AFRL-AFOSR-VA-TR-2023-0205

Numerical Methods for Prediction and Discovery with Big Data

Xiu, Dongbin
OHIO STATE UNIVERSITY
1960 KENNY RD
COUMBUS, OH,
US

11/25/2022
Final Technical Report

Air Force Research Laboratory
Air Force Office of Scientific Research
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
20221125	Final	START DATE 20180515	END DATE 20220514

4. TITLE AND SUBTITLE
Numerical Methods for Prediction and Discovery with Big Data

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
	FA9550-18-1-0102	61102F
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Dongbin Xiu

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
OHIO STATE UNIVERSITY
1960 KENNY RD
COUMBUS, OH
US

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Air Force Office of Scientific Research 875 N. Randolph St. Room 3112 Arlington, VA 22203	AFRL/AFOSR RTA2	AFRL-AFOSR-VA-TR-2023-0205

12. DISTRIBUTION/AVAILABILITY STATEMENT
A Distribution Unlimited: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The major objective of the project is to develop a mathematical and numerical framework for scientific prediction and discovery in the realm of big data. Our goal is to develop a set of mathematical and numerical tools that are applicable to big data and subsequently take advantage of the potential and opportunities offered by big data. More specifically, we aim at developing numerical algorithms to discover the physical and mathematical laws behind observational data and create reliable predictive models for the unknown systems. During the course of the project, the PI and his team made tremendous progresses on data driven discovery and prediction. Moreover, modern machine learning (ML) tools such as deep neural network (DNN) were adopted during the project and enabled us to develop highly flexible and powerful algorithms for data driven modeling. The most notable outcomes of the project include the following. - A novel framework of flow map learning (FML) for unknown dynamical systems. This establishes a rigorous mathematical foundation of data driven modeling of dynamical systems. Learning in the form of flow map enables us to design rigorous and flexible numerical predictive tools. - Learning parametric systems. The FML is extended to unknown dynamical systems with parametric dependence.The resulting learning algorithm is able to model the parameter dependence of the system and create an effective model for UQ analysis. - Learning non-autonomous systems.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U	UU	21

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
FARIBA FAHROO	426-8429

DISTRIBUTION A: Distribution approved for public release.

Final Report: AFOSR FA9550-18-1-0102
Numerical Methods for Prediction and Discovery with Big Data
Program manager: Dr. Fariba Fahroo

PI: Dongbin Xiu
Department of Mathematics, The Ohio State University
Email: xiu.16@osu.edu
Reporting date: August, 2022

Abstract

The major objective of the project is to ***develop a mathematical and numerical framework for scientific prediction and discovery in the realm of big data.*** Our goal is to develop a set of mathematical and numerical tools that are applicable to big data and subsequently take advantage of the potential and opportunities offered by big data. More specifically, we aim at *developing numerical algorithms to discover the physical and mathematical laws behind observational data and create reliable predictive models for the unknown systems.*

During the course of the project, the PI and his team made tremendous progresses on data driven discovery and prediction. Moreover, modern machine learning (ML) tools such as deep neural network (DNN) were adopted during the project and enabled us to develop highly flexible and powerful algorithms for data driven modeling. The most notable outcomes of the project include the following.

- A novel framework of flow map learning (FML) for unknown dynamical systems. This establishes a rigorous mathematical foundation of data driven modeling of dynamical systems. Learning in the form of flow map enables us to design rigorous and flexible numerical predictive tools.
- Learning parametric systems. The FML is extended to unknown dynamical systems with parametric dependence. The resulting learning algorithm is able to model the parameter dependence of the system and create an effective model for UQ analysis.
- Learning non-autonomous systems. This is a significant extension of the FML methodology to learning unknown dynamical systems with time dependent inputs. By employing a local approximation technique, the modified FML method is able to accurately predict the unknown system subject to external time dependent inputs that are never in training data.
- Learning partially observed systems. This is a development to address a practical need — data are often not available for every state variables. Instead, only a few variables can be observed. We developed a DNN learning method, driven by rigorous mathematical formulation, to effectively learn and model the unknown system for the observed variables.
- Learning partial differential equation (PDE) system. The FML for dynamical systems is extended to learning of unknown PDE. Two approaches are developed: learning in modal space and in nodal space. While the modal space learning is mathematically appealing, the nodal space learning is more practical and requires a specialized DNN structure.

The project has resulted in publication of 20 high quality journal papers. It has supported two post-doctoral researchers, one of whom finished the term during the project and took a tenure track professor position in another university.

1 Project Report Overview

The project started on May 15, 2018 and ended on May 14, 2022. During the span of the project, the PI and his supporting team successfully accomplished, and in many fronts significantly exceeded, all the major goals of the project.

1.1 Project Objectives

The major objective of the project is to *develop a mathematical and numerical framework for scientific prediction and discovery in the realm of big data.* More specifically, we aim at *developing numerical algorithms to discover the physical and mathematical laws behind observational data and create reliable predictive models for the unknown systems.* The major outcomes of the project consist of the following:

- Establishment of a novel framework for learning of unknown laws behind data. The framework employs operator learning in the form of flow map – flow map learning (FML). It is a major advancement in the field of data driven learning, and has firmly established itself as a rigorous, robust, and exceptionally powerful numerical strategy.
- Development of DNN (deep neural network) algorithms for flow map learning (FML) for practical systems involving unknown parameters, unknown external controls, missing variables, or hidden parameters. These developments significantly enhance the applicability of FML methods and enable practitioners to create highly accurate models for unknown complex systems, such as chaotic systems, even when data are missing.

1.2 Accomplishment Summary

During the course of the project, the research team, led by PI Dongbin Xiu, has conducted vigorous research revolving around the major objectives of the project. Substantial amount of progresses have been made, and all research goals are met, or even exceeded in many fronts. Our specific accomplishments include the following.

- *Framework for flow map learning (FML) of unknown dynamical systems.* We developed FML as a novel numerical framework for learning the laws behind dynamical data. Learning through flow map serves as a rigorous mathematical foundation. Moreover, it greatly enhances the learning ability of the method, as in practical systems the flow maps are often of considerably simpler form, despite the complexity of the solutions. Consequently, FML allows us to model and learn dynamical systems of much higher complexity (compared to other existing data driven learning methods).
- *Learning parametric systems for UQ.* We extended the FML method to learning unknown systems subject to uncertain parameters. By incorporating the parametric dependence in the DNN structure, the resulting DNN FML method is able to accurately predict the system behavior with different parameters. This enables us to conduct parametric study of the dynamics, and more importantly, UQ (uncertainty analysis) of the unknown system.
- *Learning non-autonomous systems.* Non-autonomous systems, i.e., systems subject to external inputs, remained a roadblock for effective learning. The difficulty was overcome by our extension of FML for non-autonomous systems. To account for the various kinds of external inputs that are not in training data, we developed a local approximation and parameterization method, in conjunction with FML, to facilitiate not only the learning of the system but also the prediction of the system for future time when the external inputs can be completely different.

- *Systems with missing variables.* For most practical systems, it is impossible to require modelers to possess data for all the system variables. Consequently, most modeling and learning tasks face data sets with (many) missing variables. Motivated by the celebrated Mori-Zwanzig formulation, we developed memory-based FML to learn and model unknown systems with missing variables. The resulting FML DNN structure possesses remarkable learning abilities – it is capable of accurately model and predict the dynamical behaviors of the observed variables, even though they do not form a mathematically close system.
- *Learning unknown partial differential equations (PDEs).* The FML framework was also extended to the learning of PDEs. Compared to dynamical systems (which are finite dimensional), PDEs are of infinite dimensions. Non-trivial extensions of the FML framework into infinte dimensions were developed, and new DNN structures were proposed. The resulting learning method is capable of accurately prediction PDE systems with shocks, even when the training data sets contain no shocks.

2 Accomplishments

We now present a technical summary of the accomplishments. To unify notation, let us consider a system of ordinary differential equations (ODEs),

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}), \qquad \mathbf{x}(0) = \mathbf{x}_0, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^n$ are the state variables. We assume that the form of the governing equations, which manisfests itself via $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$, is unknown. We assume that trajectory data are available for the state observables $\mathbf{x}$. Let N_T be the total number of such observed trajectories. For each i-th trajectory, we have

$$\mathbf{X}^{(i)} = \left\{ \mathbf{x}\left(t_k^{(i)}\right) \right\}, \qquad k = 1, \ldots, K^{(i)}, \quad i = 1, \ldots, N_T, \tag{2}$$

where $\{t_k^{(i)}\}$ are discrete time instances at which the data are available, and $K^{(i)}$ is the total number of data in the i-th trajectory. For notational convenience, we shall assume a constance time step

$$\Delta \equiv t_{k+1}^{(i)} - t_k^{(i)}, \qquad \forall k = 1, \ldots, K^{(i)} - 1, \quad i = 1, \ldots, N_T. \tag{3}$$

We then seek to develop a numerical model for the evolution dynamics of $\mathbf{x}(t)$, without knowing the true model (1).

2.1 Flow Map Learning (FML) Framework

Compared to the existing learning approaches in the literature, the flow map learning (FML) method developed in [7] is significantly more flexible in its learning ability and mathematically rigorous in its theoretical foundation. For the trajectory data (2), we can re-group the data into pairs of two adjacent time instances. Since for an autonomous system like (1), time t can be arbitrarily shifted and only the relative time difference is relevant. We can then define the data set as

$$\{\mathbf{x}_j(0), \mathbf{x}_j(\Delta)\}, \qquad j = 1, \ldots, J, \tag{4}$$

where J the total number of such data pairs.

On the other hand, the true (and unknown) system (1) defines a flow map $\mathbf{\Phi}: \mathbb{R}^n \to \mathbb{R}^n$, such that $\mathbf{x}(s_1) = \mathbf{\Phi}_{s_1-s_0}(\mathbf{x}(s_0))$. We then have

$$\begin{aligned} \mathbf{x}(\Delta) &= \mathbf{x}(0) + \int_0^\Delta \mathbf{f}(\mathbf{x}(s))ds \\ &= \mathbf{x}(0) + \int_0^\Delta \mathbf{f}(\mathbf{\Phi}_s(\mathbf{x}(0)))ds \\ &= [\mathbf{I}_n + \mathbf{\Psi}(\cdot)]\,(\mathbf{x}(0)), \end{aligned} \tag{5}$$

where $\mathbf{I}_n$ is the identity matrix of size $n \times n$, and for any $\mathbf{x} \in \mathbb{R}^n$,

$$\mathbf{\Psi}(\mathbf{x}) = \int_0^\Delta \mathbf{f}(\mathbf{\Phi}_s(\mathbf{x}))ds.$$

Based on (5), we proposed in [7] to use residual network (ResNet)to model the system. The ResNet has a structure shown in Fig. 1 and defines a map

$$\mathbf{y}^{out} = [\mathbf{I}_n + \mathbf{N}]\left(\mathbf{y}^{in}\right), \tag{6}$$

where $\mathbf{N}: \mathbb{R}^n \to \mathbb{R}^n$ is the operator corresponding to a fully connected deep neural network. Upon using the data set (4), the ResNet (6) can be trained to approximate the dynamics (5), with the deep network operator $\mathbf{N} \approx \mathbf{\Psi}$.

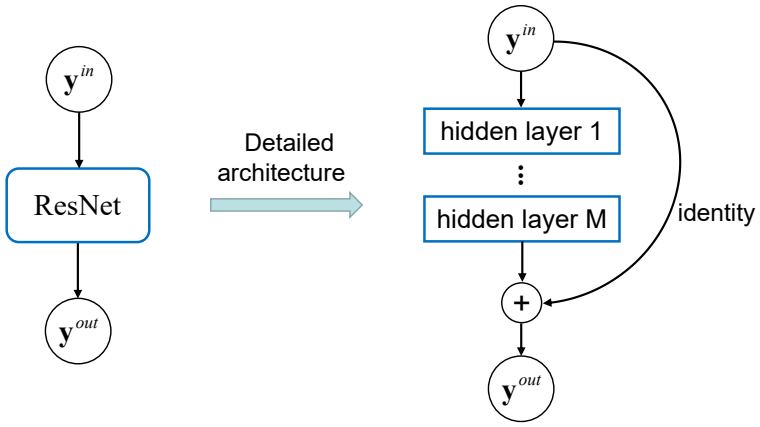


Fig. 1: Schematic of the ResNet structure for one-step approximation.

2.1.1 An Example of ResNet Learning

As an example of the FML learning, we consider a system of nonlinear differential-algebraic equations (DAE), which are used to model a genetic toggle switch in *Escherichia coli* ([3]). It is composed of two repressors and two constitutive promoters, where each promoter is inhibited by the represssor that is transcribed by the opposing promoter. The system of equations are as follows,

$$\begin{cases} \dot{x}_1 = \frac{\alpha_1}{1+x_2^\beta} - x_1, \\ \dot{x}_2 = \frac{\alpha_2}{1+z^\gamma} - x_2, \\ z = \frac{x_1}{(1+[\mathrm{IPTG}]/K)^\eta}. \end{cases}$$

Here we use the system as the unknown truth and generate synthetic training data. The FML DNN structure is trained on the synthetic training to model the dynamics of this unknown true system. We also use the true system to generate validation date to examine the prediction accuracy of the DNN predictions. In Fig. 2, the predictions by the learned DNN model are shown against the true solutions. We observe that the DNN model produces excellent accuracy in its predictions. Such results with high accuracy have been observed for many systems and reported in [7].

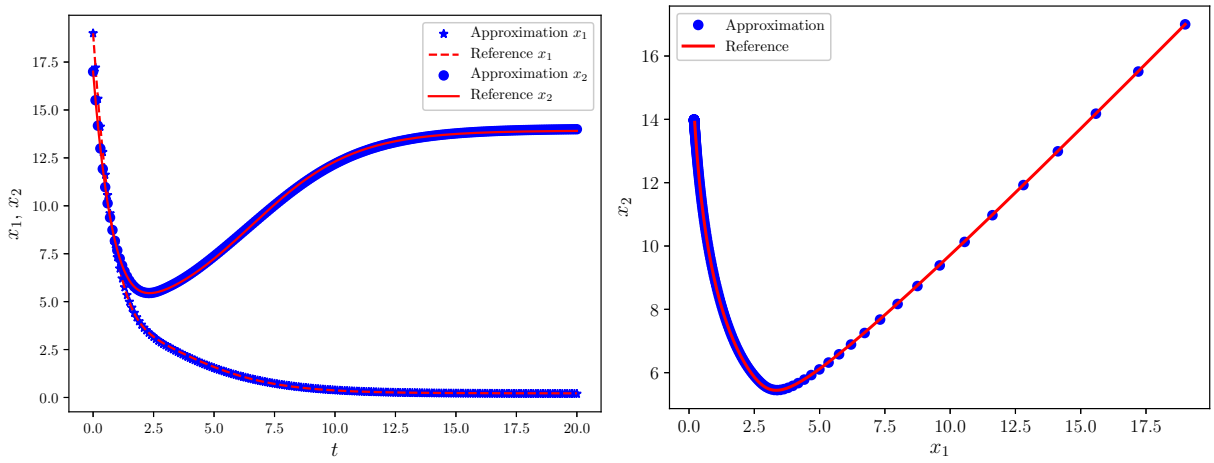


Fig. 2: ResNet FML of Differential-Algebraic system. Left: x_1 and x_2 prediction; Right: phase portrait

2.2 Learning Parametric Systems and UQ

Let us consider a parameterized system

$$\frac{d}{dt}\mathbf{x}(t;\boldsymbol{\alpha}) = \mathbf{f}(\mathbf{x},\boldsymbol{\alpha}), \qquad \mathbf{x}(0) = \mathbf{x}_0, \tag{7}$$

where $\mathbf{x} \subseteq \mathbb{R}^d$ are state variables and $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_\ell) \in I_{\boldsymbol{\alpha}} \subseteq \mathbb{R}^\ell$ are system parameters. We are interested in the solution behavior with respect to varying parameters, while having no knowledge of the governing equations (7). In the context of uncertainty quantification (UQ), the parameters are equipped with a probability measure over $I_{\boldsymbol{\alpha}}$. We are interested in understanding the various solution statistics with respect to the input $\boldsymbol{\alpha}$. Our goal is to create an accurate numerical model for the system and conduct UQ analysis, using only data of the state variable $\mathbf{x}$ and the parameters $\boldsymbol{\alpha}$.

The ResNet FML method for the system (1) can be extended to readily incorporate the parameters into the network. Moreover, in many practical problems, the time steps in the data set are not a constant. Variable time step can also be incorporated in the DNN structure as an additional input. The resulting DNN structure is shown in Fig. 3. Upon successful training of the DNN, the trained DNN serves as a parameterized model for the unknown system (7). We can conduct system prediction via iterative use of the model. Let $\delta_k \in I_\delta$ be a sequence of time steps and $\mathbf{x}_0$ be a given initial condition. Then, for any given system parameter $\boldsymbol{\alpha} \in I_{\boldsymbol{\alpha}}$ that may not be in the training data, we have a predictive model

$$\begin{cases} \widehat{\mathbf{x}}(t_0;\boldsymbol{\alpha}) = \mathbf{x}_0, \\ \widehat{\mathbf{x}}(t_{k+1};\boldsymbol{\alpha}) = \widehat{\mathbf{x}}(t_k;\boldsymbol{\alpha}) + \widehat{\mathbf{N}}(\widehat{\mathbf{x}}(t_k;\boldsymbol{\alpha}),\boldsymbol{\alpha},\delta_k;\Theta^*), \\ t_{k+1} = t_k + \delta_k, \qquad k = 0, 1, \ldots. \end{cases} \tag{8}$$

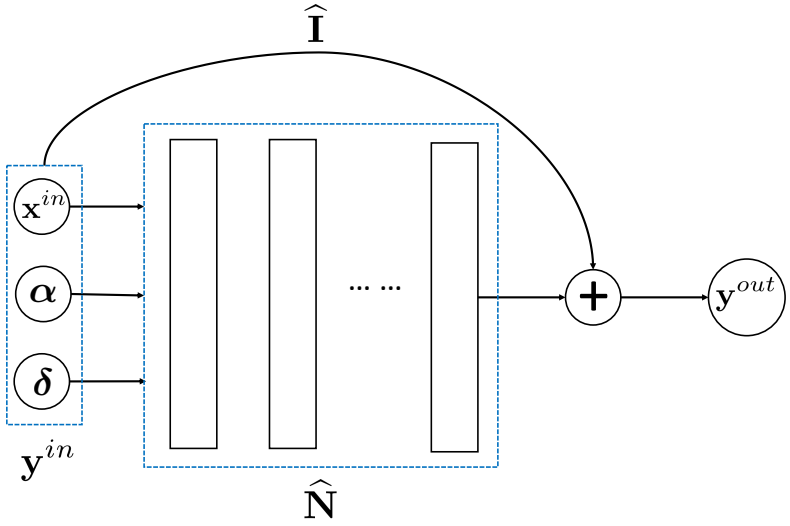


Fig. 3: Structure of DNN for learning systems subject to parameters.

This serves as a predictive model for the true solution $\mathbf{x}(t;\boldsymbol{\alpha},\mathbf{x}_0)$ of the unknown system (7) at the time instances $t \in \{t_k, k = 0, 1, \dots\}$, for any given parameter value $\boldsymbol{\alpha}$ and initial condition $\mathbf{x}_0$.

When the predictive model (8). is learned and constructed from the data, uncertainty quantification (UQ) can be readily carried out. Let $\rho_{\boldsymbol{\alpha}}$ be the probability density the parameters $\boldsymbol{\alpha}$. Statistical information of the true solution $\mathbf{x}(t;\boldsymbol{\alpha})$ of (7) can approximated by applying the required statistical analysis on the DNN model (8). For example, the mean and variance of the solution can be approximated as

$$\begin{aligned} \mathbb{E}_{\alpha}[\mathbf{x}(t;\boldsymbol{\alpha})] &\approx \int_{I_{\boldsymbol{\alpha}}} \widehat{\mathbf{x}}(t;\mathbf{y})\rho_{\boldsymbol{\alpha}}(\mathbf{y})\, d\mathbf{y}, \\ Var[\mathbf{x}] &\approx \int_{I_{\boldsymbol{\alpha}}} \left[\widehat{\mathbf{x}}(t;\mathbf{y}) - \mathbb{E}_{\boldsymbol{\alpha}}[\widehat{\mathbf{x}}(t;\mathbf{y})]\right]^2 \rho_{\boldsymbol{\alpha}}(\mathbf{y})\, d\mathbf{y}. \end{aligned} \tag{9}$$

The integrals of $\widehat{\mathbf{x}}$ can be further approximated sampling based method, e.g., Monte Carlo or quadratue rule.

More details can be found in [5], where the method was first developed by the PI's team.

2.2.1 UQ Modeling Example: Cell Signaling Cascade

For demonstration, let us consider modeling of autocrine cell-signaling loop. The true (and unknown) model is as follows.

$$\begin{aligned} \frac{de_{1p}}{dt} &= \frac{I}{1+Ge_{3p}} \frac{V_{\max,1}(1-e_{1p})}{K_{m,1}+(1-e_{1p})} - \frac{V_{\max,2}e_{1p}}{K_{m,2}+e_{1p}}, \\ \frac{de_{2p}}{dt} &= \frac{V_{\max,3}e_{1p}(1-e_{2p})}{K_{m,3}+(1-e_{2p})} - \frac{V_{\max,4}e_{2p}}{K_{m,4}+e_{2p}}, \\ \frac{de_{3p}}{dt} &= \frac{V_{\max,5}e_{2p}(1-e_{3p})}{K_{m,5}+(1-e_{3p})} - \frac{V_{\max,6}e_{3p}}{K_{m,6}+e_{3p}}, \end{aligned} \tag{10}$$

where the state variables e_{1p}, e_{2p}, and e_{3p} denote the dimensionless concentrations of the active form of the enzymes, and $K_{m,1-6}$, $V_{\max,1-6}$, and G are random parameters, along with a tuning

parameter $I \in [0, 1.5]$. For the 13 random parameters, we set their mean values as $K_{m,1-6} = 0.2$, $V_{\max,1} = 0.5$, $V_{\max,2} = 0.15$, $V_{\max,3} = 0.15$, $V_{\max,4} = 0.15$, $V_{\max,5} = 0.25$, $V_{\max,6} = 0.05$, and $G = 2$ and with $\pm 10\%$ uniform distribution around the mean values. In Fig. 4, we present the approximated mean and variance of the state variables. We observe that the DNN predictive model produces highly accurate predictions when compared to the truth for such long-time simulation.

2.3 Modeling of Non-autonomous Systems

Our next significant advancement is the modeling of non-autonomous system. The is an non-trivial extension of the FML framework for practical systems under external excitations. Let us consider

$$\begin{cases} \dfrac{d}{dt}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}, \gamma(t)), \\ \mathbf{x}(0) = \mathbf{x}_0, \end{cases} \tag{11}$$

where $\mathbf{x} \in \mathbb{R}^d$ are state variables and $\gamma(t)$ is a known time-dependent input. Two challenges are present for the modeling of such a system: (1) there is an explicit dependence on the time variable t; and (2) no matter how rich the training data set is, the excitation $\gamma(t)$ in the prediction can always be drastically different at a future time. Consequently, long-term system predictions are always outside the time domain and excitation signals covered by the training data. The straightforward learning methods will always fail after certain time during the prediction.

2.3.1 Local Parameterization and DNN Structure

To circumvent to challenges, we developed a localized learning method that is capable of prediction the solution for arbitrarily long time and under arbitrary excitation ([6]). The approach consists of the following steps: (1) parameterizing the excitation $\gamma(t)$ in the training data locally (in time); (2) decomposing the dynamical system into a modified system comprising of a sequence of local systems; and (3) learning of the parameterized local systems. More specifically, for each time interval $[t_n, t_{n+1}]$, $n = 0, \ldots, N-1$, with $\delta_n = t_{n+1} - t_n$,

$$\begin{aligned} \mathbf{x}(t_{n+1}) &= \mathbf{x}(t_n) + \int_{t_n}^{t_{n+1}} \mathbf{f}(\mathbf{x}(s), \gamma(s))ds \\ &= \mathbf{x}(t_n) + \int_0^{\delta_n} \mathbf{f}(\mathbf{x}(t_n + \tau), \gamma(t_n + \tau))d\tau. \end{aligned} \tag{12}$$

We then construct a local parameterization for the excitation locally $\gamma(t)$,

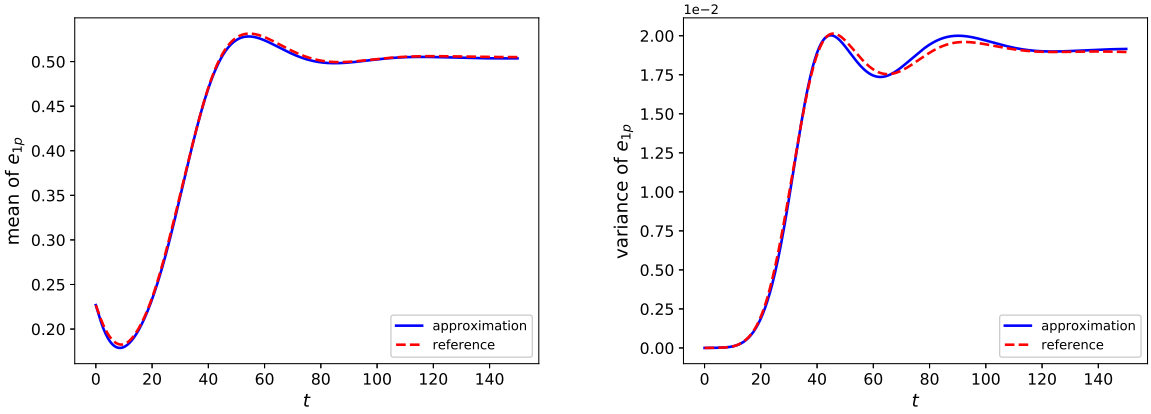
$$\widetilde{\gamma}_n(\tau; \mathbf{\Gamma}_n) := \sum_{j=1}^{n_b} \widehat{\gamma}_n^j b_j(\tau) \approx \gamma(t_n + \tau), \qquad \tau \in [0, \delta_n], \tag{13}$$

where $\{b_j(\tau), j = 1, \ldots, n_b\}$ is a set of prescribed analytical basis functions and

$$\mathbf{\Gamma}_n = (\widehat{\gamma}_n^1, \ldots, \widehat{\gamma}_n^{n_b}) \in \mathbb{R}^{n_b} \tag{14}$$

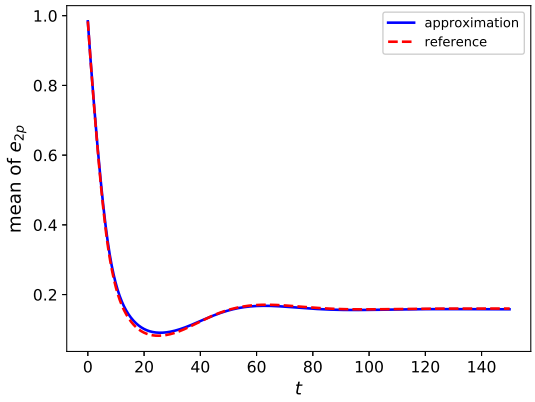
are the basis coefficients parameterizing the local input $\gamma(t)$ in $[t_n, t_{n+1}]$. Over one time step, many methods can be used to accurately parameterize γ, e.g., low-degree polynomials. We can then define a global parameterized input

$$\widetilde{\gamma}(t; \mathbf{\Gamma}) = \sum_{n=0}^{N-1} \widetilde{\gamma}_n(t - t_n; \mathbf{\Gamma}_n)\mathbb{I}_{[t_n, t_{n+1}]}(t), \tag{15}$$

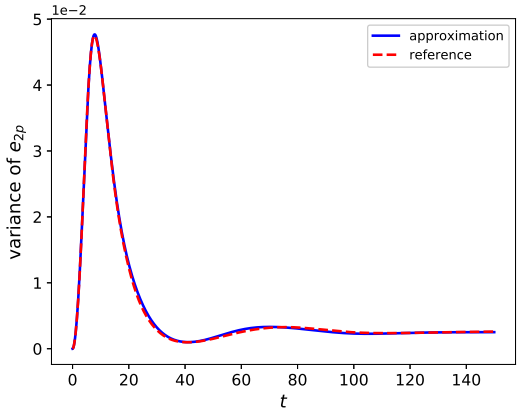


(a) mean of e_{1p}

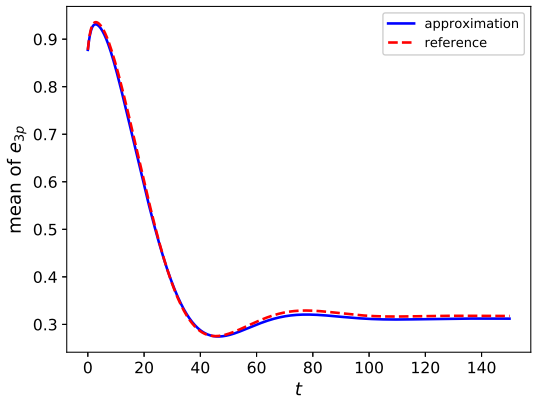
(b) variance of e_{1p}



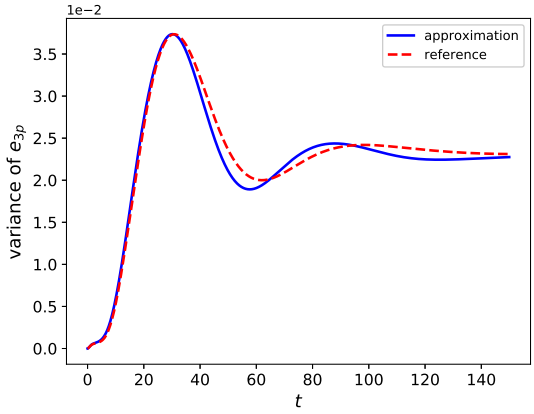
(c) mean of e_{2p}



(d) variance of e_{2p}



(e) mean of e_{3p}



(f) variance of e_{3p}

Fig. 4: Mean (left column) and variance (right column) of the predicted DNN results (blue), along with the truth from (10).

where

$$\mathbf{\Gamma} = \{\mathbf{\Gamma}_n\}_{n=0}^{N-1} \in \mathbb{R}^{N \times n_b} \tag{16}$$

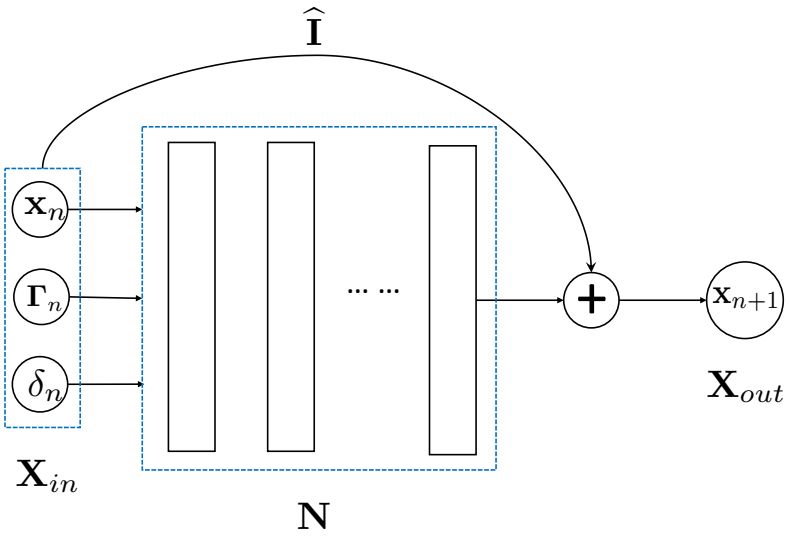


Fig. 5: DNN structure for non-autonomous systems.

is global parameter set for $\widetilde{\gamma}(t)$, and $\mathbb{I}_A$ is indicator function satisfying, for a set A, $\mathbb{I}_A(x) = 1$ if $x \in A$ and 0 otherwise.

We now define a modified system, corresponding to the true (unknown) system (11), as follows,

$$\begin{cases} \dfrac{d}{dt}\widetilde{\mathbf{x}}(t) = \mathbf{f}(\widetilde{\mathbf{x}}, \widetilde{\gamma}(t; \mathbf{\Gamma})), \\ \widetilde{\mathbf{x}}(0) = \mathbf{x}_0, \end{cases} \tag{17}$$

where $\widetilde{\gamma}(t; \mathbf{\Gamma})$ is the globally parameterized input defined in (15). Note that when the system input $\gamma(t)$ is already known or given in a parametric form, i.e. $\widetilde{\gamma}(t) = \gamma(t)$, the modified system (17) is equivalent to the original system (7). When the parameterized process $\widetilde{\gamma}(t)$ needs to be numerically constructed, the modified system (17) becomes an approximation to the true system (7). The approximation accuracy obviously depends on the accuracy in $\widetilde{\gamma}(t) \approx \gamma(t)$. For the modified system, the following results holds true for its solution evolution.

LEMMA: For the system (17), there exists a function $\widetilde{\boldsymbol{\phi}} : \mathbb{R}^d \times \mathbb{R}^{n_b} \times \mathbb{R} \to \mathbb{R}^d$, which depends on $\mathbf{f}$, such that for any time interval $[t_n, t_{n+1}]$, the solution of (17) satisfies

$$\widetilde{\mathbf{x}}(t_{n+1}) = \widetilde{\mathbf{x}}(t_n) + \widetilde{\boldsymbol{\phi}}(\widetilde{\mathbf{x}}(t_n), \mathbf{\Gamma}_n, \delta_n), \qquad n = 0, \ldots, N-1, \tag{18}$$

where $\delta_n = t_{n+1} - t_n$ and $\mathbf{\Gamma}_n$ is the local parameter set (14) for the locally parameterized input $\widetilde{\gamma}_n(t)$ (13).

This result thus indicates that a DNN model with a structure in Fig. 5 is able to learn the solution evolution.

2.3.2 Learned Model and System Prediction

Upon satisfactory training of the network parameter using the DNN in Fig. 5, we obtain a trained network model for the unknown modified system (17)

$$\mathbf{X}_{out} = \widehat{\mathbf{N}}(\mathbf{X}_{in}; \Theta^*) = [\widehat{\mathbf{I}} + \mathbf{N}(\cdot; \Theta^*)](\mathbf{X}_{in}). \tag{19}$$

For system prediction with a new excitation $\gamma(t)$, let

$$\mathbf{X}_{in} = [\mathbf{x}(t_n); \mathbf{\Gamma}_n; \delta_n]$$

be a concatenated vector consisting of the state variable at t_n, $\mathbf{\Gamma}_n$ the parameter vector for the local parameterization of the external input between $[t_n, t_{n+1}]$, and $\delta_n = t_{n+1} - t_n$. Then, the trained model produces a one-step evolution of the solution

$$\widehat{\mathbf{x}}(t_{n+1}) = \mathbf{x}(t_n) + \mathbf{N}(\mathbf{x}(t_n), \mathbf{\Gamma}_n, \delta_n). \tag{20}$$

Upon applying (20) recursively, we obtain a network model for predicting the system states of the unknown non-autonomous system (7). For a given initial condition $\mathbf{x}_0$ and external input $\gamma(t)$,

$$\begin{cases} \widehat{\mathbf{x}}(t_0) = \mathbf{x}_0, \\ \widehat{\mathbf{x}}(t_{n+1}) = \widehat{\mathbf{x}}(t_n) + \mathbf{N}(\widehat{\mathbf{x}}(t_n), \mathbf{\Gamma}_n, \delta_n), \\ t_{n+1} = t_n + \delta_n, \qquad n = 0, \ldots, N-1, \end{cases} \tag{21}$$

where $\mathbf{\Gamma}_n$ are the parameters in the local parameterization of $\gamma(t)$ in the time interval $[t_n, t_{n+1}]$.

2.3.3 Non-autonomous System Example: PDE Heat Equation with Source

We consider an unknown PDE – a heat equation with a source term,

$$\begin{aligned} &u_t = u_{xx} + q(t,x), \quad x \in [0,1], \\ &u(0,x) = u_0(x), \\ &u(t,0) = u(t,1) = 0, \end{aligned} \tag{22}$$

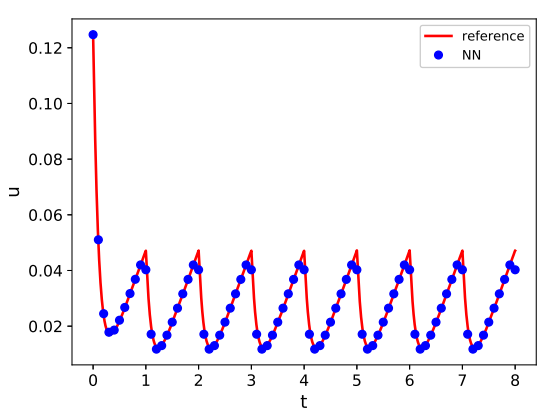
where $q(t,x)$ is the source term varying in both space and time and is set as

$$q(t,x) = \alpha(t) e^{-\frac{(x-\mu)^2}{\sigma^2}}.$$

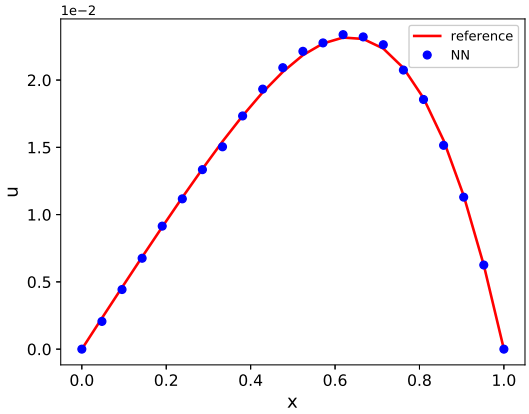
Here $\alpha(t)$ is its time varying amplitude and parameter μ and σ determine its the spatial profile. The DNN modeling of the system is shown in Fig. 6. We observe that the solution profiles and contours produced by the DNN model exhibit excellent accuracy against the true solutions.

2.4 Systems with Missing Variables

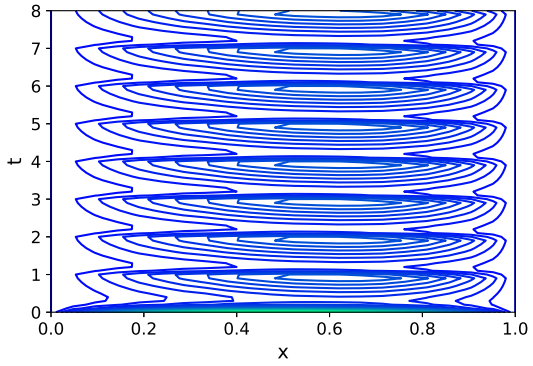
Our next major breakthrough is made on the front of modeling more practical systems [2]. That is, for many problems, one does not have data for all the system variables. Let $\mathbf{x} = (\mathbf{z}; \mathbf{w})$ be the state variables for the full system, where $\mathbf{z} \in \mathbb{R}^d$ is the subset of the state variables with available data, and $\mathbf{w} \in \mathbb{R}^{n-d}$ is the unobserved subset of the state variables. Our goal is to construct an effective model for the observed variables $\mathbf{z}$. Note that such a model does not exist mathematically, as the system is closed for the complete set of variables $\mathbf{x}$ but not for the subset $\mathbf{z}$. (Unless the $\mathbf{z}$ and $\mathbf{w}$ are decoupled, which results in a trivial system where $\mathbf{z}$ and $\mathbf{w}$ are independent from each other.)



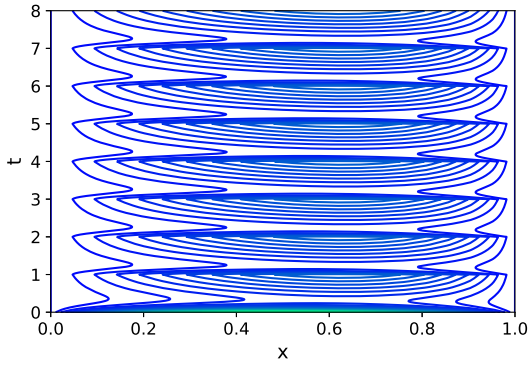
(a) Solution evolution at $x = 0.5$



(b) Solution profile at $t = 2$



(c) Reference solution contours over time



(d) DNN prediction contours over time

Fig. 6: Unknown heat equation with source: system prediction of (22) with $\alpha(t) = t - \lfloor t \rfloor$, $\mu = 1$, and $\sigma = 0.5$. Comparison between the predictions by the DNN model and the reference solution.

2.4.1 Discrete Approximate Mori-Zwanzig Formulation

The celebrated Mori-Zwanzig formulation ([4], [9]) states that a system of equations for the observables $\mathbf{z}$ exist, at least formally, in the form of the generalized Langevin equation,

$$\frac{d}{dt}\mathbf{z}(t) = \mathbf{R}(\mathbf{z}(t)) + \int_0^t \mathbf{K}(\mathbf{z}(t-s), s)ds + \mathbf{F}(t, \mathbf{x}_0). \tag{23}$$

The first term $\mathbf{R}$ depends only on the observables $\mathbf{z}$ at the current time and is Markovian. The second term, known as the memory, depends on $\mathbf{z}$ at all time, from the intial time $s = 0$ to the current time $s = t$, through a memory kernel $\mathbf{K}$. The last term is called orthogonal dynamics, which depends on the unknown initial condition of the entire variable $\mathbf{x}(0)$ and is treated as noise. Note that this formulation is an exact representation of the dynamics of the observed variables $\mathbf{z}$. The presence of the memory term makes the system non-autonomous and induces computational challenges. Note that the MZ equation exists only formally, as the terms $\mathbb{R}$ and $\mathbf{K}$ are unknown.

We make a basic assumption in the memory integral of the Mori-Zwanzig formulation (23). That is, we assume the memory kernel $\mathbf{K}$ decays over time, i.e.,

$$|\mathbf{K}(\mathbf{z}(t-s), s)| \searrow, \qquad \text{as } s \nearrow, \tag{24}$$

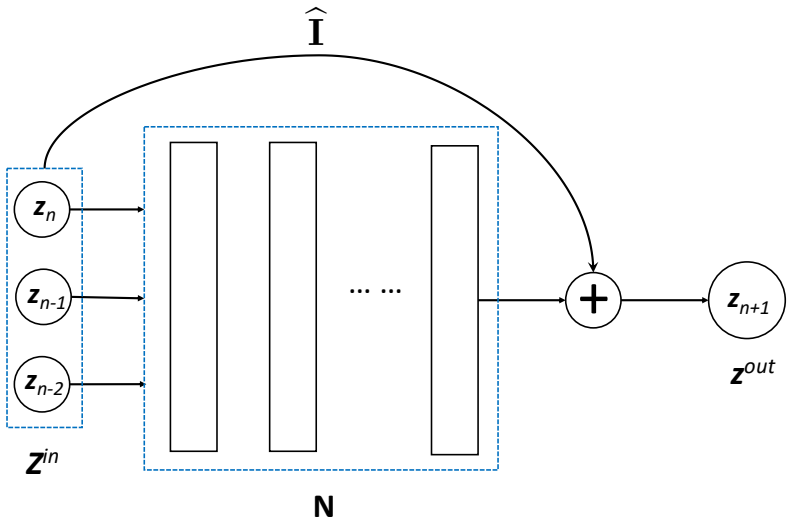


Fig. 7: Illustration of the memory based DNN for learning systems with missing variables. (Memory step $n_M = 2$ for illustration.)

and becomes negligible after an effective memory length $T_M > 0$. Consequently, we define an approximate Mori-Zwanzig (AMZ) system

$$\frac{d}{dt}\hat{\mathbf{z}}(t) = \mathbf{R}(\hat{\mathbf{z}}(t)) + \int_0^{T_M} \mathbf{K}(\hat{\mathbf{z}}(t-s), s)ds. \tag{25}$$

Moreover, we assume the finite integral can be approximated via a stencil, in the spirit of numerical integration. That is, let $\hat{\mathbf{z}}_n = \hat{\mathbf{z}}(t_n)$ be the solution at time $t_n = n\Delta$ over a constant time step Δ, there exists a function $\mathcal{M}$ such that

$$\left| \mathcal{M}(\hat{\mathbf{z}}_{n-n_M}, \ldots, \hat{\mathbf{z}}_{n-1}, \hat{\mathbf{z}}_n) - \int_0^{T_M} \mathbf{K}(\hat{\mathbf{z}}(t_n - s), s)ds \right| \leq \eta(t_n; T_M, n_M), \tag{26}$$

where $\eta \geq 0$ is the error. We then define a discrete approximate Mori-Zwanzig (d-AMZ) equation,

$$\frac{d}{dt}\widetilde{\mathbf{z}}(t)\bigg|_{t=t_n} = \mathbf{R}(\widetilde{\mathbf{z}}(t))|_{t=t_n} + \mathcal{M}(\widetilde{\mathbf{z}}_{n-n_M}, \ldots, \widetilde{\mathbf{z}}_{n-1}, \widetilde{\mathbf{z}}_n), \tag{27}$$

where $\mathcal{M}$ is defined in (26). Obviously, this is an approximation to the AMZ (25) at time level $t = t_n$. Although $\mathbf{R}$ and $\mathcal{M}$ on the right-hand-side remain unknown, the DAMZ effectively defines a discrete autonomous system with memory terms. A DNN structure in Fig. 7 thus becomes a direct realization of such a flow map. With sufficient trajectory data, such a DNN map can be readily trained and learned. The learned DNN thus defines a predictive model for the unknown dynamical system for the observed variables $\mathbf{z}$,

$$\begin{cases} \mathbf{z}_{n+1} = \mathbf{z}_n + \mathbf{N}(\mathbf{z}_n, \mathbf{z}_{n-1}, \ldots, \mathbf{z}_{n-n_M}), & n \geq n_M, \\ \mathbf{z}_n = \mathbf{z}(t_n), & n = 0, \ldots, n_M - 1. \end{cases} \tag{28}$$

With n_M initial data on $\mathbf{z}$, one can iteratively apply the network model to predict the evolution of $\mathbf{z}$ at later times.

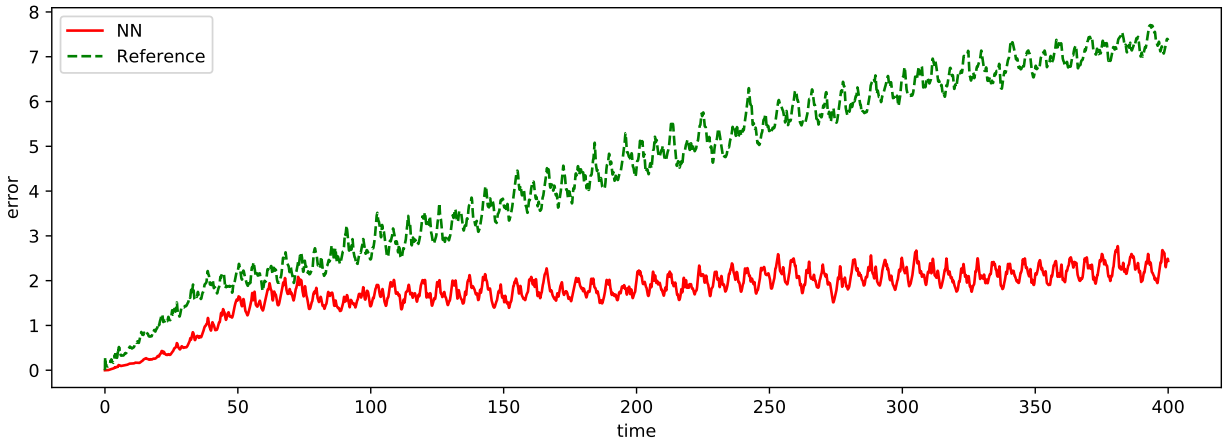


Fig. 8: Prediction errors for the slow variables in the chaotic system (29). The DNN model exhibits smaller and much more stable errors over the long-term prediction, than the homogenized system (30).

2.4.2 Learning Missing Variable System: Chaotic System

We now consider a nonlinear chaotic system

$$\begin{cases} \dot{x}_1 = -x_2 - x_3, \\ \dot{x}_2 = x_1 + \frac{1}{5}x_2, \\ \dot{x}_3 = \frac{1}{5} + y - 5x_3, \\ \dot{y} = -\frac{y}{\epsilon} + \frac{x_1 x_2}{\epsilon}, \end{cases} \tag{29}$$

where $\epsilon > 0$ is a small parameter. In this example, we choose the observed variables to be $\mathbf{z} = (x_1, x_2, x_3)^\top$, which are the slow variables of the system, and let the fast variable y be the unobserved variable. Note that for this system, there exists a homogenized system for the slow variables (x_1, x_2, x_3),

$$\begin{cases} \dot{x}_1 = -x_2 - x_3, \\ \dot{x}_2 = x_1 + \frac{1}{5}x_2, \\ \dot{x}_3 = \frac{1}{5} + x_3(x_1 - 5). \end{cases} \tag{30}$$

This homogenized system is a good approximation for the true system only when $\epsilon \ll 1$. Here we will construct NN models for the reduced variables $\mathbf{z}$ and compare the prediction results against the true solution of (29), as well as those obtained by the reduced system (30). We set $\epsilon = 0.01$, in which case the homogenized system (30) is considered an accurate approximation of the true system.

The results for long-term prediction of the chaotic system are shown in Fig. 8. We observe that the errors in the homogenized system, which is considered to be accurate, grow over time. Meanwhile, the errors by the DNN predictions are noticeably smaller, and more importantly, stay bounded and stable for the long-term prediction. This clearly demonstrate the accuracy and superiority of the DNN model. The long-term predictions of the state variables x_1 are shown in Fig. 9, for three arbitrarily chosen initial conditions. Again, we observe the DNN predictions are noticeably more accurate than the homogenized system (labeled as "Reference").

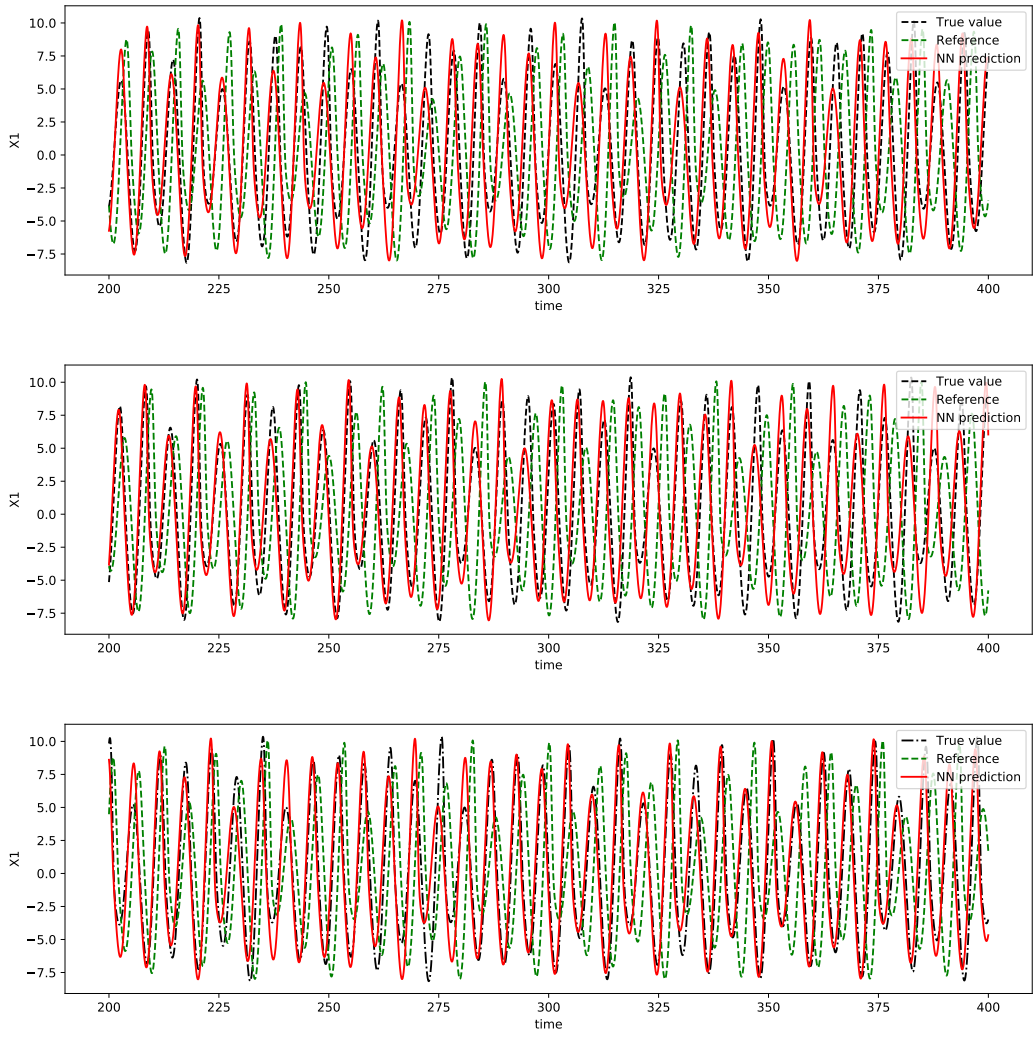


Fig. 9: Long-term model prediction of x_1 by the DNN model, the homogenized system (30) (labeled "Reference"), against the true solution. Zoomed view for $t \in [200, 400]$ with three aribitrarily chosen initial conditions.

2.5 Learning of PDE Systems

Yet another major extension of the FML methodology is its application to learning unknown PDEs. Let us consider an autonomous time-dependent PDE,

$$\begin{cases} u_t = \mathcal{L}(u), & (x,t) \in \Omega \times \mathbb{R}^+, \\ \mathcal{B}(u) = 0, & (x,t) \in \partial\Omega \times \mathbb{R}^+, \\ u(x,0) = u_0(x), & x \in \bar{\Omega}, \end{cases} \tag{31}$$

where $\Omega \subset \mathbb{R}^d$, $d = 1, 2, 3$, is the physical domain under consideration, and $\mathcal{L}$ and $\mathcal{B}$ stand for the PDE operator and boundary condition operator, respectively. Our basic assumption is that the PDE is unknown.

We assume data of the state variable u are available over a set of nodal points, or grids,

$$X_N = \{x_1, \ldots, x_N\} \subset \Omega,$$

and by using vector notiation, we write

$$\mathbf{u}(t) = (u(x_1, t), \ldots, u(x_N, t))^T.$$

Also, the data are only available at certain discrete time instances, resulting in so-called snapshots of the solution

$$\mathbf{u}(t_j^{(k)}), \qquad j = 1, \ldots, \ell^{(k)}, \quad k = 1, \ldots, N_{traj}.$$

Here the superscript k denotes the k-th "trajectory", which implies all $\ell^{(k)}$ snapshots are evolved from the same (unknown) initial state, and N_{traj} denotes the total number of such "trajectories". We then group the solution snapshots into pairs at two consecutive time instances,

$$(\mathbf{u}(t_j^{(k)}), \quad \mathbf{u}(t_{j+1}^{(k)})), \qquad j = 1, \ldots, \ell^{(k)} - 1, \quad k = 1, \ldots, N_{traj}.$$

Our goal is to construct an accurate approximation of the evolution/dynamics of the unknown governing equation (31) via the snapshot data. Once the approximation is constructed, it can serve as a predictive model to provide prediction and analysis of the unknown PDE system.

2.5.1 Learning in Modal Space

In [8], the flow-map based deep learning method was extended to PDE learning. The key is to conduct the learning in modal space, i.e., generalized Fourier space. Let $\{b_j(x),\, j = 1, \ldots, N_b\}$ be a basis in the physical domain Ω. The solution can be expressed as as a finite-term series

$$u(x, t) = \sum_{j=1}^{N_b} \hat{u}_j(t) b_j(x),$$

where $\hat{\mathbf{u}}(t) = [\hat{u}_1(t), \ldots, \hat{u}_{N_b}(t)]^T$ are the expansion coefficients. Thus there exists a system of ODEs for the expansion coefficients, in the form of

$$\frac{d\hat{\mathbf{u}}}{dt} = \mathbf{f}(\hat{\mathbf{u}}).$$

If the governing PDE equation is known, such a system for $\hat{\mathbf{u}}$ can be derived via a numerical approximation technique, e.g., Galerkin method. When the governing PDE is unknown, the system for $\hat{\mathbf{u}}$ is unknown as well. When solution data are available, this unknown ODE system can be learned in the modal space by extending flow-map based method from [7]. Details of the modal space PDE learning, including its proper mathematical formulation and data processing procedure, can be found in [8].

2.5.2 Learning in Physical Space

A more practical modeling scenario is to learn the unknown PDE when data are available in the physical space, as nodal values on a set of grids. The noval learning approach was developed in [1]. Without loss of generality, we consider a p-th order autonomous PDE in the following general form,

$$u_t = \mathcal{L}(u, \partial^{(1)} u, \ldots \partial^{(p)} u), \qquad p \geq 1, \tag{32}$$

where, for any $1 \leq m \leq p$,

$$\partial^{(m)} = \{\partial_{x_1}^{\alpha_1} \cdots \partial_{x_d}^{\alpha_d} : |\alpha| = m\}.$$

Note that for multi-dimensional PDE with $d > 1$, each $\partial^{(|\alpha|)}$ represents multiple partial derivative operators. Also, $u = \partial^{(0)} u$. Without considering boundary conditions, we assume the PDE (32) (1) is well-posed; and (2) has finite number $N_D \geq p$ of partial derivative terms.

We assume that when the PDE (32) is known, it can be approximated by a one-step Euler forward type explicit numerical scheme with sufficiently small local truncation error. More specifically, let

$$\mathbf{v}_N^{k+1} = \mathbf{v}_N^k + \Delta t \cdot \mathbf{\Psi}(\mathbf{v}_N^k), \qquad k = 0, \ldots, K-1, \tag{33}$$

be the numerical scheme, where $\mathbf{v}_N^k = (v(x_1, t^k), \ldots, v(x_N, t^k))^T$ is the numerical solution over X_N at time t^k, and $\mathbf{\Psi}$ is the incremental function.

PROPOSITION: Under the assumption that the PDE (32) admits the Euler forward explicit numerical method (33), there exists a set of functions $\{F_i : \mathbb{R}^N \to \mathbb{R}^N, i = 1, \ldots, J\}$, $J \geq 1$, and an iterative scheme

$$\mathbf{v}_N^{k+1} = \mathbf{v}_N^k + \Delta t \cdot \mathcal{M}\left[F_1(\mathbf{v}_N^k), \cdots, F_J(\mathbf{v}_N^k)\right], \qquad k = 0, \ldots, K-1, \tag{34}$$

where $\mathcal{M}$ is a (nonlinear) function operated component-by-component, such that for sufficiently large J, the exact solution of (32) satisfies

$$\mathbf{u}_N^{k+1} = \mathbf{u}_N^k + \Delta t \cdot \left(\mathcal{M}\left[F_1(\mathbf{u}_N^k), \cdots, F_J(\mathbf{u}_N^k)\right] + \boldsymbol{\eta}_N^{k+1}\right), \qquad k = 0, \ldots, K-1, \tag{35}$$

where $\|\boldsymbol{\eta}_N^{k+1}\| \leq \epsilon(X_N, \Delta t)$, the local truncation error of the Euler method (33), $\forall k$.

This result motivates a DNN structure for learning PDE in physical space. The DNN structure is illustrated in Fig. 10. It consists of the following components:

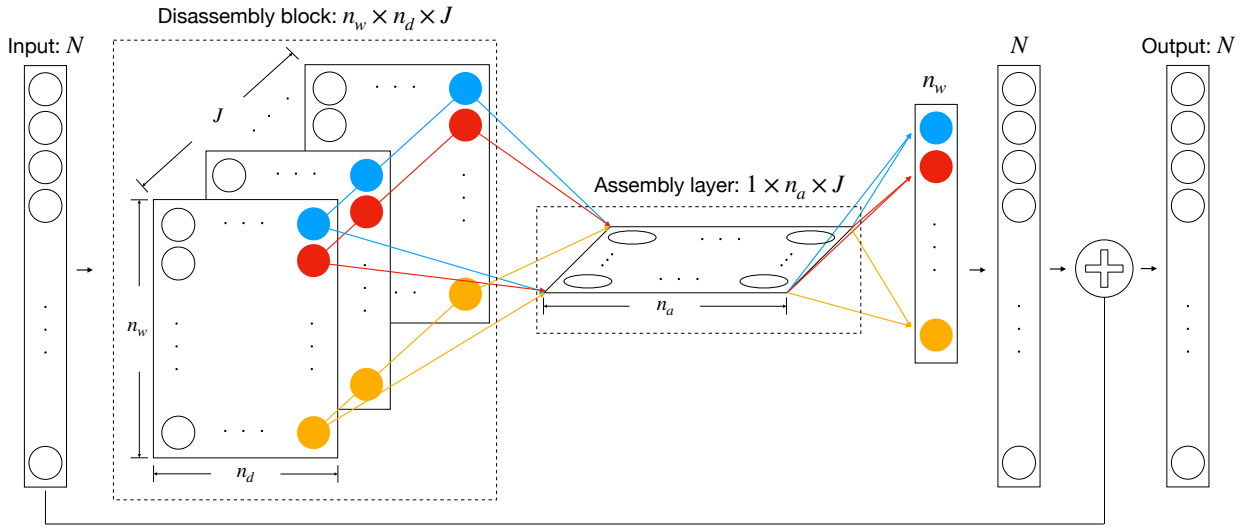


Fig. 10: The basic DNN structure.

- Input layer, where the number of neurons is N, the dimensionality of $\mathbf{u}$.

- Disassembly block. It has $J \geq 1$ fully connected FNNs "in parallel", where each FNN has width n_w and depth n_d and receives inputs from the input layer. As illustrated in Figure 10, the disassembly block thus creates a 3-dimensional tensor structure with dimension $n_w \times n_d \times J$, where J shall be referred to as the "thickness" of the disassembly block hereafter. The output layers of the disassembly block create a matrix structure of $n_w \times J$, spanning the width and thickness directions of the block.
- Assembly layer. It is a standard fully connected FNN, with width J and depth n_a. It operates along the thickness direction of the disassembly block. It receives input from one of the "rows" of the disassembly block output matrix, whose size is $n_w \times J$, and produces a scalar output. This is repeated for each of the n_w rows of the disassembly output matrix and produces an output vector of n_w. (In other words, one can also view the assembly layer as a block of n_w identical FNNs, i.e., with shared parameters, stacked vertically.)
- Output layer, where the number of neurons is N, the dimensionality of $\mathbf{u}$. The output layer is mapped from the output of the assembly layer, whose dimension is n_w. The input layer is re-introduced before the final output, in the same manner of residual network (ResNet).

Once trained, we obtain a predictive model over the grid set X_N for the underlying unknown PDE. For an arbitrarily given initial condition $\mathbf{u}_N(0)$ over the grid X_N, we have

$$\begin{cases} \mathbf{v}_N^0 = \mathbf{u}_N(0), \\ \mathbf{v}_N^{k+1} = \mathbf{N}(\mathbf{v}^k) = \mathbf{v}_N^k + \mathcal{F}\left[\mathbf{A}(\mathbf{N}_1(\mathbf{v}_N^k), \ldots, \mathbf{N}_J(\mathbf{v}_N^k))\right], \qquad k = 0, \ldots. \end{cases} \tag{36}$$

2.5.3 Inviscid Burgers' Equation

We now consider inviscid Burgers' equation with 2π-periodic boundary condition,

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x}\left(\frac{1}{2}u^2\right) = 0, \quad x \in [-\pi, \pi], \tag{37}$$

The training data are generated by solving the equation using a high-resolution numerical solver (Fourier collocation), by using arbitrary finite Fourier series and solving for one time step. Thus, all the training data are smooth functions. It is remarkably to emphasize that the learned DNN model, which never "sees" jump solution, produces accurate prediction and correctly generate shock solution at the proper time, as shown in Fig. 11.

2.5.4 2D PDE on Unstructured Grid

We now consider a two-dimensional advection-diffusion equation

$$\frac{\partial u}{\partial t} + \nabla \cdot (\boldsymbol{\alpha} u) = \nabla \cdot (\kappa \nabla u)$$

on an unstructured grid over domain $(x, y) \in [-1, 1] \times [-1, 1]$ with zero Dirichlet boundary condition. The transport velocity field is set as $\boldsymbol{\alpha}(x, y) = (y, -x)^T$, and the viscosity is set as $\kappa = 5 \times 10^{-3}$.

The unstructured grids are shown in Figure 12. They consist of 200 points in the interior $(-1, 1) \times (-1, 1)$, 8 points along each edge (resulting 32 points over the edges), and the 4 corner points. The interior points are generated using 2D Sobol sequence in $[0, \pi] \times [0, \pi]$, followed by cosine transformation. The edge points are generated by uniform random distribution in $(0, \pi)$, followed by cosine transformation.

The numerical prediction of the trained DNN model are shown in Fig. 13. We observe excellent agreement between the DNN predictions and the true solutions.

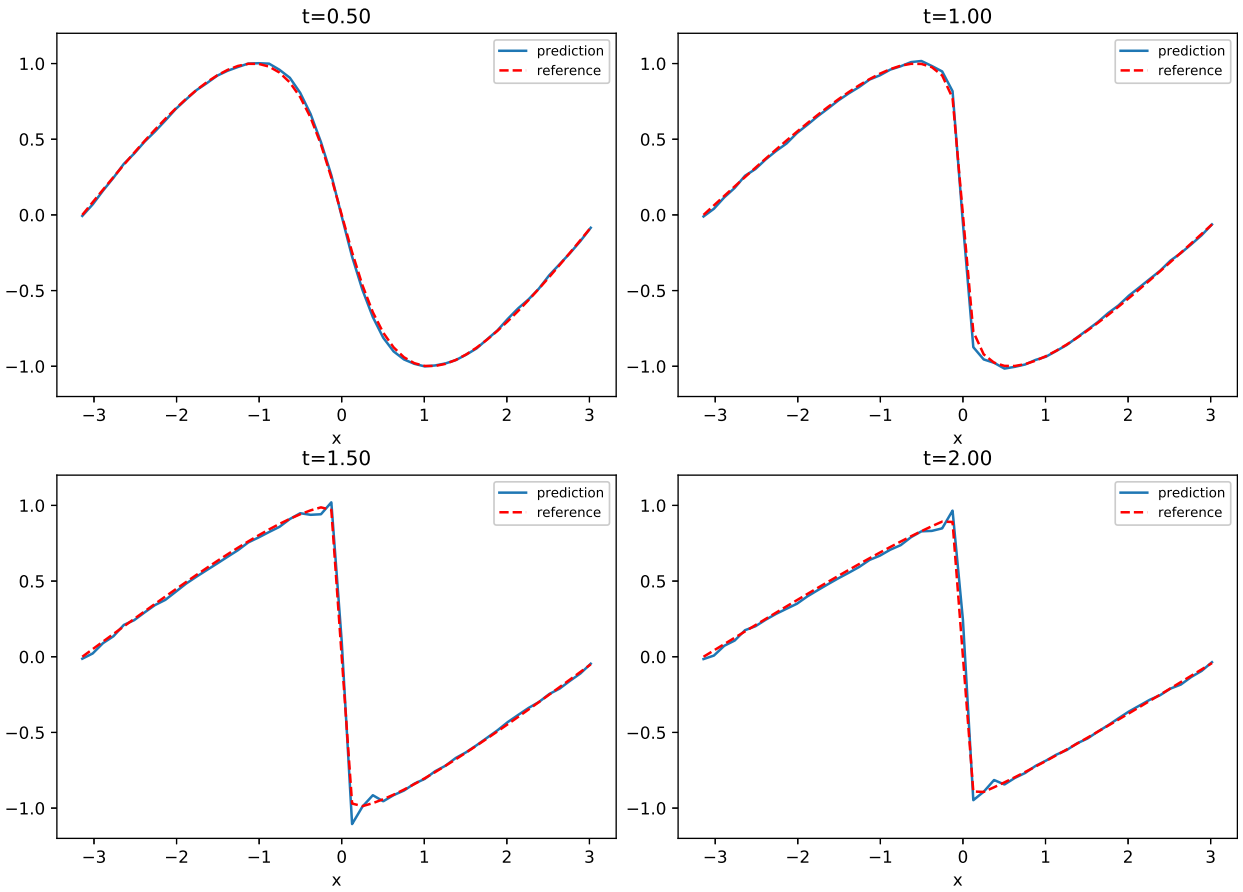


Fig. 11: Inviscid Burgers' equation. Comparison of DNN model prediction and reference solution. Left to right, up to bottom. Note in this case shock develops at $t = 2$.

3 Project Summary

The project resulted in a systematic development of data driven modeling of unknown systems – FML (flow map learning). The FML approach was established on a rigorous mathematical foundation and extended to learning and modeling of a set of progressively more practical situations. These include how to learn and model systems when only partial observation is available, when parameters are missing, and with unknown partial differential operators.

The project has partially funded two post-doc researchers and resulted in 20 high quality journal papers:

1. Z. Chen, V. Churchill, K. Wu and D. Xiu, "Deep Neural Network Modeling of Unknown Partial Differential Equations in Nodal Space", *Journal of Computational Physics*, Vol. 449, 110782, 2022.
2. S. Wang, Z. Zhou, L.-B. Chang and D. Xiu, "Construction of discontinuity detectors using convolutional neural networks", *Journal of Scientific Computing*, Vol. 91(40), 2022.
3. W.-H. Su, C.-S. Chou, and D. Xiu, "Deep Learning of Biological Models from Data: Applications to ODE Models", *Bulletin of Mathematical Biology*, Vol. 83 (19), 2021.
4. J. Hou, T. Qin, K. Wu and D. Xiu, "A Non-intrusive Correction Algorithm for Classification Problems with Corrupted Data", *Communications on Applied Mathematics and Computation*,

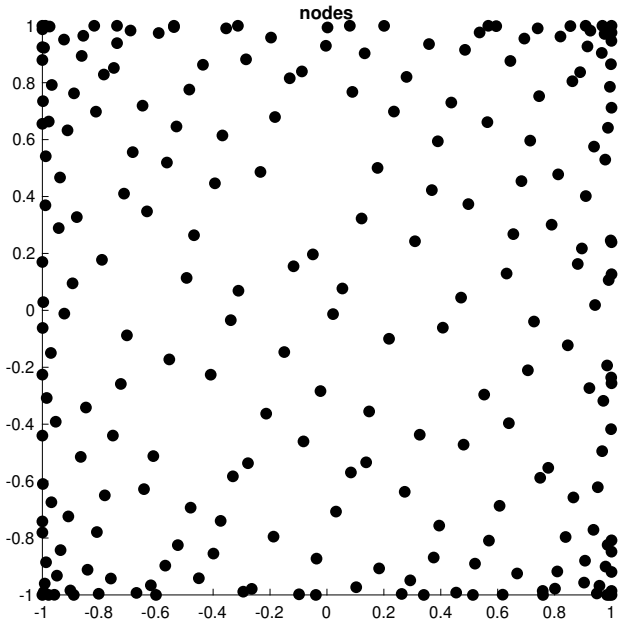


Fig. 12: 2D advection-diffusion. Unstructured grids.

Vol. 3(2), 337-356, 2021.

5. Z. Chen and D. Xiu, "On Generalized Residual Network for Deep Learning of Unknown Dynamical Systems", *Journal of Computational Physics'*, Vol. 438, 110362, 2021.
6. J. Hou, Y. Shin and D. Xiu, "Identification of Corrupted Data via k-Means Clustering for Function Approximation", *CSIAM Transactions for Applied Mathematics*, Vol. 2(1), 81-107, 2021.
7. T. Qin, Z. Chen, J. Jakeman and D. Xiu, "Data-driven Learning of Nonautonomous Systems", *SIAM Journal on Scientific Computing*, Vol. 43 (3), A1607-A1624, 2021.
8. W.-H. Su, C.-S. Chou and D. Xiu, "Deep Learning of Biological Models from Data: Applications to ODE Models", *Bulletin of Mathematical Biology'*, Vol. 83:19 2021.
9. T. Qin, Z. Chen, J. Jakeman and D. Xiu, "Deep Learning of Parameterized Equations with Applications to Uncertainty Quantification", *International Journal for Uncertainty Quantification*, Vol. 11, 63-82, 2021.
10. K. Wu, T. Qin and D. Xiu, "Structure-Preserving Method for Reconstructing Unknown Hamiltonian Systems from Trajectory Data", *SIAM Journal on Scientific Computing*, Vol. 42(6), A3704-3729, 2020.
11. Z. Chen, K. Wu and D. Xiu, "Methods to Recover Unknown Processes in Partial Differential Equations Using Data", *Journal of Scientific Computing*, Vol. 85, 23, 2020.
12. T. Qin, L. Zhou and D. Xiu, "Reducing Parameter Space for Neural Network Training", *Theoretical and Applied Mechanics Letters*, Vol. 10(3), 170-181, 2020.
13. K. Wu and D. Xiu, "Data-driven Deep Learning of Partial Differential Equations in Model Space", *Journal of Computational Physics*, Vol. 408, 109307, 2020.
14. J. Hou, T. Qin, K. Wu and D. Xiu, "A Non-intrusive Correction Algorithm for Classification Problems with Corrupted Data", *Communications on Applied Mathematics and Computation*, https://doi.org/10.1007/s42967-020-00084-4, 2020.
15. X. Fu, L.-B. Chang and D. Xiu, "Learning Reduced Systems via Deep Neural Networks with

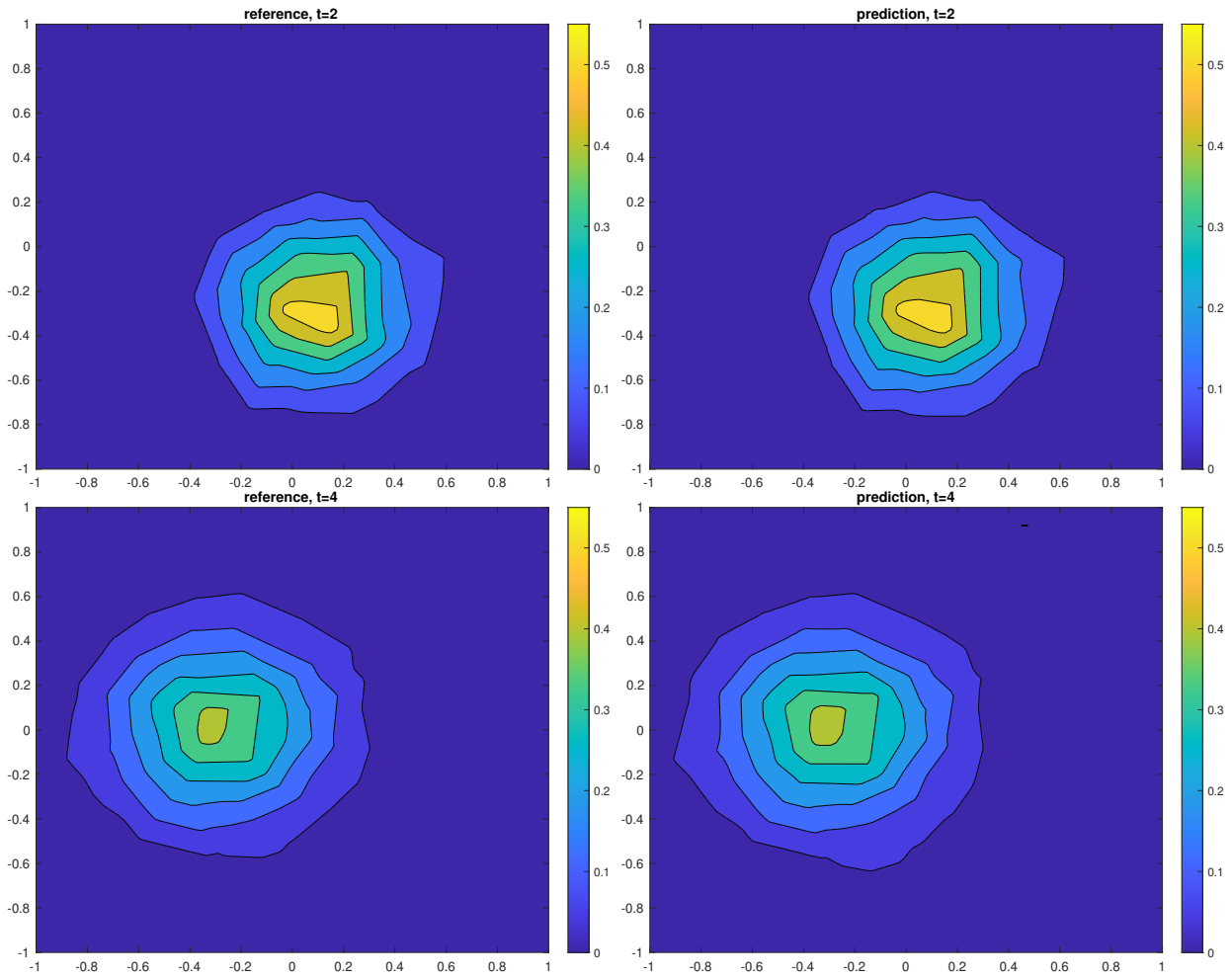


Fig. 13: 2D advection-diffusion over unstructured grids: Comparison of DNN model prediction (left column) and reference solution (right column).

Memory", *Journal of Machine Learning for Modeling and Computing*, Vol. 1(2), 97-118, 2020.

16. T. Qin, K. Wu and D. Xiu, "Data Driven Governing Equations Approximation Using Deep Neural Networks", *Journal of Computational Physics*, Vol. 395, 620-635, 2019.
17. C. Canuto, S. Pieraccini, and D. Xiu, "Uncertainty Quantification of Discontinuous Outputs via a Non-Intrusive Bifidelity Strategy", *Journal of Computational Physics*, Vol. 398, 108885, 2019.
18. K. Wu and D. Xiu, "Sequential Approximation of Functions in Sobolev Spaces Using Random Samples", *Communications on Applied Mathematics and Computation*, Vol. 1, 449-466, 2019.
19. K. Wu and D. Xiu, "Numerical Aspects for Approximating Governing Equations Using Data", *Journal of Computational Physics*, Vol. 384, 200-221, 2019.
20. Y. Shin, K. Wu and D. Xiu, "Sequential Function Approximation with Noisy Data", *Journal of Computational Physics*, Vol. 371, 363-381,2018.

References

[1] Zhen Chen, Victor Churchill, Kailiang Wu, and Dongbin Xiu. Deep neural network modeling of unknown partial differential equations in nodal space. *Journal of Computational Physics*, 449:110782, 2022.

[2] Xiaohan Fu, Lo-Bin Chang, and Dongbin Xiu. Learning reduced systems via deep neural networks with memory. *J. Machine Learning Model. Comput.*, 1(2):97–118, 2020.

[3] T. S. Gardner, C. R. Cantor, and J. J. Collins. Construction of a genetic toggle switch in escherichia coli. *Nature*, 403(6767):339, 2000.

[4] Hazime Mori. Transport, collective motion, and brownian motion. *Progress of theoretical physics*, 33(3):423–455, 1965.

[5] Tong Qin, Zhen Chen, John Jakeman, and Dongbin Xiu. Deep learning of parameterized equations with applications to uncertainty quantification. *Int. J. Uncertainty Quantification*, page 10.1615/Int.J.UncertaintyQuantification.2020034123, 2020.

[6] Tong Qin, Zhen Chen, John Jakeman, and Dongbin Xiu. Data-driven learning of non-autonomous systems. *SIAM J. Sci. Comput.*, 43(3):A1607–A1624, 2021.

[7] Tong Qin, Kailiang Wu, and Dongbin Xiu. Data driven governing equations approximation using deep neural networks. *J. Comput. Phys.*, 395:620 – 635, 2019.

[8] Kailiang Wu and Dongbin Xiu. Data-driven deep learning of partial differential equations in modal space. *J. Comput. Phys.*, 408:109307, 2020.

[9] Robert Zwanzig. Nonlinear generalized langevin equations. *Journal of Statistical Physics*, 9(3):215–220, 1973.