Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 55-30 (2022) 430–435

Convolutional Autoencoders and Clustering for Low-dimensional Parametrization of Incompressible Flows

Jan Heiland [*,**] **Yongho Kim** [*,**,⋆]

[*] *Max Planck Institute for Dynamics of Complex Technical Systems, Magdeburg, Germany*
(e-mail: {heiland, ykim}@mpi-magdeburg.mpg.de)
[**] *Department of Mathematics, Otto von Guericke University, Magdeburg, Germany*

Abstract: The design of controllers for general nonlinear PDE models is a difficult task because of the high dimensionality of the partially discretized equations. It has been observed that the embedding of nonlinear systems into the class of linear parameter varying systems (LPV) gives way to apply linear theory and methods from numerical linear algebra for controller design. The feasibility of the LPV approach hinges on the dimension of the inherent parametrization. In this work we propose and evaluate combinations of convolutional neural networks and clustering algorithms for very low-dimensional parametrizations of incompressible Navier-Stokes equations.

Keywords: convolutional autoencoders, clustering, linear parameter-varying (LPV) systems, model reduction, incompressible flows.

1. INTRODUCTION

Controller design for general nonlinear and large-scale systems of type

$$\dot{\mathbf{v}}(t) = \mathbf{f}(\mathbf{v}(t)) + Bu(t)$$

is a theoretical and computational challenge. A widely applicable approach is *model predictive control* (MPC) (Grüne and Pannek (2017)) that does not resort to physical insights like *backstepping* (Kokotovic (1992) or *sliding mode* (Dodds (2015)) control and, thus, is suitable for general purposes. Nonetheless, the need for repeated solves of possibly nonlinear optimization problems puts a natural limit in applicability for large-scale systems.

Linear methods, on the other side, have been sucessfully applied to really large-scale systems but, by design, only work as long as a linearization is a suitable model.

Several efforts have been made, to utilize linear methods and, thus, powerful computational backends for realizing nonlinear controllers. A possible way is the repeated linearization about the current state as in so-called *state-dependent coefficient* (SDC) formulations in which the flow is factorized as

$$f(\mathbf{v}) = A(\mathbf{v})\,\mathbf{v}$$

with a state-dependent matrix; see, e.g. Banks et al. (2007). Although, as shown in Benner and Heiland (2016), further exploitation of linear methods are possible even with performance guarantees, the needed computational efforts are still a limiting factor. If for the expression of the factorization $A(\mathbf{v})$ a possibly low-dimensional representation $\boldsymbol{\rho}$ of $\mathbf{v}$, i.e. $A(\mathbf{v}) = \tilde{A}(\boldsymbol{\rho}(\mathbf{v}))$ can be used and if A is linear in $\boldsymbol{\rho}$, then the nonlinear model is embedded in the class of (affine-linear) linear parameter varying (LPV) systems for which *gain scheduling* (see, e.g. Leith and Leithead (2000)) as an approach that uses linear methods for nonlinear controller design can be efficiently applied; see, e.g., Hashemi and Werner (2011); Koelewijn and Tóth (2020); Trudgen and Velni (2018).

As it involves the identification of salient parameter configurations (so-called *working points*) and the interpolation between them, *gain scheduling* is tractable only for low dimensional parameter spaces. Therefore, we propose (following, e.g., Hashemi and Werner (2011)) an approximation $A(\mathbf{v}) \approx \tilde{A}(\boldsymbol{\rho}(\mathbf{v}))$ with a particular emphasis on a low dimensional and affine-linear parametrization.

Since for flow equations, linear projection-based approximations like *Proper Orthogonal Decomposition* (POD) are likely outperformed by nonlinear approaches at very low parameter dimensions (cp., e.g., Lee and Carlberg (2020)), we investigate how convolutional neural networks (CAEs) can be used for very low-dimensional, possibly affine-linear, approximative parametrizations of solution trajectories of semi-discrete incompressible Navier-Stokes equations. In addition, anticipating the need for *working points* we investigate, how to combine CAEs with clustering algorithms.

2. PRELIMINARY CONSIDERATIONS

In this section, we discuss how any model order reduction scheme can be used for low-dimensional LPV approximations of incompressible Navier-Stokes equations, give reasoning why neural network based approaches may outperform standard approaches.

⋆ The author is supported by the German Research Foundation (DFG) research training group 2297 “MathCoRe”, Magdeburg.

10.1016/j.ifacol.2022.11.091

We consider the Navier-Stokes equations

$$\frac{\partial}{\partial t}v + (v\cdot\nabla)v - \frac{1}{Re}\Delta v + \nabla p = f \qquad (1)$$
$$\nabla\cdot v = 0$$

after semidiscretization and in divergence-free coordinates (see, e.g. Ahmad et al. (2017) for a derivation):

$$\dot{\mathbf{v}}(t) + N(\mathbf{v}(t))\mathbf{v}(t) + A\mathbf{v}(t) = \mathbf{f}(t), \qquad (2)$$

where $\dot{\mathbf{v}}(t), \mathbf{v}(t) \in \mathbb{R}^{n_v}$ denote the state of the velocity and its derivative at time t, where $\mathbf{f}(t) \in \mathbb{R}^{n_v}$ is an inhomogeneity, where the matrix $A \in \mathbb{R}^{n_v\times n_v}$ accounts for the diffusion part and where $N(\mathbf{v}(t)) \in \mathbb{R}^{n_v\times n_v}$ represents the discrete approximation of the convection induced by the current state of the velocity.

In what follows we will drop the time dependency.

Since $N(\mathbf{v})$ is a matrix, the system (2) is already in SDC form. In order to approximate it by an (affine) LPV form, the following considerations are relevant.

(1) Any model order reduction scheme that provides a compression to reduced coordinates via $\boldsymbol{\rho} = \mu(\mathbf{v})$ and a lifting via $\tilde{v} = \lambda(\boldsymbol{\rho})$, defines an LPV approximation via $\tilde{N}(\rho) = N(\lambda(\boldsymbol{\rho}))$.
(2) Since $N(\cdot)$ is linear, any linear lifting λ, i.e. $\tilde{v} = \sum_i \rho_i w_i$ for some basis $\{w_i\}$, defines an affine LPV approximation via

$$\tilde{N}(\boldsymbol{\rho}) = N(\sum_i \rho_i w_i) = \sum_i \rho_i N(w_i).$$

For example, in the method of *Proper orthogonal decomposition* (POD; Moore (1981)), a POD basis $V = [w_1 \cdots w_r]$ is extracted from samples of the velocity trajectory and used to define a compression to POD coordinates as $\boldsymbol{\rho} = V^T\mathbf{v}$ and a lifting as $\tilde{V}\boldsymbol{\rho}$. In fact, the POD basis is defined as minimizing the projection error

$$\mathbf{v} - \tilde{\mathbf{v}} = (I - VV^T)\mathbf{v}$$

over all bases of dimension r for a given set of samplings.

This paper is directed to the development of very low-dimensional encodings of the velocity variable $\mathbf{v}$ for being used as parametrization in LPV approximations of the Navier-Stokes equations in a convection dominated regime. It has been commonly observed and rigorously analyzed, e.g., in Ohlberger and Rave (2016), that linear projections need a minimum dimension to well encode convective phenomena. Nonlinear methods based on neural networks, however, have shown good results in representing such nonlinear relations with a low number of coordinates; cp., e.g., Lee and Carlberg (2020); Kim et al. (2020).

In this work, we will expand on these previous results in two ways.

Firstly, we note that the compression part $\boldsymbol{\rho} = \mu(\mathbf{v})$ may well be nonlinear since for affine linearity of the LPV for the Navier-Stokes equations, only the dependency of $\rho \mapsto N(\rho)$ matters. Accordingly, we will investigate convolutional autoencoders with truly linear decoding parts.

Secondly, in view of both better reconstruction and, perspectively, of controller design, we investigate how the low-dimensional encodings can be used for clustering of the flow regimes. In a first application, we will find that with decoders that are trained for the clusters individually, low reconstruction errors at very low dimensions can be achieved.

3. CONVOLUTIONAL AUTOENCODER FOR AFFINE LPV APPROXIMATIONS

We consider convolutional autoencoders (CAEs) (cp., e.g., Masci et al. (2011)) for encoding and decoding of the velocity variable v. In view of the design of affine LPV approximations to the Navier-Stokes equations (2) we analyze and explore how the decoding or reconstruction part can be realized with only linear operations.

Generally, we call a composition

$$\tilde{\mathbf{v}} = \mu^{-1}(\mu(\mathbf{v}))$$

consisting of an encoder

$$\mu : \mathbb{R}^{n_v} \to \mathbb{R}^{n_\rho}, \boldsymbol{\rho} = \mu(\mathbf{v}), n_\rho \ll n_v \qquad (3)$$

and a decoder

$$\mu^{-1} : \mathbb{R}^{n_\rho} \to \mathbb{R}^{n_v}, \tilde{\mathbf{v}} = \mu^{-1}(\boldsymbol{\rho}) \qquad (4)$$

an autoencoder. If the parts μ and μ^{-1} are given by convolutional neural networks, we call the autoencoder a convolutional autoencoder (CAE).

It is standard and convenient practice to denote the decoder with μ^{-1} although it is, in general, not the inverse mapping to μ, the more that μ as a compression is unlikely to be invertible.

The common task of μ and μ^{-1} is to encode a v in a low-dimensional variable ρ which can be decoded again in a reconstruction $\tilde{\mathbf{v}}$ that well approximates the initial $\mathbf{v}$:

$$\mathbf{v} \approx \tilde{\mathbf{v}} = \mu^{-1}(\mu(\mathbf{v})) = \mu^{-1}(\boldsymbol{\rho}(\mathbf{v})).$$

We will focus on μ^{-1} being an affine map, so that it can be expressed as

$$\mu^{-1}(\boldsymbol{\rho}) = \mu^{-1}(\rho_1\mathbf{e}_1 + \rho_2\mathbf{e}_2 + \cdots + \rho_{n_\rho}\mathbf{e}_{n_\rho}) \qquad (5)$$
$$= \mathbf{w}_0 + \sum_{i=1}^{n_\rho} \rho_i \mu^{-1}(\mathbf{e}_i) \qquad (6)$$

where $\boldsymbol{\rho} = [\rho_1, \rho_2, \cdots, \rho_{n_\rho}]^T$, $\{\mathbf{e}_1, \cdots, \mathbf{e}_{n_\rho}\}$ is a standard basis and $\mathbf{w}_0$ is the final nonzero bias obtained from the decoder. In this case, we can consider the vectors $\mathbf{w}_i = \mu^{-1}(\mathbf{e}_i)$, $i = 1, 2, \cdots, n_\rho$ as a generating system for the affine space

$$W = w_0 + \mathrm{span}\{\mathbf{w}_1, \cdots, \mathbf{w}_{n_\rho}\}$$

which contains the image of the decoder.

In what follows, we will explore how convolutional neural networks (CNNs) can be used for the design of such autoencoders. For a general introduction to CNNs, we refer to O'Shea and Nash (2015), for an application to dynamical equations with states that have two or three spatial dimensions to Lee and Carlberg (2020).

A general neural network is composed of an input layer, one or more hidden layers, and an output layer. Except of the input layer, each layer is defined by it's size and a map from the previous layer into the current layer. Typically, this map is defined by an affine linear map $Wx+b$ with a weight matrix W and a bias term b, possibly followed by a nonlinear *activation function* that acts on each component of the result of the linear map. The biggest

benefit of neural networks is that the architecture can be built without restriction compared to other machine learning methods. Thus, a variety of neural networks such as convolutional neural networks, autoencoders, recurrent neural networks, and generative adversarial networks have been developed successfully.

We will make particular use of the following observation that is formulated for so-called *transposed convolutional layers* but that holds for any composition of standard layers in a neural network.

Proposition 1. The composition of transposed convolutions without application of a (nonlinear) activation function is affine linear.

Proof. Transposed convolution can be expressed as an affine linear transformation $\mathbf{x}_O = T\mathbf{x}_I + \mathbf{b}$ where $\mathbf{b}$ is a bias vector, T is a matrix, input X_I and output X_O matrices are vectorized as $\mathbf{x}_O$ and $\mathbf{x}_I$ respectively. If no nonlinear activation function is applied in between, we can see that the overall transposed convolution is a concatenation of affine linear maps. Let $L_k(\mathbf{x}) = T_k\mathbf{x}+\mathbf{b}_k$, $k = 1, 2, \cdots, n$ be transposed convolutions where n refers to n-th convolutional layer. Thus, a deep decoder network μ^{-1} is defined as $\mu^{-1}(\mathbf{x}) = (L_n \circ L_{n-1} \circ \cdots \circ L_2 \circ L_1)(\mathbf{x})$.

$$
\begin{aligned}
(L_2 \circ L_1)(\mathbf{x}) &= T_2T_1\mathbf{x} + T_2\mathbf{b}_1 + \mathbf{b}_2\\
(L_3 \circ L_2 \circ L_1)(\mathbf{x}) &= T_3T_2T_1\mathbf{x} + T_3T_2\mathbf{b}_1 + T_3\mathbf{b}_2 + \mathbf{b}_3\\
&\vdots\\
(L_n \circ \cdots \circ L_1)(\mathbf{x}) &= T_n\cdots T_1\mathbf{x} + T_n\cdots T_2\mathbf{b}_1\\
&\quad + T_n\cdots T_3\mathbf{b}_2 + \cdots + T_n\mathbf{b}_{n-1} + \mathbf{b}_n\\
&= T\mathbf{x} + \mathbf{b}
\end{aligned}
$$

where $T = T_n\cdots T_1$ and $\mathbf{b} = T_n\cdots T_2\mathbf{b}_1 + T_n\cdots T_3\mathbf{b}_2 + \cdots + T_n\mathbf{b}_{n-1} + \mathbf{b}_n$. Therefore, the composition of affine transformations is an affine transformation.

Remark 2. The resulting decoder μ^{-1} can be represented by a single affine linear map and, thus, a single linear layer. However, convolutional layers are completely defined through their convolution kernels via a limited number of parameters and independent of the layer width. Accordingly,

(1) in order to keep the local structures of convolutions and their small number of weights despite possibly large layer widths and
(2) since this approach is inline with pregiven structures used in machine learning toolboxes,

we realize μ^{-1} as a sequence of the individual linear maps. In view of performance at runtime, however, the inference of the matrix that represents μ^{-1} should be considered.

Now by linearity of $N(\cdot)$, for an affine linear μ^{-1}, we can derive

$$
\begin{aligned}
N(\mathbf{v}) &\approx N(\tilde{\mathbf{v}})\\
&= N(\mu^{-1}(\boldsymbol{\rho})) = N(\mathbf{w}_0 + \sum_{i=1}^{n_\rho}\rho_i\mathbf{w}_i)\\
&= N(\mathbf{w}_0) + \sum_{i=1}^{n_\rho}\rho_i N(\mathbf{w}_i)
\end{aligned}
$$

and define the low-dimensional LPV approximation to (2) as

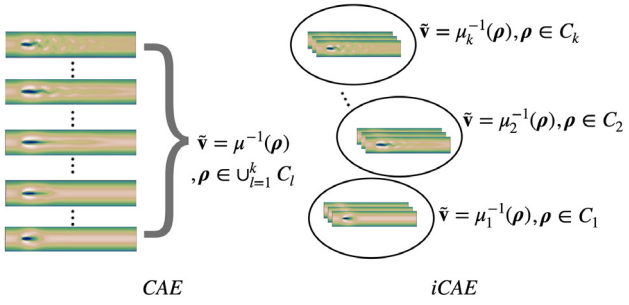

Fig. 1. Basic idea of individual CAEs (iCAEs)

$$
\dot{\mathbf{v}} + (N(\mathbf{w}_0) + \sum_{i=1}^{n_\rho}\rho_i N(\mathbf{w}_i))\mathbf{v} + A\mathbf{v} = \mathbf{f}. \tag{7}
$$

4. CONVOLUTIONAL AUTOENCODERS AND CLUSTERING

Once a low-dimensional representation of the dynamics is obtained, it can be used for clustering of the continuous data into a number of representative regimes. This can of use for designing nonlinear controllers based on a limited number of likely regimes and, as we will show, for improved reconstruction methods.

We investigate the use for k-means clustering of the reduced parameters $\boldsymbol{\rho}$ to classify flow regimes in a low dimensional space. Moreover, assuming that a number of k clusters is identified, it is possible to efficiently compute a further approximation to the LPV realization (7) as follows: Denote the centroid of each cluster as $\mathbf{c}^{(l)} \in \mathbb{R}^{n_\rho}$, $l = 1, 2, \cdots, k$. Then k possible states

$$
N(\tilde{\mathbf{v}}) \approx N(\mu^{-1}(\mathbf{c}^{(l)})) \tag{8}
$$

of the coefficient N can be precomputed and chosen at run time according to the cluster that the current velocity $\mathbf{v}$ is assigned to. Certainly, since $\mathbf{c}^{(l)}$ is but the mean of the variables $\boldsymbol{\rho}$ in the l-th cluster C_l this approximation introduces an additional reconstruction error. Nevertheless, we will see that this approach still outperforms POD in very low-dimensional spaces in Section 7.

5. INDIVIDUAL CONVOLUTIONAL AUTOENCODERS

To avoid the performance degradation caused by the usage of centroids (8), the centroids are not applied directly to the system. Instead, we propose individual CAEs (iCAEs) that reconstruct state vectors using a decoder in each cluster.

$$
N^{(l)}(\tilde{\mathbf{v}}) = N(\mathbf{w}_0^{(l)}) + \sum_{i=1}^{n_\rho}\rho_i N(\mathbf{w}_i^{(l)})
$$

where $\boldsymbol{\rho} = [\rho_1, \rho_2, \cdots, \rho_{n_\rho}]^T \in C_l$, $\mathbf{w}_i^{(l)} = \mu_l^{-1}(\boldsymbol{e}_i)$, $i = 1, 2, \cdots, n_\rho$, and $\mathbf{w}_0^{(l)}$ is the final bias of the decoder μ_l^{-1} of the l-th cluster. While the encoder μ trained by the whole set of training snapshots, the individual decoders are trained solely on the data-points that have been assigned to the corresponding clusters C_l. Thus, the encoding $\rho(\mathbf{v})$ happens in the same way for all encountered velocity

states, whereas the reconstruction $\tilde{\mathbf{v}} = \mu^{-1}(\boldsymbol{\rho})$ is given by the individual decoder $\mu^{(\ell)}$ depending on the relevant cluster.

6. OPTIMIZATION

We briefly explain the basic approaches and chosen optimization targets for training the autoencoders and for identifying the clusters.

6.1 Autoencoder

For the training of the autoencoders, we used the target function

$$\sum_{i=1}^{T} \|\mathbf{v}_i - \tilde{\mathbf{v}}_i\|_2^2. \tag{9}$$

Thus, the optimization (9) designs the encoder μ and decoder μ^{-1} such that the reconstruction $\tilde{\mathbf{v}} = \mu^{-1}(\mu(\mathbf{v}))$ minimizes the summation of the l_2 distances between $\mathbf{v}_i$ and $\tilde{\mathbf{v}}_i$ within a batch T instances of training data.

Once the clusters are detected, the training data can be classified and, with the parameters of the encoder frozen at their current values, the individual iCAEs can be trained.

6.2 K-means clustering

Given an encoder μ, for the detection of clusters, a number k is fixed and the criterion

$$\sum_{i=1}^{T} \| \mathbf{c} - \boldsymbol{\rho}_i \|_2^2$$

minimized over all possible centers $\mathbf{c} \in \{\mathbf{c}^{(1)}, \cdots, \mathbf{c}^{(k)}\}$ to become nearest centroids to $\boldsymbol{\rho}_i = \mu(\mathbf{f}_i)$ in terms of the Euclidean distance function. The centroids and autoencoder parameters can be optimizated simultaneously by using a joint loss function $L = L_c + L_r$ where L_c is a clustering loss and L_r is a reconstruction loss. However, the clustering optimization disturbs the reconstruction optimization so the optimization strategy is important to obtain general model parameters. In this paper, the k-mean clustering approach by Sculley (2010) is optimized separately after pre-training the CAE.

7. RESULTS

To train the CAEs, 400 snapshots of x and y directional velocities are used. Also, 800 snapshots are used for the evaluation. All the data $\mathbf{v}$ are in $[0, 10]$ and were generated by a finite element simulation on a nonuniform triangulation of a Navier-Stokes model for a cylinder wake with a Reynolds number of 40.

To function as an input for the convolutional neural networks, the velocity snapshot where interpolated to a rectangular mesh of size $n_v = 5922$(calculated $2 \times 47 \times 63$, i.e. the two spatial dimensions of the velocity times the nodes in both dimensions of the domain $(0, 5) \times (0, 1)$)). The size of the training and the evaluation data sets are, thus, $400 \times 2 \times 47 \times 63$ and $800 \times 2 \times 47 \times 63$ respectively. See Figure 3 for an illustration of the computational setup and a snapshot of the fully developed flow in full resolution.

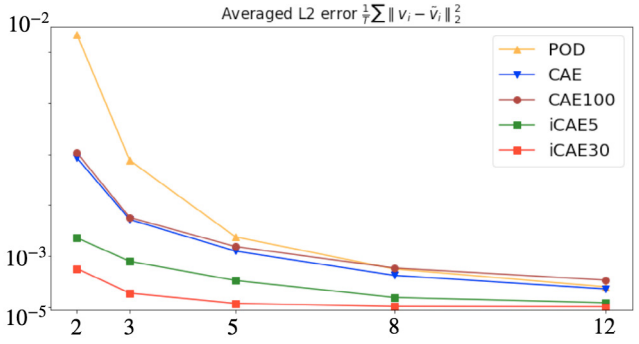

Fig. 2. Reconstruction errors depending on reduced parameter dimensions (2, 3, 5, 8, 12): (brown) CAE using 100 centroids, (orange) POD, (blue) CAE, (green) iCAE based on 5 clusters (red) iCAE based on 30 clusters

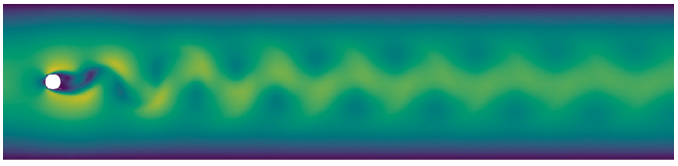

Fig. 3. Illustration of the computational domain and a magnitude plot of the fully developed flow.

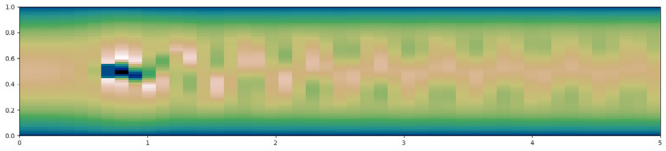

Fig. 4. The data interpolated to the 47×63 regular grid as input for the CNNs.

The resolution used for training the neural network is shown in Figure 4.

For the encoder, we use five convolutional layers without bias and the ELU activation function Clevert et al. (2016) is used in each layer. The decoder consists of five transposed convolutional layers without any activation functions.

For the optimization, as the batch size, we chose $T = 50$. The Adam optimizer is used with a learning rate of 0.001.

Figure 2 shows that the CAE outperforms POD and the clustering approach CAE100 causes performance degradation as we expected but it is still better than POD. The iCAEs using individual decoders depending on clusters outperform the other methods by one order of magnitude in the relevant very low-dimensional setup.

However, despite the individual decoders being affine linear functions, the association of ρ with the cluster ℓ is a nonlinear operation so that the resulting LPV approximation is no more affine linear.

In Figure 6 we report the reconstruction performance of the different approaches for code size $n_\rho = 2$ at a few selected points in space over the full simulation horizon. As it can be expected from the averaged errors, the CAE

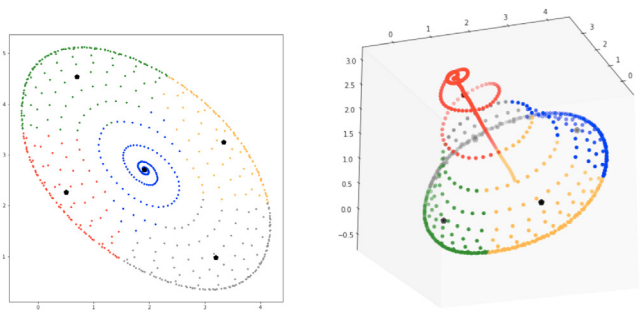

Fig. 5. Latent variables $\boldsymbol{\rho} = \mu(\mathbf{v})$ in low dimensional spaces: (left) $\mu : \mathbb{R}^{5922} \to \mathbb{R}^2$, (right) $\mu : \mathbb{R}^{5922} \to \mathbb{R}^3$. Each star is the centroid of individual five clusters.

approach outperforms the POD whereas the nonlinear iCAE approximation follows the base-line surprisingly well.

Finally, we show the phase portraits of the reduced variables $\boldsymbol{\rho}$ for $n_\rho = 2, 3$ and the location of the centers in the 2D and 3D space, respectively in Figure 5.

8. CODE AVAILABILITY

The source code of the implementations used to compute the presented results is available from

`https://doi.org/10.5281/zenodo.6817442`

under the Creative Commons Attribution 4.0 international license and is authored by Jan Heiland and Yongho Kim.

REFERENCES

Ahmad, M. I., Benner, P., Goyal, P., Heiland, J., 2017. Moment-matching based model reduction for Navier-Stokes type quadratic-bilinear descriptor systems. Z. Angew. Math. Mech. 97 (10), 1252–1267.

Banks, H., Lewis, B., Tran, H., 2007. Nonlinear feedback controllers and compensators: a state-dependent Riccati equation approach 37 (2), 177–218.

Benner, P., Heiland, J., 2016. Robust stabilization of laminar flows in varying flow regimes. IFAC-PapersOnLine 49 (8), 31–36, 2nd IFAC Workshop on Control of Systems Governed by Partial Differential Equations CPDE 2016, Bertinoro, Italy, 13–15 June 2016.

Clevert, D., Unterthiner, T., Hochreiter, S., 2016. Fast and accurate deep network learning by exponential linear units (elus). In: Bengio, Y., LeCun, Y. (Eds.), 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings.
URL `http://arxiv.org/abs/1511.07289`

Dodds, S. J., 2015. Sliding Mode Control and Its Relatives. Springer London, London, pp. 705–792.

Grüne, L., Pannek, J., 2017. Nonlinear model predictive control. Theory and algorithms. Cham: Springer.

Hashemi, S. M., Werner, H., 2011. LPV modelling and control of Burgers' equation. IFAC Proceedings Volumes 44 (1), 5430–5435, 18th IFAC World Congress.

Kim, Y., Choi, Y., Widemann, D., Zohdi, T., 2020. Efficient nonlinear manifold reduced order model. CoRR abs/2011.07727.

Koelewijn, P. J. W., Tóth, R., 2020. Scheduling dimension reduction of LPV models – a deep Neural Network approach.

Kokotovic, P., 1992. The joy of feedback: nonlinear and adaptive. IEEE Control Systems Magazine 12 (3), 7–17.

Lee, K., Carlberg, K. T., 2020. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. J. Comput. Phys. 404.

Leith, D. J., Leithead, W. E., 2000. Survey of gain-scheduling analysis and design. Int. J. Control 73 (11), 1001–1025.

Masci, J., Meier, U., Ciresan, D. C., Schmidhuber, J., 2011. Stacked convolutional auto-encoders for hierarchical feature extraction. In: Honkela, T., Duch, W., Girolami, M. A., Kaski, S. (Eds.), Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I. Vol. 6791 of Lecture Notes in Computer Science. Springer, pp. 52–59.

Moore, B. C., 1981. Principal component analysis in linear systems: controllability, observability, and model reduction. IEEE Trans. Autom. Control AC–26 (1), 17–32.

Ohlberger, M., Rave, S., 2016. Reduced basis methods: Success, limitations and future challenges. Proceedings of the Conference Algoritmy, 1–12.

O'Shea, K., Nash, R., 2015. An introduction to convolutional neural networks. CoRR abs/1511.08458.
URL `http://arxiv.org/abs/1511.08458`

Sculley, D., 2010. Web-scale k-means clustering. In: Proceedings of the 19th International Conference on World Wide Web. WWW '10. Association for Computing Machinery, New York, NY, USA, p. 1177–1178.

Trudgen, M., Velni, J. M., 2018. Linear parameter-varying approach for modeling and control of rapid thermal processes. International Journal of Control, Automation and Systems 16 (1), 207–216.

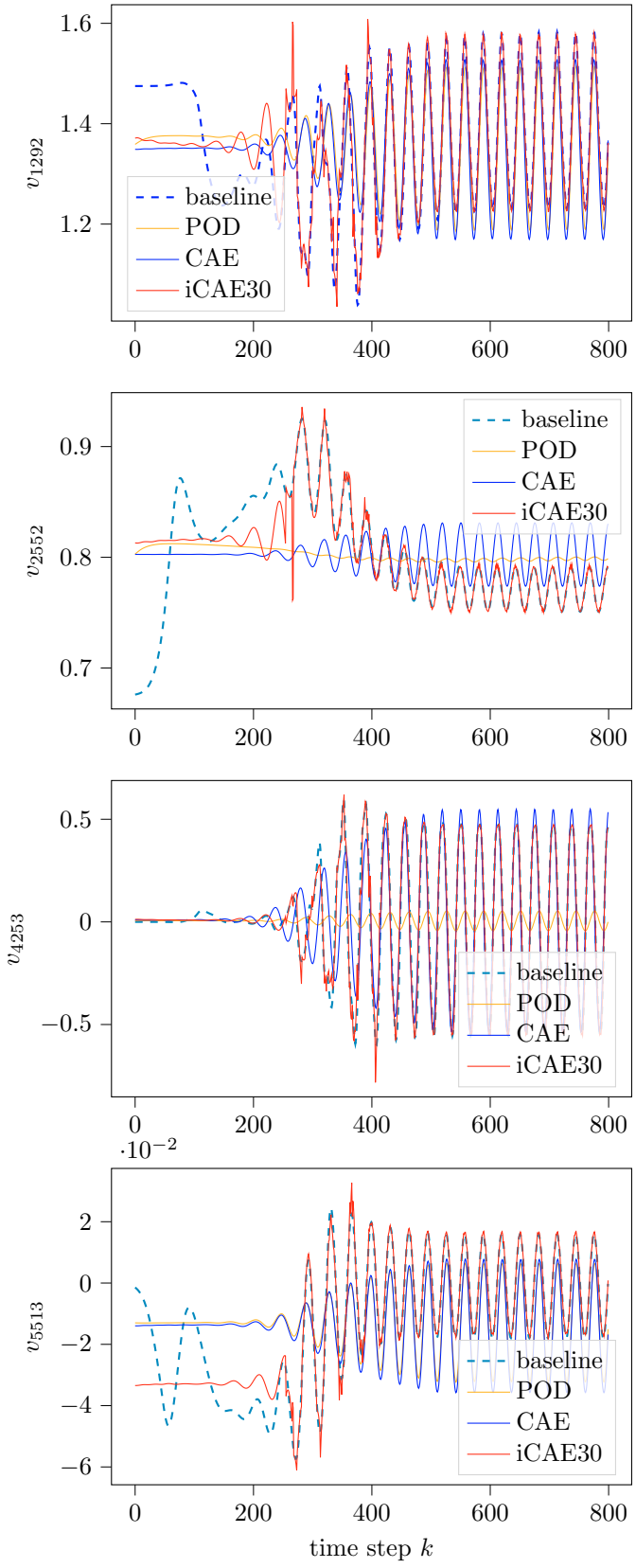

Fig. 6. Comparison between the baseline and reconstructed parameters where $n_\rho = 2$, $t = 1, \cdots, 800$, $i = 1299, 2552, 4253, 5513$: (dashed) baseline $\mathbf{v}_{t,i}$ (orange) POD-based, (blue) CAE-based, (red) iCAE30-based reconstruction $\tilde{\mathbf{v}}_{t,i}$