Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 59-19 (2025) 550–555

Leveraging time and parameters for nonlinear model reduction methods ⋆

Silke Glas * **Benjamin Unger** **

* *Department of Applied Mathematics, University of Twente, 7500 AE Enschede, The Netherlands (e-mail: s.m.glas@utwente.nl).*
** *Institute for Applied and Numerical Mathematics, Karlsruhe Institute of Technology, 76131 Karlsruhe, Germany (e-mail: benjamin.unger@kit.edu)*

Abstract: In this paper, we consider model order reduction (MOR) methods for problems with slowly decaying Kolmogorov n-widths as, e.g., certain wave-like or transport-dominated problems. To overcome this Kolmogorov barrier within MOR, nonlinear projections are used, which are often realized numerically using autoencoders. These autoencoders generally consist of a nonlinear encoder and a nonlinear decoder and involve costly training of the hyperparameters to obtain a good approximation quality of the reduced system. To facilitate the training process, we show that extending the to-be-reduced system and its corresponding training data makes it possible to replace the nonlinear encoder with a linear encoder without sacrificing accuracy, thus roughly halving the number of hyperparameters to be trained.

Keywords: Model reduction, nonlinear projections, autoencoder, hyperparameter optimization.

1. INTRODUCTION

In high-fidelity simulation-driven design processes, large-scale differential or differential-algebraic equations must be evaluated for many different sets of parameters, amounting to a significant demand on computing resources. Replacing the high-fidelity simulation with a cheap-to-evaluate surrogate model is expected to reduce the computing time and speed up the design cycle. Due to its efficient and numerically stable construction, generalization capabilities, and error certificates, *model order reduction* (MOR) is a typical approach for assembling the surrogate. Over the last decades, MOR was successfully utilized in various application domains – we refer to Antoulas (2005); Benner et al. (2017) and the references therein for examples.

In this paper, we consider projection-based MOR methods, which construct the ROM by projecting the differential equation onto a low-dimensional subspace using a Petrov–Galerkin framework. Although this approach is successful in many applications, transport-dominated problems with a strong coupling of the spatial and temporal domain typically require a rather large subspace to achieve good approximation quality. The theoretical limit is given by the Kolmogorov n-width (Kolmogoroff, 1936), respectively the Hankel singular values for control systems (see Unger and Gugercin (2019)). Different nonlinear approaches have been proposed in the literature to overcome this Kolmogorov barrier. Prominent examples are the shifted proper orthogonal decomposition (Reiss et al., 2018; Black et al., 2020, 2022), the registration method (Ferrero et al., 2022), and methods based on (shallow) autoencoders (Lee and Carlberg, 2020; Kim et al., 2022; Buchfink et al., 2023). We refer to Buchfink et al. (2024b) for a more detailed literature review and a unifying framework for nonlinear projections (consisting of an encoder and a decoder when using autoencoders).

Most data-driven nonlinear approaches have in common that constructing the involved nonlinear projection entails solving complex and large-scale optimization problems. These are extremely expensive and require extensive fine-tuning to converge to a (good) local minimum. In addition, the theory-to-practice gap in deep learning (Grohs and Voigtlaender, 2024) and the stability of the numerical method (Cohen et al., 2022) prevent practical algorithms based on sampled data from achieving the theoretical optimum. Moreover, even if evaluated fast, these nonlinear approaches also require considerable computing power once they are deployed, see e.g., Desislavov et al. (2023).

Our main contribution is to simplify the structure of the nonlinear approach in the sense that the nonlinear projection can be computed with significantly fewer optimization parameters. Commonly, autoencoders consist of a nonlinear encoder and a nonlinear decoder. To achieve good accuracy, they involve a large number of hyperparameters. However, we demonstrate that extending the to-be-reduced system and its corresponding training data makes it possible to replace the nonlinear encoder with a linear encoder without sacrificing accuracy, thus roughly halving the number of hyperparameters to be trained.

The paper is structured as follows. We briefly introduce the setting and known work in Section 2. Subsequently, we present the main idea of this paper, i.e., the extension of

⋆ The research of BU was mainly conducted while he was affiliated with the University of Stuttgart. BU is funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 - 390740016 and by the BMBF (grant no. 05M22VSA). Further, BU acknowledges the support by the Stuttgart Center for Simulation Science (SimTech).

10.1016/j.ifacol.2025.11.092

the training data, and some theoretical results in Section 3. In Section 4, we consider the numerical investigation of two proof-of-concept examples, the Burgers' equation and the advection equation. We conclude in Section 5.

2. SETTING AND KNOWN WORK

Consider the initial value problem (IVP)

$$\begin{cases} \dot{x}(t;\mu) = f(t, x(t;\mu);\mu), \\ x(t_0;\mu) = x_0(\mu), \end{cases} \tag{1}$$

with state $x(t;\mu) \in \mathbb{R}^n$, set of parameters $\mu \in \mathcal{P} \subseteq \mathbb{R}^p$, and initial value $x_0(\mu) \in \mathbb{R}^n$. Our assumption is that n is large and that the IVP (1) must be evaluated for many different $\mu \in \mathcal{P}$. The goal is thus to replace (1) with a *reduced-order model* (ROM) of the form

$$\begin{cases} \dot{\widehat{x}}(t;\mu) = \widehat{f}(t, \widehat{x}(t;\mu);\mu), \\ \widehat{x}(t_0;\mu) = \widehat{x}_0(\mu), \end{cases} \tag{2}$$

with reduced state $\widehat{x}(t;\mu) \in \mathbb{R}^r$, $r \ll n$ as an efficient surrogate of the parameter-to-solution map. In standard linear-subspace MOR, the ROM (2) is constructed via a Petrov–Galerkin projection, i.e., one determines bi-orthogonal matrices $V, W \in \mathbb{R}^{n\times r}$ and constructs

$$\widehat{f}(t, \widehat{x};\mu) := W^{\mathsf{T}} f(t, V\widehat{x};\mu).$$

When using linear-subspace MOR methods, the best possible error that the ROM can achieve is lower bounded by the Kolmogorov n-width. To overcome this Kolmogorov barrier, the linear mappings encoded by the matrices V and W^{T} are replaced by smooth nonlinear mappings

$$\varphi\colon \mathbb{R}^r \to \mathbb{R}^n, \qquad \rho\colon \mathbb{R}^n \to \mathbb{R}^r,$$

which we refer to as the decoder and encoder, respectively. In the context of MOR, these mappings have to satisfy the *point projection property*, cf. Buchfink et al. (2024b), i.e.,

$$\rho \circ \varphi = \mathrm{Id}_{\mathbb{R}^r}, \tag{3}$$

where $\mathrm{Id}_{\mathbb{R}^r}$ denotes the identity mapping in $\mathbb{R}^r$. Allowing for this kind of nonlinear approximations, then, under certain smoothness assumptions, we have the following result taken from (Buchfink et al., 2024b, Cor. 3.6), where also the exact setting and the proof can be found.

Theorem 1. If the (time×parameter)-to-solution map is sufficiently smooth, then there exists a ROM of dimension $p+1$ that yields an error-free surrogate for (1).

For practical purposes, the mappings ρ and φ are obtained from snapshots, i.e., evaluations of (1) for selected parameter and time values, denoted by x_i for $i = 1, \dots, M$.

3. CONTRIBUTION

The key observation for the proof of Theorem 1 is, that the (time×parameter)-to-solution map can serve as the decoder function, and the coordinates of the ROM are simply the time-variable and the parameters. Nevertheless, in almost all linear-subspace and nonlinear MOR methods, this information is not given during the training. Our key idea is thus to include the time (and the parameter) information in the training data. By doing so, we provide the autoencoder with additional information, which allows the decoder to (approximately) learn the (time×parameter)-to-solution map from the given data. Subsequently, this means that the encoder can be realized by a linear mapping only, thus facilitating the training complexity. In more detail, we propose to perform an autonomization step (even if f has no explicit time dependency), i.e., we consider the *extended system* with respect to time and parameter

$$\begin{cases} \dot{z}(t;\mu) = f_{\mathrm{ext}}(z(t;\mu);\mu), \\ z(t_0;\mu) = z_0(\mu), \end{cases} \tag{4}$$

with extended state and vector field

$$z(t;\mu) := \begin{bmatrix} t \\ x(t;\mu) \\ \mu \end{bmatrix}, \quad f_{\mathrm{ext}}(z(t;\mu);\mu) := \begin{bmatrix} 1 \\ f(t, x(t;\mu);\mu) \\ 0 \end{bmatrix}.$$

The advantage of the formulation (4) over (1) is twofold:

1. Using data from (4) explicitly encodes the time variable and the parameters, which are now explicitly available in the MOR process.
2. There is no need for a nonlinear encoder function ρ, since the time and the parameter can be extracted explicitly without the need for a nonlinear mapping.

We denote the snapshots for (4) with $\{z_i\}_{i=1}^M$. As a consequence of Theorem 1, we obtain the following result.

Theorem 2. If the (time×parameter)-to-solution map is sufficiently smooth and $r = p+1$, then

$$\begin{aligned} \min_{\substack{\rho\colon \mathbb{R}^n\to\mathbb{R}^r \\ \varphi\colon \mathbb{R}^r\to\mathbb{R}^n}} \sum_{i=1}^M \|x_i - \varphi(\rho(x_i))\|_2^2 \\ = \min_{\substack{W_{\mathrm{ext}}\in\mathbb{R}^{(1+n+p)\times r} \\ \varphi_{\mathrm{ext}}\colon \mathbb{R}^r\to\mathbb{R}^{(1+n+p)}}} \sum_{i=1}^M \|z_i - \varphi_{\mathrm{ext}}(W_{\mathrm{ext}}^{\mathsf{T}} z_i)\|_2^2, \end{aligned}$$

with

$$W_{\mathrm{ext}} = \left[W_{\mathrm{t}}^{\mathsf{T}}, W_{\mathrm{x}}^{\mathsf{T}}, W_{\mathrm{p}}^{\mathsf{T}}\right]^{\mathsf{T}} \in \mathbb{R}^{(1+n+p)\times r}.$$

Proof. We start by noticing, that the first term equals zero if we choose φ as the (time×parameter)-to-solution map and $\rho = \varphi^{-1}$, due to Theorem 1. To show the equality to the extended system, we particularly choose

$$W_{\mathrm{t}} = [1, 0, \dots, 0] \in \mathbb{R}^{1\times r}, \qquad W_{\mathrm{x}} = 0 \in \mathbb{R}^{n\times r},$$
$$W_{\mathrm{p}} = [0_{p\times 1}, \mathrm{Id}_{\mathbb{R}^p}] \in \mathbb{R}^{p\times r},$$

which leads to the coordinates of the ROM being the time-variable and the parameters. Now setting

$$\varphi_{\mathrm{ext}}\colon \mathbb{R}^r \to \mathbb{R}^{(1+n+p)}, \quad [t,\mu]^{\mathsf{T}} \mapsto [1, \varphi(t,\mu), 1_{1\times p}]^{\mathsf{T}},$$

yields the desired result.

Remark 3. The use of an affine encoder as we propose it here is a common choice in MOR based on quadratically or polynomially embedded manifolds, exemplified by Geelen et al. (2023); Barnett and Farhat (2022). Nevertheless, in these cases, the encoder is applied on the original state x and not on the extended state z. Moreover, these approaches still suffer from (a modified) Kolmogorov barrier, which is due to the design choice of the decoder; see Buchfink et al. (2024a) for details.

Before we provide numerical results, we emphasize that it is solely possible to replace the encoder with a linear mapping, but not the decoder. If we happen to choose the decoder as a linear mapping we obtain the following result.

Theorem 4. Consider snapshots $S = \{x_i\}_{i=1}^M \in \mathbb{R}^{n\times M}$ and let σ_i denote the i-th singular value of S. Then

$$\min_{\substack{V\in\mathbb{R}^{n\times r} \\ \rho\colon \mathbb{R}^n\to\mathbb{R}^r}} \sum_{i=1}^M \|x_i - V\rho(x_i)\|_2^2 = \sum_{j=r+1}^{\min(M,n)} \sigma_j^2. \tag{5}$$

Particularly, an optimal $V \in \mathbb{R}^{n\times r}$ is given by the leading r left singular vectors. Choosing a linear or nonlinear encoder doesn't affect the approximation error in this setting.

Proof. Without loss of generality, we can assume that the columns of V are orthonormal. Then, the best approximation of each snapshot is given by the orthogonal projection onto the columns of V, i.e., for given orthogonal $V \in \mathbb{R}^{n\times r}$,

$$\min_{\rho\colon \mathbb{R}^n\to\mathbb{R}^r} \sum_{i=1}^{M} \|x_i - V\rho(x_i)\|_2^2 = \sum_{i=1}^{M} \|x_i - VV^{\mathsf{T}}x_i\|_2^2.$$

The claim follows from the Schmidt–Eckart–Young–Mirsky theorem, see, e.g., (Antoulas, 2005, Thm. 3.6).

We emphasize that in contrast to Theorem 1, we do not distinguish between state x and extended state z in Theorem 4. If we consider z in Theorem 4, we get the best approximation error with respect to $\|\cdot\|_{\mathrm{F}}$ by

$$\min_{\rho_{\mathrm{ext}}} \sum_{i=1}^{M} \|z_i - V_{\mathrm{ext}}\rho_{\mathrm{ext}}(z_i)\|_2^2 = \sum_{i=1}^{M} \|z_i - V_{\mathrm{ext}}V_{\mathrm{ext}}^{\mathsf{T}}z_i\|_2^2,$$

with $V_{\mathrm{ext}} \in \mathbb{R}^{(1+n+p)\times r}$ and where we optimize over all $\rho_{\mathrm{ext}}\colon \mathbb{R}^{n+1} \to \mathbb{R}^r$. Naturally, the resulting partial error in the state x in the latter equation can be (at best) as good as in (5), where we specifically chose the best approximation with respect to x. This means, that we cannot expect any improvement in the state approximation accuracy by using the extended variable z only. Indeed, the following numerical example demonstrates this aspect.

Example 5. Consider $t_1 = 1$, $t_2 = 2$, and snapshots

$$x_1 = \begin{bmatrix}1\\0\end{bmatrix}, \quad x_2 = \begin{bmatrix}0\\0.9\end{bmatrix}, \quad z_1 = \begin{bmatrix}1\\1\\0\end{bmatrix}, \quad z_2 = \begin{bmatrix}2\\0\\0.9\end{bmatrix}.$$

We obtain that for $r = 1$ the best approximation yields an error of 0.9 in the Frobenius norm. Whereas, if we use the optimal approximation for the extended variable z, then the associated approximation of the state x yields an error of approximately 0.98 with respect to $\|\cdot\|_{\mathrm{F}}$. This can also easily be seen via the interlacing property of the singular values (Golub and Van Loan, 2013, Cor. 8.6.3), as adding a row only enlarges the smallest singular value of a matrix. Moreover, if we split the matrix V_{ext} in the following way

$$V_{\mathrm{ext}} = \begin{bmatrix}V_{\mathrm{t}}\\V_{\mathrm{x}}\\V_{\mathrm{p}}\end{bmatrix} \in \mathbb{R}^{(1+n+p)\times r},$$

and consider the projection error with respect to the state

$$\sum_{i=1}^{M} \|x_i - V_{\mathrm{x}}V_{\mathrm{x}}^{\mathsf{T}}(x_i)\|_2^2,$$

this value is of course also lower bounded by (5). However, this error also can be significantly larger, which is the case here, where the error measured in $\|\cdot\|_{\mathrm{F}}$ is 1.26.

4. NUMERICAL EXAMPLES

To demonstrate our theoretical findings, we consider two proof-of-concept examples, i.e., the Burgers' (Section 4.2) and the advection equation (Section 4.3). For both settings, we compare different autoencoder architectures, for which we describe the setup and the training process in Section 4.1.

4.1 Autoencoder Training

We generate training data by sampling both examples, i.e., the Burgers' equation (Section 4.2) and the advection equation (Section 4.3). We pick a one-dimensional spatial domain Ω for both settings, which we discretize into $n = 2^9 = 512$ equidistant points. For the time domain, we choose 300 equidistant points and the generated simulation data is split into training, validation, and testing data by distributing the time points into t_{3i}, t_{3i+1}, and t_{3i+2} for $i = 1, \dots, M = 100$ such that we have training data $\Theta \in \mathbb{R}^{n\times M}$. We fix the parameter for both examples and only use time for the extended data. Consequently, the extended training data Θ_{ext} has dimension $(1+n)\times M$. With these data sets, we train five kinds of autoencoders:

(I) a *nonlinear–nonlinear autoencoder* (NNA) with nonlinear encoder and decoder trained on Θ,
(II) a *linear-nonlinear autoencoder* ($\mathsf{LNA}_{\mathrm{ext}}$) with linear encoder and nonlinear decoder trained on the time extended data Θ_{ext},
(III) a *linear-nonlinear autoencoder* (LNA) with linear encoder and nonlinear decoder trained on Θ,
(IV) a *linear-nonlinear autoencoder* ($\mathsf{LNA}_{\mathrm{ext,fix}}$) with fixed linear encoder given by the first unit vector and nonlinear decoder trained on Θ_{ext}, and
(V) a *nonlinear-linear autoencoder* (NLA) with nonlinear encoder and linear decoder trained on data Θ.

For the implementation, we use `pytorch` (Paszke et al., 2019) and `Adam` (Kingma, 2014) with default settings as optimization algorithm. The target function is the least-squares error and a batch size of 20 is set in all trainings. The optimization is terminated if there is no further improvement on the validation data after 100 epochs (with a maximum of 20000 epochs) to avoid overfitting. To reduce the impact of randomness in the model initialization and the training, we initialize and optimize each model 100 times and choose the best run with respect to the lowest target function value on the validation error.

To ensure comparability, we use the same architecture for all autoencoders: As an activation function, the Leaky ReLU is utilized. For the layers of the decoder, we consider the following three scenarios:

(A) 7 layers with dimensions 1, 3, 9, 27, 81, 243, N,
(B) 6 layers with dimensions 1, 4, 16, 64, 256, N,
(C) 5 layers with dimensions 1, 5, 25, 125, N,

where N is either given by the spatial dimension, i.e., $N = n$ (for NNA, LNA, NLA), or for the extended data by $N = 1 + n$ (for $\mathsf{LNA}_{\mathrm{ext}}$, $\mathsf{LNA}_{\mathrm{ext,fix}}$). If the encoder is nonlinear, we use the mirrored setting of the decoder.

We report the minimum and the average error with respect to the testing data, with the error measure being the average over time of the relative $\|\cdot\|_2$-norm in space, i.e.,

$$\frac{1}{M}\sum_{i=1}^{M} \frac{\|x_i - x_{i,\mathrm{approx}}\|_2}{\|x_i\|_2}. \tag{6}$$

4.2 Burgers' equation

As the first example, we consider the one-dimensional viscous Burgers' equation

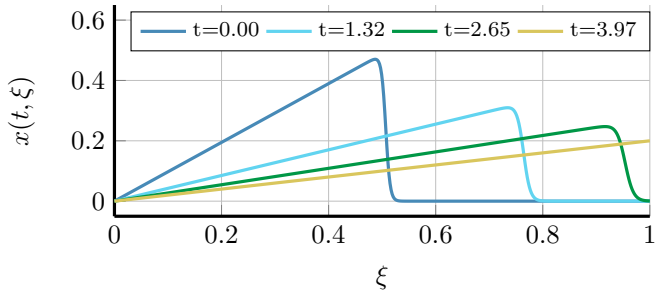

Fig. 1. Burgers' eq.: solution at selected times.

$$\partial_t x(t,\xi) = \tfrac{1}{\mathrm{Re}}\partial_{\xi^2} x(t,\xi) - x(t,\xi)\partial_\xi x(t,\xi),$$

with known analytical solution

$$x(t,\xi) = \frac{\xi}{t+1}\left(1+\sqrt{\frac{t+1}{\exp\left(\frac{\mathrm{Re}}{8}\right)}}\exp\left(\mathrm{Re}\tfrac{\xi^2}{4t+4}\right)\right)^{-1}, \quad (7)$$

taken from Mauli et al. (2021) (see also Black et al. (2022), where a similar setting is used). The spatial domain is set to $\Omega := [0,1]$, the time interval to $\mathbb{T} := [0,4]$, and the Reynolds number is given by $\mathrm{Re} = 1000$. For an illustration of the solution of the full-order model, we refer to Figure 1.

The error on the testing data for the best run is reported in Table 1. The highlighted cell corresponds to the minimal value in the column. We report that the

Table 1. Burgers' eq.: Best relative testing error (6).

method	(A)	(B)	(C)
NNA	0.0110	0.0111	0.0132
$\mathsf{LNA}_{\mathrm{ext}}$	0.0091	0.0108	0.0128
LNA	0.0117	0.0133	0.0202
$\mathsf{LNA}_{\mathrm{ext,fix}}$	0.0097	0.0110	0.0152
NLA	0.4919	0.4918	0.4918

best relative testing errors are achieved by the $\mathsf{LNA}_{\mathrm{ext}}$, with the second-best error obtained by either the standard NNA or the $\mathsf{LNA}_{\mathrm{ext,fix}}$. Further, we observe that the time-extended data set is preferable in this example for linear-nonlinear autoencoder configurations to obtain a good approximation error, as the LNA obtains worse results than the standard NNA. As expected from Theorem 4, the approximation error of the NLA is bounded below by the Kolmogorov barrier. Thus, it is not expected to yield better results than a linear-linear configuration. Indeed, the NLA yields relative errors that almost precisely match the relative error of POD with reduced dimension $r = 1$, i.e., POD-error 0.4927. To achieve a relative error of less than 2% with POD, we must use a reduced dimension of at least $r = 21$. For visualization, we plot the different approximations and their errors for the time $t = 1.35$ in Figure 2. To further illustrate how similar the approximation results of the NLA and the POD are, we plot both in Figure 3 for two separate time instances.

To further inspect the training of the different methods, we also report the average testing error of all 100 model runs in Table 2. We observe that the best average testing error is either obtained by the $\mathsf{LNA}_{\mathrm{ext,fix}}$ or by the $\mathsf{LNA}_{\mathrm{ext}}$, which thus yield the most stable configurations. So again, the lowest errors are reported by configurations utilizing

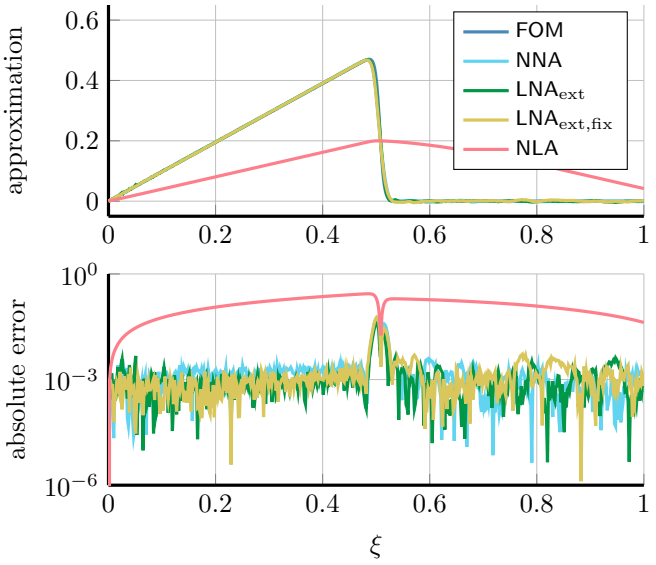

Fig. 2. Burgers' eq.: approximations (top) and absolute errors (bottom) at $t = 0.03$ over different autoencoders with $r = 1$ and scenario (A). The first four lines in the top figure nearly coincide.

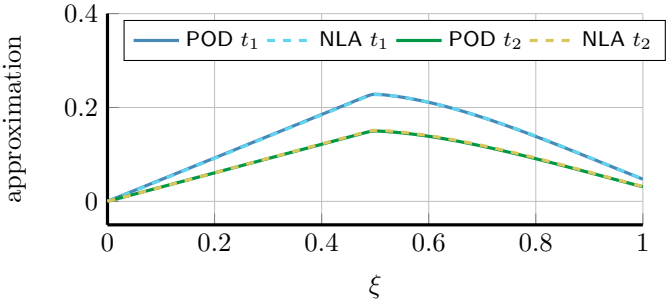

Fig. 3. Burgers' eq.: POD (solid) and NLA (dashed) approximation for times $t_1 = 1.35$, $t_2 = 4.00$.

Table 2. Burgers' eq.: average relative testing error (6).

method	(A)	(B)	(C)
NNA	0.0373	0.0243	0.0265
$\mathsf{LNA}_{\mathrm{ext}}$	0.0162	0.0159	0.0205
LNA	0.0241	0.0248	0.0283
$\mathsf{LNA}_{\mathrm{ext,fix}}$	0.0161	0.0156	0.0212
NLA	0.4920	0.4921	0.4922

the additional time information in training, where the simpler configuration $\mathsf{LNA}_{\mathrm{ext,fix}}$ outperforms the $\mathsf{LNA}_{\mathrm{ext}}$ in two instances. On the other hand, autoencoders trained on the normal data set increase by a higher percentage, rendering the configurations less robust. Again, as expected, the NLA yields robust results, as it mimicks a linear-linear model.

In Table 3, we compare the average training time per epoch and consider all configurations with a nonlinear decoder. First, note that using a linear encoder pays off in the average training time, as all linear-nonlinear autoencoder configurations attain a training time of 60% or less compared to the NNA. Further, we observe that the $\mathsf{LNA}_{\mathrm{ext,fix}}$ achieves the fastest times, which we assume is due to the even simpler design in the encoder. Thus,

Table 3. Burgers' eq.: average training time per epoch.

method	(A)	(B)	(C)
NNA	0.0410	0.0347	0.0293
$\mathsf{LNA}_{\text{ext}}$	0.0227	0.0186	0.0161
LNA	0.0214	0.0180	0.0177
$\mathsf{LNA}_{\text{ext,fix}}$	0.0203	0.0171	0.0152

we achieve a similar or better approximation error with a linear-nonlinear autoencoder with time-extended data while needing less average training time per epoch.

4.3 Advection equation

For our second example, we use the advection equation

$$\partial_t x(t,\xi) + c\partial_\xi x(t,\xi) = 0,$$

with sawtooth initial value

$$x_0(\xi) = \begin{cases} \frac{1}{\sigma}(\xi-\beta), & \beta \le \xi \le \beta+\sigma, \\ 0, & \text{else}, \end{cases}$$

such that the solution is given by $x(t,\xi) = x_0(\xi - ct)$. For the computational setup, we have $\Omega = [0,1]$, $\mathbb{T} = [0,1]$, $c = 1$, $\sigma = 0.1$ and $\beta = 0$. Note that with this choice, the time-to-solution map is not smooth, and hence, it does not satisfy the assumptions from Theorems 1 and 2. Despite the assumptions not being satisfied, we achieve similar results as in the previous example, suggesting that the assumptions in these theorems can be relaxed.

First, we report the best error in Table 4. Although not covered by our theory, we observe that linear-nonlinear autoencoder configurations achieve the best errors in agreement with our first experiment. In particular, $\mathsf{LNA}_{\text{ext,fix}}$ performs best over all parametrizations, with NNA yielding the second-best results. In contrast, NLA is, again as expected, significantly worse; its error matches the relative error of POD with reduced dimension $r = 1$ (POD-error 0.9615). For a relative error of less than 30% with POD, we need a reduced dimension of $r = 35$ in this example. For visualization we plot the different approximations and their errors for the time $t = 0.49$ in Figure 4.

Table 4. Advection eq.: best relative testing error with respect to (6).

method	(A)	(B)	(C)
NNA	0.3662	0.3918	0.3796
$\mathsf{LNA}_{\text{ext}}$	0.3806	0.4050	0.4005
LNA	0.3927	0.3909	0.4075
$\mathsf{LNA}_{\text{ext,fix}}$	0.2825	0.2968	0.3235
NLA	0.9615	0.9615	0.9616

The best average testing errors are reported in Table 5. The $\mathsf{LNA}_{\text{ext,fix}}$ yields the most robust configuration, as observed in the previous example. We notice that the percentage increase is lowest for the $\mathsf{LNA}_{\text{ext,fix}}$ with 2%-16% compared to the other configurations, with 29%-50% increase. We excluded the NLA because, despite its stability, it has a significantly higher error than other configurations due to the Kolmogorov barrier.

Finally, we investigate the point projection property (3) for the different autoencoders. We emphasize that neither

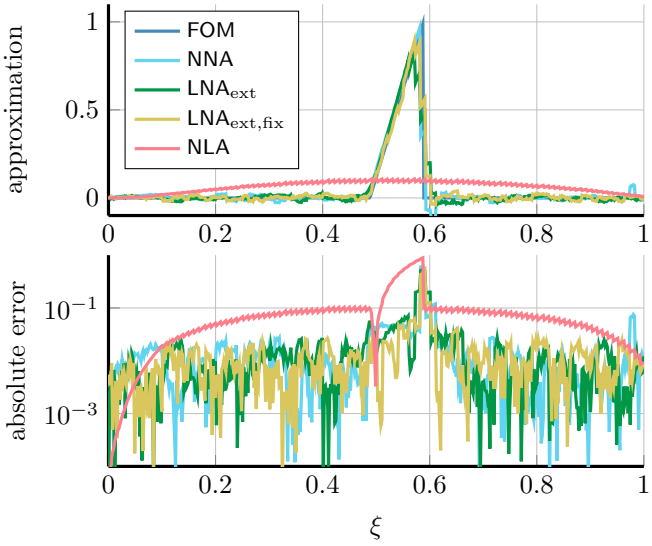

Fig. 4. Advection eq.: approximations (top) and absolute errors (bottom) at $t = 0.49$ over different autoencoders with $r = 1$ and scenario (A).

Table 5. Advection eq.: average relative testing error (6).

method	(A)	(B)	(C)
NNA	0.5511	0.5036	0.5141
$\mathsf{LNA}_{\text{ext}}$	0.5680	0.5867	0.5996
LNA	0.5647	0.5656	0.6070
$\mathsf{LNA}_{\text{ext,fix}}$	0.3288	0.3336	0.3728
NLA	0.9615	0.9615	0.9616

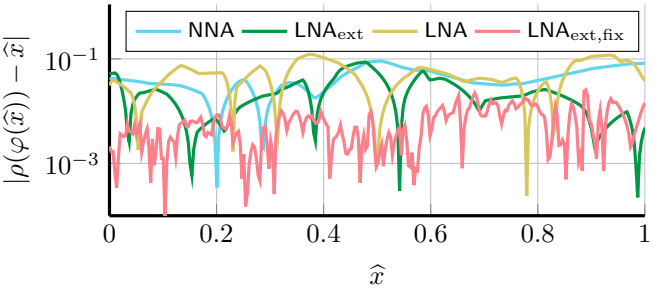

Fig. 5. Advection eq.: point projection property (3) for the trained autoencoders with $r = 1$ and scenario (A).

of the autoencoders strictly enforces the point projection property during training, nor a deviation of it is penalized. However, in (Buchfink et al., 2024b, Thm. 6.4), it is shown that the error of the point projection can be bounded by the least squares error in training. In particular, we expect a smaller deviation from the point projection property for configurations with smaller validation errors. We investigate this numerically for the autoencoder configurations NNA, $\mathsf{LNA}_{\text{ext}}$, LNA, $\mathsf{LNA}_{\text{ext,fix}}$ in Figure 5, where we plot the deviation from the point projection property (in absolute value). We observe that even if this property was not enforced directly in the training, the $\mathsf{LNA}_{\text{ext,fix}}$ obtains the smallest deviation, which is reasonable since it had the best average approximation error.

5. CONCLUSION

We present a novel approach for autoencoder training for MOR by incorporating time and parameter information into the training data. We show that a linear encoder suffices to achieve the same accuracy as the NNA, illustrated in two proof-of-concept examples. For the Burgers' equation, the $\mathsf{LNA}_{\text{ext}}$ matches the NNA's accuracy while having faster training times per epoch. The second example with the advection equation shows that similar results can be obtained even when theoretical assumptions are violated. Additionally, we assess the quality of the point projection property, which is not explicitly included in training. Nevertheless, the $\mathsf{LNA}_{\text{ext,fix}}$ configuration yields stable results consistent with the low training error.

With these proof-of-concept examples, the next step is to numerically validate the theory on further benchmarks and for different autoencoder configurations, such as convolutional autoencoders. Further, we aim to extend this work to the parametric setting and examine how the ROM error relates to the projection error. Eventually, we plan to use the resulting outcomes for the efficient (optimal) control of transport-dominated partial different equations, e.g., extending the recent work by Breiten et al. (2024).

REFERENCES

Antoulas, A.C. (2005). *Approximation of large-scale dynamical systems*. Advances in Design and Control. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA. doi:10.1137/1.9780898718713.

Barnett, J. and Farhat, C. (2022). Quadratic approximation manifold for mitigating the Kolmogorov barrier in nonlinear projection-based model order reduction. *J. Comput. Phys.*, 464, 111348. doi: 10.1016/j.jcp.2022.111348.

Benner, P., Cohen, A., Ohlberger, M., and Willcox, K. (2017). *Model Reduction and Approximation*. Computational Science & Engineering. Society for Industrial and Applied Mathematics, Philadelphia, PA. doi: 10.1137/1.9781611974829.

Black, F., Schulze, P., and Unger, B. (2020). Projection-based model reduction with dynamically transformed modes. *ESAIM: Math. Model. Numer. Anal.*, 54(6), 2011–2043. doi:10.1051/m2an/2020046.

Black, F., Schulze, P., and Unger, B. (2022). Modal decomposition of flow data via gradient-based transport optimization. In R. King and D. Peitsch (eds.), *Active Flow and Combustion Control 2021*, 203–224. Springer International Publishing, Cham. doi:10.1007/978-3-030-90727-3_13.

Breiten, T., Burela, S., and Schulze, P. (2024). Optimal control for a class of linear transport-dominated systems via the shifted proper orthogonal decomposition. *arXiv preprint arXiv:2412.18950*.

Buchfink, P., Glas, S., and Haasdonk, B. (2023). Symplectic model reduction of Hamiltonian systems on nonlinear manifolds and approximation with weakly symplectic autoencoder. *SIAM J. Sci. Comput.*, 45(2), A289–A311. doi:10.1137/21M1466657.

Buchfink, P., Glas, S., and Haasdonk, B. (2024a). Approximation bounds for model reduction on polynomially mapped manifolds. *Comptes Rendus. Mathématique*, 362, 1881–1891. doi:10.5802/crmath.632.

Buchfink, P., Glas, S., Haasdonk, B., and Unger, B. (2024b). Model reduction on manifolds: a differential geometric framework. *Phys. D*, 468, 134299. doi: 10.1016/j.physd.2024.134299.

Cohen, A., DeVore, R., Petrova, G., and Wojtaszczyk, P. (2022). Optimal stable nonlinear approximation. *Found. Comput. Math.*, 22, 607–648. doi:10.1007/s10208-021-09494-z.

Desislavov, R., Martinez-Plumed, F., and Hernández-Orallo, J. (2023). Trends in AI inference energy consumption: Beyond the performance-vs-parameter laws of deep learning. *Sustain. Comput.: Inform. Syst.*, 38, 100857. doi:10.1016/j.suscom.2023.100857.

Ferrero, A., Taddei, T., and Zhang, L. (2022). Registration-based model reduction of parameterized two-dimensional conservation laws. *J. Comput. Phys.*, 457, 111068. doi:10.1016/j.jcp.2022.111068.

Geelen, R., Wright, S., and Willcox, K. (2023). Operator inference for non-intrusive model reduction with quadratic manifolds. *Comput. Meth. Appl. Mech. Eng.*, 403, 115717. doi:10.1016/j.cma.2022.115717.

Golub, G.H. and Van Loan, C.F. (2013). Matrix computations. *Johns Hopkins University Press, 4th edition*.

Grohs, P. and Voigtlaender, F. (2024). Proof of the theory-to-practice gap in dep learning via sampling complexity bounds for neural network approximation spaces. *Found. Comput. Math.*, 24, 1085–1143. doi: 10.1007/s10208-023-09607-w.

Kim, Y., Choi, Y., Widemann, D., and Zohdi, T. (2022). A fast and accurate physics-informed neural network reduced order model with shallow masked autoencoder. *J. Comput. Phys.*, 451, 110841. doi: 10.1016/j.jcp.2021.110841.

Kingma, D.P. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kolmogoroff, A. (1936). Über die beste Annäherung von Funktionen einer gegebenen Funktionenklasse. *Ann. of Math. (2)*, 37(1), 107–110. doi:10.2307/1968691.

Lee, K. and Carlberg, K.T. (2020). Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *J. Comput. Phys.*, 404, 108973. doi:10.1016/j.jcp.2019.108973.

Mauli, R., Lusch, B., and Balaprakash, P. (2021). Reduced-order modeling of advection-dominated systems with recurrent neural networks and convolutional autoencoders. *Phys. Fluids*, 33, 037106. doi: 10.1063/5.0039986.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Reiss, J., Schulze, P., Sesterhenn, J., and Mehrmann, V. (2018). The shifted proper orthogonal decomposition: A mode decomposition for multiple transport phenomena. *SIAM J. Sci. Comput.*, 40(3), A1322–A1344. doi: 10.1137/17M1140571.

Unger, B. and Gugercin, S. (2019). Kolmogorov n-widths for linear dynamical systems. *Adv. Comput. Math.*, 45(5-6), 2273–2286. doi:10.1007/s10444-019-09701-0.