Neural-network-augmented Empirical Interpolation Method for field reconstruction with noise and vibration tolerance

Han Li[a], Helin Gong[a,*], Chuanju Xu[b]

[a]*Paris Elite Institute of Technology, Shanghai Jiao Tong University, Shanghai 200240, China*
[b]*School of Mathematical Sciences and Fujian Provincial Key Laboratory of Mathematical Modeling and High Performance Scientific Computing, Xiamen University, Xiamen 361005, China*

Abstract

Field reconstruction methods based on machine learning often face challenges when handling unstructured data and designing complex model structures. On the other hand, Reduced Order Models (ROM) struggle with robustness in the presence of observation noise and sensor vibrations. Striking a balance between lightweight design, the ability to handle unstructured data, and robustness presents a significant challenge. In this paper, we introduce the EIM-NN algorithm, which leverages neural networks to determine the coefficients of the subspace found by the EIM algorithm. Furthermore, we present the EIM-TNN algorithm, which enhances robustness by designing a loss function incorporating Tikhonov regularization. Our neural network consists of only two fully connected layers, allowing it to handle unstructured data while maintaining a lightweight profile. Experimental results demonstrate the algorithm's ability to significantly enhance robustness against noise and sensor vibrations without compromising its accuracy in fitting the original data. Additionally, the algorithm's lightweight nature ensures that the added training time and memory requirements over EIM remain acceptable, making it adaptable to a range of industrial applications.

Keywords: Field reconstruction, EIM-TNN, Lightweight, Observation noise, Sensor vibration

1. Introduction

In the past, the primary approach to field reconstruction involved the utilization of numerical optimization methods, particularly by developing computationally efficient reduced-order models (ROMs) [1, 2, 3]. Among the various ROM techniques, Reduced Basis (RB) methods [4, 5, 6] have established themselves as a well-established and widely adopted class of ROM techniques. RB methods typically revolve around identifying a set of reduced basis functions within a lower-dimensional subspace. Subsequently, they map low-dimensional observations to high-dimensional physical fields by determining the coefficients of these basis functions.

*Corresponding author: gonghelin@sjtu.edu.cn

Different approaches can be employed to construct the reduced basis, including greedy algorithms [7, 8] and Proper Orthogonal Decomposition (POD) [9, 10]. Additionally, the Empirical Interpolation Method (EIM) [11, 12, 13] represents a non-intrusive ROM technique that constructs an empirical interpolation from the solutions of the parametrized mathematical model or from observations of the physical state to be approximated. EIM and its generalized version, GEIM [14, 15, 16, 17], enable the integration of the mathematical model and observation data to reconstruct full order fields.

However, in real industrial environments, sensor observations are often subject to contamination, which can stem from errors in observation readings, vibrations in the sensor's position, and other sources. In the presence of contaminated observations, the performance of EIM and GEIM is compromised, as they may no longer converge and can even exacerbate observation noise [18]. It is important to note that these ROMs often entail solving complex optimization problems, demanding a substantial number of iterations and, consequently, leading to high computational requirements.

In recent years, there has been a notable upward trend in the integration of machine learning techniques into field reconstruction endeavors. This trend aims to enhance both computational efficiency and robustness, capitalizing on the nonlinear capabilities of neural networks [19, 20, 21, 22, 23]. For instance, in a specific case, Fukami et al. [24] utilized Voronoi tessellation to construct a structured-grid representation based on sensor positions. Subsequently, they employed Convolutional Neural Networks (CNNs) to establish a mapping from mobile sensors to the physical field. In a related vein, Gong et al. [25] proposed the application of the VCNN model to nuclear reactors to address the observation inaccuracy arising from sensor position vibrations due to reactor aging. Building upon these foundations, Li et al. [26] took further strides in optimizing their machine learning model. They achieved this by amalgamating the Voronoi tessellation method with the Residual neural network [27] and deliberately incorporating sensor vibration and observation noise into the training data. This innovative approach led to the development of a Noise and Vibration Tolerant ResNet (NVT-ResNet), which was specifically designed to enhance the robustness of the reconstruction model in the presence of noise and sensor vibrations. This fusion approach resulted in improved robustness for the reconstruction model against noise and sensor vibrations, leading to enhanced reconstruction accuracy.

While machine learning models offer robust and efficient solutions [28], it is essential to acknowledge that ROMs grounded in physical principles possess distinct advantages. These advantages include generalizability, interpretability, and adherence to well-established physical principles [29]. As a result, there has been a concerted effort in recent research to introduce nonlinear operators into the low-dimensional space within ROMs. This approach aims to enhance the robustness of the models.

For instance, some works in this area focus on manifold learning [30, 31, 32, 33], seeking a low-dimensional approximation space within the solution manifold. They then introduce nonlinear operators for learning within that subspace, thereby improving the model's robustness. For example, one approach [33] provides certified recovery bounds for inversion procedures based on nonlinear approximation spaces, while another [32] proposes to surpass

the limitations of linear performance by using straightforward nonlinear reduced models composed of a finite union of linear spaces.

Additionally, another series of research endeavors aims to integrate machine learning techniques into ROMs [34, 35, 36, 37, 38]. In these cases, machine learning plays the role of a nonlinear operator within the ROM framework. This integration of machine learning adds further flexibility and capability to ROMs while preserving their physical grounding. For instance, [36] introduces a novel variable data assimilation scheme known as "Voronoi-tessellation Inverse operator for Variational Data Assimilation (VIVID)", which effectively leverages Voronoi subdivision and the capabilities of convolutional neural networks to handle sparse, unstructured, and time-varying sensor data. In [38], a novel approach called the learned singular value decomposition (L-SVD) is introduced. This method combines autoencoders to simultaneously learn and interconnect latent codes for the target signals and the provided measurements, allowing for the solution of inverse problems.

Previous attempts to integrate machine learning into low-dimensional spaces have not fully leveraged the inherent advantages of low dimensionality, in contrast to manifold learning methods [32, 33]. Frequently, these endeavors have relied on multi-layer CNNs [36, 25, 26], or even more intricate network architectures [37]. This reliance on complex neural network structures is primarily due to the fact that the process typically operates on a discrete representation of the mesh that defines the physical field of interest. In engineering problems, this mesh discretization can involve millions of dimensions or even more. However, it's important to note that this approach often comes at the cost of substantial memory usage and additional computational time, both during the training and testing phases. In reality, learning within the low-dimensional subspace of the manifold space is naturally conducive to dimensionality reduction and noise filtering. Therefore, it is only necessary to design a lightweight neural network to introduce nonlinearity within that space. In this paper, we adopt a neural network configuration comprising only two fully connected layers to achieve this objective. This lightweight design offers efficient and effective nonlinear modeling within the low-dimensional subspace.

It's also important to highlight that CNNs typically require structured, gridded input data, necessitating additional pre-processing steps prior to data input. Techniques such as Voronoi tessellation, as mentioned in [24], have been employed to address this challenge. However, these methods entail larger data storage requirements since they store data in the same shape as the entire spatial domain.

In addition to this, data in real industrial environments often face problems such as vibration of sensor positions [39, 25, 26] and observation data with noise [40, 41, 26]. Prior field reconstruction techniques based on optimization methods and machine learning have primarily focused on addressing observation noise [36, 42, 24]. These approaches often do not consider the potential random vibration of sensor positions due to external influences. Consequently, the robustness and applicability of these methods have not been thoroughly demonstrated in the face of such real-world challenges.

Indeed, the current state of physical field reconstruction techniques confronts two primary challenges:

- **Linear ROM Methods and Sensitivity to Noise**: Traditional ROM techniques such as EIM [11, 12, 13], GEIM [43, 42], and POD predominantly rely on linear combinations of basis functions. Consequently, the reconstruction error tends to increase linearly with the escalation of sensor vibrations or observation noise. This limitation restricts their effectiveness in handling scenarios with substantial noise or vibration.

- **Large and Resource-Intensive Machine Learning Models**: Machine learning approaches like VCNN [24], NVT-ResNet [26], and others leverage the nonlinear capabilities of neural networks. However, these models often exhibit substantial size, memory, and disk space requirements. They can also be computationally intensive, which poses challenges for practical application. Additionally, they may not handle unstructured input data effectively. While the Voronoi tessellation relaxes the requirement for structured input, it tends to increase data size. The more recent VIVID [36] method, although innovative, demands complex design and may not be lightweight enough for certain applications, requiring the careful selection of multiple hyperparameters.

Figure 1 provides a conceptual overview of the EIM-NN approach. This algorithm harnesses the robust fitting capabilities of EIM [11] when dealing with the original data while also taking advantage of the nonlinear capabilities offered by neural networks. In the training phase, the linear EIM algorithm is employed to identify a low-dimensional subspace, and the basis for this subspace is stored. Subsequently, a nonlinear neural network is trained to determine the coefficients associated with each measurement within this subspace. These coefficients are then used to compute the reconstructed field. The algorithm's robustness is assessed during the testing phase. This can be done by introducing noise to the measurements or by simulating vibrations in the sensor positions of these measurements. Additionally, the paper introduces the EIM-TNN method, which extends the EIM-NN approach and enhances its robustness. This is accomplished by incorporating Tikhonov Regularization [44] along with the standard Mean Square Error loss function during training.

One of the distinguishing features of our approach is its simplicity. Unlike methods that rely on complex neural network architectures or other techniques [24, 36, 37], our method employs only two fully connected layers. This design choice results in a significantly smaller model size and reduces the time required for both training and testing phases. Specifically, our model size is a mere 4.95% of VCNN [24], and the training time amounts to just 1% of the VCNN's training duration.

Furthermore, our approach offers remarkable versatility, as both EIM and the fully connected layer are agnostic to whether the input data is structured or unstructured. Consequently, our model does not require preprocessing of unstructured data, making it applicable to a wide range of scenarios.

In addition, we rigorously evaluate the robustness of our model in the presence of sensor vibrations and noisy observations. The experimental results clearly demonstrate that our model outperforms VCNN, EIM, EIM-Tikhonov [42, 43] in terms of robustness. These findings underscore the practicality and effectiveness of our proposed approach in real-world applications.

In summary, we make the following main contributions in this study:

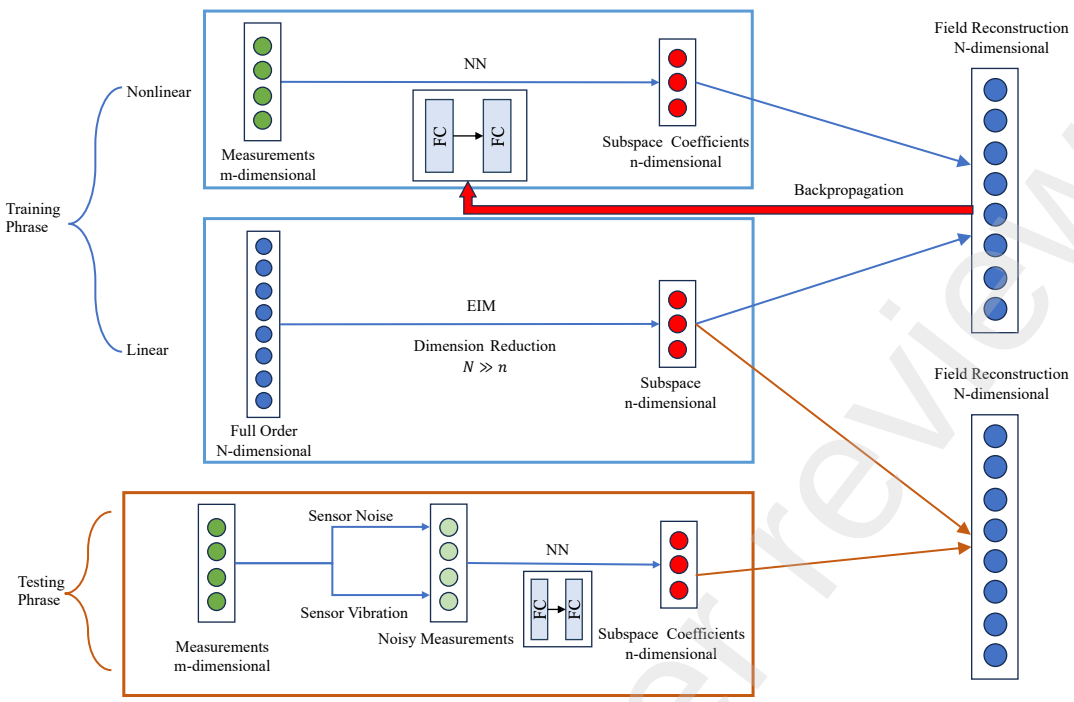


Figure 1: Conceptual scheme of EIM-NN.

- **Introduction of lightweight EIM-NN and EIM-TNN**: We propose EIM-NN and EIM-TNN, two lightweight algorithms integrated with neural networks. These algorithms optimize the neural network size without compromising model performance, thereby reducing training time and storage requirements.

- **Applicability to Unstructured Data**: Both EIM and fully-connected neural networks in our approach do not impose any structural requirements on the input data. This versatility allows our method to handle unstructured data seamlessly, eliminating the need for pre-processing.

- **Enhanced Robustness**: Our model exhibits remarkable robustness in the presence of sensor vibrations and noisy observations. This resilience is particularly valuable in real-world environments, where these factors often affect data quality. It enhances the practicality and effectiveness of our method in such scenarios.

The rest of this paper is organized as follows: In Section 2, we begin by introducing the fundamentals of EIM and EIM-Tikhonov. Subsequently, we present our novel EIM-NN method, which combines numerical methods and neural networks. We also introduce the concept of Tikhonov Regularization within the neural network's loss function, leading to the development of the TMSE loss function, and we name this extended method EIM-TNN. The Section 3 is dedicated to the empirical evaluation of our methods, assessing their performance in various scenarios to demonstrate their effectiveness and robustness. In

Section 4, we provide a comprehensive summary of the entire paper. Additionally, we offer insights into potential avenues for future research and developments in this field.

2. Methodology

We introduce mathematical notations, starting with $\mathcal{X}$, which represents a Hilbert space over $\Omega \in \mathbb{R}^{N_x}$. Typically, our primary focus is on 3D space, where $N_x = 3$. In this Hilbert space, we have an inner product denoted as $(\cdot, \cdot)$ and a corresponding norm $|\cdot|$. The parameter space is designated as $\mathcal{D}$, which is a subset of $\mathbb{R}^{N_\mu}$.

We define $\mathcal{S}$ as the solution manifold, expressed as $\mathcal{S} := \{u(\mu), \mu \in \mathcal{D}\}$, where u is a function defined on $\mathcal{D}$, and for each $\mu \in \mathcal{D}$, $u(\mu)$ belongs to the Hilbert space $\mathcal{X}$. Typically, we concentrate on the discrete form of u, which resides in a subset denoted as $\mathcal{U} \subset \mathbb{R}^N$. In the subsequent sections, our focus will remain exclusively on the discrete representation of u.

The EIM generates n-dimensional subspaces V_n of $\mathcal{S}$. We further introduce a dictionary of linear functionals $\mathcal{Y} \subset \mathcal{L}(\mathcal{X})$ with element $\ell^i \in \mathcal{Y}$ such that $\ell^i(u) \in \mathbb{R}$ for any $u \in \mathcal{X}$. In this article, $\ell^i(u)$ reflects the observation quantity of a given function u with respect to the sensor ℓ^i.

For any $u \in \mathcal{S}$, the interpolant $\tilde{u}(u) \in V_n$, meeting the specified interpolation property:

$$\tilde{u}(u) := \sum_{i=1}^{n} \alpha_u^i q^i \in V_n, \tag{2.1}$$

where $\{q^i\}_{i=1}^n$ are basis functions of V_n, and $\{\alpha_u^i\}_{i=1}^n$ are corresponding scalar coefficients defined by the following interpolation condition:

$$\ell^i(u) = \ell^i(\tilde{u}(u)) \quad \forall i = 1, \ldots, n, \tag{2.2}$$

with ℓ^i being linear functionals linked to the mass center x_i.

The GEIM [16, 17] advances the classical EIM by substituting point evaluations with a spectrum of continuous linear functionals. This modification not only mitigates the continuity constraints of functions to be interpolated but also bridges the gap towards practical data assimilation applications. In real-world scenarios, sensors exhibit a non-zero point spread, effectively functioning as local averages rather than discrete point evaluations, as represented by:

$$\ell^i(u) = \int_\Omega u(x) g_s(x - x_i) dx, \tag{2.3}$$

where g_s is a radial function influenced by a scalar $s > 0$, and satisfies $\int_\Omega g_s = 1$. This study predominantly concentrates on the applications and nuances of EIM.

In cases where ℓ^i are Dirac masses, they represent point evaluations at x_i:

$$\ell^i(u) = \delta_{x_i}(u) = u(x_i), \tag{2.4}$$

where δ_{x_i} represents the Dirac masses located at point x_i. In this paper, we utilize the Dirac mass for ℓ.

2.1. EIM greedy algorithm

The construction of interpolation spaces V_n and the choice of linear functionals in the EIM follow a recursive, greedy algorithm, similar to the one employed in classical EIM [11]. In this particular study, we adopt the GEIM approach to select basis functions. This choice favors GEIM over Dirac masses defined in Equation 2.4 due to the broader applicability and generality of the GEIM algorithm [43, 42]. The algorithm initiates with the selection of the first generating function and linear form, determined by:

$$|\ell^1(u^1)| = \arg\max_{u\in\mathcal{S},\ell\in\mathcal{Y}} |\ell(u)|. \tag{2.5}$$

Subsequently, the first basis function is defined as:

$$q^1 = \frac{u^1}{\ell^1(u^1)}, \tag{2.6}$$

leading to $V_1 = \text{span}\{q^1\}$. The corresponding interpolation operator $\tilde{u}_1 : \mathcal{S} \to V_1$ ensures that Equation 2.3 is satisfied for $n = 1$, i.e.,

$$\ell^1(u) = \ell^1(\tilde{u}_1(u)) = \alpha_u^1 \ell^1(q^1) \quad \forall u \in \mathcal{S}. \tag{2.7}$$

Assuming the established sets of interpolating basis functions $\{q^1, q^2, \ldots, q^{n-1}\}$ and the corresponding linear forms $\{\ell^1, \ell^2, \ldots, \ell^{n-1}\}$, we address the interpolation problem for $1 < n < n_{\max}$ via induction:

Find coefficients $\{\alpha_u^j\}_{j=1}^{n-1}$ satisfying:

$$\ell^i(u) = \sum_{j=1}^{n-1} \alpha_u^j B_{i,j} \quad \forall i = 1, \ldots, n-1, \tag{2.8}$$

where $B_{i,j}$ denotes the coefficients of the $(n-1)\times(n-1)$ matrix $B = (\ell^i(q^j))_{i,j\le n-1}$.

We note

$$\boldsymbol{\ell}_{n-1}(u) := \begin{pmatrix} \ell^1(u) \\ \vdots \\ \ell^{n-1}(u) \end{pmatrix} \in \mathbb{R}^{n-1},$$

$$\boldsymbol{\alpha}_{u,n-1} := \begin{pmatrix} \alpha_u^1 \\ \vdots \\ \alpha_u^{n-1} \end{pmatrix} \in \mathbb{R}^{n-1},$$

$$Q := \left(q^1, \ldots, q^{n-1}\right) \in \mathbb{R}^{N\times(n-1)}.$$

From Equation 2.8, it follows that:

$$\boldsymbol{\ell}_{n-1}(u) = \boldsymbol{\alpha}_{u,n-1}^T B. \tag{2.9}$$

Naturally, one should contemplate the least-squares minimization problem linked to the linear system of equations.

$$\min_{\boldsymbol{\alpha}_{u,n-1}\in\mathbb{R}^{n-1}} \|\boldsymbol{\ell}_{n-1}(u) - \boldsymbol{\alpha}_{u,n-1}^T B\|_2^2. \tag{2.10}$$

The solution to Equation 2.10 is given by:

$$\boldsymbol{\alpha}_{u,n-1} = (B^T B)^{-1} B^T \boldsymbol{\ell}_{n-1}(u). \tag{2.11}$$

So the interpolation of u is given by:

$$\tilde{u}_{n-1}(u) = Q\boldsymbol{\alpha}_{u,n-1} = Q(B^T B)^{-1} B^T \boldsymbol{\ell}_{n-1}(u). \tag{2.12}$$

The reconstruction residual is computed as:

$$r_{n-1}(u) = \tilde{u}_{n-1}(u) - u. \tag{2.13}$$

The reconstruction error is computed as:

$$\epsilon_{n-1}(u) = \|\tilde{u}_{n-1}(u) - u\|. \tag{2.14}$$

The n-th generating function and sensor are selected to maximize the reconstruction residual:

$$(\ell^n, u^n) = \arg\max_{u\in\mathcal{S},\ell\in\mathcal{Y}} |\ell(r_{n-1}(u))|. \tag{2.15}$$

The n-th basis function is defined as:

$$q^n = \frac{r_{n-1}(u^n)}{\ell^n(r_{n-1}(u^n))} = \frac{\tilde{u}_{n-1}(u^n) - u^n}{\ell^n(\tilde{u}_{n-1}(u^n) - u^n)}, \tag{2.16}$$

resulting in $V_n = \text{span}\{q^i, 1 \le i \le n\}$. The interpolation operator $\tilde{u}_n : \mathcal{S} \to V_n$ is expressed as:

$$\tilde{u}_n(u) = \sum_{i=1}^{n} \alpha_u^i q^i, \quad \forall u \in \mathcal{S}, \tag{2.17}$$

with coefficients $\{\alpha_u^i\}_{i=1}^n$ satisfying

$$\ell^j(u) = \sum_{i=1}^{n} \alpha_u^i B_{j,i} \quad \forall j = 1, \ldots, n, \tag{2.18}$$

where $B = (\ell^i(q^j))_{1\le i,j\le n}$.

In the context of $\mathcal{X}$ as a Banach space, the following properties are established [16]:

- The basis functions and linear functionals chosen through the greedy algorithm are linearly independent.
- The matrix B exhibits a lower triangular structure with a unit diagonal, guaranteeing its invertibility, and its non-diagonal entries fall within the interval $[-1, 1]$.

2.2. EIM-Tikhonov

As discussed in [43, 42], the (G)EIM faces instability challenges when dealing with data contaminated by random noise. To mitigate this issue, the prevailing solution is the Constrained Stabilized GEIM (CS-GEIM) [18, 45], which is widely acknowledged for its reliability. However, it's worth highlighting that this method entails formulating the stabilized problem as a constrained minimization, which necessitates solving a convex quadratic programming problem.

In contrast, the Tikhonov regularization method offers an alternative approach that involves solving a simpler linear system, resulting in reduced computational costs. Consequently, we have chosen to employ the Tikhonov regularization method for stabilizing EIM, as proposed in [43, 42].

In presence of noisy measurements:

$$\ell^{noise}(u) = \ell(u) + \delta \quad \delta := \text{noise vector.} \tag{2.19}$$

In [43], the author's proposed approximation to Equation 2.10, by using the Tikhonov regularization [44]:

$$\min_{\boldsymbol{\alpha}_{u,n} \in \mathbb{R}^n} \|\boldsymbol{\ell}_n^{noise}(u) - \boldsymbol{\alpha}_{u,n}^T B\|_2^2 + \zeta \|\boldsymbol{\alpha}_{u,n}^T C_n\|_2^2, \tag{2.20}$$

where

- $\boldsymbol{\ell}_n^{noise}(u)$ is the observation vector with noise.

$$\boldsymbol{\ell}_n^{noise}(u) = \begin{pmatrix} \ell^{1,noise}(u) \\ \vdots \\ \ell^{n,noise}(u) \end{pmatrix} \in \mathbb{R}^n.$$

- ζ is the penalty or regularization factor.
- $C_n \in \mathbb{R}^{n \times n}$ is the regularization matrix:

$$C_n = \begin{bmatrix} \frac{1}{\epsilon_1} & 0 & \cdots & 0 \\ 0 & \frac{1}{\epsilon_2} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{\epsilon_n} \end{bmatrix}. \tag{2.21}$$

Equation 2.20 represents a standard approach for regularization using Tikhonov regularization. In this equation, the first term minimizes the observation error, while the second term serves as the regularization component. Several criteria are available for the choice of the regularization parameter ζ, as discussed in [42]. Such as the discrepancy principle [46], the generalized cross-validation (GCV) [47], the L-curve criterion [48, 49, 50]. Solving Equation 2.20 can be achieved as follows:

$$\boldsymbol{\alpha}_{u,n} = (B^T B + \zeta C_n^T C_n)^{-1} B^T \boldsymbol{\ell}_n^{noise}(u). \tag{2.22}$$

So the Tikhonov regularization based interpolation of u in the (n)-th iteration is given by:

$$\tilde{u}_n(u) = Q\boldsymbol{\alpha}_{u,n} = Q(B^TB + \zeta C_n^T C_n)^{-1} B^T \boldsymbol{\ell}_n^{noise}(u). \tag{2.23}$$

2.3. EIM-NN

After the basis functions $\{q_j\}_{j=1}^n$ is found by EIM, solving the corresponding scalar coefficients $\boldsymbol{\alpha}_{u,n}$ by Equation 2.22 is very time-consuming, and relying solely on linear equations to solve the weights leads to the problem of poor robustness to observation noise. Therefore, we propose here EIM-NN to solve the corresponding scalar coefficients $\boldsymbol{\alpha}_{u,n}$ by a nonlinear neural network.

Initially, we derive the basis functions $\{q_i\}_{i=1}^n$ within the n-dimensional space V_n by employing the EIM-Tikhonov method, as discussed in Section 2.2. We note:

$$Q = \left(\ q_1, \ldots, q_n \ \right) \in \mathbb{R}^{N \times n}.$$

Moving forward, we introduce the observation vector $y \in \mathbb{R}^m$ into the neural network, where m represents the number of sensors. The corresponding full order field for y is denoted as u. From the neural network, we obtain the coefficients $\boldsymbol{\alpha} \in \mathbb{R}^n$ for that specific observation.

The reconstructed field is then formulated as follows:

$$\tilde{u}(u) = Q\boldsymbol{\alpha}. \tag{2.24}$$

We denote $f_{\boldsymbol{w}} : \mathbb{R}^m \to \mathbb{R}^n$ to represent the neural network, where $\boldsymbol{w}$ represents the network parameters. Our training process can be expressed as:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \left\| u - Q f_{\boldsymbol{w}}(y) \right\|_2. \tag{2.25}$$

In our approach, we utilize a neural network comprising only two fully connected layers (FC) with the Tanh activation function. The architecture is detailed in Table 2.1. This design yields a lightweight structure, ensuring minimal computational overhead and memory usage.

Table 2.1: Neural Network Architecture

Layer	Output Shape	Activation
Input Layer	[162]	-
FC1	[512]	Tanh
FC2	[Basis Number n]	-

When utilizing neural networks directly for field reconstruction, the challenge is to create a mapping function from a m-dimensional space to a N-dimensional space, where N is typically much larger than m. However, EIM establishes the mapping from the discrete space $\mathcal{U} \subset \mathbb{R}^N$ to a n-dimensional subspace V_n within the solution manifold $\mathcal{S}$. Consequently, EIM-NN only needs to implement the mapping within this subspace V_n. In other words, it constructs a mapping function from a m-dimensional space to a n-dimensional space, where

$n \leq m$. Since the goal is to map to the low-dimensional manifold space, the EIM-NN method imposes minimal requirements on neural networks. It can achieve this with just two fully connected layers, which characterizes it as a lightweight approach.

Additionally, as our neural network solely employs fully connected layers, it does not require structured input data, unlike CNNs [36, 37]. This flexibility means that our approach can accommodate observations or sensors at arbitrary, unstructured locations, significantly expanding its applicability.

Furthermore, compared to directly using the EIM to compute the coefficients, our method leverages the nonlinearity and robustness of neural networks, resulting in improved performance when dealing with noise and positional variations in the input data. Additionally, unlike Equation 2.10 and Equation 2.20 for solving the coefficients, the neural network avoids the need to compute matrix inverses, reducing computational overhead and enhancing speed, especially when dealing with large datasets. Finally, the use of neural networks overcomes issues related to singular matrices when solving matrix inverses and can be seamlessly integrated with various types of ROM algorithms.

To mitigate the diminished influence of neural networks on solving the fit for the original data, we introduce a new loss function called TMSE (Tikhonov Mean Square Error) during the training process. We refer to the entire architecture as EIM-TNN

Specifically, we incorporate an additional loss term for the basis coefficients alongside the original Mean Square Error. The formula is expressed as follows:

$$\boldsymbol{w} = \arg\min_{\boldsymbol{w}} \|u - Q f_{\boldsymbol{w}}(y)\|_2 + \lambda \|C_n(f_{\boldsymbol{w}}(y) - \boldsymbol{\alpha}_y)\|_2. \tag{2.26}$$

Here, λ serves as the penalty or regularization factor. $f_{\boldsymbol{w}}(y)$ represents the output basis coefficients of the neural network, while $\boldsymbol{\alpha}_y$ denotes the basis coefficients determined by the EIM-Tikhonov method. C_n is the regularization matrix used in the Tikhonov regularization approach for EIM defined in Equation 2.21. We refer to the first term in Equation 2.26 as the Mean Squared Error (MSE) and the second term as the Tikhonov Error.

The additional regularization term we've introduced serves to constrain the similarity between the basis coefficients generated by the neural network and those obtained through the EIM linear equations. It effectively leverages the Tikhonov regularization matrix, ensuring that the neural network fits the reconstructed physical field without causing significant deviations from the EIM-derived coefficients.

Furthermore, the incorporation of the Tikhonov regularization matrix bolsters the robustness of the weights against noise in the observation data. This, in combination with the nonlinearity inherent in the neural network, further strengthens the overall robustness of the model.

2.4. Analyse of the robustness of neural networks

In the forthcoming analysis, we shall delve into the rationale for utilizing neural networks to bolster the resilience of the EIM algorithm.

Let us commence by examining a snapshot, denoted as u, together with the respective measurements denoted as ℓ, and the measurements subject to noise δ, $\ell^{noise} = \ell + \delta$. As per Equation 2.22, we can deduce the EIM-Tikhonov reconstructions as follows:

$$\tilde{u}_{ET}(u) = ET \circ \ell. \tag{2.27}$$

Here, the operator ET signifies a linear mapping from $\mathbb{R}^m$ to $\mathbb{R}^N$, which is articulated as follows in Equation 2.28:

$$\begin{aligned} ET: \quad & \mathbb{R}^m \to \mathbb{R}^N \\ & \ell \to Q(B^T B + \zeta C_{n-1}^T C_{n-1})^{-1} B^T \ell. \end{aligned} \tag{2.28}$$

It is noteworthy that the term $Q(B^T B + \zeta C_{n-1}^T C_{n-1})^{-1} B^T$ within Equation 2.28 results in a matrix, confirming the linearity of the mapping function ET. Consequently, we can compute the difference between the reconstructions of the measurement and the noisy measurement as follows:

$$\Delta u_{ET}(u) = ET \circ \delta. \tag{2.29}$$

Conversely, the neural network utilized in both EIM-NN and EIM-TNN can also be expressed as a mapping function, denoted as NN. The EIM-TNN reconstructions can be defined as:

$$\tilde{u}_{NN}(u) = NN \circ \ell, \tag{2.30}$$

where NN is computed layer by layer as specified in Equation 2.31:

$$\begin{aligned} NN: \quad & \mathbb{R}^m \to \mathbb{R}^N \\ & \ell \to Q \times (FC2 \circ \tanh \circ FC1 \circ \ell). \end{aligned} \tag{2.31}$$

Notably, the inclusion of the tanh activation function within Equation 2.31 renders NN a nonlinear mapping function. Consequently, we can determine the difference between the reconstructions of the measurement and the noisy measurement as follows:

$$\Delta u_{NN}(u) = NN \circ (\ell + \delta) - NN \circ \ell. \tag{2.32}$$

Numerous studies have highlighted the noise-filtering capabilities of neural networks due to their inherent nonlinearity [51, 52, 53, 54]. Therefore, in this context, our objective is to assess and juxtapose the impact of input noise on the output of the algorithm when employing both the EIM-Tikhonov (ET) and EIM-TNN (NN) methods. Specifically, we aim to compare the quantities $\Delta u_{ET}(u)$ and $\Delta u_{NN}(u)$. As illustrated in Figure 2, the schematic representation demonstrates that the NN operator encompasses an "acceptable noise boundary". Within this boundary, the input noise undergoes filtration by the neural network, resulting in a considerably smaller impact on the output of the NN operator compared to the ET operator, as depicted in the figure. The comparison between $\Delta u_{NN}(u)$ and $\Delta u_{ET}(u)$ within the right-hand box is evident.

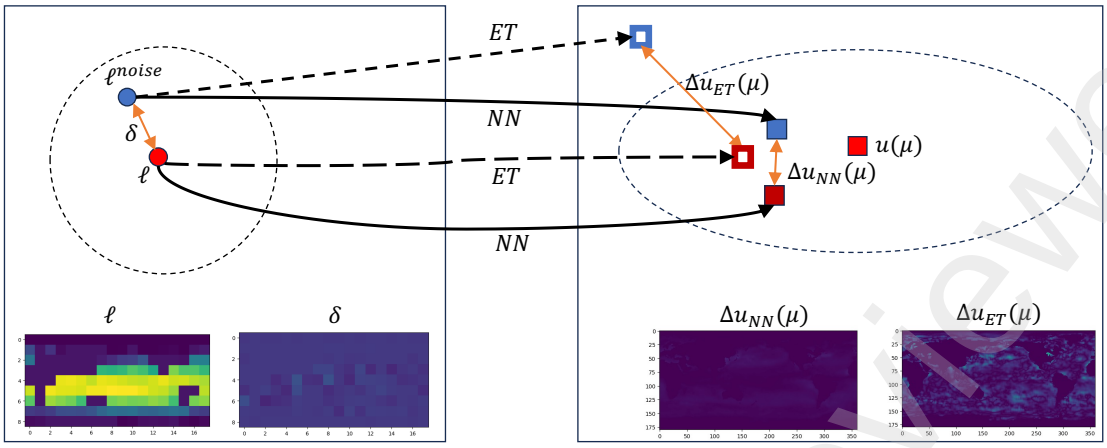


Figure 2: Schematic Comparison of the ET Operator and the NN Operator. The left box signifies the observation space, while the right box represents the reconstructed field space. The dashed circle on the left delineates the "acceptable noise boundary" of the NN operator, where noise within this boundary undergoes mapping by the NN operator to the corresponding "acceptable reconstruction boundary" within the dashed area on the right. This mapping ensures that the reconstructed field maintains limited fluctuations.

In addition, we also compare the error performance of the EIM-Tikhonov algorithm and the EIM-TNN algorithm when the δ is different. To simulate this noisy measurements, we introduce priority Gaussian noise to the measurements. We start with our original measurements denoted as $\{\ell^i\}_{i=1}^m$, where m represents the number of sensors. Then, we introduce Gaussian noise using the following formula:

$$\ell^{i,noise} = (1 + \sigma\varepsilon_i)\,\ell^i, \forall i \in \{1, ..., m\}\,, \tag{2.33}$$

where $\ell^{i,noise}$ signifies the noisy measurements, where σ represents the sensor's noise scale, and ε_i is a randomly generated number drawn from the standard Gaussian distribution $\mathcal{N}(0, 1)$.

In this analysis, we have chosen 20 snapshots and introduced noise with $\sigma = 0.05$ and $\sigma = 0.15$ in Equation 2.33 to simulate cases with small and large noise, respectively. We have calculated the relative L_2 errors between these snapshots and the real values using Equation 3.1. The results have been sorted in ascending order based on the error values obtained from the EIM-Tikhonov algorithm. Figure 3 illustrates these results.

Notably, it is evident that as the noise level increases, the error of the NN operator remains relatively stable, whereas the error associated with the ET operator exhibits a significant increase. Furthermore, when confronted with the same noise conditions, the error of the NN operator is considerably smaller than that of the ET operator. For a more comprehensive examination of the test results, we refer readers to Section 3.6 for detailed findings.

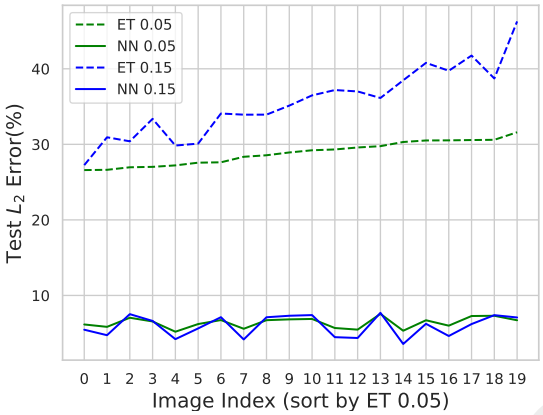


Figure 3: L_2 percentage error in test set. "ET 0.05" denotes the use of the EIM-Tikhonov algorithm with a test set containing $\sigma = 0.05$ noise. "NN 0.15" denotes the use of the EIM-TNN algorithm with a test set containing $\sigma = 0.15$ noise.

3. Numerical results and robustness analysis

3.1. Experiment setups

To demonstrate the practical applicability of both EIM-NN and EIM-TNN in real-world scenarios, we utilize the NOAA's sea surface temperature dataset. This dataset contains field data based on a one-degree grid with a spatial resolution of 360×180, derived from satellite and ship observations, and does not require an in-depth understanding of the underlying governing equations. Our data partitioning strategy involves allocating 60% of the dataset for training, 10% for validation, and the remaining 30% for testing purposes. Additionally, we select 162 (9×18) sensors and assess the algorithm's performance under scenarios where the sensor positions are both structured and unstructured, as detailed in Section 3.4.

To evaluate the performance of our models, we conduct a comparative analysis involving EIM-NN and EIM-TNN alongside three other models: VCNN [24], a complete machine learning-based method, EIM [11], and EIM-Tikhonov [43], two ROMs. In order to mitigate result uncertainty, we conduct each experiment a total of 15 times. In doing so, we employ a random process to generate distinct training, validation, and test sets for each iteration. The final result is computed as the average of the test set errors across the 15 runs.

In the case of EIM, our default number of basis functions is set at 70. We also assess the robustness of our algorithm by experimenting with different numbers of basis functions in Section 3.3.

The optimizer utilized for training our neural network model is Adam [55], with the initial learning rate set to 0.001. We employ a learning rate adjustment strategy using StepLR, which reduce the learning rate by a factor of 10 every 100 epochs. For the loss function, we use mean square error(MSE) for EIM-NN and TMSE (See Equation 2.26) for EIM-TNN. In all the following experiments, we conduct 200 epochs of training (VCNN, EIM-NN and EIM-TNN), and the model parameters that exhibit the best performance on the validation set are saved.

When assessing the final model's performance, we evaluat it using the following two normalized loss metrics on the test set:

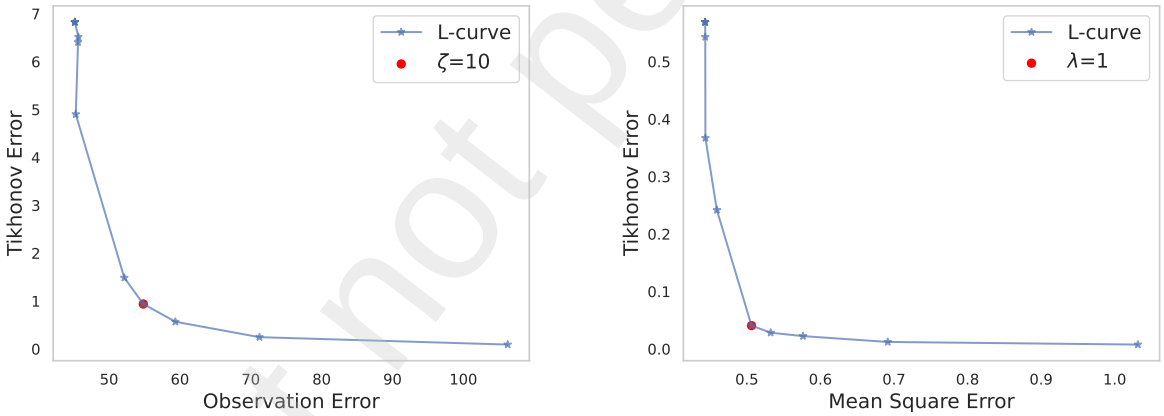
$$L_2(u) := \frac{1}{N}\sum_{i=1}^{N}\frac{\|u_{r,i}-u_i\|_2}{\|u_i\|_2}, \tag{3.1}$$

$$L_\infty(u) := \frac{1}{N}\sum_{i=1}^{N}\frac{\|u_{r,i}-u_i\|_\infty}{\|u_i\|_\infty}, \tag{3.2}$$

here, u_r represents the field reconstruction, and u represents the true value. The variable N denotes the number of samples in the dataset.

Additionally, for the hyperparameter ζ in EIM-Tikhonov (See Equation 2.20), we utilize the relative L_2 error defined in Equation 3.1 on the validation set to compute the L-curve [48, 49, 50].This approach is mirrored in the selection of hyperparameters λ for EIM-TNN (See Equation 2.26) as well.

Figure 4a and Figure 4b presented below illustrate the L-curve for the hyperparameter ζ in EIM-Tikhonov and the L-curve for the hyperparameter λ in EIM-TNN. We choose the turning point of the L-curve as our hyperparameter. Consequently, we adopt the values $\zeta = 10$ and $\lambda = 1$ for all subsequent experiments.



(a) L-curve for ζ in EIM-Tikhonov

(b) L-curve for λ in EIM-TNN

Figure 4: L-curves for two hyperparameters on the validation set with 70 basis functions

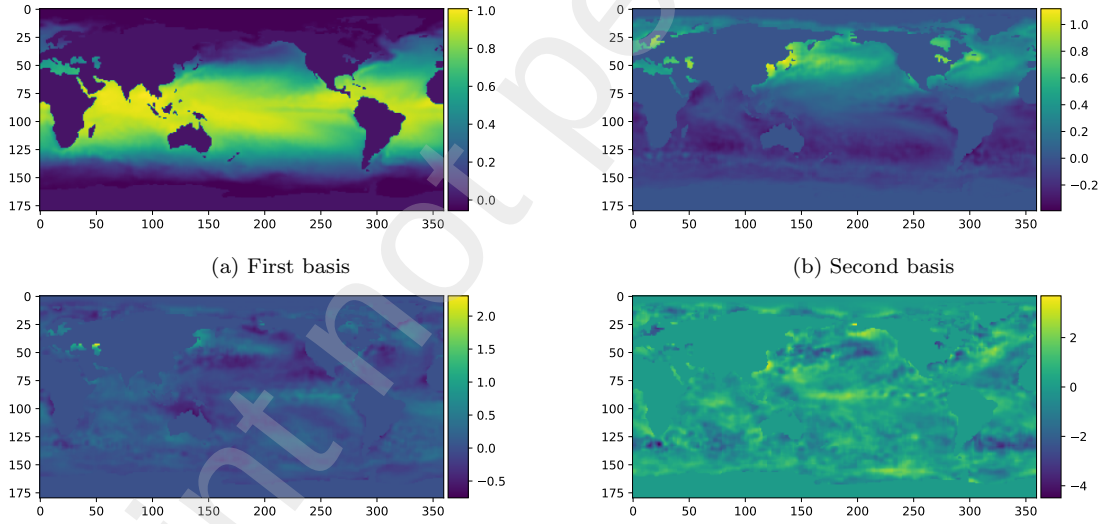
3.2. Performance with different basis selection methods

In this subsection, we initiate a comparison between the performance of using EIM and the widely employed Proper Orthogonal Decomposition (POD) method [56, 57, 58, 59] as the basis selection method. Given a dataset $S = \{u_1, u_2, \ldots, u_M\}$, sampled according to M different parameters, we commence by computing the Singular Value Decomposition (SVD) of this matrix as follows:

$$S = U\Sigma V^T = \sum_{i=1}^{r} \sigma_i U_i V_i^T. \tag{3.3}$$

Here, r signifies the rank of S, U represents an $N \times r$ matrix with orthonormal columns denoted as U_i, V denotes an $M \times r$ matrix with orthonormal columns represented by V_i, and Σ is an $r \times r$ diagonal matrix with diagonal entries in descending order: $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r \geq 0$. The subspace V_n is spanned by $\{U_1, \ldots, U_n\}$, $n \leq r$.

First, in Figure 5, we present the visualization results of EIM with a basis of 70. Specifically, we depict the outcomes for the 1st, 2nd, 3rd, and 70th bases. Next, we compare the performance of the four algorithms while utilizing the same dimension of basis functions, selected by EIM and POD, respectively. Figure 6 displays the L_2 error and L_∞ error of EIM and POD on the test set with 70 basis functions. It is evident that the bases extracted by POD consistently underperform those obtained via EIM in all scenarios, corroborating the findings reported in [60], which assert that EIM provides more efficient and reliable subspaces compared to POD. Additionally, EIM possesses the capability to select locations with the largest errors within a restricted area by defining $\mathcal{Y}$ in a given sensor feasible area, whereas POD is constrained to selecting bases across the entire space. Consequently, EIM offers superior utility in this context. As a result, we opt to utilize EIM for extracting the substrate in this paper.



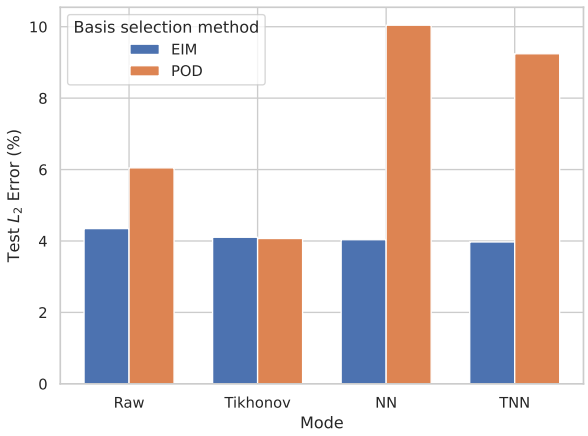
(a) First basis

(b) Second basis

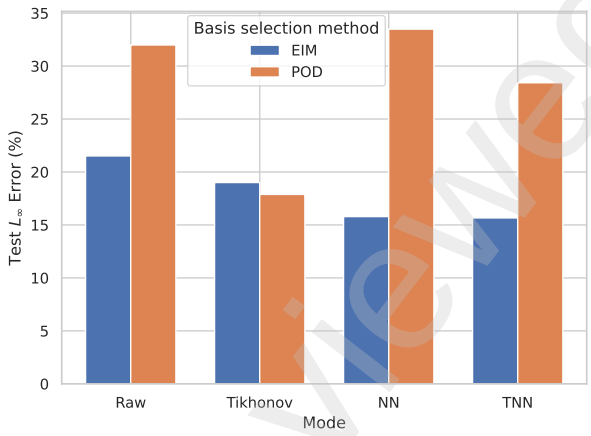
(c) Third basis

(d) Last basis

Figure 5: Visualization diagram of basis. Shown are the results for the 1st, 2nd, 3rd, and 70th bases for a total of 70 basis functions



(a) L_2 percentage errors



(b) L_∞ percentage errors

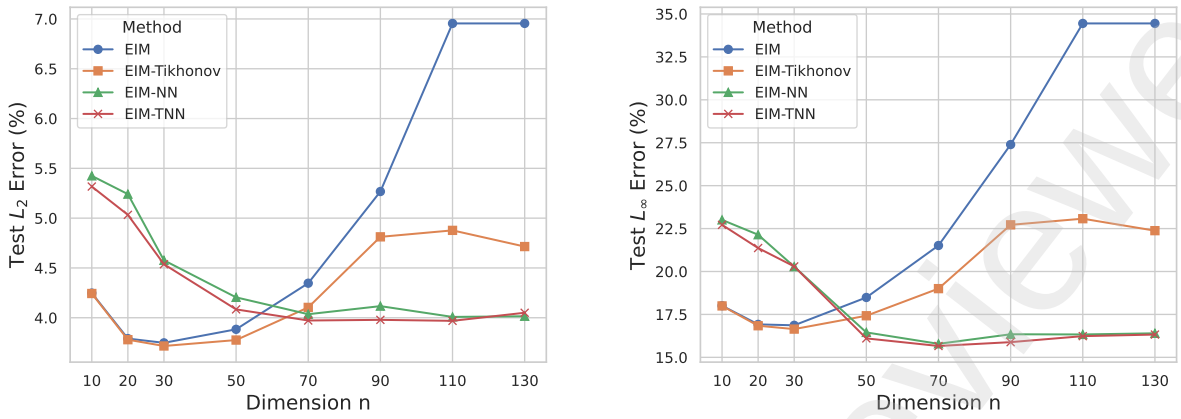
Figure 6: Errors on the test set for various basis selection methods.The term "Raw" represents solving the basis coefficients by minimizing the reconstructed observation error. "Tikhonov" signifies the addition of the Tikhonov regularity term to the observation error during solving the basis coefficients. "NN" denotes the use of the Mean Squared Error (MSE) loss function to train the neural network for basis coefficient determination after selecting the basis. Finally, "TNN" indicates the utilization of the Tikhonov MSE (TMSE) loss function to train the neural network for basis coefficient determination following basis selection.

3.3. Performance with different number of basis functions

To evaluate the performance of our proposed algorithms in fitting the original data and to assess the robustness concerning the number of basis functions (i.e., the subspace dimension n), we conducted experiments using five algorithms with varying values of n (10, 20, 30, 50, 70, 90, 110, and 130). VCNN's performance remains consistent across different n values as it is independent of this dimension.

Figure 7 visually compare the error sizes for different algorithms using curves. It's important to note that due to the significant errors of VCNN ($L_2 = 11.6739\%$, $L_\infty = 31.4100\%$), it is not included in the figures.

We observe a consistent reduction in test set errors for both EIM-NN and EIM-TNN as the dimensionality n increases. However, it's noteworthy that EIM experiences a gradual increase in error rates. Additionally, EIM-TNN demonstrates a slightly superior performance compared to EIM-NN. For the subsequent experiments, we fix the number of basis functions at 70. In Figure 8, we also present the input data for the four EIM algorithms, the reconstructed physical field, the real physical field, and their associated error values when the number of basis functions is set to 70.



(a) L_2 percentage errors

(b) L_∞ percentage errors

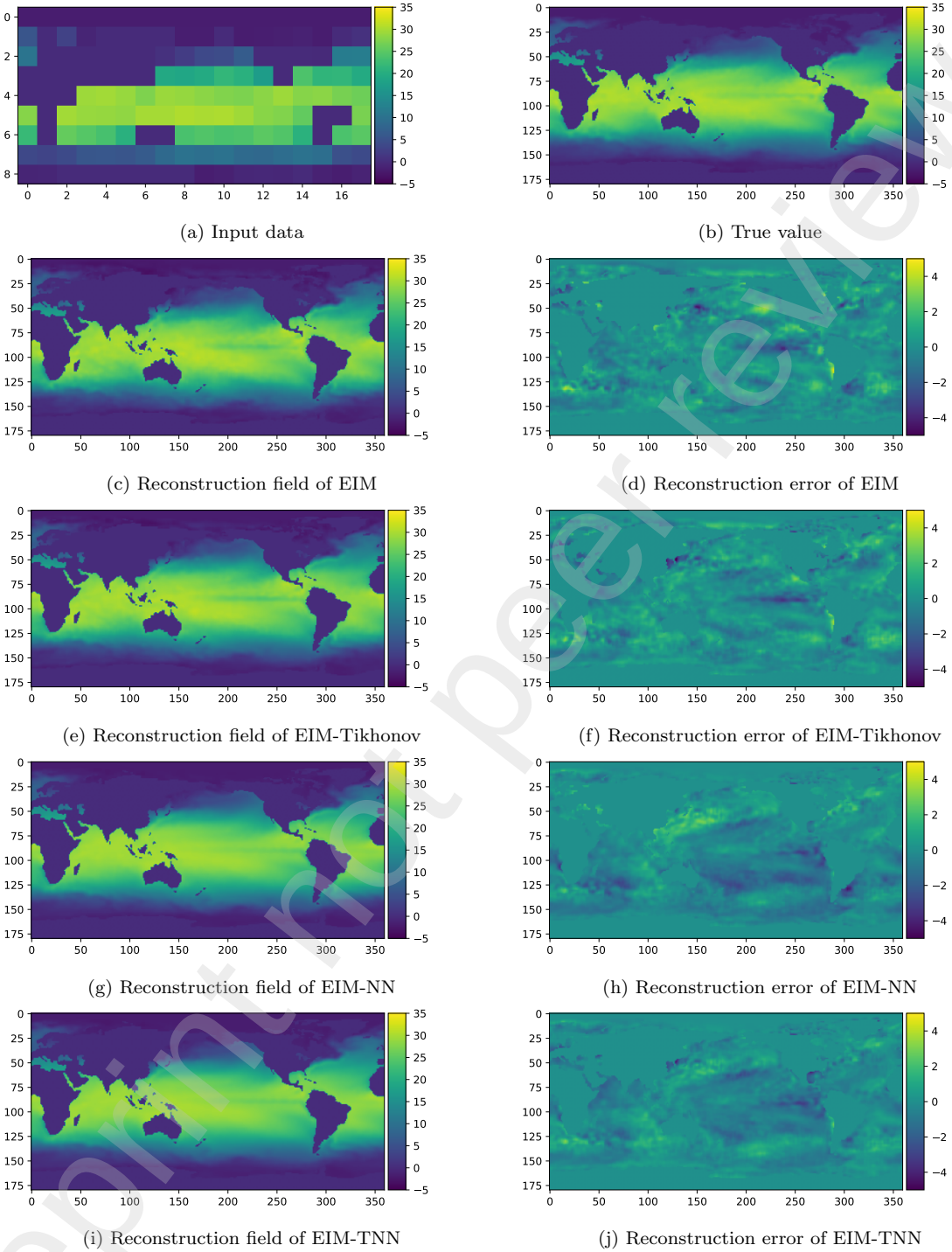
Figure 7: Errors on the test set for various algorithms with different subspace dimensions n.

3.4. Performance with unstructured observation

When implementing production environments, achieving structured sensor placement, such as a grid configuration, can be challenging. This presents a limitation for methods like Convolutional Neural Networks (CNNs), which rely on structured input data. To address this issue, the Voronoi tessellation method [24] is employed to expand unstructured input data across the entire physical field before subjecting it to CNN processing. However, this approach increases the input data's size, consequently demanding more extensive memory space and storage capacity.

In contrast, our proposed EIM-NN and EIM-TNN algorithms, which amalgamate the attributes of EIM and NN, eliminate the need for structured data, enabling them to efficiently process unstructured data. Thus, in this subsection, we compare the performance of these five algorithms in handling both structured and unstructured data. Figure 9a and 9b depict the sensor locations as red dots within the figures.

Table 3.1 displays the performance of the five algorithms when subjected to structured and unstructured observations. It is noteworthy that our proposed EIM-NN and EIM-TNN consistently deliver robust and stable performance in both scenarios. In contrast, VCNN, which heavily relies on spatial data processing algorithms such as CNNs, displays a significantly lower performance when confronted with unstructured (random) observations, in contrast to its performance with structured observations.



(a) Input data

(b) True value

(c) Reconstruction field of EIM

(d) Reconstruction error of EIM

(e) Reconstruction field of EIM-Tikhonov

(f) Reconstruction error of EIM-Tikhonov

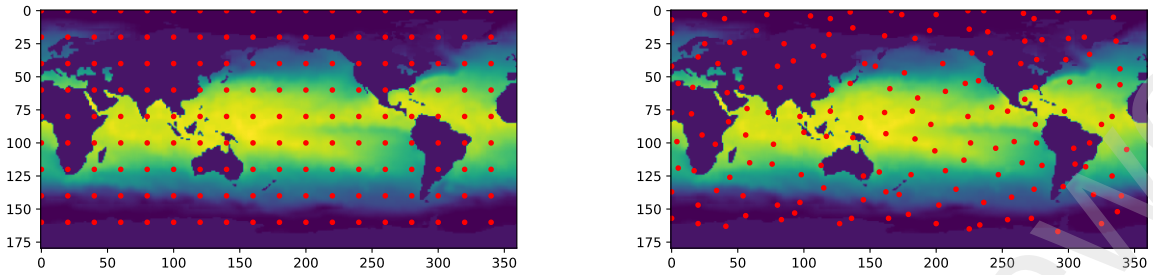
(g) Reconstruction field of EIM-NN

(h) Reconstruction error of EIM-NN

(i) Reconstruction field of EIM-TNN

(j) Reconstruction error of EIM-TNN

Figure 8: Examples of input data, true value, reconstruction field, and reconstruction error for four algorithms



(a) Structured observation

(b) Unstructured observation

Figure 9: Structured and unstructured observation. The red circles in the figure are sensor locations.

Table 3.1: Errors on the test set for various algorithms with structured and unstructured observation.

Observation	Structured		Unstructured	
Algorithm	$L_2(\%)$	$L_\infty(\%)$	$L_2(\%)$	$L_\infty(\%)$
VCNN	11.6739	31.4100	21.8453	44.8453
EIM	4.3469	21.5147	3.9030	17.8475
EIM-Tikhonov	4.1039	19.0022	**3.7334**	17.1967
EIM-NN	4.0355	15.7867	4.0197	16.1301
EIM-TNN	**3.9723**	**15.6616**	3.9374	**15.4396**

3.5. Performance with vibrations

In real-world scenarios, it is essential to acknowledge that sensor positions are often subject to variability, such as vibrations induced by factors like ocean currents. In this subsection, we assess the performance of each algorithm when confronted with vibrations in sensor positions.

To simulate sensor position vibrations, we consider that each sensor can potentially occupy any location within a square region with a side length of (2 * vibration range + 1), centered around its initial position. Figure 10 provides an illustrative example of sensor vibration, the right side, displayed in Figure 10, offers an enlarged view of the red box from the left-side figure. Assuming a vibration range of 3, the blue square box in the right-side figure delineates the area within which the sensor may be situated, representing the range of potential sensor positions.

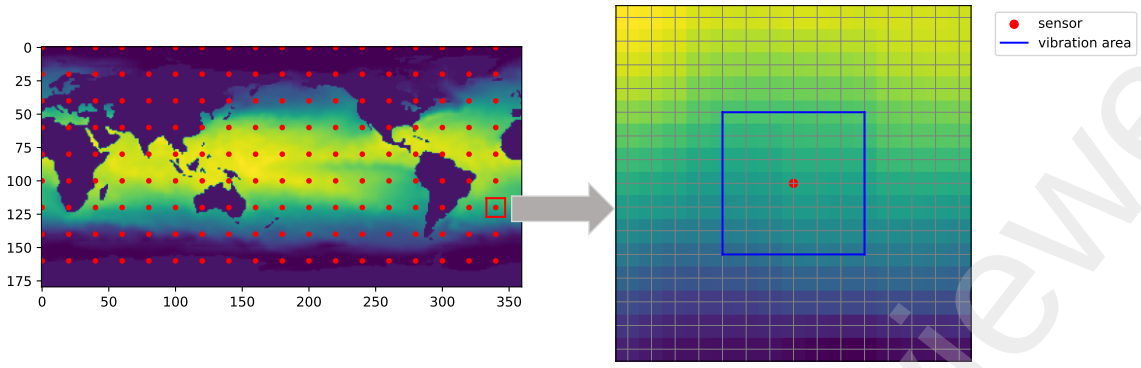


Figure 10: Sensor Vibration. The image on the right is an enlarged view of the image enclosed within the red box on the left. In the enlarged image, the sensor can be relocated within any of the 49 points depicted within the blue box when the vibration range is set to 3.

To evaluate the robustness of various algorithms against sensor vibrations, we employ data without vibrations for training and validation, while the test set consists of observations taken after sensor vibrations have occurred. Table 3.2 present the test set errors for the five algorithms across different test set vibration ranges of 1, 2, and 3.

Notably, it is evident that EIM-NN and EIM-TNN exhibit the least susceptibility to vibrations, with EIM-TNN demonstrating the highest level of robustness against these vibrations.

Table 3.2: Errors on the test set for various algorithms with different vibration ranges.

Vibration Range	1		2		3	
Algorithm	$L_2(\%)$	$L_\infty(\%)$	$L_2(\%)$	$L_\infty(\%)$	$L_2(\%)$	$L_\infty(\%)$
VCNN	14.8846	68.2645	17.1331	82.3524	20.7052	92.9537
EIM	12.8846	63.9176	22.0887	114.3137	25.9085	132.9381
EIM-Tikhonov	8.8153	41.8638	15.5764	76.3162	18.3673	89.2798
EIM-NN	4.8996	19.5719	5.3641	23.1176	5.6675	25.0487
EIM-TNN	**4.2699**	**18.5807**	**4.8232**	**21.8977**	**5.0823**	**23.0339**

3.6. Performance with noises

Furthermore, in real industrial production environments, observation data is inevitably subject to the influence of noise. Consequently, the ability of algorithms to maintain robust performance when confronted with noisy observation data significantly impacts their practical applicability in such settings.

To simulate this scenario, we introduce priority noise to the observation data using Equation 2.33. In this context, we have selected σ values of 0.05, 0.10, and 0.15 for training and validation using noise-free data, while introducing observations with noise. Table 3.3 presents the performance of different algorithms across varying noise ratios.

It is evident that both EIM-NN and EIM-TNN consistently demonstrate exceptional robustness, surpassing the performance of other algorithms. Notably, when subjected to an

increase in noise from 0.05 to 0.15, the L_2 error of EIM-TNN experiences only a marginal increase, rising from 4.0009% to 4.4512%, representing a mere 11.25% error increment compared to $\sigma = 0.05$. In stark contrast, the L_2 error of EIM undergoes a substantial expansion, increasing from 8.3057% to 21.5991%, resulting in a notable 160.05% error increment compared to $\sigma = 0.05$. These results underscore the robustness of our proposed algorithm when confronted with noisy observations.

Table 3.3: Errors on the test set for various algorithms with different noise ratios.

Noise Ratio	0.05		0.1		0.15	
Algorithm	$L_2(\%)$	$L_\infty(\%)$	$L_2(\%)$	$L_\infty(\%)$	$L_2(\%)$	$L_\infty(\%)$
VCNN	12.7664	66.4818	16.6027	76.6265	19.0568	87.6947
EIM	8.3057	41.2756	14.7694	73.7707	21.5991	108.1451
EIM-Tikhonov	5.8247	25.8695	9.8987	43.8478	14.3057	63.5160
EIM-NN	4.3073	16.1680	4.4886	17.0983	4.9553	18.3555
EIM-TNN	**4.0009**	**15.7166**	**4.1014**	**16.2512**	**4.4512**	**17.0875**

3.7. Computational Resources and Costs

In our experiments, we utilized the A4000 GPU with 16 GB of memory capacity. Both the training and testing phases of our model were executed on a single GPU. Table 3.4 shows the model and data storage comparison of different algorithms. Model storage relates to the size of the model's parameters, while data storage pertains to the size of the input data used for model training, validation, or inference.

In the context of model storage, VCNN [24] consists of a total of 682,417 parameters and occupies a memory footprint of 5.40 MB. In contrast, the fully connected model used in EIM-NN and EIM-TNN contains only 119,366 parameters and occupies a memory footprint of 0.93 MB. Notably, our model demonstrates a minimal 0.8% increase in model storage compared to EIM, making it a lightweight addition to the EIM framework.

In terms of data storage, the input data size required by the methods in the EIM class is significantly smaller, constituting only 0.15% of the input data size needed by VCNN. This discrepancy arises because VCNN necessitates the expansion of observation data across the entire physical field using the Voronoi tessellation method to handle unstructured data. This results in substantially larger input data, which in turn can reduce both training and inference speeds to some extent.

Table 3.4: Model and Data Storage Comparison. The term "Model storage" represents the storage space required for model parameters. The term "Data Storage" represents the storage space occupied by observation data input to the model.

	Model Storage	Data Storage	
Algorithm	-	Total Storage	Per Image
VCNN	5.40MB	1.58GB	0.82MB
EIM	115.00MB	2.40MB	1.25KB
EIM-Tikhonov	115.00MB	2.40MB	1.25KB
EIM-NN	115.93MB	2.40MB	1.25KB
EIM-TNN	115.93MB	2.40MB	1.25KB

Table 3.5 provides insights into the computational time required by different algorithms during both the training and testing phases. In the training phase, the EIM class of methods demonstrates significantly reduced computational demands compared to VCNN. Specifically, EIM-NN and EIM-TNN expend approximately one percent of the training time needed by VCNN for each image, while EIM and EIM-Tikhonov require just a fraction of that, around one-thousandth of VCNN's training time. Even though EIM-NN and EIM-TNN involve additional neural network training segments, the supplementary time spent remains acceptable owing to the compact size of the network.

Moving to the testing phase, the inference time of EIM-like methods on each image is approximately half that of VCNN. Furthermore, EIM-NN and EIM-TNN only marginally increase the inference time by roughly 10% when compared to EIM. These findings highlight the computational efficiency and practicality of the EIM class of methods, particularly in terms of training and inference times, while still delivering competitive performance.

Table 3.5: Computation Time in Training and Testing Phases. The term "Total Time" encompasses the cumulative time invested in either the training or testing set, while "Per Image" signifies the mean time allocated per image.

	Computational Time (s)			
	Training Phrase		Test Phrase	
Algorithm	Total Time	Per Image	Total Time	Per Image
VCNN(200 epochs)	4650.60	4.05	7.11	1.23×10^{-2}
EIM	6.26	5.45×10^{-3}	2.78	4.84×10^{-3}
EIM-Tikhonov	7.91	6.89×10^{-3}	2.81	4.89×10^{-3}
EIM-NN(200 epochs)	48.53	4.22×10^{-2}	3.12	5.43×10^{-3}
EIM-TNN(200 epochs)	51.55	4.49×10^{-2}	3.14	5.47×10^{-3}

4. Conclusions and future works

Many contemporary field reconstruction techniques, predominantly based on CNNs, encounter significant challenges when dealing with unstructured data, necessitating additional

preprocessing steps and consuming substantial storage space. Additionally, numerical optimization methods often struggle in the presence of observation noise and sensor vibrations. In this paper, we introduce two innovative field reconstruction algorithms: EIM-NN and EIM-TNN.

EIM-NN takes advantage of the subspace identified by EIM and utilizes neural networks to determine the corresponding coefficients. Additionally, we introduce a novel loss function called TMSE, which incorporates Tikhonov Regularization, and introduce the EIM-TNN method to further enhance robustness. Our neural network architecture comprises just two fully connected layers, ensuring versatility in handling a broad spectrum of unstructured data while maintaining a lightweight and efficient design.

Exploiting the nonlinearity of neural networks, our algorithm exhibits remarkable robustness in the presence of noise and detector vibrations. Experimental results on the NOAA dataset consistently demonstrate the algorithm's performance with both structured and unstructured data. Notably, when faced with sensor vibrations with a range of 3, our algorithm's L_2 error increases only from 3.9723% to 5.0823%, whereas the conventional EIM-Tikhonov algorithm with the addition of Tikhonov's regularization term increases from 4.1039% to 8.3673%. When confronted with 15% observation noise, our L_2 error is 4.4512%, compared to 14.3057% for the EIM-Tikhonov algorithm, underscoring the robustness of our approach.

Furthermore, our algorithm requires a mere 0.15% of the input data size necessary for a CNN-dependent VCNN method, with a model size that is only 1/5 of it, resulting in an exceptionally lightweight model. In terms of training duration, our model demands only approximately 10% more time in the inference phase compared to EIM, which is considered acceptable.

Future research in this field presents several promising avenues for exploration. Firstly, there is a need to establish and demonstrate the robustness and stability of employing neural networks for computing basis coefficients. This can be achieved through more comprehensive theoretical analysis and rigorous proofs. Secondly, extending the application of our methodology to larger datasets, such as 3D physical field reconstruction, holds great potential for real-world applications. Traditional Convolutional Neural Networks (CNNs) often encounter challenges related to model size and data limitations in such scenarios. Exploring ways to address these challenges and apply our approach effectively to larger-scale problems can be a fruitful direction. Lastly, in light of the growing importance of uncertainty quantification (UQ) in the field of scientific machine learning, future research can focus on integrating our findings with UQ methodologies [61, 62, 63, 64]. Such integration has the potential to enhance the robustness and reliability of predictions, particularly when dealing with uncertainties in diverse scientific and engineering domains.

Acknowledgments

This research is partially sponsored by the Natural Science Foundation of Shanghai (No.23ZR1429300), and the Innovation Funds of CNNC (Lingchuang Fund).

[1] Alfio Quarteroni, Andrea Manzoni, and Federico Negri. *Reduced basis methods for partial differential equations: an introduction*, volume 92. Springer, 2015.

[2] Yvon Maday and Einar M Rønquist. A reduced-basis element method. *Journal of scientific computing*, 17:447–459, 2002.

[3] Denise Degen, Karen Veroy, and Florian Wellmann. How uncertainty quantification and reduced order modeling change our model understanding. In *AGU Fall Meeting Abstracts*, volume 2020, pages T015–0002, 2020.

[4] Alfio Quarteroni, Gianluigi Rozza, et al. *Reduced order methods for modeling and computational reduction*, volume 9. Springer, 2014.

[5] Jan S Hesthaven, Gianluigi Rozza, Benjamin Stamm, et al. *Certified reduced basis methods for parametrized partial differential equations*, volume 590. Springer, 2016.

[6] Denise Degen, Karen Veroy, and Florian Wellmann. Certified reduced basis method in geosciences: Addressing the challenge of high-dimensional problems. *Computational Geosciences*, 24:241–259, 2020.

[7] Yvon Maday. Reduced basis method for the rapid and reliable solution of partial differential equations. In *International Congress of Mathematicians*, volume 3, pages 1255–1270. Citeseer, 2006.

[8] Annalisa Buffa, Yvon Maday, Anthony T Patera, Christophe Prud'homme, and Gabriel Turinici. A priori convergence of the greedy algorithm for the parametrized reduced basis method. *ESAIM: Mathematical modelling and numerical analysis*, 46(3):595–603, 2012.

[9] Anindya Chatterjee. An introduction to the proper orthogonal decomposition. *Current science*, pages 808–817, 2000.

[10] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[11] Maxime Barrault, Yvon Maday, Ngoc Cuong Nguyen, and Anthony T Patera. An 'empirical interpolation'method: application to efficient reduced-basis discretization of partial differential equations. *Comptes Rendus Mathematique*, 339(9):667–672, 2004.

[12] Yvon Maday, Ngoc Cuong Nguyen, Anthony T Patera, and George SH Pau. A general, multipurpose interpolation procedure: the magic points. 2007.

[13] Francesco AB Silva, Stefano Lorenzi, and Antonio Cammi. An empirical interpolation method for two-dimensional vector fields and vector measurements. *International Journal for Numerical Methods in Engineering*, 122(15):3733–3748, 2021.

[14] J-P Argaud, Bertrand Bouriquet, F De Caso, Helin Gong, Yvon Maday, and Olga Mula. Sensor placement in nuclear reactors based on the generalized empirical interpolation method. *Journal of Computational Physics*, 363:354–370, 2018.

[15] Helin Gong, Zhang Chen, and Qing Li. Generalized empirical interpolation method with h1 regularization: Application to nuclear reactor physics. *Frontiers in Energy Research*, 9:804018, 2022.

[16] Yvon Maday, Olga Mula, Anthony T Patera, and Masayuki Yano. The generalized empirical interpolation method: stability theory on hilbert spaces with an application to the stokes equation. *Computer Methods in Applied Mechanics and Engineering*, 287:310–334, 2015.

[17] Yvon Maday, Olga Mula, and Gabriel Turinici. Convergence analysis of the generalized empirical interpolation method. *SIAM Journal on Numerical Analysis*, 54(3):1713–1731, 2016.

[18] JP Argaud, B Bouriquet, H Gong, Yvon Maday, and Olga Mula. Stabilization of (g) eim in presence of measurement noise: application to nuclear reactor physics. In *Spectral and High Order Methods for Partial Differential Equations ICOSAHOM 2016: Selected Papers from the ICOSAHOM conference, June 27-July 1, 2016, Rio de Janeiro, Brazil*, pages 133–145. Springer, 2017.

[19] Jinlong Fu, Dunhui Xiao, Dongfeng Li, Hywel R Thomas, and Chenfeng Li. Stochastic reconstruction of 3d microstructures from 2d cross-sectional images using machine learning-based characterization. *Computer Methods in Applied Mechanics and Engineering*, 390:114532, 2022.

[20] Helin Gong, Sibo Cheng, Zhang Chen, Qing Li, César Quilodrán-Casas, Dunhui Xiao, and Rossella Arcucci. An efficient digital twin based on machine learning svd autoencoder and generalised latent assimilation for nuclear reactor physics. *Annals of nuclear energy*, 179:109431, 2022.

[21] Rui Fu, Dunhui Xiao, Ionel Michael Navon, Fangxin Fang, and Liang Yang. A nonlinear non-intrusive reduced order model of fluid flow by auto-encoder and self-attention deep learning methods. *International Journal for Numerical Methods in Engineering*, 2023.

[22] Jinlong Fu, Shaoqing Cui, Song Cen, and Chenfeng Li. Statistical characterization and reconstruction of heterogeneous microstructures using deep neural network. *Computer Methods in Applied Mechanics and Engineering*, 373:113516, 2021.

[23] Yang Liu, Rui Hu, Adam Kraus, Prasanna Balaprakash, and Aleksandr Obabko. Data-driven modeling of coarse mesh turbulence for reactor transient analysis using convolutional recurrent neural networks. *Nuclear Engineering and Design*, 390:111716, 2022.

[24] Kai Fukami, Romit Maulik, Nesar Ramachandra, Koji Fukagata, and Kunihiko Taira. Global field reconstruction from sparse sensors with voronoi tessellation-assisted deep learning. *Nature Machine Intelligence*, 3(11):945–951, 2021.

[25] Helin Gong, Han Li, Dunhui Xiao, and Sibo Cheng. Reactor field reconstruction from sparse and movable sensors with voronoi tessellation-assisted convolutional neural networks. *Nuclear Science and Techniques*, 2023.

[26] Han Li, Jialiang Lu, Hongjun Ji, Lizhan Hong, and Helin Gong. A noise and vibration tolerant resnet for field reconstruction with sparse sensor. *Communications in Computational Physics*, 2023.

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[28] Wenhui Zhou, Jiangwei Shi, Yongjie Hong, Lili Lin, and Ercan Engin Kuruoglu. Robust dense light field reconstruction from sparse noisy sampling. *Signal Processing*, 186:108121, 2021.

[29] Alan J Geer. Learning earth system models from observations: machine learning or data assimilation? *Philosophical Transactions of the Royal Society A*, 379(2194):20200089, 2021.

[30] Peter Binev, Albert Cohen, Wolfgang Dahmen, Ronald DeVore, Guergana Petrova, and Przemyslaw Wojtaszczyk. Data assimilation in reduced modeling. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):1–29, 2017.

[31] Albert Cohen, Wolfgang Dahmen, Ronald DeVore, Jalal Fadili, Olga Mula, and James Nichols. Optimal reduced model algorithms for data-based state estimation. *SIAM Journal on Numerical Analysis*, 58(6):3355–3381, 2020.

[32] Albert Cohen, Wolfgang Dahmen, Olga Mula, and James Nichols. Nonlinear reduced models for state and parameter estimation. *SIAM/ASA Journal on Uncertainty Quantification*, 10(1):227–267, 2022.

[33] Albert Cohen, Matthieu Dolbeault, Olga Mula, and Agustin Somacal. Nonlinear approximation spaces for inverse problems. *Analysis and Applications*, 21(01):217–253, 2023.

[34] Caterina Buizza, César Quilodrán Casas, Philip Nadler, Julian Mack, Stefano Marrone, Zainab Titus, Clémence Le Cornec, Evelyn Heylen, Tolga Dur, Luis Baca Ruiz, et al. Data learning: Integrating data assimilation and machine learning. *Journal of Computational Science*, 58:101525, 2022.

[35] Thomas Frerix, Dmitrii Kochkov, Jamie Smith, Daniel Cremers, Michael Brenner, and Stephan Hoyer. Variational data assimilation with a learned inverse observation operator. In *International Conference on Machine Learning*, pages 3449–3458. PMLR, 2021.

[36] Sibo Cheng, Che Liu, Yike Guo, and Rossella Arcucci. Efficient deep data assimilation with sparse observations and time-varying sensors. *Journal of Computational Physics*, 496:112581, 2024.

[37] Ashesh Chattopadhyay, Ebrahim Nabizadeh, Eviatar Bach, and Pedram Hassanzadeh. Deep learning-enhanced ensemble-based data assimilation for high-dimensional nonlinear dynamical systems. *Journal of Computational Physics*, 477:111918, 2023.

[38] Yoeri E Boink and Christoph Brune. Learned svd: solving inverse problems via hybrid autoencoding. *arXiv preprint arXiv:1912.10840*, 2019.

[39] Paolo Vinai, Huaiqian Yi, Christophe Demaziere, Amélie Rouchon, and Andrea Zoia. On the simulation of neutron noise induced by vibrations of fuel pins in a fuel assembly. *Annals of Nuclear Energy*, 181:109521, 2023.

[40] Yu Yang, Helin Gong, Qihong Yang, Yangtao Deng, Qiaolin He, and Shiquan Zhang. On the uncertainty analysis of the data-enabled physics-informed neural network for solving neutron diffusion eigenvalue problem. *arXiv preprint arXiv:2303.08455*, 2023.

[41] Alessandro Nordio, Carla-Fabiana Chiasserini, and Emanuele Viterbo. Performance of linear field reconstruction techniques with noise and uncertain sensor locations. *IEEE Transactions on signal Processing*, 56(8):3535–3547, 2008.

[42] Carolina Introini, Simone Cavalleri, Stefano Lorenzi, Stefano Riva, and Antonio Cammi. Stabilization of generalized empirical interpolation method (geim) in presence of noise: A novel approach based on tikhonov regularization. *Computer Methods in Applied Mechanics and Engineering*, 404:115773, 2023.

[43] Helin Gong. *Data assimilation with reduced basis and noisy measurement: Applications to nuclear reactor cores*. PhD thesis, Sorbonne université, 2018.

[44] Andrei Nikolaevich Tikhonov. On the solution of ill-posed problems and the method of regularization. In *Doklady akademii nauk*, volume 151, pages 501–504. Russian Academy of Sciences, 1963.

[45] H Gong, JP Argaud, B Bouriquet, Y Maday, and O Mula. Monitoring flux and power in nuclear reactors with data assimilation and reduced models. 2017.

[46] Otmar Scherzer. The use of morozov's discrepancy principle for tikhonov regularization for solving nonlinear ill-posed problems. *Computing*, 51(1):45–60, 1993.

[47] JH Venter and JLJ Snyman. A note on the generalised cross-validation criterion in linear model selection. *Biometrika*, 82(1):215–219, 1995.

[48] Per Christian Hansen. Analysis of discrete ill-posed problems by means of the l-curve. *SIAM review*, 34(4):561–580, 1992.

[49] Per Christian Hansen and Dianne Prost O'Leary. The use of the l-curve in the regularization of discrete ill-posed problems. *SIAM journal on scientific computing*, 14(6):1487–1503, 1993.

[50] Åke Björck. *Numerical methods for least squares problems*. SIAM, 1996.

[51] T Fechner. Nonlinear noise filtering with neural networks: comparison with weiner optimal filtering. In *1993 Third International Conference on Artificial Neural Networks*, pages 143–147. IET, 1993.

[52] Xiao-Ping Zhang. Thresholding neural network for adaptive noise reduction. *IEEE transactions on neural networks*, 12(3):567–584, 2001.

[53] Feras N Hasoon, Jabar H Yousif, Nebras N Hasson, and Abd Rahman Ramli. Image enhancement using nonlinear filtering based neural network. *Journal of Computing*, 3(5):171–176, 2011.

[54] José R Dorronsoro, Vicente López, CS Cruz, and JA Siguenza. Autoassociative neural networks and noise filtering. *IEEE Transactions on Signal Processing*, 51(5):1431–1438, 2003.

[55] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[56] Joseph H Citriniti and William K George. Reconstruction of the global velocity field in the axisymmetric mixing layer utilizing the proper orthogonal decomposition. *Journal of Fluid Mechanics*, 418:137–166, 2000.

[57] Genghui Jiang, Ming Kang, Zhenwei Cai, Han Wang, Yingzheng Liu, and Weizhe Wang. Online reconstruction of 3d temperature field fused with pod-based reduced order approach and sparse sensor data. *International Journal of Thermal Sciences*, 175:107489, 2022.

[58] Nicola Demo, Marco Tezzele, and Gianluigi Rozza. A non-intrusive approach for the reconstruction of pod modal coefficients through active subspaces. *Comptes Rendus Mécanique*, 347(11):873–881, 2019.

[59] Helin Gong, Yingrui Yu, and Qing Li. Reactor power distribution detection and estimation via a stabilized gappy proper orthogonal decomposition method. *Nuclear Engineering and Design*, 370:110833, 2020.

[60] Helin Gong, Zhang Chen, Yvon Maday, and Qing Li. Optimal and fast field reconstruction with reduced basis and limited observations: Application to reactor core online monitoring. *Nuclear Engineering and Design*, 377:111113, 2021.

[61] Zongren Zou, Xuhui Meng, and George Em Karniadakis. Uncertainty quantification for noisy inputs-outputs in physics-informed neural networks and neural operators. *arXiv preprint arXiv:2311.11262*, 2023.

[62] Masaki Morimoto, Kai Fukami, Romit Maulik, Ricardo Vinuesa, and Koji Fukagata. Assessments of epistemic uncertainty using gaussian stochastic weight averaging for fluid-flow regression. *Physica D: Nonlinear Phenomena*, 440:133454, 2022.

[63] Apostolos F Psaros, Xuhui Meng, Zongren Zou, Ling Guo, and George Em Karniadakis. Uncertainty quantification in scientific machine learning: Methods, metrics, and comparisons. *Journal of Computational Physics*, 477:111902, 2023.

[64] Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, et al. A survey of uncertainty in deep neural networks. *arXiv preprint arXiv:2107.03342*, 2021.