Journal of Computational Physics 478 (2023) 111972

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A physics-informed diffusion model for high-fidelity flow field reconstruction

Dule Shu [a,1], Zijie Li [a,1], Amir Barati Farimani [a,b,c,*]

[a] Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh PA, USA
[b] Machine Learning Department, Carnegie Mellon University, Pittsburgh PA, USA
[c] Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh PA, USA

ARTICLE INFO

Article history:
Received 2 December 2022
Received in revised form 11 January 2023
Accepted 22 January 2023
Available online 2 February 2023

Keywords:
Denoising diffusion probabilistic models
Computational fluid dynamics
Super-resolution

ABSTRACT

Machine learning models are gaining increasing popularity in the domain of fluid dynamics for their potential to accelerate the production of high-fidelity computational fluid dynamics data. However, many recently proposed machine learning models for high-fidelity data reconstruction require low-fidelity data for model training. Such requirement restrains the application performance of these models, since their data reconstruction accuracy would drop significantly if the low-fidelity input data used in model test has a large deviation from the training data. To overcome this restraint, we propose a diffusion model which only uses high-fidelity data at training. With different configurations, our model is able to reconstruct high-fidelity data from either a regular low-fidelity sample or a randomly measured sample, and is also able to gain an accuracy increase by using physics-informed conditioning information from a known partial differential equation when that is available. Experimental results demonstrate that our model can produce accurate reconstruction results for 2d turbulent flows based on different input sources without retraining.

© 2023 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

Introduction

High-fidelity numerical simulation of fluid dynamics offers valuable information on how engineering systems interact with fluid flows, and is therefore of great interest to engineering design and related fields. Numerical methods for high-fidelity computational fluid dynamics (CFD) simulation, such as the Direct Numerical Simulation (DNS) [1], often require numerically solving the Navier-Stokes equations at a fine scale in both space and time. Such methods have a high computational expense, especially for simulations with high Reynolds numbers. Various fluid modeling techniques have been developed to reduce the computational cost of DNS, including the Reynolds Averaged Navier-Stokes (RANS) modeling [2,3], the Large Eddy Simulations (LES) [4,5], hybrid RANS-LES models [6–11], functional sub-grid models which determine an effective eddy viscosity by wave number and an *a priori* specified mixing length [12–16], and the Explicit LES closure models [17–21]. These techniques increase the computational efficiency of the CFD simulation at the expense of reducing the fidelity of the simulation results. The underlying conflict between computational complexity and simulation fidelity mo-

* Corresponding author.
E-mail address: barati@cmu.edu (A. Barati Farimani).
[1] Contributed equally to this work.

https://doi.org/10.1016/j.jcp.2023.111972
0021-9991/© 2023 The Author(s). Published by Elsevier Inc. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

tivates the study of using machine learning for surrogate modeling to accelerate CFD simulation. Machine learning-based surrogate models cover a wide range of data representations and problem formulations for CFD data, including learned CFD solvers based on Lagrangian representation [22–25] or Eulerian representation [26–35]. In this work, we aim to develop a machine learning model which reconstructs high-fidelity CFD data from low-fidelity input. Since low-fidelity data requires less computational resources to generate, by using a machine learning model to reconstruct high-fidelity data from the numerically-solved low-fidelity one, we can mitigate the conflicts between computational cost and simulation accuracy, and increase the cost-effectiveness of CFD simulations.

Inspired by the various research progress in deep learning for image super-resolution, such as the progressive GAN training approach [36–38], the transformer-based approach [39–41], and the diffusion model-based approach [42–44], several neural network models have been proposed to increase the resolution and reconstruct under-resolved CFD simulations. Pant et al. [45] proposed a CNN-based U-Net model to reconstruct turbulent DNS data from the filtered data and achieved state-of-the-art results in terms of Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Metric (SSIM). With a design using skip-connections and multi-scale filters, Fukami et al. [46,47] proposed a convolutional neural network-based model to reconstruct high-resolution flow field data from low-resolution data in both space and time domains. A multi-scale enhanced super-resolution generative adversarial network with a physics-based loss function was proposed by Yousif et al. [48] to reconstruct high-fidelity turbulent flow with minimal use of training data. To address the case where low and high-resolution flow fields are not matched, Kim et al. [49] proposed a CycleGAN [50]-based model trained with unpaired turbulence data for super-resolution. The effectiveness of generative neural networks in CFD data super-resolution is not limited to the 2d domain but also 3d volumetric flow data [51,52] and particle-based fluids [53,54]. The learned reconstruction networks can also be coupled with a numerical solver to build effective coarse-grained simulators. Kochkov et al. [55] proposed to use a convolutional neural network to correct the error of a simulator that runs on an under-resolved grid. Um et al. [56] generalized the idea of using neural networks to correct the error of numerical PDE solution arising in under-resolved discretization, and incorporated the solver into the training loop by running differentiable solver on the reconstructed results to account for future loss.

The aforementioned deep learning models have yielded promising results in their respective applications and problem settings, yet they share one common limitation: The models are all trained to fit a particular type of under-resolved CFD data as specified by their corresponding training datasets (*e.g.*, a specific filter). As a result, if a trained model is used to reconstruct high-fidelity CFD data from low-fidelity input that significantly deviates from the training dataset (*e.g.*, in terms of resolution or a Gaussian blurring process), the accuracy of data reconstruction will drop significantly. In other words, to ensure the best performance, the users will always have to retrain those models when a new set of under-resolved CFD data is given. The dependency on model retraining has significantly restrained the applicability of deep learning tools in high-fidelity CFD data reconstruction. To resolve this issue, we propose a diffusion-based deep learning framework for CFD data super-resolution. In this framework, we reformulate the problem of reconstructing high-fidelity CFD data from low-fidelity input as a problem of data denoising, and use a Denoising Diffusion Probabilistic Model (DDPM) [57] to reconstruct high-fidelity CFD data from noisy input. Motivated by the advancement in solving fluid mechanics problems using the Physics-informed neural networks (PINNs) [58–60], we propose a procedure to incorporate physics-informed conditioning information in diffusion model training and sampling, which increases the data reconstruction accuracy by making use of the PDE information that determines the fluid flow. Experimental results show that our diffusion model is able to produce comparable results to the state-of-the-art models on the task of high-fidelity CFD data, where it remains accurate in terms of kinetic energy spectrum and PDE residual loss under different input data distribution but without any retraining.

Method

Problem formulation

Let $f_\theta : X \to Y$ be a machine learning model which maps data samples from a low-fidelity data domain X to a high-fidelity data domain Y. By optimizing model parameters θ over a training dataset $\left(X^{(\text{train})}, Y^{(\text{train})}\right)$, we aim to make f_θ achieve a high data reconstruction accuracy on a test dataset $\left(X^{(\text{test})}, Y^{(\text{test})}\right)$. Let $p_X^{(\text{train})}$, $p_X^{(\text{test})}$ denote the data distributions of $X^{(\text{train})}$ and $X^{(\text{test})}$, respectively. Ideally, for a well-trained model to achieve high data reconstruction performance on the test dataset, we need to have $p_X^{(\text{train})} \approx p_X^{(\text{test})}$. In practice, however, such condition is generally not satisfied. If $p_X^{(\text{train})}$ and $p_X^{(\text{test})}$ have a large difference, the data reconstruction accuracy will drop significantly. One idea to mitigate this potential issue is to introduce a preprocessing procedure $g : X \to X$, such that the preprocessed low-fidelity data sampled from $X^{(\text{train})}$ and $X^{(\text{test})}$ have similar distributions (*e.g.*, $p_{g(X)}^{(\text{train})} \approx p_{g(X)}^{(\text{test})}$). One way to increase the similarity between $p_{g(X)}^{(\text{train})}$ and $p_{g(X)}^{(\text{test})}$ is to add Gaussian noise to the low-fidelity data samples, such that both $p_{g(X)}^{(\text{train})}$ and $p_{g(X)}^{(\text{test})}$ are drawn towards a Gaussian distribution from $p_X^{(\text{train})}$ and $p_X^{(\text{test})}$, and subsequently become more similar to each other. Since our goal is to reconstruct high-fidelity CFD data, after introducing Gaussian noise via the preprocessing procedure $g(\cdot)$, we need a denoising procedure to obtain noise-free high-fidelity results. Recent works [61–64] have shown successful examples in using a pretrained DDPM

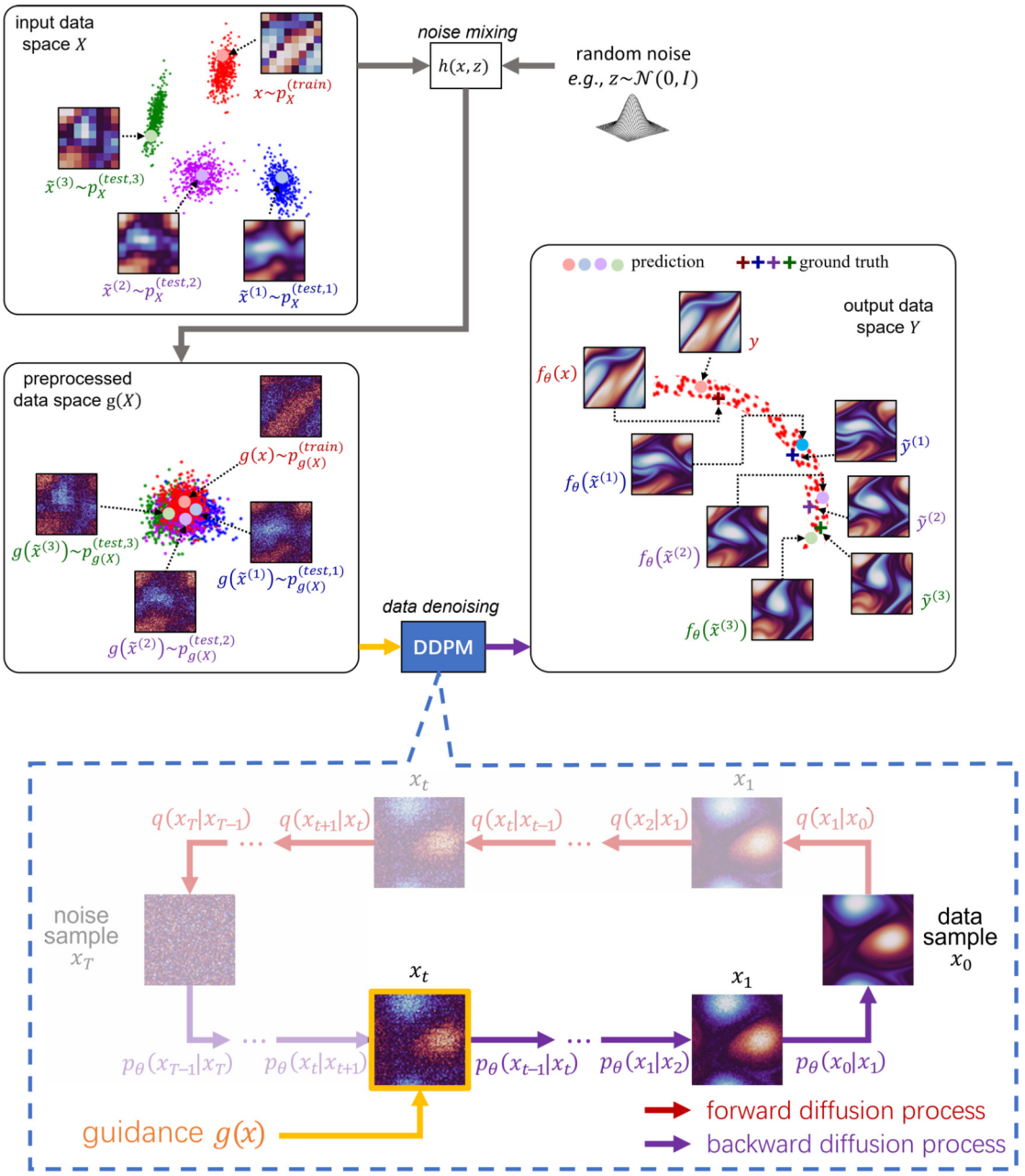


Fig. 1. An overview of our proposed framework for high-fidelity CFD data reconstruction using DDPM model. Given input of low-fidelity CFD data samples $(x, \tilde{x}^{(1)}, \tilde{x}^{(2)}, \tilde{x}^{(3)})$ with various distributions ($p_X^{(\text{train})}$, $p_X^{(\text{test},1)}$, $p_X^{(\text{test},2)}$, $p_X^{(\text{test},3)}$, respectively), we apply a preprocessing procedure $g := h(x,z)$ which mixes the input data x with random noise z, such that the resulting noisy data samples $(g(x), g(\tilde{x}^{(1)}), g(\tilde{x}^{(2)}), g(\tilde{x}^{(3)}))$ have much more similar distributions ($p_{g(X)}^{(\text{train})}$, $p_{g(X)}^{(\text{test},1)}$, $p_{g(X)}^{(\text{test},2)}$, $p_{g(X)}^{(\text{test},3)}$, respectively). We then send the noisy data samples to a pretrained DDPM model for data denoising. The output of the DDPM model is a group of noise-free data samples $(f_\theta(x), f_\theta(\tilde{x}^{(1)}), f_\theta(\tilde{x}^{(2)}), f_\theta(\tilde{x}^{(3)}))$ which are close reconstruction of the corresponding ground truth samples $(y, \tilde{y}^{(1)}, \tilde{y}^{(2)}, \tilde{y}^{(3)})$ of high-fidelity. Inside the DDPM-based data denoising module, we implement a partial backward diffusion process, which is a Markov process starting from a conditioning sample $x_t = g(x)$ and ends at a denoised data sample $x_0 = f_\theta(x)$, as shown in the dotted blue box. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

model for guided image synthesis and editing. These examples demonstrate the potential of DDPM model in eliminating Gaussian noise from CFD data, and motivate us to use it for our denoising task. An overview of our high-fidelity CFD data reconstruction framework with DDPM as the denoising module is shown in Fig. 1. More details of the DDPM model and how it is used for CFD data reconstruction are provided in the following subsections.

Denoising diffusion probabilistic model

A DDPM model is a generative model that generates data of interest using a Markov chain starting from a sample of standard Gaussian distribution. Let x_0 be the data sample to generate, and let θ be a set of neural network parameters of the DDPM model, the Markov chain can be represented as follows.

$$p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t) \tag{1}$$

where $p(x_T) := \mathcal{N}(x_T; \mathbf{0}, \mathbf{I})$ and the probability transition $p_\theta(x_{t-1}|x_t)$ is chosen as $p_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$. In diffusion model, such a process to convert a data sample to a random noise sample (*e.g.*, a sample from the standard Gaussian distribution) is referred to as the *forward process* or the *forward diffusion process*. If a neural network model can be used to estimate the inverse of the forward process (which is referred to as the *reverse process* or the *backward diffusion process*), then it can be used to generate data from noise. Formally, starting from the data sample x_0, a forward process of length T can be constructed as follows.

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1}), \quad q(x_t|x_{t-1}) := \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t \mathbf{I}\right)$$

where $\beta_1, ..., \beta_T$ are a sequence of scaling factors to scale the variance of noise added to each step of the forward process. In order to generate an authentic data sample x_0, the backward diffusion process implemented by the DDPM model needs to be trained to maximize the probability distribution $p_\theta(x_0)$, or equivalently to minimize the negative log-likelihood $-\log p_\theta(x_0)$. Ho et al. [57] showed that the negative log-likelihood term can be upper-bounded by the variational lower bound [65], from which the following model training objective can be derived by reparameterizing the Gaussian probability density function and ignoring the weighting terms containing β_t's and Σ_θ.

$$L_t^{\text{simple}} = \mathbb{E}_{t\sim[1,T],x_0,\epsilon}\left[\|\epsilon_t - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t\right)\|^2\right] \tag{2}$$

where $\alpha_t := 1 - \beta_t$, $\bar{\alpha}_t := \prod_{i=1}^{t} \alpha_i$, $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is the standard Gaussian noise sampled at time t, and $\epsilon_\theta(\cdot)$ denotes the prediction of the DDPM neural network model given $\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t$ and t as inputs. An illustration of the model training procedure is shown in the upper subplot of Fig. 2.

Guided data synthesis with DDPM

The original DDPM sampling algorithm [57] generates data samples via a Markov chain starting from $x_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. However, the randomness of x_T makes it difficult to control the data generation process and therefore does not satisfy the need for high-fidelity CFD data reconstruction from a low-fidelity reference. To address this issue, a guided data generation procedure is needed where the low-fidelity CFD data is used as the condition to generate the high-fidelity one. As suggested by Meng et al. [61], the Markovian property of the backward diffusion process means that the process to generate x_0 does not have to start from x_T but can start from any time-step $t \in \{1, ..., T\}$ provided that x_t is available. This property allows users to select an intermediate sample at a particular time-step of the backward diffusion process, and send it to the remaining part of the Markov chain to obtain x_0. By controlling the signal component of the intermediate sample, the user can control the content of x_0. More specifically, let x_0^{forward} denote the initial data sample at the beginning of the forward diffusion process, then the intermediate sample x_t at an arbitrary time-step t can always be calculated as follows.

$$x_t = \sqrt{\bar{\alpha}_t}x_0^{\text{forward}} + \sqrt{1-\bar{\alpha}_t}\epsilon_t \tag{3}$$

Equation (3) indicates that x_t is a weighted combination of a signal component x_0^{forward} and a random noise component ϵ_t. Hence, for a guided data synthesis, one can assign a guidance data point $x^{(g)}$ to the signal component x_0^{forward}. The resulting Markov chain to generate a data sample x_0 conditioned on a reference $x^{(g)}$ is shown as follows.

$$p_\theta(x_{0:t}) := p(x_t) \prod_{i=1}^{t} p_\theta(x_{i-1}|x_i) \quad \text{where} \quad x_t := \sqrt{\bar{\alpha}_t}x^{(g)} + \sqrt{1-\bar{\alpha}_t}\epsilon_t \tag{4}$$

Note that the intermediate denoising result x_t obtained from Eq. (3) and Eq. (4) are different since $x_0^{\text{forward}} \neq x_0^{(g)}$. During model training, DDPM learns to generate a data sample x_0 using a Markov process starting from the x_t in Eq. (3), such that x_0 approximates x_0^{forward} (In the context of CFD data reconstruction, x_0^{forward} is a high-fidelity CFD data sample from a training dataset, and x_0 is a reconstruction of x_0^{forward} by DDPM). Nevertheless, when applying a trained DDPM model for conditional sampling, x_0 is generated using a Markov process starting from the x_t in Eq. (4), given a guidance data point $x^{(g)}$. On one

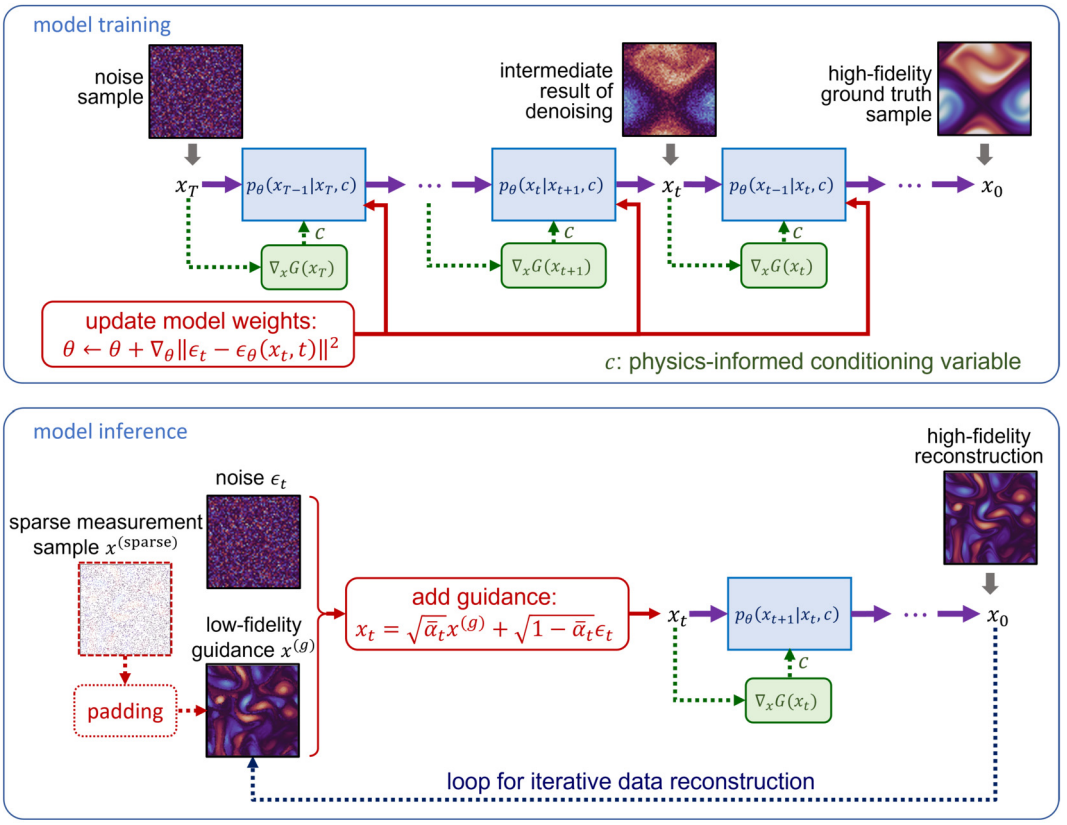


Fig. 2. The procedures of model training (upper subplot) using Algorithm 1 and model inference (lower subplot) using Algorithms 2. Each blue box represents a step in the backward diffusion process computed by a neural network model with parameter θ. Blocks and arrows with dotted lines (*e.g.*, randomly measured samples and physics-informed conditioning) represents optional steps in the framework.

hand, we choose to use $x^{(g)}$ in data generation despite its difference from x_0^{forward}, because $x^{(g)}$ contains the conditioning information (*e.g.*, a low-fidelity CFD data sample) which controls the backward diffusion process; on the other hand, by mixing the signal component (x_0^{forward} or $x_0^{(g)}$) with random noise component ϵ_t, we try to bring $\sqrt{\bar{\alpha}_t}x^{(g)} + \sqrt{1-\bar{\alpha}_t}\epsilon_t$ closer to $\sqrt{\bar{\alpha}_t}x^{\text{forward}} + \sqrt{1-\bar{\alpha}_t}\epsilon_t$ in the statistical sense (*e.g.*, the distributions of the two quantities are both closer towards Gaussian), such that the disturbance to the quality of conditional sampling due to the discrepancy between x_0^{forward} and $x^{(g)}$ will be mitigated.

Improved procedures for DDPM-based conditional sampling

A related problem to the reconstruction of high-fidelity CFD data from low-fidelity input is the problem of reconstructing a high-fidelity data sample from an incomplete and randomly measured data sample, as shown in the lower subplot of Fig. 2. Unmeasured components from random locations in a data sample are quite challenging to data reconstruction since they reduce the amount of information to use. One method to address this challenge is to add an iteration loop to the data reconstruction procedure. Given a sparse data sample measured at random locations, $x^{(\text{sparse})}$, we first apply nearest-neighbor padding to fill the unmeasured data space, and then send the padded data sample to an iteration of DDPM-based conditional sampling procedure, where the reconstructed sample from the previous iteration is used as a low-fidelity reference to guide the next iteration of conditional sampling. This iteration can be repeated until the reconstruction quality has improved sufficiently. In general, the necessary number of iterations increases as the difference between x_0^{forward} and $x^{(g)}$ becomes larger.

The iterative sampling procedure above is one modification to the basic diffusion-based method made to address the special case of reconstructing high-fidelity CFD data from randomly measured data samples. Additionally, we have considered a second special case where the analytical form of the Partial Differential Equation (PDE) determining the CFD data is known. To improve the backward diffusion process with the knowledge of the PDE, we propose another modification to the basic data sampling procedure using DDPM model. Let the following equation be the known PDE operator that determines a fluid flow simulation from which the ground truth CFD data is collected,

$$G\left(u, \frac{\partial u}{\partial \xi_1}, \cdots, \frac{\partial u}{\partial \xi_i}, \cdots, \frac{\partial^2 u}{\partial \xi_i \xi_j}, \cdots; \Theta\right) = 0, \quad \xi = (\xi_1, \xi_2, \cdots) \in \Omega, \tag{5}$$

where G denotes the differential operator for the corresponding PDE, $u(\xi)$ is the solution to the PDE, Θ denotes the parameters in the PDE (e.g. viscosity, constants in forcing term), and Ω denotes the computational domain on which the PDE is defined. If we substitute $u = x_t$ (where x_t the intermediate step of the backward diffusion process) into the left-hand-side of Eq. (5), due to model prediction error and truncation error on the discretization grid, we will in general obtain a non-zero quantity on the right-hand-side of Eq. (5), *e.g.*, $G|_{u=x_t} = r_t,\ r_t \neq 0$. The non-zero term r_t is usually referred to as the residual of the PDE and can be used to evaluate the accuracy of a numerical solution to the PDE. For conditional data generation, Ho et al. [66] proposed a classifier-free diffusion model that generates data samples from a conditional distribution $p_\theta(x_t|c)$ where c is the class label of the data sample. Inspired by this work, we propose to use the gradient of the PDE residual as the conditioning information, which yields the following data sampling process.

$$p_\theta(x_{0:t}) := p(x_t)\prod_{i=1}^{t} p_\theta(x_{i-1}|x_i, c) \quad \text{where} \quad x_t = \sqrt{\bar{\alpha}_t}x^{(g)} + \sqrt{1-\bar{\alpha}_t}\epsilon_t, \quad c = \frac{\partial r_t}{\partial x} \tag{6}$$

We refer to the conditioning information c in Eq. (6) as the physics-informed condition, as it is obtained from the PDE which governs the physical process of the fluid flow. An overview of the model training and inference procedures for the physics-informed CFD data reconstruction is shown in Fig. 2. Two methods are proposed in this work to implement the physics-informed data generation process shown in Eq. (6): 1. Data sampling with learned encoding of physics-informed condition. 2. Data sampling with direct gradient descent of physics-informed condition. In the first method, we modified the architecture of the original DDPM model by adding an extra module which encodes the PDE residual gradient $c = \partial r_t/\partial x_t$, so that the model can be used to sample $x_{i-1},\ i \in \{1, ..., t\}$ from $p_\theta(x_{i-1}|x_i, c)$ rather than $p_\theta(x_{i-1}|x_i)$. The modified model is trained using the following Algorithm 1.

Algorithm 1 Physics-informed DDPM Model Training.

Require: p_u (probability of unconditional training), $G = 0$ (PDE that determines the CFD data)

1: **repeat**
2: $x_0 \sim q(x_0)$
3: $t \sim \text{Uniform}(\{1, ..., T\})$
4: $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: $r_t = G|_{x=x_t}$ ▷ Compute the residual of the PDE.
6: $c = \begin{cases} \partial r_t/\partial x_t, & \text{with probability } 1 - p_u \\ \emptyset, & \text{with probability } p_u \end{cases}$ ▷ Randomly discard conditioning information to train unconditionally.
7: Take a step of gradient descent on θ with the following gradient.
$\nabla_\theta \| \epsilon_t - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon_t, t, c\right) \|^2$
8: **until** converged

Once the model is trained with Algorithm 1, it can be used in the data generation procedure specified by the following Algorithm 2.

Algorithm 2 Physics-informed Conditional Sampling with DDPM.

Require: $x^{(g)}$ (guide), t (time-step location of $x^{(g)}$ in the backward diffusion process, $t < T$), $\tau = \{\tau_0, \tau_1, ..., \tau_S\}$ (an increasing subsequence of $[0, 1, ..., t]$ where $\tau_0 = 0$ and $\tau_S = t$), ϵ_θ (a pretrained DDPM model), $G = 0$ (PDE that determines the CFD data), w (conditioning strength)

1: **for** $k = 1, 2, ..., K$ **do**
2: $\epsilon_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
3: $x_t = \sqrt{\bar{\alpha}_t}x^{(g)} + \sqrt{1-\bar{\alpha}_t}\epsilon_t$
4: **for** $i = S, S-1, ..., 1$ **do**
5: $z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $i > 1$ else $z = 0$
6: $r_{\tau_i} = G|_{u=x_{\tau_i}}$ ▷ Compute the residual of the PDE.
7: $c = \partial r_{\tau_i}/\partial x_{\tau_i}$
8: $\tilde{\epsilon}_\theta = \epsilon_\theta(x_{\tau_i}, \tau_i, c) + w\left[\epsilon_\theta(x_{\tau_i}, \tau_i, c) - \epsilon_\theta(x_{\tau_i}, \tau_i, \emptyset)\right]$
9: $x_{\tau_{i-1}} = \sqrt{\frac{\bar{\alpha}_{\tau_{i-1}}}{\bar{\alpha}_{\tau_i}}}\left(x_{\tau_i} - \sqrt{1-\bar{\alpha}_{\tau_i}}\cdot\tilde{\epsilon}_\theta\right) + \sqrt{1-\bar{\alpha}_{\tau_{i-1}} - \sigma_{\tau_i}^2}\cdot\tilde{\epsilon}_\theta + \sigma_{\tau_i}^2\epsilon_{\tau_i}$
10: **end for**
11: $x^{(g)} = x_0$ ▷ $x_{\tau_0} = x_0$ since $\tau_0 := 0$.
12: **end for**
13: **return** x_0

In the context of high-fidelity CFD data reconstruction, $x^{(g)}$ represents a sample of low-fidelity CFD data, x_0 represents the corresponding sample of high-fidelity CFD data, the DDPM model ϵ_θ is pretrained on a high-fidelity CFD dataset, and t is a main hyper-parameter for conditional data sampling using Algorithm 2. In practice, t is selected from the interval $\left[0, \frac{T}{2}\right]$ for a more accurate and less noisy data reconstruction. Instead of following the data sampling procedure from the original DDPM model, we designed Algorithm 2 based on the accelerated sampling procedure proposed by Song et al. [67], named the Denoising Diffusion Implicit Model (DDIM). We also adopted the same design choice $\sigma_{\tau_i} = 0\ \forall\ i$ as suggested in the original DDIM model. The intuition behind Line 8 in Algorithm 2 is briefly discussed as follows. Ho et al. pointed out in

their work [57] that the data sampling process presented in the original DDPM resembles the Langevin dynamics with ϵ_θ as a learned gradient of the data density, referred to as the score function [68]. The resemblance between ϵ_θ and the score function enables the following approximation.

$$\epsilon_\theta\left(x_i, i, \emptyset\right) \approx -\sigma_i \nabla_{x_i} \log p_\theta\left(x_i\right), \quad \epsilon_\theta\left(x_i, i, c\right) \approx -\sigma_i \nabla_{x_i} \log p_\theta\left(x_i, c\right) \tag{7}$$

Eq. (7) indicates that the neural network model ϵ_θ in the DDPM and DDIM models can be considered as an estimator of the score function. Substitute Eq. (7) into Line 8 of Algorithm 2, we have

$$\begin{aligned}\tilde{\epsilon}_\theta &= \epsilon_\theta\left(x_i, i, c\right) + w\left[\epsilon_\theta\left(x_i, i, c\right) - \epsilon_\theta\left(x_i, i, \emptyset\right)\right] \\ &\approx -\sigma_i \nabla_{x_i} \log p_\theta\left(x_i, c\right) - w\left[\sigma_i \nabla_{x_i} \log p_\theta\left(x_i, c\right) - \sigma_i \nabla_{x_i} \log p_\theta\left(x_i\right)\right] \\ &= -\sigma_i \nabla_{x_i} \log p_\theta\left(x_i, c\right) - w\sigma_i \nabla_{x_i} \log p_\theta\left(c|x_i\right)\end{aligned} \tag{8}$$

Equation (8) shows that Algorithm 2 is designed to sample x_{i-1} using a weighted combination of the data prediction $p_\theta\left(x_i, c\right)$ and the PDE residual gradient prediction $p_\theta\left(c|x_i\right)$, where the weight is determined by the conditioning strength w. In our second method to implement the physics-informed data generation process, data sampling with direct gradient descent of physics-informed condition, we do not modify the original DDPM model architecture or training procedure to accommodate PDE residual gradient c in model input. Instead, we directly incorporate gradient descent in the conditional sampling process of DDIM model. The intuition for this method is as follows. Consider the PDE in Eq. (5) which models the fluid flow. Any ground truth CFD data sample should satisfy this PDE and produce a zero residual. Therefore, a valid goal of high-fidelity CFD data reconstruction is to optimize the DDPM model preditction x_0, such that the corresponding residual $r_0 := G|_{u=x_0}$ is minimized. A gradient-descent method to achieve this goal is to search for the high-fidelity CFD data using the following update rule, $x_{i-1} = x_i - \lambda \partial r_i / \partial x_i$, where λ represents the step size of gradient descent. For an improved performance of high-fidelity CFD data reconstruction, we combined the original goal of DDPM-based data sampling (which is to minimize the KL-divergence between the forward and the backward diffusion processes for an authentic data generation) with the goal of minimizing the residual of CFD data reconstruction. More specifically, to implement our data sampling method with direct gradient descent of physics-informed condition, we modified Line 8 of Algorithm 2 as follows.

$$x_{\tau_{i-1}} = \sqrt{\frac{\bar{\alpha}_{\tau_{i-1}}}{\bar{\alpha}_{\tau_i}}}\left(x_{\tau_i} - \sqrt{1-\bar{\alpha}_{\tau_i}} \cdot \tilde{\epsilon}_\theta\right) + \sqrt{1-\bar{\alpha}_{\tau_{i-1}} - \sigma_{\tau_i}^2} \cdot \tilde{\epsilon}_\theta - \lambda c + \sigma_{\tau_i}^2 \epsilon_{\tau_i} \tag{9}$$

The sampling procedure introduced in (9) is essentially a linear combination between score function and the negative gradient of PDE residual (which we denote as "Linear" in the Experiment section), whereas Algorithm 2 is a learned combination of two components (which we denote as "Learned" in the Experiment section).

Experiments

Implementation

The dataset we considered is the 2-dimensional Kolmogorov flow [69], governed by the Navier-Stokes equation for incompressible flow. The vorticity form of it reads as,

$$\begin{aligned}\frac{\partial \omega(\mathbf{x}, t)}{\partial t} + \mathbf{u}(\mathbf{x}, t) \cdot \nabla \omega(\mathbf{x}, t) &= \frac{1}{Re} \nabla^2 \omega(\mathbf{x}, t) + f(\mathbf{x}), \quad \mathbf{x} \in (0, 2\pi)^2, t \in (0, T], \\ \nabla \cdot \mathbf{u}(\mathbf{x}, t) &= 0, \quad \mathbf{x} \in (0, 2\pi)^2, t \in (0, T], \\ \omega(\mathbf{x}, 0) &= \omega_0(\mathbf{x}), \quad \mathbf{x} \in (0, 2\pi)^2,\end{aligned} \tag{10}$$

where ω is the vorticity, $\mathbf{u}$ represents the velocity field, Re denotes the Reynolds number which is set to 1000 in this problem, $f(\mathbf{x})$ is the forcing term, and $\mathbf{x} = [x_1, x_2]$. The boundary condition considered is periodic boundary condition. The forcing term for 2-d Kolmogorov flow studied here is defined as: $f(\mathbf{x}) = -4\cos(4x_2) - 0.1\omega(\mathbf{x}, t)$, which contains an additional drag force term $0.1\omega(\mathbf{x}, t)$ similar to Kochkov et al. [70], in order to prevent energy accumulation at large scales [71]. We numerically solve Equation (10) using a pseudo-spectral solver implemented in PyTorch [72] from Li et al. [73], which samples initial condition $\omega_0(\mathbf{x})$ from a Gaussian random field $\mathcal{N}(0, 7^{3/2}(-\Delta + 49I)^{-5/2})$. The discretization grid used is a 2048×2048 uniform grid. The data derived from the direct numerical simulation are considered the high-fidelity data we aim to reconstruct using the machine learning model, and are referred to as ground truth in the dataset.

We simulate 40 sequences in total, each with a temporal length of 10 seconds ($T = 10$). We then downsample these data to a 256×256 grid spatially with a fixed time interval $\Delta t = 1/32s$, resulting in a dataset comprising 40 sequences each with 320 frames. Among them, we use the first 36 sequences as the training set and the rest 4 for testing. As our model operates on the same input and output grid, we use nearest (based on Euclidean norm) interpolation to process the input such that it has the same resolution as the target.

For the calculation of the PDE's residual, we use discrete Fourier transform to calculate spatial derivatives and finite difference to calculate time derivatives. Given that the proposed diffusion model operates on the vorticity of three consecutive frames $[\omega_{t-1}(\mathbf{x}), \omega_t(\mathbf{x}), \omega_{t+1}(\mathbf{x})]$, we can approximate $\partial_t \omega$ via $\partial_t \omega \approx (\omega_{t+1}(\mathbf{x}) - \omega_{t-1}(\mathbf{x}))/(2\Delta t)$. For the convection term and diffusion term, we approximate them by calculating the Laplacian and gradient of vorticity in the Fourier space, derive velocity using the stream function ψ: $\mathbf{u} = \nabla \times \psi, -\nabla^2 \psi = \omega$, and then convert them back to the physical space via inverse Fourier transform.

The proposed denoising diffusion model is parameterized by a UNet [74], which has empirically been shown effective for estimating the score function [57,68] (or equivalently the noise in DDPM). UNet is a neural network architecture with hierarchical convolution blocks and multi-level skip connections, which allows it to better capture the dependency of different ranges (resembles multigrid method). In addition, we use self-attention [75] in the bottleneck layer (corresponds to the coarsest resolution) to further increase the receptive field. Based on the UNet architecture, we investigate three types of training strategies as below.

- Diffusion model with learned residual condition. The residual information is directly provided to the diffusion model as input features, i.e. $\epsilon_\theta \left[x_t, \nabla_{x_t} G(x_t)\right]$, where $G(\cdot)$ is the corresponding differential operator of the Navier-Stokes equation. In this way, the final guidance is the learned combination between residual gradient and score function.
- Original diffusion model. In this case, the diffusion process will not take residual information into account.
- A UNet that learns a fixed mapping: $f : X \mapsto Y$ where X is the low fidelity data domain and Y is the high fidelity data domain. This is the setup which most deep-learning-based flow reconstruction methods have adopted [45–48]. This setup requires paired training data and the learned mapping is usually locked to a specific distribution in X.

We adopt a recursive refinement strategy for which we apply $K = 3$ (see Algorithm 2 for detailed definition of K) times of backward diffusion process recursively for $4\times$ upsampling and 5% points' sparse reconstruction tasks, and we set $S = 160, 114, 80$ for $k = 1, 2, 3$ respectively. However, for input data that is farther away from the target data distribution, we observe that a single diffusion chain with larger injected noise (i.e. larger t in line 3 of Algorithm 2) is often more beneficial. Therefore, for $8\times$ upsampling task we use $S = 320$ and set $K = 1$, and $S = 400, K = 1$ for 1.5625% reconstruction task.

Results

We conduct experiments on the following tasks to investigate the capability of the diffusion model on reconstructing high fidelity flow field. The task in the first experiment is reconstructing high-resolution field from low-resolution field, where the low-resolution field is uniformly downsampled from the high-resolution one. The task in the second experiment is to reconstruct high-resolution field from randomly sampled collocation points (not necessarily equidistant). The second task aims to reconstruct dense field from sparse sensory observation data. For the first task, we test our reconstruction model on two levels of input resolution, $64 \times 64 \mapsto 256 \times 256$ ($4\times$ upsampling) and $32 \times 32 \mapsto 256 \times 256$ ($8\times$ upsampling). For the second task, we also test our reconstruction model on two different levels of sparsity, where we sampled 5% and 1.5625% (same amount of grid points as 32×32 grid). The collocation points is randomly sampled with uniform probability. Note that for all the experiments (except the ablation for different conditioning methods), we use the same diffusion model (with learned guidance), which means we do not retrain the model for a specific task. To reduce the aliasing effect when applying direct mapping model to out-of-distribution data, we applied Gaussian smoothing kernel with $\sigma = 5$ to data used in $32 \times 32 \mapsto 256 \times 256$ and $1.5625\% \mapsto 256 \times 256$ tasks.

The visualization of our model's reconstruction results are shown in the Figs. 3, 4, 5, 6. We can observe that both bicubic interpolation and diffusion model generate satisfactory results for the easier task of $4\times$ upsampling (Fig. 3), but the diffusion model can recover more details for the more challenging scenarios (Fig. 4, 5, 6). This qualitatively demonstrates the capability of using diffusion model to reconstruct high-resolution flow field given inputs with different distributions.

To quantitatively evaluate the reconstruction results, we use L_2 norm to measure the pointwise error between prediction and ground truth on each grid point (with n representing the total number of grid points per sample):

$$D_{L_2}(\hat{\omega}, \omega) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{\omega}_i - \omega_i)^2}. \tag{11}$$

We also evaluate the normalized residual of each predicted result that comprise three consecutive frames. This measures if the results are aligned with the underlying governing equation.

$$D_{\text{equation}}(\hat{\omega}, \omega) = \frac{1}{n}\frac{\sum_{i=1}^{n}|G(\hat{\omega}_i)|^2}{||\omega||_2^2}, \tag{12}$$

where $G(\cdot)$ is the differential operator as described in Equation (10), $||\omega||_2^2$ is the average squared L_2 norm of the vorticity. In addition to the pointwise metric, we compare the statistics between the reconstructed results and the ground truth.

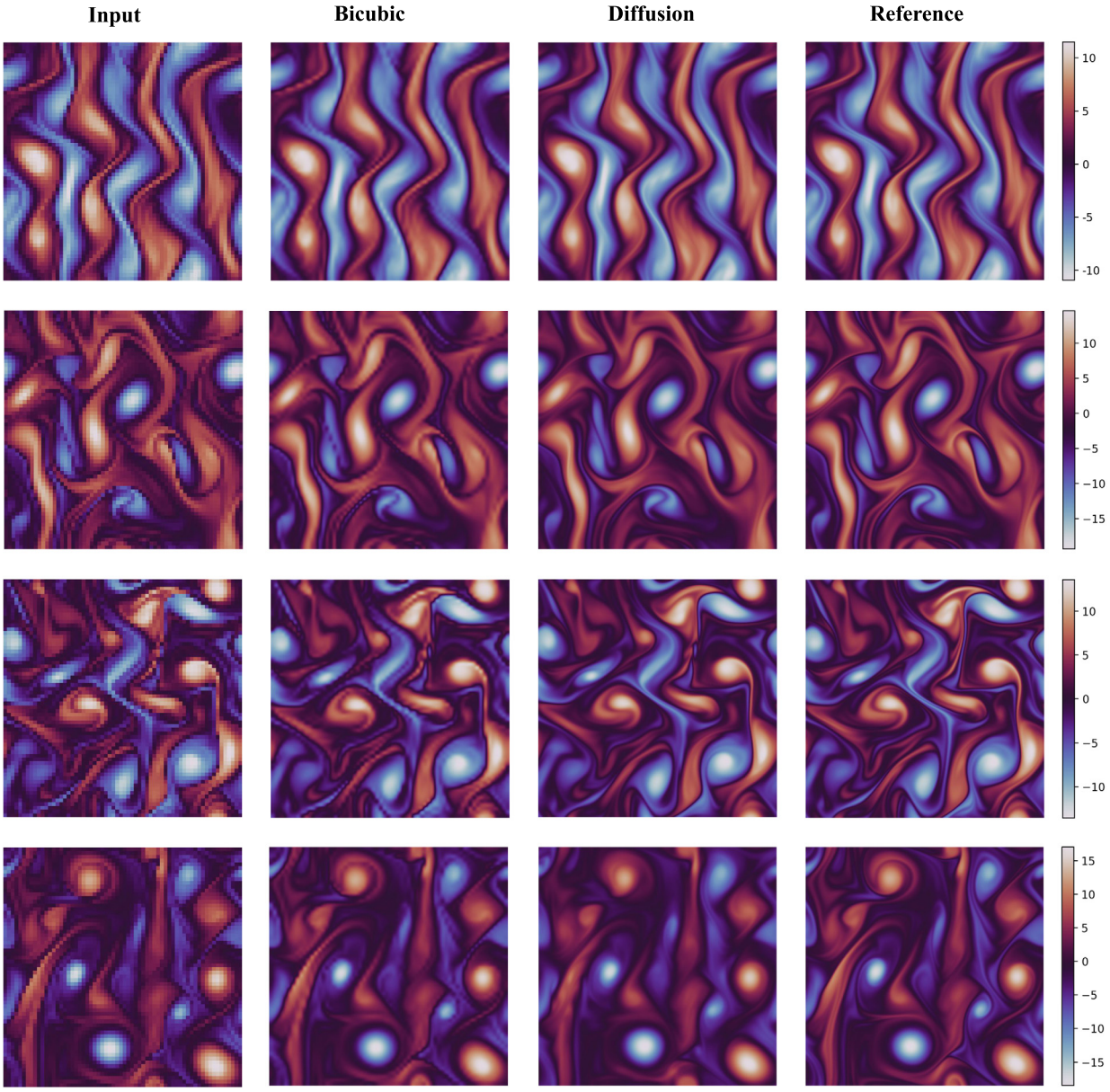


Fig. 3. Qualitative comparison of different upsampling methods on 4x upsampling task.

More specifically, we inspect the energy spectrum and vorticity distribution to validate the alignment between prediction and ground truth distribution.

The pointwise error of different models' reconstruction results are shown in the Table 1. Both the diffusion model and the learned direct mapping model outperform bicubic interpolation by a margin, which indicates the strength of data-driven learning-based methods. In terms of L_2 loss, direct mapping model has similar performance as diffusion model, yet the margin enlarges when applying the direct mapping model to a new data distribution (mapping from observation on 5% of the grid points to full resolution grid). Both the L_2 loss and the equation loss increase significantly for the direct mapping model, while the diffusion model shows more robustness. As shown in Fig. 8a, the learned direct mapping model's prediction is much blurrier when applied to out-of-distribution data. This signifies the difference between the two learning paradigms. Direct mapping model is sensitive with respect to the input distribution, while in diffusion model, the input is perturbed with noises and thus the perturbed input distribution still intersects with the distribution of which the diffusion model can effectively estimate the score function. Furthermore, with physics-informed condition, the PDE residual of diffusion model's predicted results is lower than other models that do not take PDE information into account.

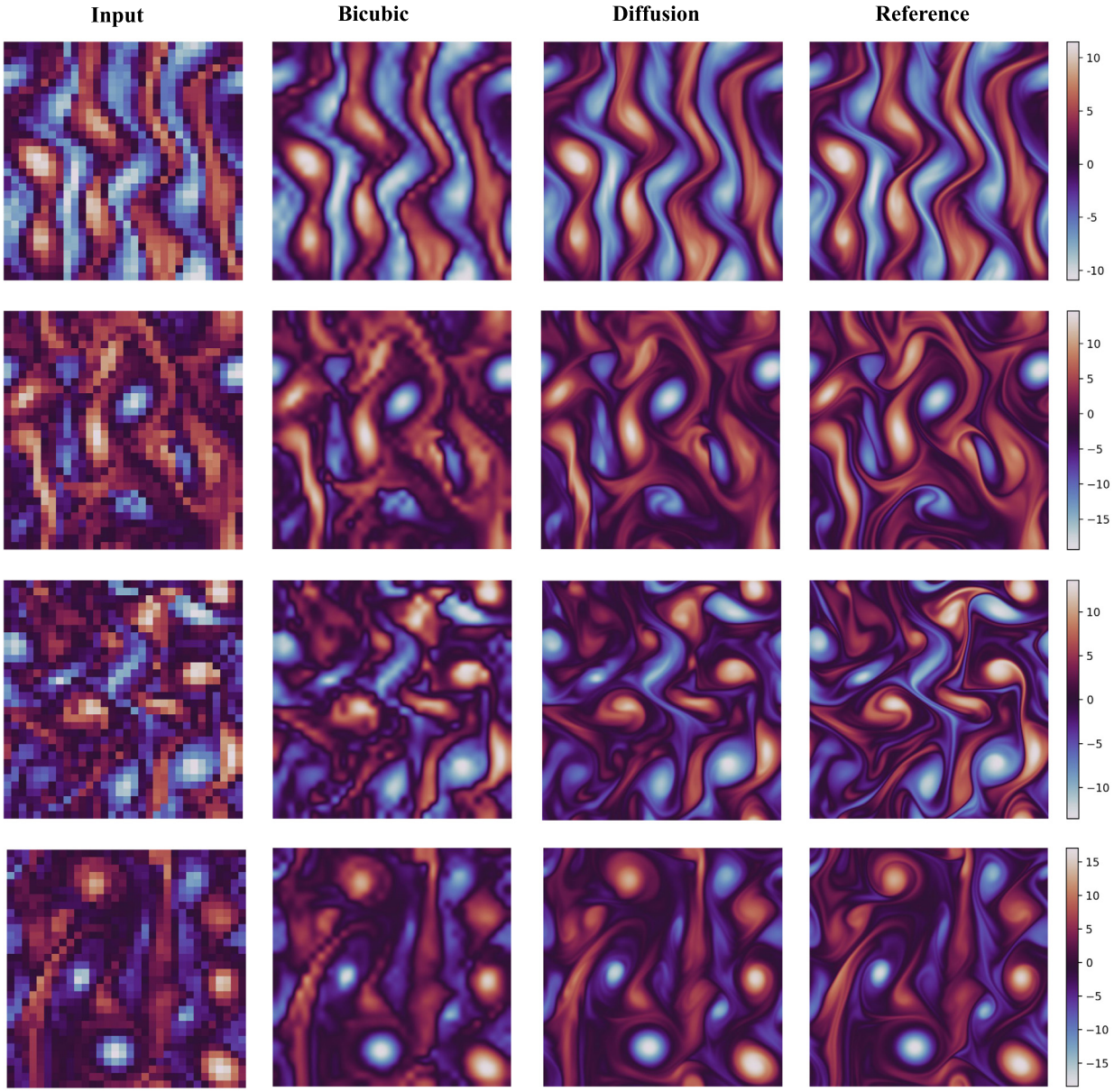


Fig. 4. Qualitative comparison of different upsampling methods on 8x upsampling task.

Table 1
Quantitative comparison of diffusion model and other interpolation/reconstruction methods. The "Direct map" model is trained on $64 \times 64 \mapsto 256 \times 256$ data, data in other tasks are considered out-of-distribution data for it.

Task	L_2 norm			Equation loss		
	Diffusion	Direct map	Bicubic	Diffusion	Direct map	Bicubic
$64 \times 64 \mapsto 256 \times 256$	0.5622	0.6048	1.3355	0.2178	0.5438	10.6639
$32 \times 32 \mapsto 256 \times 256$	1.3035	1.3700	2.5179	0.2039	9.9291	20.0008
$5\% \mapsto 256 \times 256$	0.9318	1.2426	-	0.2815	20.2853	-
$1.5625\% \mapsto 256 \times 256$	1.8213	1.9149	-	0.2290	17.5249	-

Next we compare the vorticity distribution between the predicted data and the ground truth. Fig. 7b shows that the results of the diffusion model have a sharper distribution than both the ground truth and bicubic interpolated data. This is likely caused by the noise estimation error of the diffusion model. Before the backward diffusion process starts, the input data is mixed with Gaussian noise, then the diffusion model gradually removes the noise and steps towards the

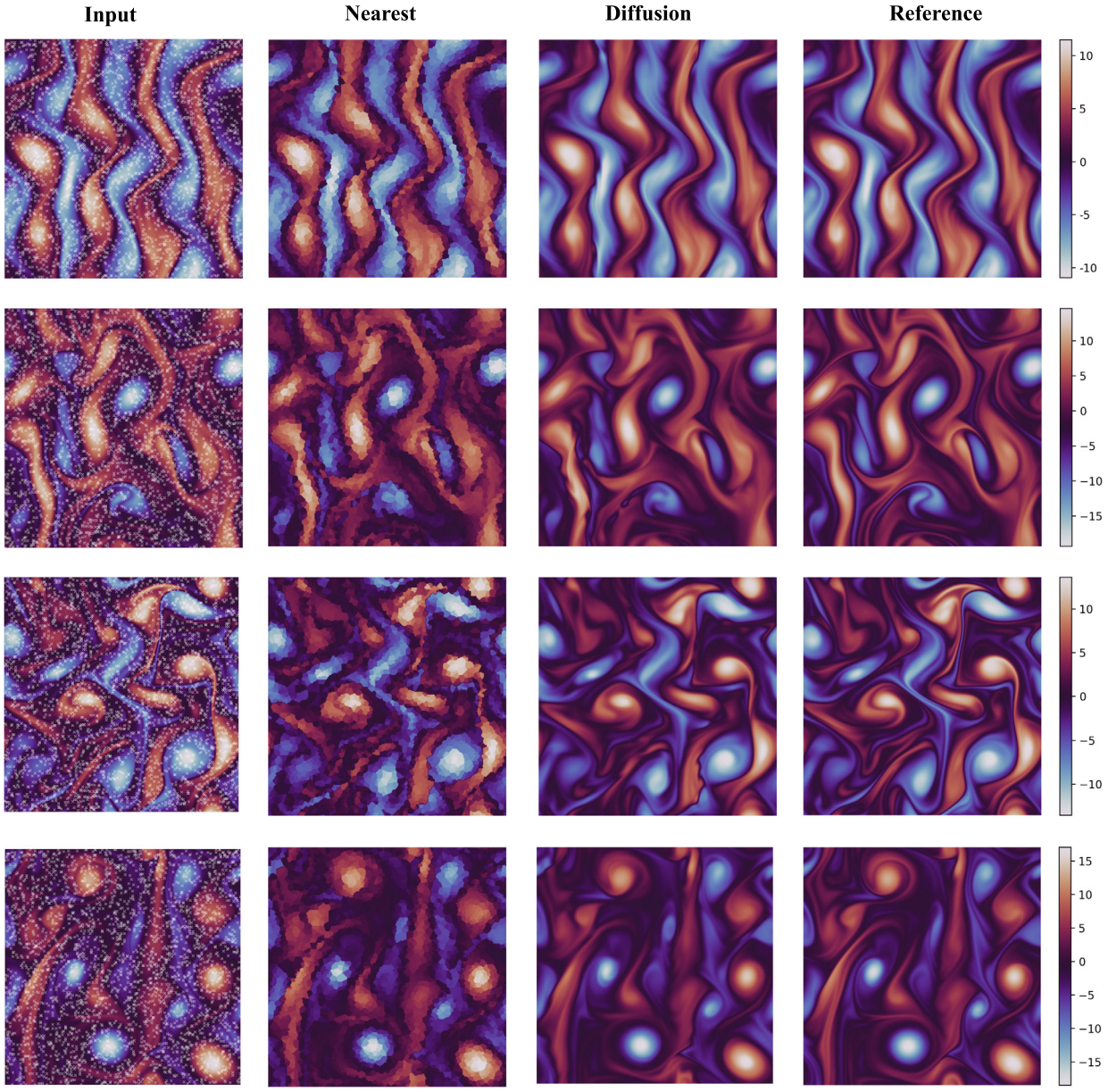


Fig. 5. Qualitative comparison of different upsampling methods on non-equidistant sparse reconstruction task using 5% of grid points. White cross in the input indicates the collocation points (zoom in for clarity).

target distribution. However, as the noise estimation from diffusion model always contains error to some degrees, the final result is not guaranteed to be perfectly noise-free and exhibits a slight distribution drift. This is more obvious for upsampling case with higher sparsity, where we inject the input data with larger noise and diffuse for more steps. For the energy spectrum, compared to bicubic interpolation, we observe that diffusion model can better capture the trend in high wavenumber regime and exhibits less truncation effect. As for the direct mapping model, it has reasonable accuracy on the data it was trained on. However, when extrapolated to the out-of-distribution data, its performance degrades and produces results that has much higher residual. As shown in Fig. 8a, its prediction becomes non-smooth and less physical coherent.

PDE's residual calculation is sensitive with respect to noise and aliasing (due to nearest padding). To accommodate for this, for task with high sparsity (1.5625%), we apply Gaussian filter to smooth the field before inputting them to the model. From Table 2 and Fig. 8b, we observe that while smoothing can alleviate direct mapping model's degradation, it still struggles to predict physically coherent result compared to diffusion model. Furthermore, despite a smoothing process

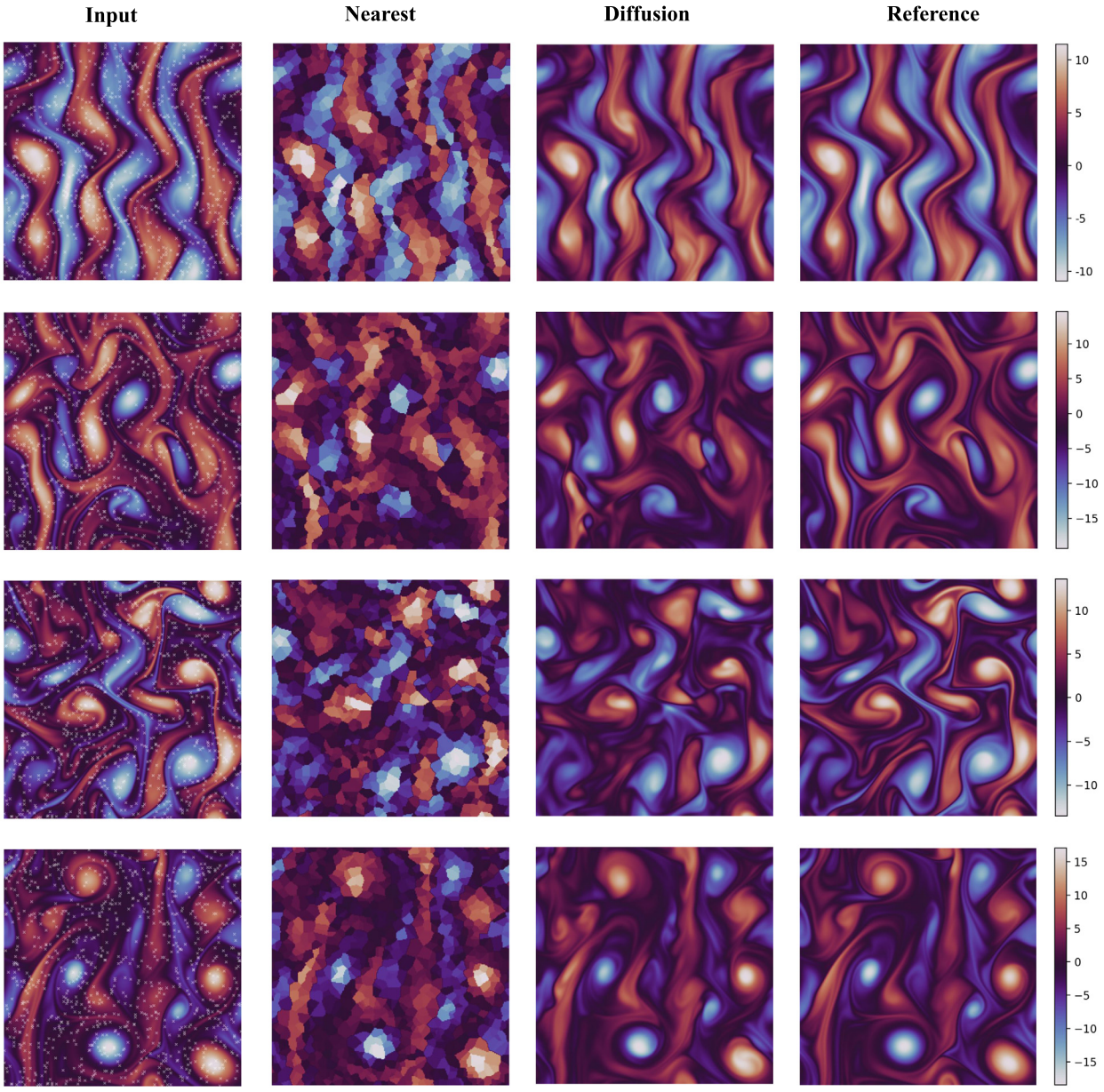
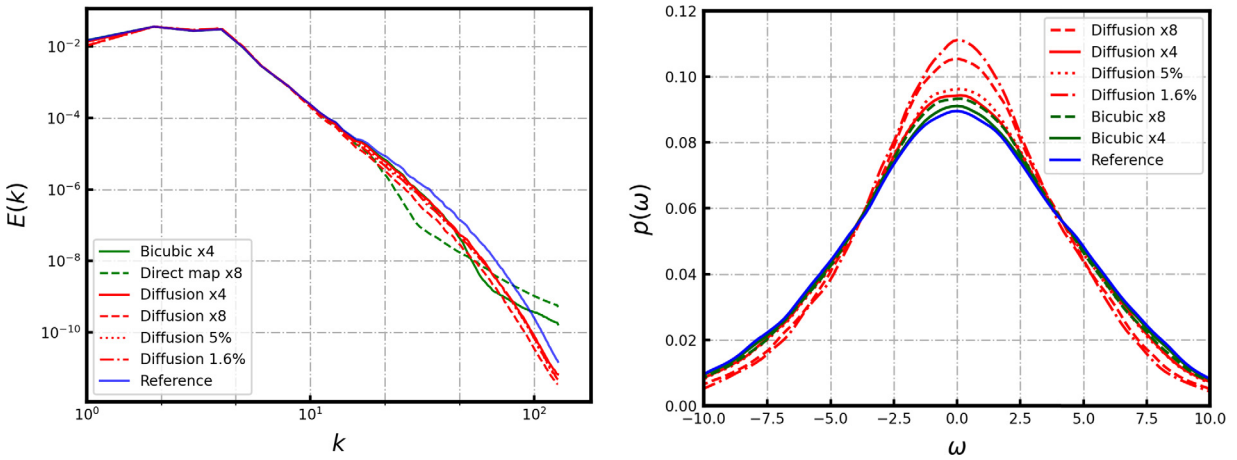


Fig. 6. Qualitative comparison of different upsampling methods on non-equidistant sparse reconstruction task using 1.5625% of grid points. White cross in the input indicates the collocation points (zoom in for clarity).

Table 2
Quantitative comparison of diffusion model and direct mapping model on sparse reconstruction task with 1.5625% sparsity using different filter scales.

Data	L_2 norm		Equation loss	
	Diffusion	Direct map	Diffusion	Direct map
No filter	1.8802	2.1144	0.5428	42.9115
$\sigma = 5$	1.8213	1.9149	0.2290	17.5249
$\sigma = 7$	1.8188	1.8408	0.1682	12.5251

can help improve prediction's L_2 norm, a low-pass filter like Gaussian filter will lose energy in higher frequency regime (as shown in Fig. 8b).



(a) Kinetic energy spectrum (b) Vorticity distribution

Fig. 7. Statistics of different reconstruction methods and reference result.

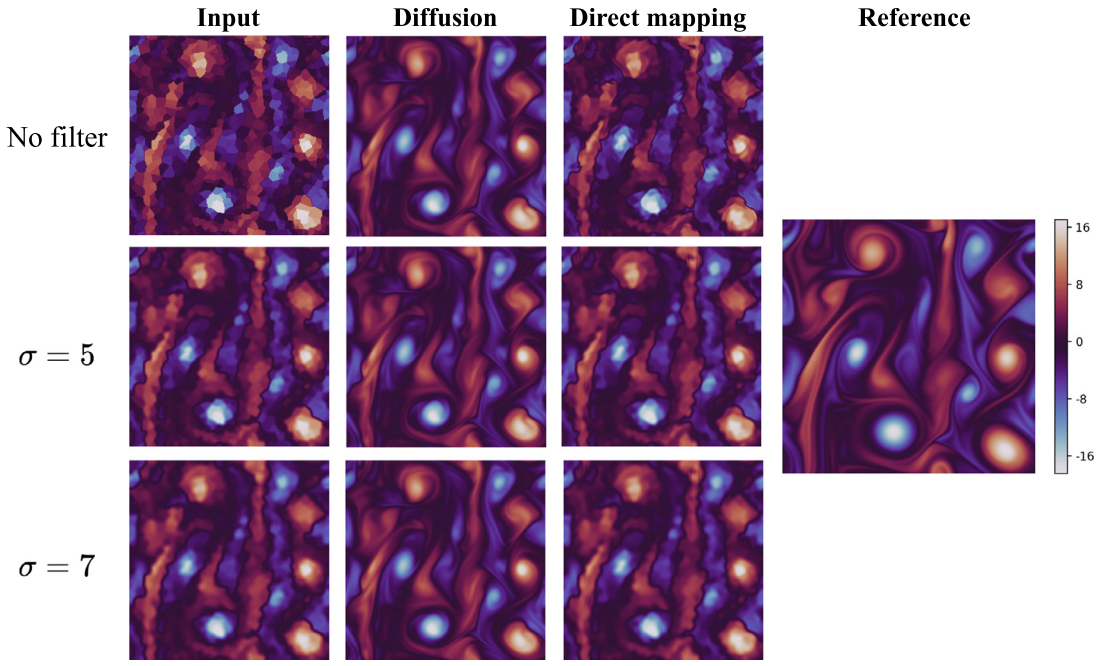
Table 3
A comparison of convergence trend for different methods of combining residual information, where the baseline is the original diffusion model that does not use any residual information. S denotes the total number of backward diffusion steps, while *Iter* denotes the outer diffusion loop as described in Algorithm 2. The upper table is evaluated on the non-equidistant sparse reconstruction task, the bottom table is evaluated on the 8× upsampling task.

Method	L_2 norm				Equation loss			
	Iter 1 $S=8$	Iter 1 $S=16$	Iter 2 $S=30$	Iter 3 $S=36$	Iter 1 $S=8$	Iter 1 $S=16$	Iter 2 $S=24$	Iter 3 $S=36$
Learned	1.74	1.00	0.94	0.93	419.80	7.79	0.36	0.27
Linear	1.75	0.99	0.94	0.93	755.23	8.83	0.52	0.38
Baseline	1.75	0.99	0.94	0.93	832.42	11.17	0.55	0.39

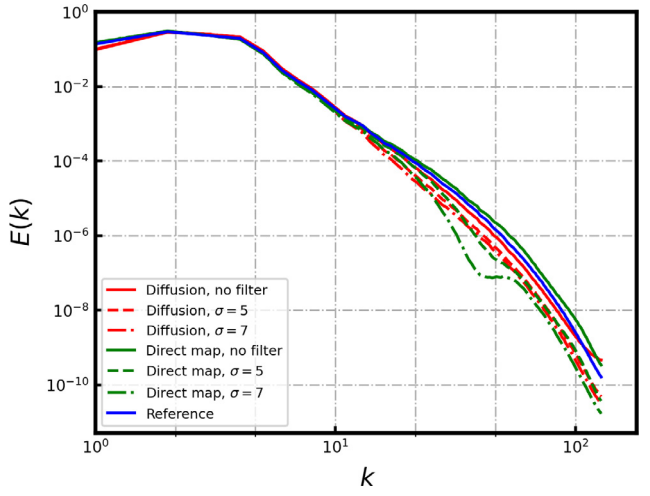
Method	L_2 norm				Equation loss			
	Iter 1 $S=8$	Iter 1 $S=16$	Iter 1 $S=24$	Iter 1 $S=32$	Iter 1 $S=8$	Iter 1 $S=16$	Iter 1 $S=24$	Iter 1 $S=32$
Learned	3.69	2.72	1.77	1.30	1572.02	744.18	177.68	0.20
Linear	3.70	2.74	1.79	1.26	2546.79	1623.54	498.23	0.99
Baseline	3.70	2.75	1.80	1.26	2740.99	1876.36	621.12	1.47

We also study the influence of adding physics-informed condition using different ways. The two different ways are the learned and linear combination between the gradient of data distribution and the gradient of residuals. First, all models have a similar trend on L_2 norm and a negligible difference in the final results (Fig. 9a, 10a). Yet with the residual information, diffusion model converges faster on the equation loss and has a lower final residual (Fig. 9b, 10b, especially for more challenging task where stronger Gaussian noise are added to the input data. More specifically, we observe better convergence for the learned combination which combines the residual gradient with the score function non-linearly via the neural network. This is because at the early stage of backward diffusion the sampled data is very noisy and thus the residual gradient is not very informative, so balancing score function with the gradient of density distribution using a learned neural network is often better than simply combining them linearly. (See Table 3.)

As shown in the previous part of Results section, we have selected three performance metrics to evaluate the model for high-fidelity CFD data reconstruction: 1) L_2 norm, 2) equation loss, and 3) kinetic energy spectrum. L_2 norm is a straightforward way to measure the reconstruction error with respect to the ground truth and has been widely used as a performance metric in many data prediction tasks. L_2 norm is also the loss function used in the training and test of the direct mapping method that benchmarks our proposed method. However, the L_2 norm has two limitations in evaluating the CFD data reconstruction. First, the L_2 norm is less sensitive to the blurring of data. In the case where a large amount of blurring effect is added (*e.g.*, through Gaussian blur) to a CFD data sample, its L_2 norm tends to have an insignificant increase (or even decrease). Second, despite being a popular universal metric to measure the distance between two data points, L_2 norm is not specialized to reveal how accurately a model's prediction reflects the physical characteristics of the fluid, for example, it does not indicate how well the prediction fits the Navier-Stokes equation, or whether the eddies of different scales in the prediction contain the accurate amount of turbulence kinetic energy. While the L_2



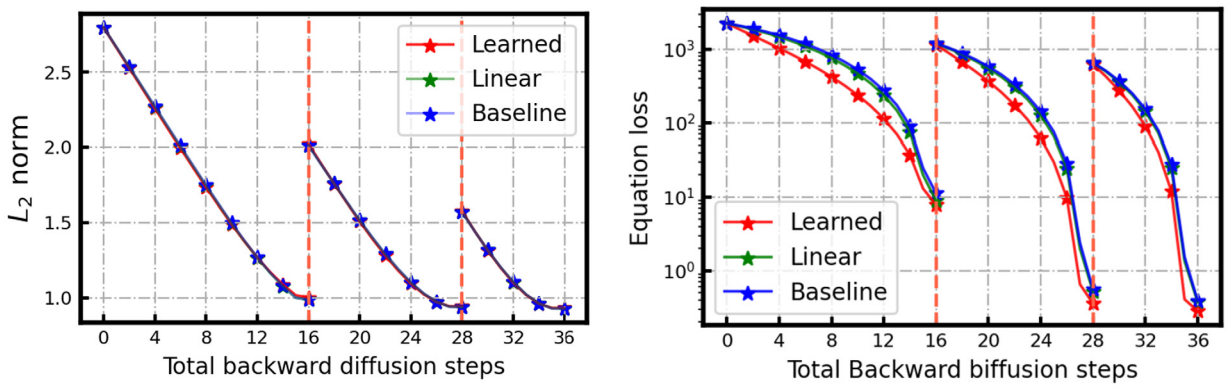
(a) Qualitative comparison of different models' prediction on the sparse reconstruction task with different filtered input.



(b) Energy spectrum of different models' prediction on sparse reconstruction task.

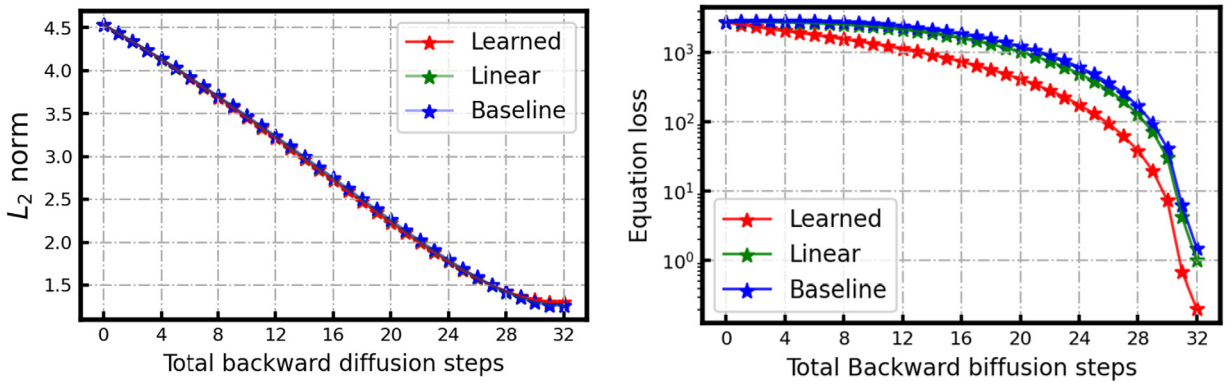
Fig. 8. Comparison of the UNet that learns a direct mapping and the UNet that learns to estimate noise in the diffusion process.

norm is a direct indicator of the averaged point-wise reconstruction error, other metrics such as the equation loss and the kinetic energy spectrum help to reveal how well a reconstructed CFD data sample fits the expected physical characteristics seen in the ground truth sample. Therefore, we include equation loss and kinetic energy spectrum in addition to L_2 norm as the performance metrics in order to have a more comprehensive evaluation of the reconstruction methods. We consider that our proposed method outperforms the benchmark methods such as the direct mapping model and the bicubic interpolation, given the case that our model shows a marginal improvement on L_2 loss, a significant improvement on equation loss, and a closer alignment to the ground truth in terms of the kinetic energy spectrum, as such experimental result indicates that the reconstruction obtained by our proposed method has a comparable L_2-loss



(a) Number of steps - L_2 norm (b) Number of steps - Equation loss (log scale)

Fig. 9. Convergence plots on sparse (5% collocation points) reconstruction using different methods of combining residual information. The vertical red dotted line indicates a new Gaussian noise injection.



(a) Number of steps - L_2 norm (b) Number of steps - Equation loss (log scale)

Fig. 10. Convergence plots on 8× upsampling using different methods of combining residual information. The vertical red dotted line indicates a new Gaussian noise injection.

accuracy with the benchmark method while being more coherent with the ground truth physical characteristics of the fluid.

Conclusion

In this work, we presented a diffusion model for high-fidelity CFD data reconstruction from low-fidelity input. We showed that a diffusion model is able to solve the data reconstruction problem as a conditional data denoising problem. Compared with the benchmark method which learns the direct-mapping from low-fidelity to high-fidelity data, our model has a similar (marginally better) reconstruction accuracy in terms of the L_2 loss, while having the advantage of being much more robust to variation in the low-fidelity input data and being more accurate in terms of data kinetic energy spectrum. In addition, we showed how to incorporate the physics-informed information such as the PDE residual gradient in model training and model inference for an improved performance, and how to reconstruct high-fidelity CFD data from sparsely measured inputs using nearest padding and iterative reconstruction.

Compared with the direct-mapping models, a diffusion model is not trained to directly minimize the reconstruction loss in the sense of an L_p norm. Instead, it is trained to minimize the KL-divergence between the forward diffusion process and the backward diffusion process. As a result, a diffusion model is essentially designed to be only sensitive to data reconstruction error in a statistical sense with the presence of Gaussian noise. We consider the absence of an L_p reconstruction loss as a potential limitation that prevents a diffusion model from further improving its reconstruction accuracy. To resolve such limitation, a new design of the data sampling procedure for the diffusion model is needed. Alternatively, it might be interesting to investigate an ensemble learning method which incorporates a diffusion model and a direct-mapping model, so that the merits of both methods can be retained. These potential ways to resolve the limitation of diffusion models for data reconstruction will be the main direction of our future work.

CRediT authorship contribution statement

Dule Shu: Conceptualization, Software, Visualization, Writing – original draft. **Zijie Li:** Conceptualization, Investigation, Software, Writing – original draft. **Amir Barati Farimani:** Conceptualization, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data and code for this work are publicly available at https://github.com/BaratiLab/Diffusion-based-Fluid-Super-resolution.

Acknowledgement

This work is supported by the National Science Foundation under Grant No. 1953222.

Appendix A. Supplementary material

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jcp.2023.111972.

References

[1] P. Moin, K. Mahesh, Direct numerical simulation: a tool in turbulence research, Annu. Rev. Fluid Mech. 30 (1998) 539–578.
[2] B. Launder, B. Sharma, Application of the energy-dissipation model of turbulence to the calculation of flow near a spinning disc, Lett. Heat Mass Transf. 1 (1974) 131–137.
[3] D.C. Wilcox, Reassessment of the scale-determining equation for advanced turbulence models, AIAA J. 26 (1988) 1299–1310.
[4] J. Smagorinsky, General circulation experiments with the primitive equations: I. The basic experiment, Mon. Weather Rev. 91 (1963) 99–164.
[5] H. Pitsch, Large-eddy simulation of turbulent combustion, Annu. Rev. Fluid Mech. 38 (2006) 453–482.
[6] M.L. Shur, P.R. Spalart, M.K. Strelets, A.K. Travin, A hybrid RANS-LES approach with delayed-DES and wall-modelled LES capabilities, Int. J. Heat Fluid Flow 29 (2008) 1638–1649.
[7] L. Davidson, M. Billson, Hybrid LES-RANS using synthesized turbulent fluctuations for forcing in the interface region, Int. J. Heat Fluid Flow 27 (2006) 1028–1042.
[8] D.K. Walters, S. Bhushan, M. Alam, D.S. Thompson, Investigation of a dynamic hybrid RANS/LES modelling methodology for finite-volume CFD simulations, Flow Turbul. Combust. 91 (2013) 643–667.
[9] S. Jakirlić, G. Kadavelil, M. Kornhaas, M. Schäfer, D. Sternel, C. Tropea, Numerical and physical aspects in LES and hybrid LES/RANS of turbulent flow separation in a 3-D diffuser, Int. J. Heat Fluid Flow 31 (2010) 820–832.
[10] A. Rona, M. El-Dosoky, D. Adebayo, A hybrid RANS model of wing-body junction flow, Eur. J. Mech. B, Fluids 79 (2020) 283–296.
[11] C.W. Li, L. Yu, Hybrid LES/RANS modelling of free surface flow through vegetation, Comput. Fluids 39 (2010) 1722–1732.
[12] R. Maulik, O. San, J.D. Jacob, C. Crick, Sub-grid scale model classification and blending through deep learning, J. Fluid Mech. 870 (2019) 784–812.
[13] P. Tabeling, Two-dimensional turbulence: a physicist approach, Phys. Rep. 362 (2002) 1–62.
[14] G. Boffetta, R.E. Ecke, et al., Two-dimensional turbulence, Annu. Rev. Fluid Mech. 44 (2012) 427–451.
[15] B. Pearson, B. Fox-Kemper, S. Bachman, F. Bryan, Evaluation of scale-aware subgrid mesoscale eddy models in a global eddy-rich model, Ocean Model. 115 (2017) 42–58.
[16] B. Pearson, B. Fox-Kemper, Log-normal turbulence dissipation in global ocean models, Phys. Rev. Lett. 120 (2018) 094501.
[17] J. Bardina, J. Ferziger, W. Reynolds, Improved subgrid-scale models for large-eddy simulation, in: 13th Fluid and Plasmadynamics Conference, 1980, p. 1357.
[18] S. Stolz, N.A. Adams, An approximate deconvolution procedure for large-eddy simulation, Phys. Fluids 11 (1999) 1699–1701.
[19] W. Layton, R. Lewandowski, A simple and stable scale-similarity model for large eddy simulation: energy balance and existence of weak solutions, Appl. Math. Lett. 16 (2003) 1205–1209.
[20] J. Mathew, R. Lechner, H. Foysi, J. Sesterhenn, R. Friedrich, An explicit filtering method for large eddy simulation of compressible flows, Phys. Fluids 15 (2003) 2279–2289.
[21] O. San, P. Vedula, Generalized deconvolution procedure for structural modeling of turbulence, J. Sci. Comput. 75 (2018) 1187–1206.
[22] B. Ummenhofer, L. Prantl, N. Thuerey, V. Koltun, Lagrangian fluid simulation with continuous convolutions, in: International Conference on Learning Representations, 2019.
[23] Z. Li, A. Barati Farimani, Graph neural network-accelerated Lagrangian fluid simulation, Comput. Graph. 103 (2022) 201–211.
[24] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, P.W. Battaglia, Learning to simulate complex physics with graph networks, https://arxiv.org/abs/2002.09405, 2020.
[25] L. Ladický, S. Jeong, B. Solenthaler, M. Pollefeys, M. Gross, Data-driven fluid simulations using regression forests, ACM Trans. Graph. 34 (2015).
[26] A. Zhumekenov, M. Uteuliyeva, O. Kabdolov, R. Takhanov, Z. Assylbekov, A.J. Castro, Fourier neural networks: a comparative study, arXiv preprint, arXiv:1902.03011, 2019.
[27] R. Wang, K. Kashinath, M. Mustafa, A. Albert, R. Yu, Towards physics-informed deep learning for turbulent flow prediction, https://arxiv.org/abs/1911.08655, 2019.
[28] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, P.W. Battaglia, Learning Mesh-Based Simulation with Graph Networks, 2020.
[29] K. Stachenfeld, D.B. Fielding, D. Kochkov, M. Cranmer, T. Pfaff, J. Godwin, C. Cui, S. Ho, P. Battaglia, A. Sanchez-Gonzalez, Learned coarse models for efficient turbulence simulation, https://arxiv.org/abs/2112.15275, 2021.
[30] J. Brandstetter, D. Worrall, M. Welling, Message passing neural PDE solvers, https://arxiv.org/abs/2202.03376, 2022.
[31] S. Wiewel, M. Becher, N.Thuerey, Latent-space physics: towards learning the temporal evolution of fluid flow, https://arxiv.org/abs/1802.10123, 2018.

[32] N. Thuerey, K. Weißenow, L. Prantl, X. Hu, Deep learning methods for Reynolds-averaged Navier–Stokes simulations of airfoil flows, AIAA J. 58 (2020) 25–36.
[33] Z. Li, K. Meidani, A.B. Farimani, Transformer for partial differential equations' operator learning, arXiv preprint, arXiv:2205.13671, 2022.
[34] P. Pant, R. Doshi, P. Bahl, A. Barati Farimani, Deep learning for reduced order modelling and efficient temporal evolution of fluid simulations, Phys. Fluids 33 (2021) 107101.
[35] A. Barati Farimani, J. Gomes, V.Pande, Deep Learning Fluid Mechanics, in: APS Division of Fluid Dynamics Meeting Abstracts, 2017, pp. E31–004.
[36] T. Karras, T. Aila, S. Laine, J. Lehtinen, Progressive growing of gans for improved quality, stability, and variation, arXiv preprint, arXiv:1710.10196, 2017.
[37] T. Park, M.-Y. Liu, T.-C. Wang, J.-Y. Zhu, Semantic image synthesis with spatially-adaptive normalization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 2337–2346.
[38] Y. Wang, F. Perazzi, B. McWilliams, A. Sorkine-Hornung, O. Sorkine-Hornung, C. Schroers, A fully progressive approach to single-image super-resolution, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, pp. 864–873.
[39] Z. Lu, J. Li, H. Liu, C. Huang, L. Zhang, T. Zeng, Transformer for single image super-resolution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 457–466.
[40] F. Yang, H. Yang, J. Fu, H. Lu, B. Guo, Learning texture transformer network for image super-resolution, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 5791–5800.
[41] J. Cao, Y. Li, K. Zhang, L. Van Gool, Video super-resolution transformer, arXiv preprint, arXiv:2106.06847, 2021.
[42] C. Saharia, J. Ho, W. Chan, T. Salimans, D.J. Fleet, M. Norouzi, Image super-resolution via iterative refinement, IEEE Trans. Pattern Anal. Mach. Intell. PP (2022) 1–14.
[43] J. Ho, C. Saharia, W. Chan, D.J. Fleet, M. Norouzi, T. Salimans, Cascaded diffusion models for high fidelity image generation, J. Mach. Learn. Res. 23 (2022) 1–33.
[44] H. Li, Y. Yang, M. Chang, S. Chen, H. Feng, Z. Xu, Q. Li, Y. Chen, SRDiff: single image super-resolution with diffusion probabilistic models, Neurocomputing 479 (2022) 47–59.
[45] P. Pant, A.B. Farimani, Deep learning for efficient reconstruction of high-resolution turbulent DNS data, arXiv preprint, arXiv:2010.11348, 2020.
[46] K. Fukami, K. Fukagata, K. Taira, Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows, J. Fluid Mech. 909 (2021).
[47] K. Fukami, K. Fukagata, K. Taira, Super-resolution reconstruction of turbulent flows with machine learning, J. Fluid Mech. 870 (2019) 106–120.
[48] M.Z. Yousif, L. Yu, H.-C. Lim, High-fidelity reconstruction of turbulent flow from spatially limited data using enhanced super-resolution generative adversarial network, Phys. Fluids 33 (2021) 125119.
[49] H. Kim, J. Kim, S. Won, C. Lee, Unsupervised deep learning for super-resolution reconstruction of turbulence, J. Fluid Mech. 910 (2021).
[50] J.-Y. Zhu, T. Park, P. Isola, A.A. Efros, Unpaired image-to-image translation using cycle-consistent adversarial networks, in: Proceedings of the IEEE International Conference on Computer Vision, 2017, pp. 2223–2232.
[51] M. Matsuo, T. Nakamura, M. Morimoto, K. Fukami, K. Fukagata, Supervised convolutional network for three-dimensional fluid data reconstruction from sectional flow fields with adaptive super-resolution assistance, https://arxiv.org/abs/2103.09020, 2021.
[52] Y. Xie, E. Franz, M. Chu, N. Thuerey, tempoGAN: a temporally coherent, volumetric gan for super-resolution fluid flow, ACM Trans. Graph. (TOG) 37 (2018) 1–15.
[53] Z. Li, T. Li, A.B. Farimani, TPU-GAN: learning temporal coherence from dynamic point cloud sequences, in: International Conference on Learning Representations, 2021.
[54] L. Prantl, N. Chentanez, S. Jeschke, N.Thuerey, Tranquil clouds: neural networks for learning temporally coherent features in point clouds, https://arxiv.org/abs/1907.05279, 2019.
[55] D. Kochkov, J.A. Smith, A. Alieva, Q. Wang, M.P. Brenner, S. Hoyer, Machine learning–accelerated computational fluid dynamics, Proc. Natl. Acad. Sci. USA 118 (2021) e2101784118.
[56] K. Um, R. Brand, Y.R. Fei, P. Holl, N. Thuerey, Solver-in-the-loop: learning from differentiable physics to interact with iterative pde-solvers, Adv. Neural Inf. Process. Syst. 33 (2020) 6111–6122.
[57] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, Adv. Neural Inf. Process. Syst. 33 (2020) 6840–6851.
[58] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. 378 (2019) 686–707.
[59] G.E. Karniadakis, I.G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, L. Yang, Physics-informed machine learning, Nat. Rev. Phys. 3 (2021) 422–440.
[60] S. Cai, Z. Mao, Z. Wang, M. Yin, G.E. Karniadakis, Physics-informed neural networks (PINNs) for fluid mechanics: a review, Acta Mech. Sin. (2022) 1–12.
[61] C. Meng, Y. Song, J. Song, J. Wu, J.-Y. Zhu, S. Ermon, SDEdit: image synthesis and editing with stochastic differential equations, arXiv preprint, arXiv:2108.01073, 2021.
[62] A. Lugmayr, M. Danelljan, A. Romero, F. Yu, R. Timofte, L.Van Gool, Repaint inpainting using denoising diffusion probabilistic models, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 11461–11471.
[63] H. Sasaki, C.G. Willcocks, T.P. Breckon, UNIT-DDPM: unpaired image translation with denoising diffusion probabilistic models, arXiv preprint, arXiv:2104.05358, 2021.
[64] C. Saharia, W. Chan, H. Chang, C. Lee, J. Ho, T. Salimans, D. Fleet, M. Norouzi, Palette image-to-image diffusion models, in: ACM SIGGRAPH 2022 Conference Proceedings, 2022, pp. 1–10.
[65] D.P. Kingma, M. Welling, Auto-encoding variational Bayes, arXiv preprint, arXiv:1312.6114, 2013.
[66] J. Ho, T. Salimans, Classifier-free diffusion guidance, arXiv preprint, arXiv:2207.12598, 2022.
[67] J. Song, C. Meng, S. Ermon, Denoising diffusion implicit models, arXiv preprint, arXiv:2010.02502, 2020.
[68] Y. Song, S.Ermon, Generative modeling by estimating gradients of the data distribution, https://arxiv.org/abs/1907.05600, 2019.
[69] G.J. Chandler, R.R. Kerswell, Invariant recurrent solutions embedded in a turbulent two-dimensional Kolmogorov flow, J. Fluid Mech. 722 (2013) 554–595.
[70] D. Kochkov, J.A. Smith, A. Alieva, Q. Wang, M.P. Brenner, S. Hoyer, Machine learning–accelerated computational fluid dynamics, Proc. Natl. Acad. Sci. USA 118 (2021).
[71] G. Boffetta, R.E. Ecke, Two-dimensional turbulence, Annu. Rev. Fluid Mech. 44 (2012) 427–451.
[72] A. Paszke, et al., PyTorch: an imperative style, high-performance deep learning library, https://arxiv.org/abs/1912.01703, 2019.
[73] Z. Li, H. Zheng, N. Kovachki, D. Jin, H. Chen, B. Liu, K. Azizzadenesheli, A. Anandkumar, Physics-informed neural operator for learning partial differential equations, https://arxiv.org/abs/2111.03794, 2021.
[74] O. Ronneberger, P. Fischer, T. Brox, U-Net: convolutional networks for biomedical image segmentation, https://arxiv.org/abs/1505.04597, 2015.
[75] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, https://arxiv.org/abs/1706.03762, 2017.