Reconstructing Turbulent Flows Using Physics-Aware Spatio-Temporal Dynamics and Test-Time Refinement

SHENGYU CHEN, University of Pittsburgh, United States
TIANSHU BAO, University of Vanderbilt, United States
PEYMAN GIVI, University of Pittsburgh, United States
CAN ZHENG, University of Pittsburgh, United States
XIAOWEI JIA, University of Pittsburgh, United States

Simulating turbulence is critical for many societally important applications in aerospace engineering, environmental science, the energy industry, and biomedicine. Large eddy simulation (LES) has been widely used as an alternative to direct numerical simulation (DNS) for simulating turbulent flows due to its reduced computational cost. However, LES is unable to capture all of the scales of turbulent transport accurately. Reconstructing DNS from low-resolution LES is critical for many scientific and engineering disciplines, but it poses many challenges to existing super-resolution methods due to the spatio-temporal complexity of turbulent flows. In this work, we propose a new physics-guided neural network for reconstructing the sequential DNS from low-resolution LES data. The proposed method leverages the partial differential equation that underlies the flow dynamics in the design of spatio-temporal model architecture. A degradation-based refinement method is also developed to enforce physical constraints and further reduce the accumulated reconstruction errors over long periods. The results on two different types of turbulent flow data confirm the superiority of the proposed method in reconstructing the high-resolution DNS data and preserving the physical characteristics of flow transport.

CCS Concepts: • ;

ACM Reference Format:
Shengyu Chen, Tianshu Bao, Peyman Givi, Can Zheng, and Xiaowei Jia. 2018. Reconstructing Turbulent Flows Using Physics-Aware Spatio-Temporal Dynamics and Test-Time Refinement. *J. ACM* 37, 4, Article 111 (August 2018), 18 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Understanding and simulating turbulence is a key element to our comprehension of many natural and technological processes in science and engineering. Direct numerical simulation (DNS) of the Navier-Stokes equations is a brute-force method and is commonly regarded as the most high-fidelity method for capturing turbulence dynamics [16]. The computation cost of such simulations can be very expensive for complex turbulence, i.e., flows with high Reynolds numbers. Large eddy simulation (LES) is the most widely used alternative, concentrating on the larger scale energy-containing eddies and filtering the small scales of transport [32]. By this filtering, LES can be conducted on coarser grids as compared to those required by DNS, but the fidelity of the LES data is generally lower than DNS [29].

Authors' addresses: Shengyu Chen, University of Pittsburgh, Pittsburgh, United States; Tianshu Bao, University of Vanderbilt, Nashville, United States; Peyman Givi, University of Pittsburgh, Pittsburgh, United States; Can Zheng, University of Pittsburgh, Pittsburgh, United States; Xiaowei Jia, University of Pittsburgh, Pittsburgh, United States.

0004-5411/2018/8-ART111 $15.00
https://doi.org/XXXXXXX.XXXXXXX

Machine learning, including super-resolution (SR) methods [30], provides the unrealized potential for reconstructing high-fidelity DNS from LES data at a coarser resolution. These techniques have already shown tremendous success in reconstructing high-resolution data in various commercial applications. Most existing SR models use convolutional network layers (CNNs) [3] to extract representative spatial features and transform them through complex non-linear mappings to recover high-resolution images. From the end-to-end convolutional SRCNN model [12], researchers have explored adding other structural components such as skip-connections [2, 10, 13, 13, 43, 44] and channel attention [43], and using adversarial training objectives [8, 9, 22, 26, 34–36, 38].

Given their success in computer vision, SR methods are increasingly used in turbulence reconstruction [11, 14, 15, 27, 36, 42]. Although these methods have shown promise in simple and isolated benchmark data, they are faced with several major challenges before they can be employed for reconstructing various types of turbulent flows over long periods. Existing SR models are not designed for modeling complex spatio-temporal evolution and interactions amongst multiple physical variables in turbulence. Besides, high-fidelity DNS samples are often limited for training the SR models due to the computational cost of the brute-force method. Moreover, existing SR models and temporal models are limited in representing continuous flow dynamics using data samples at discrete time steps. As a result, these models can learn spurious patterns between sparse observations, and such patterns are often not generalizable.

We develop a novel method, termed Continuous Networks Using Differential Equation (CNDE), to improve the turbulent flow reconstruction in spatial and temporal fields. This development is by leveraging underlying physical relationships to guide the learning of generalizable spatial and temporal patterns in the reconstruction process. In particular, our proposed method consists of three components, Runge-Kutta Transition Unit (RKTU), Temporally-Enhancing Layer (TEL), and degradation-based refinement. The RKTU structure is designed based on the underlying partial differential equations (PDE) and is used for capturing continuous spatial and temporal dynamics of turbulent flows. The TEL is designed based on the LSTM model and is responsible for capturing long-term temporal dependencies. In addition, we propose a degradation-based refinement method to adjust the reconstructed data over time by enforcing consistency with the LES data and known physical constraints.

Our evaluations on the Forced Isotropic Turbulence (FIT) dataset [1] and the Taylor-Green Vortex (TGV) dataset [5] have shown the superiority of the proposed CNDE in terms of the reconstruction performance over space and time. We also demonstrate the effectiveness of each component of our design by showing the improvement both qualitatively and quantitatively. Moreover, we verify that the proposed method can preserve important physical characteristics of turbulent flows.

2 RELATED WORK

2.1 Overview of Super Resolution Method

Researchers have developed many deep learning-based methods for single image super resolution (SISR) in computer vision. The power of these methods comes mainly from the use of convolutional network layers [3], which can extract the spatial texture features and transform them through complex non-linear mappings to recover high-resolution data. The earliest SR method that uses deep convolutional networks for the SISR problem is SRCNN [12], which directly learns the end-to-end mapping between coarse-resolution and high-resolution images using a series of convolutional layers. After that, Residual Channel Attention Network (RCAN) [43] uses a very deep trainable structure with additional skip-connection layers to bypass the abundant low-frequency information and focus more on the relevant information. The functionality of skip connections is also known to improve the stability of the optimization process for deep neural networks. Recently, there are

also other popular methods based on residual structures such as HDRN [13], SAN [10], RDN [44], CARN [2], and DRRN [33].

Another popular super-resolution model is the SRGAN model [26], which employs a generative adversarial network (GAN) for the SISR problem. SRGAN model not only stacks the deep residual network to build a deeper generative network for image super resolution, but also introduces a discriminator network to distinguish between reconstructed images and real images using an adversarial loss function. The ultimate goal is to train the generative network such that the reconstructed images cannot be easily distinguished by the discriminator. Compared with other models, one major advantage of the SRGAN model is that the discriminator can help extract representative features from high-resolution data and enforce such features in the reconstructed images. Recently, there are also other extensions to the SRGAN method that further improve the performance [8, 9, 22, 34–36, 38].

2.2 Super Resolution Method for Reconstructing Flow Data

Given the importance of simulating high-resolution flows, there is a surge of interest in using super-resolution techniques for reconstructing high-resolution flow data. Fukami et al. [14, 15, 27] propose an improved CNN-based hybrid DSC/MS model to explore multiple scales of turbulence and capture the spatial-temporal turbulence dynamics. Similarly, Liu et al. [27] also propose another CNN-based model MTPC to simultaneously handle spatial and temporal information in turbulent flow simultaneously to fully capture features in different time ranges. Here are also other approaches that are inspired by GAN. For example, Xie et al. [42] introduce tempoGAN, which augments a generative adversarial network with an additional discriminator network along with additional loss function terms that preserve temporal coherence in the generation of physics-based simulations of fluid flow. Deng et al. [11] demonstrate that both SRGAN and ESRGAN [36] can produce a good reconstruction of high-resolution turbulent flow in their datasets. However, these existing SR methods face several challenges when reconstructing turbulent flows. Such flows involve multiple physical variables and often exhibit complex dynamic patterns, i.e., multiple physical variables evolve and interact at different scales. In the absence of underlying physical processes, pure data-driven SR methods require a large number of training samples to capture the correct physics.

Hence, researchers have also explored integrating physics into the SR models. Chen et al. [7] propose a PGSRN method to enforce divergence-free properties in incompressible flows, but this method remains limited in capturing the temporal flow dynamics. Bao et al. [4] use the Navier-Stokes equation in a recurrent network to predict DNS data from historical DNSbut this method does not fully leverage the LES data and also has accumulated errors in long-term prediction.

2.3 Physics-Guided Machine Learning

Recent research has shown immense success in integrating physics knowledge into machine learning models to improve predictive performance and solve general scientific problems [41]. A majority of these methods enforce physics in the loss function. For example, Hanson et al. [17] introduced ecological principles as physical constraints into the loss function to improve the lake surface water phosphorus prediction. Karpatne et al. [21] proposed a hybrid machine learning and physics model to guarantee that the density of water at a lower depth is always greater than the density of water at any depth above. Then, Jia et al. [19] and Read et al. [31] further extended this idea by including an additional penalty for the violation of the law of energy conservation. Despite the promise of these methods, they may lead to slow convergence in optimization and performance degradation, especially when the physical relationships are complex or have uncertain parameters. To address this issue, an alternative direction is to build new model structures using

physics [4, 23, 28]. These models naturally encode known physical relationships while also allowing the flexibility to adjust the model using training data.

3 PROBLEM DEFINITION

We consider a general three-dimensional turbulent flow over space and time $\mathbf{Q}(x, y, z, t)$, where (x, y, z) denotes the spatial coordinates, t represents the time step, and $\mathbf{Q}(x, y, z, t)$ consists of multiple variables that describe turbulent transport, including the three components of the velocity along the x, y, and z axes, denoted by u, v and w, and the thermodynamic pressure, denoted by p. The flow variables in $\mathbf{Q}(x, y, z, t)$ also follow the Navier-Stokes equation, which governs the transport of these variables in space (x, y, z) and time t. In the training process, we are provided with the available DNS data at a regular time interval δ, as $\mathbf{Q}^d = \{\mathbf{Q}_d^t\}$ within the time $\{t_0, t_0 + \delta, ..., t_0 + K\delta\}$. Our objective is to predict high-resolution DNS data after the historical data, at time $\{t_0 + (K + 1)\delta, ..., t_0 + M\delta\}$. We also use $\mathbf{Q}^l(x, y, z, t)$ to represent the low-resolution LES data at time step t. Since the LES data can be created at a lower computational cost, they can be available for both training and testing periods and at a higher frequency. We use $\mathbf{Q}^l = \{\mathbf{Q}_l^t\}$ to represent LES data within the time range $[t_0, t_0 + M\delta]$.

4 METHOD

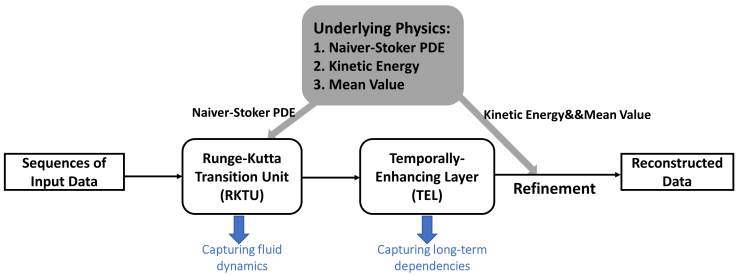


Fig. 1. The overall structure of proposed Continuous Networks Using Differential Equation (CNDE) method.

Our proposed CNDE framework consists of two structural components: RKTU and TEL, and two learning processes, supervised super-resolution training and degradation-based refinement, as shown in Fig. 1. In the following, we will describe the structural components RKTU and TEL, as well as the degradation-based refinement process.

4.1 Runge-Kutta Transition Unit (RKTU)

4.1.1 Overall RKTU Structure. The complex turbulent flow data $\mathbf{Q}$ involve multiple physical variables (e.g., velocity, pressure) that interact with each other and evolve at different speeds for different locations. The traditional temporal models, e.g., long-short term memory (LSTM) [18], rely on large and consecutive training samples to capture the underlying temporal patterns over time. However, high-fidelity DNS data are often limited and less frequent because simulating high-resolution DNS data is computationally expensive. We propose the RKTU structure for reconstructing flow variables in a long period given an initial DNS sample $\mathbf{Q}^d$ at t and frequent low-resolution LES data samples $\mathbf{Q}^l$. The prediction follows an auto-regressive process, in which the predicted DNS $\hat{\mathbf{Q}}^d(x, y, z, t)$ at time t and frequent LES data $\mathbf{Q}^l$ in time $[t,t + \delta]$ are used to predict the DNS at next time step $\hat{\mathbf{Q}}^d(x, y, z, t + \delta)$. Here we assume LES can be conducted more frequently due to its reduced computational cost.

The central idea of the RKTU method is to leverage the continuous physical relationship described by the underlying PDE to bridge the gap between discrete data samples and continuous flow dynamics. The RKTU can also be generally applied to many dynamical systems with governing PDEs. In particular, most PDEs can be expressed as:

$$\mathbf{Q}_t = \mathbf{f}(t, \mathbf{Q}; \theta), \tag{1}$$

where $\mathbf{Q}_t$ is the temporal derivative of $\mathbf{Q}$, and $\mathbf{f}(t, \mathbf{Q}; \theta)$ is a non-linear function (parameterized by coefficient θ) that summarizes the current value of $\mathbf{Q}$ and its spatial context. The turbulence data follows Navier-Stoke PDE. In particular, for incompressible flows, the Navier-Stokes equations for the velocity field can be described as follows:

$$\mathbf{f}(\mathbf{Q}) = \frac{-1}{\rho}\nabla p + \nu\Delta\mathbf{Q} - (\mathbf{Q}\cdot\nabla)\mathbf{Q}, \tag{2}$$

where ρ, p, and ν denote the fluid density, the thermodynamic pressure, and the viscosity, respectively, and $\Delta\mathbf{Q}$ and $\nabla\cdot\mathbf{Q}$ denote the Laplacian and divergence of $\mathbf{Q}$, respectively. We omit the independent variable t in the function $\mathbf{f}(\cdot)$ because $\mathbf{f}(\mathbf{Q})$ in the Navier-Stokes equation is for a specific time t (same with t in $\mathbf{Q}_t$). We consider p as a known variable, and $\theta = \{\rho, \nu\}$.

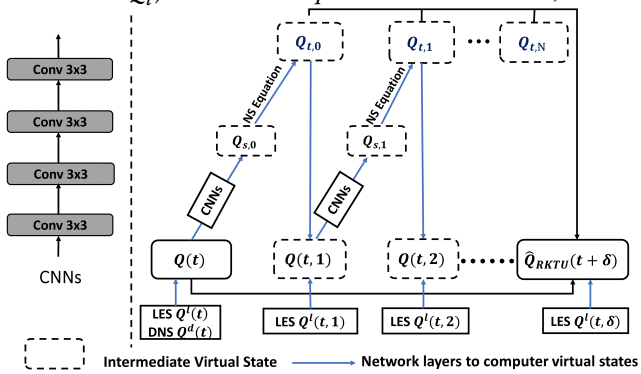


Fig. 2. The proposed Runge-Kutta Transition Unit (RKTU) based on Naiver Stoke equation for reconstructing turbulent flow data in the spatio-temporal field.

The RKTU is inspired by the classical numerical Runge–Kutta (RK) method [6], which has been widely used in temporal discretization for the approximate solutions of differential equations. Fig. 2 shows the overall structure of the RKTU. The RKTU enhances capturing the continuous dynamics by creating smaller intervals between t and $t+\delta$ and interpolating virtual intermediate state variables and creating smaller intervals between time step t and $t+\delta$. As shown in Fig. 2, starting from $\mathbf{Q}(t,0) = \mathbf{Q}(t)$, RKTU estimates the temporal gradient of this point as $\mathbf{Q}_{t,0}$, and then move $\mathbf{Q}(t)$ towards the gradient direction to create the next intermediate state $\mathbf{Q}(t,1)$. Then RKTU iteratively repeats this process until it obtains temporal gradients at N intermediate states.

Specifically, for the starting data point $\mathbf{Q}(t)$, we adopt an augmentation mechanism combining the DNS data with LES data, as $\mathbf{Q}(t) = W^d\mathbf{Q}^d(t) + W^l\mathbf{Q}^l(t)$, where W^d and W^l are trainable model parameters, and $\mathbf{Q}^l(t)$ is the up-sampled LES data in the same resolution as DNS. Then, RKTU estimates the first temporal gradient $\mathbf{Q}_{t,0} = \mathbf{f}(\mathbf{Q}(t))$ using the Navier-Stokes equation and computes the next intermediate state variable $\mathbf{Q}(t,1)$ by moving the flow data $\mathbf{Q}(t)$ along the direction of obtained temporal derivatives. We will discuss more details about the function $\mathbf{f}(\cdot)$ later.

Given frequent LES data, we also augment the $\mathbf{Q}(t,n)$ using LES data $\mathbf{Q}^l(t,n)$, as $\mathbf{Q}(t,n) = W^d\mathbf{Q}(t,n) + W^l\mathbf{Q}^l(t,n)$, and move the $\mathbf{Q}(t,n)$ to compute the next intermediate states $\mathbf{Q}(t,n+1)$. In our test, we follow the most popular 4^{th} order ($N=3$) RK method for computing the next three

intermediate state variables, as follows:

$$\begin{aligned}\mathbf{Q}(t,1) &= \mathbf{Q}(t) + \delta\frac{\mathbf{Q}_{t,0}}{2},\\ \mathbf{Q}(t,2) &= \mathbf{Q}(t) + \delta\frac{\mathbf{Q}_{t,1}}{2},\\ \mathbf{Q}(t,3) &= \mathbf{Q}(t) + \delta\mathbf{Q}_{t,2}.\end{aligned} \tag{3}$$

The temporal derivative $\mathbf{Q}_{t,3}$ is then computed from the last intermediate state by $\mathbf{f}(\mathbf{Q}(t,3))$. The 4^{th} order RK method has the total accumulated error of $O(\delta^4)$. According to Eq. 3, the intermediate LES data $\mathbf{Q}^I(t,n)$ are selected as $\mathbf{Q}^I(t,1) = \mathbf{Q}^I(t+\delta/2)$, $\mathbf{Q}^I(t,2) = \mathbf{Q}^I(t+\delta/2)$, and $\mathbf{Q}^I(t,3) = \mathbf{Q}^I(t+\delta)$. Finally, RKTU combines all the intermediate temporal derivatives as a composite gradient to calculate the next step flow data $\hat{\mathbf{Q}}_{\text{RKTU}}(t+d)$, which can be expressed as:

$$\hat{\mathbf{Q}}_{\text{RKTU}}(t+\delta) = \mathbf{Q}(t) + \sum_{n=0}^{N} w_n \mathbf{Q}_{t,n}, \tag{4}$$

where $\{w_n\}_{n=1}^{N}$ are the trainable model parameters.

In the following, we will describe how to compute the spatial and temporal derivative of the function $\mathbf{f}(\cdot)$. We will also investigate the boundary conditions of the function $\mathbf{f}(\cdot)$ and the stability of RKTU for long-term prediction.

4.1.2 Spatial and Temporal Derivative. The RKTU estimates the temporal derivatives through the function $\mathbf{f}(\cdot)$. According to Eq. 2, the evaluation of $\mathbf{f}(\cdot)$ requires explicitly estimating the first-order and second-order spatial derivatives. One of the most popular approaches for evaluating spatial derivatives is through finite difference methods (FDMs) [39]. However, the discretization in FDMs can cause larger errors for locations with complex dynamics. Thus, we design convolutional neural network layers (CNNs) in RKTU structure to replace the FDMs, shown in Fig. 2. The main power of CNNs is used to learn extra non-linear relationships from data and capture the spatial derivative to be used in the Navier-Stokes equation. After estimating the first-order and second-order spatial derivatives, we can use them to calculate the divergence and Laplacian in Eq. 2, and finally the temporal derivative $\mathbf{Q}_{t,n}$.

4.1.3 Boundary Padding and Stability. Boundary conditions of turbulent flow describe how the flow data interact with the external environment. How preserve the boundary condition of flow data is critical for flow data reconstruction. In this study, we consider the periodic boundary conditions, which are defined in a specified periodic domain indicating that it repeats its own values in all directions. The formal definition of a cubic periodic boundary condition can be represented by:

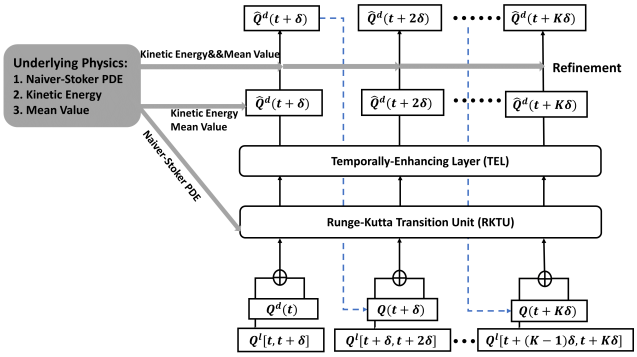
$$\begin{aligned}\mathbf{Q}(L_x,y,z,t) &= \mathbf{Q}(R_x,y,z,t),\\ \mathbf{Q}(x,L_y,z,t) &= \mathbf{Q}(x,R_y,z,t),\\ \mathbf{Q}(x,y,R_z,t) &= \mathbf{Q}(x,y,R_z,t),\end{aligned} \tag{5}$$

where L_x, L_y, L_z are the three left boundaries with respect with x, y, z coordinates and R_x, R_y, R_z are the three right boundaries with respect with x, y, z coordinates. We make a periodic data augmentation for each of the 6 faces (of the 3D cubic data) with an additional two layers of data before feeding it to the SR model.

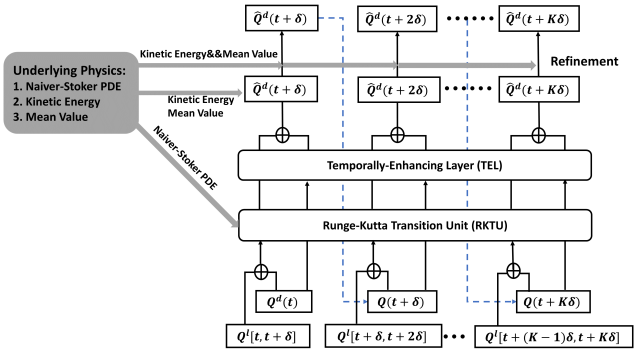
We also consider the stability issue of the classical 4^{th} order RK method. This RK method will encounter a stability issue if the step size is not properly chosen. To illustrate this problem, we consider a simple scalar example $Q_t = \lambda Q$. The 4^{th} order RK for this equation can be written as

$$Q((n+1)\delta) \approx (1 + \lambda\delta + \frac{\lambda\delta^2}{2} + \frac{\lambda\delta^3}{6} + \frac{\lambda\delta^4}{24})Q(n\delta). \tag{6}$$

We indicate that $R(\delta)$=1 + δ + $\frac{\delta^2}{2}$ + $\frac{\delta^3}{6}$ + $\frac{\delta^4}{24}$, and we have $Q((n+1)\delta) = R(\delta)Q(n\delta)$. Hence, the analytical solution can be expressed as $Q((n+1)d\delta) = \exp(\lambda\delta)Q(n\delta)$, and the accumulated error is $err_{n+1} = (\exp(\lambda\delta) - R(\delta))err_n$. This indicates that $err_{n+1} = O(\delta^5)err_n$ according to the Taylor expansion.



(a) Enhancing Method



(b) Residual Learning Method

Fig. 3. The details of proposed Continuous Networks Using Differential Equation (CNDE) for reconstructing turbulent flows **Q**. (a) and (b) show the two different approaches to incorporating the TEL component, respectively.

4.2 Temporally-Enhancing Layer (TEL)

The RKTU can capture the flow data in the spatial and temporal field between a pair of consecutive data points but it may cause large reconstruction errors in the long-time prediction if the time interval δ is large enough. Temporal models, such as long-short-term memory (LSTM) [18], and temporal convolutional network (TCN) [25] are widely used for capturing long-term dependencies in time series prediction. In this case, we incorporate the LSTM model in a Temporally-Enhancing Layer (TEL) to further enhance the ability of the RKTU in capturing long-term temporal dependencies. This TEL model structure can be replaced by other existing temporal models such as TCN. Figs. 3 show two different approaches for integrating the TEL structure with the RKTU structure.

In the first enhancing method shown in Fig. 3(a), we feed the RKTU output flow data $\hat{\mathbf{Q}}_{\text{RKTU}}$ to the TEL structure, which is essentially an LSTM layer. After further processing through the TEL structure, the model produces the reconstructed flow data $\hat{\mathbf{Q}}^{\text{d}}(t)$, Given true DNS data $\mathbf{Q}^d(t)$ in the training set, the reconstructed loss $\mathcal{L}_{\text{recon}}$ can be expressed using the mean squared error (MSE)

loss, as

$$\mathcal{L}_{\text{recon}} = \sum_{t} \text{MSE}(\hat{\mathbf{Q}}^d(t), \mathbf{Q}^d(t)). \tag{7}$$

The second method aims to use the TEL structure to complement the output of the RKTU structure, i.e., learning the residual of the RKTU output, as shown in Fig. 3(b). In the training process, we use both true DNS data $\mathbf{Q}^d$ at time $\{t, ...t + (K-1)\delta\}$ and RKTU output $\hat{\mathbf{Q}}_{\text{RKTU}}$ to produce the corresponding temporal output feature $\hat{\mathbf{Q}}_{\text{TEL}}$ at time $\{t+\delta, ..., t+K\delta\}$. Then in the testing process, we will use only the initial true DNS data $\mathbf{Q}^d$ in time $t + K\delta$ and the next series of predicted DNS data $\hat{\mathbf{Q}}^d$ as the DNS input to generate $\hat{\mathbf{Q}}_{\text{TEL}}$. Finally, we adopts a linear combination to combine the RKTU output $\hat{\mathbf{Q}}_{\text{RKTU}}$ and corresponding TEL output $\hat{\mathbf{Q}}_{\text{TEL}}$ to obtain the final reconstructed output $\hat{\mathbf{Q}}^d$, which can be represented as

$$\hat{\mathbf{Q}}_d^t = w_r^t \hat{\mathbf{Q}}_{\text{RKTU}}^t + w_t^t \hat{\mathbf{Q}}_{\text{TEL}}^t, \tag{8}$$

where w_r^t and w_t^t are trainable parameters. Finally, the reconstructed loss $\mathcal{L}_{\text{recon}}$ can also be represented by Eq. (7).

4.3 Physical Constraints and Refinement

4.3.1 Physical Constraints. To ensure reconstructed flow data following known physical laws, we leverage additional physical constraints to regularize the reconstructed data $\hat{\mathbf{Q}}^d$. In this work, we consider the physical constraints on the mean flow value and the Kinetic Energy for incompressible flows. More details about the flow dataset will be provided in Section 5.1.

Firstly, we aim to ensure that the mean value of reconstructed flow variables is consistent with that of the true flow variables, which is referred to as the equal-mean property. To preserve the equal-mean property of the flow data, we create an equal-mean loss function $\mathcal{L}_{\text{mean}}$ between reconstructed data $\hat{\mathbf{Q}}^d$ and true DNS data $\mathbf{Q}^d$. This loss can be expressed as

$$\mathcal{L}_{\text{mean}} = |\overline{\mathbf{Q}^d} - \overline{\hat{\mathbf{Q}}^d}|, \tag{9}$$

where $\overline{\mathbf{Q}^d}$ and $\overline{\hat{\mathbf{Q}}^d}$ denote the mean value of $\mathbf{Q}^d$ and $\hat{\mathbf{Q}}^d$.

Second, the kinetic energy of turbulent flow $\mathcal{E}$ is known as one of the most important flow characteristics and is defined as

$$\mathcal{E} = \frac{1}{2}(u^2 + v^2 + w^2). \tag{10}$$

To preserve the kinetic energy property of reconstructed flow data $\hat{\mathbf{Q}}^d$, we design a kinetic energy loss $\mathcal{L}_{\text{kinetic}}$, as:

$$\mathcal{L}_{\text{kinetic}} = |\mathcal{E}(\mathbf{Q}^d) - \mathcal{E}(\hat{\mathbf{Q}}^d)|, \tag{11}$$

where $\mathcal{E}(\mathbf{Q}^d)$ and $\mathcal{E}(\hat{\mathbf{Q}}^d)$ denote the kinetic energy of $\mathbf{Q}^d$ and $\hat{\mathbf{Q}}^d$, respectively. Hence, the final loss function $\mathcal{L}$ of the complete CNDE method is:

$$\mathcal{L} = \alpha_0 \mathcal{L}_{\text{recon}} + \alpha_1 \mathcal{L}_{\text{mean}} + \alpha_2 \mathcal{L}_{\text{kinetic}}, \tag{12}$$

where α_0, α_1 and α_2 represent the hyperparameters to control the balance amongst $\mathcal{L}_{\text{recon}}$, $\mathcal{L}_{\text{mean}}$ and $\mathcal{L}_{\text{kinetic}}$.

4.3.2 Degradation-based Refinement. As shown in Fig. 3, we not only preserve the underlying physical rules of flow data in the training process but also employ these two physical constraints in the testing phase as a degradation-based refinement process. The objective is to mitigate accumulated errors and structural distortions over long-term prediction by enforcing the physical consistency. In this refinement process, we include the same set of three components of the loss function: degradation loss $\mathcal{L}_{\text{deg}}$, equal-mean loss $\mathcal{L}^{'}_{\text{mean}}$ and optional kinetic energy loss $\mathcal{L}^{'}_{\text{kinetic}}$ loss. Since we cannot access true DNS data in the testing phase, we adjust the loss functions as described below.

First, because true DNS data are not available in the testing phase, we cannot directly minimize the difference between true DNS data $\mathbf{Q}^d$ and reconstructed data $\hat{\mathbf{Q}}^d$. Thus, in order to protect the overall structure of flow data from the explosion in the long prediction, we introduce a reverse degradation process, by using a separate convolutional network, for mapping reconstructed data $\hat{\mathbf{Q}}^d$ to corresponding low-resolution LES data $\hat{\mathbf{Q}}^l$. We then calculate the loss $\mathcal{L}_{\text{deg}}$ between $\hat{\mathbf{Q}}^l$ and real LES data $\mathbf{Q}^l$, as:

$$\mathcal{L}_{\text{deg}} = \text{MSE}(\hat{\mathbf{Q}}^l, \mathbf{Q}^l). \tag{13}$$

Second, as we do not have access to true DNS in the testing phase, we cannot use the mean value of true DNS as a reference in the equal-mean loss function. Instead, we consider using the mean value of LES data to approximate the mean value of DNS data. Hence, we can directly minimize the equal-mean loss $\mathcal{L}^{'}_{\text{mean}}$ between reconstructed flow data $\hat{\mathbf{Q}}^d$ and true LES data $\mathbf{Q}^l$, which can be described as:

$$\mathcal{L}^{'}_{\text{mean}} = |\overline{\mathbf{Q}^l} - \overline{\hat{\mathbf{Q}}^d}|, \tag{14}$$

Next, the exact kinetic energy of flow data also cannot be computed as true DNS data are not available in the testing phase. However, the kinetic energy is constant for many incompressible flows (e.g., the Forced Isotropic Turbulence dataset [1] used in this work) and thus can be calculated from available DNS training data as $\tilde{\mathcal{E}}$. We then represent the kinetic energy loss as

$$\mathcal{L}^{'}{}_{\text{kinetic}} = |\mathcal{E}(\hat{\mathbf{Q}}^d) - \tilde{\mathcal{E}}|. \tag{15}$$

It is also possible that the kinetic energy changes over time for some flow data, but the dynamics of kinetic energy are often simple (e.g., linear decay over time) and can be approximated by a separate function. We will keep this as future work.

The final refinement loss function is in the same format $\mathcal{L}' = \alpha_0\mathcal{L}_{\text{deg}} + \alpha_1\mathcal{L}'{}_{\text{mean}} + \alpha_2\mathcal{L}'{}_{\text{kinetic}}$. In our test, we adopt $\mathcal{L}^{'}$ to directly adjust the state of reconstructed data for 10 epochs and achieve improved reconstruction performance.

5 EXPERIMENT

In this section, we compare the performance of the proposed methods with the proposed methods in both the Forced Isotropic Turbulence (FIT) dataset [1] and Taylor-Green vortex (TGV) [5] dataset. We first introduce the datasets used in the experiments and the experimental settings. Then we will provide experimental results and analysis.

5.1 Dataset

The Taylor-Green vortex (TGV) [5] is created by the solution of the constant density Navier-Stokes equation:

$$\frac{\partial \mathbf{V}}{\partial t} + (\mathbf{V}.\nabla)\mathbf{V} = \frac{-1}{\rho}\nabla p + \nu\Delta\mathbf{V}, \tag{16}$$

Table 1. Reconstruction performance (measured by SSIM, and Laplacian difference) on (u, v, w) channels by different methods in the FIT dataset. The performance is measured by the average results of the first 10 time steps

Method	SSIM	Laplacian difference
SRCNN	(0.859, 0.851, 0.851)	(0.301, 0.303, 0.303)
RCAN	(0.861, 0.859, 0.859)	(0.299, 0.301, 0.300)
DCS/MS	(0.861, 0.862, 0.862)	0.298, 0.295, 0.294)
SRGAN	(0.862, 0.861, 0.863)	(0.296, 0.294, 0.294)
CTN	(0.881, 0.880, 0.881)	(0.253, 0.254, 0.254)
RKTU	(0.898, 0.899, 0.898)	(0.260, 0.261, 0.259)
CNDEp-E	(0.909, 0.909, 0.907)	(0.244, 0.243, 0.245)
CNDEp-R	(0.904, 0.905, 0.905)	(0.249, 0.248, 0.248)
CNDE-E	(0.945, 0.944, 0.943)	(0.171, 0.172, 0.172)
CNDE-R	(0.943, 0.942, 0.941)	(0.172, 0.174, 0.174)

where $\rho(\mathbf{x}, t)$ and $p(\mathbf{x}, t)$ denote the fluid density and the thermodynamic pressure, respectively. The evolution of the TGV includes the enhancement of vorticity stretching and the consequent production of small-scale eddies. Initially, large vortices are placed in a cubic periodic domain of $[-\pi, \pi]$ (in all three directions), with initial conditions:

$$\begin{aligned} u(x, y, z, 0) &= \sin(x)\cos(y)\cos(z), \\ v(x, y, z, t) &= -\cos(x)\sin(y)\cos(z), \\ w(x, y, z, t) &= 0. \end{aligned} \tag{17}$$

Then the value of the Reynolds number is set to $Re = 1600$. We have both LES and DNS data in several times steps with a time interval of 1s. In each time step, we consider the three components of the velocity along the x, y, and z axis, denoted by u, v, and w, respectively, and also consider the thermodynamic pressure of DNS data denoted by p. In particular, both LES and DNS data are produced along 65 grid points along the z axis under equal intervals. The grid of LES and DNS are 32-by-32 and 128-by-128 grid points, respectively, along the xy directions. Hence, the DNS data are of 4 times higher resolution compared to LES data.

The Forced Isotropic Turbulence (FIT) [1] is also created by the same solution of the constant density Navier-Stokes equation in Eq. 16. This dataset only contains the DNS of forced isotropic turbulence on a 1024-1024-1024 periodic grid, using a pseudo-spectral parallel code. Time integration of the viscous term is done analytically using an integrating scale of 1.364. The simulation is de-aliased using phase-shift and a $2\sqrt{2}\sqrt{3}$ truncation. Energy is injected by keeping constant the total energy in modes such that their wave-number magnitude is less or equal to 2. In particular, this DNS data contains 5024 time steps with time intervals of 0.002s and includes the three components of the velocity (u, v, w) and the thermodynamic pressure p, respectively. The original grid of DNS is 1024-1024-1024. But we use the grid 64-64-64, directly downsampling from the original grid, in our test. Then, we use average filter [20] to convert the DNS data into LES data with the gird 16-16-16. Hence, the DNS data are 4 times the higher resolution compared to LES data. We do not use the loss $\mathcal{L}'_{\text{kinetic}}$ in the refinement process as the kinetic energy varies over time.

5.2 Experimental Settings

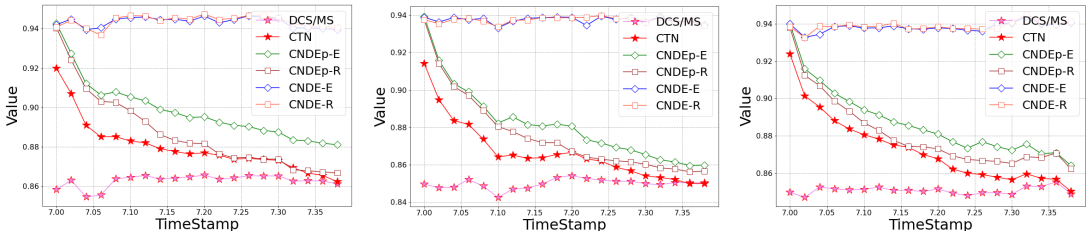
5.2.1 Baselines. We evaluate the performance of the proposed CNDE method with several existing methods that have been widely used for image super-resolution and turbulent flow downscaling. Specifically, we implement our proposed methods: CNDE-E (Enhancing-based TEL Method), CNDE-R (Residual Learning-based TEL Method)[1]. We also implement SRCNN [12], RCAN [43], SRGAN [26],

[1]The source code is at https://drive.google.com/drive/folders/15PhF_q1HcJpXZIvxnR1mMbd8hbkxBbT_?usp=share_link

Table 2. Reconstruction performance (measured by SSIM, and Laplacian difference) on (u, v, w) channels by different methods in the TGV dataset. The performance is measured by the average results of the first 10 time steps

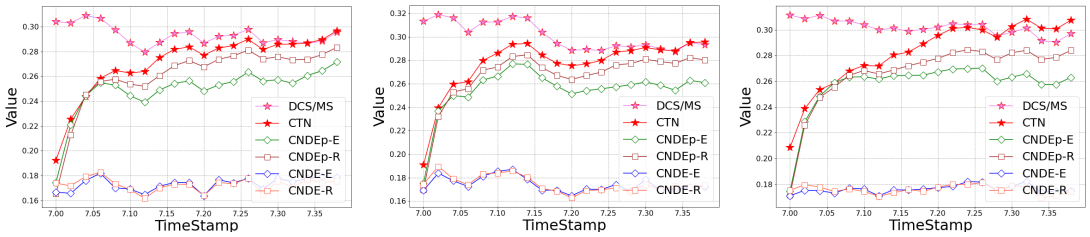
Method	SSIM	Laplacian difference×10
SRCNN	(0.602, 0.603, 0.626)	(0.083, 0.087, 0.079)
RCAN	(0.627, 0.622, 0.631)	(0.073, 0.074, 0.071)
DSC/MS	(0.647, 0.649, 0.649)	0.070, 0.071, 0.065)
SRGAN	(0.661, 0.658, 0.666)	(0.068, 0.067,0.058)
CTN	(0.623, 0.624, 0.627)	(0.093, 0.096, 0.087)
RKTU	(0.708, 0.708, 0.688)	(0.049, 0.046, 0.043)
CNDEp-E	(0.724, 0.723, 0.708)	(0.046, 0.041, 0.039)
CNDEp-R	(0.720, 0.719, 0.701)	(0.046, 0.045, 0.040)
CNDE-E	(0.937, 0.938, 0.941)	(0.017, 0.018, 0.015)
CNDE-R	(0.931, 0.932, 0.936)	(0.017, 0.018, 0.014)

and a popular dynamic fluid downscaling method, DCS/MS [14], as baselines. To better verify the effectiveness of each proposed component, i.e., RKTU, TEL, and the degradation-based refinement approach, we further implement three extra baselines: Convolutional Transition Network (CTN), RKTU, CNDEp-E, and CNDEp-R. In particular, CTN is created by combining SRCNN and LSTM [18]. CNDEp-based methods are created by removing the refinement process from the CNDE method. We aim to show the advantage of the RKTU in spatio-temporal DNS reconstruction by comparing CTN with the RKTU. And CNDEp-based methods are created for demonstrating the effectiveness of the TEL structure and refinement process.



(a) u Channel. (b) v Channel. (c) w Channel.

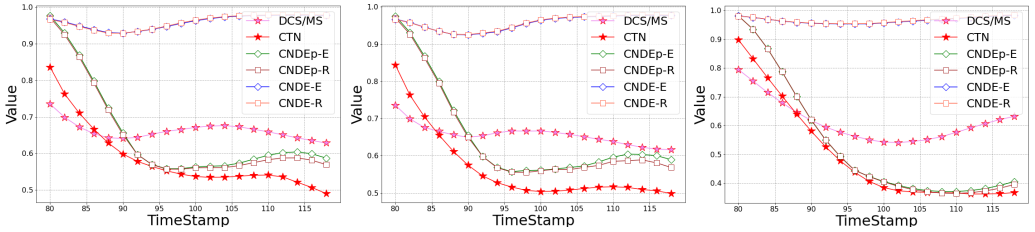
Fig. 4. Change of SSIM values produced by different models from 1st (7s) to 20th (7.4s) time step in the FIT dataset.



(a) u Channel. (b) v Channel. (c) w Channel.

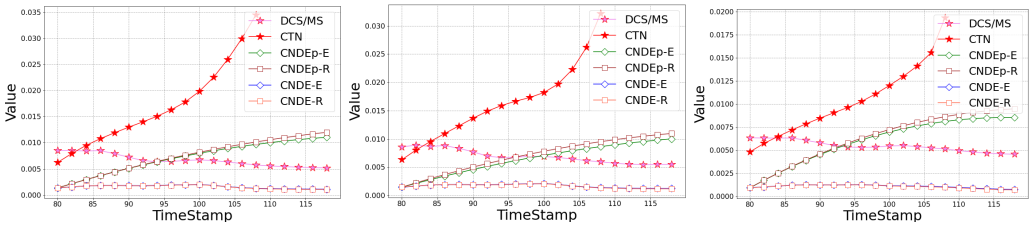
Fig. 5. Change of Laplacian difference by different models from 1st (7s) to 20th (7.4s) time step in the FIT dataset.

5.2.2 Experimental Designs. We test the proposed methods and baselines in both the FIT and the TGV datasets. We train the models using the FIT data from a consecutive 1 second period with a



(a) u Channel. (b) v Channel. (c) w Channel.

Fig. 6. Change of SSIM values produced by different models from 1st (80s) to 20th (120s) time step in the TGV dataset.



(a) u Channel. (b) v Channel. (c) w Channel.

Fig. 7. Change of Laplacian difference produced by different models from 1st (80s) to 20th (120s) time step in the TGV dataset.

time interval $\delta = 0.02s$ (the original was 0.002s), total 50 time steps, and then apply the trained model into the next 0.4 second period, total 20 time steps, for performance evaluation. For the TGV dataset, we use a consecutive 40-second period with a time interval $\delta = 2s$ (the original was 1s) for training and the next 40 seconds of data for testing.

We evaluate the performance of DNS reconstruction using two different metrics, structural similarity index measure (SSIM) [37], and Laplacian [40]. SSIM is used to appraise the similarity between reconstructed data and target DNS on three aspects, luminance, contrast, and overall structure. The higher value of SSIM indicates better reconstruction performance. The Laplacian operator is used to assess the performance of capturing the flow gradients. The Laplacian of each of the three components of the velocity vector (x, y, and z directions) will be evaluated. The Laplacian operator of $\mathbf{Q}$ is:

$$\text{Lap} = (\frac{\partial \mathbf{Q}}{\partial x})^2 + (\frac{\partial \mathbf{Q}}{\partial y})^2 + (\frac{\partial \mathbf{Q}}{\partial z})^2. \tag{18}$$

we use Lap to measure the difference in flow gradient between true DNS and generated data. The Laplacian difference metric can be represented as $|\text{Lap}(\mathbf{Q}^d) - \text{Lap}(\hat{\mathbf{Q}}^d)|$. The lower value of the Laplacian difference indicates better performance.

5.2.3 Environmental Settings and Implementation Details. We implement our proposed method using Tensorflow 2.x with a GTX3080 GPU. The model is firstly trained in 500 epochs with ADAM optimizer [24] from an initial learning rate of 0.001. In the refinement step, we lower the learning rate to 0.0005 and only iterate the training by 10 epochs. All the hidden variables and gating variables are in 32 dimensions. The values of α_0, α_1, and α_2 are set as 1, 0.1, and 0.1, respectively.

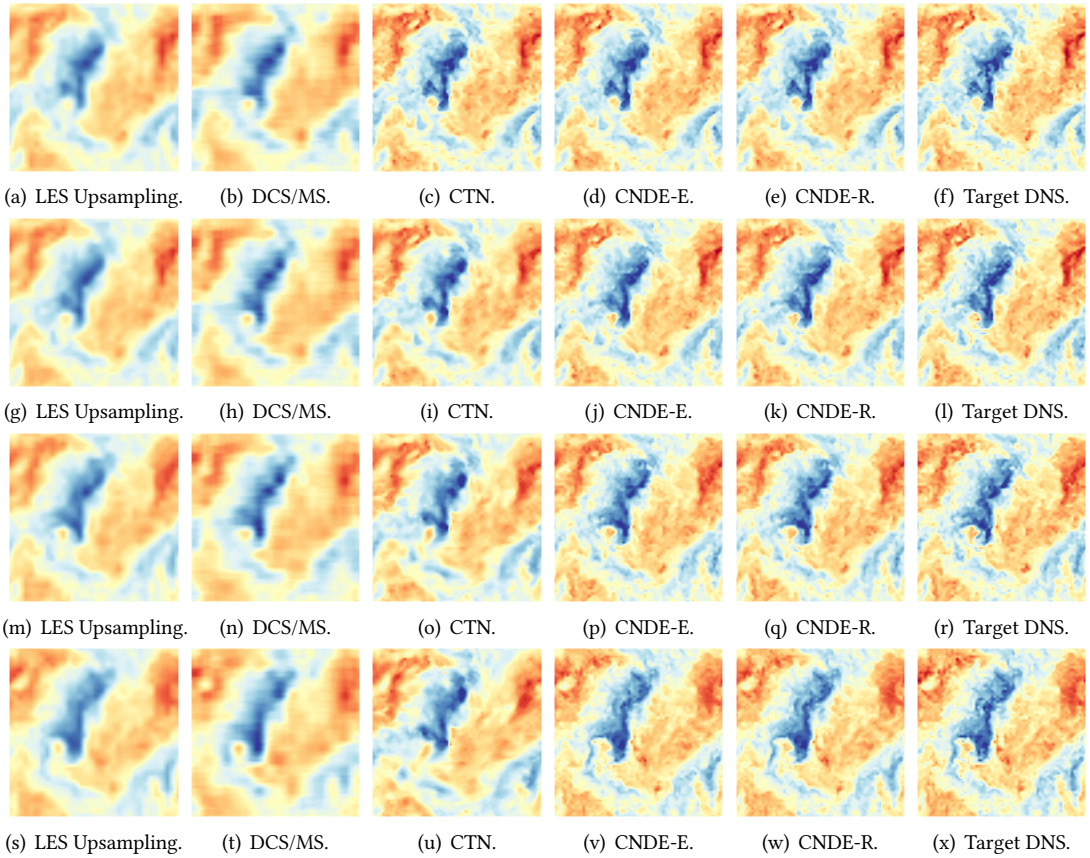


Fig. 8. Reconstructed w channel by each method on a sample testing slice along the z dimension in the FIT dataset. We show the reconstruction results at 1st (7s), 5th (7.1s) 10th (7.2s) and 20th (7.4s) in (a)-(f), (g)-(l), (m)-(r) and (s)-(x), respectively.

5.3 Reconstruction Performance

Quantitative Results. We compare our proposed CNDE-based methods with baseline methods. Table 1 and Table 2 summarize the average performance over the first 10 time steps in the testing phase on both the FIT dataset and the TGV dataset. Compared with baselines, CNDE-based methods perform the best in both evaluation ways, obtaining the highest SSIM value and lowest Laplacian difference. According to Table 1 and Table 2, the CNDE-based methods in general outperform other baselines for velocity components $\{u, v, w\}$. We also have several observations from these tables: (1) When comparing the CNDE-based methods with SR baselines and DCS/MS model, we observe that these baseline methods cannot recover turbulent flow well and get the worse performance in terms of SSIM and Laplacian difference. (2) Compared with the SRCNN, the CTN, which uses the LSTM model, shows a significant improvement in both evaluation metrics. It confirms the effectiveness of a temporal model (e.g, LSTM) in capturing temporal dependency. (3) When we see the comparison amongst CTN, RKTU, CNDEp-based methods, and CNDE-based methods, we can see significant improvements by incorporating each component (RKTU, TEL, and refinement) of the proposed method. In particular, the refinement method brings the most significant improvement in terms of SSIM and Laplacian difference.

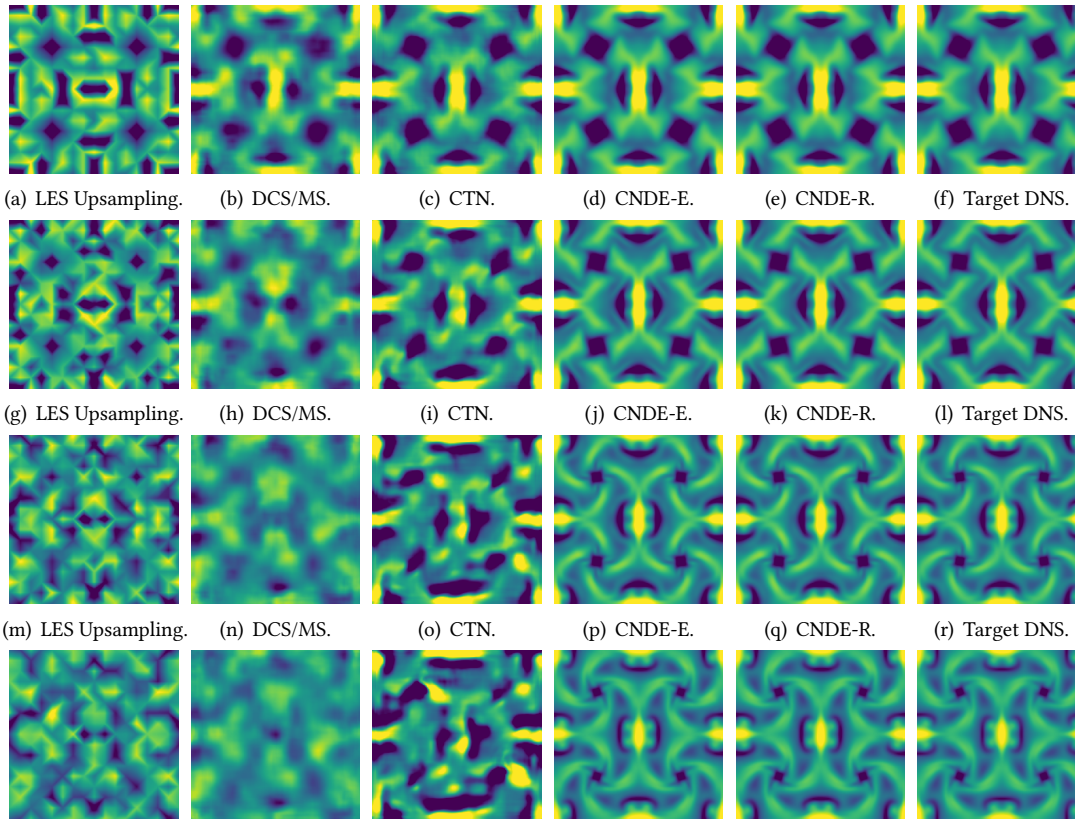


Fig. 9. Reconstructed w channel by each method on a sample testing slice along the z dimension in the TGV dataset. We show the reconstruction results at 1st (80s), 5th (90s), and 10th (100s) in (a)-(f), (g)-(l), and (m)-(r), respectively.

Temporal Analysis. In the temporal analysis from the FIT dataset, we show the change in performance as we reconstruct DNS data over 20 time steps after the training data. The performance change using SSIM and Laplacian difference is shown in Fig. 4 and Fig. 5, respectively. Several observations are highlighted from these results: (1) With larger time intervals between training data and prediction data, the performance becomes worse. In general, our CNDE-based methods are more stable over a long prediction period and show much better performance than other baselines. (2) We also observe that the temporal model (e.g. LSTM) can bring significant improvement in long-term prediction by comparing DCS/MS with CTN. (3) By comparing CNDEp-based methods and CNDE-based methods, the CNDE-based methods using refinement outperform the CNDEp-based methods without using this adjustment over time. It demonstrates that the refinement method can adjust the state bias in the long-term prediction of flow data. (4) We also find that CNDEp-based methods achieve better performance after the 5th time steps compared with the temporal baseline CTN model. It proves the advantage of the proposed RKTU structure in the long-term prediction. (5) By comparing two versions of CNDEp-based methods, we observe that CNDEp-E slightly outperforms the CNDEp-R in the long-term prediction. We can obtain a similar conclusion from

two versions of CNDE-based methods. We will investigate the selection of temporal enhancing methods in future works.

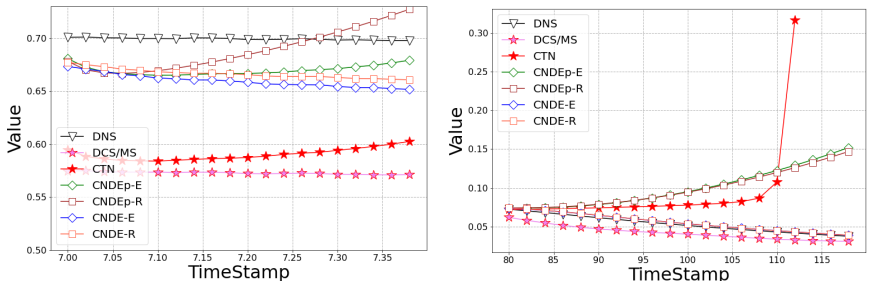
In Fig. 6 and Fig. 7, we present the performance of temporal prediction on the TGV dataset. It also shows that our proposed method can achieve a better performance in terms of SSIM and Laplacian difference. Some observations are highlighted: (1) By comparing DCS/MS, CNDEp-based methods, and CNDE-based methods, the CNDE-based methods using refinement perform much better than CNDEp-based methods and DCS/MS. Moreover, the performance of the CNDEp-based methods becomes worse than the baseline DCS/MS after the 5th time step. This is because the TGV data varies dynamically in large time intervals (δ is 2s) and the testing data are very different from the initial data point, which causes the CNDEp-based methods to fail to capture the correct flow dynamic without refinement. It also proves the advantages of the refinement method for adjusting the state of flow data in the long-term prediction. (2) The CTN almost fails to capture the flow dynamics after the 5th time step, which indicates that the CTN is not suitable for the TGV dataset.

Visual Results. In Fig. 8, we show the reconstructed data at multiple time steps (1st, 5th, 10th, and 20th time steps) after the training period. For each time step, we only show the slice of the w component at a specified z value. At the 1st step, both the CNDE-based methods and the baseline CTN model can obtain ideal reconstruction results. This is because the test data are similar to the training data At the last time step. Second, although the baseline DSC/MS [14] was developed in the context of turbulent flow downscaling, it does not work well in capturing the complex pattern of DNS data of the FIT dataset. The reconstructed image is more similar to LES data.

Beginning at the 5th time step, our proposed CNDE-based methods perform better than baselines. We can see a more significant difference at the 20th time step. All the baselines almost fail to capture the correct pattern of flow data. In contrast, our proposed CNDE-based methods can still effectively accurately fine-level capture textures and patterns, reduce color amplitude difference, and thus achieve much better performance. These observations demonstrate the advantages of our proposed method in long-term prediction.

Similar results on the TGV dataset are presented in Fig. 9, and we have similar observations. These results demonstrate that our proposed method can achieve a better reconstruction performance in prediction over time.

Validation based on Physical Metrics. We also compare the performance of long-term prediction in terms of kinetic energy, which is considered an important metric for studying the property of turbulence. In Fig. 10, we show the kinetic energy of the target DNS and the kinetic energy of reconstructed flow data by baselines and the proposed CNDE-based methods in both the FIT and the TGV datasets. Here are some observations from Fig. 10(a) provided by the FIT dataset: (1) The CNDE-based methods in general can perform much better than baseline method DCS/MS and CTN. The change of kinetic energy from CNDE-based methods is close to the target DNS. Even without using the refinement process, the CNDEp-based methods still outperform DCS/MS and CTN models. It shows that our proposed method can follow the underlying physical rule well in the long-term prediction. This is because our method is designed based on underlying PDEs. We also incorporate the underlying physical constraints to adjust the state of flow data. All of these designs bring the improvement for the CNDE-based methods to accurately capture the underlying dynamics of physical variables. (2) CNDEp-based methods perform worse and worse after the 8th time step. This is because the accumulated error gets amplified in every time step, leading to an error explosion. It also justifies the effectiveness of using the refinement process for adjusting the bias of reconstructed data. We present the results on the TGV dataset in Fig. 10(b), from which we also draw similar conclusions.



(a) FIT data. (b) TGV data.

Fig. 10. Change of Kinetic Energy produced by the reference DNS and different models in both the FIT and the TGV datasets, respectively.

6 CONCLUSION

We propose a novel method to reconstruct high-resolution flow data in spatial and temporal fields. The proposed method has several components. The RKTU structure leverages the physical knowledge embodied in the Navier-Stokes equation to capture the spatial resolution and temporal dynamic of turbulent flow from frequent LES data. We also use TEL to capture long-term temporal dynamics. We further develop a degradation-based refinement method to adjust the reconstructed data over time by enforcing the consistency with LES data and known physical constraints. We have demonstrated the superiority of the proposed method in the spatio-temporal reconstruction of turbulent flow. More importantly, the proposed CNDE method can be easily applicable to many scientific problems with similar properties, e.g., complex temporal dynamics, and high computational cost for creating high-resolution and high-fidelity simulations. The components in CNDE, e.g., RKTU and the refinement method, can also be used as a building block to enhance existing deep learning models for modeling complex dynamics with the guidance of known governing PDEs. For example, simulations of cloud-resolving models (CRM) at sub-kilometer horizontal resolution are critical for effectively representing boundary-layer eddies and low clouds. However, it is not feasible to generate simulations at such fine resolution even with the most powerful commuters expected to be available in the near future. Hence, the method developed in this paper can provide great potential for reconstructing high-resolution simulations.

ACKNOWLEDGMENTS

The material presented in this paper is based upon work supported by the National Science Foundation (NSF) through Grant OAC-2203581. Computational resources are provided, in part, by the University of Pittsburgh Center for Research Computing (CRC).

REFERENCES

[1] 2019. Forced Isotropic Turbulence Dataset (Extended). https://doi.org/10.7281/T1KK98XB.

[2] Namhyuk Ahn, Byungkon Kang, and Kyung-Ah Sohn. 2018. Fast, Accurate, and Lightweight Super-Resolution with Cascading Residual Network. arXiv:1803.08664 [cs.CV]

[3] Saad Albawi, Tareq Abed Mohammed, and Saad Al-Zawi. 2017. Understanding of a convolutional neural network. In *2017 international conference on engineering and technology (ICET)*. Ieee, 1–6.

[4] Tianshu Bao, Shengyu Chen, Taylor T Johnson, Peyman Givi, Shervin Sammak, and Xiaowei Jia. 2022. Physics Guided Neural Networks for Spatio-temporal Super-resolution of Turbulent Flows. In *The 38th Conference on Uncertainty in Artificial Intelligence*.

[5] Marc E Brachet, D Meiron, S Orszag, B Nickel, R Morf, and Uriel Frisch. 1984. The Taylor-Green vortex and fully developed turbulence. *Journal of Statistical Physics* 34, 5 (1984), 1049–1063.

[6] John Butcher. 2007. Runge-kutta methods. *Scholarpedia* 2, 9 (2007), 3147.

[7] Shengyu Chen, Shervin Sammak, Peyman Givi, Joseph P Yurko, and Xiaowei Jia. 2021. Reconstructing High-resolution Turbulent Flows Using Physics-Guided Neural Networks. *arXiv preprint arXiv:2109.03327* (2021).

[8] Yu Chen, Ying Tai, Xiaoming Liu, Chunhua Shen, and Jian Yang. 2017. FSRNet: End-to-End Learning Face Super-Resolution with Facial Priors. arXiv:1711.10703 [cs.CV]

[9] Wenlong Cheng, Mingbo Zhao, Zhiling Ye, and Shuhang Gu. 2021. MFAGAN: A Compression Framework for Memory-Efficient On-Device Super-Resolution GAN. arXiv:2107.12679 [cs.AR]

[10] Tao Dai, Jianrui Cai, Yongbing Zhang, Shu-Tao Xia, and Lei Zhang. 2019. Second-Order Attention Network for Single Image Super-Resolution. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 11057–11066. https://doi.org/10.1109/CVPR.2019.01132

[11] Zhiwen Deng, Chuangxin He, Yingzheng Liu, and Kyung Chun Kim. 2019. Super-resolution reconstruction of turbulent velocity fields using a generative adversarial network-based artificial intelligence framework. *Physics of Fluids* 31, 12 (2019), 125111.

[12] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. 2014. Learning a deep convolutional network for image super-resolution. In *European conference on computer vision*. Springer, 184–199.

[13] Vinh Van Duong, Thuc Nguyen Huu, Jonghoon Yim, and Byeungwoo Jeon. 2021. A Fast and Efficient Super-Resolution Network Using Hierarchical Dense Residual Learning. In *2021 IEEE International Conference on Image Processing (ICIP)*. 1809–1813. https://doi.org/10.1109/ICIP42928.2021.9506786

[14] Kai Fukami, Koji Fukagata, and Kunihiko Taira. 2019. Super-resolution reconstruction of turbulent flows with machine learning. *Journal of Fluid Mechanics* 870 (2019), 106–120.

[15] Kai Fukami, Koji Fukagata, and Kunihiko Taira. 2020. Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows. *Journal of Fluid Mechanics* 909 (Dec 2020). https://doi.org/10.1017/jfm.2020.948

[16] P. Givi. 1994. Spectral and Random Vortex Methods in Turbulent Reacting Flows. In *Turbulent Reacting Flows*, P. A. Libby and F. A. Williams (Eds.). Academic Press, London, England, Chapter 8, 475–572.

[17] Paul C Hanson et al. 2020. Predicting lake surface water phosphorus dynamics using process-guided machine learning. *Ecological Modelling* 430 (2020), 109136.

[18] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[19] Xiaowei Jia, Jared Willard, Anuj Karpatne, Jordan Reed, Jacob Zwart, Michael Steinbach, and Vipin Kumar. 2019. Physics Guided RNNs for Modeling Dynamical Systems: A Case Study in Simulating Lake Temperature Profiles. In *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM.

[20] Cai Jing, Yang Jinsheng, and Ding Runtao. 2000. Fuzzy weighted average filter. In *WCC 2000-ICSP 2000. 2000 5th International Conference on Signal Processing Proceedings. 16th World Computer Congress 2000*, Vol. 1. IEEE, 525–528.

[21] Anuj Karpatne, William Watkins, Jordan Read, and Vipin Kumar. 2017. Physics-guided Neural Networks (PGNN): An Application in Lake Temperature Modeling. *arXiv preprint arXiv:1710.11431* (2017).

[22] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. 2018. Progressive Growing of GANs for Improved Quality, Stability, and Variation. arXiv:1710.10196 [cs.NE]

[23] Ankush Khandelwal, Shaoming Xu, Xiang Li, Xiaowei Jia, Michael Stienbach, Christopher Duffy, John Nieber, and Vipin Kumar. 2020. Physics guided machine learning methods for hydrology. *arXiv preprint arXiv:2012.02854* (2020).

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

[25] Colin Lea, Rene Vidal, Austin Reiter, and Gregory D Hager. 2016. Temporal convolutional networks: A unified approach to action segmentation. In *European conference on computer vision*. Springer, 47–54.

[26] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. 2017. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4681–4690.

[27] Bo Liu, Jiupeng Tang, Haibo Huang, and Xi-Yun Lu. 2020. Deep learning methods for super-resolution reconstruction of turbulent flows. *Physics of Fluids* 32, 2 (2020), 025105.

[28] Nikhil Muralidhar, Jie Bu, Ze Cao, Long He, Naren Ramakrishnan, Danesh Tafti, and Anuj Karpatne. 2020. PhyNet: Physics Guided Neural Networks for Particle Drag Force Prediction in Assembly. In *Proceedings of the 2020 SIAM International Conference on Data Mining*. SIAM, 559–567.

[29] A. G. Nouri, M. B. Nik, P. Givi, D. Livescu, and S. B. Pope. 2017. Self-Contained Filtered Density Function. *Physical Review Fluids* 2 (Sep 2017), 094603. Issue 9. https://doi.org/10.1103/PhysRevFluids.2.094603

[30] Sung Cheol Park, Min Kyu Park, and Moon Gi Kang. 2003. Super-resolution image reconstruction: a technical overview. *IEEE Signal Processing Magazine* 20, 3 (2003), 21–36.

[31] Jordan S Read, Xiaowei Jia, Jared Willard, Alison P Appling, Jacob A Zwart, Samantha K Oliver, Anuj Karpatne, Gretchen JA Hansen, Paul C Hanson, William Watkins, et al. 2019. Process-guided deep learning predictions of lake

water temperature. *Water Resources Research* 55, 11 (2019), 9173–9190.
[32] P. Sagaut. 2005. *Large Eddy Simulation for Incompressible Flows*. Springer-Verlag, New York, NY.
[33] Ying Tai, Jian Yang, and Xiaoming Liu. 2017. Image Super-Resolution via Deep Recursive Residual Network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2790–2798. https://doi.org/10.1109/CVPR.2017.298
[34] Uddeshya Upadhyay and Suyash P. Awate. 2019. Robust Super-Resolution Gan, with Manifold-Based and Perception Loss. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*. 1372–1376. https://doi.org/10.1109/ISBI.2019.8759375
[35] Xintao Wang, Ke Yu, Chao Dong, and Chen Change Loy. 2018. Recovering Realistic Texture in Image Super-resolution by Deep Spatial Feature Transform. arXiv:1804.02815 [cs.CV]
[36] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. 2018. Esrgan: Enhanced super-resolution generative adversarial networks. In *ECCV Workshops*.
[37] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.
[38] Zhang Wenlong, Liu Yihao, Chao Dong, and Yu Qiao. 2021. RankSRGAN: Generative Adversarial Networks with Ranker for Image Super-Resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021), 1–1. https://doi.org/10.1109/TPAMI.2021.3096327
[39] Wikipedia contributors. 2022. Finite difference method — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Finite_difference_method&oldid=1126400243 [Online; accessed 25-January-2023].
[40] Wikipedia contributors. 2022. Laplace operator — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Laplace_operator&oldid=1127277109 [Online; accessed 23-January-2023].
[41] Jared Willard, Xiaowei Jia, Shaoming Xu, Michael Steinbach, and Vipin Kumar. 2020. Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919* (2020).
[42] You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. 2018. tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–15.
[43] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. 2018. Image super-resolution using very deep residual channel attention networks. In *Proceedings of the European conference on computer vision (ECCV)*. 286–301.
[44] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. 2018. Residual Dense Network for Image Super-Resolution. arXiv:1802.08797 [cs.CV]