Invited Articles

Data assimilation empowered neural network parametrizations for subgrid processes in geophysical flows

Suraj Pawar and Omer San*
School of Mechanical & Aerospace Engineering, Oklahoma State University, Stillwater, Oklahoma 74078, USA

(Received 15 June 2020; accepted 16 November 2020; published 12 May 2021)

In the past couple of years, there has been a proliferation in the use of machine learning approaches to represent subgrid-scale processes in geophysical flows with an aim to improve the forecasting capability and to accelerate numerical simulations of these flows. Despite its success for different types of flow, the online deployment of a data-driven closure model can cause instabilities and biases in modeling the overall effect of subgrid-scale processes, which in turn leads to inaccurate prediction. To tackle this issue, we exploit the data assimilation technique to correct the physics-based model coupled with the neural network as a surrogate for unresolved flow dynamics in multiscale systems. In particular, we use a set of neural network architectures to learn the correlation between resolved flow variables and the parametrizations of unresolved flow dynamics and formulate a data assimilation approach to correct the hybrid model during their online deployment. We illustrate our framework in a set of applications of the multiscale Lorenz 96 system for which the parametrization model for unresolved scales is exactly known, and the two-dimensional Kraichnan turbulence system for which the parametrization model for unresolved scales is not known *a priori*. Our analysis, therefore, comprises a predictive dynamical core empowered by (i) a data-driven closure model for subgrid-scale processes, (ii) a data assimilation approach for forecast error correction, and (iii) both data-driven closure and data assimilation procedures. We show significant improvement in the long-term prediction of the underlying chaotic dynamics with our framework compared to using only neural network parametrizations for future prediction. Moreover, we demonstrate that these data-driven parametrization models can handle the non-Gaussian statistics of subgrid-scale processes, and effectively improve the accuracy of outer data assimilation workflow loops in a modular nonintrusive way.

DOI: 10.1103/PhysRevFluids.6.050501

I. INTRODUCTION

Geophysical flows are characterized by the multiscale nature of flows where there is a massive difference between the largest and smallest scales, and these scales interact with each other to exchange heat, momentum, and moisture. This makes the numerical simulations of geophysical flows in which every flow feature is resolved computationally unmanageable, even though the physical laws governing these processes are well known. Therefore, the atmosphere and ocean models compute the approximate numerical solution on the computational grid that consists of $O(10^7)$ to $O(10^8)$ grids with a spacing of $O(10\,\text{km})$ to $O(100\,\text{km})$. The effect of unresolved scales is taken into account by using several parametrization schemes, which represent the dynamics of subgrid-scale processes as a function of resolved dynamics [1–3]. However, the weather projection

*osan@okstate.edu

is marred by large uncertainties in the parameters of these parametrization schemes, and also due to incorrect structure of these parametrizations equations itself [4–6].

Typically, the parameters of these parametrization schemes are estimated by the model tuning process based on the observations from experimental and field measurements or the data generated from high-resolution numerical simulations [7,8]. The nonlinear and multiscale nature of geophysical flows makes this tuning procedure cumbersome and can impede accurate climate prediction [9]. A recent development in machine learning, particularly deep learning [10], along with the huge volume of data gathered from high-resolution numerical simulations [11] and remote sensors measurements [12] offers an alternative to the physics-based parametrization schemes and can pave the way for improved climate and weather models. Deep-learning approaches have been demonstrated to be successful for different scientific tasks in Earth system science, such as extreme weather pattern detection [13], precipitation nowcasting [14], transport process modeling [15], and many more. Deep learning has also been utilized to represent subgrid-scale processes in climate models. Rasp, Pritchard, and Gentine [16] trained a deep neural network (DNN) to emulate a cloud resolving model and formulated a procedure to produce stable results in the online deployments close to the original super-parameterized global circulation model. Gentine *et al.* [17] used an ensemble of random forests as a machine learning (ML) algorithm to parametrize the moist convection and implemented it in a global circulation model. They demonstrated the stable and robust performance of ML-based parametrization in capturing important climate statistics including precipitation extremes.

Along with the Earth system science, there is a surge in the application of machine learning for fluid mechanics. Readers are directed to an excellent review by Brunton, Noack, and Koumoutsakos [18] on how ML algorithms are being used for augmenting the domain knowledge, automating tasks such as flow-control and optimization by the fluid mechanics' community. In a recent perspective, Brenner, Eldredge, and Freund [19] discuss the strength and limitations of ML-based algorithms to advance fluid mechanics. The closure problem in turbulence modeling is similar to the parametrization in climate modeling and is encountered in Reynolds-Averaged Navier-Stokes (RANS) and large eddy simulation (LES) which are widely adopted for engineering flow simulations. There have been several studies that use ML algorithms to address the turbulence closure problem [20–23]. Ling, Kurzawski, and Templeton [24] proposed a novel neural network architecture with embedded Galilean invariance for the prediction of Reynolds stress anisotropy tensor. Wang, Wu, and Xiao [25] employed random forest as an ML algorithm to reconstruct the discrepancy in RANS-modeled Reynolds stresses and evaluated its performance for fully developed turbulent flows and separated flows. Deep learning has also been utilized for LES of turbulent flows, for example, subgrid-scale closure modeling of Kraichnan turbulence [26], decaying homogeneous isotropic turbulence [27], forced isotropic turbulence [28], compressible isotropic turbulence [29], and wall-bounded turbulence [30]. The feasibility of deep learning has been investigated to produce a predictive model for turbulent fluxes, such as heat fluxes [31] and anomalous fluxes in drift-wave turbulence [32]. In a recent work, Novati, de Laroussilhe, and Koumoutsakos [33] introduced a multi-agent reinforcement learning framework as an automated discovery tool for turbulence models and applied it to forced homogeneous isotropic turbulence. Besides turbulence closure modeling, deep learning has been proved to be very successful for challenging problems such as super-resolution of turbulent flows [34–36], data-driven modeling of chaotic systems [37–39], reduced-order modeling of high-dimensional multiphysics systems [40–43], and developing forecast models for complex physical systems [44–46].

Despite the development of deep-learning algorithms as a powerful tool to extract spatio-temporal patterns from the data, these methods are criticized for their black-box nature and are prone to produce physically inconsistent results due to their lack of generalizability [47,48]. Moreover, the increase in spatial and temporal dimensionalities raises a computational challenge in terms of the training. Hence, it is essential to integrate machine learning with physics-based modeling to address the challenge of interpretability, physical consistency, and computational burden [49]. One way to combine machine learning with physics-based modeling is by incorporating physical

conservation laws into training through a regularization term added to the loss function of a neural network [36,50–53]. Another way is to change the structure of neural network architecture to enforce physical conservation laws as hard constraints [54,55]. The hybrid modeling in which a submodel within the physics-based model is replaced by machine learning methods is another approach to address the limitation of pure data-driven methods [15,49,56]. One of the issues with hybrid models is that the trained neural network often suffers from instability once they are deployed in the forward model. For example, a small change in the training dataset or the input and output vector of the neural network led to unpredictable blow-ups in the global circulation model that employs a neural network to emulate cloud resolving model [16,57]. Similarly, Brenowitz and Bretherton [58] found that the nonphysical correlations learned by neural networks were the cause of instabilities in their online deployment within the global circulation model [59] and developed an approach to ensure stability. Wu *et al.* [60] highlighted the gap between *a priori* and *a posteriori* performance of data-driven Reynolds stress closure models as the RANS equations with such model can be ill-conditioned. Therefore, even though data-driven turbulence closure models predicted better closure terms, their online deployment does not lead to significant improvement in the mean velocity field prediction [22,25]. Wu *et al.* [60] proposed a metric to evaluate the conditioning of RANS equations in the *a priori* settings and showed that the implicit treatment of Reynolds stresses leads to reduced error in mean velocity prediction.

Data assimilation (DA) is a well-established discipline where observations are blended with the model to take uncertainties into account for improving the numerical prediction of the system [61–66] and can be applied to achieve accurate prediction in hybrid models that employ data-driven model as a submodel for some processes (for example subgrid-scale processes). DA tools are being extensively utilized in geoscience and numerical weather forecast centers to correct background predictions based on a combination of heterogeneous measurement data coming from ground observations and satellite remote-sensing. These techniques have been also investigated recently for integrating experimental data into large-eddy simulations of engineering flows [67–70]. In a DA workflow, we merge forward model predictions with observational data. However, it has been often remarked that no-model is correct but some of them are useful. In typical DA studies and twin experiments, therefore, the subgrid-scale processes have been modeled as Gaussian noise due to a lack of structural information on their mechanisms. If we would know their dynamics either structurally or functionally, for sure it would be wise to include them in the model before a DA analysis is executed. However, the subgrid-scale processes in turbulent flows often cannot be accurately modeled by Gaussian noise, and ML methodologies can be adopted to get a grip on subgrid-scale processes. Hence, we put forth a neural network-based statistical learning approach to improve model uncertainty and incorporate this information as a data-driven closure term to the forward model. We examine how the forecast error reduces due by including ML-based closure term to the underlying forward model. Indeed, the integration of DA with ML methodologies holds immense potential in various fields of physical science [71–75] and we demonstrate this through our study.

In this work, we propose a neural network closure framework in developing hybrid physics-ML models supplemented with DA for multiscale systems. In particular, we advocate the use of sequential DA techniques to improve the state estimate of the system by incorporating observations into a model equipped with neural network parametrization schemes for unresolved physics. To this end, we use real-time measurements to regularize ML empowered predictive tools through ensemble Kalman filter-based approach. Our first example is a two-level Lorenz 96 model [76] for our numerical experiments since it generates a controllable test case for advancing turbulence parametrization theories, especially in the age of data-driven models. The Lorenz 96 is an idealized model of atmospheric circulation and is used widely to test research ideas [77–80]. Even though the dynamics of both large and small scales are known exactly for a two-level Lorenz 96 model, it is very difficult to predict it because of the strong interplay between fast and slow subsystems. Therefore, we select this multiscale model for the assessments of data-driven closures for capturing the physics of subgrid scales. Since we use an "explicit" evolution equation for the closure parametrizations,

we can easily assess the data-driven models in *a posteriori* simulations. This often comprises a challenging task in LES computations since the low-pass filtering operation is "implicitly" applied to the governing equations. We further extend our framework to Kraichnan turbulence [81], where it is shown that the DA improves the state estimate of the hybrid physics-ML model and this leads to better prediction for statistical properties like kinetic energy spectra and vorticity structure functions in comparison with high-fidelity direct numerical simulation (DNS). Our approach is multifaceted in at least two ways. We first show that the infusion of the DA approaches improves the forecasting quality of predictive models equipped with data-driven parametrizations. Second, we also demonstrate that the data-driven parametrizations help significantly to reduce forecast errors in DA workflows. Therefore, our modular framework can be considered as a way to incorporate real-time observations that are prevalent in today's weather forecast station into hybrid models constituted from a physics-based model as the dynamical core of the system, and a data-driven model to describe unresolved physics.

The paper is structured as follows. In Sec. II, we discuss the problem of parametrizations using a two-level Lorenz 96 model and Kraichnan turbulence as a prototypical examples. Section III details two types of neural network utilized in this study for learning the mapping between resolved variables and parametrizations of unresolved scales. We explain the methodology of sequential data assimilation and ensemble Kalman filter-based algorithms in Sec. IV. In Sec. V, we discuss the findings of our numerical experiments with a two-level Lorenz 96 model and Kraichnan turbulence. Finally, we conclude with the summary and direction for future work in Sec. VI.

II. PARAMETERIZATIONS IN MULTISCALE SYSTEMS

A. Two-level Lorenz 96 model

In this section, we describe the two-level variant of the Lorenz 96 model proposed by Lorenz [76]. This model has been extensively investigated to study stochastic parametrization schemes [82–84], scale-adaptive parametrizations [85], and neural network parametrizations [57,86]. The two-level Lorenz 96 model can be written as

$$\frac{dX_i}{dt} = -X_{i-1}(X_{i-2} - X_{i+1}) - X_i - \frac{hc}{b}\sum_{j=1}^{J} Y_{j,i} + F, \tag{1}$$

$$\frac{dY_{j,i}}{dt} = -cbY_{j+1,i}(Y_{j+2,i} - Y_{j-1,i}) - cY_{j,i} + \frac{hc}{b}X_i, \tag{2}$$

where Eq. (1) represents the evolution of slow, high-amplitude variables X_i $(i = 1, \ldots, n)$, and Eq. (2) provides the evolution of a coupled fast, low-amplitude variable $Y_{j,i}$ $(j = 1, \ldots, J)$. We use $n = 36$ and $J = 10$ in our computational experiments. We utilize $c = 10$ and $b = 10$, which implies that the small scales fluctuate 10 times faster than the larger scales. Also, the coupling coefficient h between two scales is equal to 1 and the forcing is set at $F = 10$ to make both variables exhibit the chaotic behavior. The boundary conditions for the slow and fast variables are detained in Sec. V along with the generation of initial condition for the two-level Lorenz 96 system.

In parametrization research, small scale variables are not resolved and their effect is typically parameterized as a function of resolved large-scale variables. A forecast model for the resolved variables given in Eq. (1) can be constructed with the parametrization for unresolved variables as follows:

$$\frac{d\widetilde{X}_i}{dt} = -\widetilde{X}_{i-1}(\widetilde{X}_{i-2} - \widetilde{X}_{i+1}) - \widetilde{X}_i - \frac{hc}{b}G_i + F, \tag{3}$$

where the tilde is used to denote the fact that the parametrization G_i is used to represent the effect of unresolved variables. Typically, the parametrizations is a function of resolved variables and can

be written mathematically as

$$\sum_{j=1}^{J} Y_{j,i} :\approx G_i = \mathbf{N}(\widetilde{\mathbf{X}}), \tag{4}$$

where $\mathbf{N}(\cdot)$ is the nonlinear mapping of resolved variables to the parametrizations at the ith grid point. This mapping can be based on certain physical arguments or can also be learned with any data-driven methods. Therefore in parametrization research for multiscale systems, the underlying physical laws governing the dynamics of resolved variables are assumed to be known exactly, and the effect of unresolved variables is considered through parametrizations G_i. If we use data-driven methods to represent the parametrization G_i, then the forecast model given in Eq. (3) can be considered as a hybrid model. Our main objective in this work is to improve the forecasting capability of multiscale systems that are represented by a hybrid model embedded with data-driven parametrizations and we achieve this through data assimilation techniques.

B. Kraichnan turbulence

Here, we summarize the mathematical background of subgrid-scale parametrizations in the LES of two-dimensional turbulence. Even though two-dimensional turbulence cannot be realized in practice, it is extensively used for modeling geophysical flows in the atmosphere and ocean [87,88]. The confinement of fluid turbulence to two spatial dimensions leads to Kraichnan-Batchelor-Leith (KBL) theory of dual cascade with an inverse energy cascade to larger scales and direct enstrophy cascade to smaller scales. The nondimensional vorticity transport equation for incompressible flows can be written as

$$\frac{\partial \omega}{\partial t} + J(\omega, \psi) = \frac{1}{\mathrm{Re}} \nabla^2 \omega, \tag{5}$$

$$J(\omega, \psi) = \frac{\partial \omega}{\partial x}\frac{\partial \psi}{\partial y} - \frac{\partial \omega}{\partial y}\frac{\partial \psi}{\partial x}, \tag{6}$$

where ω is the vorticity, ψ is the streamfunction, J is the Jacobian (or the nonlinear term), and Re is the Reynolds number of the flow. The vorticity and streamfunction are related to each other through the Poisson equation given by

$$\nabla^2 \psi = -\omega. \tag{7}$$

The governing equations for LES are obtained by applying a low-pass filtering operation, and the filtered vorticity transport equation can be written as

$$\frac{\partial \overline{\omega}}{\partial t} + \overline{J(\omega, \psi)} = \frac{1}{\mathrm{Re}} \nabla^2 \overline{\omega}. \tag{8}$$

The above equation can be rewritten as

$$\frac{\partial \overline{\omega}}{\partial t} + J(\overline{\omega}, \overline{\psi}) = \frac{1}{\mathrm{Re}} \nabla^2 \overline{\omega} + \Pi, \tag{9}$$

where the overbar quantities represent filtered variables and are evolved on a grid which is significantly coarse than required for the DNS. The effect of the unresolved scales due to truncation of high wave number flow scales is encompassed in a subgrid-scale source term Π and must be modeled. Mathematically, the true source term Π can be expressed as

$$\Pi = J(\overline{\omega}, \overline{\psi}) - \overline{J(\omega, \psi)}. \tag{10}$$

The approximation of subgrid processes plays an important role in determining the accuracy of large-scale flows and therefore the subgrid-scale parametrizations are critical to accurate LES simulations of geophysical flows [89]. Different models have been proposed in the literature for

subgrid-scale parametrizations in geophysical flows [90–95], and remains an active area of research due to complexity of the subgrid-scale closure modeling. In the present work, we put forth a data-driven framework based on a neural network to predict the approximate value of the source term Π as a function of resolved flow variables on the coarser grid. One of the main advantages of data-driven closure modeling is that they are computationally faster than dynamic closure modeling procedure that involves several test filtering operations [21,96,97].

III. NEURAL NETWORK PARAMETERIZATIONS

The parametrization problem in multiscale flows can be posed as a regression problem where the mapping between resolved scales and unresolved scales has to be determined. We consider supervised class of machine learning algorithms, where the optimal map between inputs and outputs is learned. In this section, we describe an artificial neural network (ANN) also called as multilayer perceptron, and convolutional neural network (CNN) to build data-driven parametrization models.

A. Artificial neural network

An artificial neural network is made up of several layers consisting of the predefined number of neurons. Each neuron consists of certain coefficients called weights and some bias. The weight determines how significant certain input feature is to the output. The input from the previous layer is multiplied by a weight matrix as follows:

$$S^l = \mathbf{W}^l \mathcal{X}^{l-1}, \tag{11}$$

where $\mathcal{X}^{l-1}$ is the output of the $(l-1)$th layer, $\mathbf{W}^l$ is the matrix of weights for the lth layer. The summation of the above input-weight product and the bias is then passed through a node's activation function which is usually some nonlinear function. The introduction of nonlinearity through activation function allows the neural network to learn highly complex relations between the input and output. The output of the lth layer can be written as

$$\mathcal{X}^l = \zeta(S^l + B^l), \tag{12}$$

where B^l is the vector of biasing parameters for the lth layer and ζ is the activation function. If there are L layers between the input and the output in a neural network, then the output of the neural network can be represented mathematically as follows:

$$\tilde{\mathcal{Y}} = \zeta_L\{\mathbf{W}^L, B^L, \ldots, \zeta_2[\mathbf{W}^2, B^2, \zeta_1(\mathbf{W}^1, B^1, \mathcal{X})]\}, \tag{13}$$

where $\mathcal{X}$ and $\tilde{\mathcal{Y}}$ are the input and output of the ANN, respectively. There are several activation functions that provides different nonlinearity. Some of the widely used activation functions are sigmoid $\zeta(\phi) = 1/(1+e^{-\phi})$, hyperbolic tangent (tanh) $\zeta(\phi) = (e^{\phi} - e^{-\phi})/(e^{\phi} + e^{-\phi})$, and rectified linear unit (ReLU) $\zeta(\phi) = \max[0, \phi]$.

The matrix $\mathbf{W}$ and B are determined through the minimization of the loss function (for example mean squared error between true and predicted labels). The gradient of the objective function with respect to weights and biases are calculated with the backpropagation algorithm. The optimization algorithms like the stochastic gradient descent method [98] provide a rapid way to learn optimal weights. The training procedure for ANN can be summarized as:

(1) The input and output of the neural network are specified along with some initial weights initialization for neurons.

(2) The training data is run through the network to produce output $\tilde{\mathcal{Y}}$ whose true label is $\mathcal{Y}$.

(3) The derivative of the objective function with each of the training weight is computed using the chain rule.

(4) The weights are then updated based on the learning rate and the optimization algorithm.

We continue to iterate through this procedure until convergence or the maximum number of iterations is reached. There are different ways in which the relationship between resolved and

unresolved variables in multiscale systems can be learned with the ANN. The most common method is to employ point-to-point mapping, where the input features at a single grid point are utilized to learn the output labels at that point [22,29,99]. Another method is to include the information at neighboring grid points to determine the output label at a single point [26,96]. For a two-level Lorenz system, we train our ANN by including information at different number of neighboring grid points and assess how does this additional information affects in learning the correlation between resolved and unresolved variables. We investigate three types of ANN models and they can be written as

$$\text{ANN-3} : \{X_{i-1}, X_i, X_{i+1}\} \in \mathbb{R}^3 \rightarrow \{\tilde{G}_i\} \in \mathbb{R}^1, \tag{14}$$

$$\text{ANN-5} : \{X_{i-2} \ldots, X_{i+2}\} \in \mathbb{R}^5 \rightarrow \{\tilde{G}_i\} \in \mathbb{R}^1, \tag{15}$$

$$\text{ANN-7} : \{X_{i-3} \ldots, X_{i+3}\} \in \mathbb{R}^7 \rightarrow \{\tilde{G}_i\} \in \mathbb{R}^1, \tag{16}$$

where $\tilde{G}_i$ is the predicted parametrization at ith grid point and X_i is the resolved variable. For the training, we assume that the resolved variables and the parametrizations are known exactly and are computed by solving Eqs. (1) and (2) in a coupled manner. For all ANN architectures used in this study, we apply two hidden layers with 40 neurons and ReLU activation function for all hidden layers. For the output layer, the linear activation function is used. The ANN is trained using an Adam optimizer for 300 iterations.

B. Convolutional neural network

The convolutional neural network (CNN) is particularly attractive when the data is in the form of two-dimensional images [100]. Here, we present the CNN architecture assuming that the input and output of the neural network have the structure of two-dimensional images. This formulation can be easily applied to one-dimensional images when the dimension in one direction is collapsed to one. The Conv layers are the fundamental building blocks of the CNN. Each Conv layer has a predefined number of filters (also called kernels) whose weights have to be learned using the backpropagation algorithm. The shape of the filter is usually smaller than the actual image and it extends through the full depth of the input volume from the previous layer. For example, if the input to the CNN has $256 \times 256 \times 1$ dimension where 1 is the number of input features, then the kernels of the first Conv layer can have $3 \times 3 \times 1$ shape. During the forward propagation, the filter is convolved across the width and height of the input volume to produce the two-dimensional map. The two-dimensional map is constructed by computing the dot product between the weights of the filter and the input volume at any position and then sliding it over the whole volume. Mathematically the convolution operation corresponding to one filter can be written as

$$S^l_{ij} = \sum_{p=-\Delta_i/2}^{\Delta_i/2} \sum_{q=-\Delta_j/2}^{\Delta_j/2} \sum_{r=-\Delta_k/2}^{\Delta_k/2} \mathbf{W}^l_{pqr} \mathcal{X}^{l-1}_{i+p\, j+q\, k+r} + B_{pqr}, \tag{17}$$

where Δ_i, Δ_j, Δ_k are the sizes of filter in each direction, $\mathbf{W}^l_{pqr}$ are the entries of the filter for lth Conv layer, B_{pqr} is the biasing parameter, and $\mathcal{X}^{l-1}_{ijk}$ is the input from $(l-1)$th layer. Each Conv layer will have a set of predefined filters and the two-dimensional map produced by each filter is then stacked in the depth dimension to produce a three-dimensional output volume. This output volume is passed through an activation function to produce a nonlinear map between inputs and outputs. The output of the lth layer is given by

$$\mathcal{X}^l_{ijk} = \zeta\left(S^l_{ijk}\right), \tag{18}$$

where ζ is the activation function. It should be noted that as we convolve the filter across the input volume, the size of the input volume shrinks in height and width dimension. Therefore, it is common

practice to pad the input volume with zeros called zero-padding. The zero-padding permits us to control the shape of the output volume and is used in our neural network parametrization framework to preserve the shape so that input and output width and height are the same. The main advantage of CNN is its weight sharing property because the filter of the smaller size is shared across the whole image which is larger in size. This allows CNN to handle large data without the significant computational overhead. The CNN mapping for learning parametrizations in a two-level Lorenz model can be mathematically presented as

$$\text{CNN} : \{X_1, \ldots, X_n\} \in \mathbb{R}^n \rightarrow \{\tilde{G}_1, \ldots \tilde{G}_n\} \in \mathbb{R}^n, \tag{19}$$

where X_i is the resolved variable and $\tilde{G}_i$ is the predicted parametrization. Therefore, the solution at a single time step corresponds to one training example for training the CNN. In our CNN architecture, we use only one hidden layer between the input and output. This hidden layer has 128 filters with 7×1 shape. We apply ReLU activation function for all hidden layers and the linear activation function for the output layer. Also, the zero-padding is used to keep the input and output shape the same. The CNN is trained with an Adam optimizer for 400 iterations. The hyperparameters of both ANN and CNN architectures were obtained through parametric study for a different number of neurons/filters and the number of hidden layers. 80% of the total data selected randomly was used for training and the remaining 20% of the data was used for validation. The selection of hyperparameters is done in such a way that the mean squared error between the actual and predicted parametrization drops smoothly for both training and validation dataset so that overfitting is avoided and our model generalizes well to the unseen data. There are other methods like regularization, dropout, ensembling from different models, early stopping that can be adopted to prevent overfitting. An extensive hyperparameter search can be carried out for complex geophysical flows using neural architecture search packages like DeepHyper [101] and Tune [102].

For the two-dimensional turbulence, we learn the source term Π as a function of resolved flow variables, i.e., the vorticity ω, the streamfunction ψ, and two eddy-viscosity kernels as input features. The use of eddy-viscosity kernels as input features can be considered as a feature engineering step where certain important quantities are precomputed and the neural network is presented with it as raw input features to facilitate the faster training and robust prediction. The CNN map for two-dimensional turbulence can be mathematically written as

$$\text{CNN} : \{\bar{\omega}, \bar{\psi}, |\bar{S}|, |\nabla\bar{\omega}|\} \rightarrow \{\tilde{\Pi}\}, \tag{20}$$

where $\tilde{\Pi}$ is the predicted source term, $|\bar{S}|$ is the Smagorinsky kernel, and $|\nabla\bar{\omega}|$ is the Leith kernel. The Smagorinsky and Leith kernels are computed as follows:

$$|\bar{S}| = \sqrt{4\left(\frac{\partial^2\bar{\psi}}{\partial x\partial y}\right)^2 + \left(\frac{\partial^2\bar{\psi}}{\partial x^2} - \frac{\partial^2\bar{\psi}}{\partial y^2}\right)^2}, \tag{21}$$

$$|\nabla\bar{\omega}| = \sqrt{\left(\frac{\partial\bar{\omega}}{\partial x}\right)^2 + \left(\frac{\partial\bar{\omega}}{\partial y}\right)^2}. \tag{22}$$

The CNN architecture to learn the parametrization model for Kraichnan turbulence consist of 6 hidden layers and 16 filters in each hidden layers. The size of the filter in each hidden layer is 3×3 and the ReLU activation function is utilized for hidden layers. The training is performed for 800 epochs using the Adam optimizer. We note here that the predicted source term by the CNN is further post-processed during the deployment before it is injected into the vorticity transport equation to ensure numerical stability and we detail that procedure in Sec. V.

IV. DATA ASSIMILATION

As highlighted in many studies, neural network parametrizations suffer from instabilities and biases once the trained model is deployed in a forward solver [16,16,58–60]. From our numerical

experiments with the two-level Lorenz system, we observe that the forward model with only neural network parametrizations delivers accurate prediction only up to some time and after that the model starts deviating from the true trajectory. To address this issue and improve the long-term forecast with hybrid models, we utilize the data assimilation (DA) to incorporate noisy measurements for the prediction of future state. The main theme of DA is to extract the information from observational data to correct dynamical models and improve their prediction. There is a rich literature on DA [61–63,103,104] and here we discuss only sequential data assimilation problem and then outline the algorithm procedure for perturbed observations ensemble Kalman filter (EnKF), and the deterministic ensemble Kalman filter (DEnKF).

We consider the dynamical system whose evolution can be represented as

$$\mathbf{x}_{k+1} = \mathbf{M}_{t_k \to t_{k+1}}(\mathbf{x}_k) + \mathbf{w}_{k+1}, \tag{23}$$

where $\mathbf{x}_k \in \mathbb{R}^n$ is the state of the dynamical system at discrete time t_k, $\mathbf{M} : \mathbb{R}^n \to \mathbb{R}^n$ is the nonlinear model operator that defines the evolution of the system. The term $\mathbf{w}_{k+1}$ denotes the model noise that takes into account any type of uncertainty in the model that can be attributed to boundary conditions, imperfect models, etc. Let $\mathbf{z}_k \in \mathbb{R}^m$ be observations of the state vector obtained from noisy measurements and cane be written as

$$\mathbf{z}_k = \mathbf{h}(\mathbf{x}_k) + \mathbf{v}_k, \tag{24}$$

where $\mathbf{h}(\cdot)$ is a nonlinear function that maps $\mathbb{R}^n \to \mathbb{R}^m$, and $\mathbf{v}_k \in \mathbb{R}^m$ is the measurement noise. We assume that the measurement noise is a white Gaussian noise with zero mean and the covariance matrix $\mathbf{R}_k$, i.e., $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$. Additionally, the noise vectors $\mathbf{w}_k$ and $\mathbf{v}_k$ are assumed to be uncorrelated to each other at all time steps. The sequential data assimilation can be considered as a problem of estimating the state $\mathbf{x}_k$ of the system given the observations up to time t_k, i.e., $\mathbf{z}_1, \ldots, \mathbf{z}_k$. When we utilize observations to estimate the state of the system, we say that the data are assimilated into the model. We will use the notation $\widehat{\mathbf{x}}_k$ to denote an analyzed state of the system at time t_k when all of the observations up to and including time t_k are used in determining the state of the system. When all the observations before (but not including) time t_k are utilized for estimating the state of the system, then we call it the forecast estimate and denote it as $\mathbf{x}_k^f$.

The ensemble Kalman filter (EnKF) [105] follows the Monte Carlo approach to approximate the probability distribution in the Kalman filter equations [106]. We start by initializing the state of the system for different ensemble members as follows:

$$\widehat{\mathbf{X}}_0(i) = \mathbf{m}_0 + \mathbf{y}_0(i), \quad i = 1 \ldots N, \tag{25}$$

where $\mathbf{y}_0(i) \sim \mathcal{N}(0, \mathbf{P}_0)$, $\mathbf{m}_0$ is some assumed mean state of the system, $\mathbf{P}_0$ is the initial covariance error matrix, and N is the number of ensemble members. The propagation of the state for each ensemble over the time interval $[t_k, t_{k+1}]$ can be written as

$$\mathbf{X}_{k+1}^f(i) = \mathbf{M}_{t_k \to t_{k+1}}(\widehat{\mathbf{X}}_k(i)) + \mathbf{w}_{k+1}. \tag{26}$$

The term $\mathbf{w}_{k+1}$ accounting for model imperfections is usually assumed to be Gaussian noise. In this study, we consider the model error by means of multiplicative inflation [107] and without loss of generality we set $\mathbf{w}_{k+1} = 0$. The prior state and the prior covariance matrix are approximated using the sample mean and error covariance matrix $\mathbf{P}_{k+1}^f$ as follows:

$$\mathbf{x}_{k+1}^f = \frac{1}{N} \sum_{i=1}^{N} \mathbf{X}_{k+1}^f(i), \tag{27}$$

$$\mathbf{E}_{k+1}^f(i) = \mathbf{X}_{k+1}^f(i) - \mathbf{x}_{k+1}^f, \tag{28}$$

$$\mathbf{P}_{k+1}^f = \frac{1}{N-1} \sum_{i=1}^{N} \mathbf{E}_{k+1}^f(i) \big(\mathbf{E}_{k+1}^f(i)\big)^{\mathrm{T}}. \tag{29}$$

Once the observations are available at time t_{k+1}, we generate N realizations of perturbed observations as follows:

$$\mathbf{Z}_{k+1}(i) = \mathbf{z}_{k+1} + \mathbf{v}_{k+1}(i), \tag{30}$$

where $\mathbf{v}_{k+1}(i) \sim \mathcal{N}(0, \mathbf{R}_{k+1})$. Each member of the forecast ensemble $\mathbf{X}^f_{k+1}(i)$ is analyzed using the Kalman filter formulas as follows:

$$\widehat{\mathbf{X}}_{k+1}(i) = \mathbf{X}^f_{k+1}(i) + \mathbf{K}_{k+1}\big\{\mathbf{Z}_{k+1}(i) - \mathbf{h}\big[\mathbf{X}^f_{k+1}(i)\big]\big\}, \tag{31}$$

$$\mathbf{K}_{k+1} = \mathbf{P}^f_{k+1}\mathbf{H}^{\mathrm{T}}_{k+1}\big[\mathbf{H}_{k+1}\mathbf{P}^f_{k+1}\mathbf{H}^{\mathrm{T}}_{k+1} + \mathbf{R}_{k+1}\big]^{-1}, \tag{32}$$

where $\mathbf{H} \in \mathbb{R}^{m\times n}$ is the Jacobian of observation function $\mathbf{h}(\cdot)$. The analysis state estimate at time t_{k+1} is computed using the sample mean of all ensemble members as

$$\widehat{\mathbf{x}}_{k+1} = \frac{1}{N}\sum_{i=1}^{N}\widehat{\mathbf{X}}_{k+1}(i). \tag{33}$$

To take model imperfections into account, all ensemble members are updated by applying inflation to all ensemble anomalies as follows:

$$\widehat{\mathbf{X}}_{k+1}(i) \leftarrow \widehat{\mathbf{x}}_{k+1} + \lambda\cdot(\widehat{\mathbf{X}}_{k+1}(i) - \widehat{\mathbf{x}}_{k+1}), \tag{34}$$

where λ is the inflation factor. The inflation also helps to address the problem of covariance underestimation due to small number of ensembles [63]. The inflation factor can either be a scalar or it can be made space and time dependent to improve the filter performance [108,109]. In this study, we use the constant value of the inflation factor over the entire space at all times.

As a variant of the low-rank sequential nonlinear filtering framework, we also utilize the deterministic EnKF (DEnKF) algorithm proposed by Sakov and Oke [110] for the data assimilation. We start the DEnKF algorithm by initializing the state estimate for all ensemble members similar to the EnKF algorithm as given in Eq. (25). The anomalies between the forecast estimate of all ensembles and its sample mean [calculated using Eq. (27)] is

$$\mathbf{A}^f_{k+1}(i) = \mathbf{X}^f_{k+1}(i) - \mathbf{x}^f_{k+1}. \tag{35}$$

Once the observations are available at time t_{k+1}, the forecast state estimate is assimilated using the Kalman filter analysis equation as follows:

$$\widehat{\mathbf{x}}_{k+1} = \mathbf{x}^f_{k+1} + \mathbf{K}_{k+1}\big[\mathbf{z}_{k+1} - \mathbf{h}\big(\mathbf{x}^f_{k+1}\big)\big]. \tag{36}$$

Here, the Kalman gain matrix is computed using its square root version (without storing or computing $\mathbf{P}^f_{k+1}$ explicitly) as follows:

$$\mathbf{K}_{k+1} = \frac{\mathcal{A}^f_{k+1}\big(\mathbf{H}_{k+1}\mathcal{A}^f_{k+1}\big)^{\mathrm{T}}}{N-1}\left[\frac{\big(\mathbf{H}_{k+1}\mathcal{A}^f_{k+1}\big)\big(\mathbf{H}_{k+1}\mathcal{A}^f_{k+1}\big)^{\mathrm{T}}}{N-1} + \mathbf{R}_{k+1}\right]^{-1}, \tag{37}$$

where $\mathbf{H} \in \mathbb{R}^{m\times n}$ is the Jacobian of the observation operator (i.e., $H_{kl} = \frac{\partial h_k}{\partial x_l}$), and the matrix $\mathcal{A}^f_{k+1} \in \mathbb{R}^{n\times N}$ is concatenated as follows:

$$\mathcal{A}^f_{k+1} = \big[\mathbf{A}^f_{k+1}(1), \mathbf{A}^f_{k+1}(2), \ldots, \mathbf{A}^f_{k+1}(N)\big]. \tag{38}$$

The anomalies for all ensemble members are then updated separately with half the Kalman gain as follows:

$$\widehat{\mathbf{A}}_{k+1}(i) = \mathbf{A}^f_{k+1}(i) - \tfrac{1}{2}\mathbf{K}_{k+1}\mathbf{H}_{k+1}\mathbf{A}^f_{k+1}(i). \tag{39}$$

The analysis state for all ensemble members is obtained by adding ensemble anomalies and can be written as

$$\widehat{\mathbf{X}}_{k+1}(i) = \widehat{\mathbf{x}}_{k+1} + \lambda \cdot \widehat{\mathbf{A}}_{k+1}(i), \tag{40}$$

where λ is the inflation factor. We validate our implementation of the DEnKF algorithm using the one-level Lorenz 96 model and is detailed in the Appendix.

V. NUMERICAL EXPERIMENTS

In the following, we present the findings of our numerical experiments with a two-level Lorenz 96 model and Kraichnan turbulence.

A. Two-level Lorenz 96 model

In this subsection, we discuss the results of numerical experiments with a two-level variant of the Lorenz 96 system embedded with neural network parametrizations for the unresolved variables. We utilize the fourth-order Runge-Kutta numerical scheme with a time step $\Delta t = 0.001$ for temporal integration of the Lorenz 96 model. We apply the periodic boundary condition for the slow variables, i.e., $X_{i-n} = X_{i+n} = X_i$. The fast variables are extended by letting $Y_{j,i-n} = Y_{j,i+n} = Y_{j,i}$, $Y_{j-J,i} = Y_{j,i-1}$, and $Y_{j+J,i} = Y_{j,i+1}$. The physical initial condition is computed by starting with an equilibrium condition at time $t = -5$ for slow variables. The equilibrium condition for slow variables is $X_i = F$ for $i \in 1, 2, \ldots, n$. We perturb the equilibrium solution for the 18th state variable as $X_{18} = F + 0.01$. At the time $t = -5$, the fast variables are assigned with random numbers between $-F/10$ and $F/10$. We integrate a two-level Lorenz 96 model by solving both Eqs. (1) and (2) in a coupled manner up to time $t = 0$. With this initial condition (i.e., at $t = 0$), we generate the training data for neural networks by integrating the two-level Lorenz 96 model from $t = 0$ to $t = 10$. Therefore, we gather 10 000 temporal snapshots to generate the training data. For all our numerical experiments, we use 80% of the data to train the neural network and 20% data to validate the training. We assess the performance of a trained neural network by deploying it in a forecast model for temporal integration between time $t = 10$ to $t = 20$. Therefore, there is no overlap between the data used for training and testing. Since the neural network has not seen the testing data during the training, the performance of neural network parametrizations in this temporal region will give us an insight on its generalizability to unseen data.

First, we present results for ANN-based parametrizations trained using neighboring stencil mapping as discussed in Sec. III A. Figure 1 displays the full state trajectory of the Lorenz 96 model from time $t = 10$ to $t = 20$ computed by solving both the evolution of slow and fast variables (i.e., True) and with ANN-based parametrizations for fast variables (i.e., ANN-3, ANN-5, ANN-7). The difference between the true solution field and the predicted solution field is also depicted in Fig. 1. It can be observed that the predicted solution field starts deviating from the true solution field at around $t \approx 12$ for all ANN-based parametrizations.

Next, we illustrate how the prediction of a two-level Lorenz 96 model with neural network parametrizations can be improved using sequential data assimilation by incorporating noisy observations in the future state prediction. For our twin experiment, we obtain observations by adding noise drawn from the Gaussian distribution with zero mean and the covariance matrix $\mathbf{R}_k$, i.e., $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$. We use $\mathbf{R}_k = \sigma_b^2 \mathbf{I}$, where σ_b is the standard deviation of measurement noise and is set at $\sigma_b = 1$. We assume that observations are sparse in space and are collected at every 10th time step. The number of ensemble members used for all numerical experiments is $N = 30$. We present two levels of observation density in space for the DA. For the first case, we employ observations at $[X_4, X_8, \ldots, X_{36}] \in R^9$ for the assimilation. The second set of observations consists of 50% of the full state of the system, i.e., $[X_2, X_4, \ldots, X_{36}] \in R^{18}$. In Fig. 2, we provide the full state trajectory prediction for the ANN-5 parametrization without any DA and with DA for two sets of observations. We can observe that there is a substantial improvement in the long-term prediction even with

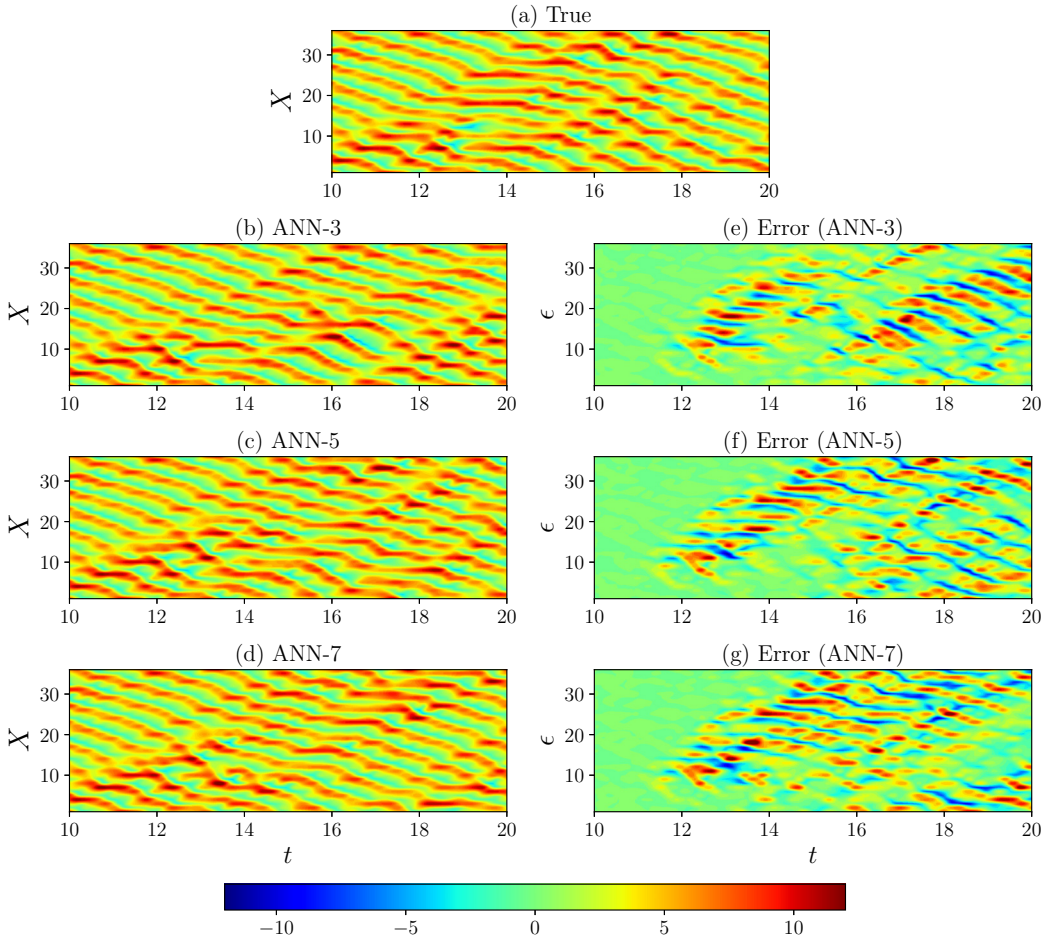

FIG. 1. Full state trajectory of the multiscale Lorenz 96 model with the closure term computed using the different neighboring stencil mapping feedforward ANN architecture.

only 25% of the observations incorporated through the DEnKF algorithm. The results in Fig. 2 provide the evidence for the good performance of the present framework in achieving accurate long-term prediction for hybrid models embedded with data-driven parametrizations. Therefore, the present framework can lead to accurate forecasting by exploiting online measurements coming from various types of sensor networks and can find applications in different fields like climate modeling, turbulence closure modeling where the subgrid-scale parametrizations are unavoidable.

Figure 3 illustrates the time evolution of the full state trajectory of a two-level Lorenz 96 model with CNN-based parametrizations for unresolved scales. CNN is fed with the entire state of the slow variables as an input and it calculates the parametrizations of fast variables at all grid points. From Fig. 3, we can deduce that the predicted state trajectory starts deviating from the true state at around $t \approx 12$ when only CNN-based parametrizations are employed in the forward model of slow variables. When we incorporate observations through DA, we observe considerable improvement in the state prediction over a longer period.

Based on results presented in Figs. 2 and 3, we can notice that the error is slightly higher between time $t = 18$ to $t = 20$ for the CNN-based parametrizations empowered with DA. One reason for the inaccurate forecast can be attributed to the uncertainty in the prediction of parametrizations by CNN.

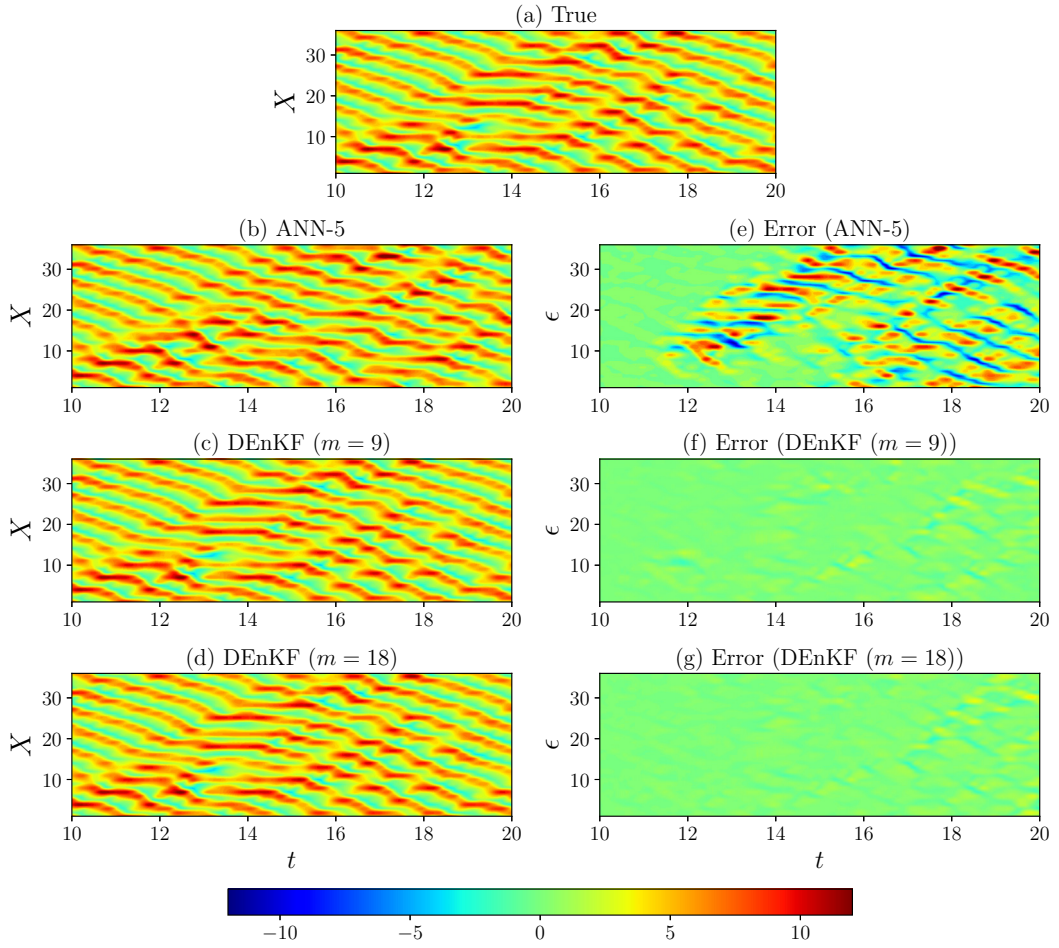

FIG. 2. Full state trajectory of the multiscale Lorenz 96 model with the closure term computed using the five-point neighboring stencil mapping feedforward ANN architecture and the DEnKF used for data assimilation.

We highlight here that both ANN-5 and CNN architectures used in this study have similar number of trainable parameters. However, we see a better performance of the ANN-5 architecture over CNN due to a more number of training examples in the case of the ANN. For the ANN, every single point of the two-level Lorenz 96 system is one training example and therefore a single time snapshot of the training data leads to 36 samples for training. However, in the case of CNN, the total number of training samples is equal to the total number of time snapshots available for training. Therefore, we observe the better performance of the ANN-5 over CNN.

Another potential reason for this discrepancy can be the stochastic nature of the parametrization model. The true parametrization model in itself is stochastic and might not follow a Gaussian distribution. To isolate the source of error, we integrate the forecast model for a two-level Lorenz 96 model without any parametrizations. The two-level Lorenz 96 model with no parametrizations is equivalent to setting the coupling coefficient $h = 0$ in Eq. (1) and it reduces to one-level Lorenz 96 model as presented in Eq. (A1). We note here that the observations used for data assimilation are the same as the numerical experiments with a two-level Lorenz 96 model. Therefore, the effect of unresolved scales is embedded in observations. The parametrization of fast variables [i.e.,

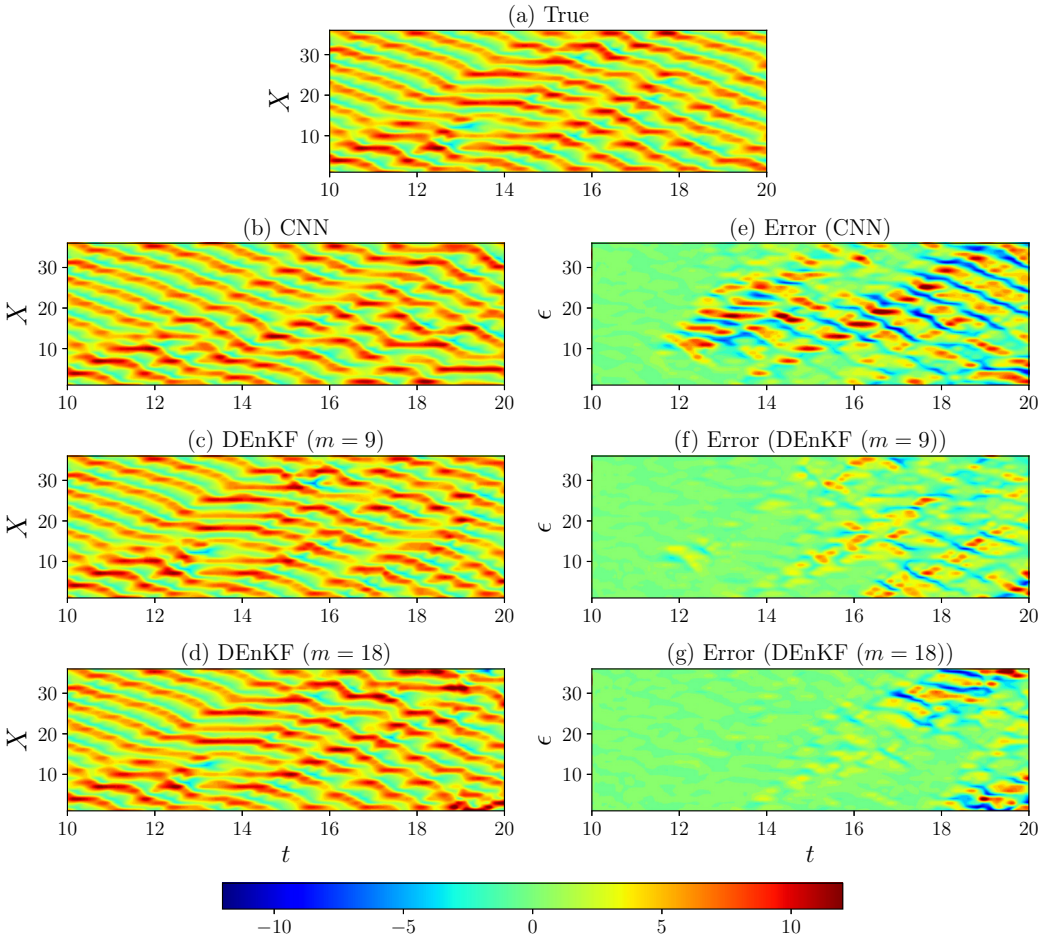

FIG. 3. Full state trajectory of the multiscale Lorenz 96 model with the closure term computed using the CNN architecture and the DEnKF used for data assimilation

$\frac{hc}{b}\sum_{j=1}^{J} Y_{j,i}$ term in Eq. (1)] can be considered as an added noise to the true state of the system for a one-level Lorenz 96 model presented in Eq. (A1).

In Fig. 4, we report the true state of a two-level Lorenz 96 model and also the predicted state trajectory using the DA framework with no parametrization. We provide the results for three sets of observations utilized in DA. The observations are incorporated at every 10^{th} time step of the model through assimilation stage. We can observe that, even when 100% of the full state is observable, we do not recover the true state trajectory of a two-level Lorenz 96 model. With this observation, we can conclude that it is essential to incorporate parametrization of unresolved scales into a forward model of the DA procedure to recover the accurate state trajectory. The root-mean-squared error between the assimilated states and true states for three sets of observations is provided in Table I.

In this numerical experiment with the truncated model, the observations include the effect of unresolved scales and can be considered as an added noise. The sequential DA methods based on Kalman filters deliver a considerably accurate solution when the model and observations noise is drawn from a Gaussian distribution and enough observations are provided. If the parametrization of unresolved scales follows a Gaussian distribution, we should be able to recover the accurate state of the system as the density of observations is increased. However, as reported in Fig. 4, there is

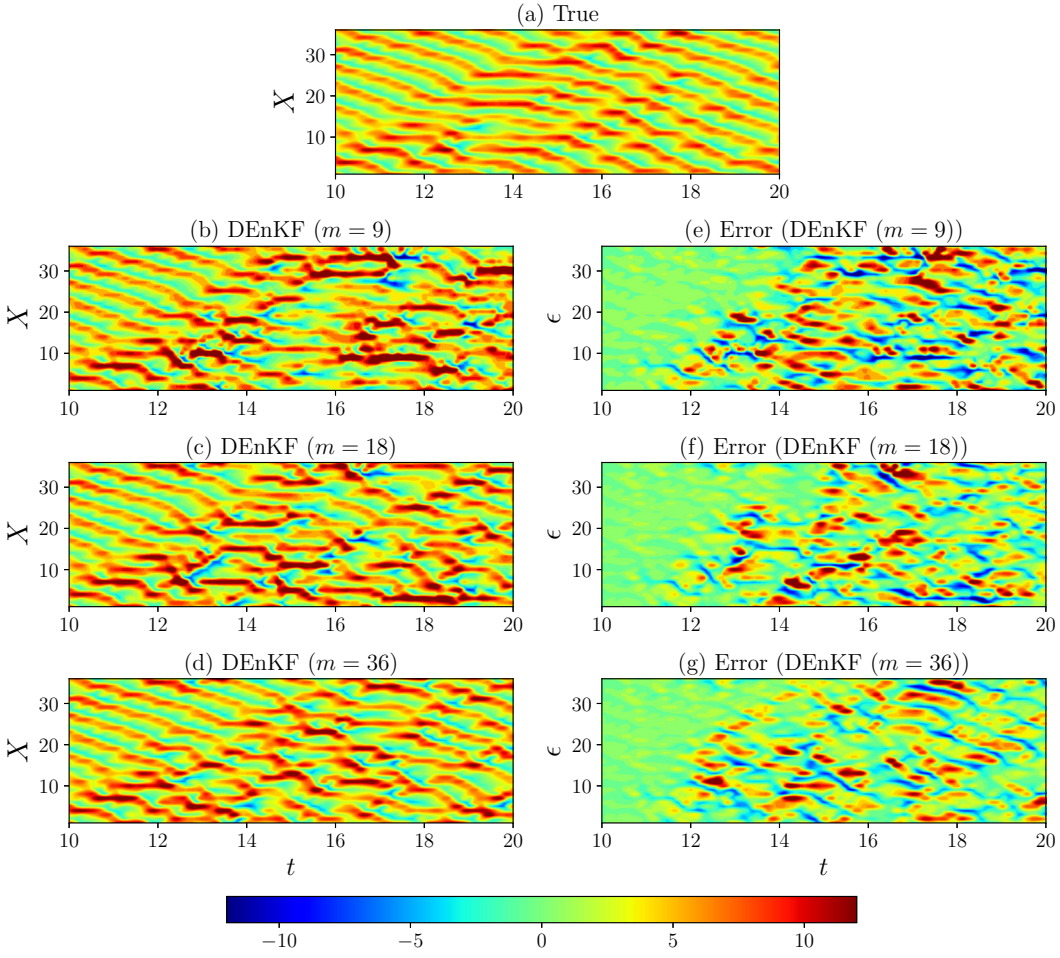

FIG. 4. Full state trajectory of the multiscale Lorenz 96 model with no closure for subgrid processes. The observation data for the DEnKF algorithm is obtained by adding measurement noise to the exact solution of the multiscale Lorenz 96 system.

a high level of inaccuracy even when 100% of the state is observable. Therefore, we can conclude that there is a considerable benefit of including neural network parametrizations compared to using no parametrization in the forecast model. The results provided in Figs. 2 and 3 also shows that the neural network parametrizations can capture the non-Gaussian statistics of subgrid-scale processes and this leads to accurate forecasting over a longer period. There are other DA approaches that deal with non-Gaussian distributions for noise vectors [111–116]. We restrict ourselves to the DEnKF algorithm for DA in this study and plan to explore other DA algorithms in our future work.

We assess the quantitative performance of different numerical experiments performed in this study using the root-mean-squared error (RMSE) between the true and predicted state of slow variables in a two-level Lorenz 96 model. The RMSE is computed as follows:

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\frac{1}{n_t}\sum_{i=1}^{n}\sum_{k=1}^{n_t}\left[X_i^{\mathrm{T}}(t_k) - X_i^{\mathrm{P}}(t_k)\right]^2}, \tag{41}$$

TABLE I. Quantitative assessment of different neural network parametrizations for subgrid-scale processes using the total root-mean-squared error given by Eq. (41).

Framework	RMSE
Only neural network parameterizations	
ANN-3	3.38
ANN-5	3.73
ANN-7	3.77
CNN	3.79
Only data assimilation	
No parametrizations ($m = 9$)	5.11
No parametrizations ($m = 18$)	4.30
No parametrizations ($m = 36$)	3.92
Neural network parametrizations with data assimilation	
ANN-5 ($m = 9$)	0.52
ANN-5 ($m = 18$)	0.53
CNN ($m = 9$)	2.13
CNN ($m = 18$)	2.20

where X_i^{T} is the true state of the system and X_i^{P} is the predicted state of the system. Table I reports the RMSE for a two-level Lorenz 96 model for all cases investigated in this work. We can see that the RMSE is very high when we do not use any parametrizations for unresolved scales even when measurements for an entire state of the system are incorporated through DA. The data assimilation alone can not account for the effect of unresolved scales, even though their effect is present in the observations data. Therefore, it is imperative to include parametrizations of fast variables in the forecast model of slow variables. We observe that the ANN architecture provides slightly more accurate results than the CNN-based parametrizations for fast variables. Also, the RMSE is minimum for the ANN-3 parametrizations and we observe a slight increase in RMSE by including more neighboring information. One potential reason for this observation can be the use of the same hyperparameters for all ANN architectures. However, this change is very small and the RMSE is the same order of magnitude for all types of neural network parametrizations. The RMSE is almost the same when 25% or 50% of the full state of the system is observed in data assimilation framework.

We highlight here that in the previous numerical experiment with the truncated model, we assumed that our forecast model is a true model. However, often the forecast models in DA are imperfect, and the model error introduced due to truncation of the submodel is usually either modeled using the Gaussian noise or covariance inflation. Indeed, the ensemble Kalman filter framework is very well established and, to the extent that if modeling errors can be represented as zero-mean with a simple correlation structure, then the DA is very effective at correcting model errors. For example, Brajard *et al.* [75] utilized a Gaussian noise with zero mean and a certain value of standard deviation (optimized by tuning experiments) to account for the model error arising due to truncation of the parametrizations in a two-level Lorenz system. Similarly, Attia and Constantinescu [109] proposed a variational framework for adaptive tuning of inflation and localization parameters and demonstrated its successful performance for a two-level Lorenz system. For a fair comparison with neural network-based parametrizations, we repeat the numerical experiments with the truncated model for different values of inflation factor. We keep the number of ensembles fixed at $N = 30$, and the inflation factor is varied from 1.0 to 1.05 with an increment of 0.01. Figure 5 reports the RMSE for the truncated model for different inflation factors, and we can notice that with the proper choice of inflation factor and sufficient observations, the truncated model can also predict the true state of the two-level Lorenz system. In contrast to Fig. 4, Fig. 6 depicts the full state trajectory of the two-level Lorenz system estimated using the DEnKF algorithm with

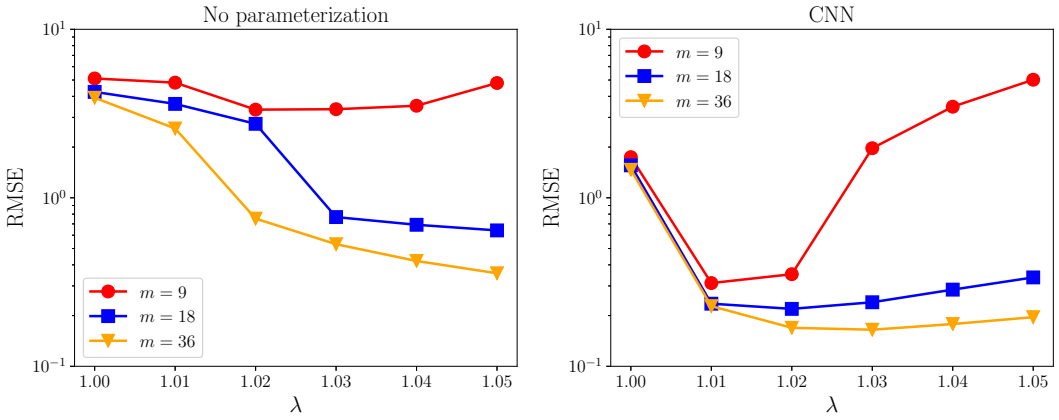

FIG. 5. The root-mean-squared error for different values of the inflation factor for three sets of observations. The number of ensembles is kept fixed at $N = 30$ for all sets of observations.

the inflation factor $\lambda = 1.03$ for three sets of observations. Overall, the results presented in Fig. 5 suggest that the true state of the two-level Lorenz system can be determined when more than 50% of the state is observable, and a proper value of inflation factor is employed for the DA with the truncated model. Moreover, the prediction of the true state of the system with neural network-based parametrization can be further improved by applying the inflation and the RMSE for different values of the inflation factor for CNN-based parametrization model are also shown in Fig. 5. It can be clearly seen that there is a significant accuracy gain by adding the CNN-based parametrizations for almost all configurations.

B. Kraichnan turbulence

We now characterize the performance of the EnKF algorithm, as described in Sec. IV, to estimate the state of the two-dimensional turbulence system when observation from high fidelity simulation are available. This test setup is particularly challenging because of the modeling of unresolved scales in the LES solver, which is employed as the forecast model for two-dimensional turbulence. The performance of the EnKF algorithm is impacted by the choice of the model and the forecast model should be accurate enough for error control techniques like covariance inflation, covariance localization, stochastic forcing, etc., to work. As we will see, if the effect of unresolved scales are not modeled, even the EnKF algorithm with high value of the inflation factor does not improve the state estimate of the two-dimensional turbulence system.

The governing equations for the two-dimensional turbulence are numerically solved using the second-order finite difference discretization. The nonlinear Jacobian term is discretized with the energy-conserving Arakawa [117] numerical scheme. A third-order total-variation-diminishing Runge-Kutta scheme is used for the temporal integration and a spectral Poisson solver is utilized to update streamfunction from the vorticity [118]. The computational domain is square in shape with dimensions $[0, 2\pi] \times [0, 2\pi]$ in x and y directions, respectively, and the periodic boundary condition is applied in both x and y directions. The training data for CNN is generated by carrying out the DNS at $\mathrm{Re} = 8000$ for two different initial conditions (independent of the truth model) on a grid resolution of 512×512 and then collecting total 800 snapshots (400 for each initial condition) between time $t = -2$ to $t = 2$. The initial condition is assigned in such a way that the maximum value of the initial energy spectra occurs at wave number $K_p = 10$. Further details of the randomization process for the initial condition can be found in related work [119]. The DNS data is coarsened to the 64×64 grid resolution using the spectral cutoff filter. The coarsened flow variables are then used to compute input features and labels for developing the data-driven subgrid-scale

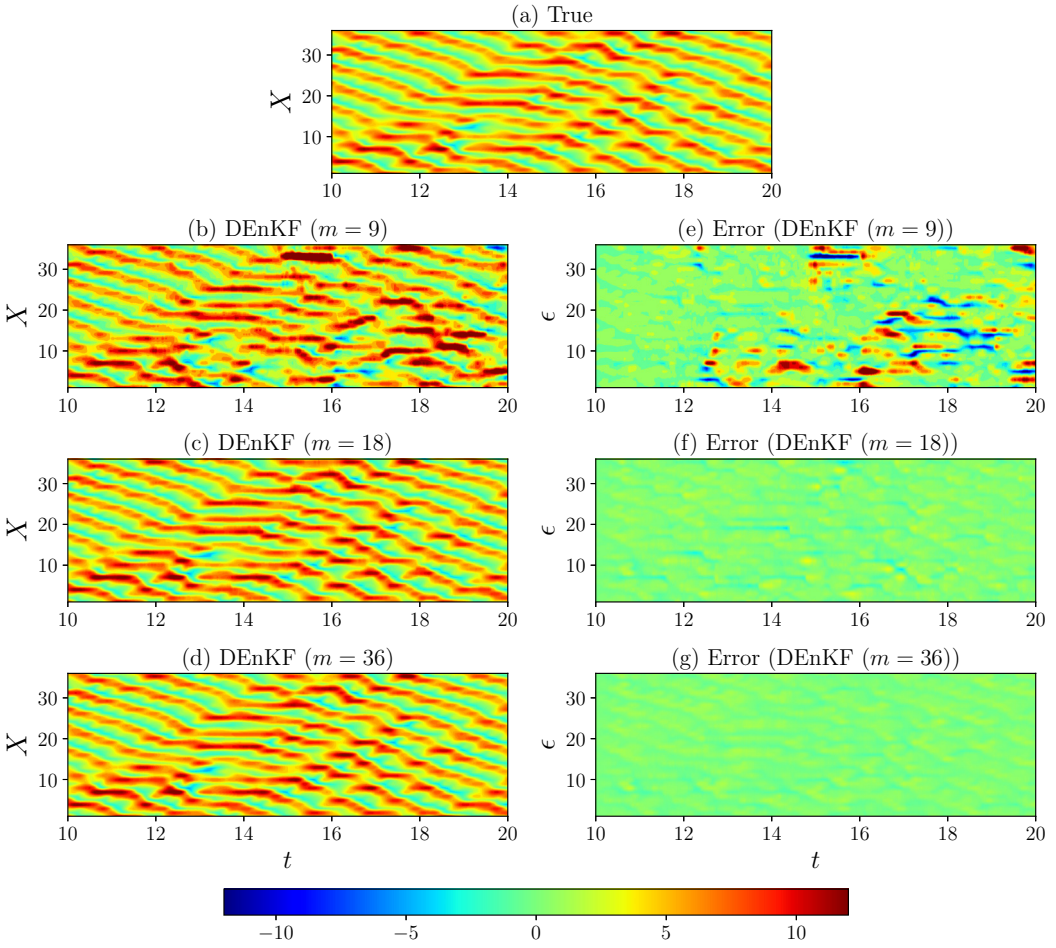

FIG. 6. Full state trajectory of the multiscale Lorenz 96 model with no closure for subgrid processes and for the inflation factor $\lambda = 1.03$. The observation data for the DEnKF algorithm is obtained by adding measurement noise to the exact solution of the multiscale Lorenz 96 system.

parametrization model as discussed in Sec. III B. Once the CNN is trained, it is deployed in the forecast model from time $t = 0$ to $t = 4$. We highlight that the data from time $t = 2$ to $t = 4$ is not seen during the training.

The predicted source term $\tilde{\Pi}$ has negative eddy viscosities embedded in it and needs to be post-processed before directly injecting it into the solver [26]. The numerical stability is ensured during the *a posteriori* deployment by truncating the learned source term $\tilde{\Pi}$ corresponding to negative numerical viscosities as follows:

$$\Pi = \begin{cases} \tilde{\Pi}, & \text{if } (\nabla^2 \bar{\omega})(\tilde{\Pi}) > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{42}$$

In addition to truncating the source term corresponding to negative eddy viscosity, the truncation is also applied at points where the local eddy viscosity is greater than the local-average average eddy viscosity. This truncation scheme can be mathematically expressed as

$$\Pi_{i,j} = \begin{cases} \tilde{\Pi}_{i,j}, & \text{if } \overline{\nu}_{i,j} > \nu_{i,j}, \\ 0, & \text{otherwise,} \end{cases} \tag{43}$$

where the eddy viscosity ν is computed as

$$\nu_{i,j} = \frac{\tilde{\Pi}_{i,j}}{\nabla^2 \bar{\omega}}, \tag{44}$$

and the local-averaged eddy viscosity $\bar{\nu}_{i,j}$ is calculated using the mean filtering kernel of size 3×3. This additional truncation scheme given in Eq. (43) aids in preserving the statistical quantities like the kinetic energy spectra close to the DNS solution as compared to utilizing only negative eddy viscosity truncation scheme. In terms of the computational cost, the data-driven subgrid-scale parametrization model is significantly fast compared to the dynamic Smagorinksy model (DSM) [120] and we observed up to 30% reduction in computational speed in the *a posteriori* runs with the CNN-based closure model.

The "truth" solution for the data assimilation is obtained by solving the vorticity transport equation with a grid resolution 512×512 for the Reynolds number $\mathrm{Re} = 8000$ and then applying the spectral cutoff filter to get the filtered DNS solution on the coarse grid with resolution 64×64. Other methods like multigridding can also be adopted to relax the solution from fine grid to coarse grid [121]. The DNS solution is generated from time $t = -2$ to $t = 0$ with $\Delta t = 1 \times 10^{-3}$. The assimilation is started at time $t = 0$ once the turbulence is developed and the initial transience from $t = -2$ to $t = 0$ is discarded. The observations are assimilated at every 10th time step of the forecast model. The synthetic observations are generated by sampling vorticity field at 32×32 (corresponding to 25% of the full state of the system) equidistant points in x and y directions from the filtered DNS solution and then contaminating them with the Gaussian noise, i.e., $\mathbf{v}_k \sim \mathcal{N}(0, \mathbf{R}_k)$, where $\mathbf{R}_k = \sigma_b^2 \mathbf{I}$. We set observation noise at $\sigma_b^2 = 2$.

The initialization of the ensemble members also plays an important role in the performance of the EnKF algorithm [122], especially in the initial period of the DA. There are different ways that have been used for the initialization of ensemble members in DA of turbulent flows, such as, using the solution field separated by a certain time from the turbulent flow simulation [70], adding random perturbation to mean flow solution [69]. We initialize all ensemble ensemble members by adding a random perturbations drawn from $\mathcal{N}(0, \mathbf{P}_0)$ to the filtered DNS solution at time $t = 0$. The initial covariance matrix is set at $\mathbf{P}_0 = \sigma_0^2 \mathbf{I}$, where $\sigma_0^2 = 1$.

We illustrate the performance of the EnKF algorithm for three different types of forecast models. The first forecast model is the unresolved numerical simulation (UNS), where subgrid-scale parametrization is completely discarded. In the second forecast model, the dynamic Smagorinsky model (DSM) [123,124] is used for modeling the source term in LES simulation. The third forecast model consists of utilizing the CNN-based subgrid-scale parametrization. For a fair comparison of the application of the EnKF algorithm to three different models, we run experiments with different combinations of the number of ensemble members and the inflation factor. The number of ensemble members is increased from 40 to 120 with an increment of 20 and the inflation factor is varied from 1.0 to 1.5 with an increment of 0.1. This gives us 30 different numerical experiments for each model. The performance of each numerical experiment is evaluated by computing the RMSE between the compensated kinetic energy spectra for the state estimated by the EnKF algorithm and the filtered DNS solution. The energy spectra is considered over the inertial range, i.e., between $K = 8$ to $K = 32$ and for 200 snapshots stored over the last half of the experiment timespan (from time $t = 2$ to $t = 4$). The RMSE results for these numerical experiments are shown in Fig. 7. Results in Fig. 7 suggest that the RMSE for the UNS model is significantly higher than the DSM and CNN model. The solution field predicted by UNS has a significant error due to the truncation of subgrid-scale parametrization and an increase in the number of ensembles or the inflation factor does not seem to help improve the state of the system predicted by the UNS model. The average RMSE for both DSM and CNN models is of a similar magnitude. Moreover, Fig. 7 indicates that the lower RMSE occurs at less number of ensembles for the CNN model with a moderate inflation factor (i.e., 1.2–1.3).

We evaluate the performance of different models through kinetic energy spectra calculation and second-order vorticity structure functions. We compute the vorticity structure function using the

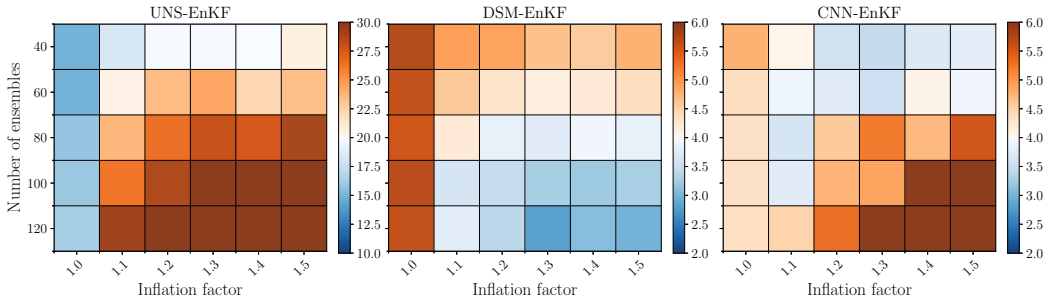

FIG. 7. Average RMSE of the compensated energy spectra for different combinations of the inflation factor and number of ensembles. The RMSE is averaged over the inertial range (i.e., from $K = 8$ to $K = 32$) considering data from $t = 2$ to $t = 4$.

formula given by [125] for two-dimensional turbulence and is as follows:

$$S_\omega(\mathbf{r}) = \langle |\bar{\omega}(\mathbf{x}+\mathbf{r}) - \bar{\omega}(\mathbf{x})|^2 \rangle, \tag{45}$$

where $\langle\,\rangle$ indicates ensemble averaging, $\mathbf{x}$ is the position on the grid, and $\mathbf{r}$ is certain distance from this location.

Figure 8 displays the kinetic energy spectra at final time $t = 4$ obtained with different models for $\mathrm{Re} = 8000$. We can observe that there is an accumulation of the energy near grid cutoff wave number in the case of the UNS model. The UNS-EnKF model is not able to correct the state estimate of the system due to very high noise in the forward model. We see an improvement in the energy spectra predicted by the DSM-EnKF and CNN-EnKF model compared to utilizing only the parametrization model. We note here that the kinetic energy spectra for the filtered DNS solution is identical to the DNS spectra till the grid cutoff wave number due to the use of a spectral cutoff filter. Figure 9 depicts the second-order vorticity structure functions at final time $t = 4$ where the evaluation with the FDNS shows that the EnKF is successful in improving the prediction of the vorticity structure function for both DSM and CNN model. We do not observe any improvement

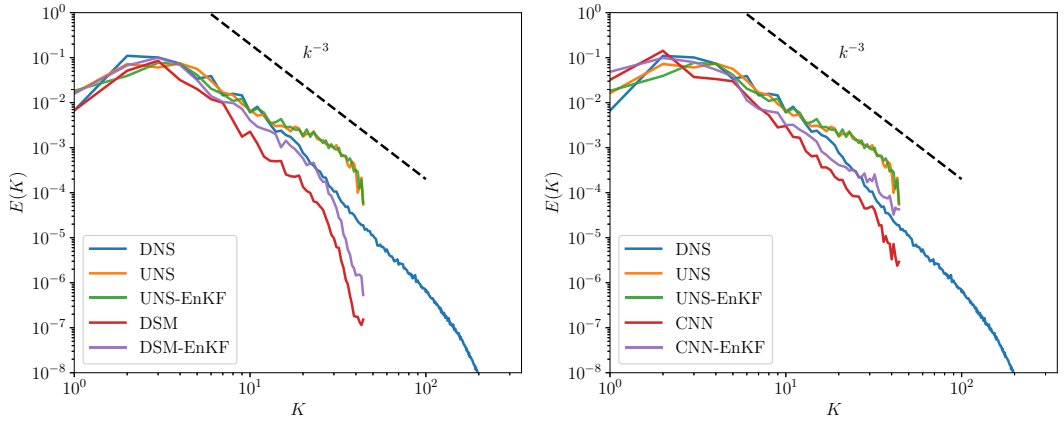

FIG. 8. *A posteriori* kinetic energy spectra at $t = 4$ at $N_x \times N_y = 64 \times 64$ grid resolution for different models. The number of ensembles and the inflation factor for the EnKF algorithm for different models corresponds to minimum value of the average RMSE between the compensated energy spectra for the filtered DNS solution and the solution predicted with different models. The EnKF related parameters are $N = 40$, $\lambda = 1.0$ for UNS-EnKF, $N = 120$, $\lambda = 1.3$ for DSM-EnKF, and $N = 40$, $\lambda = 1.3$ for CNN-EnKF.

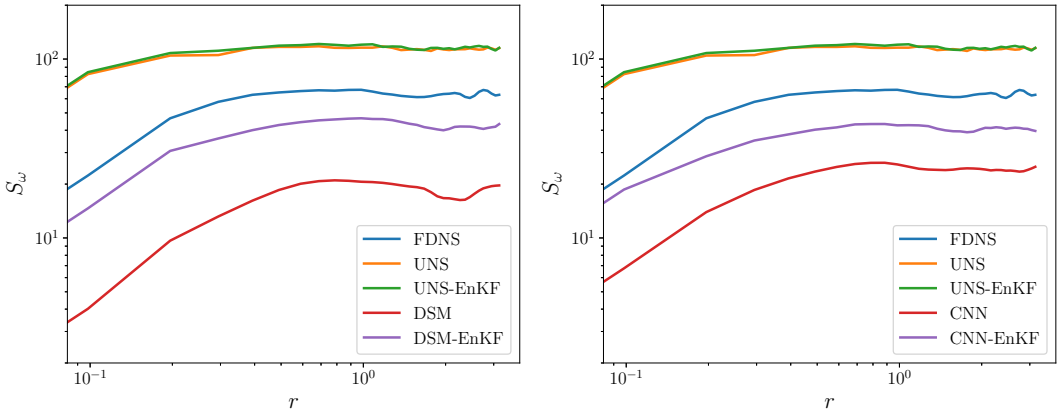

FIG. 9. *A posteriori* second-order vorticity structure functions plotted against r for $Re = 8000$ at $N_x \times N_y = 64 \times 64$ grid resolution with different models used for the subgrid-scale closure.

in the vorticity structure function prediction for the UNS model, which again emphasizes the importance of using an accurate forecast model in data assimilation.

VI. CONCLUDING REMARKS

The data-driven methods are successful in discovering model-free parametrizations from high-fidelity numerical simulations or experimental measurements and offer an alternative to parametrization models based on empirical or phenomenological arguments. The data-driven parametrization models are also computationally faster and are suitable for sequential data assimilation where multiple forward runs of a forecast model are required. To this end, we introduce a framework to apply data assimilation methods to the physics-based model embedded with data-driven parametrizations to achieve accurate long-term forecast in multiscale systems. We demonstrate that the forecasting capability of hybrid models can be significantly improved by exploiting online measurements from various types of sensor networks. Specifically, we use neural networks to learn the relation between resolved scales and the effect of unresolved scales (i.e., parametrizations). The deployment of the trained neural network in the forward simulation provides accurate prediction up to a short period and then there is a large discrepancy between true and predicted state of the system. To address this issue and to improve the long-term prediction, we exploit the sparse observations data through data assimilation.

We illustrate this framework for a two-scale variant of the Lorenz 96 model which consists of fast and slow variables whose dynamics are exactly known and for Kraichnan turbulence where the parametrization model for unresolved scales is not known *a priori*. We obtain a considerable improvement in the prediction for both test cases by combining neural network parametrizations and data assimilation compared to employing only neural network parametrizations. We also found that including an ML-based closure term seems to capture non-Gaussian statistics and significantly improve the forecast error. Based on our numerical experiments with data assimilation empowered neural network parametrizations, we can conclude that improving machine learning-based model prediction with data assimilation methods offers a promising research direction. We also highlight that the inaccuracy associated with data-driven parametrizations can be tackled with data assimilation error control techniques like covariance inflation, covariance localization, stochastic forcing, etc.

Our future work aims at leveraging the underlying physical conservation laws into neural network training to produce physically consistent parametrizations. As the deep-learning field is evolving rapidly, we can integrate modern neural network architectures and training methodology into our

framework to attain higher accuracy. In the present framework, we employ the ensemble Kalman filter-based algorithms for data assimilation. This algorithm gives accurate prediction when the uncertainty in model and observations follows a Gaussian distribution. We plan to investigate other data assimilation approaches like maximum likelihood ensemble filter methods that can handle the non-Gaussian nature of uncertainty in the mathematical model to get further improvement in the accuracy prediction. We will also test the present framework for more complex turbulent flows as a part of our future effort. Finally, we conclude by reemphasizing that the integration of data assimilation with hybrid physics-ML models can be effectively used for modeling of multiscale systems.

DATA AVAILABILITY

The data that supports the findings of this study are available within the article. Implementation details and Python scripts can be accessed from the Github repository [126].

ACKNOWLEDGMENTS

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Award No. DE-SC0019290. O.S. gratefully acknowledges their support. Disclaimer: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

APPENDIX: VALIDATION OF THE DETERMINISTIC ENSEMBLE-KALMAN FILTER

In this Appendix, we provide results of data assimilation with the DEnKF algorithm for one level Lorenz 96 model. The one level Lorenz 96 model is given as

$$\frac{dX_i}{dt} = -X_{i-1}(X_{i-2} - X_{i+1}) - X_i + F, \tag{A1}$$

for $i \in 1, 2, \ldots, 36$ and $F = 10$. The above model is completely deterministic as there is no parametrization of the unresolved scales. We use the similar settings as the two-level variant of the Lorenz 96 model for temporal integration using the fourth-order Runge-Kutta numerical scheme. The true initial condition is generated by integrating the solution starting from an equilibrium condition from $t = -5$ to $t = 0$. For all ensemble members, we start with an initial condition obtained by perturbing the true initial condition with a noise drawn from the Gaussian distribution with zero mean and the variance of 1×10^{-2}. The observations are generated for data assimilation by adding a measurement noise from the Gaussian distribution with zero mean and the variance of $\sigma_b^2 = 1$ (i.e., $\mathbf{R}_k = \mathbf{I}$) to the true state of the system. The observations are assumed to be available at every 10th time step, similar to the two-level variant of the Lorenz 96 model.

As depicted in Fig. 10, we can conclude that the DEnKF can correct the erroneous trajectory even when only 9 observations are employed for data assimilation. As the amount of observations is increased to 18, we observe a reduction in the error. We reiterate here that, we have complete control over the model (since it is deterministic) in the numerical experiments with a one-level Lorenz 96 model. As we introduce fast scale variables, the evolution of slow variables in a two-level Lorenz

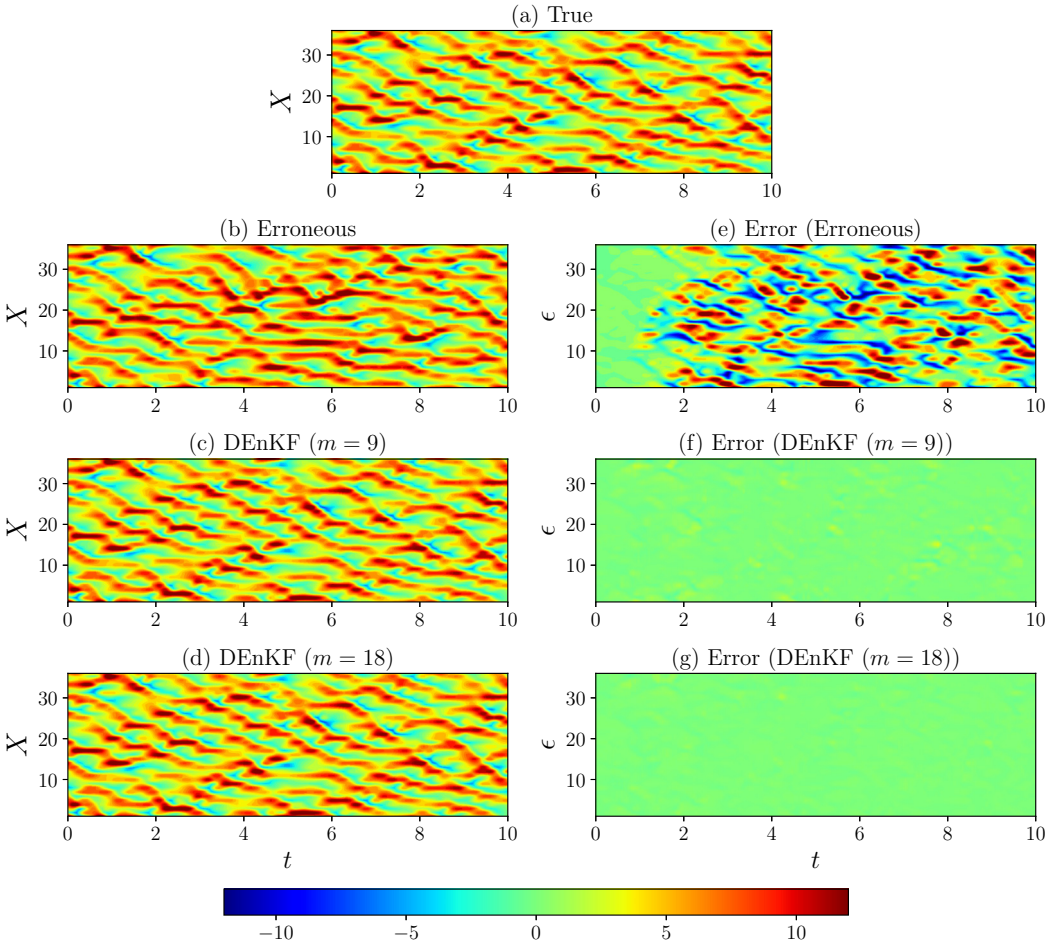

FIG. 10. Full state trajectory of the Lorenz 96 model with the the DEnKF algorithm.

96 model is no longer deterministic and simple Kalman filter-based algorithms might not be enough to give accurate prediction over a longer period.

[1] D. J. Stensrud, *Parameterization Schemes: Keys to Understanding Numerical Weather Prediction Models* (Cambridge University Press, Cambridge, UK, 2009).

[2] J. Duan and B. Nadiga, Stochastic parametrization for large eddy simulation of geophysical flows, Proc. Am. Math. Soc. **135**, 1187 (2007).

[3] D. A. Randall, Cloud parametrization for climate modeling: Status and prospects, Atmos. Res. **23**, 345 (1989).

[4] T. Schneider, S. Lan, A. Stuart, and J. Teixeira, Earth system modeling 2.0: A blueprint for models that learn from observations and targeted high-resolution simulations, Geophys. Res. Lett. **44**, 12 (2017).

[5] D. Draper, Assessment and propagation of model uncertainty, J. Roy. Statistic. Soc.: Ser. B (Methodol.) **57**, 45 (1995).

[6] C. E. Holloway and J. D. Neelin, Moisture vertical structure, column water vapor, and tropical deep convection, J. Atmos. Sci. **66**, 1665 (2009).

[7] C. Jakob, An improved strategy for the evaluation of cloud parametrizations in GCMs, Bull. Am. Meteorol. Soc. **84**, 1387 (2003).
[8] C. Jakob, Accelerating progress in global atmospheric model development through improved parametrizations: Challenges, opportunities, and strategies, Bull. Am. Meteorol. Soc. **91**, 869 (2010).
[9] M. Zhao, J.-C. Golaz, I. M. Held, V. Ramaswamy, S.-J. Lin, Y. Ming, P. Ginoux, B. Wyman, L. Donner, D. Paynter *et al.*, Uncertainty in model climate sensitivity traced to representations of cumulus precipitation microphysics, J. Clim. **29**, 543 (2016).
[10] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature **521**, 436 (2015).
[11] P. Neumann, P. Düben, P. Adamidis, P. Bauer, M. Brück, L. Kornblueh, D. Klocke, B. Stevens, N. Wedi, and J. Biercamp, Assessing the scales in numerical weather and climate predictions: Will exascale be the rescue? Philos. Trans. R. Soc., A **377**, 20180148 (2019).
[12] C. Kuenzer, M. Ottinger, M. Wegmann, H. Guo, C. Wang, J. Zhang, S. Dech, and M. Wikelski, Earth observation satellite sensors for biodiversity monitoring: Potentials and bottlenecks, Int. J. Remote Sens. **35**, 6599 (2014).
[13] Y. Liu, E. Racah, J. Correa, A. Khosrowshahi, D. Lavers, K. Kunkel, M. Wehner, W. Collins *et al.*, Application of deep convolutional neural networks for detecting extreme weather in climate datasets, arXiv:1605.01156.
[14] X. Shi, Z. Gao, L. Lausen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo, Deep learning for precipitation nowcasting: A benchmark and a new model, in *Proceedings of the Conference on Advances in Neural Information Processing Systems* (Curran Associates, Long Beach, CA, USA, 2017), pp. 5617–5627.
[15] E. de Bezenac, A. Pajot, and P. Gallinari, Deep learning for physical processes: Incorporating prior scientific knowledge, J. Stat. Mech. (2019) 124009.
[16] S. Rasp, M. S. Pritchard, and P. Gentine, Deep learning to represent subgrid processes in climate models, Proc. Natl. Acad. Sci. USA **115**, 9684 (2018).
[17] P. Gentine, M. Pritchard, S. Rasp, G. Reinaudi, and G. Yacalis, Could machine learning break the convection parametrization deadlock? Geophys. Res. Lett. **45**, 5742 (2018).
[18] S. L. Brunton, B. R. Noack, and P. Koumoutsakos, Machine learning for fluid mechanics, Annu. Rev. Fluid Mech. **52**, 477 (2020).
[19] M. P. Brenner, J. D. Eldredge, and J. B. Freund, Perspective on machine learning for advancing fluid mechanics, Phys. Rev. Fluids **4**, 100501 (2019).
[20] K. Duraisamy, G. Iaccarino, and H. Xiao, Turbulence modeling in the age of data, Annu. Rev. Fluid Mech. **51**, 357 (2019).
[21] F. Sarghini, G. De Felice, and S. Santini, Neural networks based subgrid-scale modeling in large eddy simulations, Computers & Fluids **32**, 97 (2003).
[22] M. Gamahara and Y. Hattori, Searching for turbulence models by artificial neural network, Phys. Rev. Fluids **2**, 054604 (2017).
[23] R. Maulik, H. Sharma, S. Patel, B. Lusch, and E. Jennings, A turbulent eddy-viscosity surrogate modeling framework for Reynolds-Averaged Navier-Stokes simulations, Comp. Fluids 104777 (2020).
[24] J. Ling, A. Kurzawski, and J. Templeton, Reynolds averaged turbulence modeling using deep neural networks with embedded invariance, J. Fluid Mech. **807**, 155 (2016).
[25] J.-X. Wang, J.-L. Wu, and H. Xiao, Physics-informed machine learning approach for reconstructing Reynolds stress modeling discrepancies based on DNS data, Phys. Rev. Fluids **2**, 034603 (2017).
[26] R. Maulik, O. San, A. Rasheed, and P. Vedula, Subgrid modeling for two-dimensional turbulence using neural networks, J. Fluid Mech. **858**, 122 (2019).
[27] A. Beck, D. Flad, and C.-D. Munz, Deep neural networks for data-driven LES closure models, J. Comput. Phys. **398**, 108910 (2019).
[28] C. Xie, J. Wang, and E. Weinan, Modeling subgrid-scale forces by spatial artificial neural networks in large eddy simulation of turbulence, Phys. Rev. Fluids **5**, 054606 (2020).
[29] C. Xie, J. Wang, K. Li, and C. Ma, Artificial neural network approach to large-eddy simulation of compressible isotropic turbulence, Phys. Rev. E **99**, 053113 (2019).

[30] P. Srinivasan, L. Guastoni, H. Azizpour, P. Schlatter, and R. Vinuesa, Predictions of turbulent shear flows using deep neural networks, Phys. Rev. Fluids **4**, 054603 (2019).
[31] J. Kim and C. Lee, Prediction of turbulent heat transfer using convolutional neural networks, J. Fluid Mech. **882**, A18 (2020).
[32] R. A. Heinonen and P. H. Diamond, Turbulence model reduction by deep learning, Phys. Rev. E **101**, 061201(R) (2020).
[33] G. Novati, H. L. de Laroussilhe, and P. Koumoutsakos, Automating turbulence modeling by multi-agent reinforcement learning, arXiv:2005.09023.
[34] K. Fukami, K. Fukagata, and K. Taira, Super-resolution reconstruction of turbulent flows with machine learning, J. Fluid Mech. **870**, 106 (2019).
[35] C. M. Jiang, S. Esmaeilzadeh, K. Azizzadenesheli, K. Kashinath, M. Mustafa, H. A. Tchelepi, P. Marcus, A. Anandkumar *et al.*, Meshfreeflownet: A physics-constrained deep continuous space-time super-resolution framework, arXiv:2005.01463.
[36] A. Subramaniam, M. L. Wong, R. D. Borker, S. Nimmagadda, and S. K. Lele, Turbulence enrichment using physics-informed generative adversarial networks, arXiv:2003.01907.
[37] J. Pathak, B. Hunt, M. Girvan, Z. Lu, and E. Ott, Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach, Phys. Rev. Lett. **120**, 024102 (2018).
[38] P. Vlachas, J. Pathak, B. Hunt, T. Sapsis, M. Girvan, E. Ott, and P. Koumoutsakos, Backpropagation algorithms and reservoir computing in recurrent neural networks for the forecasting of complex spatiotemporal dynamics, Neural Netw. **126**, 191 (2020).
[39] Z. Y. Wan, P. Vlachas, P. Koumoutsakos, and T. Sapsis, Data-assisted reduced-order modeling of extreme events in complex dynamical systems, PLoS ONE **13**, e0197704 (2018).
[40] K. Lee and K. Carlberg, Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders, J. Comput. Phys. **404**, 108973 (2020).
[41] A. Mohan, D. Daniel, M. Chertkov, and D. Livescu, Compressed convolutional LSTM: An efficient deep learning framework to model high fidelity 3D turbulence, arXiv:1903.00033.
[42] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox, Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems, Physica D **406**, 132401 (2020).
[43] S. M. Rahman, S. Pawar, O. San, A. Rasheed, and T. Iliescu, Nonintrusive reduced-order modeling framework for quasigeostrophic turbulence, Phys. Rev. E **100**, 053306 (2019).
[44] R. Wang, K. Kashinath, M. Mustafa, A. Albert, and R. Yu, Towards physics-informed deep learning for turbulent flow prediction, in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Association for Computing Machinery, New York, NY, 2020), pp. 1457–1466.
[45] T. Arcomano, I. Szunyogh, J. Pathak, A. Wikner, B. R. Hunt, and E. Ott, A machine learning-based global atmospheric forecast model, Geophys. Res. Lett. **47**, e2020GL087776 (2020).
[46] M. Cheng, F. Fang, C. C. Pain, and I. M. Navon, Data-driven modeling of nonlinear spatio-temporal fluid flows using a deep convolutional generative adversarial network, Comput. Methods Appl. Mech. Eng. **365**, 113000 (2020).
[47] J. H. Faghmous, A. Banerjee, S. Shekhar, M. Steinbach, V. Kumar, A. R. Ganguly, and N. Samatova, Theory-guided data science for climate change, Computer **47**, 74 (2014).
[48] N. Wagner and J. M. Rondinelli, Theory-guided machine learning in materials science, Front. Mater. **3**, 28 (2016).
[49] M. Reichstein, G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais *et al.*, Deep learning and process understanding for data-driven Earth system science, Nature **566**, 195 (2019).
[50] M. Raissi, P. Perdikaris, and 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**, 686 (2019).
[51] M. Raissi and G. E. Karniadakis, Hidden physics models: Machine learning of nonlinear partial differential equations, J. Comput. Phys. **357**, 125 (2018).
[52] J.-L. Wu, K. Kashinath, A. Albert, D. Chirila, H. Xiao *et al.*, Enforcing statistical constraints in

generative adversarial networks for modeling chaotic dynamical systems, J. Comput. Phys. **406**, 109209 (2020).
[53] N. B. Erichson, L. Mathelin, Z. Yao, S. L. Brunton, M. W. Mahoney, and J. N. Kutz, Shallow neural networks for fluid flow reconstruction with limited sensors, Proc. R. Soc. A **476**, 20200097 (2020).
[54] A. T. Mohan, N. Lubbers, D. Livescu, and M. Chertkov, Embedding hard physical constraints in neural network coarse-graining of 3D turbulence, arXiv:2002.00021.
[55] P. Márquez-Neila, M. Salzmann, and P. Fua, Imposing hard constraints on deep networks: Promises and limitations, arXiv:1706.02025.
[56] A. Karpatne, G. Atluri, J. H. Faghmous, M. Steinbach, A. Banerjee, A. Ganguly, S. Shekhar, N. Samatova, and V. Kumar, Theory-guided data science: A new paradigm for scientific discovery from data, IEEE Trans. Knowl. Data Eng. **29**, 2318 (2017).
[57] S. Rasp, Coupled online learning as a way to tackle instabilities and biases in neural network parametrizations: General algorithms and Lorenz 96 case study (v1.0), Geosci. Model Dev. **13**, 2185 (2020).
[58] N. D. Brenowitz and C. S. Bretherton, Spatially extended tests of a neural network parametrization trained by coarse-graining, J. Adv. Model. Earth Syst. **11**, 2728 (2019).
[59] N. D. Brenowitz and C. S. Bretherton, Prognostic validation of a neural network unified physics parameterization, Geophys. Res. Lett. **45**, 6289 (2018).
[60] J. Wu, H. Xiao, R. Sun, and Q. Wang, Reynolds-averaged Navier–Stokes equations with explicit data-driven Reynolds stress closure can be ill-conditioned, J. Fluid Mech. **869**, 553 (2019).
[61] J. M. Lewis, S. Lakshmivarahan, and S. Dhall, *Dynamic Data Assimilation: A Least Squares Approach*, Vol. 104 (Cambridge University Press, Cambridge, 2006).
[62] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches* (John Wiley & Sons, New York, 2006).
[63] G. Evensen, *Data Assimilation: The Ensemble Kalman Filter* (Springer Science & Business Media, Berlin, 2009).
[64] D. Xiao, J. Du, F. Fang, C. Pain, and J. Li, Parameterised nonintrusive reduced-order methods for ensemble Kalman filter data assimilation, Comput. Fluids **177**, 69 (2018).
[65] C. Zerfas, L. G. Rebholz, M. Schneier, and T. Iliescu, Continuous data assimilation reduced-order models of fluid flow, Comput. Methods Appl. Mech. Eng. **357**, 112596 (2019).
[66] R. Arcucci, L. Mottet, C. Pain, and Y.-K. Guo, Optimal reduced space for variational data assimilation, J. Comput. Phys. **379**, 51 (2019).
[67] J. W. Labahn, H. Wu, S. R. Harris, B. Coriton, J. H. Frank, and M. Ihme, Ensemble Kalman filter for assimilating experimental data into large-eddy simulations of turbulent flows, Flow, Turbul. Combust. **104**, 861 (2020).
[68] V. Mons, J.-C. Chassaing, T. Gomez, and P. Sagaut, Reconstruction of unsteady viscous flows using data assimilation schemes, J. Comput. Phys. **316**, 255 (2016).
[69] A. F. da Silva and T. Colonius, Ensemble-based state estimator for aerodynamic flows, AIAA J. **56**, 2568 (2018).
[70] C. Colburn, J. Cessna, and T. Bewley, State estimation in wall-bounded flow systems. Part 3. The ensemble Kalman filter, J. Fluid Mech. **682**, 289 (2011).
[71] X. Li, F. Liu, and M. Fang, Harmonizing models and observations: Data assimilation for earth system science, Sci. China Earth Sci. **63**, 1059 (2020).
[72] R. Lguensat, P. Tandeo, P. Ailliot, M. Pulido, and R. Fablet, The analog data assimilation, Mon. Weather Rev. **145**, 4093 (2017).
[73] M. Tang, Y. Liu, and L. J. Durlofsky, A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems, J. Comput. Phys. **413**, 109456 (2020).
[74] M. Bocquet, J. Brajard, A. Carrassi, and L. Bertino, Bayesian inference of chaotic dynamics by merging data assimilation, machine learning and expectation-maximization, Found. Data Sci. **2**, 55 (2020).
[75] J. Brajard, A. Carassi, M. Bocquet, and L. Bertino, Combining data assimilation and machine learning to emulate a dynamical model from sparse and noisy observations: A case study with the Lorenz 96 model, J. Comput. Sci. **44**, 101171 (2020).

[76] E. N. Lorenz, Predictability: A problem partly solved, in *Proceedings of the Seminar on Predictability*, Vol. 1 (ECMWF, Shinfield Park, Reading, 1996).
[77] K. J. H. Law, D. Sanz-Alonso, A. Shukla, and A. M. Stuart, Filter accuracy for the Lorenz 96 model: Fixed versus adaptive observation operators, Physica D **325**, 1 (2016).
[78] A. Karimi and M. R. Paul, Extensive chaos in the Lorenz-96 model, Chaos: An interdisciplinary J. Nonlin. Sci. **20**, 043105 (2010).
[79] S. Herrera, D. Paz ó, J. Ferná Ndez, and M. A. Rodríguez, The role of large-scale spatial patterns in the chaotic amplification of perturbations in a Lorenz'96 model, Tellus A: Dynam. Meteorol. Oceanogr. **63**, 978 (2011).
[80] D. L. van Kekem and A. E. Sterk, Symmetries in the Lorenz-96 model, Int. J. Bifurcat. Chaos **29**, 1950008 (2019).
[81] R. H. Kraichnan, Inertial ranges in two-dimensional turbulence, Phys. Fluids **10**, 1417 (1967).
[82] T. N. Palmer, A nonlinear dynamical perspective on model error: A proposal for nonlocal stochastic-dynamic parametrization in weather and climate prediction models, Quart. J. Roy. Meteorol. Soc. **127**, 279 (2001).
[83] D. S. Wilks, Effects of stochastic parametrizations in the Lorenz'96 system, Quart. J. Roy. Meteorol. Soc. **131**, 389 (2005).
[84] D. Crommelin and E. Vanden-Eijnden, Subgrid-scale parametrization with conditional markov chains, J. Atmos. Sci. **65**, 2661 (2008).
[85] G. Vissio and V. Lucarini, A proof of concept for scale-adaptive parametrizations: The case of the lorenz'96 model, Quart. J. Roy. Meteorol. Soc. **144**, 63 (2018).
[86] A. Wikner, J. Pathak, B. Hunt, M. Girvan, T. Arcomano, I. Szunyogh, A. Pomerance, and E. Ott, Combining machine learning with knowledge-based modeling for scalable forecasting and subgrid-scale closure of large, complex, spatiotemporal systems, Chaos **30**, 053111 (2020).
[87] F. Bouchet and A. Venaille, Statistical mechanics of two-dimensional and geophysical flows, Phys. Rep. **515**, 227 (2012).
[88] G. Boffetta and R. E. Ecke, Two-dimensional turbulence, Annu. Rev. Fluid Mech. **44**, 427 (2012).
[89] J. S. Frederiksen, T. J. O'Kane, and M. J. Zidikheri, Subgrid modeling for geophysical flows, Philos. Trans. R. Soc. A **371**, 20120166 (2013).
[90] J. Smagorinsky, General circulation experiments with the primitive equations: I. The basic experiment, Mon. Weather Rev. **91**, 99 (1963).
[91] C. Leith, Atmospheric predictability and two-dimensional turbulence, J. Atmos. Sci. **28**, 145 (1971).
[92] J. S. Frederiksen and S. M. Kepert, Dynamical subgrid-scale parametrizations from direct numerical simulations, J. Atmos. Sci. **63**, 3006 (2006).
[93] C. Eden and R. J. Greatbatch, Towards a mesoscale eddy closure, Ocean Model. **20**, 223 (2008).
[94] O. San, A. E. Staples, and T. Iliescu, Approximate deconvolution large eddy simulation of a stratified two-layer quasigeostrophic ocean model, Ocean Model. **63**, 1 (2013).
[95] R. Maulik and O. San, A stable and scale-aware dynamic modeling framework for subgrid-scale parametrizations of two-dimensional turbulence, Comput. Fluids **158**, 11 (2017).
[96] S. Pawar, O. San, A. Rasheed, and P. Vedula, *A priori* analysis on deep learning of subgrid-scale parametrizations for Kraichnan turbulence, Theor. Comput. Fluid Dyn. **34**, 429 (2020).
[97] A. Pal, Deep learning parametrization of subgrid scales in wall-bounded turbulent flows, arXiv:1905.12765.
[98] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv:1412.6980.
[99] X. I. A. Yang, S. Zafar, J.-X. Wang, and H. Xiao, Predictive large-eddy-simulation wall modeling via physics-informed neural networks, Phys. Rev. Fluids **4**, 034602 (2019).
[100] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE **86**, 2278 (1998).
[101] R. Maulik, R. Egele, B. Lusch, and P. Balaprakash, Recurrent neural network architecture search for geophysical emulation, arXiv:2004.10928.

[102] R. Liaw, E. Liang, R. Nishihara, P. Moritz, J. E. Gonzalez, and I. Stoica, Tune: A research platform for distributed model selection and training, arXiv:1807.05118.
[103] A. Gelb, *Applied Optimal Estimation* (MIT Press, Cambridge, MA, 1974).
[104] G. Welch and G. Bishop, An introduction to the Kalman filter, Univ. North Carolina, Chapel Hill, NC, USA, Lecture, 2001.
[105] G. Burgers, P. Jan van Leeuwen, and G. Evensen, Analysis scheme in the ensemble Kalman filter, Mon. Weather Rev. **126**, 1719 (1998).
[106] R. E. Kalman, A new approach to linear filtering and prediction problems, J. Basic Eng. **82**, 35 (1960).
[107] J. L. Anderson and S. L. Anderson, A Monte Carlo implementation of the nonlinear filtering problem to produce ensemble assimilations and forecasts, Mon. Weather Rev. **127**, 2741 (1999).
[108] J. L. Anderson, An adaptive covariance inflation error correction algorithm for ensemble filters, Tellus A: Dynam. Meteorol. Oceanogr. **59**, 210 (2007).
[109] A. Attia and E. Constantinescu, An optimal experimental design framework for adaptive inflation and covariance localization for ensemble filters, arXiv:1806.10655.
[110] P. Sakov and P. R. Oke, A deterministic formulation of the ensemble Kalman filter: An alternative to ensemble square root filters, Tellus A: Dynam. Meteorol. Oceanogr. **60**, 361 (2008).
[111] W. Li, W. S. Rosenthal, and G. Lin, Trimmed ensemble Kalman filter for nonlinear and non-Gaussian data assimilation problems, arXiv:1808.05465.
[112] J. L. Anderson, A non-Gaussian ensemble filter update for data assimilation, Mon. Weather Rev. **138**, 4186 (2010).
[113] A. Apte, M. Hairer, A. M. Stuart, and J. Voss, Sampling the posterior: An approach to non-Gaussian data assimilation, Physica D **230**, 50 (2007).
[114] M. Zupanski, Maximum likelihood ensemble filter: Theoretical aspects, Mon. Weather Rev. **133**, 1710 (2005).
[115] A. Carrassi, S. Vannitsem, D. Zupanski, and M. Zupanski, The maximum likelihood ensemble filter performances in chaotic systems, Tellus A: Dynam. Meteorol. Oceanogr. **61**, 587 (2008).
[116] E. D. Nino-Ruiz, A. Mancilla-Herrera, S. Lopez-Restrepo, and O. Quintero-Montoya, A maximum likelihood ensemble filter via a modified Cholesky decomposition for non-Gaussian data assimilation, Sensors **20**, 877 (2020).
[117] A. Arakawa, Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow. Part I, J. Comput. Phys. **135**, 103 (1997).
[118] S. Gottlieb and C.-W. Shu, Total variation diminishing Runge-Kutta schemes, Math. Comput. **67**, 73 (1998).
[119] O. San and A. E. Staples, High-order methods for decaying two-dimensional homogeneous isotropic turbulence, Comput. Fluids **63**, 105 (2012).
[120] S. Pawar, S. Rahman, H. Vaddireddy, O. San, A. Rasheed, and P. Vedula, A deep learning enabler for nonintrusive reduced-order modeling of fluid flows, Phys. Fluids **31**, 085101 (2019).
[121] A. A. Popov, A. Sandu, E. D. Nino-Ruiz, and G. Evensen, A stochastic covariance shrinkage approach in ensemble transform Kalman filtering, arXiv:2003.00354.
[122] P. L. Houtekamer and F. Zhang, Review of the ensemble Kalman filter for atmospheric data assimilation, Mon. Weather Rev. **144**, 4489 (2016).
[123] M. Germano, U. Piomelli, P. Moin, and W. H. Cabot, A dynamic subgrid-scale eddy viscosity model, Phys. Fluids A **3**, 1760 (1991).
[124] D. K. Lilly, A proposed modification of the Germano subgrid-scale closure method, Phys. Fluids A **4**, 633 (1992).
[125] S. Grossmann and P. Mertens, Structure functions in two-dimensional turbulence, Z. Phys. B **88**, 105 (1992).
[126] S. Pawar, Ml-parameterization, https://github.com/surajp92/ML-Parameterization (2020).