MACHINE LEARNING
Science and Technology

PAPER • OPEN ACCESS

Forecasting high-dimensional spatio-temporal systems from sparse measurements

To cite this article: Jialin Song *et al* 2024 *Mach. Learn.: Sci. Technol.* **5** 045067

IOP Publishing Mach. Learn.: Sci. Technol. 5 (2024) 045067 https://doi.org/10.1088/2632-2153/ad9883

MACHINE
LEARNING
Science and Technology

CrossMark

OPEN ACCESS

RECEIVED
26 August 2024

REVISED
5 November 2024

ACCEPTED FOR PUBLICATION
28 November 2024

PUBLISHED
18 December 2024

PAPER

Forecasting high-dimensional spatio-temporal systems from sparse measurements

Jialin Song[1,5], Zezheng Song[2,5,*], Pu Ren[3], N Benjamin Erichson[1,3], Michael W Mahoney[1,3,4] and Xiaoye S Li[3]

[1] International Computer Science Institute, Berkeley, CA 94704, United States of America
[2] Department of Mathematics, University of Maryland, College Park, MD 20742, United States of America
[3] Lawrence Berkeley National Laboratory, Berkeley, CA 94720, United States of America
[4] Department of Statistics, University of California, Berkeley, CA 94720, United States of America
[5] Equal contribution.
[*] Author to whom any correspondence should be addressed.

E-mail: zsong001@umd.edu

Keywords: spatio-temporal data, sparse measurements, vision transformers, dynamic modeling

Abstract

This paper introduces a new neural network architecture designed to forecast high-dimensional spatio-temporal data using only sparse measurements. The architecture uses a two-stage end-to-end framework that combines neural ordinary differential equations (NODEs) with vision transformers. Initially, our approach models the underlying dynamics of complex systems within a low-dimensional space; and then **it** reconstructs the corresponding high-dimensional spatial fields. Many traditional methods involve decoding high-dimensional spatial fields before modeling the dynamics, while some other methods use an encoder to transition from high-dimensional observations to a latent space for dynamic modeling. In contrast, our approach directly uses sparse measurements to model the dynamics, bypassing the need for an encoder. This direct approach simplifies the modeling process, reduces computational complexity, and enhances the efficiency and scalability of the method for large datasets. We demonstrate the effectiveness of our framework through applications to various spatio-temporal systems, including fluid flows and global weather patterns. Although sparse measurements have limitations, our experiments reveal that they are sufficient to forecast system dynamics accurately over long time horizons. Our results also indicate that the performance of our proposed method remains robust across different sensor placement strategies, with further improvements as the number of sensors increases. This robustness underscores the flexibility of our architecture, particularly in real-world scenarios where sensor data is often sparse and unevenly distributed.

1. Introduction

Understanding and predicting high-dimensional dynamical systems, such as atmospheric-ocean interactions, fluid dynamics, and seismic activities, is important across various scientific domains [1–3]. These systems are often studied through physics-based numerical simulations that generate complex, time-varying 2D or 3D spatial fields. However, real-world scenarios are often constrained by the limited ability to fully observe these high-dimensional fields, due to the sparse coverage of existing sensor technologies, thus leading to incomplete data acquisition [4]. For example, wave buoys in oceanography offer limited insights into surface dynamics, and sparse sensor networks in geoscience provide insufficient data on seismic activities. Consequently, reconstructing and forecasting these spatial fields from partial measurements is essential yet challenging [5, 6], as it involves solving an ill-posed inverse problem further complicated by unknown and nonlinear measurement operators.

Problem formulation. Since directly observing high-dimensional spatial fields that evolve over time is either challenging or costly we often rely on sensors that provide partial measurements of the system at specific time points. Our goal is to leverage these sparse sensor measurements to predict future spatial fields.

Specifically, given measurements $\mathbf{s}(t) \in \mathbb{R}^d$ at time t, where d denotes the number of sensors, we aim to infer the high-dimensional 2D spatial field $\mathbf{x}(t+\Delta t) \in \mathbb{R}^{m\times n}$ at a future time step $t+\Delta t$. Here, m and n represent the dimensions of the spatial field.

The challenge intensifies as the time step Δt increases, making accurate prediction of $\mathbf{x}(t+\Delta t)$ from the available measurements $\mathbf{s}(t)$ increasingly difficult. We seek to address this problem by developing a model that can infer the future spatial fields from limited measurements.

Our contributions. In this work, we aim to address this challenge by developing a new model for forecasting high-dimensional spatio-temporal systems from limited and irregularly distributed sensor measurements. This model operates at the intersection of two traditionally separate areas: (i) spatial reconstruction from low-dimensional spaces to high-dimensional fields; and (ii) temporal forecasting from historical data to future states. While recent works have attempted to combine forecasting and reconstruction (see section 2), we introduce a new approach.

We propose an end-to-end neural network (NN) architecture designed to forecast high-dimensional spatio-temporal data using sparse measurements. Our framework involves a two-step process: first, we use neural ordinary differential equations (NODEs) [7, 8] to model and forecast the system's dynamics in a low-dimensional space; and second, we employ vision transformers (ViTs) [9, 10] to reconstruct high-resolution, high-dimensional spatial fields.

We demonstrate that sparse measurements, despite their limitations, contain sufficient information to forecast the system's dynamics over a desired time horizon. Furthermore, we show that these forecasts can be used to accurately reconstruct future high-dimensional spatial fields. This approach leverages the efficiency of modeling dynamics in a lower-dimensional space, circumventing the need to directly handle high-dimensional data. Extensive experiments on various spatio-temporal systems, including fluid flows and weather data, validate the effectiveness of our approach and suggest that sparse measurements are sufficient to model the dynamics of complex systems.

Organization. The rest of the paper is organized as follows. We discuss related work on the reconstruction and forecasting of spatio-temporal systems in section 2. Our methodology for the end-to-end framework for forecasting and reconstruction is discussed in section 3: we first present details for the NODE model used for forecasting sparse measurements; and we then provide details on the ViT architecture used for reconstructing the high-dimensional quantities of interest. Section 4 presents our empirical results for a simple and challenging fluid flow examples and for real-world weather data; and section 5 provides a brief conclusion.

2. Related work

In the following, we discuss related work on (i) reconstructing spatial fields from limited measurements; (ii) forecasting spatio-temporal data; and (iii) recent works that simultaneously forecast and reconstruct spatio-temporal systems.

Spatial reconstruction. Spatial reconstruction is a task that involves recovering spatial fields from partial or sparse observations, and it is always regarded as a challenging and ill-posed inverse problem. Reconstruction scenarios include both irregular and regular sampling methods. Conventional techniques are centered around linear methods, such as basis expansion methods (e.g. POD) [11, 12] and matrix factorization [13]. With advancements of computer vision, this problem has garnered significant attention across various scientific fields by leveraging the nonlinearity of NNs, e.g. with full-field reconstruction from sparse sensors [5, 14, 15], super-resolution of scientific data [3, 16–19], and downscaling of global climate data [20]. Specifically, in the context of spatial super-resolution, many researchers leverage state-of-the-art ViTs [21, 22] and diffusion models [23, 24] for scientific data reconstruction.

Temporal forecasting. Temporal forecasting is a fundamental problem focused on identifying the underlying dynamic patterns from observations and establishing a predictive mapping between historical records and the future states of dynamical systems. The classic autoregressive scheme and state-space models are based on linear assumptions, offering reliable accuracy for short-term forecasting. Deep learning techniques, such as recurrent NNs (RNNs) [25, 26] and their variants [27, 28], have shown significant promise in capturing complex temporal dependencies and nonlinear dynamics. The RNN family can be further extended to convolutional forms for spatiotemporal forecasting [29, 30]. More recently, NODEs [7, 31] have emerged as a powerful paradigm for modeling temporal dynamics, as they aim to combine deep learning with continuous-time dynamical systems. NODEs have improved the interpretability and efficiency of time-series tools, especially in scenarios involving irregularly sampled data [32] or naturally continuous dynamics [33].

Furthermore, due to their remarkable representation capabilities, especially in computer vision tasks, transformers [34–39] and diffusion models [40, 41] have increasingly been explored for time-series forecasting. These powerful models have been adapted for spatio-temporal forecasting in various domains, such as weather and climate systems [42–48], highlighting their potential in capturing complex temporal and spatial dependencies.

Forecasting and Reconstruction. Work that combines reconstruction and forecasting within a unified framework aims to tackle the challenge of incomplete data and long-horizon predictions. Simultaneous reconstruction and forecasting aims to introduce a new opportunity to understand complex natural phenomena. Similar to spatial reconstruction, the methodologies address two types of low-dimensional spaces: (i) irregular sparse sensor measurements; and (ii) regular low-resolution data. In the context of sparse measurements, the sequential encoder–decoder model [30] and INR methods have been explored for both reconstruction and forecasting [49, 50]. Moreover, shallow recurrent decoder networks [51, 52], which can handle both recurrence and decoding with sparse sensor measurements, have been introduced as a follow-up to the Shallow Decoder [5]. For scenarios involving regular low-resolution data, spatio-temporal super-resolution techniques [53, 54] have been considered for augmenting various scientific data. A recent work [55] combines variational autoencoder and transformer models to capture dynamics in the latent space, and this has shown great potential for diverse scientific applications, such as weather forecasting and structural engineering.

3. End-to-end framework for forecasting and reconstruction

We propose to model the temporal evolution and the spatial reconstruction jointly within an end-to-end framework. Conceptually, as illustrated in figure 1, our framework includes two stages:

- **Stage 1: Modelling dynamics.** In the first stage, a Gated NODE (GNODE) architecture [56] is used for forecasting the dynamics given a measurement vector $\mathbf{s}_0$ as an initial condition. To be more concrete, the GNODE models the dynamics as

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{s}(t) = \mathcal{F}(\mathbf{s}(t);\boldsymbol{\theta}) := g\left(\mathbf{s}(t),\boldsymbol{\theta}_g\right) \odot \left[\mathcal{N}(\mathbf{s}(t);\boldsymbol{\theta}_n) - \mathbf{s}(t)\right], \quad \text{with} \quad \mathbf{s}(0) = \mathbf{s}_0, \tag{1}$$

  where $\mathcal{N}$ is a network parameterized by $\boldsymbol{\theta}_n$, and g is a gate parameterized by $\boldsymbol{\theta}_g$. The details can be found in section 3.1.
- **Stage 2: Reconstructing spatial fields.** In the second stage, we are concerned with reconstructing the high-dimensional spatial fields $\mathbf{x}(t+\Delta t)$ from the predicted measurements $\hat{\mathbf{s}}(t+\Delta t)$ at a future time step $t+\Delta t$ as

$$\hat{\mathbf{x}}(t) = \mathcal{G}(\hat{\mathbf{s}}(t);\boldsymbol{\phi}), \tag{2}$$

  where $\mathcal{G}:\mathbb{R}^d \to \mathbb{R}^{m\times n}$ is a decoder module parameterized by $\boldsymbol{\phi}$. Specifically, we leverage the Swin ViT architecture [9] in combination with an additional fully-connected up-sample layer, and a pixel shuffle layer [57] to capture multi-scale features and recover the spatial fields. The details can be found in section 3.2.

To train this framework, we use a standard supervised learning approach to optimize the learnable parameters of $\mathcal{N}$ and $\mathcal{G}$. To do so, we require sequences with input-output pairs of measurement vectors and spatial fields $\{\mathbf{s}_n,\mathbf{x}_n\}_{0,\ldots,N}$, where $\mathbf{s}_n,\mathbf{x}_n$ represent $\mathbf{s}(t_n),\mathbf{x}(t_n)$ that are discretized along the time points $t_n = n\Delta t$ for $n = 0,1,\ldots,N$. Then, we can jointly minimize the dynamics and reconstruction error using the following objective function for prediction:

$$\mathcal{N}^*,\mathcal{G}^* = \underset{\boldsymbol{\theta},\boldsymbol{\phi}}{\arg\min}\,\frac{1}{N}\sum_{n=0}^{N-1}\left[\|\mathbf{s}_{n+1} - \mathtt{ODEint}(\mathcal{F}(\mathbf{s}_n;\Delta t,\boldsymbol{\theta}))\|_2^2 + \lambda\|\mathbf{x}_{n+1} - \mathcal{G}(\mathbf{s}_{n+1};\boldsymbol{\phi})\|_F^2\right],$$

where N denotes the number of data points, $\lambda > 0$ balances the loss terms, and $\mathtt{ODEint}(\cdot)$ denotes a numerical integration method. Here $\mathcal{F}$ denotes a GNODE, and we use a leaky integrator scheme for discretizing the model for training and inference. In practice, we train the model with a multi-step prediction loss to improve the forecast horizon during inference time.

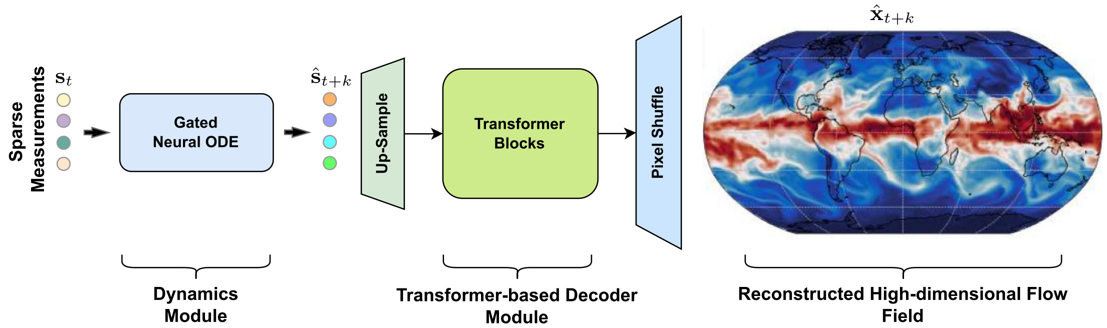

Figure 1. Illustration of the proposed end-to-end architecture for forecasting and reconstructing spatio-temporal data from sparse measurements. Given a measurement vector $\mathbf{s}_t$, the Gated NODE model is used for forecasting the dynamics. The predicted measurement vector $\hat{\mathbf{s}}_{t+k}$ is then transformed into a latent image, which in turn is decoded by a vision transformer. A final pixel shuffle layer is refining the resolution of the reconstructed flow field $\hat{\mathbf{x}}_{t+k}$.

3.1. Stage 1: neural ODEs for modeling dynamics

NODEs are lightweight models that are versatile for a range of applications in machine learning and science [7, 33, 58–64]. The idea of these models is to use a NN $\mathcal{N}$ to parameterize the vector field of an ODE

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{s}(t) = \mathcal{N}(\mathbf{s}(t);\boldsymbol{\theta}), \tag{3}$$

where $\mathbf{s}$ denotes the state vector. This formulation is interesting, because it connects NNs with dynamical systems theory [33].

Given a set of training data and an initial value $\mathbf{s}(0) = \mathbf{s}_0$, we can learn the parameters $\boldsymbol{\theta}$ of the network via back-propagation by evaluating the following integral equation

$$\mathbf{s}_{n+1} = \mathbf{s}_n + \int_{t_n}^{t_n+\Delta t} \mathcal{N}(\mathbf{s}(\xi);\boldsymbol{\theta})\,\mathrm{d}\xi, \tag{4}$$

where $\mathbf{s}_n = \mathbf{s}(t_n)$, and Δt is the discrete timestep. In practice, we use numerical schemes to approximate the integral. For example, the simple forward Euler discretization scheme leads to $\mathbf{s}_{n+1} = \mathbf{s}_n + \Delta t\,\mathcal{N}(\mathbf{s}_n;\boldsymbol{\theta})$. It is straightforward to implement this discrete model in any framework for deep learning, which in turn allows us to learn the parameters through gradient-based algorithms. The particular discretization scheme that is used to derive the discrete model has a significant impact on the model's performance [33]. For instance, the forward Euler method is easy and flexible to implement, but it offers lower accuracy, whereas higher-order methods (e.g. Runge–Kutta fourth-order) provide greater accuracy and convergence performance. Analyzing numerical schemes, and the design of novel solvers for ODEs is an active area of research [65–68].

A shortcoming of NODEs is their limited expressiveness for modeling complex dynamics. For instance, trajectories described by ODEs cannot cross each other, which induces an inductive bias [31]. While this is favorable in situations where it is known that the underlying dynamics are described by an ODE, it can be limited in other situations. One approach to improve the expressiveness of NODEs is to augment the vector space by lifting the state vectors into a higher dimension [31]. Other approaches to improve expressiveness include gating [56], time delayed feedback [69, 70], and the use of second-order ODE system [71].

In this work, we consider gating as a simple mechanism to improve the expressiveness of a standard NODE. Gating essentially introduces multiple scales into the model, which enables it to approximate more complex dynamics [28, 70, 72]. For instance, a single scale could be introduced into equation (3) via a simple time constant $\tau \in [0,1]$, leading to the following model

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{s}(t) = \tau\cdot\mathcal{N}(\mathbf{s}(t);\boldsymbol{\theta}). \tag{5}$$

Intuitively, this scale will control the dynamics: at one extreme, if $\tau = 1$, the dynamics modeled by $\mathcal{N}$ are not affected; but, on the other hand, if $\tau = 0$, then the dynamics are fully damped.

The idea of a GNODE is to introduce multiple time scales by replacing τ with a d-dimensional vector. This vector can be assumed to be dependent on $\mathbf{s}(t)$. To this end, we introduce a gating function $g:\mathbb{R}^d \to [0,1]^d$ into equation (3) so that we yield the following continuous-time Gated model

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{s}(t) = g(\mathbf{s}(t);\boldsymbol{\theta}_g)\odot\mathcal{N}(\mathbf{s}(t);\boldsymbol{\theta}), \tag{6}$$

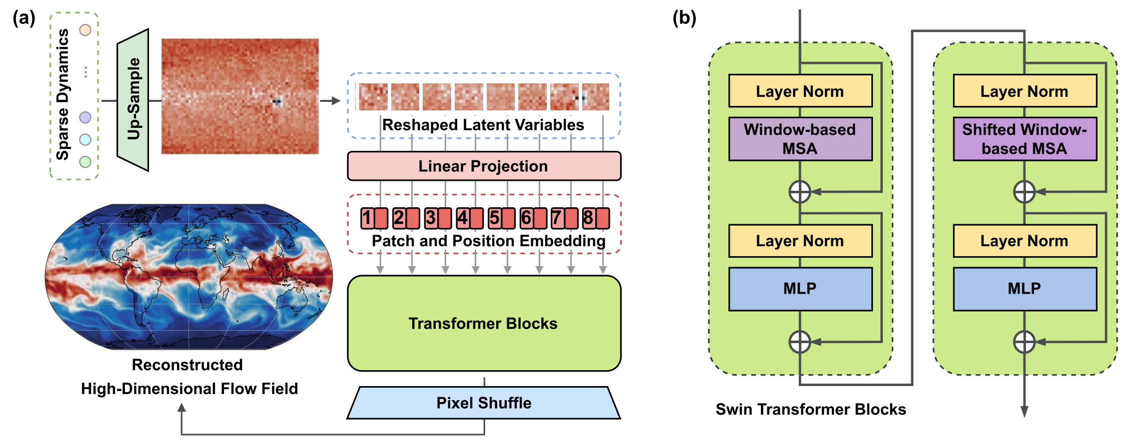

Figure 2. (a) illustrates an up-sample layer and the ViT architecture that are used for spatial reconstruction from sparse dynamics. (b) presents details of the residual swin transformer block.

where $\odot$ denotes the Hardamard product. The gating function is parameterized by a NN with learnable weights $\boldsymbol{\theta}_g$. Again, the simple forward Euler discretization scheme leads to

$$\mathbf{s}_{n+1} = \mathbf{s}_n + \Delta t\, g\left(\mathbf{s}_n; \boldsymbol{\theta}_g\right) \odot \mathcal{N}\left(\mathbf{s}_n; \boldsymbol{\theta}\right).$$

The literature also proposes alternative formulations that lead to different discretized models. For instance, [56] proposed the following GNODE

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{s}(t) = \mathcal{F} := g\left(\mathbf{s}(t), \boldsymbol{\theta}_g\right) \odot \left[\mathcal{N}\left(\mathbf{s}(t); \boldsymbol{\theta}\right) - \mathbf{s}(t)\right]. \tag{7}$$

Setting $\Delta t = 1$, and applying the forward Euler scheme yields the following discretized model

$$\mathbf{s}_{n+1} = (1-g) \odot \mathbf{s}_n + g \odot \mathcal{N}\left(\mathbf{s}_n; \boldsymbol{\theta}\right),$$

where $g := g(\mathbf{s}_n; \boldsymbol{\theta}_g)$. This model is also known as a leaky integrator [73]. The formulation in equation (7) is popular in the RNN literature [28, 70, 72], and we found that this formulation outperforms the model stated in equation (6).

The choice of the nonlinear activation function plays a crucial role in the performance of deep learning models since it significantly affects the model's capability to learn complex patterns. In this work, we replace the traditional ReLU function in the NODE component with a rational activation function [74], as this offers better flexibility and efficiency. The rational activation function is formulated with trainable parameters a_i and b_j,

$$F(x) = \frac{P(x)}{Q(x)} = \frac{\sum_{i=0}^{r_P} a_i x^i}{\sum_{j=0}^{r_Q} b_j x^j}, \quad a_P \neq 0, \quad b_Q \neq 0, \tag{8}$$

where r_P and r_Q denote the polynomial degrees of the numerator and denominator, respectively. In particular, this activation function is well-suited for learning non-smooth and highly oscillatory systems. We conduct an ablation study in section 4.7 to evaluate its effectiveness, compared to the standard ReLU function.

3.2. Stage 2: Transformer-based decoder for reconstructing flow fields

This section introduces a decoder $\mathcal{G}$, based on a Swin ViT architecture [9], for reconstructing the high-dimensional flow field $\mathbf{x}(t+\Delta t)$, given the predicted variable of sparse dynamics $\hat{\mathbf{s}}(t+\Delta t)$. The proposed architecture is illustrated in figure 2(a). Firstly, to leverage a ViT model for our problem, we need to learn a mapping from the sparse measurement vector $\hat{\mathbf{s}}(t+\Delta t) \in \mathbb{R}^d$ to a higher-dimensional latent variable $\mathbf{Z}(t+\Delta t) \in \mathbb{R}^{m' \times n'}$ with a reshaped dimension of $m' \times n'$. Specifically, the up-sampling layer is constructed with multilayer perceptrons (MLPs) to obtain $\mathbf{Z}$.

Then, we create a sequence $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_M\}$ by splitting the input $\mathbf{Z}$ into $M = \frac{m' \times n'}{p^2}$ fixed-size patches of dimension $p \times p$, which we flatten into vectors $\mathbf{z}_i \in \mathbb{R}^{p^2}$. We subsequently embed each element of the sequence in a D-dimensional embedding space through a linear mapping $\mathbf{z}_i' = \mathbf{W}\mathbf{z}_i \in \mathbb{R}^D$, where $\mathbf{W}$ is a weight matrix. In addition, a positional embedding scheme is used to encode information about the order of the embedded patches.

The embedded features are passed through a series of transformer blocks to obtain a final latent summary which is then used for reconstructing the high-dimensional flow field. A standard transformer block consists of a self-attention (SA) module [75], an MLP, and several normalization layers [76]. Given the input matrix $\mathbf{Z} \in \mathbb{R}^{N\times D} := [\mathbf{z}_1, \ldots, \mathbf{z}_i, \ldots, \mathbf{z}_N]^\top$, where the rows are the stacked embedded patches, the SA module computes a transformed matrix $\boldsymbol{Y} \in \mathbb{R}^{N\times D'}$ as follows:

$$\boldsymbol{Y} = \mathrm{Attention}\left(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}\right) := \mathrm{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{D}}\right)\boldsymbol{V}. \tag{9}$$

Here, the query, key, and value matrices $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V} \in \mathbb{R}^{N\times D'}$ are calculated from the inputs. Specifically, we obtain the query, key, and value matrices as:

$$\boldsymbol{Q} = \boldsymbol{Z}\boldsymbol{W}_q, \quad \boldsymbol{K} = \boldsymbol{Z}\boldsymbol{W}_k, \quad \boldsymbol{V} = \boldsymbol{Z}\boldsymbol{W}_v, \tag{10}$$

where $\boldsymbol{W}_q, \boldsymbol{W}_k, \boldsymbol{W}_v \in \mathbb{R}^{D\times D'}$ are learnable weight matrices. Note that the softmax layer is applied row-wise to the scaled attention matrix $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{K}^T \in \mathbb{R}^{N\times N}$. It is known that scaling the $\boldsymbol{A}$ prevents small gradients [75]. The output $\boldsymbol{Y}$ of the SA module is a weighted sum of the values $\boldsymbol{V}$.

While the simple SA mechanism is effective for computer vision tasks, recent studies have demonstrated that the shifted window (Swin) transformer block [9] is better suited for image restoration tasks [10] and super-resolution of scientific problems [18]. The swin transformer architecture presents hierarchical, shifted windows for SA mechanisms, and it helps capture both local and global patterns efficiently. This design not only mitigates computational challenges associated with processing large images, but it also enhances the model's capacity to capture fine-grained features, making it suitable for spatial reconstruction of multi-scale dynamics.

The success of the swin transformer architecture lies in the swin transformer layers (STLs) and their variant residual swin transformer blocks (RSTBs). STLs are the fundamental building blocks of the Swin transformer, and they consist of a local window-based SA layer, a global feature fusion module, and a feed-forward network. The window-based SA mechanism allows STLs to focus on local relationships between image patches, reducing computational complexity and improving efficiency. The global feature fusion module combines features from different window sizes, capturing both local and global contexts, while the feed-forward network further enhances feature representation.

RSTBs incorporate residual connections in the STL architecture, as illustrated in figure 2(b). This stabilizes information flow and alleviates the vanishing gradient problem. An RSTB consists of two STLs connected in a residual manner, where the output of the second STL is added to the input of the first STL. This residual connection allows for direct propagation of information from the input to the output, facilitating gradient flow and improving training stability.

Given a decoded low-resolution flow field, we use pixel shuffle [57] to obtain a refined high-resolution reconstruction. Pixel shuffle, which is based on a sub-pixel convolution layer, rearranges the elements with a periodic shuffling operator to convert the low-resolution feature maps to a high-resolution output.

4. Numerical examples

In this section, we aim to show the effectiveness of our method in learning spatio-temporal dynamics in the context of reconstruction and forecasting. To do so, we evaluate our proposed framework on three datasets, ranging from fluid flows to weather dynamics. We investigate the effects of various sampling methods, and we show that our method can capture the spatio-temporal dynamics accurately from sparse measurements with appropriate sensor placement strategies. We also conduct an ablation study on different types of NODE models and activation functions, which validates the superior performance of GNODEs and the Rational activation function.

4.1. Sensor placement strategies for sparse measurements

First of all, it is worthwhile to mention that the experimental setup in our paper differs from the standard encoder–decoder frameworks. Specifically, we assume that only sparse measurement data is accessible during training and inference. Encoders or sampling methods serve more as preprocessing steps for the datasets, instead of being one part of the network training itself. In our test, we experiment with two naive sampling methods, random and uniform;—and we also use a discrete empirical interpolation method (DEIM) [77], which is an efficient approach for reduced-order modeling of nonlinear dynamical systems [78–80]. With

Algorithm 1. DEIM [77].

Input: The left singular vectors $\mathbf{W}$ of snapshot matrix $\mathbf{A}$.
Output: $\mathbf{S} = [\mathbf{e}_{i_1}, \cdots, \mathbf{e}_{i_s}] \in \mathbb{R}^{n \times s}$.
$[|\rho|, t_1] = \mathtt{max}\{|\mathbf{W}_1|\}$
$\mathbf{W} = [\mathbf{W}_1], \mathbf{S} = [\mathbf{e}_{i_1}]$,
for ℓ from 2 to s **do**
  Solve $\mathbf{S}^T\mathbf{W}\mathbf{c} = \mathbf{S}^T\mathbf{W}_\ell$ for $\mathbf{c}$
  $\mathbf{r} = \mathbf{W}_\ell - \mathbf{W}\mathbf{c}$
  $[|\rho|, t_\ell] = \mathtt{max}\{|\mathbf{r}|\}$
  $\mathbf{W} \leftarrow [\mathbf{W} \quad \mathbf{W}_\ell], \mathbf{S} \leftarrow [\mathbf{S} \quad \mathbf{e}_{i_\ell}]$
end for
return $\mathbf{S}$ with the selected indices $\{i_1, \ldots, i_s\}$.

DEIM, we can approximate these nonlinear terms by considering only a few selected points in the domain, effectively sampling the high-dimensional states $\mathbf{x}_t$. Furthermore, the points selected by DEIM for interpolation inherently serve as an indication of where the most significant part of the system is observed. Therefore, these points can be interpreted as optimal sensor or measurement locations in the reduced-dimensional space. In practice, placing sensors at these locations captures important dynamics of the original high-dimensional system.

The DEIM sampling method consists of two major steps: (i) constructing a basis for the low-dimensional space that captures most of the 'energy' of the original high dimensional states $\mathbf{x}_t$; and (ii) selecting the index from the basis matrix with the DEIM algorithm. See algorithm 1. The indices selected from the second step will serve as the locations of the sensor placement in our work.

To be more specific, in the offline phase, we collect snapshots of the full-order dynamical system

$$\mathbf{A} = [\mathbf{x}_1, \ldots, \mathbf{x}_T], \tag{11}$$

where $\mathbf{x}_i \in \mathbb{R}^n$ is the ith flattened spatio-temporal snapshot. To determine spatial measurements of interest, we first use the compact singular value decomposition to factorize the snapshot matrix $\mathbf{A} \in \mathbb{R}^{n \times T}$. We denote the top r left-singular vectors by $\mathbf{W} \in \mathbb{R}^{n \times r}$, where r is the target rank. Then, the DEIM algorithm selects s distinct rows from $\mathbf{W}$. Suppose the row indices are $I = \{i_1, \ldots, i_s\}$. Then the corresponding selection operator is

$$\mathbf{S} = [\mathbf{e}_{i_1}, \cdots, \mathbf{e}_{i_s}] \in \mathbb{R}^{n \times s},$$

where $\mathbf{e}_{i_j}$ is the i_j-th column of the $n \times n$ identity matrix. Therefore, $\mathbf{S}$ chooses the measurement locations of state $\mathbf{x}_t$, i.e. $\mathbf{s}_t = \mathbf{S}^T\mathbf{x}_t$ (which will serve as the low-dimensional state in the online phase later). Algorithm 1 shows the detailed procedure. Given the columns of matrix $\mathbf{W} = [\mathbf{W}_1, \ldots, \mathbf{W}_r]$, DEIM works by determining the first index $i_1 \in \{i_1, \ldots, i_s\}$ by selecting the index of the entry with the largest magnitude in the first column $\mathbf{W}_1$. Then, iteratively, the subsequent indices are selected to maximize the residual between the next basis vector and the linear combination of the previous basis vectors.

We also tested variants of DEIM, including Q-DEIM [81] (QR factorization based) and R-DEIM [82] (randomized version). We found that both of them are comparable to the original DEIM within our setting. Therefore, we will not consider them further in this paper.

4.2. Implementation of our framework

We use the Adam optimizer [83] with the weight decay of 1×10^{-6} and the momentum parameter set to (0.9, 0.999). The Adam algorithm is a stochastic gradient-based optimization scheme widely used in machine learning. The Exponential learning scheduler is employed with $\gamma = 0.995$ over 400 training epochs. The network hyper-parameters and the training settings are determined empirically and vary across different numerical cases. The details can be found in table 1. All the numerical implementations are coded in Pytorch [84] and performed on an NVIDIA A100 GPU card (40G memory) in a standard workstation.

To evaluate the accuracy, we consider the ℓ_2 relative error (L2RE) [85], which is formulated as

$$\text{L2RE} = \frac{1}{n}\sum_{i=1}^{n}\frac{\|\mathbf{x}_i - \widehat{\mathbf{x}}_i\|_2}{\|\mathbf{x}_i\|_2} \times 100\%,$$

where $\mathbf{x}_i$ and $\widehat{\mathbf{x}}_i$ are the ground truth and reconstruction in high-dimensional states, respectively. n is the total number of spatial grids with a height and width of $h \times w$ for each sample.

Table 1. Implementation settings varied by datasets.

Dataset	Widths (NODE)	Depths (NODE)	Window size (Decoder)	Epochs	Learning rate	Batch size	Rollout steps (Training)
Fluid	64	3	2	400	8.5×10^{-4}	4	15
RBC	256	4	8	400	4×10^{-4}	16	10
Climate	256	4	9	400	5×10^{-5}	8	8

4.3. Fluid flow behind the cylinder

In this subsection, we investigate the performance of our model on fluid flow passing a cylinder [86], a benchmark problem [3, 5] to understand the fluid dynamics. This problem is governed by two-dimensional incompressible Navier–Stokes (NS) equations, and it exhibits complex and diverse fluid structures, such as periodically shedding wake patterns. It is formulated as

$$\begin{aligned} &\nabla\cdot\mathbf{u}=0, \\ &\frac{\partial\mathbf{u}}{\partial t}=-\mathbf{u}\cdot\nabla\mathbf{u}-\nabla p+\frac{1}{\mathrm{Re}}\nabla^2\mathbf{u}, \end{aligned} \tag{12}$$

where $\mathbf{u}$ and p denote the velocity field and pressure, respectively. The numerical data is obtained using the direct numerical simulation at Reynolds number $\mathrm{Re}=100$ with the immersed boundary projection method [87, 88]. Similar to the implementation in a Shallow Decoder [5], we select 151 cropped snapshots with a spatial resolution of 199×384. This captures numerous vortex-shedding cycles and discards the spatial sub-domain upstream of the cylinder. The dataset is then split into training and testing sets, with the first 100 snapshots used for training, and the remaining 51 snapshots for validation. In addition, we apply mean and-standard deviation normalization to the snapshots to ensure a consistent input scale for the model across the time span. The snapshots are then padded to dimensions of 200×384 to meet the window size $l=2$ requirement in the swin transformer decoder, as shown in table 1. Moreover, we set the number of roll-out steps in training as $T=15$ due to the tradeoff between inference performance and the training stability [89, 90]. A larger T enhances the accuracy and stability during inference for long trajectories but destabilizes the training process.

First, we study the impact of different sensor placements and varying numbers of chosen sensors in terms of reconstruction accuracy. Specifically, we consider random, uniform, and DEIM sampling methods for sensor placements. The number of sensors covers a range of $\{4,8,16,32\}$. We show a representative comparison in figure 3, where the number of sensors $n_s=16$ and the prediction time step $T=15$. The first row of figure 3 displays the ground truth snapshots and the corresponding sensor placements, where DEIM shows superior performance in placing the sensors around the critical dynamics, compared with random and uniform sampling schemes. The second and third rows show the reconstruction and the absolute error contours between the ground truth and the reconstructed snapshots, respectively. The error contour based on the DEIM scheme exhibits fewer mismatches distributed in the spatial domain, thanks to its specifically designed sampling strategy.

Second, we conduct a quantitative analysis of the reconstruction and forecasting accuracy, as shown in figures 4 and 5. Specifically, figure 4 presents the box plots of reconstruction errors by running five random seeds. Overall, using more sensors facilitates training convergence and yields more accurate recovered dynamics for all three sensor placement schemes. For random and uniform sampling methods, the uncertainty of the reconstructions increases drastically when fewer sensors are chosen, indicating their unstable performance of learning high-dimensional dynamics from sparse measurements. The DEIM strategy exhibits the best performance regarding reconstruction errors. It is noteworthy that when the numbers of sensors are small, far from capturing the rank of the dynamic system, there are fluctuations in the reconstruction error, i.e. the error is not monotonically decreasing with increasing sensors. Once the number of sensors approaches the rank of the dynamical system, the error stabilizes, meaning that further increasing the number of sensors does not lead to significant improvement. The empirical results are consistent with the theoretical guarantee of DEIM, which is expected to achieve ideal accuracy for handling nonlinear systems.

In figure 5, we present the test reconstruction/forecasting error given an increasing number of prediction steps under various sampling schemes. We train all the models with the time step $T=15$, and we calculate the forecasting results from prediction step $T=2$ to $T=29$. The reconstruction error maintains a similar level from time step $T=2$ to $T=15$, followed by the increment after time step $T=15$.

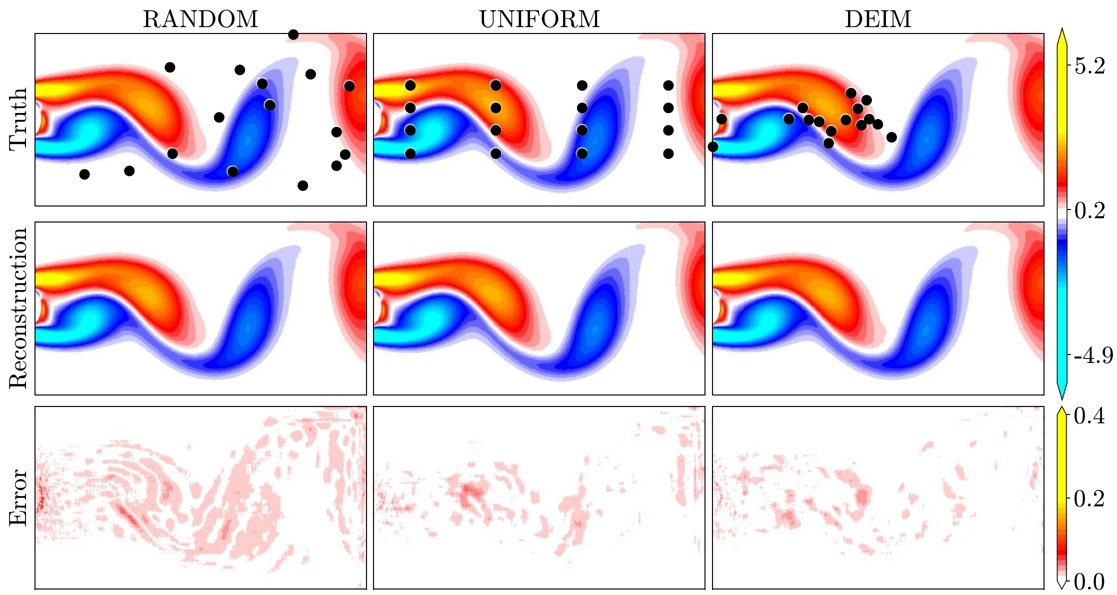

Figure 3. Reconstruction results of fluid flow snapshot with random, uniform, and DEIM sampling. The number of sensors n_s and the prediction time step T are set as 16 and 15, respectively. The first row shows the ground truth snapshots and the corresponding sensor placements. The second and third rows display the reconstructed snapshots and the absolute error contours. The black dots represent sensors.

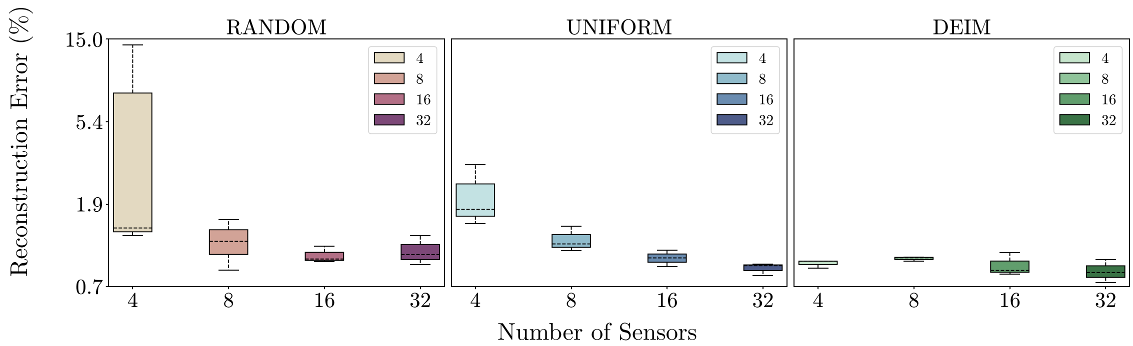

Figure 4. Box plots of relative reconstruction error for the models trained on fluid flow data with random, uniform, and DEIM sampling schemes and with various numbers of sensors. All results are obtained on 5 different random seeds. The reconstruction errors are plotted on a logarithmic scale.

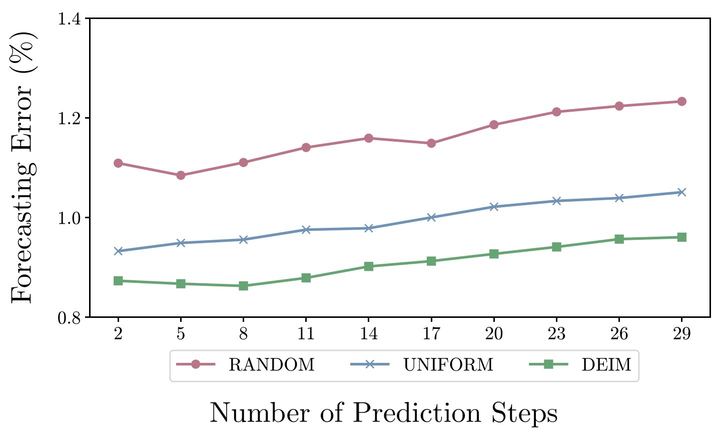

Figure 5. A representative error propagation of model inference on fluid cylinder dataset. We define 15 steps in the training and leverage 16 sensors with random, uniform, and DEIM sampling schemes. Each line shows relative forecasting errors with an increasing number of prediction steps.

4.4. Rayleigh-Bénard convection (RBC) system

In this subsection, we evaluate our model on a 2D RBC system, known for its nonlinear and chaotic behavior. The RBC model represents the flow of a fluid heated from below and cooled from above, involving intricate interactions between velocity, pressure, and temperature. This system has been studied in various scientific fields, including geophysics, meteorology, and oceanography. Specifically, the RBC system is described by a set of governing equations, which can be expressed as follows:

$$\begin{aligned} &\nabla \cdot \mathbf{u} = \mathbf{0}, \\ &\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = -\nabla p + T\mathbf{e}_x + R^* \nabla^2 \mathbf{u}, \\ &\frac{\partial T}{\partial t} + (\mathbf{u} \cdot \nabla)\, T = P^* \nabla^2 T. \end{aligned} \tag{13}$$

The velocity components in the x and y directions are denoted by $\mathbf{u} = u, v$, while p and T represent the pressure and temperature terms, respectively. The system is characterized by dimensionless Rayleigh (R_a) and Prandtl (P_r) numbers, given by the expressions $g\alpha(\Delta T)h^3/(\nu\kappa)$ and ν/κ. Herein, $g, \alpha, \Delta T, h, \nu, \kappa$ represent gravity acceleration, thermal expansion coefficient, the temperature difference between the top and bottom walls, the length between the plates, the kinematic viscosity, and the heat conductivity coefficient, respectively. Additionally, R^* and P^* are defined as $(P_r/R_a)^{0.5}$ and $(R_aP_r)^{-0.5}$, respectively. The unit vector in the x direction is represented by $\mathbf{e}_x$. We use the Dedalus solver [91] to simulate the ground truth data. We define the spatial domain as $[0,4] \times [0,1]$, which is discretized into a grid of size 512×128. The simulation spans a time period of $[0,40]$ with 4000 time steps. To learn the stabilized dynamics of the system, we extract the dataset from the later stage of the simulation from the $[20,40]$ time interval. The 2000 time steps are further separated into two sets, including a training set with the initial 1800 time steps and a testing set with the remaining 200 steps. The sets are then cropped into the dimension of 128×128 to ensure that the sensor placement from the sampling schemes remains in the region of interest. Additionally, we preprocess the data with mean-standard deviation normalization.

The RBC system presents more complex dynamical behaviors than the fluid flow behind a cylinder. Therefore, we use relatively more sensors for three sampling schemes to assess the model performance, i.e. we choose $\{16, 32, 64, 80\}$. An illustrative comparison is shown in figure 6, where the numbers of sensors and roll-out steps are chosen as 64 and 10, respectively. The figure includes the visualizations of sensor placements based on different sampling strategies and the recovered snapshots. Similar to the observation in section 4.3, DEIM demonstrates a better sensor placement by effectively capturing system dynamics and achieves less error compared to random and uniform sampling approaches in this scenario. Furthermore, figure 7 exhibits a more comprehensive analysis of reconstruction performance across various sensor placement scenarios. Generally, a larger n_s leads to improved reconstruction accuracy and reduced uncertainty. DEIM consistently outperforms the other sampling methods in minimizing the reconstruction error.

We also evaluate the performance of error propagation on the RBC dataset. All the models are trained with a time step $T = 10$ and 64 sensors are sampled across the spatial domain using three distinct strategies. We test the inference performance of those models by varying the prediction step from $T = 5$ to $T = 23$. As shown in figure 8, the models consistently achieve low forecasting errors within the designated prediction time step $T = 10$, while the errors tend to increase when the time step exceeds this set value.

4.5. Weather data

In this subsection, we conduct tests on a high-resolution weather dataset considered in `SuperBench` [18], which is specifically modified for data reconstruction tasks from the global climate dataset `ERA5` [92]. `ERA5` is a reanalysis dataset produced by the European Centre for Medium-Range Weather Forecasts, and it poses unique challenges for dynamical prediction and spatial upsampling. Its complexity arises from the highly intricate interactions among various physical processes, such as multi-scale fluid turbulence and radiation/heat transfer, across the atmosphere, ocean, and land surfaces. The `ERA5` dataset considers a global scale with a spatial resolution of 0.25° (25 km) in latitude and longitude, resulting in a 720×1440 pixel field when represented on a Cartesian grid. It also provides a broad temporal span, ranging from the year 1979 to the present day on an hourly basis. The dataset employs advanced data assimilation techniques [93], integrating diverse observational data with numerical weather prediction models to produce a consistent and precise historical record of weather conditions. In this example, we focus on the temperature and vapor variables for evaluation. The spatial resolution of the weather dataset is downsampled to 180×360 to facilitate the training process and reduce the computational memory. We select 9-year data (from the year 2005 to the year 2014) and resample the dataset on a daily basis, where the training and testing data is split with a ratio of 7 : 3. Specifically, we consider the DEIM sampling scheme with 360 sensors for both

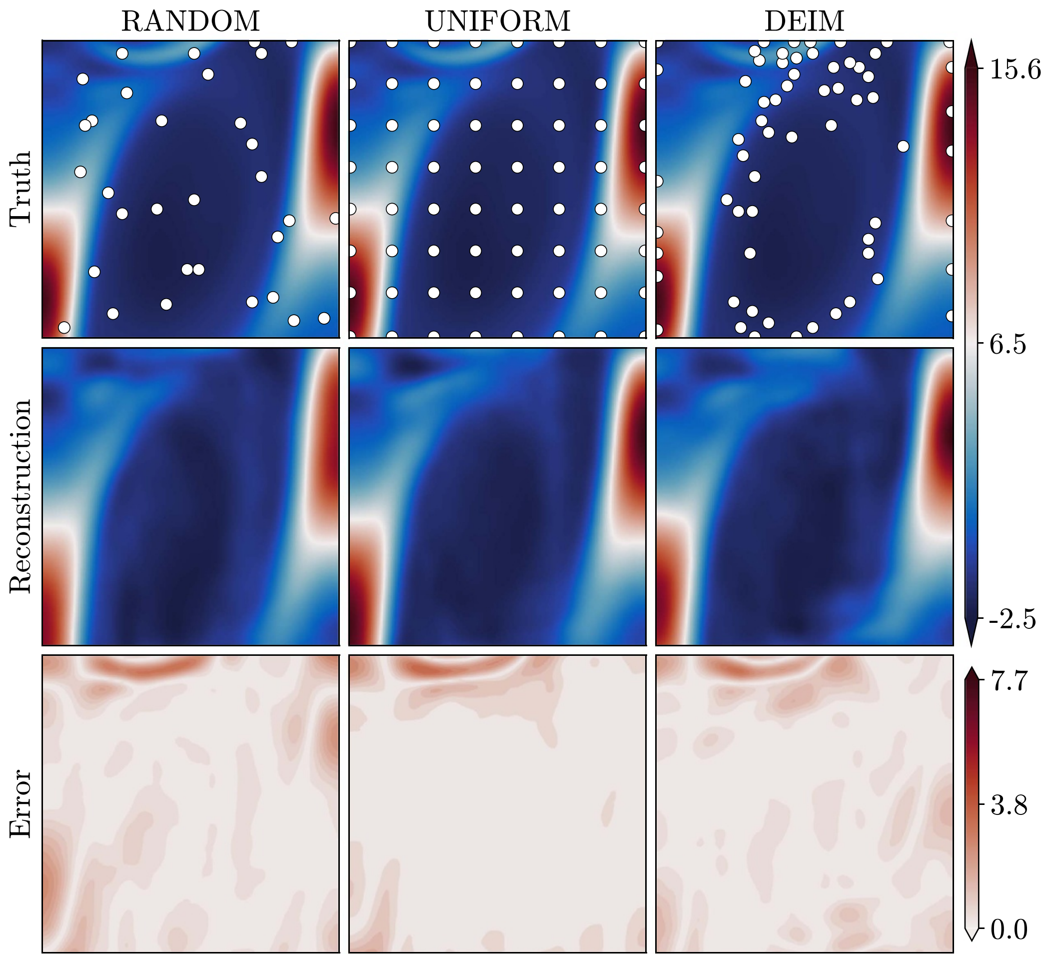

Figure 6. Reconstruction results of RBC data snapshot with random, uniform, and DEIM schemes. The number of sensors and the prediction time step are set as $n_s = 64$ and $T = 10$, respectively. The first row shows the ground truth snapshots and the corresponding sensor placements. The second and third rows display the reconstructed snapshots and the absolute error contours. The white dots represent the placed sensors.

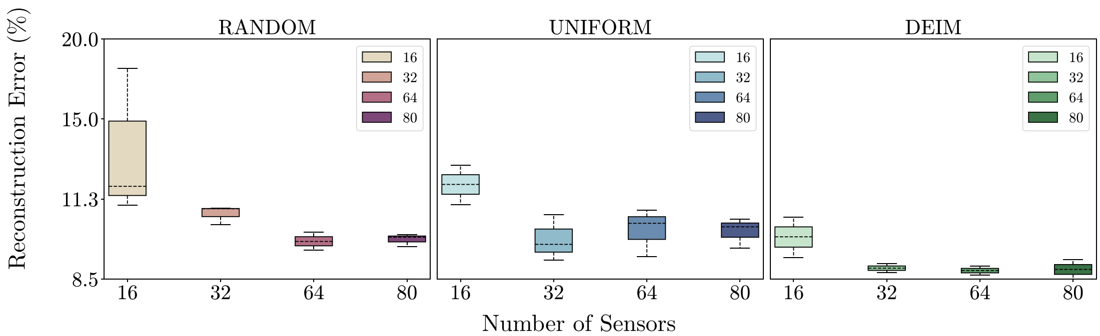

Figure 7. Box plots of relative reconstruction error for the models trained on RBC data with prediction time step $T = 10$. Three sampling schemes are implemented with various numbers of sensors. All the results are obtained by running five different random seeds. The reconstruction errors are plotted on a logarithmic scale.

temperature and vapor data, due to the performance of DEIM for sensor placement, as validated in the above two fluid cases. The number of sensors is large, compared to sections 4.3 and 4.4, due to the complexity of this weather dataset. The rollout step in training is set to 8.

Figure 9 visualizes the sensor placements (360 sensors) projected onto the global domain, where the sensors are distributed in the regions with abruptly changed dynamics or multi-scale features for temperature and vapor variables. It is evident that DEIM effectively positions sensors to capture the weather dynamics. Figure 10 illustrates the performance of our proposed framework for learning the spatio-temporal dynamics of temperature and vapor variables. It can be seen that our model can capture both global and

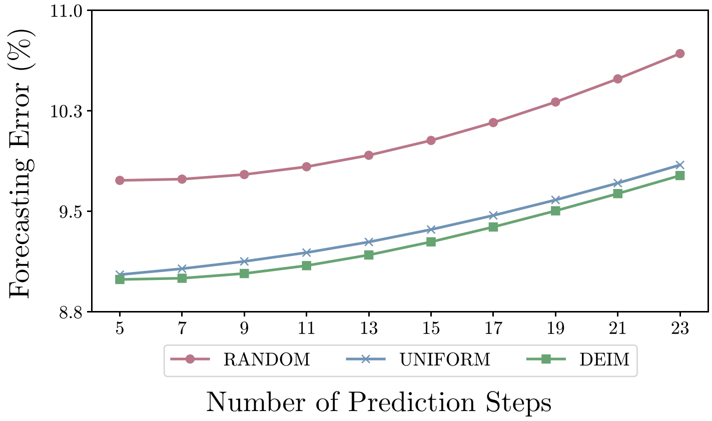

Figure 8. A representative error propagation of model inference on the RBC dataset. We define 10 steps in the training and leverage 64 sensors with random, uniform, and DEIM sampling schemes. Each line shows relative forecasting errors with an increasing number of prediction steps.

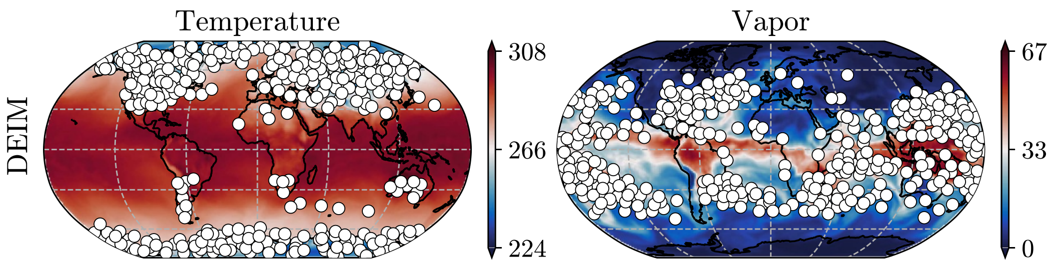

Figure 9. Visualizations of the sensor placements for temperature and vapor variables with the DEIM scheme. We consider 360 sensors (white dots) for the entire global domain.

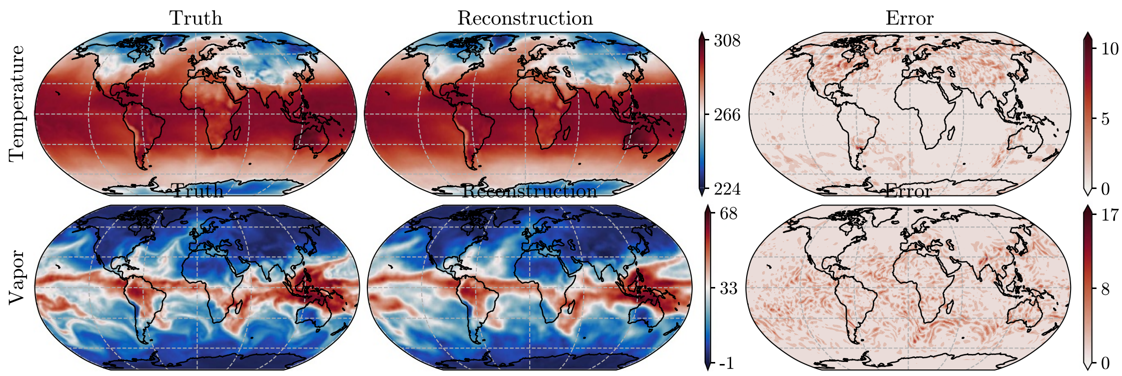

Figure 10. Reconstruction results for the weather data snapshot projected on the global map, including temperature and vapor variables. We consider the number of sensors $n_s = 360$ using the DEIM sampling method and the prediction time step $T = 8$. Each column, from left to right, shows the ground-truth snapshots, the reconstructed snapshots, and the absolute error contours.

local patterns. Moreover, vapor data exhibits more complex dynamic behaviors compared to temperature data, and this leads to relatively higher absolute reconstruction errors.

We also analyze the error propagation performance of our models concerning the temperature and vapor variables, as illustrated in figure 11. Both variables exhibit a moderate change in errors from $T = 5$ to $T = 8$. The errors of temperature data increase mildly but those in vapor variables display a relatively larger growth after $T = 8$. This discrepancy is due to the complex and unstable dynamic patterns inherent in vapor data.

4.6. Baseline comparison

In this subsection, we compare different end-to-end frameworks with our proposed approach. Instead of individual NN components, we incorporate two alternative representative approaches in the context of dynamics forecasting and reconstruction from sparse data: (i) bicubic interpolation for upsampling and

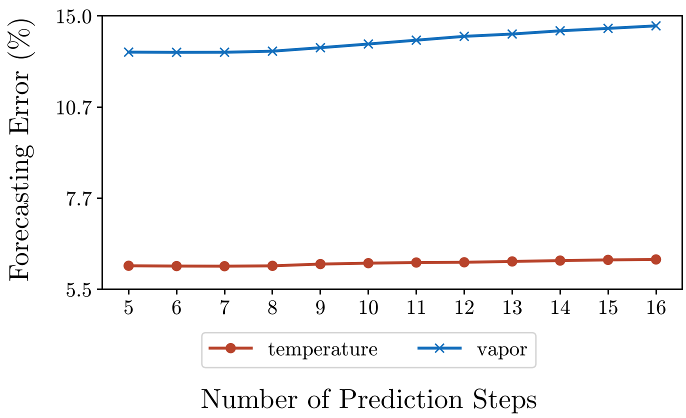

Figure 11. The error propagation of the models trained on the temperature and vapor variables. The prediction step is set to 8, and we choose 360 sensors with the DEIM sampling method.

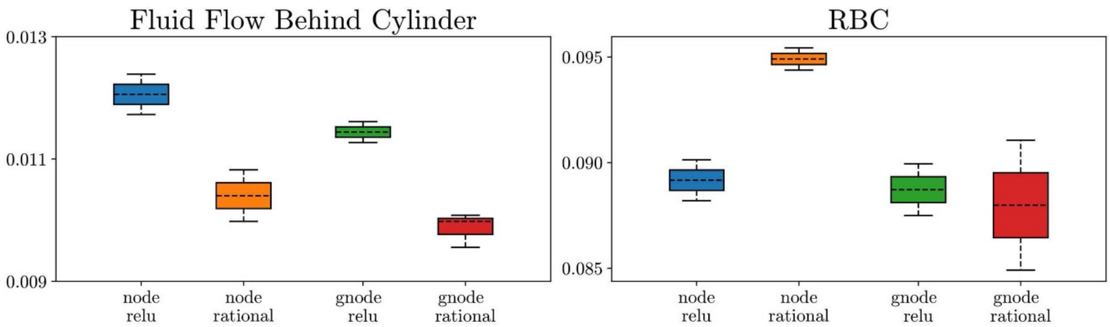

Figure 12. Ablation studies on different types of NODEs and activation functions. We test the reconstruction performance of each model on fluid flow behind a cylinder (left) and RBC data (right) with DEIM sampling, with the number of sensors n_s as 8 and 80, respectively.

FNO for modeling temporal dynamics; and (ii) masked autoencoder (MAE) for extracting latent features from sparse sensor measurements, GNODE and swin transformer for temporal and spatial modeling. We compare our framework to model (i) to show that our proposed framework is effective in handling forecasting high-dimensional systems from sparse data. When comparing our framework to model (ii), we show the superiority of our end-to-end method in terms of the sampling techniques at a very sparse level.

4.6.1. Dynamics modeling comparison

Here, we compare our proposed method with the Fourier neural operator (FNO) [94] to assess performance across three datasets. FNO has demonstrated high relevance and strong performance in handling super-resolution and forecasting tasks [95,96]. FNO is also widely recognized for its effectiveness in learning mappings for high-dimensional and complex physical systems, making it suitable for capturing spatiotemporal dynamics across various scientific machine learning applications. Therefore, we chose the FNO as the baseline for performance comparison in spatiotemporal dynamics modeling.

Since FNO cannot process sparse measurements directly, we first use the same up-sampling layer as in our framework, also shown in figure 1, to map the sparse measurements to higher-dimensional latent variables $\mathbf{Z}$, described in section 3.2. For the fluid flow behind the cylinder and RBC System datasets, we employ 32 sensors with uniform sampling, while for the Weather data, we use 360 sensors sampled via DEIM. Following prior work [95], we then use bicubic interpolation to upsample these latent variables to match the resolution of the high-dimensional outputs, which are 200×384, 128×128, and 180×360 for fluid flow behind the cylinder, RBC System, and Weather data, respectively. FNO then takes these upsampled high-resolution inputs and is trained autoregressively to predict future timesteps, matching the same rollout steps in our numerical examples.

We compare the results of our framework with this FNO-based approach, which first upsamples sparse measurements and then uses FNO to forecast high-resolution dynamics. This design ensures that both frameworks perform the same task, enabling a fair comparison. The reconstruction error results presented in

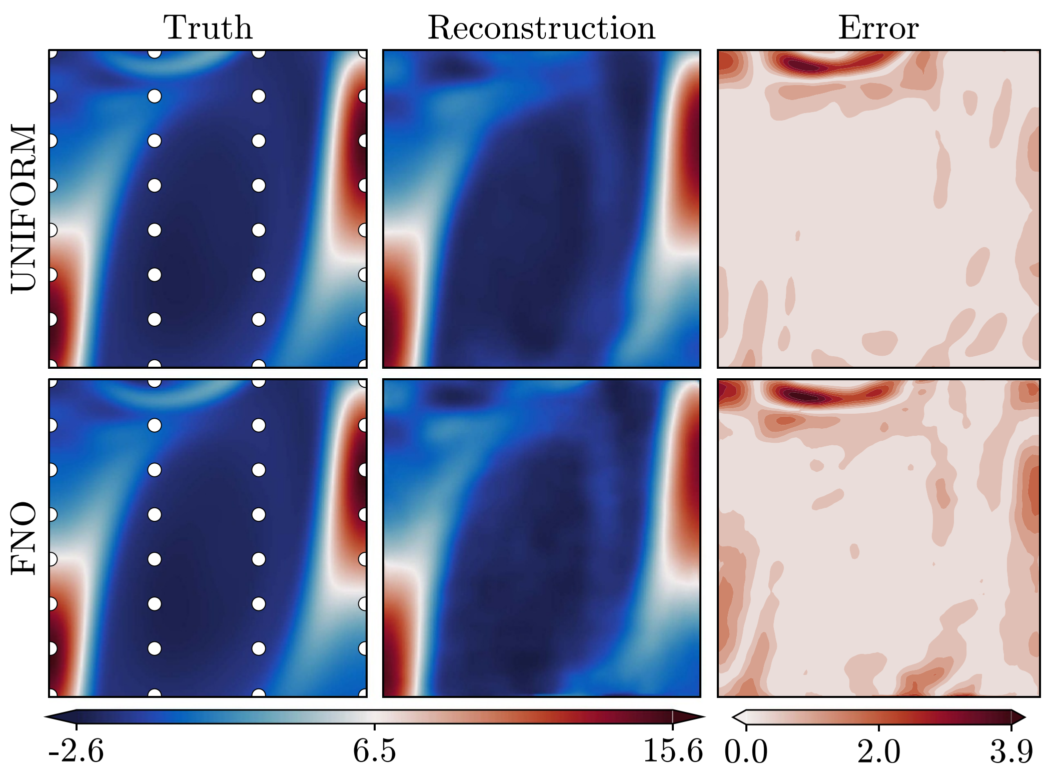

Figure 13. Comparison reconstruction results for the RBC system with uniform sampling on the proposed method and the FNO baseline. The number of sensors n_s and the prediction time step T are set as 32 and 10, respectively. The white dots represent sensors.

Table 2. A summary of baseline comparison across all datasets. FNO and GNODE denote the baseline model and our method, respectively. We evaluate the model performance using L2RE.

Dataset	FNO (%)	GNODE (%)
Fluid	2.015	1.033
RBC	11.595	10.160
Climate	9.962	6.006

table 2 demonstrate that our proposed method excels the baseline model in dynamics forecasting and reconstruction from sparse measurements.

In figure 13, we compare the snapshot between our proposed method that used the swin transformer [9] and the FNO baseline on the RBC dataset. We can observe, especially from the top-left corner and the edge area, that swin transformer excels at capturing local details, compared to the FNO baseline. This comparison demonstrates swin transformer's ability to preserve multi-scale features from its hierarchical architecture design with a shifted windowing scheme.

4.6.2. Sampling methods comparison

Here, we compare our sampling methods to the masked autoencoder (MAE) [96] for sparse measurements. MAE has shown robust performance in reconstructing missing data and representation learning, attracting increasing attention in scientific machine learning [97, 98]. MAE learns data representations by masking portions of the input data and reconstructing the masked sections during training, an approach which aligns well with the data compression and reconstruction challenges in our study. The encoder of the MAE can process the visible portions of the input data effectively and robustly. Hence, we include MAE in our comparisons for sparse measurement sampling, due to its flexibility to achieve a similar sparsity level to that in our framework.

We apply random masks to the high-resolution data to match the sparsity of our approach. For example, each high-resolution RBC snapshot (128×128) is randomly reduced to 32 (n_{sensors}) visible pixels while the other ($128 \cdot 128 - n_{\text{sensors}}$) pixels are masked. The masked snapshots are then passed into a standard ViT encoder [99]. During MAE training, the loss is computed only on the masked pixels. We increase the learning rate to 1×10^{-3} while keeping the other experimental conditions identical.

Table 3. Reconstruction error comparison between our method and MAE model on RBC data. We evaluate the model performance using L2RE.

Dataset	MAE (%)	GNODE (%)
RBC	37.575	10.160

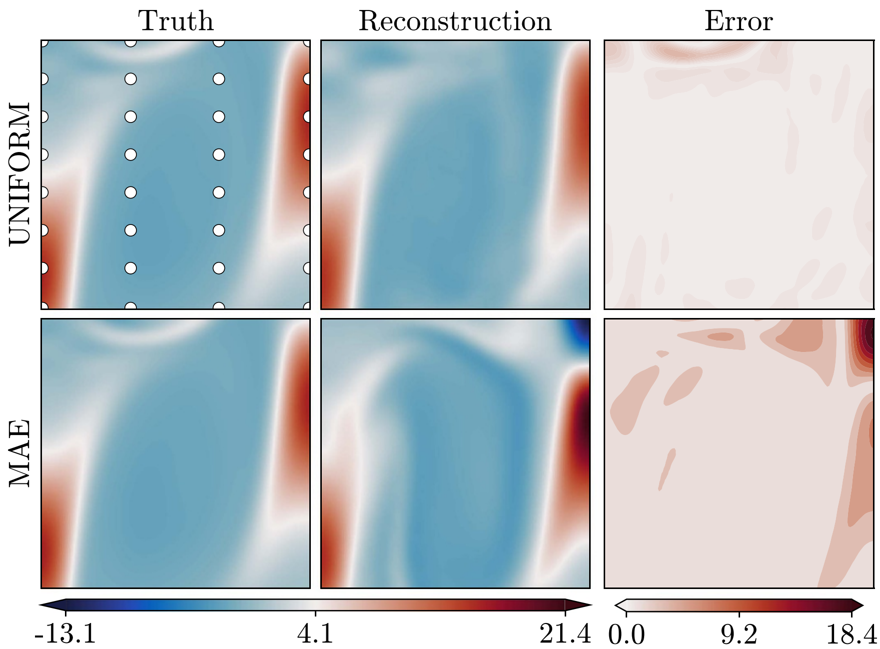

Figure 14. Comparison reconstruction results of the RBC system with uniform sampling on the proposed method and the MAE model. The number of sensors n_s and the prediction time step T are set as 32 and 10, respectively. The white dots on the proposed method represent sensors.

Table 3 compares the reconstruction error between our method and the MAE model, both using 32 visible measurements for each snapshot. We attribute the poor performance of the MAE model to the significant sparsity of the input, making effective learning challenging. The original MAE paper applies the mask after patch embedding by reducing the sequence lengths. However, solely reducing the sequence length, or increasing the mask ratio, cannot achieve comparable sparsity as in our experiment setups. Figure 14 further illustrates that the MAE model struggles to accurately reconstruct the dynamics due to the significant information loss caused by the high-ratio masks.

4.7. Ablation study

We conduct an ablation study on activation functions and different types of NODEs to validate the effectiveness of our proposed pipeline. To be more concrete, we investigate the potential of two NODE modules (i.e. the vanilla and GNODEs) for learning dynamics and two activation functions (i.e. rational and ReLU). We use fluid flow behind a cylinder and RBC data as two illustrative examples. The DEIM sampling is leveraged for testing the performance of each model. We use $\{4, 8, 16, 32\}$ and $\{16, 32, 64, 80\}$ sensors, as well as 15 and 10 steps, for fluid cylinder and RBC datasets, respectively. We run the experiments with 3 different random seeds. Generally, the rational activation function consistently outperforms the ReLU function. By using the rational function, we observe that the GNODE model presents superior performance on the fluid cylinder with fewer sensors (4 and 8) and the RBC dataset with relatively more sensors (64 and 80). This reflects that the GNODE model is capable of learning complex dynamics well with a reasonable sensor placement that approximates the rank of a specific dynamical system, as demonstrated in figure 12.

Although rational NNs have shown competitive performance in learning complex systems, they also have certain potential limitations. The inclusion of additional trainable parameters in the activation function generally increases training time and memory usage. Moreover, the higher model complexity may lead to numerical instabilities if the model is not properly initialized or constrained.

5. Conclusion

In this work, we proposed a new architecture for forecasting and reconstructing high-dimensional spatio-temporal data from sparse measurements. Our model combines a GNODE for modeling the dynamics, and a ViT backbone for reconstructing high-dimensional spatial fields.

This approach is an alternative to other methods that typically decode high-dimensional spatial fields before modeling the dynamics. Although several existing techniques model dynamics in a latent space, they require the preliminary step of learning an encoder. Our method bypasses this step by using a sparse set of measurements directly, thereby reducing computational complexity. We show that the performance of our method remains robust across various sensor placement strategies, especially as the number of sensors increases. This consistent performance across different sensor arrangements shows the flexibility of our architecture in real-world scenarios, where sensor data is frequently sparse and unevenly distributed.

In comparison to traditional physics-based methods [23, 100–106], our approach offers several advantages. While physics-based methods provide a strong foundation for reconstructing spatial fields, they are often limited by the complexity of the phenomena and the computational resources required. Moreover, traditional methods often struggle with the non-linearities and high dimensionality of the data.

Data availability statement

The code for the implementation of our proposed model is available at https://github.com/jsong2333333/neuralode_reconstruction under the MIT License.

Acknowledgment

Z S acknowledges an appointment to the NSF Mathematical Sciences Graduate Internship. N B E, P R, M W M, and X S L was supported in part by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Contract No. DE-AC02-05CH11231 at Lawrence Berkeley National Laboratory.

ORCID iDs

Jialin Song https://orcid.org/0009-0003-1381-6754
Zezheng Song https://orcid.org/0009-0009-6493-2337
Pu Ren https://orcid.org/0000-0002-6354-385X
Xiaoye S Li https://orcid.org/0000-0002-0747-698X

References

[1] Brunton S L, Noack B R and Koumoutsakos P 2020 Machine learning for fluid mechanics *Ann. Rev. Fluid Mech.* **52** 477–508
[2] Wang R and Yu R 2021 Physics-guided deep learning for dynamical systems: a survey (arXiv:2107.01272)
[3] Fukami K, Fukagata K and Taira K 2019 Super-resolution reconstruction of turbulent flows with machine learning *J. Fluid Mech.* **870** 106–20
[4] Cheng S *et al* 2023 Machine learning with data assimilation and uncertainty quantification for dynamical systems: a review *IEEE/CAA J. Autom. Sin.* **10** 1361–87
[5] Benjamin Erichson N, Mathelin L, Yao Z, Brunton S L, Mahoney M W and Nathan Kutz J 2020 Shallow neural networks for fluid flow reconstruction with limited sensors *Proc. R. Soc.* A **476** 20200097
[6] Güemes A, Discetti S, Ianiro A, Sirmacek B, Azizpour H and Vinuesa R 2021 From coarse wall measurements to turbulent velocity fields through deep learning *Phys. Fluids* **33** 2021
[7] Chen R T Q, Rubanova Y, Bettencourt J and Duvenaud D K 2018 Neural ordinary differential equations *Advances in Neural Information Processing Systems* p 31
[8] Xu X, Hasan A, Elkhalil K, Ding J, and Tarokh V 2021 Characteristic neural ordinary differential equations (arXiv:2111.13207)
[9] Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S and Guo B 2021 Swin transformer: hierarchical vision transformer using shifted windows *Proc. IEEE/CVF Int. Conf. on Computer Vision* pp 10012–22
[10] Liang J, Cao J, Sun G, Zhang K, Van Gool L and Timofte R 2021 Swinir: image restoration using swin transformer *Proc. IEEE/CVF Int. Conf. on Computer Vision* pp 1833–44
[11] Callaham J L, Maeda K and Brunton S L 2019 Robust flow reconstruction from limited measurements via sparse representation *Phys. Rev. Fluids* **4** 103907
[12] Manohar K, Brunton B W, Nathan Kutz J and Brunton S L 2018 Data-driven sparse sensor placement for reconstruction: demonstrating the benefits of exploiting known patterns *IEEE Control Syst. Mag.* **38** 63–86
[13] Chen X, Zhang C, Chen X, Saunier N and Sun L 2023 Discovering dynamic patterns from spatiotemporal data with time-varying low-rank autoregression *IEEE Trans. Knowl. Data Eng.* **36** 1–14
[14] Yu J and Hesthaven J S 2019 Flowfield reconstruction method using artificial neural network *AIAA J.* **57** 482–98
[15] Fukami K, Maulik R, Ramachandra N, Fukagata K and Taira K 2021 Global field reconstruction from sparse sensors with voronoi tessellation-assisted deep learning *Nat. Mach. Intell.* **3** 945–51

[16] Fukami K, Fukagata K and Taira K 2023 Super-resolution analysis via machine learning: a survey for fluid flows *Theor. Comput. Fluid Dyn.* **37** 1–24
[17] Vinuesa R and Brunton S L 2022 Enhancing computational fluid dynamics with machine learning *Nat. Comput. Sci.* **2** 358–66
[18] Ren P, Benjamin Erichson N, Subramanian S, San O, Lukic Z, and Mahoney M W 2023 Superbench: a super-resolution benchmark dataset for scientific machine learning (arXiv:2306.14070)
[19] Gao H, Sun L and Wang J-X 2021 Super-resolution and denoising of fluid flow using physics-informed convolutional neural networks without high-resolution labels *Phys. Fluids* **33** 33
[20] Yang Q, Hernandez-Garcia A, Harder P, Ramesh V, Sattegeri P, Szwarcman D, Watson C D, and Rolnick D 2023 Fourier neural operators for arbitrary resolution climate data downscaling (arXiv:2305.14452)
[21] Xu Q, Zhuang Z, Pan Y and Wen B 2023 Super-resolution reconstruction of turbulent flows with a transformer-based deep learning framework *Phys. Fluids* **35** 35
[22] Zhang M, Yousif M Z, Yu L and Lim H-C 2023 A swin-transformer-based model for efficient compression of turbulent flow data *Phys. Fluids* **35** 085108
[23] Shu D, Li Z and Barati Farimani A 2023 A physics-informed diffusion model for high-fidelity flow field reconstruction *J. Comput. Phys.* **478** 111972
[24] Fan X, Akhare D, and Wang J-X 2024 Neural differentiable modeling with diffusion-based super-resolution for two-dimensional spatiotemporal turbulence (arXiv:2406.20047)
[25] Hochreiter S and Schmidhuber J 1997 Long short-term memory *Neural Comput.* **9** 1735–80
[26] Chung J, Gulcehre C, Cho K, and Bengio Y 2014 Empirical evaluation of gated recurrent neural networks on sequence modeling (arXiv:1412.3555)
[27] Benjamin Erichson N, Azencot O, Queiruga A, Hodgkinson L, and Mahoney M W 2020 Lipschitz recurrent neural networks (arXiv:2006.12070)
[28] Konstantin Rusch T, Mishra S, Benjamin Erichson N and Mahoney M W 2021 Long expressive memory for sequence modeling *Int. Conf. on Learning Representations*
[29] Shi X, Chen Z, Wang H, Yeung D-Y, Wong W-K and Woo W-chun 2015 Convolutional LSTM network: a machine learning approach for precipitation nowcasting *Advances in Neural Information Processing Systems* vol 28
[30] Lyu D, Nakata R, Ren P, Mahoney M W, Pitarka A, Nakata N, and Benjamin Erichson N 2024 Wavecastnet: an AI-enabled wavefield forecasting framework for earthquake early warning (arXiv:2405.20516)
[31] Dupont E, Doucet A and Whye Teh Y 2019 Augmented neural odes *Advances in Neural Information Processing Systems* vol 32
[32] Kidger P, Morrill J, Foster J and Lyons T 2020 Neural controlled differential equations for irregular time series *Advances in Neural Information Processing Systems* vol 33 pp 6696–707
[33] Krishnapriyan A S, Queiruga A F, Benjamin Erichson N and Mahoney M W 2023 Learning continuous models for continuous physics *Commun. Phys.* **6** 319
[34] Zhou H, Zhang S, Peng J, Zhang S, Li J, Xiong H and Zhang W 2021 Informer: beyond efficient transformer for long sequence time-series forecasting *Proc. AAAI Conf. on Artificial Intelligence* pp 11106–15
[35] Wu H, Xu J, Wang J and Long M 2021 Autoformer: decomposition transformers with auto-correlation for long-term series forecasting *Advances in Neural Information Processing Systems* vol 34 pp 22419–30
[36] Zhang Y and Yan J 2023 Crossformer: transformer utilizing cross-dimension dependency for multivariate time series forecasting *The 11th Int. Conf. on Learning Representations*
[37] Wang X, Zhou T, Wen Q, Gao J, Ding B and Jin R 2024 Card: channel aligned robust blend transformer for time series forecasting *The 12th Int. Conf. on Learning Representations*
[38] Chen P, Zhang Y, Cheng Y, Shu Y, Wang Y, Wen Q, Yang B, and Guo C 2024 Pathformer: multi-scale transformers with adaptive pathways for time series forecasting (arXiv:2402.05956)
[39] Xiao J, Liu R and Dyer E L 2024 Gaformer: enhancing timeseries transformers through group-aware embeddings *The 12th Int. Conf. on Learning Representations*
[40] Feng S, Miao C, Zhang Z and Zhao P 2024 Latent diffusion transformer for probabilistic time series forecasting *Proc. AAAI Conf. on Artificial Intelligence* pp 11979–87
[41] Li Y *et al* 2024 Transformer-modulated diffusion models for probabilistic multivariate time series forecasting *The 12th Int. Conf. on Learning Representations*
[42] Gao Z, Shi X, Wang H, Zhu Y, Bernie Wang Y, Li M and Yeung D-Y 2022 Earthformer: exploring space-time transformers for earth system forecasting *Advances in Neural Information Processing Systems* vol 35 pp 25390–403
[43] Lam R *et al* 2023 Learning skillful medium-range global weather forecasting *Science* **382** 1416–21
[44] Price I, Sanchez-Gonzalez A, Alet F, Ewalds T, El-Kadi A, Stott J, Mohamed S, Battaglia P, Lam R, and Willson M 2023 Gencast: diffusion-based ensemble forecasting for medium-range weather (arXiv:2312.15796)
[45] Kochkov D *et al* 2024 Neural general circulation models for weather and climate *Nature* **632** 1060–6
[46] Zhang Y, Long M, Chen K, Xing L, Jin R, Jordan M I and Wang J 2023 Skilful nowcasting of extreme precipitation with nowcastnet *Nature* **619** 526–32
[47] Bi K, Xie L, Zhang H, Chen X, Gu X and Tian Q 2023 Accurate medium-range global weather forecasting with 3d neural networks *Nature* **619** 533–8
[48] Rühling Cachay S, Zhao B, Joren H and Rose Y 2024 Dyffusion: a dynamics-informed diffusion model for spatiotemporal forecasting *Advances in Neural Information Processing Systems* vol 36
[49] Luo X, Xu W, Nadiga B, Ren Y and Yoo S 2024 Continuous field reconstruction from sparse observations with implicit neural networks *ICLR* (OpenReview.net)
[50] Yin Y, Kirchmeyer M, Franceschi J-Y, Rakotomamonjy A and Gallinari P 2023 Continuous PDE dynamics forecasting with implicit neural representations *ICLR* (OpenReview.net)
[51] Williams J P, Zahn O, and Nathan Kutz J 2023 Sensing with shallow recurrent decoder networks (arXiv:2301.12011)
[52] Nathan Kutz J, Reza M, Faraji F, and Knoll A 2024 Shallow recurrent decoder for reduced order modeling of plasma dynamics (arXiv:2405.11955)
[53] Fukami K, Fukagata K and Taira K 2021 Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows *J. Fluid Mech.* **909** A9
[54] Ren P, Rao C, Liu Y, Ma Z, Wang Q, Wang J-X and Sun H 2023 PhySR: Physics-informed deep super-resolution for spatiotemporal data *J. Comput. Phys.* **492** 112438

[55] Solera-Rico A, Sanmiguel Vila C, Gómez-López M, Wang Y, Almashjary A, Dawson S T M and Vinuesa R 2024 β-variational autoencoders and transformers for reduced-order modelling of fluid flows *Nat. Commun.* **15** 1361

[56] Doyeon Kim T, Can T, and Krishnamurthy K 2023 Trainability, expressivity and interpretability in gated neural odes

[57] Shi W, Caballero J, Huszár F, Totz J, Aitken A P, Bishop R, Rueckert D and Wang Z 2016 Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* pp 1874–83

[58] Massaroli S, Poli M, Park J, Yamashita A and Asama H 2020 Dissecting neural odes *Advances in Neural Information Processing Systems* vol 33 pp 3952–63

[59] Queiruga A, Benjamin Erichson N, Hodgkinson L and Mahoney M W 2021 Stateful ode-nets using basis function expansions *Advances in Neural Information Processing Systems* vol 34 pp 21770–81

[60] Kidger P 2022 On neural differential equations (arXiv:2202.02435)

[61] Meade Jr A J and Fernandez A A 1994 The numerical solution of linear ordinary differential equations by feedforward neural networks *Math. Comput. Modelling* **19** 1–25

[62] Lagaris I E, Likas A and Fotiadis D I 1998 Artificial neural networks for solving ordinary and partial differential equations *IEEE Trans. Neural Netw.* **9** 987–1000

[63] Wu. J 1998 Symmetric functional differential equations and neural networks with memory *Trans. Am. Math. Soc.* **350** 4799–838

[64] Ramuhalli P, Udpa L and Udpa S S 2005 Finite-element neural networks for solving differential equations *IEEE Trans. Neural Netw.* **16** 1381–92

[65] Zhu A, Jin P, Zhu B and Tang Y 2022 On numerical integration in neural ordinary differential equations *Int. Conf. on Machine Learning* (PMLR) pp 27527–47

[66] Ott K, Katiyar P, Hennig P and Tiemann M 2020 Resnet after all: neural odes and their numerical solution *Int. Conf. on Learning Representations*

[67] Gusak J, Katrutsa A, Daulbaev T, Cichocki A, and Oseledets I 2021 Meta-solver for neural ordinary differential equations (arXiv:2103.08561)

[68] Kloberdanz E and Le W 2023 S-solver: numerically stable adaptive step size solver for neural odes *Int. Conf. on Artificial Neural Networks* (Springer) pp 388–400

[69] Zhu Q, Guo Y and Lin W 2020 Neural delay differential equations *Int. Conf. on Learning Representations*

[70] Benjamin Erichson N, Hoe Lim S, and Mahoney M W 2022 Gated recurrent neural networks with weighted time-delay feedback (arXiv:2212.00228)

[71] Norcliffe A, Bodnar C, Day B, Simidjievski N and Liò P 2020 On second order behaviour in augmented neural odes *Advances in Neural Information Processing Systems* vol 33 pp 5911–21

[72] Tallec C and Ollivier Y 2018 Can recurrent neural networks warp time? (arXiv:1804.11188)

[73] Jaeger H, Lukoševičius M, Popovici D and Siewert U 2007 Optimization and applications of echo state networks with leaky-integrator neurons *Neural Netw.* **20** 335–52

[74] Boullé N, Nakatsukasa Y and Townsend A 2020 Rational neural networks *Advances in Neural Information Processing Systems* vol 33 pp 14243–53

[75] Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A N, Kaiser Łukasz and Polosukhin I 2017 Attention is all you need *Advances in Neural Information Processing Systems* vol 30

[76] Ba J L, Ryan Kiros J, and Hinton G E 2016 Layer normalization (arXiv:1607.06450)

[77] Chaturantabut S and Sorensen D C 2010 Nonlinear model reduction via discrete empirical interpolation *SIAM J. Sci. Comput.* **32** 2737–64

[78] Antil H, Heinkenschloss M and Sorensen D C 2014 Application of the discrete empirical interpolation method to reduced order modeling of nonlinear and parametric systems *Reduced Order Methods for Modeling and Computational Reduction* pp 101–36

[79] Joshi S S, Prasad R and Choi S 2022 Discrete empirical interpolation based hyper-reduced order model for steady hypersonic flows *AIAA SCITECH 2022 Forum* p 0079

[80] Fu H, Wang H and Wang Z 2018 Pod/deim reduced-order modeling of time-fractional partial differential equations with applications in parameter identification *J. Sci. Comput.* **74** 220–43

[81] Drmac Z and Gugercin S 2016 A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions *SIAM J. Sci. Comput.* **38** A631–48

[82] Saibaba A K 2020 Randomized discrete empirical interpolation method for nonlinear model reduction *SIAM J. Sci. Comput.* **42** A1582–608

[83] Kingma D P 2014 Adam: a method for stochastic optimization (arXiv:1412.6980)

[84] Paszke A *et al* 2019 Pytorch: an imperative style, high-performance deep learning library *Advances in Neural Information Processing Systems* vol 32

[85] Hao Z, et al 2023 Pinnacle: a comprehensive benchmark of physics-informed neural networks for solving pdes (arXiv:2306.08827)

[86] Noack B R, Afanasiev K, Morzyński M, Tadmor G and Thiele F 2003 A hierarchy of low-dimensional models for the transient and post-transient cylinder wake *J. Fluid Mech.* **497** 335–63

[87] Taira K and Colonius T 2007 The immersed boundary method: a projection approach *J. Comput. Phys.* **225** 2118–37

[88] Colonius T and Taira K 2008 A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions *Comput. Methods Appl. Mech. Eng.* **197** 2131–46

[89] Um K, Brand R, Raymond Fei Y, Holl P and Thuerey N 2020 Solver-in-the-loop: learning from differentiable physics to interact with iterative pde-solvers *Advances in Neural Information Processing Systems* vol 33 pp 6111–22

[90] Kochkov D, Smith J A, Alieva A, Wang Q, Brenner M P and Hoyer S 2021 Machine learning–accelerated computational fluid dynamics *Proc. Natl Acad. Sci.* **118** e2101784118

[91] Burns K J, Vasil G M, Oishi J S, Lecoanet D and Brown B P 2020 Dedalus: a flexible framework for numerical simulations with spectral methods *Phys. Rev. Res.* **2** 023068

[92] Hersbach H *et al* 2020 The era5 global reanalysis *Q. J. R. Meteorol. Soc.* **146** 1999–2049

[93] Kalnay E 2003 *Atmospheric Modeling, Data Assimilation and Predictability* vol 341 (Cambridge University Press)

[94] Li Z, Borislavov Kovachki N, Azizzadenesheli K, Liu B, Bhattacharya K, Stuart A M and Anandkumar A 2021 Fourier neural operator for parametric partial differential equations *ICLR* (OpenReview.net)

[95] Jiang P, Yang Z, Wang J, Huang C, Xue P, Chakraborty T C, Chen X and Qian Y 2023 Efficient super-resolution of near-surface climate modeling using the fourier neural operator *J. Adv. Model. Earth Syst.* **15** e2023MS003800

[96] He K, Chen X, Xie S, Li Y, Dollár P, and Girshick R 2021 Masked autoencoders are scalable vision learners

[97] Ashiqur Rahman Md et al 2024 Pretraining codomain attention neural operators for solving multiphysics pdes (arXiv:2403.12553)
[98] Chen W, Song J, Ren P, Subramanian S, Morozov D, and Mahoney M W 2024 Data-efficient operator learning via unsupervised pretraining and in-context learning (arXiv:2402.15734)
[99] Dosovitskiy A et al 2020 An image is worth 16×16 words: transformers for image recognition at scale (arXiv:2010.11929)
[100] Kirby M and Armbruster D 1992 Reconstructing phase space from pde simulations *Z. Angew. Math. Phys.* **43** 999–1022
[101] Xu S, Sun Z, Huang R, Guo D, Yang G and Ju. S 2023 A practical approach to flow field reconstruction with sparse or incomplete data through physics informed neural network *Acta Mech. Sin.* **39** 322302
[102] Rowley C W, Kevrekidis I G, Marsden J E and Lust K 2003 Reduction and reconstruction for self-similar dynamical systems *Nonlinearity* **16** 1257
[103] Zhao H-K, Osher S and Fedkiw R 2001 Fast surface reconstruction using the level set method *Proc. IEEE Workshop on Variational and Level Set Methods in Computer Vision* (IEEE) pp 194–201
[104] Duan Y, Yang L, Qin H and Samaras D 2004 Shape reconstruction from 3d and 2d data using pde-based deformable surfaces *Computer Vision-ECCV 2004: 8th European Conf. on Computer Vision, Proc., Part III 8 (Prague, Czech Republic, 11–14 May 2004)* vol 8 (Springer) pp 238–51
[105] Alonso A A, Frouzakis C E and Kevrekidis I G 2004 Optimal sensor placement for state reconstruction of distributed process systems *AIChE J.* **50** 1438–52
[106] Dsilva C J, Talmon R, Rabin N, Coifman R R and Kevrekidis I G 2013 Nonlinear intrinsic variables and state reconstruction in multiscale simulations *J. Chem. Phys.* **139** 184109