MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
8 September 2025

REVISED
23 March 2026

ACCEPTED FOR PUBLICATION
7 April 2026

PUBLISHED
24 April 2026

PAPER

A local squared Wasserstein-2 method for efficient reconstruction of models with uncertainty

Mingtao Xia[1,2,*] and **Qijing Shen**[3]

[1] Department of Mathematics, University of Houston, Houston, TX 77204, United States of America
[2] School of Mathematics, University of Birmingham, Birmingham B15 2TT, United Kingdom
[3] Nuffield Department of Medicine, University of Oxford, Oxford OX3 7BN, United Kingdom
[*] Author to whom any correspondence should be addressed.

E-mail: mxia4@uh.edu, xiamingtao97@g.ucla.edu and qijing.shen@ndm.ox.ac.uk

Keywords: uncertainty quantification, Wasserstein distance, inverse problem, linear regression, neural network

Abstract

In this paper, we propose a local squared Wasserstein-2 (W_2) method to solve the inverse problem of reconstructing models with uncertain latent variables or parameters. A key advantage of our approach is that it does not require prior information on the distribution of the latent variables or parameters in the underlying models. Instead, through minimizing our proposed local squared W_2 loss function, linear regression models or neural networks can be directly trained to efficiently reconstruct the distributions of the output associated with different inputs based on empirical distributions of observation data. We demonstrate the effectiveness of our proposed method across several uncertainty quantification tasks, including linear regression with coefficient uncertainty, training neural networks with weight uncertainty, and reconstructing ordinary differential equations with a latent random variable.

1. Introduction

Models incorporating uncertainty have been extensively utilized across various fields. For example, models incorporating measurement errors are widely used [1–3]. Additionally, models involving latent unobserved variables are frequently employed in uncertainty quantification (UQ) [4], with applications in stock price modeling [5] and image processing [6]. In bioinformatics, when analyzing individual susceptibility to infection to get infected by certain types of genotype-influenced diseases, dimensionality reduction techniques are often employed to eliminate genes with minor relevance to the disease [7]. In these models, instead of offering a single deterministic output, the output is sampled from a distribution influenced by the input.

In this work, we study the following model with uncertainty (random field):

$$\boldsymbol{y}(\boldsymbol{x};\omega) := \boldsymbol{f}(\boldsymbol{x},\omega)\,,\ \boldsymbol{x}\in D\subseteq\mathbb{R}^n \tag{1}$$

where $\boldsymbol{f}(\cdot;\cdot):\mathbb{R}^n\times\Omega\to\mathbb{R}^d$ is a continuous function in $\boldsymbol{x}$; $\omega\in\Omega$ is a latent random variable in a sample space Ω. D is a bounded measurable subset of $\mathbb{R}^n$. Only $\boldsymbol{x}$ is observed (referred to as the input). Therefore, $\boldsymbol{y}(\boldsymbol{x};\omega)$ follows a distribution determined by $\boldsymbol{x}$. Inverse problems of UQ arise in many real-world applications, including medical imaging, geophysical inversion, and astronomical imaging. In such problems, one seeks to quantify uncertainty in the outcome from indirect and noisy observations, which may come from measurement noise, limited resolution, or other sources of observational error. For example, in x-ray CT, the observed data $\boldsymbol{y}$ are projection measurements (sinograms), and uncertainty may arise from reduced radiation dose, sparse-view acquisition, limited-angle scanning, and measurement noise [8, 9]. In geophysical inversion, $\boldsymbol{y}$ may represent seismic recordings or pressure data, whose uncertainty is affected by band-limited acquisition, ambient noise, incomplete spatial coverage, and the intrinsic nonuniqueness of subsurface inversion [10, 11].

Our goal is to reconstruct a model:

$$\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega}) := \hat{\boldsymbol{f}}(\boldsymbol{x},\hat{\omega})\,,\ \boldsymbol{x}\in D\subseteq\mathbb{R}^n \tag{2}$$

as an approximation to equation (1) in the sense that the distribution of $\boldsymbol{y}(\boldsymbol{x};\omega)$ can be matched by the distribution of $\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})$ for the same input $\boldsymbol{x}$. In equation (2), $\hat{\boldsymbol{f}}$ is a stochastic neural network with weight uncertainties (detailed in figure 1) and $\hat{\omega}\in\hat{\Omega}$ is another random variable (which is the uncertain weights in the neural network) in another sample space $\hat{\Omega}$ (we do not require $\hat{\Omega}$ to be the same as Ω). To our knowledge, there are few existing methods that directly reconstruct the distribution of $\boldsymbol{y}(\boldsymbol{x};\omega)$ for different $\boldsymbol{x}$ in equation (1) without requiring a specific form of $\boldsymbol{f}$ or a prior distribution of ω.

1.1. Related work

The reconstruction of models with uncertainty from data has received significant research interest [12, 13]. Traditional methods for reconstructing models with uncertainty primarily focus on parameter inference. These approaches typically start by assuming a specific model form with several unknown parameters and then aim to infer the mean and variance of these parameters from the data [14, 15]. Recent advancements in Bayesian methods, especially the Bayesian neural network (BNN) [16, 17], make it possible to learn the posterior distribution of unknown and uncertain model parameters given their prior distributions as well as observed data.

The Wasserstein distance, which measures the discrepancy between two probability distributions [18, 19], has recently become a popular tool in UQ [20]. For example, entropy-regularized Wasserstein distance methods have been proposed for multi-label prediction problems [21] and imaging applications [22]. Additionally, the Wasserstein generative adversarial network (WGAN) [23] has been applied to various tasks, such as image generation [24, 25] and generating the distribution of solutions to partial differential equations with latent parameters [23]. However, training a generative adversarial network model can be challenging and computationally expensive [26]. Also, a large number of samples could be required to reconstruct a distribution using the WGAN method.

On the other hand, our local squared W_2 loss is related to the recently introduced notion of conditional Wasserstein distance, which measures an expected Wasserstein discrepancy between conditional distributions and has been studied in, e.g. [27, 28]. In this sense, our construction may be interpreted as a local empirical approximation of a conditional transport discrepancy over neighborhoods in the input space when the neighborhood size is small. However, unlike the conditional Wasserstein formulations in [27, 28], our method does not explicitly define transport on the full joint law via restricted conditional couplings, but instead works with neighborhood-based empirical approximations. Therefore, when $\boldsymbol{x}$ is high-dimensional while $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ are low-dimensional in equations (1) and (2), numerically evaluating the conditional Wasserstein distance could be expensive and inaccurate due to the high dimensionality, but calculating our local squared W_2 distance avoids directly solving conditional Wasserstein distance especially when $\boldsymbol{x}$ is high-dimensional while $\boldsymbol{y}$ is low-dimensional.

1.2. Our contributions

In our paper, we propose and analyze a novel local squared Wasserstein-2 (W_2) method to reconstruct a model equation (2) for approximating the uncertainty model equation (1). Our main contributions are:

1. we propose and analyze a local squared W_2 loss function for reconstructing uncertainty models in UQ, which could be efficiently evaluated by empirical distributions from a finite number of observations by using a 'neighborhood' technique;
2. unlike the Bayesian methods or previous Wasserstein-distance-based methods [29], our method does not require a prior distribution of ω nor does it necessarily require an explicit form of $\boldsymbol{f}$ in equation (1);
3. our proposed method can apply to a broad range of UQ tasks such as linear regression with coefficient uncertainty, training a stochastic neural network with weight uncertainty, and reconstructing an ordinary differential equation (ODE) with latent uncertain parameters. Specifically, the neural network with uncertainty models that our proposed local squared W_2 method could train has the capability of approximating unknown random fields (e.g. $\{F_{\boldsymbol{x}}\},\boldsymbol{x}\in D\subseteq\mathbb{R}^d$ such that given an $\boldsymbol{x}$, $F_{\boldsymbol{x}}\in\mathbb{R}^n$ is a random variable) well under mild conditions with fewer than ten hidden layers, and the number of neurons in such neural networks scales linearly with the dimensionality of inputs and outputs [30, appendix H]. Therefore, unlike the WGAN or diffusion models, which usually require deep

Table 1. Summary of commonly used notations throughout the paper.

Symbol	Description
$\boldsymbol{x}$	Input variable in $\mathbb{R}^n$.
$\boldsymbol{y}$	Target variable in the ground-truth uncertainty model equation (1) in $\mathbb{R}^d$.
$\hat{\boldsymbol{y}}$	Output of the approximate uncertainty model equation (2) in $\mathbb{R}^d$.
ω ($\hat{\omega}$)	Uncertain latent variables in the ground-truth (approximate) model.
δ	The size of the neighborhood for $\boldsymbol{x}$.
N	The number of total training samples.
$N(\boldsymbol{x},\delta)$	The number of samples $(\boldsymbol{x}_i,\boldsymbol{y}_i)$ satisfying $\lvert\boldsymbol{x}_i-\boldsymbol{x}\rvert_x\leqslant\delta$. $\lvert\cdot\rvert_x$ is some norm for the input $\boldsymbol{x}\in\mathbb{R}^n$.
$\mu_{\boldsymbol{x}}$ ($\hat{\mu}_{\boldsymbol{x}}$)	The probability measure of $\boldsymbol{y}(\boldsymbol{x};\omega)$ ($\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})$) given $\boldsymbol{x}$.
$\mu^{\mathrm{e}}_{\boldsymbol{x},\delta}$ ($\hat{\mu}^{\mathrm{e}}_{\boldsymbol{x},\delta}$)	The empirical probability measure of $\boldsymbol{y}(\tilde{\boldsymbol{x}};\omega)$ ($\hat{\boldsymbol{y}}(\tilde{\boldsymbol{x}};\hat{\omega})$) conditioned on $\lvert\tilde{\boldsymbol{x}}-\boldsymbol{x}\rvert_x\leqslant\delta$.
$\pi_{\mu,\hat{\mu}}$	A coupling measure of μ and $\hat{\mu}$ whose marginal distributions coincide with μ and $\hat{\mu}$, respectively.
$W_2(\mu,\hat{\mu})$	The Wasserstein-2 distance between μ and $\hat{\mu}$.
$\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$	The average W_2 distance between two random fields $\boldsymbol{y}(\boldsymbol{x};\omega)$ and $\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})$ (defined in equation (8)).
$\tilde{W}^{2,\mathrm{e}}_{2,\delta}(\boldsymbol{y},\hat{\boldsymbol{y}})$	Our proposed local squared W_2 loss function for $\boldsymbol{y}(\boldsymbol{x};\omega)$ and $\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})$ (defined in equation (10)).

neural networks [31, 32], our proposed local squared W_2 approach provides a promising way of directly training simple neural networks to effectively reconstruct uncertainty models, especially lower-dimensional ones.

The organization of this paper is as follows: in section 2, we introduce our novel local squared W_2 method, which relies on minimizing a local squared W_2 loss function. In section 3, we test the efficacy of our proposed method across several different UQ tasks and make comparisons with some other loss functions and benchmark methods in UQ. In section 4, we summarize our results and discuss potential future directions. Notations that are often used throughout this paper are summarized in table 1.

2. A local squared W_2 method for UQ

2.1. A local squared W_2 loss function

First, we formally define the W-distance between two n-dimensional random variables.

Definition 2.1. For two random variables $\boldsymbol{y},\hat{\boldsymbol{y}}\in\mathbb{R}^n$, we assume that

$$\mathbb{E}\left[\|\boldsymbol{y}\|^2\right]<\infty,\ \ \mathbb{E}\left[\|\hat{\boldsymbol{y}}\|^2\right]<\infty. \tag{3}$$

In the following, the norm $\|\cdot\|$ denotes the l^2 norm of a vector. We denote probability distributions of $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ by μ and $\hat{\mu}$, respectively. The W_2-distance $W_2(\mu,\hat{\mu})$ is defined as

$$W_2(\mu,\hat{\mu}):=\inf_{\pi\in\Pi_{\mu,\hat{\mu}}}\mathbb{E}_{(\boldsymbol{y},\hat{\boldsymbol{y}})\sim\pi}\left[\|\boldsymbol{y}-\hat{\boldsymbol{y}}\|^2\right]^{\frac{1}{2}}. \tag{4}$$

In equation (4),

$$\Pi(\mu,\hat{\mu}):=\{\pi\mid\pi\,\text{is a probability measure defined on}\,\mathcal{B}(\mathbb{R}^n\times\mathbb{R}^n)\,\text{with marginals}\,\mu\,\text{and}\,\hat{\mu}\}, \tag{5}$$

where $\mathcal{B}(\mathbb{R}^n\times\mathbb{R}^n)$ denotes the Borel σ-algebra associated with $\mathbb{R}^n\times\mathbb{R}^n$.

To simplify our analysis, we make the following assumptions.

Assumption 2.2 We assume that the following conditions hold for the uncertainty model equation (1) and the approximate model equation (2).

1. We assume that $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ in equations (1) and (2) are uniformly bounded:

$$\|\boldsymbol{y}\|\leqslant\sqrt{M},\ \ \|\hat{\boldsymbol{y}}\|\leqslant\sqrt{M}. \tag{6}$$

2. We assume that $\boldsymbol{f}$ and $\hat{\boldsymbol{f}}$on the right-hand sides (RHSs) of equations (1) and (2) are uniformly Lipschitz continuous on $\boldsymbol{x}$, i.e. there exists $L<\infty$ such that for any $\omega \in \Omega, \hat{\omega} \in \hat{\Omega}$:

$$\|\boldsymbol{f}(\boldsymbol{x};\omega) - \boldsymbol{f}(\hat{\boldsymbol{x}};\omega)\| \leqslant L|\boldsymbol{x}-\hat{\boldsymbol{x}}|_x,\ \|\hat{\boldsymbol{f}}(\boldsymbol{x};\hat{\omega}) - \hat{\boldsymbol{f}}(\hat{\boldsymbol{x}};\hat{\omega})\| \leqslant L|\boldsymbol{x}-\hat{\boldsymbol{x}}|_x,\ \forall x, \hat{x} \in D, \tag{7}$$

where $|\cdot|_x$ is a norm for $\boldsymbol{x} \in \mathbb{R}^d$.

3. The random variable ω is independent of $\boldsymbol{x}$; the random variable $\hat{\omega}$ is also independent of $\boldsymbol{x}$.

We consider the 'weighted average discrepancy' between the two random fields $\boldsymbol{y}(\boldsymbol{x};\omega) = \boldsymbol{f}(\boldsymbol{x};\omega)$ and $\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega}) = \hat{\boldsymbol{f}}(\boldsymbol{x};\hat{\omega})$ in equations (1) and (2):

$$\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}}) := \int_D W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\,\nu(\mathrm{d}\boldsymbol{x}). \tag{8}$$

In equation (8), intuitively, the smaller $\tilde{W}_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})$ is, the more similar the probability measure $\hat{\mu}_{\boldsymbol{x}}$ is to the probability measure $\mu_{\boldsymbol{x}}$ for most $\boldsymbol{x}$ in the support of ν. Specifically, if $\nu(\cdot)$ is strictly positive on the whole domain D, i.e.

$$\nu(x) > 0, \qquad \forall x \in D,$$

then $\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ equals 0 only when

$$W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}}) = 0,\ \text{a.e.},\boldsymbol{x} \in D. \tag{9}$$

Thus, $\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ is minimized only when the distribution of $\boldsymbol{y}(\boldsymbol{x};\omega)$ in the uncertainty model equation (1) can be perfectly matched by the distribution of $\hat{\boldsymbol{y}}(\boldsymbol{x};\omega)$ in the approximate model equation (2) a.e. in D.

We introduce a novel **local squared W_2 loss function**:

$$\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}}) := \int_D W_2^2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}},\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x}). \tag{10}$$

Note that the local squared W_2 loss function equation (10) is not a single squared W_2 distance between two probability measures. Instead, it is an average of the Wasserstein distances $W_2^2(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}},\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}})$ across all $\boldsymbol{x}$. In equation (10) and throughout this manuscript, the superscript e and the subscript δ refer to using the empirical probability measure evaluated using the neighborhood of size δ, respectively.

In theorem 2.3, we shall show that our local squared W_2 loss function is a consistent approximation to the weighted average discrepancy equation (8). In equations (8) and (10), $\nu(\cdot)$ and $\nu^{\mathrm{e}}(\cdot)$ are the distribution and the empirical distribution of $\boldsymbol{x}$, respectively. $\boldsymbol{y},\hat{\boldsymbol{y}}$ correspond to the left-hand side (LHS) of ground truth model equation (1) and the LHS of the approximate model equation (2), respectively. W_2^2 is the squared W_2 distance (detailed definition given in definition 2.1). In equations (8) and (10), $\mu_{\boldsymbol{x}}$ is the distribution of $\boldsymbol{y}(\boldsymbol{x};\omega)$ when $\boldsymbol{x}$ is fixed, and $\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}$ is the empirical conditional distribution of $\boldsymbol{y}(\tilde{\boldsymbol{x}};\omega)$ conditioned on $|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta$. Similarly, $\hat{\mu}_{\boldsymbol{x}}$ is the distribution of $\hat{\boldsymbol{y}}(\boldsymbol{x},\hat{\omega})$ when $\boldsymbol{x}$ is fixed, and $\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}$ is the empirical conditional distribution of $\hat{\boldsymbol{y}}(\tilde{\boldsymbol{x}};\hat{\omega})$ conditioned on $|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta$, respectively. $|\cdot|_x$ denotes a norm for $\boldsymbol{x} \in \mathbb{R}^d$.

However, it is usually difficult to evaluate equation (8) when only a finite set of observations $S := \{(\boldsymbol{x}_i,\boldsymbol{y}_i)\}_{i=1}^N$ is available. If $\boldsymbol{x} \in D$ is a continuous random variable with a strictly positive probability density function $\nu(\cdot)$, then almost surely $\boldsymbol{x}_i \neq \boldsymbol{x}_j$ for any $(\boldsymbol{x}_i,\boldsymbol{y}_i),(\boldsymbol{x}_j,\boldsymbol{y}_j) \in S$ when $i \neq j$. Consequently, it is challenging to evaluate $\mu_{\boldsymbol{x}}$. To tackle this problem, we propose the local squared W_2 distance loss function equation (10). We shall show that the local squared W_2 distance serves as a good approximation to $\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ in equation (8). We can prove the following theorem that gives an error bound of using the local squared W_2 loss function $\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})$ in equation (10) to approximate $\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$.

Theorem 2.3. *For each $x \in D$, we denote the number of samples $(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{y}}) \in S$ such that $|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta$ to be $N(\boldsymbol{x},\delta)$. We denote the total number of samples of the empirical distribution to be N. Assuming that each input $\boldsymbol{x}$ is independently sampled from the probability distribution ν, then we have the following error bound*

$$\mathbb{E}\left[\left|\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}}) - \tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\right|\right] \leqslant \frac{4M}{\sqrt{N}} + 8CM\mathbb{E}\left[h\left(N(\boldsymbol{x},\delta),d\right)\right] + 8\sqrt{M}L\delta \tag{11}$$

where $\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})$ is the local W_2 distance defined in equation (10) and $\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ is defined in equation (8). M is the upper bound for $\|\boldsymbol{y}\|,\|\hat{\boldsymbol{y}}\|$ in equation (6), C is a constant, N is the total number of data points $(\boldsymbol{x},\boldsymbol{y})$, and L is the Lipschitz constant in equation (7). In equation (11),

$$h(N,d) := \begin{cases} 2N^{-\frac{1}{4}}\log(1+N)^{\frac{1}{2}}, d \leqslant 4, \\ 2N^{-\frac{1}{d}}, d > 4. \end{cases} \tag{12}$$

The proof to theorem 2.3 is provided in appendix A. Specifically, there is a trade-off between the second and third terms in the error bound equation (11): if we increase δ, then $N(\boldsymbol{x},\delta)$ tends to increase, which makes the second term smaller but the third term larger. As an example, if $d>4$ and there is a constant $\nu_0>0$ such that $N(\boldsymbol{x},\delta)\geqslant \nu_0\delta^n$, then from the inequality (11), we can conclude:

$$\mathbb{E}\left[\left|\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}}) - \tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\right|\right] \leqslant \frac{4M}{\sqrt{N}} + 8C\nu_0^{-\frac{1}{d}}M\delta^{-\frac{n}{d}} + 8\sqrt{M}L\delta. \tag{13}$$

For example, if $\boldsymbol{x}$ is uniformly distributed in D and D is a rectangular box or a ball, then $\mathbb{E}[h(N(\delta,\boldsymbol{x}))]\approx N\cdot\frac{V(B_n)}{V(D)} := \nu_0\delta^n$, where $\nu_0 := \frac{NV(B_n)}{V(D)}$ with $V(B_n)$ being the volume of a unit ball in $\mathbb{R}^n$ and $V(D)$ being the volume of the domain D. equation (13) explicitly shows that when we increase the size of neighborhood δ, the second term in the error bound equation (13) for $\mathbb{E}\left[\left|\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}}) - \tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\right|\right]$ decreases, while the third term increases. Furthermore, in this case, setting $\delta\sim\nu_0^{-\frac{1}{n+d}}$ ensures that the second term and third term in equation (13) are of the same order. Therefore, when the distribution of samples is close to a uniform distribution with density ν_0 in a unit ball, a practical choice of δ could be $\nu_0^{-\frac{1}{n+d}}$ to balance the error induced from adopting the neighborhood technique and the error when estimating the empirical distribution of $\boldsymbol{y}_{\boldsymbol{x}}$ using finite many samples.

Specifically, the case when $\delta=\infty$ corresponds to commonly used squared W_2 distance $W_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ (without considering the dependence on $\boldsymbol{x}$). However, as shown on the RHS of equation (11), setting $\delta=\infty$ leads to a blow-up of the third term on the error bound. Empirically, setting $\delta=\infty$ will not work well for reconstructing the dependence of $\boldsymbol{y}$ on different $\boldsymbol{x}$, which will be shown in section 3.2.

In addition, theorem 2.3 implies that $\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ can be well approximated by $\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})$ when the number of training data N is sufficiently large such that even for a small δ, $\mathbb{E}[N(\boldsymbol{x},\delta)^{-\frac{1}{d}}]$ can be maintained small, implying the convergence of our local squared W_2 loss function to the 'weighted average discrepancy' $W_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ defined in equation (8). In this scenario, minimizing our local squared W_2 loss function is also necessary such that the distribution of $\boldsymbol{y}(\boldsymbol{x};\omega)$ can be well represented by the distribution of $\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})$ for different $\boldsymbol{x}$. Therefore, our local squared W_2 method approximates equation (1) using equation (2) by minimizing the local squared W_2 loss function $\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})$ in equation (10). Empirically, we shall show in our numerical examples that our local squared W_2 method could lead to accurate reconstruction of lower-dimensional models with only hundreds or thousands of data points when an appropriate neighborhood size δ is chosen.

Finally, theorem 2.3 reflects the well-known curse of dimensionality: as the dimensionality d increases, the convergence rate of the empirical measure toward the target measure deteriorates to $\mathcal{O}\left(N(\boldsymbol{x},\delta)^{-1/d}\right)$. This rate is known to be optimal in general settings [33], and similar dimensionality-induced challenges are widely observed in UQ problems [10]. Yet, in many practical applications, the effective dimensionality may be significantly lower due to anisotropic structure or low-dimensional variability in the underlying random fields. In such cases, alternative formulations-such as sliced Wasserstein distances [34]-may exhibit improved empirical performance and improve computational efficiency by reducing the Wasserstein distance between two multidimensional distributions to many Wasserstein distances between unidimensional projections between them, which could then be easily evaluated. Additionally, recent theoretical results also show that using the W_2 metric to learn anisotropic random field models could partially alleviate the curse of dimensionality [35]. A rigorous characterization of these effects in our setting remains a nontrivial open question and is an interesting direction for future work.

2.2. Structure of the neural-network model with weight uncertainty

Here, we provide the structure of the neural network model with weight uncertainty that shall be used in the numerical examples in section 3. When we use this neural-network model with weight uncertainty as the approximate model $\hat{f}$ in equation (2), all weights with uncertainty $\{w_{i,j,k}\}$ constitutes the random variable $\hat{\omega}$ in equation (2). The mean and variance $a_{i,j,k},\sigma_{i,j,k}^2$ of the weight $w_{i,j,k}$ as well as the bias $b_{i,k}$ for all i,j,k are to be optimized through minimizing the local squared W_2 loss function equation (10) (or other loss functions listed in appendix B).

A pseudocode of minimizing our proposed local squared W_2 loss function to train the neural network model with weight uncertainty in figure 1 is given in algorithm 1.

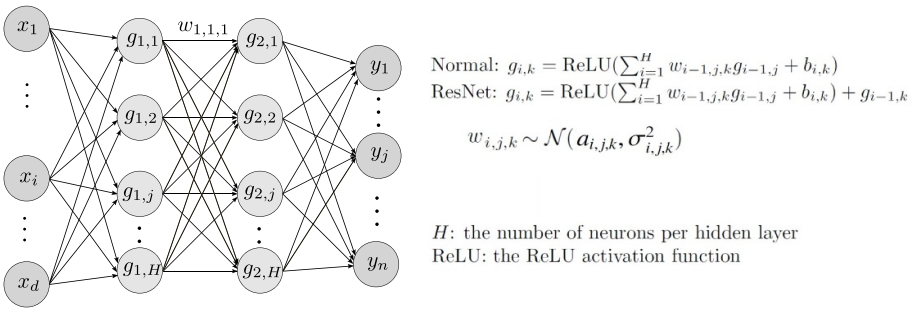

Figure 1. A sketch of the structure of the neural network model with weight uncertainty used in this paper. The weights $w_{i,j,k} \sim \mathcal{N}(a_{i,j,k}, \sigma^2_{i,j,k})$ are independently sampled, i.e. w_{i_1,j_1,k_1} is independent of w_{i_2,j_2,k_2} when $(i_1,j_1,k_1) \neq (i_2,j_2,k_2)$. When using this neural network model to make predictions, for each input $\boldsymbol{x} = (x_1,\ldots,x_d) \in D \subseteq \mathbb{R}^d$, we shall resample all weights $\{w_{i,j,k}\}$ again. Either the normal feed-forward structure for forward propagation or the ResNet technique in [36] for forward propagation is adopted. In addition to the ReLU activation function shown above, we also use the ELU activation function in section 3.4.

Algorithm 1. The pseudocode of training the neural network with weight uncertainty model by minimizing our local squared W_2 loss function (the local squared W_2 loss in the **while** loop can be replaced with other loss functions).

Given N observed data points $\{(\boldsymbol{x}_i, \boldsymbol{y}_i), i = 1,\ldots,N\}$, the stopping criteria $\epsilon > 0$, the size of the neighborhood δ, and the maximal epochs $i_{\max}$.
Initialize the neural network model in figure 1.
For each data point $\boldsymbol{x}_i$, find samples in its neighborhood $B_i := \{\boldsymbol{x}_j : |\boldsymbol{x}_j - \boldsymbol{x}_i|_x \leqslant \delta\}$.
Input $\{\boldsymbol{x}_i\}, i = 1,\ldots,N$ into the neural network model to obtain predictions $\{\hat{\boldsymbol{y}}_i\}, i = 1,\ldots,N$.
while $\tilde{W}^{2,\mathrm{e}}_{2,\delta}(\boldsymbol{y},\hat{\boldsymbol{y}}) > \epsilon$ && $i < i_{\max}$ **do**
  Perform gradient descent to minimize the loss function $\tilde{W}^{2,\mathrm{e}}_{2,\delta}(\boldsymbol{y},\hat{\boldsymbol{y}})$ and update the parameters (biases & means and variances of weights) in the neural network.
  Resample the weights in the neural network using the updated means and variances of weights.
  Input $\{\boldsymbol{x}_i\}, i = 1,\ldots,N$ into the updated neural network model to obtain predictions $\{\hat{\boldsymbol{y}}_i\}, i = 1,\ldots,N$. # for each $\boldsymbol{x}_i$, the weights in
  the neural network are sampled independently.
  $i = i + 1$.
end while
return The trained neural network with weight uncertainty model

In general, neither equation (10) nor (8) has closed form since the Wasserstein-2 distance only has closed forms under certain specific scenarios, e.g. when random variables sampled from multivariate Gaussian distribution. As a special case, assume that for each $\boldsymbol{x}$, $f(\boldsymbol{x},\omega) \sim \mathcal{N}(\boldsymbol{b}_{\boldsymbol{x}}; \Sigma_{\boldsymbol{x}})$ and $\hat{f}(\boldsymbol{x},\omega) \sim \mathcal{N}(\hat{\boldsymbol{b}}_{\boldsymbol{x}}; \hat{\Sigma}_{\boldsymbol{x}})$ where $\boldsymbol{b}_{\boldsymbol{x}}, \hat{\boldsymbol{b}}_{\boldsymbol{x}} \in \mathbb{R}^d$, $\Sigma_{\boldsymbol{x}}, \hat{\Sigma}_{\boldsymbol{x}} \in \mathbb{R}^{d\times d}$ and $\mathcal{N}(\cdot,\cdot)$ denotes the probability density function of a d-dimensional multivariate Gaussian distribution. In this case, from [37], $W_2^2(\mu_{\boldsymbol{x}}, \hat{\mu}_{\boldsymbol{x}}) = \|\boldsymbol{b}_{\boldsymbol{x}} - \hat{\boldsymbol{b}}_{\boldsymbol{x}}\|^2 + \mathrm{tr}(\Sigma_{\boldsymbol{x}} + \hat{\Sigma}_{\boldsymbol{x}} - 2(\Sigma_{\boldsymbol{x}}\hat{\Sigma}_{\boldsymbol{x}})^{\frac{1}{2}})$. When δ is small in equation (10) and the number of samples is large enough, we have:

$$\tilde{W}^{2,\mathrm{e}}_{2,\delta}(\boldsymbol{y},\hat{\boldsymbol{y}}) \approx \tilde{W}^2_2(\boldsymbol{y},\hat{\boldsymbol{y}}) = \int_D \|\boldsymbol{b}_{\boldsymbol{x}} - \hat{\boldsymbol{b}}_{\boldsymbol{x}}\|^2 + \mathrm{tr}\left(\Sigma_{\boldsymbol{x}} + \hat{\Sigma}_{\boldsymbol{x}} - 2\left(\Sigma_{\boldsymbol{x}}\hat{\Sigma}_{\boldsymbol{x}}\right)^{\frac{1}{2}}\right)\nu(\mathrm{d}\boldsymbol{x}). \tag{14}$$

Though our proposed local squared W_2 loss function equation (10) might not possess an explicit form, it could be numerically evaluated by utilizing the `POT` package in [38].

3. Applications of the local squared W_2 method to different UQ tasks

In this section, we will demonstrate the implementation of our local squared W_2 method for various UQ problems and evaluate its effectiveness across several UQ tasks. Detailed training settings and hyperparameters for each example are listed in table 2. Numerical examples in sections 3.1–3.3 are conducted using Python 3.11 on a desktop equipped with a 32-core Intel® i9-13 900KF CPU (when comparing runtime, we train each model on just one core). The numerical example in section 3.4 is conducted using NYU's HPC server.

Table 2. Training hyperparameters, hyperparameters in the neural network model, and training settings for each example. The neural network parameters include means and standard deviations $a_{i,j,k}, \sigma_{i,j,k}$ for weights $w_{i,j,k}$ as well as biases $b_{i,k}$ in figure 1.

	Section 3.1	Section 3.2	Section 3.3	Section 3.4
Gradient descent method	AdamW	AdamW	AdamW	AdamW
Forward propagation method	\	ResNet	ResNet	Normal
Learning rate	0.02	0.025	0.02	0.005
Weight decay	0.005	0.005	0.005	0.005
Number of epochs	1000	1000	1000	500
Number of training samples	1000	2000	686	100
Size of neighborhood δ in the loss equation (10)	0.1	0.025	0.05	0.1
Number of hidden layers in Θ	\	4	4	2
Activation function	\	ReLU	ReLU	ELU
Number of neurons in each layer in Θ_1	\	50	50	100
Initialization for model/neural-network parameters	1	$\mathcal{N}(0, 10^{-4})$	$\mathcal{N}(0, 10^{-4})$	$\mathcal{N}(0, 10^{-4})$
Repeat times	5	5	5	5

In this paper, the errors in the mean $\mathbb{E}[\hat{y}]$ and the standard deviation $\mathrm{SD}[\hat{y}]$ stand for the relative errors:

$$\begin{aligned} \text{error in } \mathbb{E}[\hat{y}] &:= \frac{\int_D \left|\mathbb{E}[y(\boldsymbol{x};\omega)] - \mathbb{E}[\hat{y}(\boldsymbol{x};\hat{\omega})]\right| \nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x})}{\int_D |\mathbb{E}[y(\boldsymbol{x};\omega)]|\, \nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x})}, \\ \text{error in } \mathrm{SD}[\hat{y}] &:= \frac{\int_D \left|\mathrm{SD}[y(\boldsymbol{x};\omega)] - \mathrm{SD}[\hat{y}(\boldsymbol{x};\hat{\omega})]\right| \nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x})}{\int_D |\mathrm{SD}[y(\boldsymbol{x};\omega)]|\, \nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x})}. \end{aligned} \tag{15}$$

3.1. Linear regression with coefficient uncertainty

We first apply our proposed local squared W_2 method to a linear regression problem with coefficient uncertainty. Consider the following linear model whose coefficients are sampled from the normal distribution in [39]:

$$y(\boldsymbol{x};\omega) = \sum_{i=1}^{3} \omega_i x_i + \omega_0,\ \omega_i \sim \mathcal{N}\left(b_i, \sigma_i^2\right). \tag{16}$$

We assume that $\boldsymbol{x}$ is independent of ω and ω_i is independent of ω_j when $i \neq j$. In equation (16), we set the ground truth:

$$(b_0, b_1, b_2, b_3) = (1, 1, 2, 3),\ \ (\sigma_0, \sigma_1, \sigma_2, \sigma_3) = (0.1, 0.2, 0.3, 0.4). \tag{17}$$

We aim to develop another linear model:

$$\hat{y}(\boldsymbol{x};\hat{\omega}) = \sum_{i=1}^{3} \hat{\omega}_i x_i + \hat{\omega}_0,\ \ \hat{\omega}_i \sim \mathcal{N}\left(\hat{b}_i, \hat{\sigma}_i^2\right),\ \hat{\boldsymbol{b}} := \left(\hat{b}_0, \hat{b}_1, \hat{b}_2, \hat{b}_3\right),\ \hat{\boldsymbol{\sigma}} := (\hat{\sigma}_0, \hat{\sigma}_1, \hat{\sigma}_2, \hat{\sigma}_3) \tag{18}$$

to approximate equation (16) so that the distribution of $y(\boldsymbol{x};\omega)$ can be matched by the distribution of $\hat{y}(\boldsymbol{x};\hat{\omega})$ when fixing $\boldsymbol{x}$. In equation (18), we assume that $\boldsymbol{x}$ is independent of $\hat{\omega}$ and $\hat{\omega}_i$ is independent of $\hat{\omega}_j$ when $i \neq j$. For the training data $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$, we let x_1, x_2, x_3 be independent of each other and sample $\boldsymbol{x} := (x_1, x_2, x_3)$ from the following distributions:

$$x_1 \sim \mathrm{Exp}(4),\ x_2 \sim \mathcal{N}(0, 0.25),\ x_3 \sim \mathrm{Be}(5, 5). \tag{19}$$

Here, $\mathrm{Exp}(4)$ denotes the exponential distribution with intensity parameter 4, and $\mathrm{Be}(5,5)$ represents the Beta distribution with both its shape parameters set to 5.

We minimize the local squared W_2 distance equation (10) in order to obtain $\hat{b}_i$ and $\hat{\sigma}_i$ in equation (18). When determining the neighborhood $|\tilde{\boldsymbol{x}} - \boldsymbol{x}| \leqslant \delta$ of $\boldsymbol{x}$ for evaluating the empirical distributions $\mu^{\mathrm{e}}_{\boldsymbol{x},\delta}$ and $\hat{\mu}^{\mathrm{e}}_{\boldsymbol{x},\delta}$ in equation (10), two different norms of $\boldsymbol{x}$ are used:

$$|\boldsymbol{x}|^2_{\mathrm{homo}} := \sum_{i=1}^{3} x_i^2,\ |\boldsymbol{x}|^2_{\mathrm{hete}} := \sum_{i=1}^{3} c_i^2 x_i^2, \tag{20}$$

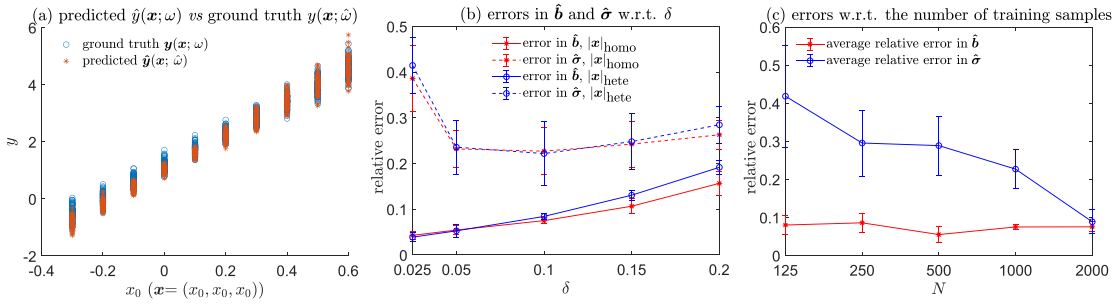

Figure 2. (a) The predicted $\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})$ versus the ground truth $\boldsymbol{y}(\boldsymbol{x},\omega)$. To illustrate, we take $\boldsymbol{x}$ on the line $\boldsymbol{x}=(x_0,x_0,x_0)$ and choose different values of $x_0=-0.3+0.1i, i=0,\ldots,9$. At each $\boldsymbol{x}$, we independently sample 100 $\omega=(\omega_0,\omega_1,\omega_2,\omega_3)$ in equation (16) as well as $\hat{\omega}=(\hat{\omega}_0,\hat{\omega}_1,\hat{\omega}_2,\hat{\omega}_3)$ in equation (18) and plot 100 ground truth $y(\boldsymbol{x};\omega)$ versus 100 predicted $\hat{y}(\boldsymbol{x};\hat{\omega})$. (b) The average relative errors in $\hat{b}_i$ and $\hat{\sigma}_i$ w.r.t. the size of neighborhood δ when using the two different norms of the input $\boldsymbol{x}$: $|\boldsymbol{x}|_{\text{homo}}$ and $|\boldsymbol{x}|_{\text{hete}}$. (c) The average relative errors in $\hat{b}_i$ and $\hat{\sigma}_i$ w.r.t. the number of training samples N. In (c), the norm for $\boldsymbol{x}$ is $|\boldsymbol{x}|_{\text{hete}}$ (defined in equation (20)) and the size of neighborhood $\delta=0.1$.

where c_i in $|\boldsymbol{x}|^2_{\text{hete}}$ are obtained from carrying out a linear regression of y w.r.t. $\boldsymbol{x}$ by minimizing:

$$\sum_{i=1}^{N}\left(y_i-\sum_{i=1}^{3}c_i x_i-c_0\right)^2. \tag{21}$$

Using $|\cdot|_{\text{hete}}$ accounts for the heterogeneity in the dependencies of y on x_1,x_2,x_3 in equation (16). We use the average relative errors to measure errors in the reconstructed $\hat{b}_i$ and $\hat{\sigma}_i, i=1,2,3,4$ in equation (18):

$$\text{Error in } \hat{\boldsymbol{b}} := \frac{\sum_{i=0}^{3}|b_i-\hat{b}_i|}{\sum_{i=0}^{3}|b_i|}, \quad \text{Error in } \hat{\boldsymbol{\sigma}} := \frac{\sum_{i=0}^{3}\left||\sigma_i|-|\hat{\sigma}_i|\right|}{\sum_{i=0}^{3}|\sigma_i|}. \tag{22}$$

In figure 2(a), the distribution of the predicted $\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})$ matches well with the distribution of the ground truth $\boldsymbol{y}(\boldsymbol{x};\omega)$ on the line $\boldsymbol{x}=(x_0,x_0,x_0)$. In figure 2(b), the errors in the reconstructed $\hat{\boldsymbol{b}}$ and $\hat{\boldsymbol{\sigma}}$ are not sensitive to whether using $|\boldsymbol{x}|_{\text{homo}}$ or $|\boldsymbol{x}|_{\text{hete}}$. However, when the size of neighborhood δ in equation (10) is too small ($\delta=0.025$), the error in the reconstructed standard deviation $\hat{\boldsymbol{\sigma}}$ is large. When δ is too small, the local squared W_2 loss equation (10) might not be a good approximation of $\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ in equation (10), leading to the poor reconstruction of equation (16). On the other hand, the error in the reconstructed mean $\hat{\boldsymbol{b}}$ gets larger when δ increases. The error in the reconstructed bias $\hat{\boldsymbol{b}}$ could be the systematic error, which is introduced by using the neighboring technique, i.e. when quantifying the uncertainty at $\boldsymbol{x}$, we also include samples with $\tilde{\boldsymbol{x}}$ if $|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta$. Errors in the reconstructed mean and standard deviation are both kept small when $\delta\in[0.05,0.1]$. The error in the reconstructed standard deviation $\hat{\boldsymbol{\sigma}}$ decreases as the number of training samples N increases, while the error in the reconstructed $\hat{\boldsymbol{b}}$ is not very sensitive to N (shown in figure 2(c)).

For $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ in equations (16) and (18), since both $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ are linearly dependent on x_i,ω_i, and $\hat{\omega}_i$ and the coefficients ω_i, and $\hat{\omega}_i$ are independently sampled from normal distributions, we have:

$$\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})=\int_D\left(\sum_{i=1}^{3}\left(\hat{b}_i-b_i\right)x_i+\left(\hat{b}_0-b_0\right)\right)^2+\left(\left(\sum_{i=1}^{3}\hat{\sigma}_i^2x_i^2+\hat{\sigma}_0^2\right)^{\frac{1}{2}}-\left(\sum_{i=1}^{3}\sigma_i^2x_i^2+\sigma_0^2\right)^{\frac{1}{2}}\right)^2\nu(\mathrm{d}\boldsymbol{x}). \tag{23}$$

$W_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})$ is defined in equation (8). Therefore, our local squared W_2 loss function:

$$\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\approx\int_D\left(\sum_{i=1}^{3}\left(\hat{b}_i-b_i\right)x_i+\left(\hat{b}_0-b_0\right)\right)^2+\left(\left(\sum_{i=1}^{3}\hat{\sigma}_i^2x_i^2+\hat{\sigma}_0^2\right)^{\frac{1}{2}}-\left(\sum_{i=1}^{3}\sigma_i^2x_i^2+\sigma_0^2\right)^{\frac{1}{2}}\right)^2\nu(\mathrm{d}\boldsymbol{x}), \tag{24}$$

and the approximation gets better when the number of samples becomes sufficiently large such that there are enough training samples around each $\boldsymbol{x}\in D$ even if we choose a very small $\delta\to0^+$. Then, both b_i and σ_i might be accurately reconstructed. With limited training data (~1000 in this case), we need to

use a moderate $\delta \in [0.05, 0.1]$ to ensure that for most $\boldsymbol{x}$, there are enough samples around its neighborhood. This leads to a tradeoff in the accuracy of the reconstructed $\hat{b}_i$ and the reconstructed $\hat{\sigma}_i$, as shown in figure 2(b).

3.2. Training a neural network model with weight uncertainty

Next, we consider reconstructing the following nonlinear uncertainty model [40, 41]:

$$y(x;\omega) = \omega_1 (1 - \exp(-\omega_2 x)) + 5, \tag{25}$$

where $\omega := (\omega_1, \omega_2)^{\mathrm{T}}$ are the latent random variables in the model. We assume that x and ω are independent. We independently generate 1000 samples for training with $x \sim \mathcal{U}(-0.5, 0.5)$ and $(\omega_1, \omega_2)^{\mathrm{T}} \sim \mathcal{N}\left((19.1426, 0.5311)^{\mathrm{T}}, \Sigma\right)$, where Σ is the covariance matrix:

$$\Sigma = \begin{bmatrix} 6.22864 & -0.4322 \\ -0.4322 & 0.04124 \end{bmatrix}. \tag{26}$$

A parameterized neural network model with weight uncertainty in figure 1 is used as $\hat{\boldsymbol{f}}$ in equation (2), which approximates equation (25). We aim to optimize the mean and variance of weights $\{w_{i,j,k}\}$ as well as the bias $\{b_{i,k}\}$ in the neural network by minimizing equation (10) such that the distribution of $\hat{y}$ aligns with the distribution of y given the same x. For testing, we generate a testing set $T = \cup_{i=0}^{10} T_i$ with each T_i containing 100 samples $(x_{r,i}, y(x_{r,i};\omega)), x_{r,i} = 0.1i - 0.5, r = 1, \ldots, 100$.

We compare our local squared W_2 loss function with other commonly used loss functions in UQ (definitions given in appendix B) as well as a BNN method in [42, 43] which minimizes the Kullback–Leibler divergence and a WGAN method in [44, 45]. For implementing the WGAN method, the generator is the same parameterized neural network model with weight uncertainty in figure 1 as used when utilizing other loss functions, while the discriminator is a feedforward neural network with one hidden layer equipped with 32 neurons and the ReLU activation function. Within each epoch for training the generator, 256 randomly selected samples will be provided to train the discriminator, repeated 10 times. After training, the generator is used to make new predictions on the testing dataset. The neural network model in figure 1 trained by minimizing the local squared W_2 loss function yields $\hat{y}(x;\hat{\omega})$ whose distribution is close to the distribution of the ground truth $y(x;\omega)$ on the testing set (figure 3(a)). The performance of minimizing the local maximum mean discrepancy (MMD) loss is comparable to minimizing our local squared W_2 loss (figure 3(b)). However, unlike the analysis of our local squared W_2 method in section 2.1, there is no theoretical guarantee justifying why the local MMD loss could be successful. Further comparisons with such local MMD loss will be presented in section 3.3. The distributions of the predicted $\hat{y}(x;\hat{\omega})$ by minimizing the local Mean squared error (MSE) and the local Mean^2+var deviate much from the distribution of the ground truth $y(x;\omega)$ at different x (figures 3(c)–(d)). Adopting any 'nonlocal' loss functions yields poor performance (figures 3(e)–(h)). The BNN method generates $\hat{y}(x;\hat{\omega})$ whose distribution fails to adequately match with the distribution of the ground truth $y(x;\omega)$, and one possible reason could be that randomly initialized means and variances of weights as well as the biases in the neural network do not provide a good prior distribution for the BNN. Finally, we find that training the discriminator and generator in the WGAN approach requires fine-tuning of hyperparameters, as reported in other literature [46]. We need to choose a learning rate equal to 0.0001 with 10 000 epochs for training the generator so that both the discriminator and the generator can be trained successfully. Nonetheless, the WGAN performs worse than our proposed local squared W_2 approach. Overall, our local squared W_2 method can most efficiently train the neural network model with weight uncertainty in figure 1 to reconstruct the nonlinear model equation (25) among all loss functions and methods, with the smallest errors in $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$ on the testing set (shown in figure 3(j)). Additionally, when adopting the neural network model (figure 1), our method does not require prior knowledge of the form of the nonlinear model equation (25), nor does it demand prior distributions of the two latent model parameters ω_1, ω_2.

We also compared the runtime and memory usage when using different loss functions or methods. From table 3, training with any 'local' loss function can be more computationally expensive than using their 'nonlocal' counterparts. The computational cost of evaluating the local squared W_2 loss function is similar to that of using the local MMD loss function, which is more expensive than that of using local MSE, local Mean^2+var loss, or the WGAN method. Yet, using the local MSE, local Mean^2+var loss function, or the WGAN method fails to accurately reconstruct the uncertainty model equation (25). On the other hand, using our local squared W_2 loss function will not lead to an increased memory usage compared to using other loss functions or methods. Theoretically, in the worst scenario, the number of floating-point operations needed to evaluate our proposed local squared W_2 loss function is

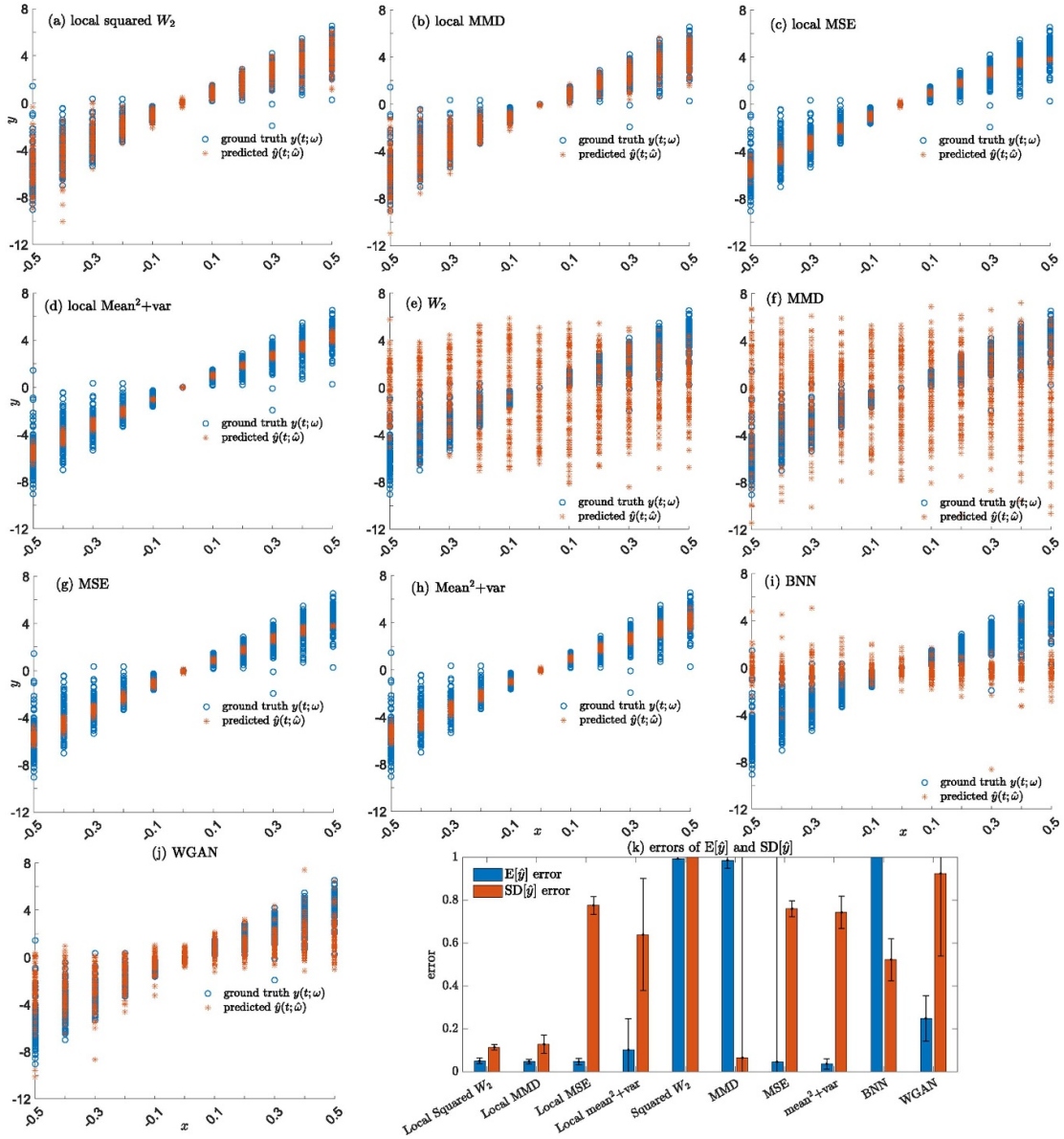

Figure 3. (a)–(j) The ground-truth $y(x;\omega)$ plotted against the predicted $\hat{y}(x;\hat{\omega})$ on the testing set. The predicted $\hat{y}$ is obtained by minimizing different loss functions (defined in appendix B) and obtained by using the BNN method and by using the WGAN method. (k) The average errors in the mean and the standard deviations of $\hat{y}$ on the testing set obtained by minimizing different loss functions and by using the BNN method as well as by using the WGAN method. The neural network model with weight uncertainty (figure 1) trained by minimizing our local squared W_2 loss yields the smallest errors among all methods. Minimizing the local MMD is comparable to minimizing the local squared W_2 loss, likely because the MMD could also somehow measure the discrepancy between two probability distributions.

Table 3. Runtime and memory usage of utilizing different loss functions and methods. Mean and standard deviations of runtime and memory usage are obtained over five repeated experiments of each method.

Method	Local squared W_2	Local MMD	Local MSE	Local mean^2+var	W_2
Time (s)	3346 ± 213	3915 ± 179	558 ± 202	1115 ± 835	570 ± 16
Memory (MiB)	1151.3 ± 168.4	2867.9 ± 291.8	1029.6 ± 25.9	857.3 ± 61.0	854.3 ± 105.9
Method	MMD	MSE	Mean^2+var	BNN	WGAN
Time (s)	842 ± 37	100 ± 1	157 ± 14	$73\,310 \pm 4057$	2166 ± 105
Memory (MiB)	1328.9 ± 278.2	853.6 ± 1.2	723.6 ± 137.4	3196.4 ± 342.0	313.6 ± 6.3

$\mathcal{O}(N\mathbb{E}[N^3(\boldsymbol{x},\delta)\log N(\boldsymbol{x},\delta)])$ with standard network simplex methods, where N is the total number of samples and $N(\boldsymbol{x},\delta)$ is the number of samples in a neighborhood around $\boldsymbol{x}$. Therefore, a proper choice of δ is important to guarantee $N(\boldsymbol{x};\delta)$ is not too small (not sufficient data to quantify uncertainty,

e.g. when $\delta = 0.025$) or too large (computationally expensive and introducing large systematic errors).

Two additional experiments are performed. First, we alter the standard deviations of the two parameters ω_1, ω_2 in equation (25) and the standard deviation of the input x. We find that larger standard deviations in the latent model parameters and a larger standard deviation in the input x (sparser data) both lead to a poorer reconstruction of the nonlinear model equation (25), as shown in appendix C.

Second, we adjust the structure of the neural network model depicted in figure 1. We discover that using a neural network with 2 hidden layers and 50 neurons per hidden layer equipped with the ResNet technique in [36] leads to the smallest errors in the reconstructed $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$. These results are presented in appendix D.

3.3. Application: reconstructing the distributions of concrete compressive strength associated with selected variables

As an application of our method, we reconstruct the distribution of the concrete compressive strength associated with selected continuous variables in the concrete compressive strength dataset in [47]. This dataset documents concrete compressive strength along with various influential factors affecting it. We reconstruct the distribution of the concrete compressive strength (measured in MPa) based on six recorded continuous variables (measured in $\mathrm{kg\,m^{-3}}$): cement, fly ash, water, superplasticizer, coarse aggregate, and fine aggregate. Previous models, such as those presented in [48, 49], depict concrete compressive strength as a continuous function of these variables. We exclude two discrete, integer-valued variables: blast furnace slag and age. Additionally, other factors that might affect the concrete compressive strength are not recorded in this dataset. Thus, the concrete compressive strength might not be a deterministic function of the six selected variables. We can regard the six selected variables as the input $\boldsymbol{x}$, the neglected variables as the latent variables ω, and the concrete compressive strength as y in equation (1). Then, we can use the approximate model equation (2) to approximate the distribution of the concrete compressive strength given $\boldsymbol{x}$. In our analysis, we normalize each selected variable with a mean of 0 and a variance of 1.

We compare the neural network model with weight uncertainty in figure 1, trained by minimizing the local squared W_2 loss function equation (10), against a neural network without weight uncertainty (i.e. setting $\sigma_{i,j,k} \equiv 0$ for the weights $w_{i,j,k}$ in figure 1), trained by minimizing the MSE loss (defined in appendix B). When using a neural network without weight uncertainty, the approximate model is deterministic:

$$\hat{y} = \hat{f}(\boldsymbol{x})\,. \tag{27}$$

Additionally, to make further comparison with MMD-based methods, we also compare training the neural network with weight uncertainty using our local squared W_2 loss function against using the local MMD loss function (detailed in appendix B).

The training set S consists of the first two-thirds of the samples in the dataset. The remaining one-third of the samples constitute the testing set, denoted by T. When calculating the errors in the predicted mean and standard deviation defined in equation (22), we use $\mathbb{E}\left[y(\tilde{\boldsymbol{x}};\omega)\middle||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0\right]$ to approximate $\mathbb{E}[y(\boldsymbol{x},\omega)]$, and $\mathbb{E}\left[\hat{y}(\tilde{\boldsymbol{x}};\hat{\omega})\middle||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0\right]$ to approximate $\mathbb{E}[\hat{y}(\boldsymbol{x},\omega)]$, respectively. We also use $\mathrm{SD}\left[y(\tilde{\boldsymbol{x}};\omega)\middle||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0\right]$ to approximate $\mathrm{SD}[y(\boldsymbol{x},\omega)]$ and $\mathrm{SD}\left[\hat{y}(\tilde{\boldsymbol{x}};\hat{\omega})\middle||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0\right]$ to approximate $\mathrm{SD}[\hat{y}(\boldsymbol{x},\hat{\omega})]$. Only $(\boldsymbol{x},y) \in T$ for which there are at least 5 samples $(\tilde{\boldsymbol{x}},y(\tilde{\boldsymbol{x}},\omega)) \in T$ satisfying $|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0$ are used for calculating the errors in the predicted mean and standard deviation. We take $\delta_0 = 0.2$ and $|\boldsymbol{x}|_x^2 := \sum_{i=1}^6 c_i^2 x_i^2$, where c_i is obtained by minimizing:

$$\sum_{j=1}^{|S|\cup|T|}\left(y_j\left(\boldsymbol{x}_j\right) - \sum_{i=1}^{6} c_i x_{j,i} - c_0\right)^2. \tag{28}$$

Our local squared W_2 method yields distributions of $\hat{y}(\tilde{\boldsymbol{x}},\hat{\omega}), |\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0$ that align well with the distributions of the ground truth $y(\tilde{\boldsymbol{x}},\omega), |\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0$ on the testing set for different $\boldsymbol{x}$. As illustrations, in figures 4(a) and (b), we plot the distributions of the predicted $\hat{y}(\tilde{\boldsymbol{x}},\hat{\omega}), |\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0$ against the distributions of the ground truth $y(\tilde{\boldsymbol{x}},\omega), |\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta_0$ for 10 randomly selected samples $(\boldsymbol{x}_i,y_i) \in T$ such that the cardinality of the set $\left|\left\{(\boldsymbol{x}_j,y_j) \in T : |\boldsymbol{x}_j-\boldsymbol{x}|_x \leqslant \delta_0\right\}\right| \geqslant 5$. In figure 4(c), we plot the errors in the predicted mean $\mathbb{E}[\hat{y}(\boldsymbol{x};\hat{\omega})]$ and standard deviation $\mathrm{SD}[\hat{y}(\boldsymbol{x};\hat{\omega})]$ obtained from the three methods: (i) using the neural network model with weight uncertainty trained by minimizing the local squared W_2 loss equation (10), (ii) using the neural network model with weight uncertainty trained by minimizing the

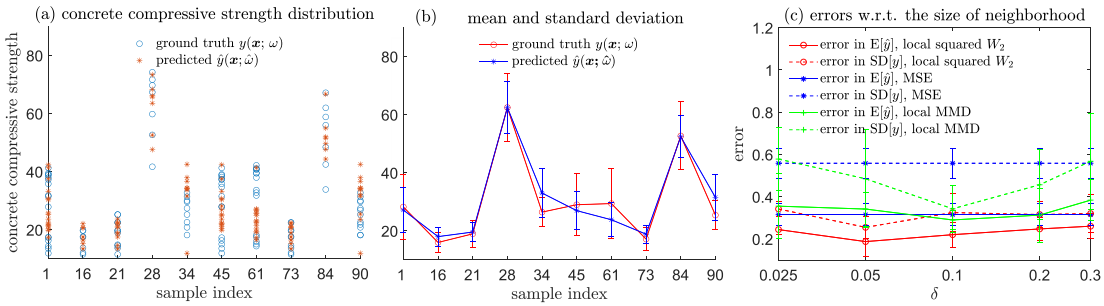

Figure 4. (a) The predicted $\hat{\boldsymbol{y}}(\tilde{\boldsymbol{x}};\hat{\omega})$ using the neural network with weight uncertainty in figure 1 versus the ground truth $\boldsymbol{y}(\tilde{\boldsymbol{x}};\omega)$ for 10 randomly selected samples $(\boldsymbol{x},y)$ in the testing set T satisfying $\left|\left\{\tilde{\boldsymbol{x}}\in T:|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta_0\right\}\right|\geqslant 5$. (b) The predicted means $\mathbb{E}\left[\hat{y}(\tilde{\boldsymbol{x}};\hat{\omega})\middle||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta_0\right]$ and the predicted standard deviations $\mathrm{SD}[\hat{y}(\tilde{\boldsymbol{x}};\hat{\omega})\big||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta_0]$ using the neural network with weight uncertainty model in figure 1 versus the ground truth mean $\mathbb{E}[y(\tilde{\boldsymbol{x}};\omega)\big||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta_0]$ and standard deviation $\mathrm{SD}[y(\tilde{\boldsymbol{x}};\omega)\big||\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta_0]$ for 10 randomly selected samples $(\boldsymbol{x},y)$ in the testing set T satisfying $\left|\left\{\tilde{\boldsymbol{x}}\in T:|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta_0\right\}\right|\geqslant 5$. In both (a) and (b), we use the neural network with weight uncertainty model trained by minimizing the local squared W_2 loss equation (10) with $\delta=0.05$ (δ is the size of the neighborhood in the loss function and is different from $\delta_0=0.2$). (c) The average relative errors in the mean and standard deviation of predictions $\hat{y}$ generated by the neural network model with weight uncertainty, trained by minimizing the local W_2 loss equation (10) or by minimizing the local MMD loss function, versus the average relative errors in the mean and standard deviation of predictions $\hat{y}$ generated by the neural network model without weight uncertainty, trained by minimizing the MSE loss. Note that in (c), the size of the neighborhood δ only applies to using the neural network model with weight uncertainty trained by minimizing the local W_2 loss equation (10) or the local MMD loss. Thus, when using the neural network model without weight uncertainty trained by minimizing the MSE loss, the results do not change with δ. Reproduced from [47]. CC BY 4.0.

local MMD loss, and (iii) using the neural network model without weight uncertainty trained by minimizing the MSE loss. The error in the predicted standard deviation is much smaller for (i) than for (ii) or (iii). Therefore, our proposed local squared W_2 method could more accurately reconstruct the distribution of the concrete compressive strength associated with different $\boldsymbol{x}$, compared to training a neural network with weight uncertainty by minimizing the local MMD loss or using a neural network model without weight uncertainty trained by minimizing the MSE. Training a deterministic neural network with the MSE loss function cannot capture the uncertainty in the concrete compressive strength, given the six selected variables. On the other hand, in addition to requiring an appropriate neighborhood size δ, the local MMD loss function is kernel-based and might require fine-tuning of the kernel functions for good performance, making it less flexible than using our local squared W_2 loss function.

Similar to the results shown in figure 2(b) on reconstructing the linear model equation (16), it is most appropriate to choose a moderate δ in the loss function equation (10) when using our local squared W_2 method to reconstruct the distribution of concrete compressive strength on the six selected variables. Errors in the predicted mean and standard deviation can both be well controlled when $\delta\in[0.05,0.1]$, as shown in figure 4(c). When δ is too small or too large, the accuracy of the predicted mean and standard deviation decreases.

3.4. Reconstructing an ODE with parameter uncertainty

Finally, we consider an ODE with uncertain latent parameters:

$$\frac{\mathrm{d}\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)}{\mathrm{d}t}=\boldsymbol{g}\left(\boldsymbol{y}\left(\boldsymbol{y}_0,t;\omega\right),t,\omega\right),\ \omega\in\Omega,t\in[0,T],\ \boldsymbol{y}(\boldsymbol{y}_0,0;\omega)=\boldsymbol{y}_0\in\mathbb{R}^n, \tag{29}$$

where ω are latent parameters with uncertainty. We aim at using another ODE to approximate equation (29):

$$\frac{\mathrm{d}\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})}{\mathrm{d}t}=\hat{\boldsymbol{g}}\left(\hat{\boldsymbol{y}}\left(\boldsymbol{y}_0,t;\hat{\omega}\right),t,\hat{\omega}\right),\ \hat{\omega}\in\hat{\Omega},t\in[0,T],\ \hat{\boldsymbol{y}}(\boldsymbol{y}_0,0;\hat{\omega})=\boldsymbol{y}_0\in\mathbb{R}^n, \tag{30}$$

where $\hat{\omega}$ are uncertain parameters in $\hat{\boldsymbol{g}}$. In the following, we regard the initial condition $\boldsymbol{y}_0$ as the input and $\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)$ as the output (the norms of the input $\boldsymbol{y}_0$ and output $\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)$ are the same l^2 norm $\|\cdot\|$ for vectors). For each $t\in[0,T]$, suppose there exists a Lipschitz constant L, uniform in t, such that for any $\omega\in\Omega$ and $\hat{\omega}\in\hat{\Omega}$:

$$\left\|\boldsymbol{y}\left(\boldsymbol{y}_0,t;\omega\right)-\boldsymbol{y}\left(\tilde{\boldsymbol{y}}_0,t;\omega\right)\right\|\leqslant L\|\boldsymbol{y}_0-\tilde{\boldsymbol{y}}_0\|,\ \left\|\hat{\boldsymbol{y}}\left(\boldsymbol{y}_0,t;\hat{\omega}\right)-\hat{\boldsymbol{y}}\left(\tilde{\boldsymbol{y}}_0,t;\hat{\omega}\right)\right\|\leqslant L\|\boldsymbol{y}_0-\tilde{\boldsymbol{y}}_0\|,\ \forall\boldsymbol{y}_0,\tilde{\boldsymbol{y}}_0\in\mathbb{R}^n, \tag{31}$$

then theorem 2.3 implies that minimizing the local squared W_2 distance $\tilde{W}^{2,\mathrm{e}}_{2,\delta}(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega}))$ in equation (10) could be effective in comparing the distributions of $\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)$ and $\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})$.

Additionally, if $\boldsymbol{g}$ and $\hat{\boldsymbol{g}}$ in equations (29) and (30) are uniformly Lipschitz continuous in $\boldsymbol{y}$ and t, then a large $\tilde{W}^{2,\mathrm{e}}_{2,\delta}(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),\boldsymbol{y}(\boldsymbol{y}_0,t;\hat{\omega}))$ implies that there exists a pair $(\boldsymbol{y},s)\in\mathbb{R}^d\times[0,t]$ such that $\hat{\eta}_{\boldsymbol{y},s}$ fails to align well with $\eta_{\boldsymbol{y},s}$ ($\eta_{\boldsymbol{y},s}$ and $\hat{\eta}_{\boldsymbol{y},s}$ denote the distributions of $\boldsymbol{g}(\boldsymbol{y},s,\omega)$ and $\hat{\boldsymbol{g}}(\boldsymbol{y},s,\hat{\omega})$, respectively). We present the following result.

Proposition 3.1. *Suppose the following Lipschitz conditions on $\boldsymbol{g}$ and $\hat{\boldsymbol{g}}$ hold:*

$$\begin{aligned}&\left\|\boldsymbol{g}(\boldsymbol{y}_1,t_1,\omega)-\boldsymbol{g}(\boldsymbol{y}_2,t_2,\omega)\right\|\leqslant L_g\left(\|\boldsymbol{y}_1-\boldsymbol{y}_2\|+|t_2-t_1|\right),\\&\left\|\hat{\boldsymbol{g}}(\boldsymbol{y}_1,t_1,\hat{\omega})-\hat{\boldsymbol{g}}(\boldsymbol{y}_2,t_2,\hat{\omega})\right\|\leqslant L_g\left(\|\boldsymbol{y}_1-\boldsymbol{y}_2\|+|t_2-t_1|\right),\ 0<L_g<\infty,\forall\boldsymbol{y}_1,\boldsymbol{y}_2\in\mathbb{R}^n,\forall t_1,t_2\geqslant 0,\forall\omega\in\Omega,\forall\hat{\omega}\in\hat{\Omega}.\end{aligned}\tag{32}$$

Then, there exist two constants C_0 depending on t and C_1 depending on t and $\mathbb{E}[\|\boldsymbol{y}\|^2]$ such that:

$$W_2^2\left(\mu_{\boldsymbol{y}_0,t},\hat{\mu}_{\boldsymbol{y}_0,t}\right)\leqslant C_0\sup_{\boldsymbol{y},0\leqslant s\leqslant t}W_2^2\left(\eta_{\boldsymbol{y},s},\hat{\eta}_{\boldsymbol{y},s}\right)+C_1,\ \forall\boldsymbol{y}_0\in\mathbb{R}^n.\tag{33}$$

In equation (33), $\mu_{\boldsymbol{y}_0,t}$ and $\hat{\mu}_{\boldsymbol{y}_0,t}$ are the distributions of $\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)$ and $\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})$; $\eta_{\boldsymbol{y},s},\hat{\eta}_{\boldsymbol{y},s}$ are the distributions of $\boldsymbol{g}(\boldsymbol{y},s,\omega)$ and $\hat{\boldsymbol{g}}(\boldsymbol{y},s,\hat{\omega})$ in equations (29) and (30).

Proof to proposition 3.1 is provided in appendix E. Proposition 3.1 implies that minimizing

$$\tilde{W}^{2,\mathrm{e}}_{2,\delta}\left(\boldsymbol{y}\left(\boldsymbol{y}_0,t;\omega\right),\hat{\boldsymbol{y}}\left(\boldsymbol{y}_0,t;\hat{\omega}\right)\right)\approx\int_{\mathbb{R}^n}W_2^2\left(\mu_{\boldsymbol{y}_0,t},\hat{\mu}_{\boldsymbol{y}_0,t}\right)\mathrm{d}\nu\left(\boldsymbol{y}_0\right)\leqslant\left(C_0\sup_{\boldsymbol{y},0\leqslant s\leqslant t}W_2^2\left(\eta_{\boldsymbol{y},s},\hat{\eta}_{\boldsymbol{y},s}\right)+C_1\right),\tag{34}$$

where $\nu(\boldsymbol{y}_0)$ is the probability measure of the initial condition $\boldsymbol{y}_0$, is a necessary condition such that $W_2^2(\eta_{\boldsymbol{y},s},\hat{\eta}_{\boldsymbol{y},s})$ is small for any $\boldsymbol{y}\in\mathbb{R}^n$ and $0\leqslant s\leqslant t$. In other words, if $\tilde{W}^{2,\mathrm{e}}_{2,\delta}\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\right)$ is large, then there exists a pair $(\boldsymbol{y},s),0\leqslant s\leqslant t$ such that $W_2^2(\eta_{\boldsymbol{y},s},\hat{\eta}_{\boldsymbol{y},s})$ is large and thus the distribution of $\boldsymbol{g}(\boldsymbol{y},s,\omega)$ cannot be well approximated by the distribution of $\hat{\boldsymbol{g}}(\boldsymbol{y},s,\hat{\omega})$. Empirically, minimizing the LFS of equation (34) is used as a proxy for recovering the model equation (29) by equation (30). Specifically, in the LHS of equation (36), we only use the initial condition $\boldsymbol{y}_0$ to determine the neighborhood, and the time variable t is considered fixed. Specifically, the first term on the RHS follows from the Gronwall's inequality, whereas the constant C_1 arises from the mismatch between the functional forms of $\boldsymbol{g}$ and $\hat{\boldsymbol{g}}$. The estimate in proposition 3.1 can be sharpened in certain special cases. In particular, when $\boldsymbol{g}=\hat{\boldsymbol{g}}$, the constant C_1 vanishes; see [30, theorem 2.2].

We reconstruct the following 4D ODE with a latent random variable (example 4.3 in [50]):

$$\begin{aligned}\frac{\mathrm{d}y_1}{\mathrm{d}t}&=(0.05+\omega)y_1+0.05y_3-\left(1-\omega^2\right)y_2,\\\frac{\mathrm{d}y_2}{\mathrm{d}t}&=\left(1-\omega^2\right)y_1+(0.05+\omega)y_2+0.05y_4,\\\frac{\mathrm{d}y_3}{\mathrm{d}t}&=(-0.05+\omega)y_3-\left(1-\omega^2\right)y_4,\\\frac{\mathrm{d}y_4}{\mathrm{d}t}&=(-0.05+\omega)y_4+\left(1-\omega^2\right)y_3,\ \ t\in[0,2].\end{aligned}\tag{35}$$

Let $\boldsymbol{g}(\boldsymbol{y},\omega):=\left(g_1(\boldsymbol{y},\omega),g_2(\boldsymbol{y},\omega),g_3(\boldsymbol{y},\omega),g_4(\boldsymbol{y},\omega)\right)^{\mathrm{T}}$ represent the RHS of equation (35). We set the initial condition $\boldsymbol{y}_0\sim\mathcal{N}((1,1,1,1)^{\mathrm{T}},a^2I_4)$, where $I_4\in\mathbb{R}^{4\times4}$ denotes the identity matrix. In equation (35), we let $\omega\sim\mathcal{U}(-\sigma,\sigma)$. We independently sample the initial condition $\boldsymbol{y}_0$ and ω, generating 100 trajectories for both the training and testing sets. The neural network model with weight uncertainty in figure 1 is adopted as the RHS $\hat{\boldsymbol{g}}$ in the approximate ODE model equation (30), which aims at approximating equation (35) (we also set $\hat{\boldsymbol{g}}$ to be time-homogeneous, i.e. $\hat{\boldsymbol{g}}=\hat{\boldsymbol{g}}(\boldsymbol{y},\hat{\omega})$). We shall use the `odeint` function with the `rk4` method in the `torchdiffeq` package [51] for numerically integrating the ground-truth ODE (35) and the approximate ODE (30). The means and variances of the weights, as well as the biases in the neural network, are optimized by minimizing the time-averaged local squared W_2 distance:

$$\frac{1}{m+1}\sum_{t=0}^{m}\tilde{W}^{2,\mathrm{e}}_{2,\delta}\left(\boldsymbol{y}\left(\boldsymbol{y}_0,t_i;\omega\right),\hat{\boldsymbol{y}}\left(\boldsymbol{y}_0,t_i;\hat{\omega}\right)\right),\ t_i=i\Delta t,\ \Delta t=\frac{2}{m}.\tag{36}$$

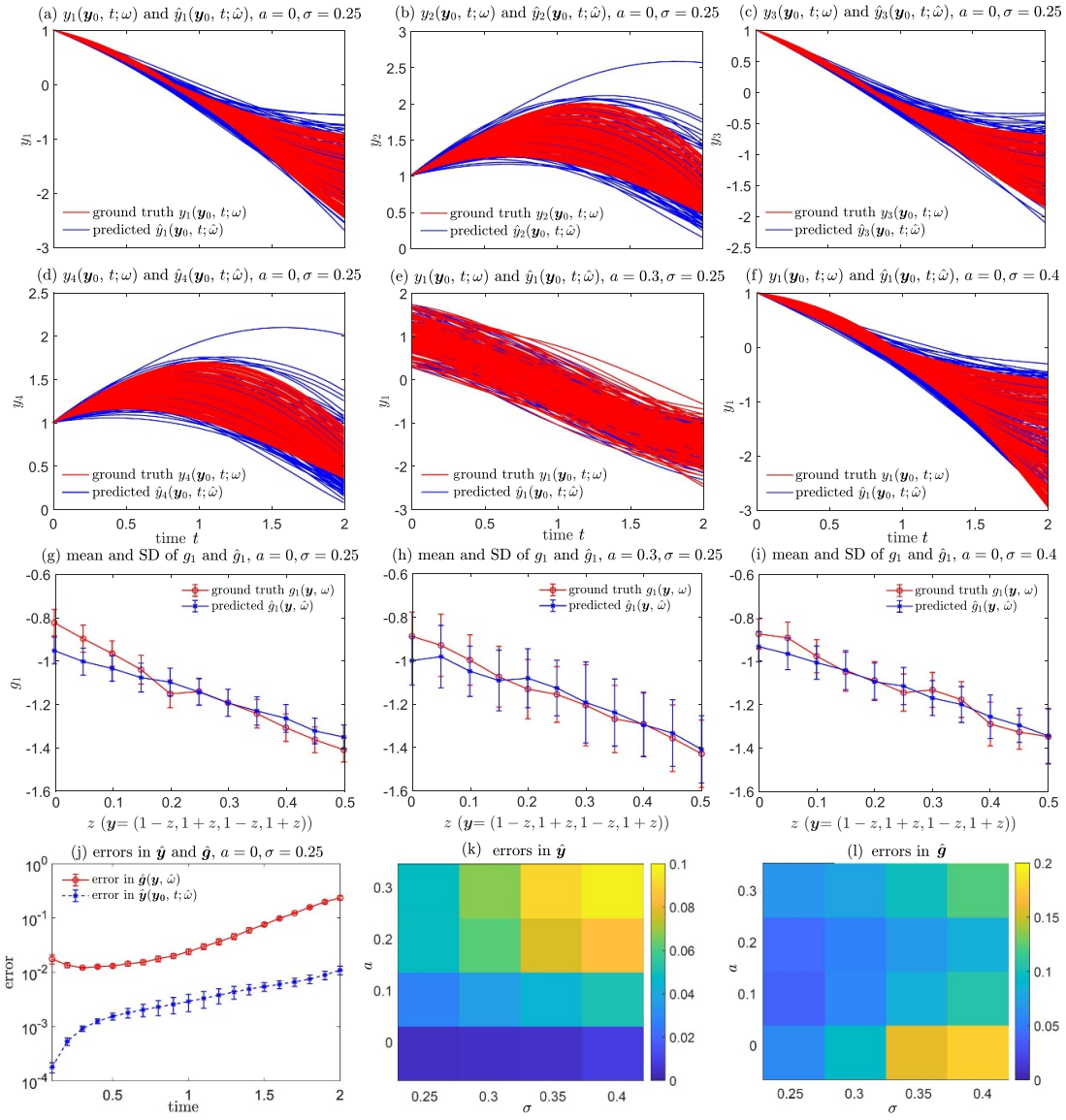

Figure 5. (a)–(d) Comparison between the ground truth $y_i(\boldsymbol{y}_0, t;\omega)$ and the predicted $\hat{y}_i(\boldsymbol{y}_0, t;\hat{\omega}), i=1,2,3,4$ when the standard deviation of the initial condition $a=0$ and $\sigma=0.25$ in the distribution of the model parameter ω. (e) Comparison between the ground truth $y_1(\boldsymbol{y}_0, t;\omega)$ and the predicted $\hat{y}_1(\boldsymbol{y}_0, t;\hat{\omega})$ $(a=0.3, \sigma=0.25)$. (f) The ground truth $y_1(\boldsymbol{y}_0, t;\omega)$ versus the predicted $\hat{y}_1(\boldsymbol{y}_0, t;\hat{\omega})$ $(a=0, \sigma=0.4)$. In (a)–(f), the ground truth $y_i(\boldsymbol{y}_0, t;\omega)$ are trajectories in the testing set and the predicted $\hat{y}_i(\boldsymbol{y}_0, t;\omega)$ are generated based on the initial conditions in the testing set (the testing set and the training set share the same a, σ). (g) Means and standard deviations of the ground truth $g_1(\boldsymbol{y}, \omega)$ versus the predicted $\hat{g}_1(\boldsymbol{y}, \hat{\omega})$ $(a=0, \sigma=0.25)$. (h) Means and standard deviations of the ground truth $g_1(\boldsymbol{y}, \hat{\omega})$ versus the predicted $\hat{g}_1(\boldsymbol{y}, \hat{\omega})$ $(a=0, \sigma=0.4)$. (i) Means and standard deviations of the ground truth $g_1(\boldsymbol{y}, \omega)$ versus the predicted $\hat{g}_1(\boldsymbol{y}, \hat{\omega})$ $(a=0.3, \sigma=0.25)$. In (g)–(i), we let $\boldsymbol{y}=(1-z, 1+z, 1-z, 1+z)$, $z=0.05i, i=0, \ldots, 10$. (j) The errors $\frac{\mathbb{E}\left[W_2^2(\mu_t, \hat{\mu}_t)\right]}{\mathbb{E}\left[\|\boldsymbol{y}(\boldsymbol{y}_0, t;\omega)\|^2\right]}$ (μ_t and $\hat{\mu}_t$ are the probability measures of $\boldsymbol{y}(\cdot, t;\omega)$ and $\hat{\boldsymbol{y}}(\cdot, t;\hat{\omega})$, respectively) and $\frac{\mathbb{E}\left[W_2^2\left(\eta_{\boldsymbol{y}(t),t}, \hat{\eta}_{\boldsymbol{y}(t),t}\right)\right]}{\mathbb{E}\left[\|\boldsymbol{g}(\boldsymbol{y}(t), \omega)\|^2\right]}$ in $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{g}}$ at different time t when $a=0, \sigma=0.25$. $\eta_{\boldsymbol{y}(t),t}$ and $\hat{\eta}_{\boldsymbol{y}(t),t}$ denote the distributions of $\boldsymbol{g}(\boldsymbol{y}(t), \omega)$ and $\hat{\boldsymbol{g}}(\boldsymbol{y}(t), \hat{\omega})$, respectively. (k) Errors in $\hat{\boldsymbol{y}}$ (defined in equation (37)) for different a and σ . (l) Errors in $\hat{\boldsymbol{g}}$ and $\hat{\boldsymbol{y}}$ (defined in equation (37)) for different a and σ. The errors of $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{g}}$ are evaluated on the testing sets.

Specifically, the time variable t is not used to determine the neighborhood of $\boldsymbol{y}(\boldsymbol{y}_0, t_i;\omega)$ in equation (36). The following error metrics:

$$\text{error in } \hat{\boldsymbol{y}} := \frac{\int_0^2 \tilde{W}_{2,\delta_0}^{2,\mathrm{e}}\left(\boldsymbol{y}(\boldsymbol{y}_0, s;\omega), \hat{\boldsymbol{y}}(\boldsymbol{y}_0, s;\hat{\omega})\right) \mathrm{d}s}{\int_0^2 \tilde{W}_{2,\delta_0}^{2,\mathrm{e}}\left(\boldsymbol{y}(\boldsymbol{y}_0, s;\omega), 0\right) \mathrm{d}s}, \quad \text{error in } \hat{\boldsymbol{g}} := \frac{\int_0^2 \mathbb{E}\left[W_2^2\left(\eta_{\boldsymbol{y}(s),s}, \hat{\eta}_{\boldsymbol{y}(s),s}\right)\right] \mathrm{d}s}{\int_0^2 \mathbb{E}\left[\|\boldsymbol{g}(\boldsymbol{y}(s), \omega)\|^2\right] \mathrm{d}s} \tag{37}$$

are used to quantify the errors of $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{g}}$ in the reconstructed ODE (30), respectively. We set $\delta=\delta_0=0.1$ in equations (36) and (37) and $m=100$ in equation (36).

Overall, by minimizing the time-averaged local squared W_2 distance equation (36) and using the neural network with weight uncertainty as $\hat{\boldsymbol{g}}$, the distribution of trajectories generated by our reconstructed model, equation (30), closely aligns with the distribution of trajectories generated by the ground truth ODE (35), across different values of a, σ. To demonstrate this, we plot the reconstructed $\hat{y}_i(\boldsymbol{y}_0, t; \hat{\omega})$ against the ground truth $y_i(\boldsymbol{y}_0, t; \omega), i = 1, 2, 3, 4$ when: $a = 0, \sigma = 0.25$ (figures 5(a)–(d)). We also plot the ground truth $y_1(\boldsymbol{y}_0, t; \omega)$ against the reconstructed $\hat{y}_1(\boldsymbol{y}_0, t; \hat{\omega})$ for $a = 0, \sigma = 0.4$ (figure 5(e)) and $a = 0.3, \sigma = 0.25$ (figure 5(f)). Additionally, the distributions of the ground truth $\boldsymbol{g}$ are effectively represented by the distribution of the reconstructed $\hat{\boldsymbol{g}}$ when inputting the same $\boldsymbol{y}$ for different values of σ and a. As an example, we plot the means and standard deviations of ground truth g_1 against those of the predicted $\hat{g}_1$ along the line $\boldsymbol{y} = (1 - z, 1 + z, 1 - z, 1 + z)$ when: $a = 0, \sigma = 0.25$ (figure 5(g)), $a = 0.3, \sigma = 0.25$ (figure 5(h)), and $a = 0, \sigma = 0.4$ (figure 5(i)). In figure 5(j), the error in $\hat{\boldsymbol{y}}$ grows over time due to error accumulation but is kept below 0.1 for all t. From figure 5(k), larger values of a and σ both correspond to larger errors in $\hat{\boldsymbol{y}}$. One potential explanation is that larger values of a and σ result in sparser training trajectories, making it more challenging to accurately reconstruct the underlying model equation (35). In figure 5(l), larger values of σ lead to larger errors in the reconstructed $\hat{\boldsymbol{g}}$, while a larger a does not necessarily lead to larger errors in $\hat{\boldsymbol{g}}$, i.e. the reconstruction of the RHS of the ODE (35) is not very sensitive to the distribution of the initial condition but could be less accurate when the intrinsic noise in the ground truth right-hand side of the ODE (35) is stronger.

4. Summary & conclusion

In our paper, we proposed a local squared W_2 method for reconstructing uncertainty models in UQ through minimizing a local squared W_2 loss equation (10). The local squared W_2 loss function could be efficiently evaluated using empirical distributions of observed data. We showcased the effectiveness of our approach across various UQ tasks and showed that it outperformed some benchmark methods.

As future directions, it would be promising to conduct further analysis to determine the optimal size of neighborhood δ in equation (10) as well as to identify an appropriate norm $|\cdot|_x$ for the input $\boldsymbol{x}$ for our method. The choice of a proper δ could be problem-specific and depend on the quantity and distribution of observed data. Additionally, entropy-regularized optimal transport methods-such as the Sinkhorn algorithm and Sinkhorn divergences [52, 53]-could be incorporated to solve the corresponding optimal transport problems more efficiently. In particular, these approaches may reduce the computational complexity to $\mathcal{O}(N\mathbb{E}[N^2(\boldsymbol{x}, \delta)])$ and offer improved statistical and computational performance in high-dimensional settings. Since computing the squared W_2 distance is independent in each neighborhood, it is worth further exploration on adopting recent GPU-based optimal transport solvers [54] or GPU-based entropy-regularized Wasserstein distance solvers [55] to parallelize the calculation of different squared W_2 distances in different neighborhoods to accelerate the computation of the loss function. Deriving theoretical justifications, such as a generalization error bound similar to that in theorem 2.3, for using an entropic-regularized Wasserstein distance could be a nontrivial problem, but a promising field of future research. It would be beneficial to develop an appropriate surrogate model as $\hat{\boldsymbol{f}}$ in equation (2). Furthermore, exploring the application of our local squared W_2 method to other UQ problems merits further investigation. For example, integrating our local squared W_2 method with recent stochastic differential equation reconstruction methods [56, 57] could enable the reconstruction of dynamical systems characterized by both uncertain parameters and intrinsic fluctuations. A further direction for future work is to strengthen the connection between our framework and Wasserstein robustness, distributionally robust optimization, and Wasserstein-regularized inverse problems. In particular, it would be interesting to investigate how ideas from Wasserstein ambiguity sets, robust UQ, and Wasserstein-based fidelity or regularization terms could be incorporated into our local squared W_2 framework, especially for inverse problems with significant observational uncertainty and model misspecification [58–62]. Another related direction is to investigate how Wasserstein-based robustness principles in neural network training can be incorporated into the learning procedure of our model [63]. Finally, we mainly focused on the reconstruction of lower-dimensional uncertainty models in this work, as the generalization error bound could become poor when the dimensionality d gets large in equations (11) and (12), and the computational cost could be high without using entropy-regularized Wasserstein distance when a large dataset is used. Additional techniques, such as dimension reduction techniques, could be necessary when applying Wasserstein-distance-type loss functions to higher-dimensional, large-scale problems [64], which is beyond the scope of this paper. Overall, it is a promising research direction to further explore how to adapt our proposed local squared W_2 method to the reconstruction of higher-dimensional uncertainty models.

Acknowledgments

The authors thank Prof. Tom Chou from UCLA and Prof. Alex Mogilner from New York University for their valuable suggestions on this work. This work was partially funded through the University of Houston startup Grant. This work was supported in part through the NYU IT High Performance Computing resources, services, and staff expertise.

Data availability statement

No new data were created in this research. The concrete compressive strength data used in section 3.3 are publicly available in [47] or at the following URL: https://archive.ics.uci.edu/dataset/165/concrete+compressive+strength. The code used in this research is publicly available at https://github.com/mtxia99/optimal_transport_stochastic_neural_network.

Author contributions

Mingtao Xia 0000-0002-2116-4712
Conceptualization (equal), Formal analysis (equal), Investigation (equal), Methodology (equal), Project administration (equal), Software (equal), Validation (equal), Writing – original draft (equal), Writing – review & editing (equal)

Qijing Shen
Formal analysis (equal), Investigation (equal), Methodology (equal), Writing – original draft (equal), Writing – review & editing (equal)

Appendix A. Proof to theorem 2.3

Here, we shall prove theorem 2.3. First, we have

$$\mathbb{E}\left[\left|\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})-\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\right|\right]\leqslant\mathbb{E}\left[\left|\tilde{W}_2^2(\boldsymbol{y},\hat{\boldsymbol{y}})-\tilde{W}_2^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\right|\right]+\mathbb{E}\left[\left|\tilde{W}_2^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})-\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\right|\right],\tag{38}$$

where

$$\tilde{W}_2^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}}):=\int_D W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\,\nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x}),\tag{39}$$

and $\nu^{\mathrm{e}}(\mathrm{d}x)$ is the empirical distribution of $\boldsymbol{x}$. For the first term in equation (38), the following inequality holds:

$$\begin{aligned}&\mathbb{E}\left[\left|\int_D W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\,\nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x})-\int_D W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\,\nu(\mathrm{d}\boldsymbol{x})\right|\right]\\&\quad\leqslant\mathbb{E}\left[\left(\int_D W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\,\nu^{\mathrm{e}}(\mathrm{d}\boldsymbol{x})-\int_D W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\,\nu(\mathrm{d}\boldsymbol{x})\right)^2\right]^{\frac{1}{2}}\\&\quad\leqslant\frac{1}{\sqrt{N}}\mathbb{E}\left[\left(W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})-\mathbb{E}\left[W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\right]\right)^2\right]^{\frac{1}{2}}\leqslant\frac{4M}{\sqrt{N}}.\end{aligned}\tag{40}$$

The last inequality holds because for any $\boldsymbol{x}\in D$, using the assumption equation (6), we have

$$0\leqslant W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}})\leqslant 2\left(\mathbb{E}\left[\|y(\boldsymbol{x};\omega)\|^2\right]+\mathbb{E}\left[\|\hat{y}(\boldsymbol{x};\hat{\omega})\|^2\right]\right)=4M.\tag{41}$$

Next, we estimate the second term in equation (38):

$$\mathbb{E}\left[\left|\int_D W_2^2(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}}))\nu^{\mathrm{e}}(\mathrm{d}x)-\int_D W_2^2(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}},\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}})\nu^{\mathrm{e}}(\mathrm{d}x)\right|\right].\tag{42}$$

We denote $\mu_{\boldsymbol{x},\delta}$ ($\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}$) to be the conditional distribution (empirical conditional distribution) of $\boldsymbol{y}(\tilde{\boldsymbol{x}};\omega)$ conditioned on $|\tilde{\boldsymbol{x}}-\boldsymbol{x}|\leqslant\delta$ and $\hat{\mu}_{\boldsymbol{x},\delta}$ ($\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}$) to be the conditional distribution (empirical conditional distribution) of $\hat{\boldsymbol{y}}(\tilde{\boldsymbol{x}};\hat{\omega})$ conditioned on $|\tilde{\boldsymbol{x}}-\boldsymbol{x}|\leqslant\delta$, respectively.

For any $\boldsymbol{x} \in D$, we have

$$\begin{aligned}\left|W_2^2\left(\mu_{\boldsymbol{x}}, \hat{\mu}_{\boldsymbol{x}}\right)-W_2^2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}, \hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right| &\leqslant\left|W_2\left(\mu_{\boldsymbol{x}}, \hat{\mu}_{\boldsymbol{x}}\right)-W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}, \hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right| \cdot\left(W_2\left(\mu_{\boldsymbol{x}}, \hat{\mu}_{\boldsymbol{x}}\right)+W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}, \hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right) \\ &\leqslant 4 \sqrt{M}\left|W_2\left(\mu_{\boldsymbol{x}}, \hat{\mu}_{\boldsymbol{x}}\right)-W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}, \hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right|.\end{aligned} \tag{43}$$

Using the triangle inequality of the Wasserstein distance in [65], for any $\boldsymbol{x}$, we have

$$\begin{aligned}\left|W_2\left(\mu_{\boldsymbol{x}}, \hat{\mu}_{\boldsymbol{x}}\right)-W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}, \hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right| &\leqslant\left|W_2\left(\hat{\mu}_{\boldsymbol{x}}, \mu_{\boldsymbol{x}}\right)-W_2\left(\hat{\mu}_{\boldsymbol{x}}, \mu_{\boldsymbol{x},\delta}\right)\right|+\left|W_2\left(\hat{\mu}_{\boldsymbol{x}}, \mu_{\boldsymbol{x},\delta}\right)-W_2\left(\mu_{\boldsymbol{x},\delta}, \hat{\mu}_{\boldsymbol{x},\delta}\right)\right| \\ &\quad+\left|W_2\left(\mu_{\boldsymbol{x},\delta}, \hat{\mu}_{\boldsymbol{x},\delta}\right)-W_2\left(\hat{\mu}_{\boldsymbol{x},\delta}, \mu_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right|+\left|W_2\left(\hat{\mu}_{\boldsymbol{x},\delta}, \mu_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)-W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}, \hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right|. \\ &\leqslant W_2\left(\mu_{\boldsymbol{x},\delta}, \mu_{\boldsymbol{x}}\right)+W_2\left(\hat{\mu}_{\boldsymbol{x},\delta}, \hat{\mu}_{\boldsymbol{x}}\right)+W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}}, \mu_{\boldsymbol{x},\delta}\right)+W_2\left(\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}, \hat{\mu}_{\boldsymbol{x},\delta}\right)\end{aligned} \tag{44}$$

We shall estimate the first term in the last inequality of equation (44). We define a new random variable $\tilde{\boldsymbol{y}}(\tilde{\boldsymbol{x}};\omega)$ coupled with $\boldsymbol{y}(\boldsymbol{x};\omega)$ such that given $\boldsymbol{x},\omega$ in $\boldsymbol{y}(\boldsymbol{x};\omega)$:

$$\tilde{\boldsymbol{y}}(\tilde{\boldsymbol{x}};\omega) := \boldsymbol{f}(\tilde{\boldsymbol{x}},\omega), \tag{45}$$

where the random variable $\tilde{\boldsymbol{x}}$ is independent of $\boldsymbol{x}$ and independent of ω, and we let $\tilde{\boldsymbol{x}}$ have a probability density $\frac{\mathbb{I}_{|\tilde{\boldsymbol{x}}-\boldsymbol{x}|\leqslant\delta}}{P(A_{\boldsymbol{x}})}\cdot\nu(\mathrm{d}\tilde{\boldsymbol{x}})$. A_x denotes the set $\{\tilde{\boldsymbol{x}}\in D : |\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x \leqslant \delta\}$ and $\mathbb{I}_{|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta}$ is the indicator function:

$$\mathbb{I}_{|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta}=1, |\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta,\ \ \mathbb{I}_{|\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x\leqslant\delta}=0, |\tilde{\boldsymbol{x}}-\boldsymbol{x}|_x>\delta. \tag{46}$$

Since $\boldsymbol{f}$ is Lipschitz in $\boldsymbol{x}$, we have

$$\left\|\tilde{\boldsymbol{y}}(\tilde{\boldsymbol{x}};\omega)-\boldsymbol{y}(\boldsymbol{x};\omega)\right\| \leqslant L\delta. \tag{47}$$

Additionally, the distribution of $\tilde{\boldsymbol{y}}$ is also $\mu_{\boldsymbol{x},\delta}$ because $\tilde{x}$ and ω are independent.

We take a special coupling probability measure $\tilde{\pi}_{\mu_{\boldsymbol{x}},\mu_{\boldsymbol{x},\delta}} = (\boldsymbol{y},\tilde{\boldsymbol{y}})_* P$ such that for all $A\in\mathcal{B}(\mathbb{R}^n\times\mathbb{R}^n)$,

$$\tilde{\pi}_{\mu_{\boldsymbol{x}},\mu_{\boldsymbol{x},\delta}}(A) = P\left((\boldsymbol{y},\tilde{\boldsymbol{y}})^{-1}(A)\right), \tag{48}$$

where $(\boldsymbol{y},\tilde{\boldsymbol{y}})$ is interpreted as a measurable map from $\Omega\times(\mathbb{R}^d\times\Omega)$ to $\mathbb{R}^n\times\mathbb{R}^n$. $(\boldsymbol{y},\tilde{\boldsymbol{y}})^{-1}(A)\in\Omega\times(\mathbb{R}^d\times\Omega)$ is the preimage of A under $(\boldsymbol{y},\tilde{\boldsymbol{y}})$ and $P\big((\boldsymbol{y},\tilde{\boldsymbol{y}})^{-1}(A)\big)$ is the probability measure of the set $(\boldsymbol{y},\tilde{\boldsymbol{y}})^{-1}(A)$. Therefore, we have

$$W_2\left(\mu_{\boldsymbol{x},\delta},\mu_{\boldsymbol{x}}\right)\leqslant\mathbb{E}_{(\boldsymbol{y},\tilde{\boldsymbol{y}})\sim\tilde{\pi}_{\mu_{\boldsymbol{x}},\mu_{\boldsymbol{x},\delta}}}\left[\left\|\boldsymbol{y}-\tilde{\boldsymbol{y}}\right\|^2\right]^{\frac{1}{2}}\leqslant L\delta. \tag{49}$$

Similarly, for the second term in the last inequality of equation (44), from the Lipschitz continuity condition of $\hat{\boldsymbol{f}}$ in assumption equation (7), we can also show that

$$W_2\left(\hat{\mu}_{\boldsymbol{x},\delta},\hat{\mu}_{\boldsymbol{x}}\right)\leqslant L\delta. \tag{50}$$

For the third and fourth terms in the last inequality of equation (44), from theorem 1 in [33], there exists a constant C such that

$$\mathbb{E}\left[W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}},\mu_{\boldsymbol{x},\delta}\right)\right]\leqslant\mathbb{E}\left[W_2^2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}},\mu_{\boldsymbol{x},\delta}\right)\right]^{\frac{1}{2}}\leqslant C\mathbb{E}\left[\|\boldsymbol{y}(\boldsymbol{x};\omega)\|_6^6\right]^{\frac{1}{6}}h\left(N(x,\delta),d\right)\leqslant C\sqrt{M}h\left(N(\boldsymbol{x},\delta),d\right) \tag{51}$$

and

$$\mathbb{E}\left[W_2\left(\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}},\hat{\mu}_{\boldsymbol{x},\delta}\right)\right]\leqslant\mathbb{E}\left[W_2^2\left(\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}},\hat{\mu}_{\boldsymbol{x},\delta}\right)\right]^{\frac{1}{2}}\leqslant C\mathbb{E}\left[\|\hat{\boldsymbol{y}}(\boldsymbol{x};\hat{\omega})\|_6^6\right]^{\frac{1}{6}}h\left(N(\boldsymbol{x},\delta),d\right)\leqslant C\sqrt{M}h\left(N(\boldsymbol{x},\delta),d\right), \tag{52}$$

respectively. Here, $\|\cdot\|_6$ is the l^6 norm of a vector and we have $\|\boldsymbol{y}\|_6\leqslant\|\boldsymbol{y}\|$. In equation (52), the function h is defined as:

$$h(N,d)=\begin{cases}N^{-\frac{1}{4}}\log(1+N)^{\frac{1}{2}}, d\leqslant 4,\\ N^{-\frac{1}{d}}, d>4.\end{cases} \tag{53}$$

Plugging equations (51) and (52), equations (49), and (50) into equation (44), we have proved that:

$$\mathbb{E}\left[\left|W_2\left(\mu_{\boldsymbol{x}},\hat{\mu}_{\boldsymbol{x}}\right)-W_2\left(\mu_{\boldsymbol{x},\delta}^{\mathrm{e}},\hat{\mu}_{\boldsymbol{x},\delta}^{\mathrm{e}}\right)\right|\right]\leqslant 2C\sqrt{M}h\left(N(\boldsymbol{x},\delta),d\right)+2L\delta. \tag{54}$$

Therefore, combining the two inequalities equations (43) and (54) and taking the expectation of equation (54) w.r.t. the empirical distribution $\nu^{\mathrm{e}}(\boldsymbol{x})$, we have

$$\mathbb{E}\left[\left|\tilde{W}_2^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})-\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\right|\right]\leqslant 8CM\mathbb{E}\left[h\left(N(\boldsymbol{x},\delta),d\right)\right]+8\sqrt{M}L\delta. \tag{55}$$

Combining the two inequalities equations (40) and (55), the inequality (11) holds, completing the proof of theorem 2.3.

Appendix B. Definitions of different loss metrics

Below, we provide descriptions and definitions for different loss functions used in this study. In the following, N denotes the number of samples.

1. The squared W_2 distance

$$W_2^2(\mu,\hat{\mu})\approx W_2^2(\mu^{\mathrm{e}},\hat{\mu}^{\mathrm{e}}),$$

where μ^{e} and $\hat{\mu}^{\mathrm{e}}$ are the empirical distributions of $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$, respectively. It is estimated by

$$W_2^2(\mu_N^{\mathrm{e}},\hat{\mu}_N^{\mathrm{e}})\approx \texttt{ot.emd2}\left(\frac{1}{N}\boldsymbol{I}_N,\frac{1}{N}\boldsymbol{I}_N,\boldsymbol{C}\right), \tag{56}$$

where `ot.emd2` is the function for solving the earth mover's distance problem in the `POT` package of Python in [38]. N is the number of ground truth and predicted samples, $\boldsymbol{I}_N$ is an N-dimensional vector whose elements are all 1, and $\boldsymbol{C}\in\mathbb{R}^{N\times N}$ is a matrix with entries $(\boldsymbol{C})_{ij}=\|\boldsymbol{y}_i-\hat{\boldsymbol{y}}_j\|^2$. $\boldsymbol{y}_i$ and $\hat{\boldsymbol{y}}_j$ are the ground truth data and prediction associated with $\boldsymbol{x}_j$, respectively.

2. MMD (maximum mean discrepancy) [66]:

$$\mathrm{MMD}(\boldsymbol{y},\hat{\boldsymbol{y}})=\mathbb{E}\left[K(\boldsymbol{y},\boldsymbol{y})\right]-2\mathbb{E}\left[K(\boldsymbol{y},\hat{\boldsymbol{y}})\right]+\mathbb{E}\left[K(\hat{\boldsymbol{y}},\hat{\boldsymbol{y}})\right],$$

where K is the standard radial basis function (or Gaussian kernel) with the multiplier 2 and number of kernels 5. $\boldsymbol{y}$ and $\hat{\boldsymbol{y}}$ are the ground truth observation and prediction, respectively.

3. Mean squared error (MSE): where N is the total number of data:

$$\mathrm{MSE}(\boldsymbol{y},\hat{\boldsymbol{y}})=\sum_{i=1}^{N}\frac{1}{N}\|y_i-\hat{y}_i\|^2.$$

4. Mean^2+var loss function:

$$\left(\mathrm{Mean}^2+\mathrm{var}\right)(\boldsymbol{y},\hat{\boldsymbol{y}})=\sum_{i=1}^{N}\sum_{i=1}^{N}\frac{1}{N}\|y_i-\hat{y}_i\|^2+\left|\mathrm{Var}(\boldsymbol{y})-\mathrm{Var}(\hat{\boldsymbol{y}})\right|$$

where

$$\mathrm{Var}(\boldsymbol{y})=\sum_{i=1}^{N}\left\|\boldsymbol{y}_i-\sum_{i=1}^{N}\frac{1}{N}\boldsymbol{y}_i\right\|^2,\ \ \mathrm{Var}(\hat{\boldsymbol{y}})=\sum_{i=1}^{N}\left\|\hat{\boldsymbol{y}}_i-\sum_{i=1}^{N}\frac{1}{N}\hat{\boldsymbol{y}}_i\right\|^2. \tag{57}$$

5. The local squared W_2 distance

$$\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})$$

defined in equation (10). It is estimated by

$$\tilde{W}_{2,\delta}^{2,\mathrm{e}}(\boldsymbol{y},\hat{\boldsymbol{y}})\approx\sum_{i=1}^{N}\frac{1}{N}\texttt{ot.emd2}\left(\frac{1}{N(\boldsymbol{x}_i,\delta)}\boldsymbol{I}_{N(\boldsymbol{x}_i,\delta)},\frac{1}{N(\boldsymbol{x}_i,\delta)}\boldsymbol{I}_{N(\boldsymbol{x}_i,\delta)},\boldsymbol{C}(\boldsymbol{x}_i)\right), \tag{58}$$

where `ot.emd2` is the function for solving the earth mover's distance problem. N is the number of ground truth and predicted samples, $\boldsymbol{I}_{N(\boldsymbol{x}_i,\delta)}$ is an $N(\boldsymbol{x}_i,\delta)$-dimensional vector whose elements are all 1, and $\boldsymbol{C}(\boldsymbol{x}_i)\in\mathbb{R}^{N(\boldsymbol{x}_i,\delta)\times N(\boldsymbol{x}_i,\delta)}$. For each $\boldsymbol{x}_i$, we denote $S(\boldsymbol{x}_i,\delta):=\{\boldsymbol{x}_{i_1},\ldots,\boldsymbol{x}_{i_{N(\boldsymbol{x}_i,\delta)}}\}$ to be the set such

that $S(\boldsymbol{x}_i,\delta)=\{\tilde{x}\in S:|\tilde{\boldsymbol{x}}-\boldsymbol{x}_i|_x\leqslant\delta\}$. The entries in $\boldsymbol{C}$ are: $(\boldsymbol{C})_{sj}=\|\boldsymbol{y}_{i_s}-\hat{\boldsymbol{y}}_{i_j}\|^2$. $\|\cdot\|$ is the l^2 norm of a vector. $|\cdot|_x$ is the norm of the input $\boldsymbol{x}$ (specified in each example). $\boldsymbol{y}_{i_s}$ and $\hat{\boldsymbol{y}}_{i_s}$ are the ground truth $\boldsymbol{y}_{i_s}=\boldsymbol{f}(\boldsymbol{x}_{i_s},\omega_{i_s})$ and predicted $\hat{\boldsymbol{y}}_{i_s}=\hat{\boldsymbol{f}}(\boldsymbol{x}_{i_s},\hat{\omega}_{i_s})$ for $\boldsymbol{x}_{i_s}\in S(\boldsymbol{x}_i,\delta),s=1,\ldots,N(\boldsymbol{x}_i,\delta)$.

6. Local MMD:

$$\mathrm{MMD}_\delta(\boldsymbol{y},\hat{\boldsymbol{y}})=\sum_{i=1}^{N}\frac{1}{N}\left(\mathbb{E}\left[K\left(\boldsymbol{y}\left[\boldsymbol{x}_i,\delta\right],\boldsymbol{y}\left[\boldsymbol{x}_i,\delta\right]\right)\right]-2\mathbb{E}\left[K\left(\boldsymbol{y}\left[\boldsymbol{x}_i,\delta\right],\hat{\boldsymbol{y}}\left[\boldsymbol{x}_i,\delta\right]\right)\right]+\mathbb{E}\left[K\left(\hat{\boldsymbol{y}}\left[\boldsymbol{x}_i,\delta\right],\hat{\boldsymbol{y}}\left[\boldsymbol{x}_i,\delta\right]\right)\right]\right),$$

where K is the standard radial basis function (or Gaussian kernel) with multiplier 2 and number of kernels 5. $\boldsymbol{y}[\boldsymbol{x}_i,\delta]$ is the set of ground truth $\left\{\boldsymbol{y}_{i_s}=\boldsymbol{f}(\boldsymbol{x}_{i_s};\omega_{i_s})\right\}$ such that $\boldsymbol{x}_{i_s}\in S(\boldsymbol{x}_i,\delta)$. $\hat{\boldsymbol{y}}[\boldsymbol{x}_{i_s},\delta]$ is the set of reconstructed $\left\{\hat{\boldsymbol{y}}_{i_s}=\hat{\boldsymbol{f}}(\boldsymbol{x}_{i_s};\hat{\omega}_{i_s})\right\}$ such that $\boldsymbol{x}_{i_s}\in S(\boldsymbol{x}_i,\delta)$. $S(\boldsymbol{x}_i,\delta):=\{\boldsymbol{x}_{i_1},\ldots,\boldsymbol{x}_{i_{N(\boldsymbol{x}_i,\delta)}}\}$ has the same meaning as defined in the local squared W_2 distance.

7. Local MSE:

$$\mathrm{MSE}_\delta(\boldsymbol{y},\hat{\boldsymbol{y}})=\sum_{i=1}^{N}\frac{1}{N}\mathrm{MSE}\Big(\boldsymbol{y}\left[\boldsymbol{x}_i,\delta\right],\boldsymbol{y}\left[\boldsymbol{x}_i,\delta\right)]\Big),$$

where $\boldsymbol{y}[\boldsymbol{x}_i,\delta]$ and $\hat{\boldsymbol{y}}[\boldsymbol{x}_i,\delta]$ have the same meaning as defined in the local MMD loss function.

8. Local mean^2+var:

$$(\mathrm{Mean}^2+\mathrm{var})_\delta(\boldsymbol{y},\hat{\boldsymbol{y}})=\sum_{i=1}^{N}\frac{1}{N}(\mathrm{Mean}^2+\mathrm{var})\Big(\boldsymbol{y}\left[\boldsymbol{x}_i,\delta\right],\boldsymbol{y}\left[\boldsymbol{x}_i,\delta\right)]\Big),$$

where $\boldsymbol{y}[\boldsymbol{x}_i,\delta]$ and $\hat{\boldsymbol{y}}[\boldsymbol{x}_i,\delta]$ have the same meaning as defined in the local MMD loss function.

Appendix C. How distributions of model parameters and the input affects the accuracy of reconstructing equation (25)

We carry out two additional experiments on reconstructing equation (25) in section 3.2, aiming to investigate how the distributions of the model parameters (ω_1,ω_2) and the input x affect the accuracy of reconstructing the nonlinear model equation (25).

First, we varied the distribution of the uncertain parameters (ω_1,ω_2). We set $x\sim\mathcal{U}(-0.5,0.5)$ and $(\omega_1,\omega_2)^{\mathrm{T}}\sim\mathcal{N}((19.1426,0.5311)^{\mathrm{T}},\sigma^2I_2)$ in equation (25), where $I_2\in\mathbb{R}^{2\times2}$ is the identity matrix, to generate the training samples. For the testing samples, we set $x\in\{x_i:x_i=-0.5+0.1i,i=0,\ldots,10\}$ and $(\omega_1,\omega_2)^{\mathrm{T}}\sim\mathcal{N}\left((19.1426,0.5311)^{\mathrm{T}},\sigma^2I_2\right)$. At each x_i, 100 testing samples are generated. The variables x and (ω_1,ω_2) were independently sampled for both the training and testing sets. We varied σ, the standard deviation of the latent model parameters ω_1 and ω_2, for both the training and testing samples.

Second, we alter the distribution of x in the training set. We let $(\omega_1,\omega_2)\sim\mathcal{N}\left((19.1426,0.5311)^{\mathrm{T}},I_2\right)$ and sample $x\sim\mathcal{U}(-a,a)$ with different a for the training set. For testing, we generate a testing set $T=\cup_{i=0}^{10}T_i$ with each T_i containing 100 samples $(x_{r,i},y(x_{r,i},\omega)),x_{r,i}=0.1i-0.5,r=1,\ldots,100$ and $(\omega_1,\omega_2)\sim\mathcal{N}\left((19.1426,0.5311)^{\mathrm{T}},I_2\right)$. x and (ω_1,ω_2) are independently sampled for both training and testing sets.

For both experiments, we use the same neural network model with weight uncertainty, training settings, and hyperparameters as used in section 3.2 (listed in table 2). The errors in the predicted mean $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and standard deviation $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$ are calculated on the testing set.

As depicted in figure 6(a) and (b), larger standard deviations in the model parameters (ω_1,ω_2) and a larger standard deviation in the input x of the training set both result in larger errors in the predicted mean $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and in the predicted standard deviation $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$. A possible reason could be that larger standard deviations in the model parameters (ω_1,ω_2) and a larger standard deviation in the input x both lead to more sparsely distributed training samples, making it more difficult to reconstruct the underlying nonlinear model equation (25).

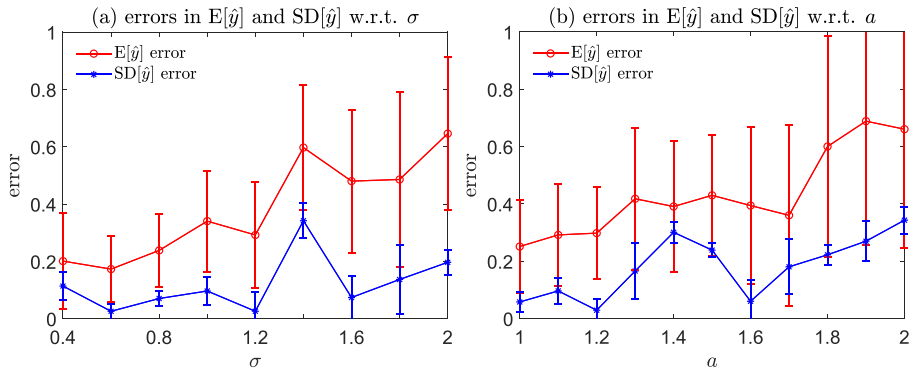

Figure 6. (a) The errors in the predicted mean $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and predicted standard deviation $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$ w.r.t. the standard deviations of the uncertain model parameters (ω_1, ω_2). (b) The errors in the predicted mean $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and predicted standard deviation $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$ when varying the standard deviation of x in the training set. The errors are evaluated on the testing set.

Appendix D. Neural network architecture

We carry out an additional experiment for section 3.2 to explore how the structure of the neural network model in figure 1 affects the accuracy of reconstructing the nonlinear model equation (25). The local squared W_2 distance equation (10) is used as the loss function for training the neural network with the size of the neighborhood $\delta = 0.1$. We change the number of hidden layers as well as the number of neurons per hidden layer. Additionally, we compare the performance of the ResNet technique against the standard feed-forward structure for forward propagation.

Table 4. Means and standard deviations of errors in the predicted $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$ in section 3.2, calculated over 5 independent experiments. The errors are calculated on the same testing set as used in section 3.2.

Width	Depth	Error in $\mathbb{E}[\hat{y}(x;\hat{\omega})]$	Error in $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$	Runtime (s)
12	4(ResNet)	0.053 ± 0.013	0.127 ± 0.023	3099 ± 561
25	4(ResNet)	0.035 ± 0.005	0.126 ± 0.026	2469 ± 502
50	4(ResNet)	0.051 ± 0.013	0.114 ± 0.012	3346 ± 213
100	4(ResNet)	0.232 ± 0.409	0.190 ± 0.138	3706 ± 781
50	1(ResNet)	0.044 ± 0.005	0.106 ± 0.018	2853 ± 593
50	2(ResNet)	0.034 ± 0.003	0.106 ± 0.034	2801 ± 299
50	3(ResNet)	0.043 ± 0.010	0.115 ± 0.024	2428 ± 499
50	1(feed-forward)	0.049 ± 0.005	0.106 ± 0.023	2503 ± 90
50	2(feed-forward)	0.044 ± 0.011	0.129 ± 0.019	2526 ± 560
50	3(feed-forward)	0.075 ± 0.031	0.161 ± 0.020	2526 ± 560
50	4(feed-forward)	0.060 ± 0.028	0.175 ± 0.022	2820 ± 622

In table 4, the errors of the predicted mean and standard deviation, $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$, may increase as the number of hidden layers in the neural network increases if the ResNet technique is not implemented. However, when the ResNet technique is employed, the errors of the predicted mean and standard deviation do not deteriorate as the number of hidden layers increases. This improvement may be attributed to the ResNet technique, mitigating the gradient vanishing issue [67] that affects simple feed-forward neural networks. On the other hand, if the number of neurons is too small or too large, then the errors in $\mathbb{E}[\hat{y}(x;\hat{\omega})]$ and $\mathrm{SD}[\hat{y}(x;\hat{\omega})]$ becomes large. A too-small number of neurons per layer could be insufficient, while an excessively large number of neurons can complicate the optimization of weights under uncertainty. A neural network with 3 hidden layers and 50 neurons in each layer, equipped with the ResNet technique, appears to be effective configuration for reconstructing the nonlinear model equation (25).

Appendix E. Proof to proposition 3.1

Below, we provide proof to proposition 3.1. First, note that

$$\begin{aligned}
\mathrm{d}\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2 &= 2\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega}),\boldsymbol{g}\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),t,\omega\right)-\hat{\boldsymbol{g}}\left(\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega}),t,\hat{\omega}\right)\right)\\
&\leqslant \|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2+\|\boldsymbol{g}\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),t,\omega\right)\\
&\quad-\hat{\boldsymbol{g}}\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\hat{\omega}),t,\hat{\omega}\right)+\hat{\boldsymbol{g}}\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\hat{\omega}),t,\hat{\omega}\right)-\hat{\boldsymbol{g}}\left(\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega}),t,\hat{\omega}\right)\|^2\\
&\leqslant \left(1+2L_g^2\right)\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2+2\|\boldsymbol{g}\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),t,\omega\right)-\hat{\boldsymbol{g}}(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),t,\hat{\omega})\|^2\\
&\leqslant (1+2L_g^2)\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2+2\|\boldsymbol{g}(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),t,\omega)-\boldsymbol{g}(0,0,\omega)\\
&\quad+\boldsymbol{g}(0,0,\omega)-\hat{\boldsymbol{g}}(0,0,\hat{\omega})+\hat{\boldsymbol{g}}(0,0,\hat{\omega})-\hat{\boldsymbol{g}}(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),t,\hat{\omega})\|^2\\
&\leqslant (1+2L_g^2)\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2\\
&\quad+6\|\boldsymbol{g}(0,0,\omega)-\hat{\boldsymbol{g}}(0,0,\hat{\omega})\|^2+12L_g^2\left(\|\boldsymbol{y}(\boldsymbol{y}_0,s;\omega)\|^2+t^2\right),
\end{aligned}\tag{59}$$

where $(\cdot,\cdot)$ denotes the inner product of two n-dimensional vectors. By applying the Gronwall's inequality to the quantity $\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2$ in equation (59), we can conclude that:

$$\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2\leqslant 6\exp\left(t+2L_g^2t\right)\cdot\int_0^t\left(\|\boldsymbol{g}(0,0,\omega)-\hat{\boldsymbol{g}}(0,0,\hat{\omega})\|^2+2L_g^2\left(\|\boldsymbol{y}(\boldsymbol{y}_0,s;\omega)\|^2+s^2\right)\right)\mathrm{d}s.\tag{60}$$

In equation (60), $\left(\boldsymbol{g}(0,0,\omega),\hat{\boldsymbol{g}}(0,0,\hat{\omega})\right)$ is seen as a random variables in $\mathbb{R}^{2n}$. For any coupling probability measure $\pi_{\eta_{0,0},\hat{\eta}_{0,0}}\left(\boldsymbol{g}(0,0,\omega),\hat{\boldsymbol{g}}(0,0,\hat{\omega})\right)$ such that its marginal distributions are $\eta_{0,0}$ and $\hat{\eta}_{0,0}$ ($\eta_{\boldsymbol{y},s}$ and $\hat{\eta}_{\boldsymbol{y},s}$ denote the probability measures of $\boldsymbol{g}(\boldsymbol{y},s,\omega)$ and $\hat{\boldsymbol{g}}(\boldsymbol{y},s,\hat{\omega})$, respectively), we denote $\pi^*\left(\omega,\hat{\omega}\right)$ such that

$$\pi^*(A,B)=\pi_{\eta_{0,0},\hat{\eta}_{0,0}}\left(\left(\boldsymbol{g}(0,0,\omega),\hat{\boldsymbol{g}}(0,0,\hat{\omega})\right)(A,B)\right),\tag{61}$$

where $A\in\mathcal{B}(\Omega),B\in\mathcal{B}(\hat{\Omega})$. In other words, $\pi_{\eta_{0,0},\hat{\eta}_{0,0}}$ is the pushforward measure of π^*. Here, we consider $\left(\boldsymbol{g}(0,0,\omega),\hat{\boldsymbol{g}}(0,0,\hat{\omega})\right)$ as a measurable map from $\Omega\times\hat{\Omega}$ to $\mathbb{R}^n\times\mathbb{R}^n$. Specifically, if we take the expectation of equation (60) and taking the infimum over all $\pi_{\eta_{0,0},\hat{\eta}_{0,0}}(\boldsymbol{g}(0,0,\omega),\hat{\boldsymbol{g}}(0,0,\hat{\omega}))$, we conclude that

$$\begin{aligned}
&\mathbb{E}_{(\omega,\hat{\omega})\sim\pi^*(\omega,\hat{\omega})}\left[\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\|^2\right]\\
&\quad\leqslant 6\exp\left(t+2L_g^2t\right)\int_0^t W_2^2\left(\eta_{0,0},\hat{\eta}_{0,0}\right)+2L_g^2\left(\mathbb{E}\left[\|\boldsymbol{y}(\boldsymbol{y}_0,s;\omega)\|^2\right]+s^2\right)\mathrm{d}s\\
&\quad\leqslant 12L_g^2\exp\left(t+2L_gt\right)\int_0^t\left(\mathbb{E}\left[\|\boldsymbol{y}(\boldsymbol{y}_0,s;\omega)\|^2\right]+s^2\right)\mathrm{d}s+6t\exp\left(t+2L_g^2t\right)\cdot\sup_{\boldsymbol{y},0\leqslant s\leqslant t}W_2^2\left(\eta_{\boldsymbol{y},s},\hat{\eta}_{\boldsymbol{y},s}\right).
\end{aligned}\tag{62}$$

It is easy to verify that the marginal distributions of $\pi^*(\omega,\hat{\omega})$ are the distributions of ω and $\hat{\omega}$, respectively. We denote $\pi^\sharp(C,D)$, $(C,D)\in\mathcal{B}(\mathbb{R}^n)\times\mathcal{B}(\mathbb{R}^n)$ to be the pushforward probability measure of π^* such that

$$\pi^\sharp(C,D)=\pi^*\left(\left(\boldsymbol{y}(\boldsymbol{y}_0,t;\omega),\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\omega)\right)^{-1}(C,D)\right),\tag{63}$$

where $\left(\boldsymbol{y}(\boldsymbol{y}_0,t,\omega),\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t,\omega)\right)^{-1}(C,D)$ is the preimage of (C,D) in $\mathcal{B}(\Omega)\times\mathcal{B}(\hat{\Omega})$. Additionally, we can verify that the marginal distributions of $\pi^\sharp$ are $\mu_{\boldsymbol{y}_0,t}$ and $\hat{\mu}_{\boldsymbol{y}_0,t}$, respectively. Therefore, using the inequality (62), the following inequality holds:

$$\begin{aligned}
W_2^2\left(\mu_{\boldsymbol{y}_0,t},\hat{\mu}_{\boldsymbol{y}_0,t}\right)&\leqslant\mathbb{E}_{(\boldsymbol{y},\hat{\boldsymbol{y}})\sim\pi^\sharp(\boldsymbol{y},\hat{\boldsymbol{y}})}\left[\left\|\boldsymbol{y}-\hat{\boldsymbol{y}}\right\|^2\right]=\mathbb{E}_{(\omega,\hat{\omega})\sim\pi^*(\omega,\hat{\omega})}\left[\left\|\boldsymbol{y}(\boldsymbol{y}_0,t;\omega)-\hat{\boldsymbol{y}}(\boldsymbol{y}_0,t;\hat{\omega})\right\|^2\right]\\
&\leqslant 12L_g^2\exp\left(t+2L_g^2t\right)\cdot\int_0^t\left(\mathbb{E}\left[\|\boldsymbol{y}(\boldsymbol{y}_0,s;\omega)\|^2\right]+s^2\right)\mathrm{d}s\\
&\quad+6t\exp\left(1+2L_g^2t\right)\cdot\sup_{\boldsymbol{y},0\leqslant s\leqslant t}W_2^2\left(\eta_{\boldsymbol{y},s},\hat{\eta}_{\boldsymbol{y},s}\right),
\end{aligned}\tag{64}$$

which completes the proof of proposition 3.1. The two constants C_0,C_1 in proposition 3.1 are:

$$C_0:=6t\exp\left(1+2L_g^2t\right),\quad C_1:=12L_g^2\exp\left(t+2L_g^2t\right)\cdot\int_0^t\left(\mathbb{E}\left[\|\boldsymbol{y}(\boldsymbol{y}_0,s;\omega)\|^2\right]+s^2\right)\mathrm{d}s.\tag{65}$$

References

[1] Fuller W A 2009 *Measurement Error Models* (Wiley)
[2] Carroll R J, Ruppert D and Stefanski L A 1995 *Measurement Error in Nonlinear Models* vol 105 (CRC Press)
[3] Schennach S M 2004 Estimation of nonlinear models with measurement error *Econometrica* **72** 33–75
[4] Bishop C M 1998 Latent variable models *Learning in Graphical Models* (Springer) pp 371–403
[5] Yang X, Liu Y and Park G-K 2020 Parameter estimation of uncertain differential equation with application to financial market *Chaos Solitons Fractals* **139** 110026
[6] Hadi Aliakbarpour V B S P, Palaniappan K, Seetharaman G and Dias J 2016 Heterogeneous multi-view information fusion: review of 3-D reconstruction methods and a new registration with uncertainty modeling *IEEE Access* **4** 8264–85
[7] Ansari M A *et al* 2017 Genome-to-genome analysis highlights the effect of the human innate and adaptive immune systems on the hepatitis C virus *Nat. Genet.* **49** 666–73
[8] Arridge S, Maass P, Öktem O and Schönlieb C-B 2019 Solving inverse problems using data-driven models *Acta Numer.* **28** 1–174
[9] Leuschner J, Schmidt M, Baguer D O and Maass P 2021 LoDoPaB-CT, a benchmark dataset for low-dose computed tomography reconstruction *Sci. Data* **8** 109
[10] Stuart A M 2010 Inverse problems: a Bayesian perspective *Acta Numer.* **19** 451–559
[11] Romero J, Heidrich W and Ravasi M 2025 Bayesian seismic inversion with implicit neural representations *Geophys. J. Int.* **242** ggaf249
[12] Cheng S *et al* 2023 Machine learning with data assimilation and uncertainty quantification for dynamical systems: a review *IEEE/CAA J. Autom. Sinica* **10** 1361–87
[13] Zhang E, Antoni Jérôme and Feissel P 2012 Bayesian force reconstruction with an uncertain model *J. Sound Vib.* **331** 798–814
[14] Kuczera G and Parent E 1998 Monte Carlo assessment of parameter uncertainty in conceptual catchment models: the Metropolis algorithm *J. Hydrol.* **211** 69–85
[15] Arcieri G, Hoelzl C, Schwery O, Straub D, Papakonstantinou K G and Chatzi E 2023 Bridging POMDPs and Bayesian decision making for robust maintenance planning under model uncertainty: an application to railway systems *Reliab. Eng. Syst. Saf.* **239** 109496
[16] Goan E and Fookes C 2020 Bayesian neural networks: an introduction and survey *Case Studies in Applied Bayesian Data Science: Cirm Jean-Morlet Chair, Fall 2018* (Springer) pp 45–87
[17] Shang R, O'Brien M A, Wang F, Situ G and Luke G P 2023 Approximating the uncertainty of deep learning reconstruction predictions in single-pixel imaging *Commun. Eng.* **2** 53
[18] Villani C *et al* 2009 *Optimal Transport: Old and New* vol 338 (Springer)
[19] Zheng W, Wang F-Y and Gou C 2020 Nonparametric different-feature selection using Wasserstein distance *2020 IEEE 32nd Int. Conf. on Tools With Artificial Intelligence (ICTAI)* (IEEE) pp 982–8
[20] Kidger P, Foster J, Li X and Lyons T J 2021 Neural SDEs as infinite-dimensional GANs *Int. Conf. on Machine Learning* (PMLR) pp 5453–63
[21] Frogner C, Zhang C, Mobahi H, Araya M and Poggio T A 2015 Learning with a Wasserstein loss *Advances in Neural Information Processing Systems* vol **28**
[22] Adler J, Ringh A, Öktem O, and Karlsson J 2017 Learning to solve inverse problems using Wasserstein loss (arXiv:1710.10898)
[23] Gao Y and Michael K N 2022 Wasserstein generative adversarial uncertainty quantification in physics-informed neural networks *J. Comput. Phys.* **463** 111270
[24] Jin Q, Luo X, Shi Y and Kita K 2019 Image generation method based on improved condition GAN *2019 6th Int. Conf. on Systems and Informatics (ICSAI)* (IEEE) pp 1290–4
[25] Wang J, Jiale W, Huang X and Xiong Z 2023 Improved WGAN for image generation methods *Int. Conf. on Mobile Networks and Management* (Springer) pp 199–211
[26] Saxena D and Cao J 2021 Generative adversarial networks (GANs): challenges, solutions and future directions *ACM Comput. Surv. (CSUR)* **54** 1–42
[27] Chemseddine J, Hagemann P, Steidl G and Wald C 2025 Conditional Wasserstein Distances with Applications in Bayesian OT Flow Matching *J. Mach. Learn. Res.* **26** 1–47
[28] Kim Y-geun, Lee K and Paik M C 2023 Conditional Wasserstein generator *IEEE Trans. Pattern Anal. Mach. Intell.* **45** 7208–19
[29] Yang L, Han D and Wang P 2022 Imprecise probabilistic model updating using a Wasserstein distance-based uncertainty quantification metric *J. Mech. Eng.* **58** 300–11
[30] Xia M, Shen Q, Maini P, Gaffney E and Mogilner A 2025 A new local time-decoupled squared Wasserstein-2 method for training stochastic neural networks to reconstruct uncertain parameters in dynamical systems (arXiv:2503.05068)
[31] Bouallegue G and Djemal R 2020 EEG data augmentation using Wasserstein GAN *2020 20th Int. Conf. on Sciences and Techniques of Automatic Control and Computer Engineering (STA)* (IEEE) pp 40–45
[32] Croitoru F-A, Hondru V, Ionescu R T and Shah M 2023 Diffusion models in vision: a survey *IEEE Trans. Pattern Anal. Mach. Intell.* **45** 10850–69
[33] Fournier N and Guillin A 2015 On the rate of convergence in Wasserstein distance of the empirical measure *Probab. Theory Relat. Fields* **162** 707–38
[34] Kolouri S, Nadjahi K, Simsekli U, Badeau R and Rohde G 2019 Generalized sliced Wasserstein distances *Advances in Neural Information Processing Systems* vol 32
[35] Xia M and Shen Q 2025 Efficient reconstruction of multidimensional random field models with heterogeneous data using stochastic neural networks (arXiv:2511.13977)
[36] Kaiming H, Zhang X, Ren S and Sun J 2016 Deep residual learning for image recognition *Proc. IEEE Conf. on Computer Vision and Pattern Recognition* pp 770–8
[37] Dowson D C and Landau B 1982 The Fréchet distance between multivariate normal distributions *J. Multivariate Anal.* **12** 450–5
[38] Flamary R *et al* 2021 POT: Python optimal transport *J. Mach. Learn. Res.* **22** 1–8
[39] Raftery A, Hoeting J and Madigan D 1993 *Model Selection and Accounting for Model Uncertainty in Linear Regression Models* (Citeseer)
[40] Rooney W C and Biegler L T 2001 Design for model parameter uncertainty using nonlinear confidence regions *AIChE J.* **47** 1794–804
[41] Bates D M 1988 *Nonlinear Regression Analysis and Its Applications* (Wiley)
[42] Mullachery V, Khera A and Husain A 2018 Bayesian neural networks (arXiv:1801.07710)

[43] Bayesian neural network derivation 2020 (available at: www.zhihu.com/tardis/bd/art/263053978)
[44] Arjovsky M, Chintala S and Bottou L 2017 Wasserstein generative adversarial networks *Int. Conf. on Machine Learning* (PMLR) pp 214–23
[45] Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V and Courville A C 2017 Improved training of Wasserstein GANs *Advances in Neural Information Processing Systems* vol 30
[46] Hoffer E, Hubara I and Soudry D 2017 Train longer, generalize better: closing the generalization gap in large batch training of neural networks *Advances in Neural Information Processing Systems* vol 30
[47] Yeh I-C 2007 Concrete compressive strength *UCI Mach. Learn. Repository* **10** C5K67
[48] Yeh I-C 1998 Modeling of strength of high-performance concrete using artificial neural networks *Cem. Concr. Res.* **28** 1797–808
[49] Chang T-P, Chuang F-C and Lin H-C 1996 A mix proportioning methodology for high-performance concrete *J. Chin. Inst. Eng.* **19** 645–55
[50] Sonday B E, Berry R D, Najm H N and Debusschere B J 2011 Eigenvalues of the Jacobian of a Galerkin-projected uncertain ODE system *SIAM J. Sci. Comput.* **33** 1212–33
[51] Chen R T Q, Rubanova Y, Bettencourt J and Duvenaud D K 2018 Neural ordinary differential equations *Advances in Neural Information Processing Systems* vol 31
[52] Cuturi M 2013 Sinkhorn distances: lightspeed computation of optimal transport *Advances in Neural Information Processing Systems* vol 26
[53] Genevay A, Chizat L, Bach F, Cuturi M and Peyré G 2019 Sample complexity of Sinkhorn divergences *The 22nd Int. Conf. on Artificial Intelligence and Statistics* (PMLR) pp 1574–83
[54] Haihao L and Yang J 2024 PDOT: a practical primal-dual algorithm and a GPU-based solver for optimal transport (arXiv:2407.19689)
[55] Viehmann T 2019 Implementation of batched Sinkhorn iterations for entropy-regularized Wasserstein loss (arXiv:1907.01729)
[56] Xia M, Li X, Shen Q and Chou T 2024 An efficient Wasserstein-distance approach for reconstructing jump-diffusion processes using parameterized neural networks (arXiv:2406.01653)
[57] Xia M, Li X, Shen Q and Chou T 2024 Squared Wasserstein-2 distance for efficient reconstruction of stochastic differential equations (arXiv:2401.11354)
[58] Esfahani P M and Kuhn D 2018 Data-driven distributionally robust optimization using the Wasserstein metric: performance guarantees and tractable reformulations *Math. Program.* **171** 115–66
[59] Hanasusanto G A, Roitch V, Kuhn D and Wiesemann W 2015 A distributionally robust perspective on uncertainty quantification and chance constrained programming *Math. Program.* **151** 35–62
[60] Engquist B, Ren K and Yang Y 2016 The quadratic Wasserstein metric for inverse data matching *Inverse Problems* **32** 115001
[61] Adler J, Ringh A, Öktem O, and Karlsson J 2018 Learning to solve inverse problems using Wasserstein Loss (arXiv:1710.10898)
[62] van Maarschalkerwaart F, Mukherjee S, Sabaté Landman M, Brune C and Carioni M 2025 Perturbation-aware distributionally robust optimization for inverse problems *Int. Conf. on Scale Space and Variational Methods in Computer Vision* (Springer) pp 109–22
[63] Bai X, Guangyi H, Jiang Y and Obloj J 2023 Wasserstein distributional robustness of neural networks *Advances in Neural Information Processing Systems* vol 36 pp 26322–47
[64] Zhang J, Li X, Guo X, You Z, Böttcher L, Mogilner A, Hoffman A, Chou T and Xia M 2025 Reconstructing noisy gene regulation dynamics using extrinsic-noise-driven neural stochastic differential equations (arXiv:2503.09007)
[65] Clement P and Desch W 2008 An elementary proof of the triangle inequality for the Wasserstein metric *Proc. Am. Math. Soc.* **136** 333–9
[66] Li Y, Swersky K and Zemel R 2015 Generative moment matching networks *Int. Conf. on Machine Learning* (PMLR) pp 1718–27
[67] Borawar L and Kaur R 2023 ResNet: solving vanishing gradient in deep networks *Proc. Int. Conf. on Recent Trends in Computing: ICRTC 2022* (Springer) pp 235–47