Multistep Neural Networks for Data-driven Discovery of Nonlinear Dynamical Systems

Maziar Raissi[1], Paris Perdikaris[2], and George Em Karniadakis[1]

[1] *Division of Applied Mathematics, Brown University,*
Providence, RI, 02912, USA
[2] *Department of Mechanical Engineering and Applied Mechanics,*
University of Pennsylvania,
Philadelphia, PA, 19104, USA

Abstract

The process of transforming observed data into predictive mathematical models of the physical world has always been paramount in science and engineering. Although data is currently being collected at an ever-increasing pace, devising meaningful models out of such observations in an automated fashion still remains an open problem. In this work, we put forth a machine learning approach for identifying nonlinear dynamical systems from data. Specifically, we blend classical tools from numerical analysis, namely the multi-step time-stepping schemes, with powerful nonlinear function approximators, namely deep neural networks, to distill the mechanisms that govern the evolution of a given data-set. We test the effectiveness of our approach for several benchmark problems involving the identification of complex, nonlinear and chaotic dynamics, and we demonstrate how this allows us to accurately learn the dynamics, forecast future states, and identify basins of attraction. In particular, we study the Lorenz system, the fluid flow behind a cylinder, the Hopf bifurcation, and the Glycoltic oscillator model as an example of complicated nonlinear dynamics typical of biological systems.

1. Introduction

Dynamical systems play a key role in shaping our understanding of the physical world and in defining our ability to predict the evolution of a given process. From the simple swinging motion of a clock pendulum to the complex flow around an airplane wing, the mathematical modeling of dynamical systems can yield a set of tools with which we can analyze the way the current state of the system depends on the past, and predict the possible states we may encounter in the future. Often such tools are precisely known, usually coming in the form of differential equations that are derived from first physical principles, such as the conservation of energy, mass, and momentum [1]. However, in many cases, the sheer complexity of a system can prohibit our complete understanding and render a first principles approach infeasible. In this setting, one may be only able to postulate crude and potentially overly simplified models based on a given a set of empirical observations (see e.g., models for tumor growth [2], social dynamics [3], and the stock market [4]). In the present era of abundant data and advanced machine learning capabilities, a natural question arises: can we automatically discover sufficiently sophisticated and accurate mathematical models of complex dynamical systems directly from data?

The answer to this question pertains to the well-established field of systems identification [5]. Discriminating between *white-, gray-, and black-box* approaches, depending on whether a first principles modeling approach is fully, partially, or not admissible, systems identification aims to devise mathematical models for predicting a future state of a system, given the evolution of a set of previously observed or latent states. Specifically, in the context of identifying nonlinear dynamics, there exist several deterministic and probabilistic tools including radial basis functions [6], neural networks [7], Gaussian processes [8, 9, 10, 11, 12], and nonlinear auto-regressive models such as NARMAX [13] and recurrent neural networks [14]. A common theme among all such methods is the pursuit of learning a nonlinear and potentially multi-variate mapping $\boldsymbol{f}$ that predicts the future system states given a set of data describing the present and past states. More recently, approaches based on symbolic regression [15], sparse regression, and compressive sensing [16, 17] were able to go beyond estimating a *black-box* approximation of the dynamics given by $\boldsymbol{f}$, and return more interpretable models that can uncover the full parametric form of an underlying governing equation. However, in

order to obtain sparse representation of the dynamics, the aforementioned approaches have to rely upon the nontrivial task of choosing "appropriate" sets of basis functions. Consequently, investigating ways of incorporating broader function search spaces is an important area of current and future research.

In this work, we introduce a novel approach to nonlinear systems identification that combines the classical multistep family of time-stepping schemes from numerical analysis [18] with deep neural networks. Inspired by recent developments in *physics-informed deep learning* [19, 20], we construct structured nonlinear regression models that can discover the dynamic dependencies in a given set of temporal data-snapshots, and return a closed form model that can be subsequently used to forecast future states or identify basins of attraction. In contrast to recent approaches to systems identification [16, 17], here we do not have to have direct access or approximations to temporal gradients because the time derivatives are discretized using classical time-stepping rules. Moreover, we are using a richer family of function approximators and consequently we do not have to commit to a particular class of basis functions such as polynomials or sines and cosines. This comes at the cost of losing interpretability of the learned dynamics. However, there is nothing hindering the use of a particular class of basis functions and obtain more interpretable equations.

This paper is structured as follows. In section 2 we provide a detailed overview of the proposed methodology. In section 3.1, we investigate the performance of the proposed framework by applying our algorithm to the two-dimensional damped harmonic oscillator. We then explore the identification of chaotic dynamics of the Lorenz system in section 3.2. As an example of a high dimensional dynamical systems, in section 3.3, we study the Navier-Stokes equations describing the fluid flow behind a cylinder. To illustrate the ability of our method to identify parameterized dynamics, we consider the Hopf normal form in section 3.4. As an example of complicated nonlinear dynamics typical of biological systems, we explore the glycolytic oscillator model in section 3.5. It should be highlighted that all of the examples considered in this work are inspired by the pioneering work of Brunton *et. al.* [16]. Moreover, all data and codes used in this manuscript are publicly available on GitHub at https://github.com/maziarraissi/MultistepNNs.

2. Problem setup and solution methodology

Let us consider nonlinear dynamical systems of the form[1]

$$\frac{d}{dt}\boldsymbol{x}(t) = \boldsymbol{f}\left(\boldsymbol{x}(t)\right), \tag{1}$$

where the vector $\boldsymbol{x}(t) \in \mathbb{R}^D$ denotes the state of the system at time t and the function $\boldsymbol{f}$ describes the evolution of the system. Given noisy measurements of the state $\boldsymbol{x}(t)$ of the system at several time instances $t_1, t_2, \ldots, t_N$, our goal is to determine the function $\boldsymbol{f}$ and consequently discover the underlying dynamical system (1) from data. We proceed by applying the general form of a linear multistep method with M steps to equation (1) and obtain

$$\sum_{m=0}^{M} [\alpha_m \boldsymbol{x}_{n-m} + \Delta t \beta_m \boldsymbol{f}(\boldsymbol{x}_{n-m})] = 0, \quad n = M, \ldots, N. \tag{2}$$

Here, $\boldsymbol{x}_{n-m}$ denotes the state of the system $\boldsymbol{x}(t_{n-m})$ at time t_{n-m}. Different choices for the parameters α_m and β_m result in specific schemes. For instance, the trapezoidal rule

$$\boldsymbol{x}_n = \boldsymbol{x}_{n-1} + \frac{1}{2}\Delta t \left(\boldsymbol{f}(\boldsymbol{x}_n) + \boldsymbol{f}(\boldsymbol{x}_{n-1})\right), \quad n = 1, \ldots, N, \tag{3}$$

corresponds to the case where $M = 1$, $\alpha_0 = -1$, $\alpha_1 = 1$, and $\beta_0 = \beta_1 = 0.5$. We proceed by placing a neural network prior on the function $\boldsymbol{f}$. The parameters of this neural network can be learned by minimizing the mean squared error loss function

$$MSE := \frac{1}{N - M + 1} \sum_{n=M}^{N} |\boldsymbol{y}_n|^2, \tag{4}$$

---

[1]It is straightforward to generalize the dynamics to include parameterization, time dependence, and forcing. In particular, parameterization, time dependence, and external forcing or feedback control $\boldsymbol{u}(t)$ may be added to the vector field according to

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}, t; \lambda\right), \quad \dot{t} = 1, \quad \dot{\lambda} = 0.$$

where

$$\boldsymbol{y}_n := \sum_{m=0}^{M} \left[\alpha_m \boldsymbol{x}_{n-m} + \Delta t \beta_m \boldsymbol{f}(\boldsymbol{x}_{n-m})\right], \quad n = M, \ldots, N, \tag{5}$$

is obtained from the multistep scheme (2).

3. Results

3.1. Two-dimensional damped oscillator

As a first illustrative example, let us consider the two-dimensional damped harmonic oscillator with cubic dynamics; i.e.,

$$\begin{aligned} \dot{x} &= -0.1 \ x^3 + 2.0 \ y^3, \\ \dot{y} &= -2.0 \ x^3 - 0.1 \ y^3. \end{aligned} \tag{6}$$

We use $[x_0 \ y_0]^T = [2 \ 0]^T$ as initial condition and collect data from $t = 0$ to $t = 25$ with a time-step size of $\Delta t = 0.01$. The data are plotted in figure 1. We employ a neural network with one hidden layer and 256 neurons to represent the nonlinear dynamics. As for the multistep scheme (2) we use Adams-Moulton with $M = 1$ steps (i.e., the trapezoidal rule). Upon training the neural network, we solve the identified system using the same initial condition as the one above. Figure 1 provides a qualitative assessment of the accuracy in identifying the correct nonlinear dynamics. Specifically, by comparing the exact and predicted trajectories of the system, as well as the resulting phase portraits, we observe that the algorithm can correctly capture the dynamic evolution of the system.

To investigate the performance of the proposed work-flow with respect to different linear multi-step methods, we have considered the three families that are most commonly used in practice: Adams-Bashforth (AB) methods, Adams-Moulton (AM) methods, and the backward differentiation formulas (BDFs). In tables 1 and 2, we report the relative $\mathcal{L}_2$ error between trajectories of the exact and the identified systems for different members of the class of linear multi-step methods. Interestingly, the Adams-Moulton scheme seems to consistently return more accurate results compared to the Adams-Bashforth and BDF approaches. One intuitive explanation for this
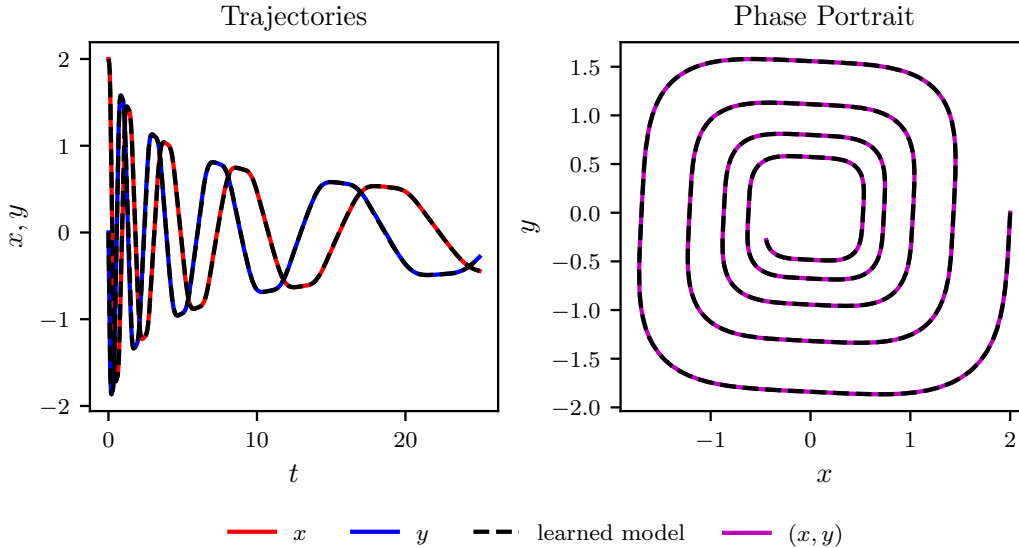
Figure 1: *Harmonic Oscillator:* Trajectories of the two-dimensional damped harmonic oscillator with cubic dynamics are depicted in the left panel while the corresponding phase portrait is plotted in the right panel. Solid colored lines represent the exact dynamics while the dashed black lines demonstrate the learned dynamics. The identified system correctly captures the form of the dynamics and accurately reproduces the phase portrait.

behavior stems from a closer inspection of equation 2. Specifically, the arrangement of the resulting terms for the Adams-Moulton schemes leads to a higher throughput of training data flowing through the neural network during model training as compared to the Adams-Bashforth and BDF cases. This helps regularize the neural network and eventually achieve a better calibration during training. Also, out of the Adams-Moulton family, the trapezoidal rule seems to work the best in practice perhaps due to its superior stability properties [18]. These performance characteristics should be interpreted as product of empirical evidence, and not as concrete theoretical properties of the method. Identification of the latter requires more extensive systematic studies that go beyond the scope of this paper.

In tables 3 and 4, we study the robustness of our results with respect to the gap Δt between pairs of data and with respect to noise in the observations of the system. These results fail to reveal a consistent pattern as larger time-step sizes Δt and larger noise corruption levels sometimes lead to superior accuracy and other times to inferior. In the latter cases, the reasons

6

Scheme \ M	1	2	3	4	5
Adams-Bashforth	1.5e+00	3.1e-02	1.2e-01	4.3e-02	1.2e-02
Adams-Moulton	8.8e-03	1.2e-02	1.6e-02	6.3e-03	1.1e-02
BDF	1.3e+00	8.8e-03	1.3e-02	1.4e-02	1.7e-02

Table 1: *Harmonic Oscillator:* Relative $\mathcal{L}_2$ error between the predicted and the exact trajectory for the first dynamic component $x(t)$ integrated up to time $t = 25$, for different member families of the class of multistep methods, and different number of steps M. Here, the training data is assumed to be noise free, and the neural network architecture is kept fixed to have one hidden layer and 256 neurons.

Scheme \ M	1	2	3	4	5
Adams-Bashforth	1.5e+00	3.0e-02	9.7e-02	3.5e-02	1.2e-02
Adams-Moulton	8.8e-03	1.0e-02	1.6e-02	5.8e-03	1.1e-02
BDF	1.3e+00	8.6e-03	9.9e-03	1.4e-02	1.5e-02

Table 2: *Harmonic Oscillator:* Relative $\mathcal{L}_2$ error between the predicted and the exact trajectory for the second dynamic component $y(t)$ integrated up to time $t = 25$, for different member families of the class of multistep methods, and different number of steps M. Here, the training data is assumed to be noise free, and the neural network architecture is kept fixed to have one hidden layer and 256 neurons.

are obvious, namely that a larger gap Δt makes the approximation in time less accurate, while too much noise is devastating because it would be harder to distinguish between noise and the true dynamics. On the other hand, we do observe some cases in which larger Δt and noise levels may actually help. In these cases, we believe that input noise can act as a regularization mechanism that increases the robustness of the model training procedure, similar to how it has been previously proposed in the neural network literature (see for e.g., denoising autoencoders [21]). Along the same lines, a bigger temporal gap Δt helps because it makes two consecutive time snapshots carry more information simply because they are more dissimilar to one another. On the contrary, if Δt is too small, the importance of the neural network becomes less and less pronounced as seen in equation (2), hence model training becomes infeasible. These empirical results indicate that there exists a problem-dependent sweet spot for the admissible values of the time-step and noise levels that can lead to the best predictive accuracy. Although one may

noise Δt	0.00%	0.01%	0.02%
0.01	5.7e-03	2.4e-02	2.2e-01
0.02	1.8e-02	1.1e-01	1.3e-01
0.03	3.8e-02	9.2e-02	7.8e-01
0.04	5.4e-02	4.0e-02	8.7e-01
0.05	8.3e-02	2.9e-01	9.2e-02

Table 3: *Harmonic Oscillator:* Relative $\mathcal{L}_2$ error between the predicted and the exact trajectory for the first dynamic component $x(t)$ integrated up to time $t = 25$, for different noise magnitudes, and different gap Δt between pairs of snapshots. Here, we are employing the trapezoidal time-stepping scheme and the neural network architecture is kept fix to have one hidden layer and 256 neurons.

noise Δt	0.00%	0.01%	0.02%
0.01	5.9e-03	2.1e-02	2.2e-01
0.02	1.7e-02	1.0e-01	1.1e-01
0.03	3.3e-02	9.0e-02	7.9e-01
0.04	5.2e-02	3.8e-02	8.7e-01
0.05	7.2e-02	2.6e-01	8.2e-02

Table 4: *Harmonic Oscillator:* Relative $\mathcal{L}_2$ error between the predicted and the exact trajectory for the first dynamic component $y(t)$ integrated up to time $t = 25$, for different noise magnitudes, and different gap Δt between pairs of snapshots. Here, we are employing the trapezoidal time-stepping scheme and the neural network architecture is kept fix to have one hidden layer and 256 neurons.

have no control over the noise corrupting the data, the temporal gap Δt could be treated as another hyper-parameter like the number of neurons and hidden layers when setting up the neural network.

Finally, tables 5 and 6 study the robustness of our results with respect to the neural network structure. For this case, more accurate results seem to be obtained with increasing network depth, although increasing the network width seems to have a negative affect for more than 128 neurons per layer. To fully quantify sensitivity with respect to network architecture a more systematic study involving multiple data-sets is needed.

Neurons / Layers	64	128	256
1	9.8e-03	6.1e-03	3.7e-02
2	3.6e-03	1.2e-02	2.4e-02
3	3.4e-03	1.6e-02	4.2e-02

Table 5: *Harmonic Oscillator:* Relative $\mathcal{L}_2$ error between the predicted and the exact trajectory for the first dynamic component $x(t)$ integrated up to time $t = 25$, for different neural network architectures. Here, the training data is assumed to be noise free, the time step size is kept fixed at $\Delta t = 0.01$, and the number of Adams-Moulton steps is fixed at $M = 1$.

Neurons / Layers	64	128	256
1	7.2e-03	5.4e-03	3.5e-02
2	3.3e-03	9.1e-03	2.0e-02
3	3.0e-03	1.4e-02	3.7e-02

Table 6: *Harmonic Oscillator:* Relative $\mathcal{L}_2$ error between the predicted and the exact trajectory for the second dynamic component $y(t)$ integrated up to time $t = 25$, for different neural network architectures. Here, the training data is assumed to be noise free, the time step size is kept fixed at $\Delta t = 0.01$, and the number of Adams-Moulton steps is fixed at $M = 1$.

3.2. Lorenz system

To explore the identification of chaotic dynamics evolving on a finite dimensional attractor, we consider the nonlinear Lorenz system [22]

$$
\begin{aligned}
\dot{x} &= 10(y - x), \\
\dot{y} &= x(28 - z) - y, \\
\dot{z} &= xy - (8/3)z.
\end{aligned}
\tag{7}
$$

We use $[x_0\ y_0\ z_0]^T = [-8\ 7\ 27]^T$ as initial condition and collect data from $t = 0$ to $t = 25$ with a time-step size of $\Delta t = 0.01$. The data are plotted in figures 2 and 3. We employ a neural network with one hidden layer and 256 neurons to represent the nonlinear dynamics. As for the multistep scheme (2) we use Adams-Moulton with $M = 1$ steps (i.e., the trapezoidal rule). Upon training the neural network, we solve the identified system using the same initial condition as the one above. As depicted in figure 2, the learned system correctly captures the form of the attractor.
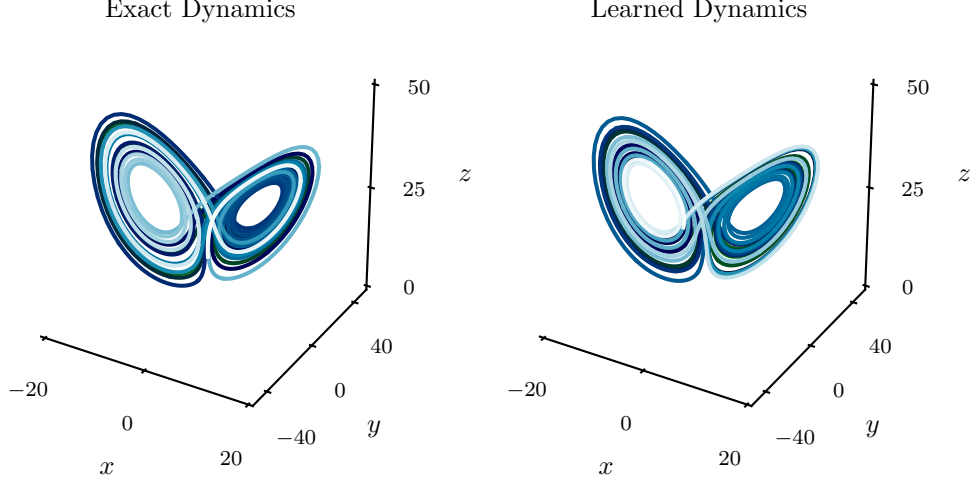
9

Exact Dynamics   Learned Dynamics

Figure 2: *Lorenz System:* The exact phase portrait of the Lorenz system (left panel) is compared to the corresponding phase portrait of the learned dynamics (right panel).

The Lorenz system has a positive Lyapunov exponent, and small differences between the exact and learned models grow exponentially, even though the attractor remains intact. This behavior is evident in figure 3, as we compare the exact versus the predicted trajectories. Small discrepancies due to finite accuracy in the predicted dynamics lead to large errors in the forecasted time-series after $t > 4$, despite the fact that the bi-stable structure of the attractor is well captured (see figure 2).

3.3. Fluid flow behind a cylinder

In this example we collect data for the fluid flow past a cylinder (see figure 4) at Reynolds number 100 using direct numerical simulations of the two dimensional Navier-Stokes equations. In particular, following the problem setup presented in [23] and [24], we simulate the Navier-Stokes equations describing the two-dimensional fluid flow past a circular cylinder at Reynolds number 100 using the Immersed Boundary Projection Method [25, 26]. This approach utilizes a multi-domain scheme with four nested domains, each successive grid being twice as large as the previous one. Length and time are non-dimensionalized so that the cylinder has unit diameter and the flow has unit velocity. Data is collected on the finest domain with dimensions 9×4 at a grid resolution of 449×199. The flow solver uses a 3rd-order Runge Kutta
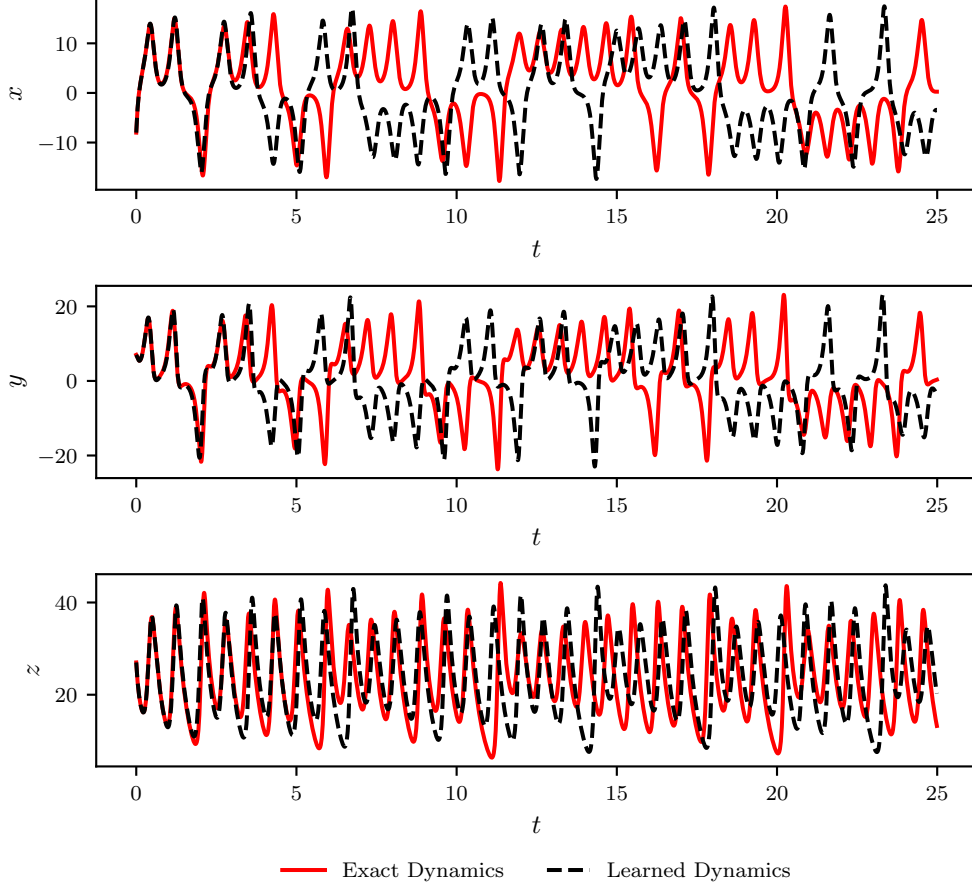
Figure 3: *Lorenz System:* The exact trajectories of the Lorenz systems is compared to the corresponding trajectories of the learned dynamics. Solid blue lines represent the exact dynamics while the dashed black lines demonstrate the learned dynamics.

integration scheme with a time step of t = 0.02, which has been verified to yield well-resolved and converged flow fields. After simulations converge to steady periodic vortex shedding, flow snapshots are saved every $\Delta t = 0.02$. We then reduce the dimension of the system by proper orthogonal decomposition (POD) [27, 16]. The POD results in a hierarchy of orthonormal modes that, when truncated, capture most of the energy of the original system for the given rank truncation. The first two most energetic POD modes capture a significant portion of the energy; the steady-state vortex shedding is a limit
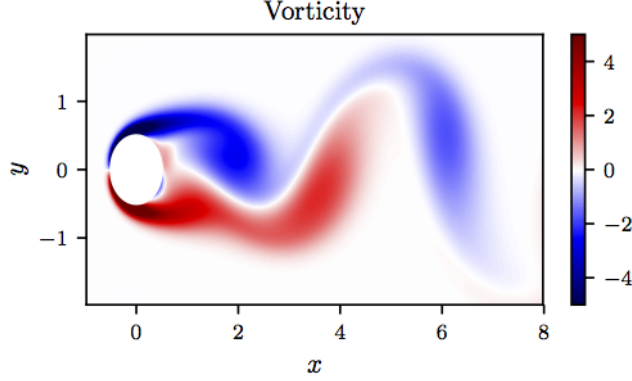
Figure 4: *Flow past a cylinder:* A snapshot of the vorticity field of a solution to the Navier-Stokes equations for the fluid flow past a cylinder.

cycle in these coordinates [16]. An additional mode, called the shift mode, is included to capture the transient dynamics connecting the unstable steady state with the mean of the limit cycle [28]. The resulting POD coefficients are depicted in figure 5.

We employ a neural network with one hidden layer and 256 neurons to represent the nonlinear dynamics shown in figure 5. As for the linear multistep scheme (2) we use Adams-Moulton with $M = 1$ steps (i.e., the trapezoidal rule). Upon training the neural network, we solve the identified system. As depicted in figure 5, the learned system correctly captures the form of the dynamics and accurately reproduces the phase portrait, including both the transient regime as well as the limit cycle attained once the flow dynamics converge to the well known Kárman vortex street.

3.4. Hopf bifurcation

Many real-world systems depend on parameters and, when the parameters are varied, they may go through bifurcations. To illustrate the ability of our method to identify parameterized dynamics, let us consider the Hopf normal form

$$\begin{aligned}
\dot{x} &= \mu x + y - x(x^2 + y^2), \\
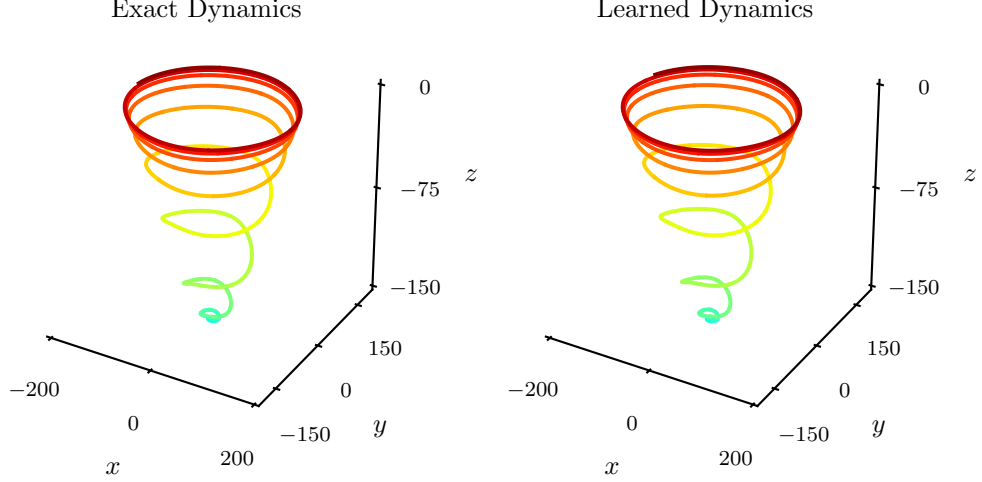\dot{y} &= -x + \mu y - y(x^2 + y^2).
\end{aligned} \tag{8}$$

12

Figure 5: *Flow past a cylinder:* The exact phase portrait of the cylinder wake trajectory in reduced coordinates (left panel) is compared to the corresponding phase portrait of the learned dynamics (right panel).

Our algorithm can be readily extended to encompass parameterized systems. In particular, the system (8) can be equivalently written as

$$
\begin{aligned}
\dot{\mu} &= 0, \\
\dot{x} &= \mu x + y - x(x^2 + y^2), \\
\dot{y} &= -x + \mu y - y(x^2 + y^2).
\end{aligned}
\tag{9}
$$

We collect data from the Hopf system (9) for various initial conditions corresponding to different parameter values for μ. The data is depicted in figure 6. The identified parameterized dynamics is shown in figure 6 for a set of parameter values different from the ones used during model training. The learned system correctly captures the transition from the fixed point for $\mu < 0$ to the limit cycle for $\mu > 0$.

3.5. Glycolytic oscillator

As an example of complicated nonlinear dynamics typical of biological systems, we simulate the glycolytic oscillator model presented in [29] and [16]. The model consists of ordinary differential equations for the concentrations
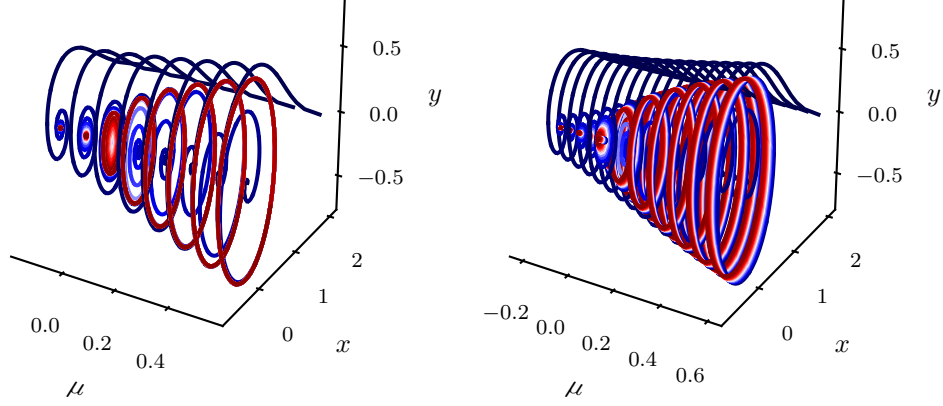
Figure 6: *Hopf bifurcation:* Training data from the Hopf system for various initial conditions corresponding to different parameter values for μ (left panel) is compared to the corresponding phase portrait of the learned dynamics (right panel). It is worth highlighting that the algorithm is tested on initial conditions different from the ones used during training.

of 7 biochemical species; i.e.,

$$
\begin{aligned}
\frac{dS_1}{dt} &= J_0 - \frac{k_1 S_1 S_6}{1 + (S_6/K_1)^q}, \\
\frac{dS_2}{dt} &= 2\frac{k_1 S_1 S_6}{1 + (S_6/K_1)^q} - k_2 S_2(N - S_5) - k_6 S_2 S_5, \\
\frac{dS_3}{dt} &= k_2 S_2(N - S_5) - k_3 S_3(N - S_6), \\
\frac{dS_4}{dt} &= k_3 S_3(A - S_6) - k_4 S_4 S_5 - \kappa(S_4 - S_7), \\
\frac{dS_5}{dt} &= k_2 S_2(N - S_5) - k_4 S_4 S_5 - k_6 S_2 S_5, \\
\frac{dS_6}{dt} &= -2\frac{k_1 S_1 S_6}{1 + (S_6/K_1)^q} + 2k_3 S_3(A - S_6) - k_5 S_6, \\
\frac{dS_7}{dt} &= \psi\kappa(S_4 - S_7) - k S_7.
\end{aligned}
\tag{10}
$$

The parameters of the model are chosen according to table 1 of [29]. As shown in figure 7, data from a simulation of equation (10) are collected from $t = 0$ to $t = 10$ with a time-step size of $\Delta t = 0.01$. We employ a neural network with one hidden layer and 256 neurons to represent the nonlinear dynamics. As for the multi-step scheme (2) we use Adams-Moulton with $M = 1$ steps (i.e., the trapezoidal rule). Upon training the neural network, we solve the identified system using the same initial condition as the ones used for the exact system. As depicted in figure 7, the learned system correctly captures the form of the dynamics.

4. Summary and Discussion

We have presented a machine learning approach for extracting nonlinear dynamical systems from time-series data. The proposed algorithm leverages the structure of well studied multi-step time-stepping schemes such as Adams-Bashforth, Adams Moulton, and BDF families, to construct efficient algorithms for learning dynamical systems using deep neural networks. A key property of the proposed approach is the use of multiple steps which enables us to incorporate memory effects in learning the temporal dynamics and tackle problems with a nonlinear and non-Markovian dynamical structure. Specifically, the use of M steps allows us to decouple the regression complexity due to several temporal lags, ultimately leading to a simpler D-dimensional regression problem, as opposed to an $(M \times D)$-dimensional problem in the case of a brute force NARMAX or recurrent neural network approaches. Although state-of-the-art results are presented for a diverse collection of benchmark problems, there exist a series of open questions mandating further investigation. How could one handle a variable temporal gap Δt, i.e., irregularly sampled data in time? How would common techniques such as batch normalization, drop out, and $\mathcal{L}_1/\mathcal{L}_2$ regularization enhance the robustness of the proposed algorithm and mitigate the effects of over-fitting? How could one incorporate partial knowledge of the dynamical system in cases where certain interaction terms are already known? In terms of future work, interesting directions include the application of convolutional architectures [14] for mitigating the complexity associated with very high-dimensional inputs, as well as studying possible connections with recent studies linking deep neural networks with numerical methods and dynamical systems [30, 31].
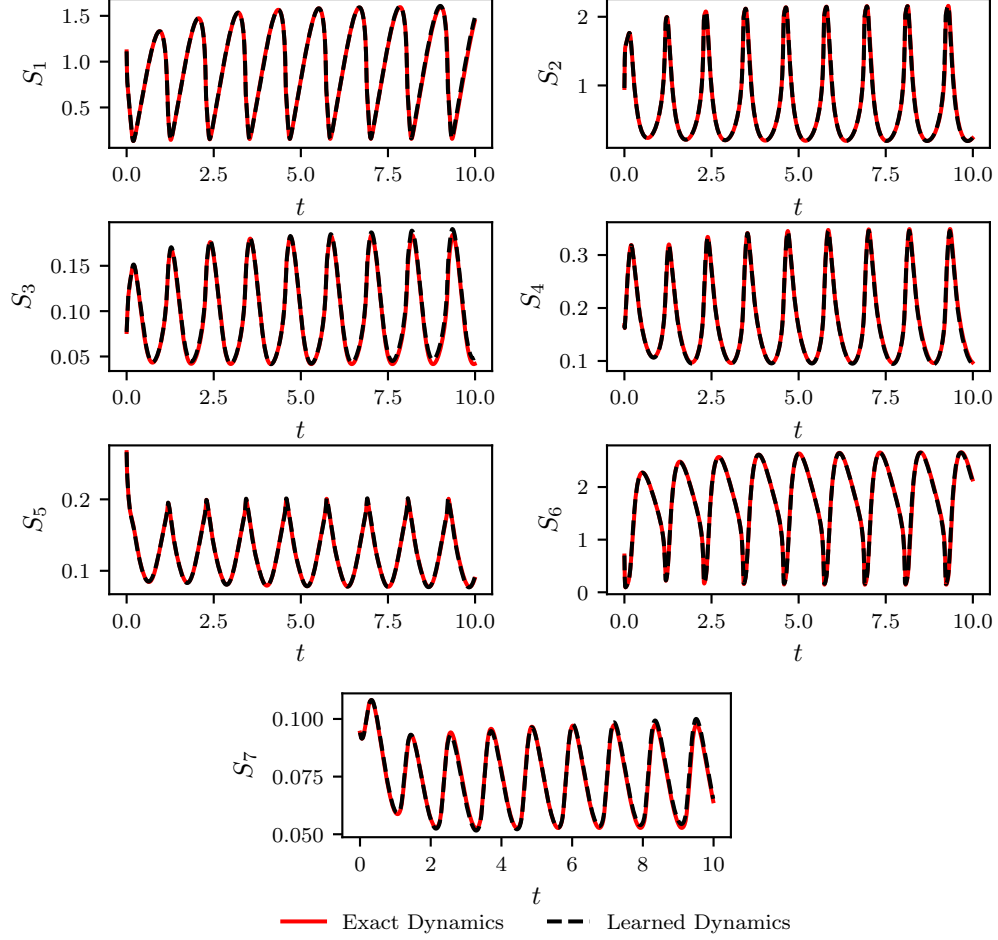
15

Figure 7: *Glycolytic oscillator:* Exact versus learned dynamics for random initial conditions chosen from the ranges provided in table 2 of [29].

Acknowledgements

References

[1] R. Courant, D. Hilbert, Methods of Mathematical Physics, Volume 2: Differential Equations, John Wiley & Sons, 2008.

[2] F. Michor, Y. Iwasa, M. A. Nowak, Dynamics of cancer progression, Nature reviews cancer 4 (2004) 197–205.

[3] C. Castellano, S. Fortunato, V. Loreto, Statistical physics of social dynamics, Reviews of modern physics 81 (2009) 591.

[4] M. Gavin, The stock market and exchange rate dynamics, Journal of international money and finance 8 (1989) 181–200.

[5] L. Ljung, System identification, in: Signal analysis and prediction, Springer, 1998, pp. 163–173.

[6] S. Chen, S. Billings, C. Cowan, P. Grant, Non-linear systems identification using radial basis functions, International Journal of Systems Science 21 (1990) 2513–2539.

[7] A. Cochocki, R. Unbehauen, Neural networks for optimization and signal processing, John Wiley & Sons, Inc., 1993.

[8] J. Kocijan, A. Girard, B. Banko, R. Murray-Smith, Dynamic systems identification with gaussian processes, Mathematical and Computer Modelling of Dynamical Systems 11 (2005) 411–424.

[9] M. Raissi, P. Perdikaris, G. E. Karniadakis, Machine learning of linear differential equations using Gaussian processes, Journal of Computational Physics 348 (2017) 683 – 693.

[10] M. Raissi, G. E. Karniadakis, Hidden physics models: Machine learning of nonlinear partial differential equations, arXiv preprint arXiv:1708.00588 (2017).

[11] M. Raissi, P. Perdikaris, G. E. Karniadakis, Inferring solutions of differential equations using noisy multi-fidelity data, Journal of Computational Physics 335 (2017) 736–746.

[12] M. Raissi, P. Perdikaris, G. E. Karniadakis, Numerical Gaussian processes for time-dependent and non-linear partial differential equations, arXiv preprint arXiv:1703.10230 (2017).

[13] S. A. Billings, Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains, John Wiley & Sons, 2013.

[14] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, MIT press, 2016.

[15] M. Schmidt, H. Lipson, Distilling free-form natural laws from experimental data, science 324 (2009) 81–85.

[16] S. L. Brunton, J. L. Proctor, J. N. Kutz, Discovering governing equations from data by sparse identification of nonlinear dynamical systems, Proceedings of the National Academy of Sciences 113 (2016) 3932–3937.

[17] S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz, Data-driven discovery of partial differential equations, Science Advances 3 (2017) e1602614.

[18] A. Iserles, A first course in the numerical analysis of differential equations, 44, Cambridge University Press, 2009.

[19] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics informed deep learning (part ii): Data-driven discovery of nonlinear partial differential equations, arXiv preprint arXiv:1711.10566 (2017).

[20] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations, arXiv preprint arXiv:1711.10561 (2017).

[21] P. Vincent, H. Larochelle, Y. Bengio, P.-A. Manzagol, Extracting and composing robust features with denoising autoencoders, in: Proceedings of the 25th international conference on Machine learning, ACM, pp. 1096–1103.

[22] E. N. Lorenz, Deterministic nonperiodic flow, Journal of the atmospheric sciences 20 (1963) 130–141.

[23] J. N. Kutz, S. L. Brunton, B. W. Brunton, J. L. Proctor, Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems, volume 149, SIAM, 2016.

[24] S. H. Rudy, S. L. Brunton, J. L. Proctor, J. N. Kutz, Data-driven discovery of partial differential equations, Science Advances 3 (2017).

[25] K. Taira, T. Colonius, The immersed boundary method: a projection approach, Journal of Computational Physics 225 (2007) 2118–2137.

[26] T. Colonius, K. Taira, A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions, Computer Methods in Applied Mechanics and Engineering 197 (2008) 2131–2146.

[27] G. Berkooz, P. Holmes, J. L. Lumley, The proper orthogonal decomposition in the analysis of turbulent flows, Annual review of fluid mechanics 25 (1993) 539–575.

[28] B. R. Noack, K. Afanasiev, M. MORZYŃSKI, G. Tadmor, F. Thiele, A hierarchy of low-dimensional models for the transient and post-transient cylinder wake, Journal of Fluid Mechanics 497 (2003) 335–363.

[29] B. C. Daniels, I. Nemenman, Efficient inference of parsimonious phenomenological models of cellular dynamics using s-systems and alternating regression, PloS one 10 (2015) e0119821.

[30] B. Chang, L. Meng, E. Haber, F. Tung, D. Begert, Multi-level residual networks from dynamical systems view, arXiv preprint arXiv:1710.10348 (2017).

[31] Y. Lu, A. Zhong, Q. Li, B. Dong, Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations, arXiv preprint arXiv:1710.10121 (2017).