

# Generalized Quadratic-Embeddings for Nonlinear Dynamics using Deep Learning

Pawan Goyal\*    Peter Benner<sup>†‡</sup>

\*Max Planck Institute for Dynamics of Complex Technical Systems, 39106 Magdeburg, Germany.

Email: [goyalp@mpi-magdeburg.mpg.de](mailto:goyalp@mpi-magdeburg.mpg.de), ORCID: 0000-0003-3072-7780

<sup>†</sup>Max Planck Institute for Dynamics of Complex Technical Systems, 39106 Magdeburg, Germany.

Email: [benner@mpi-magdeburg.mpg.de](mailto:benner@mpi-magdeburg.mpg.de), ORCID: 0000-0003-3362-4103

<sup>‡</sup>Otto von Guericke University, Universitätsplatz 2, 39106 Magdeburg, Germany

Email: [peter.benner@ovgu.de](mailto:peter.benner@ovgu.de)

**Abstract:** The engineering design process (e.g., control and forecasting) relies on mathematical modeling, describing the underlying dynamic behavior. For complex dynamics behavior, modeling procedures, as well as models, can be intricate, which can make the design process cumbersome. Therefore, it is desirable to have a common model structure, which is also simple enough, for all nonlinear dynamics to enhance design processes. The simplest dynamical model—one can think of—is linear, but linear models are often not expressive enough to apprehend complex dynamics. In this work, we propose a modeling approach for nonlinear dynamics and discuss a common framework to model nonlinear dynamic processes, which is built upon a *lifting-principle*. The preminent idea of the principle is that smooth nonlinear systems can be written as quadratic systems in an appropriate lifted coordinate system without any approximation error. Hand-designing these coordinates is not straightforward. In this work, we utilize deep learning capabilities and discuss suitable neural network architectures to find such a coordinate system using data. We present innovative neural architectures and the corresponding objective criterion to achieve our goal. We illustrate the approach using data coming from applications in engineering and biology.

**Keywords:** Deep learning, lifting-principle for nonlinear dynamics, machine learning, neural networks, nonlinear dynamics, quadratic models.

## Novelty statement:

1. Proposed data-driven approach to learning nonlinear dynamical models.
2. Utilized a lifting-principle, allowing to write smooth nonlinear systems as quadratic systems in appropriate coordinates, which we refer to as quadratic-embeddings.
3. Discovery of quadratic-embeddings using deep neural networks.

Code link: [https://gitlab.mpi-magdeburg.mpg.de/goyalp/quadembed\\_nonlineardyns](https://gitlab.mpi-magdeburg.mpg.de/goyalp/quadembed_nonlineardyns)

## 1. Introduction

Mathematical modeling is a fundamental mainstay of engineering design, e.g., control, optimization, and forecasting. Models accurately capturing dynamical behavior are essential to perform engineering studies. Classically, such models are developed based on first principles or fundamental hypotheses by the field experts. Although the approach is successful in many applications, modeling complex processes such as

climate dynamics and modern robotics dynamics are still difficult. Hence, there is a need for data-driven modeling of dynamical models using time-series data, supported by rapid development in sensor technology, allowing access to a vast amount of data.

Data-driven modeling of dynamical systems from time-series data has been an active research field, see, e.g., [1–14]. With powerful approximation capability of deep neural networks (DNNs), several methods to model dynamic behavior using DNNs has been proposed [15–23]. A beauty of DNN-based modeling is that to fit any arbitrary function, we do not require any hand-design features. Instead, essential features to approximate a function are learned by extracting patterns using data. Despite neural networks being very flexible to model complex dynamical behavior from data, engineering studies such as control and optimization are rather difficult using black box models obtained as DNNs. Therefore, it is desirable to have simple analytic dynamical models capable of capturing complex dynamic behavior. Linear models are among the simplest models for which several comprehensive methods to perform engineering studies have been developed, e.g., for stability and control analysis and prediction [24–26]. However, often linear models in the measured space are not fully capable of capturing a rich nonlinear dynamical behavior. Hence, some rationales are required to incorporate nonlinear behavior. In this direction, the Koopman operator theory provides a tool to write nonlinear systems as linear systems in an infinite-dimensional Hilbert space [27]. Dynamic-mode decomposition and its variants aim at determining a finite-dimensional approximation of the infinite-dimensional Koopman operator using suitable coordinates or observables, see, e.g. [6, 10, 28–31]. Despite being successfully applied in various applications, determining suitable observables for complex processes to approximate the Koopman operator using DMD is not trivial. There has been an attempt to discover a finite-dimensional approximation of the Koopman operator or linear embeddings by utilizing the powerful approximation capabilities of DNNs [21]. It is a promising approach, but for models with eigenspectrum containing continuous parts, it becomes intractably hard to determine a coordinate system whose dynamics can be explained by a linear model. Moreover, mapping to the original measurement coordinates from the discovered Koopman embedding is given by a nonlinear mapping using DNNs.

On the other hand, there exists a lifting principle that enables to write nonlinear dynamic models as quadratic models in a lifted coordinate system, which is finite-dimensional, in contrast to Koopman operator theory [32–34]. Moreover, a linear mapping can define the relationship between the lifted coordinates and the space in which data are collected or the quantity of interest. One can determine such a lifted coordinate system for a given analytical expression of a nonlinear system, but we seek to identify such coordinates in a data-driven framework. To that end, in this work, we develop a DNN framework to learn the lifted coordinates using data. Our approach involves a customized autoencoder design, in which a DNN defines an encoder, and the decoder is just a linear function. Such an autoencoder structure enables the discovery of lifted coordinates whose dynamics can be defined by a quadratic dynamical model. We depict our primary objective in Figure 1.1. We determine the parameters of the autoencoder and quadratic models for the lifted coordinates jointly. We demonstrate the success of our approach to learning dynamics via quadratic embeddings using three examples: the nonlinear pendulum example, the Michaelis-Menten model explaining enzymatic kinetics, and the high-dimensional spiral wave—a reaction-diffusion model. These examples demonstrate the capabilities of the approach to obtain parsimonious dynamical models in lifted coordinates. Notably, the proposed framework provides dynamical models in a state-space form whose structure is close to linear state-space models, thus easing the engineering design process for nonlinear systems.

The remainder of the paper is structured as follows. The next section briefly recaps the lifting principle to determine quadratic embeddings for nonlinear systems. In Section 3, we present a data-driven approach to learning lifted coordinates and the corresponding quadratic models. Section 4 demonstrates the proposed method using three examples from different disciplines. We finally conclude the paper with a short summary and future avenues for further research in Section 5.

## 2. Quadratic-embeddings for nonlinear systems

McCormick in 1976 [35] proposed a convex relaxation to solve nonlinear non-convex optimization problems. The central idea lies in lifting a given nonlinear non-convex optimization problem into a high-dimensional using auxiliary variables. Despite increasing the dimension of the original problem, we can solve the lifted problem more efficiently than the original one. A similar philosophy has been developed in the context of

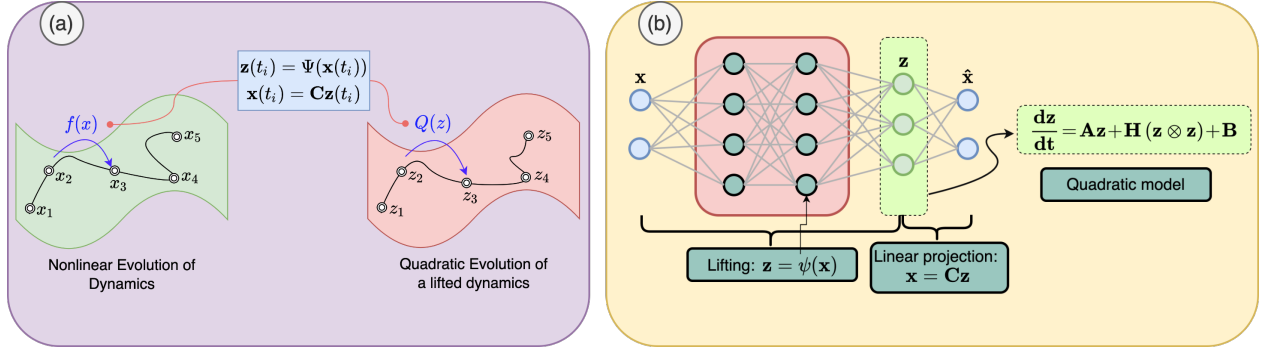


Figure 1.1.: The figure illustrates that nonlinear dynamical systems can be written as quadratic dynamical systems in an appropriate finite-dimensional lifted coordinate system. Moreover, from the lifted coordinate, we can obtain the quantity of interest by a linear projection. In the right plot, we depict a neural network architecture to learn a lifted coordinate system that has the desired quadratic embeddings.

dynamical systems. To ease the engineering design process, particularly control-design for general nonlinear systems, we can recast nonlinear systems as polynomial systems, e.g., [32, 33]. Furthermore, for constructing low-dimensional models for large-scale nonlinear dynamical systems, Gu [34] proposed a methodology that first rewrites nonlinear systems as quadratic systems and then employs reduced-order modeling techniques for quadratic systems. For given nonlinear analytic systems, there are approaches to rewrite them as polynomial or quadratic systems [32–34]. It has been shown that nonlinear systems consisting of basic elementary smooth nonlinear functions (e.g., trigonometric, exponential, rational), or a composition of these elementary functions, can always be rewritten as a quadratic system in the lifted coordinates without any approximation error. Moreover, the dimension of the lifted coordinates increases linearly with respect to the dimension of the original nonlinear system, and one can bound the dimension of the lifted coordinate system as well.

Inspired by the lifting philosophy, we can transform all smooth nonlinear systems satisfying the above assumptions as a quadratic system. Furthermore, we can construct the original state vector or our measurement space using a linear projection of the lifted variables. To illustrate the approach, we consider a simple rational nonlinear system that often appears in biological applications. The nonlinear model is as follows:

$$\dot{\mathbf{x}}(t) = -\frac{\mathbf{x}(t)}{1 + \mathbf{x}(t)}. \tag{2.1}$$

To write the above nonlinear system as a quadratic system, we define lifted coordinates as follows:

$$\mathcal{L}(\mathbf{x}) := \begin{bmatrix} \mathbf{x} \\ 1 \\ \frac{1}{1 + \mathbf{x}} \\ \mathbf{x} \\ \frac{\mathbf{x}}{(1 + \mathbf{x})^2} \end{bmatrix} \equiv \mathbf{y}. \tag{2.2}$$

Note that  $\mathbf{x}$  can be constructed using a linear projection of  $\mathbf{y}$ ; precisely, using a matrix  $\mathbf{C} = [1, 0, 0]$ , i.e.,  $\mathbf{x} = \mathbf{C}\mathbf{y}$ . Moreover, the differential equation for  $\mathbf{y}$  can be given as

$$\begin{bmatrix} \dot{y}_1 \\ \dot{y}_2 \\ \dot{y}_3 \end{bmatrix} = \begin{bmatrix} -y_1 y_2 \\ y_2 y_3 \\ y_3 (y_2 - 2y_3) \end{bmatrix}, \tag{2.3}$$

which is a quadratic system with  $y_i$  being the  $i$ -th component of  $\mathbf{y}$ . These discussions indicate that given an analytical form of a nonlinear system, we can find a lifted coordinate system in which we can write the dynamics as quadratic systems. We highlight that the lifted variables are not unique. Since we only have access to data, we do not have the gratification of hand-designing the lifted coordinate. Even if we had, hand-designed coordinates are not straightforward for complex processes. Therefore, we next discuss a data-driven approach to learning a desired lifted coordinate system.

### 3. Data-driven discovery of quadratic-embeddings using deep learning

Inspired by the lifting principle, we highlight that smooth nonlinear systems can be rewritten as a quadratic system in a lifted coordinate. Also, the coordinate allows the construction of the quantity of interest by a linear projection, as depicted in Figure 1.1(a). In the following, we formulate our problem statement.

**Problem 3.1.** *Given data  $\{\mathbf{x}(t_1), \dots, \mathbf{x}(t_N)\}$  and derivative information  $\{\dot{\mathbf{x}}(t_1), \dots, \dot{\mathbf{x}}(t_N)\}$ , we seek to determine a lifting  $\mathbf{x} \mapsto \mathbf{z}$  such that*

(a)  $\mathbf{z}$  satisfies

$$\dot{\mathbf{z}}(t) = \mathbf{A}\mathbf{z} + \mathbf{H}(\mathbf{z} \otimes \mathbf{z}) + \mathbf{B}, \quad \text{and} \quad (3.1)$$

(b)  $\mathbf{x}$  can be recovered by a linear projection of  $\mathbf{z}$ , i.e.,  $\mathbf{x} = \mathbf{C}\mathbf{z}$ .

Since we only have access to data, we cannot identify the lifted coordinates analytically. We rather learn such them using data. For this, we utilize the profound approximation capabilities of DNNs to learn the lifting. To that end, we propose a particular autoencoder design, where the encoder is defined by a DNN denoted by  $\Psi$ , which is parameterized by  $\theta$ , such that it provides a lifting for a given  $\mathbf{x}$ , i.e.,  $\mathbf{z} = \Psi(\mathbf{x})$ . The decoder is given by a linear function which maps the lifted variables back to the space of measurements by a linear function, i.e.,  $\mathbf{x} = \mathbf{C}\mathbf{z}$ . If we train the autoencoder alone, then we may obtain any arbitrary mapping, and it is not necessary that the dynamics of the learned lifting can be described by a quadratic model. Therefore, it is essential to learn the parameters of the autoencoder to determine liftings so that among infinitely many possibilities, we can identify ones that fulfill the conditions (a) and (b) in Problem 3.1.

Towards achieving our goals, we assume to have access to the derivative information of  $\mathbf{x}$ . Then, we can compute the derivative of the lifted variable  $\mathbf{z}$  using the chain-rule and automatic differentiation as  $\dot{\mathbf{z}} = \nabla_{\mathbf{x}}\Psi(\mathbf{x})\dot{\mathbf{x}}$ . We enforce that a quadratic equation as in (3.1) can also provide the derivative information of  $\mathbf{z}$ ; hence, we add the following term in the loss function:

$$\mathcal{L}_{\dot{\mathbf{z}}\dot{\mathbf{x}}} = \|\nabla_{\mathbf{x}}\Psi(\mathbf{x})\dot{\mathbf{x}} - (\mathbf{A}\mathbf{z} + \mathbf{H}(\mathbf{z} \otimes \mathbf{z}) + \mathbf{B})\| \quad \text{with } \mathbf{z} = \Psi(\mathbf{x}). \quad (3.2)$$

Moreover, as in [20], we can also reconstruct the derivative information of the original variable  $\mathbf{x}$  using the derivative information of the lifted variable  $\mathbf{z}$ . We know that there is a linear map between  $\mathbf{x}$  and  $\mathbf{z}$ , thus we have

$$\dot{\mathbf{x}} = \mathbf{C}\dot{\mathbf{z}} = \mathbf{C}(\mathbf{A}\mathbf{z} + \mathbf{H}(\mathbf{z} \otimes \mathbf{z}) + \mathbf{B}). \quad (3.3)$$

This gives rise to the second part of the loss function as follows:

$$\mathcal{L}_{\dot{\mathbf{x}}\dot{\mathbf{z}}} = \|\dot{\mathbf{x}} - \mathbf{C}(\mathbf{A}\mathbf{z} + \mathbf{H}(\mathbf{z} \otimes \mathbf{z}) + \mathbf{B})\|. \quad (3.4)$$

Naturally, the construction the original variable  $\mathbf{x}$  through  $\mathbf{z}$  can be obtained as follows:

$$\mathbf{x} = \mathbf{C}\mathbf{z} = \mathbf{C}\Psi(\mathbf{x}), \quad (3.5)$$

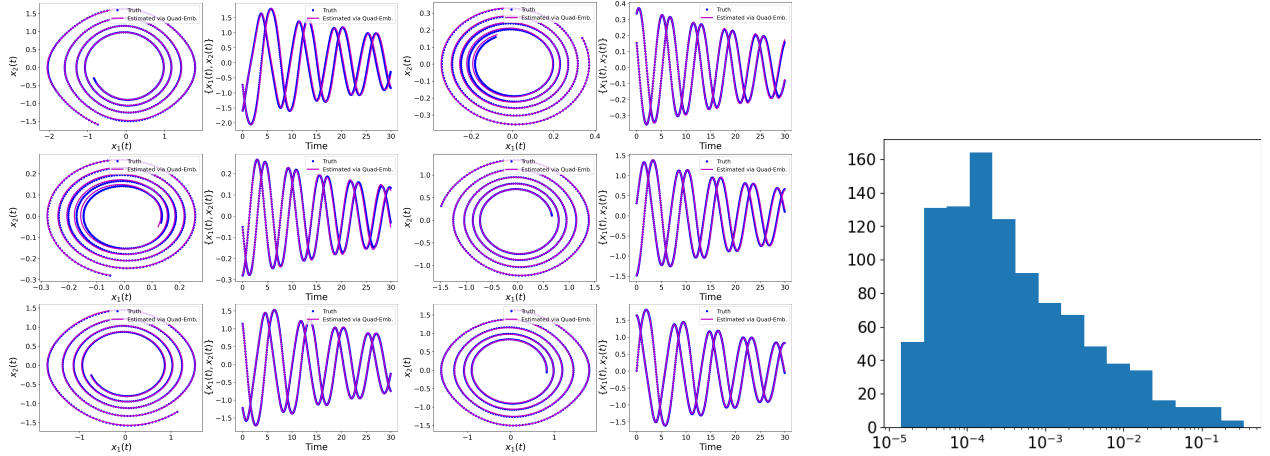
yielding the third element of the loss function

$$\mathcal{L}_{\text{encdec}} = \|\mathbf{x} - \mathbf{C}\Psi(\mathbf{x})\|. \quad (3.6)$$

Combining all these elements (3.2), (3.3), and (3.6), we have the following total loss:

$$\mathcal{L} = \lambda_1\mathcal{L}_{\text{encdec}} + \lambda_2\mathcal{L}_{\dot{\mathbf{x}}\dot{\mathbf{z}}} + \lambda_3\mathcal{L}_{\dot{\mathbf{z}}\dot{\mathbf{x}}}, \quad (3.7)$$

where  $\lambda_{\{1,2,3\}}$  are the hyper-parameters. We optimize simultaneously the parameters of the autoencoder  $\{\theta, \mathbf{C}\}$  and the system matrices  $\{\mathbf{A}, \mathbf{H}, \mathbf{B}\}$ , minimizing the loss  $\mathcal{L}$  in (3.7). Furthermore, we highlight that once the autoencoder is trained and system matrices defining the dynamics are obtained, we require the nonlinear encoder only to get the corresponding initial condition for the lifted variable  $\mathbf{z}$ . We can obtain the trajectory of  $\mathbf{z}$  using a desired numerical integration method, so the trajectory of  $\mathbf{x}$  can be determined using the decoder. Hence, the acquired dynamic modeling is parsimonious by construction compared to DNNs, which directly model the vector field of  $\mathbf{x}$ .



(a) A comparison of the trajectories obtained from the learned quadratic model with the ground truth.

(b) Histogram of the  $L_2$ -error of the trajectories.

Figure 4.1.: Nonlinear pendulum example: A comparison of the learned quadratic embedding and corresponding dynamical model on the testing phase is presented.

## 4. Results

We demonstrate our approach for three applications from different disciplines: a nonlinear pendulum, Michaelis–Menten kinetics, and a 2D reaction-diffusion system. All the information about training and hyper-parameters is given in the appendix.

### 4.1. A nonlinear pendulum

In the first example, we consider a nonlinear pendulum which is governed by the following second-order equation:

$$\ddot{\mathbf{x}}(t) = -\sin(\mathbf{x}(t)) - 0.05\dot{\mathbf{x}}, \tag{4.1}$$

which, in the first-order companion form can be written as

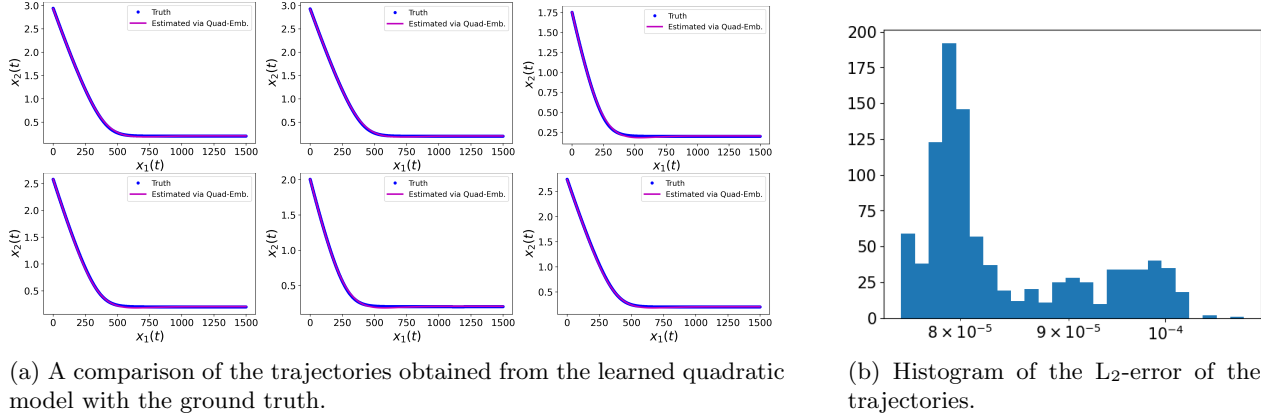
$$\begin{bmatrix} \dot{\mathbf{x}}_1(t) \\ \dot{\mathbf{x}}_2(t) \end{bmatrix} = \begin{bmatrix} -\sin(\mathbf{x}_2(t)) - 0.05\mathbf{x}_1(t) \\ \mathbf{x}_1(t) \end{bmatrix}, \tag{4.2}$$

where  $\mathbf{x}_1(t) = \dot{\mathbf{x}}(t)$  and  $\mathbf{x}_2(t) = \mathbf{x}(t)$ . We collect the data in time interval  $t \in [0, 10]s$  with 1000 different initial conditions. We choose an initial position and velocity of the pendulum in the range of  $[-2, 2]$ . We assume to know the gradient of  $\mathbf{x}_1$  and  $\mathbf{x}_2$ . We train the autoencoder and system matrices  $\{\mathbf{A}, \mathbf{H}, \mathbf{B}, \mathbf{C}\}$  by setting the dimension of the lifted variables to 4. Having learned the lifted coordinates, we test the learned autoencoder and quadratic model using 1000 initial conditions, which have not been part of training. We show 6 exemplary trajectories, choosing every 200th one, in Figure 4.1 and plot the histogram of the  $L_2$ -error of the truth trajectories and those obtained via lifting and by integrating the quadratic model. In the testing phase, we test our system for time  $[0, 30]s$ , although in the training phase, the data are collected for time  $[0, 10]s$  only. The figures indicate that the proposed methodology allows the construction of dynamical models, capturing the dynamic behavior, and most of the error in the testing is around  $10^{-4}$ .

### 4.2. Michaelis-Menten model

Next, we consider an example coming from biology, namely Michaelis-Menten kinetics model [36] which explains enzymatic reactions. The dynamics are given by a one-dimensional differential equation of the form:

$$\dot{\mathbf{x}}(t) = \frac{0.6 - 3\mathbf{x}(t)}{1 + \frac{10\mathbf{x}(t)}{3}}. \tag{4.3}$$



(a) A comparison of the trajectories obtained from the learned quadratic model with the ground truth.

(b) Histogram of the  $L_2$ -error of the trajectories.

Figure 4.2.: Michaelis-Menten model: A comparison of the learned quadratic embedding and corresponding dynamical model in the testing phase is presented.

For this example, we collect data for 50 random initial conditions in the range of  $\mathbf{x}(0) \in [1, 3]$  in the time interval  $t \in [0, 15]s$ . We learn a 2-dimensional lifted variable and a quadratic model describing its dynamics. We then test our learned model for 1000 different initial conditions and compare the trajectories obtained from the underlying truth and the learned models. The results are shown in Figure 4.2 which indicates the accurate capture of the dynamics of  $\mathbf{x}$  through the quadratic embeddings  $\mathbf{z}$ . Moreover, the histogram plot suggests most of the testing error is below  $10^{-4}$ .

### 4.3. Reaction-diffusion model

In various complex applications, dynamics are governed by partial differential equations, resulting in a high-dimensional dataset. These high-dimensional data are often highly spatially and temporarily correlated. Hence, these data can be projected into a low-dimensional subspace by using the dominant subspace, which can be determined using, for example, proper-orthogonal decomposition (POD) or principle-component analysis (PCA). Then, one can use these projected low-dimensional data to obtain a lifting and a corresponding quadratic model to describe its dynamics which closely follows the operator inference approach [8, 37].

To illustrate the methodology, we consider a nonlinear reaction-diffusion example as considered in [20], which is governed by

$$u_t = (1 - (u^2 + v^2))u + \beta(u^2 + v^2)v + d_1(u_{xx} + u_{yy}), \tag{4.4a}$$

$$v_t = -\beta(u^2 + v^2)u + (1 - (u^2 + v^2))v + d_2(v_{xx} + v_{yy}), \tag{4.4b}$$

with parameters  $\beta = 1, d_1 = d_2 = 0.1$ . The model exhibits a damped spiral wave behavior, which has two most dominant modes. The coupled PDE is discretized using a uniform  $100 \times 100$  mesh and we take 100 points from a uniform grid in the time interval  $[0, 5]s$ . Note that the dimension of the data is  $2 \cdot 10^4$ , which is projected on a 2-dimensional subspace using the first two dominant POD modes. Using the projected low-dimensional data, we aim to learn a coordinate system whose dynamics can be explained by a quadratic model. For this example, we consider the first 75 points for training and the remaining 25 points for testing. Once the lifting and quadratic model are learned, we examine the model on the testing data. For this, we integrate the learned lifted variables from  $t = 0$  to  $t = 5$  and project back the low-dimensional data on the full grid using the POD modes. We first plot the evolution of the POD coefficients through the lifted variable in Figure 4.3, which faithfully determines their values in the testing regime. Moreover, since we have an analytic quadratic model for the lifted variable, we can integrate beyond the testing regime and extrapolate the POD coefficients trajectory. This nicely illustrates the underlying damped behaviors when we plot the evolution of the POD coefficients for the time until 25s. Furthermore, We show a comparison for the testing regime in Figure 4.4, which clearly indicates a faithful capturing of the important dynamics.

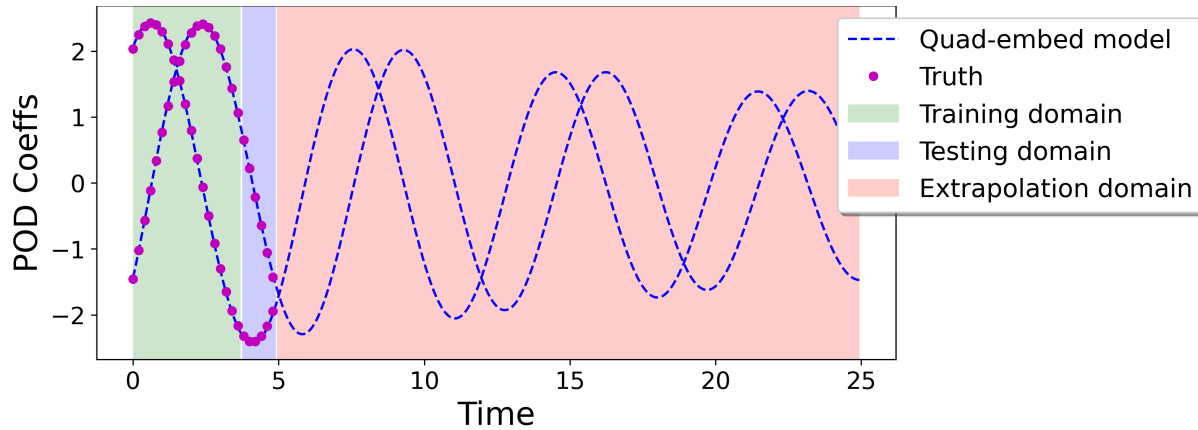


Figure 4.3.: Reaction-diffusion model: Evolution of the POD coefficients in training, testing and extrapolation regimes.

## 5. Discussions

In this work, we have discussed a unifying representation for nonlinear systems, namely quadratic embeddings. It is inspired by the fact that smooth nonlinear dynamical systems can be written as quadratic systems in an appropriate lifted coordinate system. One can potentially hand-design lifted variables for a given analytical expression for nonlinear systems. But in a data-driven setting, where the aim is to learn dynamical models, it is not straightforward to hand-design the lifting. Therefore, in this work, we have utilized profound approximation capabilities of deep neural networks to learn a lifting so that its dynamics can be given by a quadratic model, as well as the mapping between the learned coordinate system and the quantity of interest is linear. To achieve the goal, we have proposed a particular autoencoder design and an objective function for determining the parameters of the autoencoder and system matrices of the quadratic model. As a result, we obtain parsimonious models since the learned differential equations have a simple analytical expression, and the neural network for the encoder is only used to obtain initial conditions for the equation. We have illustrated the methodology using three examples from mechanical, biology, and chemical applications.

This work opens several promising avenues for further research. Since the encoder that determines lifted variables is associated with deep neural networks, it demands a diverse dataset and intense computational resources. Although recent advancements in GPUs and efficient deep learning libraries such as TensorFlow [38] and PyTorch [39] can allow overcoming computational difficulties, the interpretability and generalizability of these networks still need further investigation. Often, deep learning models are good interpolating models, but their extrapolation capabilities are questionable. Therefore, it is essential to incorporate physical knowledge and any available prior knowledge about the origin of the data in the learning process. As a result, this is not only expected to improve the interpretability and generalizability of the learned model but also may extrapolate better outside the training regime. It may also reduce the data required for training as it can be compensated by embedding physical laws. Moreover, one of the critical hyper-parameters in our approach is the dimension of the lifted variables. It is desirable to find a minimal dimension of it while keeping the desired accuracy. It would be worthwhile to develop a suitable automatic approach to determine the dimension of the lifted coordinate system. Additionally, using physical and domain knowledge in deep learning frameworks can enhance the interpretability and provide some physics-informed quadratic embeddings, i.e., quadratic embeddings for Hamiltonian systems. The proposed framework can be extended to various other classes, such as systems with parameters and control. Special treatment is required for noisy measurements, for which ideas proposed in [23, 40, 41] can be combined with our methodology. In the future, we will apply our approach to more challenging and important applications arising in science and technology to construct models where first-principle modeling remains difficult, and the learned models can enhance the engineering design process.

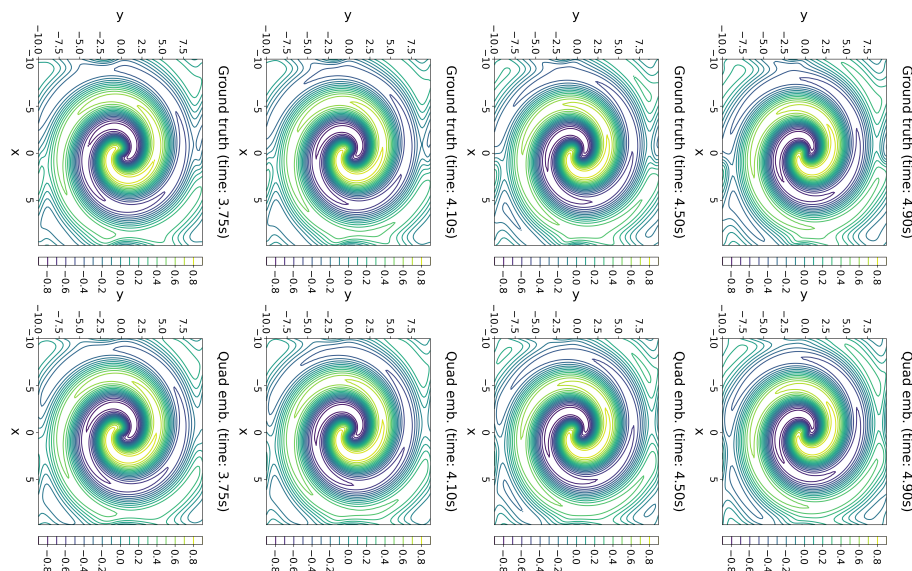


Figure 4.4.: Reaction-diffusion model: A comparison of the trajectories obtained from the learned quadratic model with the ground truth in the testing regime which has not been a part of training.

## References

- [1] J. P. Crutchfield and B. S. McNamara, “Equations of motion from a data series,” *Complex Sys.*, vol. 1, no. 417-452, p. 121, 1987.
- [2] J. Bongard and H. Lipson, “Automated reverse engineering of nonlinear dynamical systems,” *Proc. Nat. Acad. Sci. U.S.A.*, vol. 104, no. 24, pp. 9943–9948, 2007.
- [3] C. Yao and E. M. Bollt, “Modeling and nonlinear parameter estimation with Kronecker product representation for coupled oscillators and spatiotemporal systems,” *Physica D: Nonlinear Phenomena*, vol. 227, no. 1, pp. 78–99, 2007.
- [4] M. D. Schmidt, R. R. Vallabhajosyula, J. W. Jenkins, J. E. Hood, A. S. Soni, J. P. Wikswo, and H. Lipson, “Automated refinement and inference of analytical models for metabolic networks,” *Phy. Biology*, vol. 8, no. 5, p. 055011, 2011.
- [5] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. Henningson, “Spectral analysis of nonlinear flows,” *J. Fluid Mech.*, vol. 641, no. 1, pp. 115–127, 2009.
- [6] P. J. Schmid, “Dynamic mode decomposition of numerical and experimental data,” *J. Fluid Mech.*, vol. 656, pp. 5–28, 2010.
- [7] P. Benner, S. Gugercin, and K. Willcox, “A survey of projection-based model reduction methods for parametric dynamical systems,” *SIAM Rev.*, vol. 57, no. 4, pp. 483–531, 2015.
- [8] B. Peherstorfer and K. Willcox, “Data-driven operator inference for nonintrusive projection-based model reduction,” *Comp. Meth. Appl. Mech. Eng.*, vol. 306, pp. 196–215, 2016.
- [9]
- [10] N. Takeishi, Y. Kawahara, and T. Yairi, “Learning Koopman invariant subspaces for dynamic mode decomposition,” *Adv. Neural Inform. Processing Systems*, vol. 30, pp. 1130–1140, 2017.
- [11] P. Goyal and P. Benner, “Discovery of nonlinear dynamical systems using a Runge-Kutta inspired dictionary-based sparse regression approach,” *Proc. Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 478, no. 2262, p. 20210883, 2022.



- [12] A. J. Mayo and A. C. Antoulas, “A framework for the solution of the generalized realization problem,” *Linear Algebra Appl.*, vol. 425, no. 2–3, pp. 634–662, 2007. Special Issue in honor of P. A. Fuhrmann, Edited by A. C. Antoulas, U. Helmke, J. Rosenthal, V. Vinnikov, and E. Zerz.
- [13] Z. Drmač, S. Gugercin, and C. Beattie, “Vector fitting for matrix-valued rational approximation,” *SIAM J. Sci. Comput.*, vol. 37, no. 5, pp. A2346–A2379, 2015.
- [14] Y. Nakatsukasa, O. Sète, and L. N. Trefethen, “The AAA algorithm for rational approximation,” *SIAM J. Sci. Comput.*, vol. 40, no. 3, pp. A1494–A1522, 2018.
- [15] R. Rico-Martinez, J. Anderson, and I. Kevrekidis, “Continuous-time nonlinear signal processing: a neural network based approach for gray box identification,” in *Proc. IEEE Workshop on Neural Networks for Signal Processing*, pp. 596–605, IEEE, 1994.
- [16] R. Rico-Martinez, I. Kevrekidis, and K. Krischer, “Nonlinear system identification using neural networks: dynamics and instabilities,” *Neural Networks for Chemical Engineers*, pp. 409–442, 1995.
- [17] R. Gonzalez-Garcia, R. Rico-Martinez, and I. Kevrekidis, “Identification of distributed parameter systems: A neural net based approach,” *Computers & Chemical Engrg.*, vol. 22, pp. S965–S968, 1998.
- [18] A. Mardt, L. Pasquali, H. Wu, and F. Noé, “VAMPnets for deep learning of molecular kinetics,” *Nature Commun.*, vol. 9, no. 1, pp. 1–11, 2018.
- [19] P. R. Vlachas, W. Byeon, Z. Y. Wan, T. P. Sapsis, and P. Koumoutsakos, “Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks,” *Proc. Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 474, no. 2213, p. 20170844, 2018.
- [20] K. Champion, B. Lusch, J. N. Kutz, and S. L. Brunton, “Data-driven discovery of coordinates and governing equations,” *Proc. Nat. Acad. Sci. U.S.A.*, vol. 116, no. 45, pp. 22445–22451, 2019.
- [21] B. Lusch, J. N. Kutz, and S. L. Brunton, “Deep learning for universal linear embeddings of nonlinear dynamics,” *Nature Commun.*, vol. 9, no. 1, pp. 1–10, 2018.
- [22] R. T. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, “Neural ordinary differential equations,” in *Advances Neural Inform. Processing Sys.*, pp. 6571–6583, 2018.
- [23] P. Goyal and P. Benner, “Learning dynamics from noisy measurements using deep learning with a Runge-Kutta constraint,” *Workshop paper at the Symbiosis of Deep Learning and Differential Equations – NeurIPS*, 2021. Available at <https://openreview.net/forum?id=G5i2aj7v7i>.
- [24] K. J. Åström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2021.
- [25] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press, 2022.
- [26] K. Ogata *et al.*, *Modern Control Engineering*, vol. 5. Prentice Hall Upper Saddle River, NJ, 2010.
- [27] B. O. Koopman, “Hamiltonian systems and transformation in Hilbert space,” *Proc. Nat. Acad. Sci. U.S.A.*, vol. 17, no. 5, p. 315, 1931.
- [28] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, “A data-driven approximation of the Koopman operator: Extending dynamic mode decomposition,” *J. Nonlinear Sci.*, vol. 25, no. 6, pp. 1307–1346, 2015.
- [29] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. Philadelphia, USA: Society of Industrial and Applied Mathematics, 2016.

- [30] Q. Li, F. Dietrich, E. M. Bollt, and I. G. Kevrekidis, “Extended dynamic mode decomposition with dictionary learning: A data-driven adaptive spectral decomposition of the koopman operator,” *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 27, no. 10, pp. 103–111, 2017.
- [31] P. Benner, C. Himpe, and T. Mitchell, “On reduced input-output dynamic mode decomposition,” *Adv. Comput. Math.*, vol. 44, no. 6, pp. 1821–1844, 2018.
- [32] M. A. Savageau and E. O. Voit, “Recasting nonlinear differential equations as S-systems: A canonical nonlinear form,” *Mathematical Biosciences*, vol. 87, no. 1, pp. 83–115, 1987.
- [33] A. Papachristodoulou and S. Prajna, “Analysis of non-polynomial systems using the sum of squares decomposition,” in *Positive Polynomials in Control* (A. Henrion, Didierand Garulli, ed.), pp. 23–43, Springer, 2005.
- [34] C. Gu, “QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems,” *IEEE Trans. Comput. Aided Des. Integr. Circuits. Syst.*, vol. 30, no. 9, pp. 1307–1320, 2011.
- [35] G. P. McCormick, “Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems,” *Mathematical Programming*, vol. 10, no. 1, pp. 147–175, 1976.
- [36] L. Michaelis, M. L. Menten, *et al.*, “Die kinetik der invertinwirkung,” *Biochem. Z.*, vol. 49, no. 333-369, p. 352, 1913.
- [37] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox, “Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems,” *Physica D: Nonlinear Phenomena*, vol. 406, p. 132401, 2020.
- [38] M. Abadi *et al.*, “TensorFlow: Large-scale machine learning on heterogeneous systems,” 2015. Software available from tensorflow.org.
- [39] A. Paszke *et al.*, “PyTorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Inform. Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, pp. 8024–8035, 2019.
- [40] S. H. Rudy, J. N. Kutz, and S. L. Brunton, “Deep learning of dynamics and signal-noise decomposition with time-stepping constraints,” *J. Comput. Phys.*, vol. 396, pp. 483–506, 2019.
- [41] P. Goyal and P. Benner, “Neural ODEs with irregular and noisy data,” *arXiv preprint arXiv:2205.09479*, 2022.

## A. Implementation details

Here, we provide the necessary details used in our experiments. All the experiments are carried out on NVIDIA<sup>®</sup> P100 GPU. Table A.1 contains all the necessary hyper-parameters for our illustrative examples. In our training, we have set  $\lambda_{\{1,2,3\}}$  in (3.7) to one for simplicity; however, we believe that determining a good balance for these different losses using cross-validation can improve the performance of the approach. For training, we have also utilized a decaying learning rate, for which we have reduced our learning rate by  $\frac{1}{10}$ th after every 1000th epoch. Furthermore, we have utilized a hard-pruning feature in our training. After our initial desired epochs for training, we set those coefficients in matrices  $\{\mathbf{A}, \mathbf{H}, \mathbf{B}\}$  to zero, which are below a tolerance (given in Table A.1). Then, we re-train all parameters for half the number of the initial epochs with half of the initially-set learning rate.

Parameters	Pendulum example	Michaelis-Menten model	Reaction-diffusion model
Encoder layers [neurons]	[64, 64, 64]	[8, 8, 8]	[16, 16]
Lifted coordinate system dimension	4	2	2
Learning rate	$10^{-3}$	$10^{-3}$	$10^{-3}$
Batch size	$2^{15}$	$2^{15}$	75
Activation function	<b>selu</b>	<b>selu</b>	<b>selu</b>
Weight decay	$10^{-4}$	$10^{-4}$	$10^{-4}$
Epochs	2500	2500	2500
Tolerance	$5 \cdot 10^{-2}$	$1 \cdot 10^{-2}$	$5 \cdot 10^{-2}$

Table A.1.: The table contains all the hyper-parameters to learn the encoder parameters and matrices defining lifted-coordinate dynamics.