Unlike traditional extended Dynamic Mode Decomposition (eDMD) approaches that rely on predefined dictionaries, the LSTM-enhanced Deep Koopman model is dictionary-free, which mitigates the problems with the underlying dynamics being known and incorporated into the dictionary. Quantitative comparisons with extended eDMD on a simulated system demonstrate highly significant performance gains in prediction accuracy in cases where the true nonlinear dynamics are unknown and achieve comparable results to eDMD with known dynamics of a system.

*Index Terms*—Koopman operator, nonlinear systems, identification, input delays

## I. INTRODUCTION

Understanding and controlling nonlinear dynamical systems is a fundamental challenge across various scientific and engineering disciplines. Traditional linear control techniques often fall short when applied to such systems due to their inherent nonlinearity. This becomes even more challenging in time-delayed systems, where the presence of delays introduces additional complexity by coupling past states with the current system dynamics. These delays can significantly degrade control performance and stability if not adequately accounted for. A promising approach to this problem is the identification of coordinate transformations that render nonlinear dynamics approximately linear, thus enabling the application of linear theory for prediction, estimation, and control.

Koopman analysis, a technique that facilitates the linearization of nonlinear systems through the Koopman operator, has gained considerable traction in recent years [1], [2]. The Koopman operator offers a global linearization of dynamics by mapping the original nonlinear system into a higher-dimensional space where the dynamics can be represented linearly [3], [4]. This method is particularly appealing because it shifts the complexity from the system's nonlinear equations to the eigenfunctions of the Koopman operator. These eigenfunctions span an invariant subspace, enabling the system's dynamics to be represented by a finite-dimensional matrix within this subspace, simplifying the analysis and computation of the dynamics.

Finite-dimensional approximations of the Koopman operator are often achieved using Dynamic Mode Decomposition (DMD) [5]. While DMD identifies spatio-temporal coherent structures from high-dimensional systems, it typically fails to capture nonlinear transients due to its reliance on linear measurements. To address this, Extended DMD (eDMD) [6] incorporates a dictionary, improving its capability to model nonlinear systems. In this context dictionary stands for a collection or set of basis functions or candidate models used to represent or approximate the nonlinear relationships. However, eDMD faces challenges such as high dimensionality and closure issues, which arise because there is no guarantee that the nonlinear measurements form a Koopman invariant subspace [7]. Consequently, the identification and representation of Koopman eigenfunctions remain crucial tasks, motivating the use of advanced deep learning techniques.

To overcome the limitations of eDMD, deep neural networks (DNNs) have been employed to learn Koopman operator representations. DNNs remove the bottleneck of predefined dictionaries by creating linear representations of a system through nonlinear transformations of individual neurons. These neurons combine to form complex functions parametrized by tunable weights and biases, allowing the network to adaptively learn the optimal transformations during training. This

approach significantly enhances the fidelity of Koopman operator models, particularly in multi-step prediction tasks, thus improving long-term forecasting of nonlinear dynamics [8].

An essential aspect of modeling dynamical systems is accounting for time delays, which are common in many physical and industrial processes. Time delay can significantly affect system behavior and control performance. Introducing time-delayed embeddings of control action in DMD improves the identification of the system with input delays while not explicitly identifying the time delays. Various methods exist to identify and model time delays, including state-space realization approaches and correlation analysis. The state-space realization approach involves constructing a state-space representation based on input-output data from experiments, extracting time-delay information from the system's impulse response [9]. Correlation analysis, on the other hand, focuses on identifying time delays in dynamic processes with disturbances, aiming to improve control accuracy and stability [10]. The limitations of correlation analysis addressed by eDMD with time-delayed embeddings include its inability to account for system dynamics, as eDMD captures the full state evolution through time-delay coordinates, and its sensitivity to disturbances and noise, as eDMD leverages a higher-dimensional representation to better isolate underlying dynamics.

This paper presents a dictionary-free method for learning linear representations of nonlinear systems with input delays using deep neural networks. By leveraging the strengths of deep learning in generating and updating nonlinear transformations, our approach aims to overcome the limitations of traditional Koopman operator methods and provide a robust framework for modeling and controlling nonlinear systems with time delays. The following sections are structured to guide the reader through the theoretical foundations, methodological advancements, and practical insights of our approach, addressing key challenges and presenting proposed solutions in this domain.

In Section II, we detail the methodological framework of our approach. Subsection II-A introduces the implementation of the LSTM-enhanced Deep Koopman model, emphasizing its ability to capture nonlinear dynamics through latent space representations, where LSTM refers to deep neural network architecture Long-Short Term Memory, which is a type of recurrent neural network. Subsection II-B then describes the explicit loss function design, focusing on how it ensures accurate prediction and stability in the learning process.

Section III presents the results of our approach, providing quantitative and qualitative evaluations. Subsection III-A compares our method with the established extended Dynamic Mode Decomposition (eDMD), highlighting improvements in capturing time-delayed nonlinear dynamics and demonstrating the advantages of our approach in representing unknown dynamics.

## II. LSTM-ENHANCED DEEP KOOPMAN APPROACH

The foundation of the proposed approach lies in approximating the Koopman operator, a linear operator that represents the dynamics of nonlinear systems in an embedded space. Formally, we aim to find the underlying linear dynamics in state space equation covering

$$z_{k+1} = A_{\mathcal{K}} z_k + B_{\mathcal{K}} u_k. \tag{1}$$

where the Koopman matrices $A_{\mathcal{K}}$ and $B_{\mathcal{K}}$ are the solution for the optimization problem:

$$\text{argmin}_{\mathcal{K}} \sum_{k=0}^{m} \| z_{k+1} - A_{\mathcal{K}} z_k - B_{\mathcal{K}} u_k \|_F, \tag{2}$$

where $z_k \in \mathbb{R}^n$ represent the vector of lifted states and $z_{k+1}$ corresponding states in next step, respectively, $\| \cdot \|_F$ denotes the Frobenius norm and $m$ denotes the number of data snapshots. This formulation provides the basis for deriving a linear approximation of the underlying dynamics, even for systems with time delays.

The proposed approach leveraging the power of a long short-term memory (LSTM) layer encoding the history of a system is a novel approach for data-based identification of approximation of the Koopman operator. It can accurately identify the linear model for processes with time delays. Its basis comes from the Deep Koopman Operator (DKO) [11] approach.

LSTM-enhanced Deep Koopman is built to approximate the Koopman operator for the systems with time delays. Thanks to the LSTM layer that processes the system's history denoted $H$, we can effectively capture the time delays in the system. Time delays are encoded as the hidden states of the last LSTM layer denoted $h_k$. Thanks to this encoding (hidden states of LSTM), we can obtain the relevant information about the system's past.

For example, using the LSTM layer, we can obtain the chosen number (this is a hyperparameter) of hidden states. These hidden states are then used as input to the encoder part of our approach. This way, we can reduce the amount of input data to the model compared to the Deep Koopman model, which would take the whole history of the system as input. This would be very inefficient and computationally more expensive.

Also, compared to the DMD, where we would use the history of measurements, we can have a smaller Koopman matrix $\mathcal{K}$. Considering, for example, the last $l_{\text{his}}$ measurements of the system states $x \in \mathbb{R}^n$, the $A_{\mathcal{K}}$ would have the size $l_{\text{his}} n \times l_{\text{his}} n$ and the $B_{\mathcal{K}}$ would have the size $l_{\text{his}} n \times m$, where $m$ is the number of inputs to the system $u \in \mathbb{R}^m$. This is often a matrix that is more expensive to compute and store if we are dealing with a long history and many states compared to the LSTM-enhanced deep Koopman approach. On the other hand, the LSTM layer can capture the relevant information about the system's past and reduce the amount of input data in the model.

### A. Implementation of LSTM-enhanced Deep Koopman

LSTM-enhanced Deep Koopman was implemented in the Python programming language using the NeuroMANCER [12]
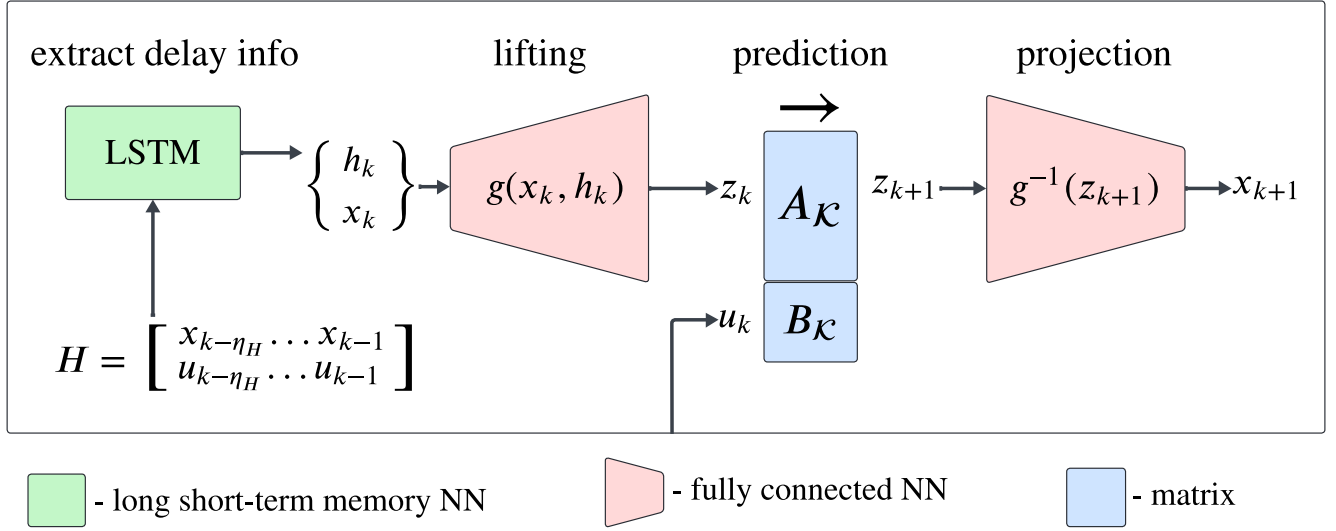
Fig. 1. The LSTM-enhanced Deep Koopman algorithm architecture. The model is built on the autoencoder architecture of the Deep Koopman Operator. The history of state and input data is first extracted using the LSTM layer. The hidden states of the last LSTM layer are then concatenated with the last state as input to the autoencoder part. After lifting the states, there is a prediction layer represented with the matrices $A_\mathcal{K}$ and $B_\mathcal{K}$. After the prediction layer, the lifted states are projected back to states using the decoder part of the autoencoder.

library. The NeuroMANCER library does not provide the use of LSTM layers, so we implemented them into the library.

The schematic of LSTM-enhanced Deep Koopman architecture can be seen in Fig. 1. The model is built on the autoencoder architecture of the DKO. Before the prediction we compute the history $H$ of our system:

$$H = \begin{bmatrix} x_{k-\eta_H} & \cdots & x_{k-1} \\ u_{k-\eta_H} & \cdots & u_{k-1} \end{bmatrix}, \tag{3}$$

where $\eta_H$ is the number of time steps in the history of the system. We set this parameter to be higher value as our observed or estimated time delay. The $H$, consisting of state and input data, is first encoded using the LSTM layer. The hidden states of the last LSTM layer are then concatenated with the current state as input to the autoencoder part. The autoencoder part remains the same as for the DKO.

### B. Explicit Loss Function

In the training of the LSTM-enhanced Deep Koopman model, we are also using the same loss function as in the training of the DKO model. This section is important for the reproducibility of the results. The loss function combines the reconstruction loss, one-step output prediction loss, output trajectory prediction loss, and latent trajectory prediction loss. These loss functions are described in the [11]. This function is one of the most important parts of the training process. Compared to DMD, where the loss function is only the mean squared error between the predicted and true future states for a one-step prediction, the Deep Koopman methods also use a multi-step prediction in a loss function. In identifying a system, by using the multi-step prediction, we can better

capture the system's dynamics with input delays as opposed to only one-step prediction loss.

*Reconstruction loss:* is the mean squared error between the state $x_k$ and the encoded (lifted) and decoded (unlifted) state $g^{-1}(g(x_k, h_k))$:

$$\mathcal{L}_{\text{rec}} = \left\| x_k - g^{-1}(g(x_k, h_k)) \right\|^2. \tag{4}$$

This loss is the basic loss function for the autoencoder part of the model. It ensures that we can correctly encode and decode the state of the system.

*One step output prediction loss:* is the mean squared error between the predicted state $\hat{x}_{k+1}$ and the true state $x_{k+1}$:

$$\mathcal{L}_{\text{step}} = \left\| \hat{x}_{k+1} - x_{k+1} \right\|^2, \tag{5a}$$

$$\hat{x}_{k+1} = g^{-1}(A_\mathcal{K} g(x_k, h_k) + B_\mathcal{K} u_k), \tag{5b}$$

where $u_k$ is the input flow rate at time $k$, the $A_\mathcal{K}$ and $B_\mathcal{K}$ are the identified Koopman matrices.

*Output trajectory prediction loss:* is the mean squared error between the predicted trajectory $\hat{x}_{k+1:k+N_L}$ and the true trajectory $x_{k+1:k+N_L}$:

$$\mathcal{L}_{\text{pred}} = \sum_{i=1}^{N_L} \left\| \hat{x}_{k+i} - x_{k+i} \right\|^2, \tag{6a}$$

$$\hat{x}_{k+i} = g^{-1}(\hat{z}_{k+i}), \tag{6b}$$

$$\hat{z}_{k+i} = A_\mathcal{K} \hat{z}_{k+i-1} + B_\mathcal{K} u_{k+i-1}, \tag{6c}$$

$$\hat{z}_k = g(x_k, h_k), \tag{6d}$$

where $N_L$ is the prediction horizon in loss function specified by a user. In this case, the predicted trajectory $\hat{x}_{k:k+N_L}$ is predicted based on the $x_k$, $h_k$ and the input trajectory

$u_{k:k+N_L-1}$. This loss function is used to capture the dynamics of the system and also input delays. By using the multi-step prediction loss, we can better capture the dynamics of the system as opposed to only one-step prediction loss.

*Latent trajectory prediction loss:* is the mean squared error between the lifted predicted trajectory $\hat{z}_{k+1:k+N_L}$ and the lifted true trajectory $z_{k+1:k+N_L}$:

$$\mathcal{L}_{\text{lpred}} = \sum_{k=1}^{N_L} \|\hat{z}_{k+i} - z_{k+i}\|^2, \tag{7a}$$

$$z_{k+i} = g(x_{k+i}, h_{k+i}), \tag{7b}$$

$$\hat{z}_{k+i} = A_{\mathcal{K}}\hat{z}_{k+i-1} + B_{\mathcal{K}}u_{k+i-1}, \tag{7c}$$

$$\hat{z}_k = g(x_k, h_k), \tag{7d}$$

where this loss function works with the same principle as the output trajectory prediction loss II-B, but in the space of lifted states.

The final loss function is a weighted sum of all the loss functions:

$$\mathcal{L} = w_{\text{rec}}\mathcal{L}_{\text{rec}} + w_{\text{step}}\mathcal{L}_{\text{step}} + w_{\text{pred}}\mathcal{L}_{\text{pred}} + w_{\text{lpred}}\mathcal{L}_{\text{lpred}}, \tag{8}$$

where $w_{\text{rec}}$, $w_{\text{step}}$, $w_{\text{pred}}$, and $w_{\text{lpred}}$ are the weights for the reconstruction loss, one-step output prediction loss, output trajectory prediction loss, and latent trajectory prediction loss, respectively. They are set by the user and are used to balance the importance of the loss functions in the training process.

## III. CASE STUDY

### A. Comparison with eDMD

This section presentes the comparison of the performance of LSTM-enhanced Deep Koopman with eDMD on a simulated two-tank system without interaction with input delays. The system is described by the following equations:

$$\frac{dh_1(t)}{dt} = q(t-\tau) - \frac{k_1}{F_1}\sqrt{h_1(t)}$$
$$\frac{dh_2(t)}{dt} = \frac{k_1}{F_2}\sqrt{h_1(t)} - \frac{k_2}{F_2}\sqrt{h_2(t)}, \tag{9}$$

where $h_1(t)$ and $h_2(t)$ are the water levels in tanks 1 and 2, respectively, $q(t)$ is the input flow rate, $\tau$ is the input delay, and $k_1$, $k_2$ are the flow rate constants of the valves and $F_1$, $F_2$ are the cross-sectional areas of the tanks.

The experimental data were acquired by simulating this system at sampling period $T_s = 10\,\text{s}$ with input delay $\tau = 20T_s$. Random change in the input flow rate from the interval $q_{\min} = 0.0\,\text{m}^3\text{s}^{-1}$, $q_{\max} = 0.03\,\text{m}^3\text{s}^{-1}$ was applied to the system. The simulation yield $4 \cdot 10^5$ samples. Gaussian noise with standard deviation of 0.1 was added to the data to simulate real-world conditions.

The data were split into training and testing sets with a ratio of 50:50. The training set was used to train the models, while the testing set was used to evaluate their performance. The models were trained to predict the water levels in tanks 1 and 2 for the whole testing set based on the applied input flow rate and the initial state of the system.

For reference, we use eDMD with dictionary of polynomials up to degree 2, the correct governing nonlinear dynamics, which is square root of the water levels and the time-delayed embeddings of the input flow rate composed of the previous 20 samples. For comparison, eDMD without the square root of the water levels was also used, simulating a situation where the true nonlinear terms are not known. Therefore, they are not included in the dictionary, which is often the case in more complex systems. Lifted states were obtained by concatenating the dictionary terms and time-delayed embeddings, and the Koopman matrix was learned using eDMD.

The LSTM-enhanced Deep Koopman model consists of an LSTM layer with one hidden layer with 8 units to extract information about the time delays in the system from the history of the system. The concatenated information is then transformed into lifted states using an encoder layer with a fully connected network. The lifting network consists of input layer with 10 neurons representing inputs $x$ and $h$, two hidden layers, each with 60 neurons and lifted state layer consisting of 40 neurons representing lifted states $z$. After lifting the states, there is a prediction layer represented with the matrices $A_{\mathcal{K}}$ ($40 \times 40$) and $B_{\mathcal{K}}$ ($40 \times 1$). After the prediction layer, the lifted states are projected back to 2 states representing the water level in the tanks, using the decoder part of the autoencoder with two hidden layers, each containing 60 neurons. For the weights in (8) we set $w_{\text{rec}} = 0$, $w_{\text{step}} = 1$, $w_{\text{pred}} = 10$, and $w_{\text{lpred}} = 1$. The model was trained using PyTorch for 1500 epochs with a batch size of 100 using the Adam optimizer with a learning rate of 0.001.

Figure 2 shows the predicted and actual water levels in tanks 1 and 2 for the two tank systems using LSTM-enhanced Deep Koopman and eDMD with and without the square root of the water levels. All three algorithms were initialized with access to same $H$. The results demonstrate that LSTM-enhanced Deep Koopman outperforms eDMD, which does not include true nonlinear behavior in terms of prediction accuracy, capturing the system's dynamics more effectively over the testing set. eDMD without the square root of the water levels exhibits systematic overestimation of the influence of input flow rate step change, leading to a positive bias in the prediction of water levels. At the same time, LSTM-enhanced Deep Koopman provides more accurate and consistent predictions. The linear model generated by LSTM-enhanced Deep Koopman displays non-minimum phase behavior, which is actually a possible approach in the identification of systems with input delays. Meanwhile, eDMD with the square root of the water levels provides accurate prediction, as it includes the true nonlinear terms in the dictionary, thanks to prior knowledge of the system's dynamics.

The performance of the models was evaluated using the mean absolute error (MAE) between the predicted and actual water levels in tanks 1 and 2. The results are presented in Table I. LSTM-enhanced Deep Koopman achieved a significantly lower MAE compared to eDMD without square root, indicating its superior performance in modeling the system with input delays in cases where true nonlinear dy-
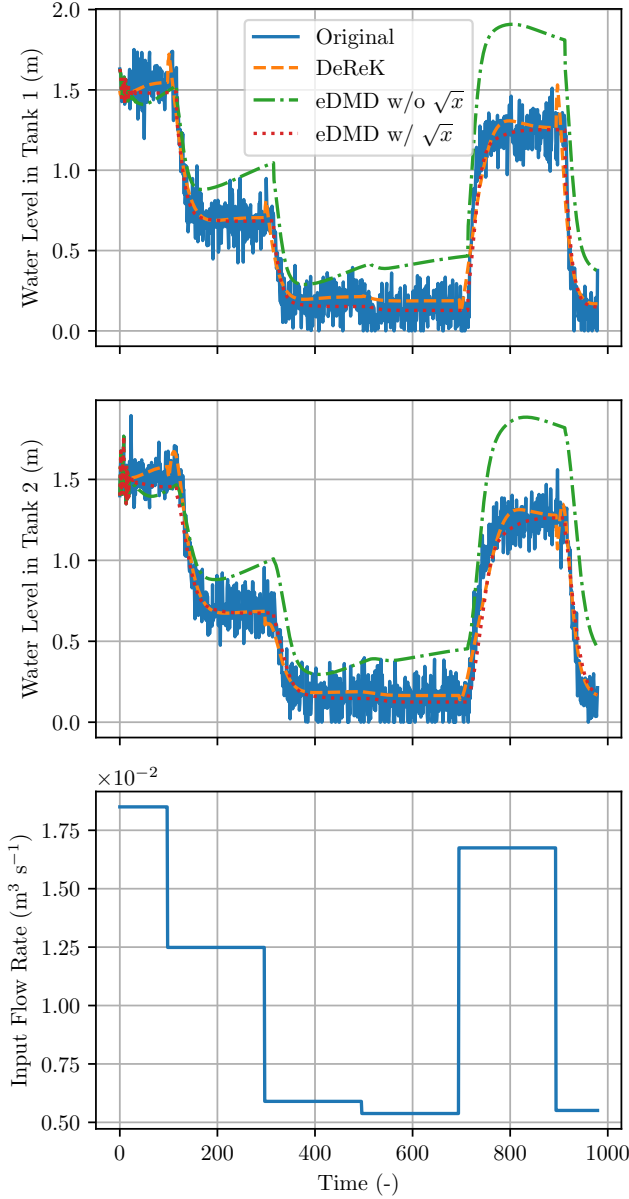
compared to eDMD. This is thanks to the fact that the history of the system is encoded and does not need to be introduced whole to the model during lifted states generation. This results in an almost 4 times smaller Koopman matrix, which is computationally less expensive to compute and store.

| Model | MAE $[m]$ | MAE $[\%]$ |
|---|---|---|
| eDMD (known dynamics) | 0.175 | 100 |
| LSTM-enhanced Deep Koopman | 0.185 | 106 |
| eDMD (unknown dynamics) | 0.606 | 346 |

Lastly, we compare the eigenvalues of the Koopman operator for the two tank systems using LSTM-enhanced Deep Koopman and eDMD. Figure 3 shows that LSTM-enhanced Deep Koopman provides more accurate and consistent eigenvalues with original eigenvalues compared to eDMD. This is an important contribution as the predictions could better align with reality and provide more accurate and consistent results.

## IV. CONCLUSION

This paper presents a novel dictionary-free method for identifying linear representations of nonlinear systems with input delays using deep neural networks. The LSTM-enhanced Deep Koopman model leverages the strengths of deep learning to generate and update nonlinear transformations, enabling the learning of high-fidelity Koopman operator models. By incorporating the history of the system, LSTM-enhanced Deep Koopman ensures precise modeling and improved long-term forecasting of nonlinear dynamics with input delays. The results demonstrate that LSTM-enhanced Deep Koopman outperforms traditional eDMD in cases where true dynamics are not anticipated in the dictionary. In this case, the LSTM-enhanced Deep Koopman approach provides improvement more than 3 times in MAE, showing the ability to capture the system's dynamics more effectively while delivering accurate and consistent predictions. With eDMD, where true dynamics are known, the LSTM-enhanced Deep Koopman underperforms by only 6% in MAE, which is a considerable trade considering that the true dynamics are not always known. Future work will focus on extending the LSTM-enhanced Deep Koopman model to more complex systems and exploring its applications in model predictive control.

## REFERENCES

[1] D. Wilson, "Koopman operator inspired nonlinear system identification," *SIAM Journal on Applied Dynamical Systems*, vol. 22, no. 2, pp. 1445–1471, 2023. [Online]. Available: https://doi.org/10.1137/22M1512272

[2] A. Mauroy and J. Goncalves, "Linear identification of nonlinear systems: A lifting technique based on the koopman operator," in *2016 IEEE 55th Conference on Decision and Control (CDC)*, Dec 2016, pp. 6500–6505.

[3] I. Mezić and A. Banaszuk, "Comparison of systems with complex behavior," *Physica D: Nonlinear Phenomena*, vol. 197, no. 1, pp. 101–133, 2004. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167278904002507
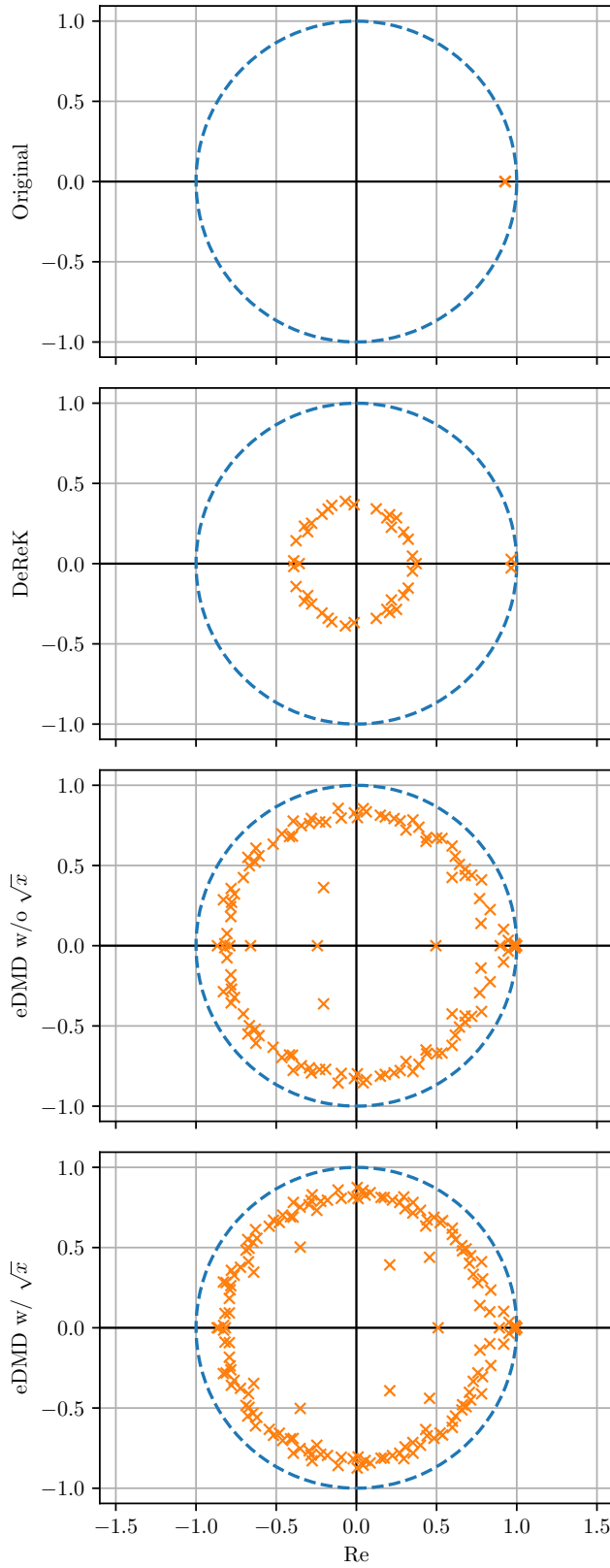


Fig. 2. Predicted and actual water levels in tanks 1 and 2 for the two tank system. Blue line shows the simulation of (9), while gray shows the noisy data, as used for training. The sequence shows a snippet of testing set which was unseen during system identification.

namics are not known. Meanwhile, LSTM-enhanced Deep Koopman provides comparable results to eDMD with square root, demonstrating its effectiveness in capturing the system's dynamics without prior knowledge of the true nonlinear terms, demonstrating the capabilities of the LSTM-enhanced Deep Koopman model in modeling nonlinear systems with input delays. The degradation in performance was, in this case, 70% in MAE as compared to eDMD with square root, nevertheless it makes only 0.04 m higher error in absolute water levels. It is important to note that the LSTM-enhanced Deep Koopman model is composed of a smaller number of lifted states

Fig. 3. Comparison of eigenvalues of the Koopman operator for the two tank system. Subplot "Original" shows the eigenvalues of the original system, lienarized around the tank levels of 1 m.

[4] I. Mezić, "Spectral properties of dynamical systems, model reduction and decompositions," *Nonlinear Dynamics*, vol. 41, no. 1, pp. 309–325, Aug 2005. [Online]. Available: https://doi.org/10.1007/s11071-005-2824-x

[5] P. J. Schmid, "Dynamic mode decomposition of numerical and experimental data," *Journal of fluid mechanics*, vol. 656, pp. 5–28, 2010.

[6] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data–driven approximation of the koopman operator: Extending dynamic mode decomposition," *Journal of Nonlinear Science*, vol. 25, pp. 1307–1346, 2015.

[7] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature Communications*, vol. 9, no. 1, p. 4950, Nov 2018. [Online]. Available: https://doi.org/10.1038/s41467-018-07210-0

[8] E. Yeung, S. Kundu, and N. Hodas, "Learning deep neural network representations for koopman operators of nonlinear dynamical systems," in *2019 American Control Conference (ACC)*. IEEE, 2019, pp. 4832–4839.

[9] R. B. Lima and P. R. Barros, "Identification of time-delay systems: a state-space realization approach," *IFAC-PapersOnLine*, vol. 48, no. 8, pp. 254–259, 2015, 9th IFAC Symposium on Advanced Control of Chemical Processes ADCHEM 2015. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896315010575

[10] L.-J. Li, T.-T. Dong, S. Zhang, X.-X. Zhang, and S.-P. Yang, "Time-delay identification in dynamic processes with disturbance via correlation analysis," *Control Engineering Practice*, vol. 62, pp. 92–101, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0967066117300655

[11] B. Lusch, J. N. Kutz, and S. L. Brunton, "Deep learning for universal linear embeddings of nonlinear dynamics," *Nature communications*, vol. 9, no. 1, p. 4950, 2018.

[12] J. Drgona, A. Tuor, J. Koch, M. Shapiro, and D. Vrabie, "NeuroMANCER: Neural Modules with Adaptive Nonlinear Constraints and Efficient Regularizations," 2023. [Online]. Available: https://github.com/pnnl/neuromancer