JAMES Journal of Advances in Modeling Earth Systems®

RESEARCH ARTICLE
10.1029/2023MS003795

Special Section:
Machine learning application to Earth system modeling

Turbulence Closure With Small, Local Neural Networks: Forced Two-Dimensional and β-Plane Flows

Kaushik Srinivasan[1], **Mickaël D. Chekroun**[1,2], **and James C. McWilliams**[1]

[1]Department of Atmospheric and Oceanic Sciences, University of California, Los Angeles, CA, USA, [2]Department of Earth and Planetary Sciences, Weizmann Institute of Science, Rehovot, Israel

Key Points:

- Shallow convolutional neural networks (CNNs) accurately parameterize high Reynolds number forced2D turbulence, with efficient training and high online large eddy simulations accuracy
- Extensive hyperparameter searching and cyclical learning rates yield robust and accurate online solutions for CNN-based turbulence models
- The success of shallow CNNs implies nearly-local dependence of SGS stresses providing insights into turbulent flow interactions

Supporting Information:

Supporting Information may be found in the online version of this article.

Correspondence to:
K. Srinivasan,
kaushiks@atmos.ucla.edu

Citation:
Srinivasan, K., Chekroun, M. D., & McWilliams, J. C. (2024). Turbulence closure with small, local neural networks: Forced two-dimensional and β-plane flows. *Journal of Advances in Modeling Earth Systems*, *16*, e2023MS003795. https://doi.org/10.1029/2023MS003795

Received 11 JULY 2023
Accepted 15 FEB 2024

Author Contributions:
Conceptualization: Kaushik Srinivasan, Mickaël D. Chekroun, James C. McWilliams
Funding acquisition: Mickaël D. Chekroun, James C. McWilliams

Abstract We parameterize sub-grid scale (SGS) fluxes in sinusoidally forced two-dimensional turbulence on the β-plane at high Reynolds numbers (Re ~25,000) using simple 2-layer convolutional neural networks (CNN) having only O(1000) parameters, two orders of magnitude smaller than recent studies employing deeper CNNs with 8–10 layers; we obtain stable, accurate, and long-term online or a posteriori solutions at 16× downscaling factors. Our methodology significantly improves training efficiency and speed of online large eddy simulations runs, while offering insights into the physics of closure in such turbulent flows. Our approach benefits from extensive hyperparameter searching in learning rate and weight decay coefficient space, as well as the use of cyclical learning rate annealing, which leads to more robust and accurate online solutions compared to fixed learning rates. Our CNNs use either the coarse velocity or the vorticity and strain fields as inputs, and output the two components of the deviatoric stress tensor, $\boldsymbol{S}_d$. We minimize a loss between the SGS vorticity flux divergence (computed from the high-resolution solver) and that obtained from the CNN-modeled $\boldsymbol{S}_d$, without requiring energy or enstrophy preserving constraints. The success of shallow CNNs in accurately parameterizing this class of turbulent flows implies that the SGS stresses have a weak non-local dependence on coarse fields; it also aligns with our physical conception that small-scales are locally controlled by larger scales such as vortices and their strained filaments. Furthermore, 2-layer CNN-parameterizations are more likely to be interpretable.

Plain Language Summary In this study, we demonstrate that simple, shallow neural networks can be used to effectively model complex turbulent flows in the atmosphere and oceans. By using these simpler NNs, we can improve the efficiency of our simulations and better understand the underlying physics of turbulent flows. We also explore different training techniques to make these models more accurate and robust. Our findings suggest that the stress in these turbulent flows has only a weak spatial dependence on larger-scale features, which has important implications for our understanding of how turbulence behaves. Overall, our work can help improve climate and weather models by providing a more efficient and interpretable way to simulate turbulence.

1. Introduction

Turbulent flows in physical systems span a vast range of spatial and temporal scales; in the Earth's oceans, for example, mesoscale eddies, the dominant reservoir of kinetic energy in the ocean, have scales of O(100 km) whereas, small-scale three dimensional motions at the air-sea interface driven by a combination of surface heating/cooling, wind action and the Earth's rotation have scales of O(1m). This vast range of spatial scales is well beyond the range of current numerical ocean models to solve given even the largest available compute facilities. Climate models face even greater challenges because they need to be run for years or decades to study climatic changes over long temporal horizons. In practice, these models are run at a certain resolution limited by available computational resources while unresolved turbulent processes are effectively represented or *parameterized*. The turbulent motions that are below the resolution of numerical models and need parameterization or representation are commonly referred to as subgrid-scale (SGS) motions, with their effect being expressed as effective fluxes (SGS fluxes) that can be added to the equations of motion already solved by numerical models.

Historically, parameterization of unresolved turbulent flows are computed through a combination of empirical data, physics-based modeling and simplified algebraic models computed using high resolution process models run over short durations of time. For example, three-dimensional small-scale turbulence near the ocean surface is parameterized using the K-profile parameterization (Large et al., 1994) framework which is a ad hoc combination

Investigation: Kaushik Srinivasan, Mickaël D. Chekroun, James C. McWilliams
Methodology: Kaushik Srinivasan
Visualization: Kaushik Srinivasan, Mickaël D. Chekroun
Writing – original draft: Kaushik Srinivasan
Writing – review & editing: Kaushik Srinivasan, Mickaël D. Chekroun, James C. McWilliams

of empirical and physics-based approaches that model various turbulent process found at the air-sea interface, including surface convection and boundary layer rotating-stratified shear turbulence. Parameterizations are also found through numerical process studies (essentially idealized numerical simulations of a specific phenomenon), typically high-resolution large eddy simulations (LES) to fit simplified algebraic models based on combination of dimensional analysis and physical models (Souza et al., 2020).

An alternative approach spurred on by the development of machine learning methods, in particular neural network (NN) based Deep Learning methods, along with the availability of large amount of data from numerical simulations, is to use high resolution process studies to generate SGS fluxes that can be accurately modeled without recourse to simplified algebraic models. With the availability of frameworks that allow incorporating NN models trained in the Python programming language into Fortran (Curcic, 2019; Ott et al., 2020), the language of most weather, ocean and climate models, one can in principle accurately model SGS fluxes in climate models, substantially adding to their predicability and fidelity and reducing their biases. In practice, a major issue that manifests is that the numerical models that incorporate NN-based models are subjected to challenges in numerical stability (Brenowitz et al., 2020).

Neural networks are hierarchical non-linear functions of simpler modular units containing a large number of tunable-parameters that can in principle represent any arbitrary non-linear function given sufficient data (Hornik et al., 1989; D.-X. Zhou, 2020). Typically complex models like these can be subject to over-fitting to a given data set and not learn general relationships that are actually present in the data. However, a multitude of regularization techniques have been discovered over the past decade that allow NNs to learn non-linear relationships that generalize well across unseen data (Ba et al., 2016; Ioffe & Szegedy, 2015; Loshchilov & Hutter, 2016; Srivastava et al., 2014). Applications of NNs to dynamical modeling, in particular to the solution of complex dynamical equations that govern the climate system, face the additional requirements that solutions of equations be numerically stable and also that they not accrue unphysical biases over long time horizons.

NN-based parameterizations in dynamical systems can classified based on how the learning framework is designed. Online learning (Ma et al., 2019; Rasp, 2020) directly incorporates the NN-parameterization into the equations of motion; the equations of motions are time-stepped forward and the parameters of the NN are learnt subject to the constraint that NN-driven flow trajectory matches the outputs of a high resolution numerical solution truncated to the resolution of the NN-based solutions. This approach, a form of *trajectory optimization* that is common in imitation learning (Hussein et al., 2017) and has the advantage that the NN-driven solution is physics-aware, promises that well optimized trajectories are not subject to biases not present in the high resolution numerical solution. A drawback, however, is that the numerical framework needs to be entirely differentiable because the errors in the trajectory need to be propagated through the numerical solver itself; this idea being closely related to adjoint models in data assimilation. Differentiable numerical models, for example, in weather and climate systems, cannot be created by simple modification of existing climate and weather models and need to be rewritten from scratch, a complex undertaking. Furthermore, for complex nonlinear dynamical systems, the trajectories need to be rolled out for multiple time steps and multi-step losses need to be optimized to ensure accuracy and stability of the learned NN-based equations (Frezat et al., 2022; Keisler, 2022; Kochkov et al., 2021; List et al., 2022); this can substantially add to the computational burden and complexity of the training pipeline.

Offline or a priori learning uses high resolution numerical models to diagnose all the SGS fluxes that are missing at lower resolutions. Then the diagnosed SGS fluxes are directly modeled using an appropriate NN architecture which takes in the coarse-grid flow variables as inputs; the NN-training is accomplished using standard supervised training (through Maximum Likelihood Estimation). Methodologically this is a substantially simpler approach and places no restrictions on the numerical solver itself as the numerical solution and the learning of the NN are decoupled. Consequently successfully trained NN-models can be directly incorporated into existing climate and weather models. Correspondingly, however, a disadvantage here is that the NN model has low awareness of the dynamics of the underlying system and small errors made by the NN SGS model can accumulate when incorporated in the numerical solver, resulting in numerical instabilities or biased solutions, the issue becoming more acute the longer the numerical model is run (Brenowitz et al., 2020; Frezat et al., 2022; Rasp, 2020). Some of the numerical stability issues can be alleviated through probabilistic modeling of the SGS fluxes instead of a fixed NN-model (Guillaumin & Zanna, 2021; Perezhogin et al., 2023).

Both offline and online-trained models must be tested for online or a posteriori stability and accuracy over justifiably long time horizons. In other words, the training or learning process can be operated offline (i.e., direct supervised learning) or online, using a differentiable numerical solver, but all trained models must be deployed on the coarse-grained equations and tested for online stability and fidelity. We re-emphasize the distinction between the offline or online-learning process from actual deployment in the equations which is, by definition, always online or a posteriori. Both offline and online NN-based learning approaches have other challenges, in particular, that NNs are optimized for Graphic Processing Units (GPUs) which most climate system models are not currently designed to run on. In particular modern Deep NNs can have millions of parameters and running them on CPUs can add a substantial overhead to the numerical solver, possibly negating the effects of the gains in building accurate SGS models.

1.1. Related Work

Decaying two dimensional turbulence is one of the most common model turbulent flow problems for discovery and testing of data-driven closure and parameterization (Maulik et al., 2019). The physics of decaying turbulence is well known (Brachet et al., 1988; Carnevale et al., 1991, 1992; McWilliams, 1984) and consists of a slow merger and interaction of vortices to form larger and larger vortices until the entire flow domain consists of a single large vortex. Recent studies have shown that even at high Reynolds numbers (Re ∼20,000), the SGS stresses in this problem can be accurately modeled with a modest number of data samples (Frezat et al., 2022; Guan, Chattopadhyay, et al., 2022; List et al., 2022), leading to accurate and stable online coarse-grid solutions at 16× or greater grid downscaling factors.

Continuously forced 2D-turbulence, however, is a substantially more complex dynamical problem (Xiao et al., 2009) that has a persistent dual cascade of energy, an inverse energy cascade from forcing scales to a larger scales and a forward enstrophy cascade to smaller scales (Kraichnan, 1971). Only a handful of recent studies over the past 2 years have successfuly obtained accurate stable online closures for this and related problems. Guan, Subel, et al. (2022) studied the problem of sinusoidally forced 2D turbulent flow solved in a doubly-periodic channel, also referred to as Kolmogorov flow in classical fluid mechanics, and demonstrated that offline learning could be effectively used to model SGS-stress using deep convolutional neural networks (CNNs) (having 11-layers, about 900,000 parameters) that resulted in stable online closures, provided sufficient amount of training data was generated, by using around 2 million high resolution model time-steps. In practice they find that it is sufficient to train on only 2000 snapshots, chosen every thousand time-steps to ensure that the snapshots were uncorrelated with each other. Guan, Subel, et al. (2022) also found that exploiting the geometrical symmetry of the doubly-periodic computational domain using rotationally equivariant convolutions substantially reduced their data requirements by a factor of 40. In a more recent study, Ross et al. (2023) considered the offline closure of a related idealized dynamical problem relevant to geophysical flows, namely two-layer quasi-geostrophic flow on the β-plane; for a brief description of the β-plane approximation, see Section 2.1. In their wide ranging study, Ross et al. (2023) examined how various choices for the inputs whether coarse-grained velocities or velocity gradients, the form of the output SGS-fields and the precise choice of the downscaling affected the online accuracy of their offline-trained CNNs having 8-layers, around 300,000 parameters.

Unlike the standard approach of parameterizing the SGS stresses using NNs, Kochkov et al. (2021) parameterized the nonlinear advection term directly through a "neural discretization" method demonstrated in simpler dynamical problems in earlier papers. The Kochkov et al. (2021) approach was trained in entirely online fashion through trajectory optimization. A follow up paper (Dresdner et al., 2022) by the same group found accurate online-trained parameterization of the same problem solved using spectral methods (instead of a finite-volume approach) though using a direct SGS-parameterizing CNN instead of the neural-discretization approach used in (Kochkov et al., 2021). A closely related study is by Frezat et al. (2022) which also employed online-learned parameterization of forced β-plane turbulence along with standard two-dimensional turbulence, but at a substantially higher Reynolds number than (Dresdner et al., 2022). Both Dresdner et al. (2022) and Frezat et al. (2022) used deep CNNs with 8–16 layers, the former work employing the encoder-process-decoder models now gaining prominence in neural turbulent forecasting models (Keisler, 2022; Stachenfeld et al., 2021).

1.2. Current Work

Our approach, presented in this study, derives from the aforementioned studies on the methodology of parameterizing sinusoidally forced two-dimensional and geostrophic turbulence on the β-plane at high Reynolds numbers (Re ∼25,000) using purely offline training of the SGS stress. However, there are substantial departures between the choices made here which lead to significant improvements in training and efficiency of the turbulent closures. These points are not only technical, but also lead to new insights into the physical and mathematical aspects of closure problems of such turbulent flows. We provide below a list that summarizes the main contributions of this study.

1. *Accurate and efficient shallow CNN-closures at high Re*. We demonstrate that simple *two-layer* CNNs are sufficient to obtain stable and long-term accurate online solutions at 16× downscaling for high Re-turbulence. The resulting CNN-parameterizations of the SGS flux contain only $\mathcal{O}(10^3)$ parameters; two orders of magnitude smaller than aforementioned recent studies, leading in turn to substantially faster and efficient online CNN-LES runs.
2. *Hyperparameter space probing.* We find the best models through extensive hyperparameter searching in the learning rate and weight decay coefficient involved in the optimization of the loss function given by Equation 28 below. This probing operation is greatly facilitated by the choice of small CNNs used here.
3. *Optimization strategy.* We show that the CNNs trained using cyclical learning rate annealing result in more robust and accurate online solutions compared to those trained using a fixed learning rate, as traditionally operated.
4. *Physical* variables. Our inputs to the CNN use either the coarse velocity field, $(\bar{u},\bar{v})$ or the vorticity and strain fields, $(\bar{\omega},\bar{\sigma}_s,\bar{\sigma}_n)$ and outputs are the two components of the deviatoric stress tensor; we do not consider $(\bar{\psi},\bar{\omega})$ inputs because the streamfunction $\bar{\psi}$ is difficult to obtain in numerical models over complex spatial domains.
5. *Loss function.* Our CNNs are optimized by minimizing a mean square error loss function that measures their parameterization defect with respect to the SGS expressed in terms of the deviatoric stress tensor; see Equation 28 below. In this study we do not need to use more elaborate physics-informed loss functions that have been used in recent work (Guan, Subel, et al., 2022; List et al., 2022) although such constraints might help in smaller sizes of training data compared to those used in this work.
6. *Weakly non-local parameterizations and physical implications.* The success of shallow CNNs in accurately parameterizing turbulence for the class of turbulence problems considered here has direct implications for our understanding of the physics of the problem itself. First, since the spatial nonlocality of CNNs grows with depth, our closure results with shallow CNNs imply that the SGS stresses have only a weak non-local dependence on coarse fields. Such a weak spatial non-locality of our shallow CNN-parameterizations also means that they are more amenable for embedding into climate models typically relying on spatial domain decomposition techniques for computation on large clusters; correspondingly deep CNNs would require substantially larger data exchange between computational nodes.

Finally, recalling that deep NNs with a great amount of parameters are typically required to approximate complicated nonlinear functions, due to the shallowness of our CNNs, we infer that the SGS stress must have here a relatively simple nonlinear dependence on the coarse flow fields. This simple observation provides a favorable ground for the nonlinear mapping underlying our shallow CNN-parameterization (Π_{CNN} in Equation 38 below) to be interpretable, which we leave for a future work. The relative simplicity of the recent analytical operator forms obtained by Ross et al. (2023) using their hybrid genetic programming combined with a sparse linear regression approach is consistent with our findings.

2. Turbulence Models and Data Generation

2.1. Dynamical Equations and Turbulent Regimes

We use a popular model fluid flow problem that exhibits complex turbulence phenomena and is closely related to large scale turbulence in the Earth system and in planetary atmospheres–namely sinusoidally forced two-dimensional turbulence on the β-plane. The flow domain is specified in (x, y)-coordinates and consists of a square domain of size (L, L), with $L = 2\pi$. The governing equations of motion are the Navier-Stokes (N-S) equations which describe the evolution of flow velocity vector field, $\boldsymbol{v}(x, y, t) = (u(x, y, t), v(x, y, t))$, at every point in the domain and in time. For the specific problem chosen here, the boundary conditions are chosen to be periodic, namely.

$$u(x,y,t) = u(x+L,y+L,t), \tag{1}$$

$$v(x,y,t) = v(x+L,y+L,t). \tag{2}$$

Note that this domain is topologically equivalent to a two-dimensional torus. In two dimensions the N-S equations in the presence of background rotation, are the evolution equations of the two components of the velocity fields, $\boldsymbol{v} \equiv (u, v)$ where u and v are the velocities along x- and y-directions, respectively. In non-dimensional form, this becomes

$$\partial_t \boldsymbol{v} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} + f(\boldsymbol{k} \times \boldsymbol{v}) = -\mu \boldsymbol{v} + \frac{1}{Re}\nabla^2 \boldsymbol{v} + \mathcal{F}_v(x,y). \tag{3}$$

Here, f, the Coriolis parameter is the local rotation rate of the Earth or some other planetary atmosphere, $\boldsymbol{k}$ is the unit vector normal to the (x, y)-plane while $_v(x,y) = \left[-\sin\left(k_f y\right), \sin\left(k_f x\right)\right]$ is the sinusoidal time-invariant forcing field that continuously drives the flow; this specific form is chosen to be the same as in (Guan, Subel, et al., 2022) with the forcing wavenumber, $k_f = 4$. The coefficient μ represents the linear drag coefficient that in geophysical flows purports to represent the effect of bottom friction and Re is the Reynolds number measuring the strength of the non-linear advection term relative to the viscous term (the second term on the left hand side relative to the second term on the right-hand side (RHS)); we choose Re = 25,000.

The velocity field is also required to satisfy mass conservation captured through the continuity equation, $\nabla \cdot \boldsymbol{v} = \partial_x u + \partial_y v = 0$. The continuity equation can be implicitly satisfied in two dimensions by defining a scalar field called streamfunction, $\psi(x, y, t)$

$$u = -\psi_y, \qquad v = \psi_x \tag{4}$$

In two dimensions, these two equations along with the continuity constraint can be expressed as an evolution equation of a single scalar field, the vorticity, that is defined as the two-dimensional curl of the velocity field

$$\omega = \partial_x v - \partial_y u = \nabla^2 \psi, \tag{5}$$

and captures the local rotation of a fluid parcel. The vorticity evolution equation in two dimensions can then be written in the standard vorticity-streamfunction (ω, ψ) form as

$$\partial_t \omega + J(\psi,\omega) + \beta v = -\mu\omega + \frac{1}{Re}\nabla^2\omega + F_\omega(x,y), \tag{6}$$

where the Jacobian, $J(\psi, \omega) = \psi_x \omega_y - \omega_x \psi_y$ captures the non-linear advection term, while the vortical form of the forcing becomes

$$F_\omega(x,y) = k_f\left[\cos\left(k_f x\right) + \cos\left(k_f y\right)\right] \tag{7}$$

The βv-term is an approximation relevant to geophysical systems and captures the effect of differential rotation experienced by the ocean and atmosphere in the Earth system in a tangent plane approximation ($\beta = \frac{\partial f}{\partial y}$). The presence of the β-term allows the flow to manifest specific kind of waves common in planetary systems, called Rossby waves which can substantially alter the turbulent flow dynamics relative to the flow which results when $\beta = 0$.

Equation 6 is solved in Fourier space using a standard pseudospectral method (Orszag & Israeli, 1974); a semi-implicit Crank-Nicholson, 2nd-order Adams-Bashforth (CN-AB2) method is used for time-stepping with a time step of $\Delta t = 5 \times 10^{-5}$. Our baseline high-fidelity model is solved on a square $N_0 \times N_0$ grid with $N_0 = 1024$ so that our grid spacing in physical space is $\Delta_{DNS} = 2\pi/N_0$. According to standard parlance we refer to these solutions as Direct Numerical Simulations (DNS) because no parameterization is used to represent turbulence below the grid scale other than quadratic viscosity. Our value of viscosity coefficient, that is, 1/Re in our non-dimensional formulation, is chosen such that either decreasing the grid size by holding Re fixed or increasing Re (decreasing viscosity) with the grid size fixed will lead invariably to a rapid pile-up of energy at the smallest wave numbers and eventually to numerical instability. As a comparison, the choice of Re ∼ 1,000 in (Dresdner et al., 2022) allows the authors to obtain stable solutions without any parameterization for coarser grids of up to

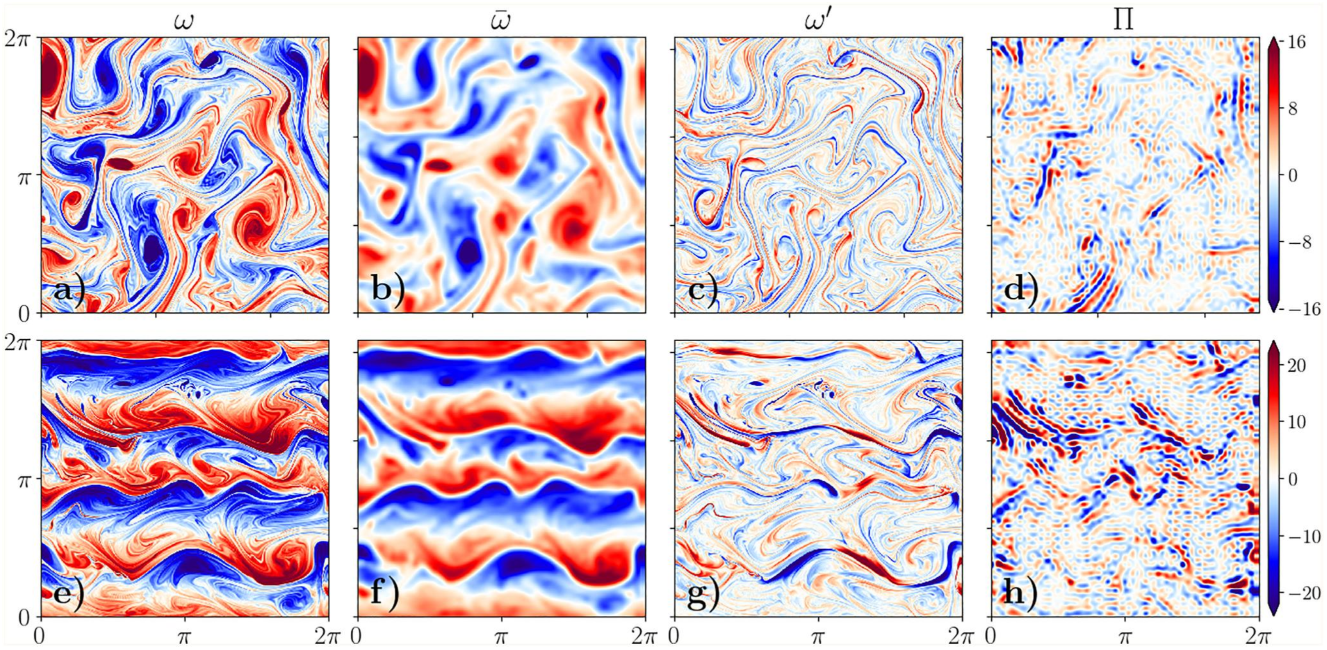

Figure 1. Snapshots after $t = 2 \times 10^6 \Delta t$ for $\beta = 0$ of (a) direct numerical simulations (DNS) vorticity ω, (b) Filtered DNS vorticity $\bar{\omega}$, (c) sub-grid scale (SGS) vorticity field $\omega' = \omega - \bar{\omega}$, (d) SGS stress divergence Π. Same for (e, f) but for $\beta = 20$; note the jets along the x-axis. Note that the first and third columns are on a $1{,}024 \times 1{,}024$ grid while the second and fourth columns are on the coarse 64×64 grid.

128×128; their problem also involves a 16× downscaling from $1{,}024 \times 1{,}024$ to 64×64. Frezat et al. (2022) have a larger value of Re than the one used in this study in their 32× downscaling online closure while Ross et al. (2023), in their study in two-layer QG turbulence (for a downscaling factor of 4× from 256×256 to 64×64) likely have a Re value in similar range as Dresdner et al. (2022).

In the absence of the β-term, the flow is statistically homogenous and isotropic in space and time. Figure 1a) shows a single snapshot of the vorticity field after $t = 2 \times 10^6 \Delta t$; large coherent vortices, typically of length scales at or larger than the forcing scale, are prominent in a sea of fine filamentary structures. The flow kinetic energy ($\mathcal{E} = u^2 + v^2$) is concentrated at the coherent vortices as a consequence of the cascade of energy to larger scales, while both the vortex cores and the filamentary structures are significant reservoirs of the flow enstrophy $\left(\mathcal{Z} = \omega^2\right)$, the latter a consequence of the direct cascade of enstrophy to small scales (Kraichnan, 1971).

For finite values of β, however, the turbulent flow undergoes a symmetry breaking instability in the y-direction resulting in the formation of alternating banded zonal jets, that is, along x-direction. The mechanism of jet formation is not a simple one and is best described as a form of turbulent instability, called zonostrophic instability (Bakas et al., 2015; Farrell & Ioannou, 2007; Marston et al., 2008; Srinivasan & Young, 2012). These jet-like structures are common in planetary atmospheres, the most striking example being the visible banded jet structures on Jupiter. Figure 1e shows a snapshot of the jets formed for this specific choice of $\beta = 20$–the coherent vortices seen in the $\beta = 0$ case are no longer visible separately but now closely interact with the jets themselves, though the fine-scale filaments are clearly seen.

2.2. Downscaling

In this section, we describe our approach to downscale or coarse-grain the high-fidelity DNS solutions described in the previous section to a lower resolution grid. Downscaling is effected in spectral space using a *cutoff* filter, also known as a sharp spectral filter which simply sets modes higher than a cutoff wavenumber to be zero. For a downscaling factor of n_d, the grid spacing of the reduced order model is thus $\Delta_c = n_d \Delta_0$ and the number of grid points decreases to $N_c = N_0/n_d$. We choose $n_d = 16$, thus projecting the DNS solutions from a 1024×1024 grid to a 64×64 grid. In practice, before applying the cutoff filter, following Guan, Subel, et al. (2022), we first apply a Gaussian filter to the fields. The reason for this choice is three-fold: First, Z. Zhou et al. (2019) demonstrated that this procedure produced SGS fluxes which have a higher correlation with the coarse-grained field, making the learning problem easier. Second, this brings the spectral learning problem closer to finite-difference and finite-volume approach common in Earth system models, because localized finite difference stencils can be represented as an effective exponential cutoff filter (Lele, 1992). Finally, only using the cutoff filter creates small scale, spatially nonlocal features in physical space that have little physical structure and are difficult to learn due to the spectral bias of neural networks toward learning lower frequency more easily than higher frequency (Rahaman et al., 2019), especially in offline-learned parameterizations where there is no inherent dynamical awareness; see discussion in Section 1. These reasons explain why using Gaussian and exponential filters became a common practice before cutoff in recent studies involving CNN-based parameterizations of two-dimensional turbulence; see for example, (Dresdner et al., 2022; Frezat et al., 2022; Guan, Chattopadhyay, et al., 2022; Guan, Subel, et al., 2022). Denoting by $\hat{\omega}(k_x,k_y)$ the Fourier transform of the vorticity, $\omega(x, y)$, the coarse-grained field (represented by an overbar) is written as

$$\overline{\hat{\omega}}(k_x,k_y) = \hat{\omega} * G_{\Delta_c}, \quad |k_x|,|k_y| < k_c, \tag{8}$$

where * denotes the convolution operator, $k_c = \pi/\Delta_c = N_c/2$ is the cutoff wavenumber (for $N_c = 64$, $k_c = 32$) and the Gaussian filter G_{Δ_c} is given by.

$$G_{\Delta_c}(k_x,k_y) = \exp\left(-\left(\frac{k_x^2 + k_y^2}{24}\right)\Delta_F^2\right) \tag{9}$$

$$= \exp\left(-\frac{\pi^2}{6}\frac{k^2}{k_c^2}\right), \tag{10}$$

with $\Delta_F = 2\Delta_c$; see (Guan, Chattopadhyay, et al., 2022). The fields filtered from the DNS solutions on the lower-resolution grid are referred to as the Filtered DNS (FDNS) fields. Figures 1b and 1f) display snapshots of the 16× downscaled fields (i.e., the FDNS fields) corresponding to the DNS snapshots in Figure 1a) and e) for the cases of $\beta = 0$ and $\beta = 20$ respectively. Interestingly, Ross et al. (2023) find that the above Gaussian filtering approach makes the learning problem more challenging but we find few issues with either the offline learning of the CNN or its online deployment.

2.3. Sub-Grid Scale Stresses

We apply the filtering step in Equation 8 to the momentum Equation 3 and the corresponding vorticity Equation 6. The momentum equation is then rewritten as

$$\partial_t \bar{\boldsymbol{v}} + \bar{\boldsymbol{v}} \cdot \nabla \bar{\boldsymbol{v}} + f(\boldsymbol{k} \times \bar{\boldsymbol{v}}) = -\mu \bar{\boldsymbol{v}} + \frac{1}{Re}\nabla^2 \bar{\boldsymbol{v}} + \mathcal{F}_{\boldsymbol{v}}(x,y) - \nabla \cdot \bar{\boldsymbol{S}}, \tag{11}$$

where the SGS momentum flux tensor (sometimes just referred to as the stress tensor) is

$$\boldsymbol{S} = \begin{bmatrix} \tau_{uu} & \tau_{uv} \\ \tau_{uv} & \tau_{vv} \end{bmatrix}, \tag{12}$$

where $\tau_{uu} = \overline{u^2} - (\bar{u})^2$, $\tau_{uv} = \overline{uv} - \bar{u}\bar{v}$ and $\tau_{vv} = \overline{v^2} - (\bar{v})^2$ are the three tensor components as $\bar{\boldsymbol{S}}$ is clearly a symmetric tensor. We decompose Equation 12 as

$$\boldsymbol{S} = \underbrace{\begin{bmatrix} \frac{\tau_{uu} - \tau_{vv}}{2} & \tau_{uv} \\ \tau_{uv} & -\frac{\tau_{uu} - \tau_{vv}}{2} \end{bmatrix}}_{S_d} + \underbrace{\frac{\tau_{uu} + \tau_{vv}}{2}}_{S_0}, \tag{13}$$

where is the identity matrix and $\mathcal{E} = \frac{\tau_{uu}+\tau_{vv}}{2}$ is the SGS kinetic energy. The first term on the RHS, $\boldsymbol{S}_d$ is referred to the deviatoric stress tensor and has only two independent components with the third independent component now appearing as the magnitude of the diagonal tensor, $\boldsymbol{S}_0$. Note the forcing terms are not affected by the filtering operation as we always chose our cutoff scale, $k_c > k_f$.

Applying the filter, Equation 8 to the vorticity equation, Equation 6, we arrive at the "filtered" equation corresponding to Equation 11 in the form

$$\partial_t \bar{\omega} + J(\bar{\psi}, \bar{\omega}) + \beta \bar{v} = -\mu \bar{\omega} + \frac{1}{Re} \nabla^2 \bar{\omega} + f(x, y) - \Pi, \tag{14}$$

where

$$\Pi = \overline{J(\psi, \omega)} - J(\bar{\psi}, \bar{\omega}), \tag{15}$$

is the SGS vorticty flux *divergence* that can be written in two other alternative forms. The first form expresses Π in terms of the divergence of the SGS vorticity flux,

$$\Pi = \nabla \cdot (\overline{\boldsymbol{v}\omega} - \bar{\boldsymbol{v}}\bar{\omega}). \tag{16}$$

The second form, more directly relevant to the approach pursued in this manuscript, relates Π to the SGS momentum flux tensor, $\boldsymbol{S}$

$$\Pi = \nabla \times (\nabla \cdot \boldsymbol{S}) = \nabla \times (\nabla \cdot \boldsymbol{S}_d) \tag{17}$$

where the curl operation above is the two dimensional curl. The second equality above results because $\boldsymbol{S}_0$ is a diagonal tensor, and it follows that $\nabla \times (\nabla \cdot \boldsymbol{S}_0) = 0$. Figures 1d and 1h) show snapshots of Π for the cases $\beta = 0$ and $\beta = 20$ respectively; the corresponding SGS vorticity fields, $\omega' = \omega - \bar{\omega}$ are also shown in the same figure for reference. There is a great deal of correlation between ω' and Π but this is, evidently, not a simple dependence.

Having a data-driven model of Π as a function of the coarse fields allows us, in principle, to solve Equation 14 though the question arises as to which form of Π above should be used for modeling. The most common form used is the one in Equation 15 where Π is directly modeled using a NN (Frezat et al., 2022; Guan, Chattopadhyay, et al., 2022; Guan, Subel, et al., 2022; Maulik et al., 2019). An alternative approach might be to model the two components of the SGS vorticity flux vector $\overline{\boldsymbol{v}\omega} - \bar{\boldsymbol{v}}\bar{\omega}$ and apply the divergence operator to compute Π in Equation 16. The approach adopted here is inspired by (Zanna & Bolton, 2020), namely to model the components of the SGS momentum flux, $\boldsymbol{S}$ and compute Π using Equation 17. Zanna and Bolton (2020) used a single convolutional NN to model the three components of $\boldsymbol{S}$, namely τ_{uu}, τ_{vv} and τ_{uv}. We propose in this study a variation of this approach by learning a CNN approximation of the deviatoric stress tensor's two components, $(\tau_{uu} - \tau_{vv})/2$ and τ_{uv}, which, as pointed above, is sufficient to obtain Π; see Equation 13.

2.4. The Locality of Resolved and SGS Scale Interactions

The interactions between the resolved fields and the SGS motions are only weakly non-local (Eyink, 2005; Eyink & Aluie, 2009). This property can be for example, inferred by examining the two-dimensional spatial cross-correlations between $\bar{\omega}$ and Π. For reasons of efficiency the spatial cross-correlation is computed for each snapshot by using fast Fourier transforms through the cross-correlation theorem, a generalization of the Wiener-Khinchin theorem (Fisher, 2008), and then averaged across time. In Figure 2, we highlight the x- and y-sections of the cross-correlation. When $\beta = 0$, the cross-correlation is essentially isotropic and examining any of these sections suffices; it can be observed in Figure 2 that the correlation is weak beyond seven grid points in space. When $\beta = 20$ the along-jet correlation mirrors the $\beta = 0$ result but the cross-jet correlation length is larger; this relates to interactions and coupling between the jets composing the flows. Even in this case, however, highly correlated regions are still relatively local. This observation motivates the choice of our NN architecture employed in this manuscript, as detailed subsequently.

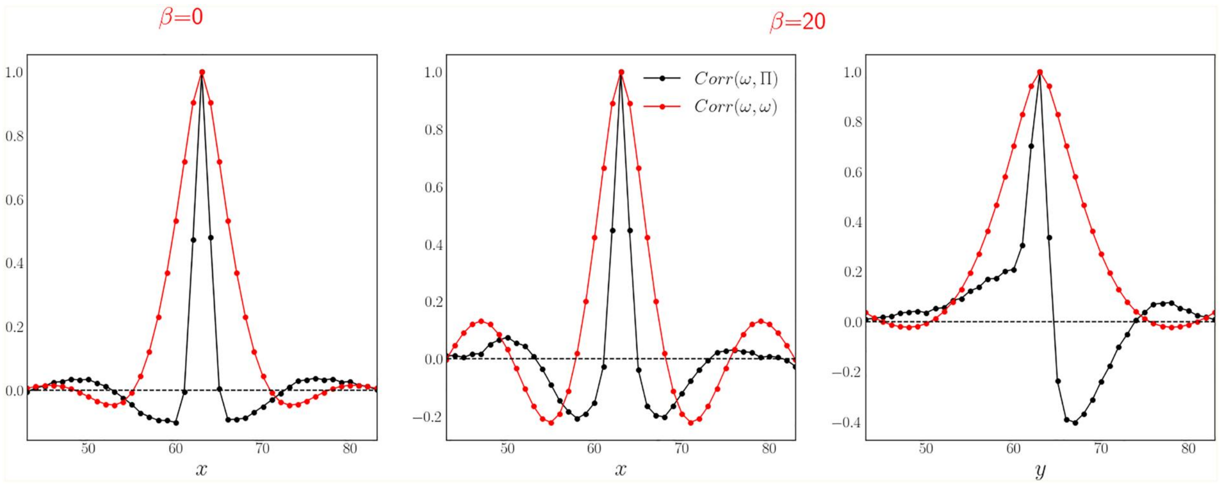

Figure 2. x- and y-sections of the spatial cross-correlations between Π and the resolved vorticity field, $\bar{\omega}$. For $\beta = 0$, the correlations along both directions are nearly symmetric so only one of them is shown. For $\beta = 20$, the cross-jet correlation length (along y) is longer than the along-jet correlation length that is comparable to that in the $\beta = 0$ case.

As a side remark, we emphasize that autocorrelation of the resolved vorticity highlights the average size of the eddies in the $\beta = 0$ case and the jet size when $\beta = 20$ (red curves in Figure 2).

2.5. Galilean Invariant SGS Models

The equations of motion, Equations 3 and 6 are Galilean invariant, that is, invariant to changes in the inertial frame of reference. To demonstrate this, and without loss of generality, we can re-write the equations in a reference frame translating with constant velocity (denoted by primes), $\boldsymbol{V} \equiv (U, V)$ along x and y directions, that is,

$$x' = x - Ut, \quad y' = y - Vt, \quad t' = t. \tag{18}$$

and

$$u' = u + U, \quad v' = v + V, \quad \psi' = \psi + Vx - Uy, \quad \omega' = \omega. \tag{19}$$

Note that the vorticity is invariant to a uniform translation while the other quantities are not; this is because the vorticity is comprised of gradients of velocity; in general all four possible gradients of velocity are invariant to Galilean transformations

$$v'_x = v_x, \quad v'_y = v_y, \quad u'_x = u_x, \quad u'_y = u_y, \tag{20}$$

as are linear combinations of these velocity gradients, including ω and the two strain components, the normal and shear strains defined respectively as,

$$\sigma_n = u_x - v_y, \tag{21}$$

and

$$\sigma_s = v_x + u_y. \tag{22}$$

An important consequence of Galilean invariance is that the filtered Equations 11 and 14 must also be Galiean invariant (the filtering operator being independent of time) and consequently so must be the SGS fluxes, $\boldsymbol{S}$ and Π.

Turbulence modeling or closure refers to the modeling of SGS fluxes, either Π directly or modeling $\boldsymbol{S}$ and then using Equation 17 to obtain Π, as functions of the coarse field variables, although different choices for the input variables are possible as mentioned above. A natural choice in solving the filtered vorticity Equation 14 is to choose the input coarse field variables to be $(\bar{\psi},\bar{\omega})$. This choice made in (Frezat et al., 2022; Guan, Chattopadhyay, et al., 2022; Guan, Subel, et al., 2022; Maulik et al., 2019) consists of Π modeled directly as $\Pi = \Pi_\theta(\bar{\psi},\bar{\omega})$, in which θ, based on common parlance, represents the set of parameters comprising the model which can range from a simple linear regression model to more complex choices like neural networks. Alternatively, one may choose the coarse-grained velocities as inputs (Dresdner et al., 2022), that is, $\Pi = \Pi_\theta(\bar{u},\bar{v})$.

Zanna and Bolton (2020) use $(\bar{u},\bar{v})$ as inputs and model the three components of $\boldsymbol{S} = \boldsymbol{S}_\theta(\bar{u},\bar{v})$ to obtain Π through Equation 17. However, it should be noted that none of these choices are Galilean invariant, because on changing the intertial frame of reference leads us to changes in $(\bar{\psi},\bar{u},\bar{v})$ given by Equation 19; thus a priori, models that are functions of the above variables will not be Galilean invariant though given sufficient data, the NN with sufficiently high number of parameters could plausibly learn this physical symmetry. However, it is possible to make a simple but sufficient modeling choice that implicitly assumes Galilean invariance. For the two components of deviatoric stress tensor, $\boldsymbol{S}_d$, we aim at finding the following parameterization

$$\boldsymbol{S}_d = \boldsymbol{S}_d^\theta(\bar{\omega},\bar{\sigma}_n,\bar{\sigma}_s), \tag{23}$$

where θ represent the set of all parameters of the CNN. Because $(\bar{\omega},\bar{\sigma}_n,\bar{\sigma}_s)$ represent all possible linear combinations of gradients of the velocity, adding a uniform velocity leaves them trivially invariant (Note that the fourth linear combination, the divergence, $\bar{\delta} = \bar{u}_x + \bar{v}_y = 0$ due to the continuity relation). We also learn below non-Galilean invariant models, following Dresdner et al. (2022); Zanna and Bolton (2020); Ross et al. (2023) as

$$\boldsymbol{S}_d = \boldsymbol{S}_d^\theta(\bar{u},\bar{v}). \tag{24}$$

We choose not to learn neural models that use as inputs $(\bar{\psi},\bar{\omega})$ (as e.g. in Frezat et al., 2022; Guan, Chattopadhyay, et al., 2022) for the simple reason that the streamfunction is generally not available in realistic geophysical models, making such choices not easy to generalize in practice. It is indeed important to note that on the spectral plane the differential operators involved are often diagonal, but in general a solution of an expensive Poisson equation is required to obtain ψ.

2.6. Baseline Parameterizations

Following recent studies (Frezat et al., 2022; Guan, Subel, et al., 2022; Ross et al., 2023), we choose the parameterizations by Smagorinsky and Leith as baselines. These models see widespread use in both contemporary LES studies and realistic numerical models currently being used in the climate system (Bachman et al., 2017; Pearson et al., 2017). The Smagorinsky parameterization is generally implemented in the momentum equations, Equation 11, and parameterizes $\boldsymbol{S}_d$ in terms of the coarse strain tensor. However, we choose the version of Smagorinsky that is directly implemented in terms of the coarse vorticity field, $\bar{\omega}$ (Maulik et al., 2019) and in divergence form (Frezat et al., 2022); note that the momentum and vorticity forms of Smagorinsky are not equivalent. Both the above parameterizations are *diffusive* parameterizations and assume that Π can be represented in diffusive form as

$$\Pi = -\nabla\cdot(\nu_e\nabla\bar{\omega}), \tag{25}$$

where the only unknown is the "eddy" diffusivity, ν_e. The Smagorinsky parameterization models ν_e as

$$\nu_e = c_s\Delta_c^2|\bar{S}|, \tag{26}$$

where $|\bar{S}|^2 = \sigma_n^2 + \sigma_S^2$ is the strain magnitude and Δ_c is the grid size of the coarse grid. The Leith paramaterization takes the form

$$\nu_e = c_l\Delta_c^3|\nabla\bar{\omega}|. \tag{27}$$

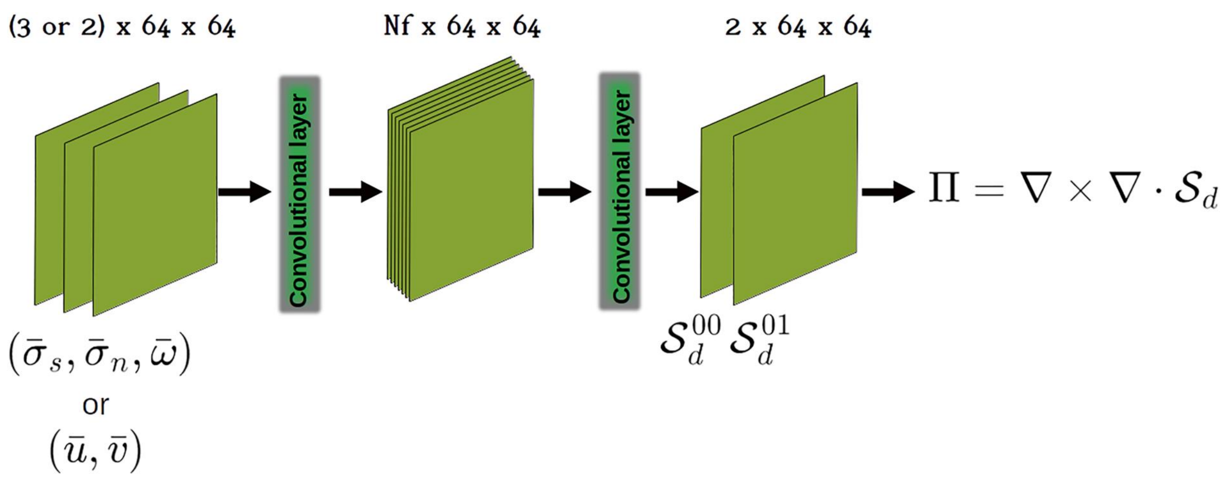
(3 or 2) x 64 x 64
Nf x 64 x 64
2 x 64 x 64
Convolutional layer
Convolutional layer
$\Pi = \nabla \times \nabla \cdot \mathcal{S}_d$
$(\bar{\sigma}_s, \bar{\sigma}_n, \bar{\omega})$
or
$(\bar{u}, \bar{v})$
$\mathcal{S}_d^{00}\ \mathcal{S}_d^{01}$

Figure 3. Pipeline of Convolutional Neural Networks used to model sub-grid scale (SGS) stresses. The input fields are either of the coarse field combinations $(\bar{\omega}, \bar{\sigma}_n, \sigma_s)$ or $(\bar{u}, \bar{v})$ while the output filters are the two relevant components of the deviatoric stress tensor, $\mathcal{S}_d^{00}$ and $\mathcal{S}_d^{01}$ from while the SGS vorticity fluxes are computed first through a tensor divergence followed by a two-dimensional curl operation.

Note that both of the above parameterizations are Galilean invariant as explained in the preceding section. Following (Ross et al., 2023), we vary the constants c_l and c_s in [0.01,1.0] and run coarse-grid solutions with Smagorinsky and Leith parameterizations to similarly long times as the CNN-LES to allow for a direct comparison. As a foreshadowing of the results ahead, we note that in the vorticity form expressed above, the flows obtained from the Leith and Smagorinsky models are extremely similar when $c_s = c_l$.

3. Machine Learning Framework

3.1. CNN Architecture and Non-Locality

We employ an entirely offline machine learning pipeline to learn the SGS fluxes in this manuscript. As described previously, to solve the filtered vorticity equation, Equation 14, we model $\boldsymbol{S}_d$ using a Convolutional NN that takes in the filtered field variables as inputs, either $(\bar{u}, \bar{v})$ or $(\bar{\omega}, \bar{\sigma}_n, \bar{\sigma}_s)$; Π is then obtained from $\boldsymbol{S}_d$ using Equation 17. Convolutional layers are translationally-equivariant and a natural choice because the Navier-Stokes equations also have the same symmetry. Recent studies (Dresdner et al., 2022; Frezat et al., 2022; Guan, Chattopadhyay, et al., 2022; Guan, Subel, et al., 2022; Ross et al., 2023) employ deep CNNs for parameterization consisting of a few hundred thousand to a million parameters. We find, however, that *shallow* CNNs, in particular, simple two-layer neural networks with only $\mathcal{O}(1000)$ parameters are sufficient to model SGS stresses. Our precise architecture is shown in Figure 3. The input layer consists of two or three inputs depending on whether the velocities or a combination of vorticity and strains are chosen as inputs; the sole hidden layer consists of N_f convolutional filters with the two output layers only having the two independent fields that constitute $\boldsymbol{S}_d$, namely $\left(S_d^{00}, S_d^{01}\right) = \left(\frac{\tau_{uu} - \tau_{vv}}{2}, \tau_{uv}\right)$; see Equation 13. The choice of the filter width, commonly referred to as the kernel size of each convolutional filter in each layer is set to $ks = 5$. N_f is a key hyperparameter, which we choose to be either 8, 16 or 32. The total number of parameters for each of these choices is 800, 1,600, and 3,200 for the case of $(\bar{u}, \bar{v})$ input respectively. In either case, the number of convolutional filters is unusually low compared to the norm; our objective being to find the smallest possible NNs which lead to accurate and stable solutions and we show that these actually suffice. We employ a single *swish activation function*, $\mathcal{S}$, after the first layer; that is, $\mathcal{S}(\xi) = \xi H(\xi)$ where $H(\xi)$ is the sigmoid function. Swish tends to work better than ReLU across a number of challenging data sets (Ramachandran et al., 2017).

Two important points underline our choice of shallow CNN, the first of which is the *effective non-locality*. This quantity is also known as the receptive field (RF) of a CNN in the computer vision literature. It is defined as the set of points in the input domain which is connected to the center of the point region that the CNN is currently operating on. Alternatively this is the effective filter width of the CNN. For example, a single convolutional layer with a 5×5 filter has a trivial $5 \times 5 = 25$ point RF, that is, an effective filter width of 5. Because NNs are compositions of layers comprising convolutional filters, the RF is a linear function of the number of layers. Thus a two-layer CNN with 5×5 filters has an RF of 9×9 points, that is, an effective filter width (Note that for a general n-layer CNN comprised of $ks = 5$ filters, its effective width is $4n + 1$) of 9. As a comparison, the 11-layer CNN employed by Guan, Subel, et al. (2022) has a RF of 45×45, a related 10-layer CNN by Frezat et al. (2022) has an RF of 41×41, which on a 64×64 domain is nearly global in its non-locality. The study by Maulik et al. (2019) used spatially local NNs, in the form of a fully connected NN instead of CNN, with a RF only equal to 3×3; too low for the current problem. Ross et al. (2023) employed an 8-layer CNN with a RF of 21×21; they examined the gradient of the output SGS fluxes with respect to the inputs at a single point for their $\beta = 0$ case and found non-zero values in only a 9×9 spatial region, consistent with the RF of our 2-layer CNN chosen in this study. However, we also find the non-trivial result that the same 2-layer CNN architecture also suffices in accurately modeling the $\beta = 20$ jets case as demonstrated in subsequent sections.

The RF is closely connected to the *stencil size* in numerical discretization and can be directly identified as such in the context of solving PDEs. While our choice of CNNs assumes implicitly non-locality as in other studies of SGS closure, in contradistinction, our closure results with 2-layer CNNs presented hereafter allow us to claim that the inherent degree of non-locality in SGS parameterization is weak, even at high Reynolds number. This observation has implications for deployment of SGS-parameterizations in ocean and atmospheric models which often use a spatial domain decomposition for solution on large compute clusters. Having a CNN with large spatial non-locality would pose severe limitations on the domain decomposition, which are alleviated by choosing shallow CNNs.

The second point of note is the effective non-linearity. Due to their hierarchical structure, deeper NNs are able to represent a more complex family of nonlinear functions than shallow NNs having the same number of parameters (Bengio & Delalleau, 2011). In computer vision studies (Wang et al., 2023), it has been demonstrated that lower layers (i.e., closer to the inputs) in the network learn simple features, while higher layers build upon these to represent more complex patterns and abstractions. Therefore our choice of 2-layer CNN implicitly makes certain assumptions toward simplicity and sparsity of the relationships between the SGS fluxes and the input coarse flow variables. It is possible to increase the degree of nonlinearity of the CNN without increasing its RF or non-locality through 1×1 "convolution" layers which are strictly speaking not convolutions but fully connected layers across the feature dimension. However, we do not find a need for such choices.

3.2. Data Generation and Loss Function

Since the output of our CNN is the deviatoric stress tensor, $\boldsymbol{S}_d$, we use Equation 17 to compute the SGS vorticity flux divergence used in Equation 14. Our loss function is a simple L_2-loss between the CNN-parameterization of Π and that obtained from FDNS, that is regularized as follows

$$\mathsf{L} = \|\nabla \times (\nabla \cdot \boldsymbol{S}_d^{\theta}) - \Pi_{\mathrm{FDNS}}\|_2^2 + wd \sum \theta_j^2. \tag{28}$$

Here, the norm $\| \cdot \|_2$ refers to the L_2-norm over the flow domain while the L_2-penalty coefficient on the sum of squares of all the weights of the NNs is referred to as the "weight decay" coefficient (wd). Compared to recent studies (Guan, Subel, et al., 2022; List et al., 2022), the use of additional physics-informed losses to further constrain our offline training, did not turn out to be an important ingredient to derive accurate closures within our shallow CNN framework. As shown in subsequent sections, we find high fidelity solutions without recourse to adding more physical constraints as part of the loss function. These constraints might, however, help in further reducing our training data size though this question has not been explored here and is left for future studies.

To train our 2-layer CNNs, we generate four separate trajectories of high resolution solutions with differing random initial conditions. Following the protocol of Guan, Subel, et al. (2022), we use weakly correlated snapshots that are 1,000 Δt iterations apart for training and testing, to promote diversity within the training data set. We use 200 snapshots each from the first two trajectories as our training set with a total of 400 snapshots corresponding to 400,000 high-resolution model iterations, and 400 snapshots of the third trajectory for the test set and the fourth trajectory (the validation trajectory) to initialize our CNN-LES run and compare with the corresponding FDNS evolution. Since we wish to characterize the fidelity of our parameterized solutions over long time horizons, we run the fourth trajectory for $T = 6 \times 10^6 \Delta t$, which is also the time for which we run our online CNN-LES solutions for validation.

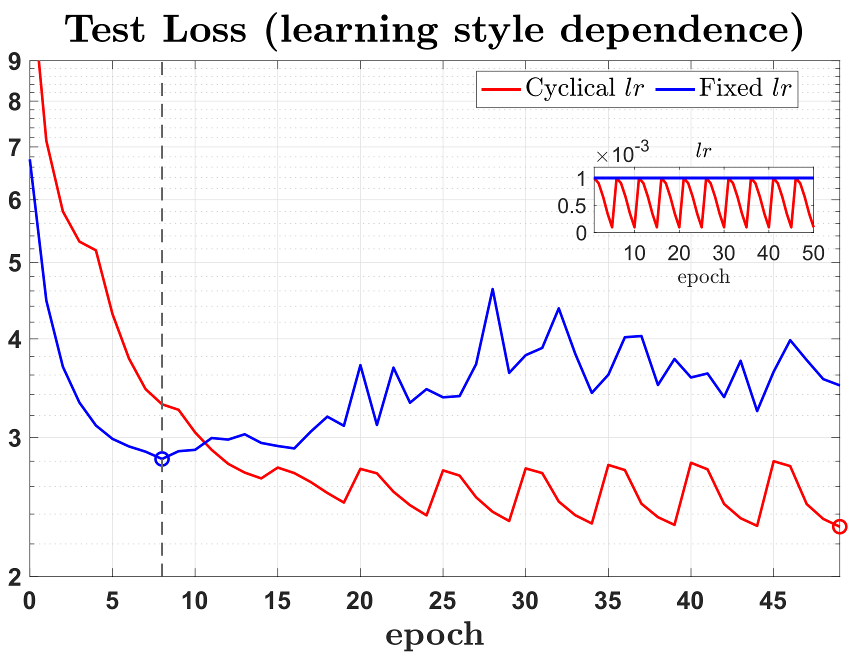

Figure 4. Test loss for the cyclical learning case (red curve) versus test loss for the fixed learning rate case (blue curve). The inset shows the learning rate used in the cosine annealing case (red) versus the fixed learning rate *lr* case (blue), as a function of epochs. The empty circles show, for each case, the learned model that has the lowest test error over the entire course of training. The Convolutional Neural Networks model is saved for this point and used to drive the online CNN-LES runs.

A minor if important detail here is that we do not normalize our input or output data before applying the CNN on it, nor do we use any normalization layers like Batch Normalization (Ioffe & Szegedy, 2015). This is likely because of our choice of shallow CNNs and the fact that our inputs have numerical values that are in the O(10) range which does not hamper the learning process.

3.3. Learning Rate Annealing and Model Selection

The learning rate (*lr*) is crucial hyperparameter in the optimization of neural networks. While the simplest training approach is for the learning rate to be kept constant through the entire training process, specific forms of *lr*-annealing can substantially improve test accuracy. We choose cosine annealing with warm restarts (Loshchilov & Hutter, 2016) because of its tendency to provide a strong implicit regularization and robust solutions across a wide range of hyperparameters.

Figure 4a shows the cyclically annealed *lr* as a function of epoch. Within this optimization framework, *lr* is initiated at a maximum value ($lr = 0.001$ in Figure 4) and is decreased to zero via the cosine function over the course of 5 epochs before suddenly ramping back to its original value. Typical test losses as a function of epoch are shown in Figure 4. For the annealed case (red curve), after the initial decrease, the test loss also rises but reaches a new local minimum after each cycle, with smaller loss function values than previously reached.

For the fixed-*lr* case, however, the test loss decreases to a minimum value at which the CNN model is chosen as the optimal one but then becomes erratic as the number of epochs increases. We also show the lower-bound envelope of the test loss for each case (in blue); note that this curve is monotonically decreasing over a larger number of epochs in the cyclically annealed case. In each case, we select the CNN model with the least test loss over the entire duration of training which corresponds to the right-most point on the blue curve as marked by the empty circles in Figure 4. Because of our specific model selection criteria, we do not need to monitor our training process, a consequence of which is that our test set needs to be diverse enough to reflect the dynamics' variability as well as sufficiently large.

3.4. Hyperparameter Grid Search

The other two central hyperparameters in the model optimization process are the batch size and weight decay. Typical batch sizes are chosen to be larger than some threshold, but we choose the batch size to be *unity* for all our model training which normally would lead to extremely slow and noisy training. The main reason here is that, as explained in Section 3.1 above, the 2-layer CNN chosen here has an effective non-locality of 9×9 which is the region in the coarse-grid domain where the CNN acts on independently of the other parts of the domain. Since our coarse-grid domain has size, 64×64, this means that our *effective* batch size is actually $64^2/9^2 \approx 50$. For sufficiently deep CNNs the effective batch size and actual batch size would be identical.

To find the best CNN parameterizations for our closure, we perform an extensive hyperparameter search in (lr, wd) search space. For each value of the hidden layer size, N_f in $\{8, 16, 32\}$ we vary our learning rate lr in $\{10^{-4}, 10^{-3}, 10^{-2}\}$ and our weight decay coefficient, wd, in $\{10^{-5}, 10^{-4}, 10^{-3}\}$ for $\beta = 0$ and wd in $\{10^{-3}, 10^{-3}, 10^{-1}\}$ for $\beta = 20$; it was found that the $\beta = 20$ cases needed substantially higher wd rates, that is, stronger regularization. The precise reason for this is still being investigated. For each of these parameter choices we train models for both fixed lr and cyclically annealed lr where the chosen value of lr for the annealed case represents the maximum value of lr. For each of the above choices, we consider two classes of inputs, $(\bar{u}, \bar{v})$ and $(\bar{\omega}, \bar{\sigma}_n, \bar{\sigma}_s)$ as described in Section 2.5. Thus we train and test a total of $3^3 \times 2 \times 2 = 108$ CNN models (2 types of optimizations—annealing or not; 2 types of inputs) each for the $\beta = 0$ and for $\beta = 20$ cases. Note that because our CNNs have such a small number of parameters, both our training times and CNN-LES runtimes are extremely fast.

Training each CNN model to 50 epochs takes only about 1.5 min on a V100 GPU while running the CNN-LES model for $T = 6 \times 10^6 \Delta t$ takes about 20 min. Due to the 16× downscaling adopted here, the time-step of our downscaled run is 16 times larger so this is equivalent to around 375,000 Δt_{LES}. We note that 84 of the 108 cases when $\beta = 0$ and 87 of the 108 cases when $\beta = 20$ are numerically stable through the course of the entire online run. We also observed two other solutions which became numerically unstable after $t = 5 \times 10^6 \Delta t$ illustrating the challenges with evaluating online stability.

3.5. Diagnostics

We evaluate the accuracy and fidelity of our online CNN-LES runs using a variety of metrics which quantify the structure and dynamics of the CNN-parameterized downscaled equations, relative to the ground truth FDNS solution. The kinetic energy spectrum, $\hat{E}(k)$, where the $\hat{\cdot}$ symbol represents that the quantity is in spectral space and $k^2 = k_x^2 + k_y^2$ is the radial wavenumber, is a fundamental metric for quantifying turbulent flows and dynamical regimes spanning spatial scales.

We can also write more dynamically relevant cross-scale kinetic energy and enstrophy fluxes associated with the SGS flux, defined in spectral space as.

$$E_{flux}(k) = \Re\left(-\Pi(k)^* \hat{\bar{\psi}}\right), \tag{29}$$

$$Z_{flux}(k) = \Re\left(\Pi(k)^* \hat{\bar{\omega}}\right), \tag{30}$$

where $\Re$ denotes the real part, $\Pi(k)$ the SGS vorticity flux divergence in spectral space and z^* represents the complex conjugate of z; positive values of these fluxes imply a forward transfer (i.e., from large to small scales) and negative values an inverse transfer (Note that to simplify notations, we have dropped the $\hat{\cdot}$ symbol over the energy, enstrophy and Π symbols). Note that the above fluxes do not represent the total flux of kinetic energy and enstrophy in the CNN-LES solutions but only the contributions associated with Π. $\Pi(k)$ is itself a metric which can be compared between the FDNS and corresponding CNN-LES runs as a test of the fidelity.

We also construct the probability distribution functions of the coarse vorticity field, $P(\bar{\omega})$ and the SGS flux, $P(\Pi)$. For turbulent flows, these quantities are not Gaussian and can have strong tails, a problem exacerbated by the intermittency of two-dimensional and geostrophic turbulence caused due to the persistence of long lived vortices. The intermittency problem ensures that without extremely long time simulations, the tail of the distributions are difficult to capture accurately and remain noisy. In general the standard approach for computing these quantities for turbulent flows is through Kernel Density Estimation (KDE) (Guan, Subel, et al., 2022; Ross et al., 2023), which depends on the choice of the underlying kernel and method used to obtain the best fit distribution (Botev et al., 2010). However, by choosing to run our online validation solutions over long time horizons of size $T = 6 \times 10^6\ \Delta t$, we obtain well defined probability density function (PDF) estimates by simply estimating histograms without recourse to KDE. We compute our spectra, fluxes and PDFs for the online CNN-LES solutions and the corresponding FDNS solution using 3,000 snapshots separated by 2,000 Δt.

For each of these spectral, fluxes and PDF quantities, we define metrics aimed at evaluating similarities between the FDNS and CNN-LES solutions. The choice of these metrics are related to those used by Ross et al. (2023). For the energy fluxes, we simply use the coefficient of restitution between the FDNS (superscript ^{D}) and online CNN-LES (superscript $^\theta$) quantities to measure the disagreement as.

$$\texttt{energy}-\texttt{flux}-\texttt{diff} = 1 - R^2\left(E^D_{flux}(k), E^\theta_{flux}(k)\right), \tag{31}$$

$$\texttt{enstrophy}-\texttt{flux}-\texttt{diff} = 1 - R^2\left(Z^D_{flux}(k), E^\theta_{flux}(k)\right), \tag{32}$$

where R^2 is the coeffecient of restitution. The energy and SGS divergence spectra of the online solutions are evaluated in similar fashion, though we replace the quantities themselves with their logarithm to ensure that errors at higher wavenumbers are adequately represented. Thus,

$$\texttt{spectral}-\texttt{diff} = 1 - R^2\left(\log\left[E^D(k)\right], \log\left[E^\theta(k)\right]\right), \tag{33}$$

$$\texttt{spectral}-\texttt{sgs}-\texttt{diff} = 1 - R^2\left(\log\left[\Pi^D(k)\right], \log\left[\Pi^\theta(k)\right]\right), \tag{34}$$

Metrics that compare probability distributions are often referred to as divergences and a variety of options exist. We choose an L_2-divergence which is simply the integrated square difference between the PDFs of the FDNS and online CNN-LES distributions,

$$\texttt{distrib}-\texttt{diff} = \int_{-\infty}^{\infty} \left[P\left(\bar{\omega}^D\right) - P\left(\bar{\omega}^\theta\right)\right]^2 \mathrm{d}\,\omega, \tag{35}$$

$$\texttt{distrib}-\texttt{sgs}-\texttt{diff} = \int_{-\infty}^{\infty} \left[P\left(\Pi^D\right) - P\left(\Pi^\theta\right)\right]^2 \mathrm{d}\,\Pi. \tag{36}$$

4. Results

4.1. Offline Accuracy

We evaluate how the CNN models perform in predicting the SGS flux divergence, Π, on the test set given the inputs, either $(\bar{u}, \bar{v})$ or $(\bar{\omega}, \bar{\sigma}_s, \bar{\sigma}_n)$. Ultimately our objective is to construct computationally efficient neural parameterizations which lead to accurate and stable online solutions. However, it is helpful to examine offline test set performance as a precursor to evaluating online performance and to potentially foreshadow a relationship between the two. Furthermore, the presence of a ground truth, absent in online solutions where the CNN-LES and trajectories diverge due to dynamical chaos, allows an examination of the nature of the modeling errors.

Given our large hyperparameter sweep in the (lr, wd)-space, we choose to show offline model comparisons for hyperparameter choices that lead to the best performance on online metrics defined in Section 3.5, as detailed further in Section 4. The results are shown in Figure 5 where we compare individual snapshots of Π^D from the test set and the CNN prediction error $(\Pi^D - \Pi^\theta)$ for different inputs, depending on whether or not learning rate annealing was used. The average test set R^2, as indicated on the corresponding snapshots in Figure 5, ranges from 0.73 to 0.8 but the prediction error shows surprisingly similar small-scale structure across different models and learning rate modalities for both the $\beta = 0$ and $\beta = 20$. This close similarity of the different models can be observed both in the power spectra and probability distribution functions, relative to that of the FDNS test data (bottom row of Figure 5). From the spectra comparisons, we note that the disagreement between the model predictions and data are limited primarily to the three largest wavenumbers, at the end of the spectrum. We found that these high-wavenumber signals can not be modeled even with deep 10-layer NNs and our choice of small

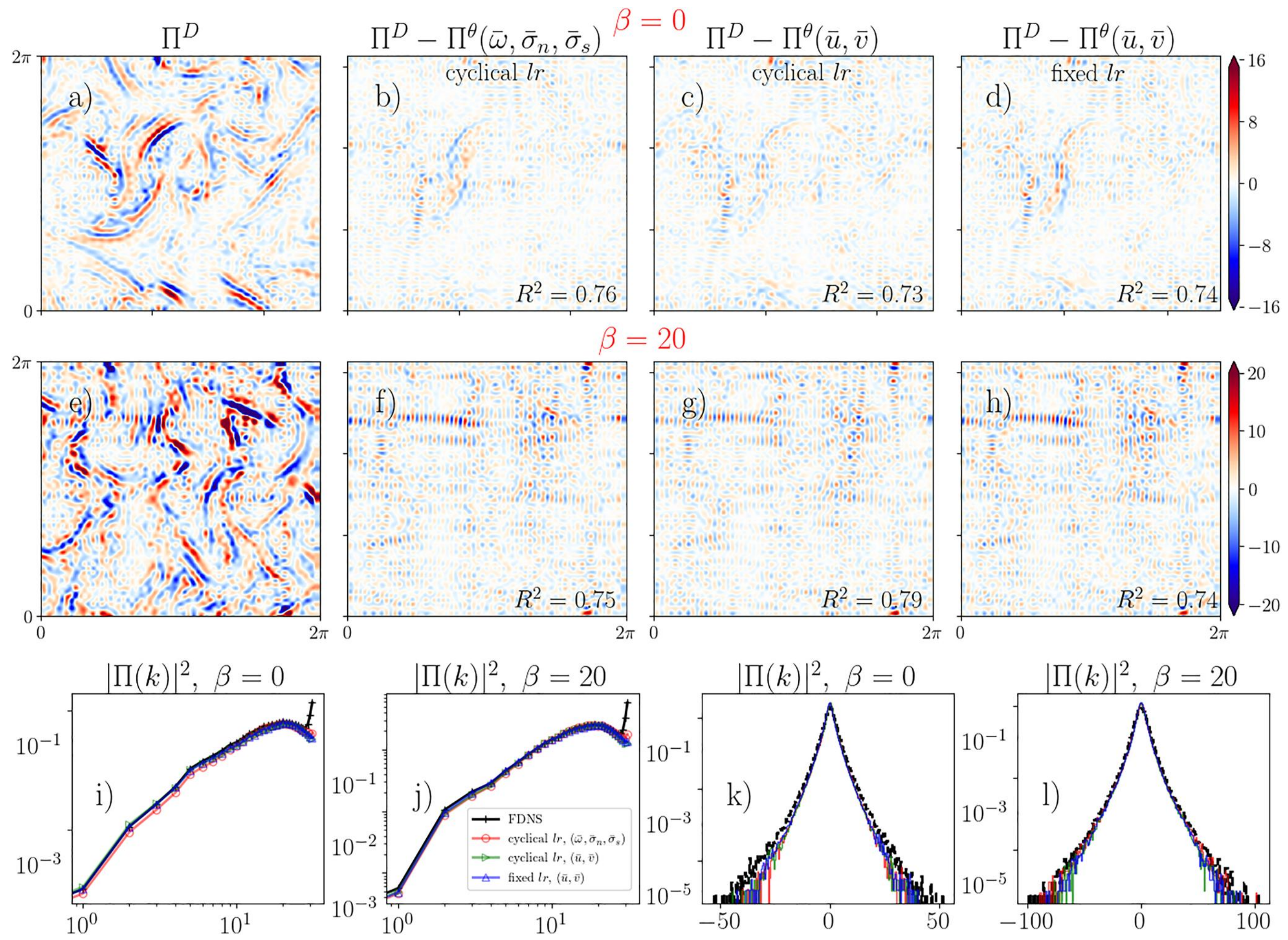

Figure 5. A comparison of the test data sub-grid scale flux divergence, Π^D with that predicted by the trained Convolutional Neural Networks (CNN) models, Π^θ having different inputs and training methodologies. *Top row:* Comparison of a single test set snapshot, Π^D with the error of the CNN predictions $\Pi^D - \Pi^\theta$ depending on whether the inputs are either $(\bar{u}, \bar{v})$ or $(\bar{\omega}, \bar{\sigma}_s, \bar{\sigma}_n)$ or whether cyclical learning rate annealing was used for training, for $\beta = 0$. *Middle row:* Same as the top row but for $\beta = 20$. Each snapshot also shows the coefficient of restitution, $R^2(\Pi^D, \Pi^\theta)$ computed over the entire test set data set (consisting of 400 snapshots). *Bottom row:* Comparison of the power spectra, $\Pi^D(k)$ (black dashed lines) with $\Pi^\theta(k)$, and the corresponding probability distributions, $P(\Pi^D)$ and $P(\Pi^\theta)$ for each of the three cases across different values of β, shown in the two row. Note that the colored curves can be difficult to discriminate because of their close overlap in a consistent way with spatial error plots such as shown in panels (b–d) and (f–h). The markers have been removed from (k, l) to reduce clutter.

parameter-CNNs are not the reason. The presence of these high-wavenumber components is a consequence of a leakage of the spectral cutoff operator that is not removed by the Gaussian filter.

4.2. Online Prediction

4.2.1. Forecast Accuracy

We start by examining the short-term forecast accuracy of our CNN-LES models. In order to achieve this, we run the vorticity closure Equation 14, initialized with a snapshot from the FDNS validation data set and solve forward in time with either the parameterized CNN model learned from data, or the Smagorinsky/Leith parameterizations (we refer to all these runs generically as LES runs). We then compare the short term evolution of the precomputed FDNS solution with the corresponding online CNN-LES solution and Smagorinsky/Leith runs in turn. Adopting this protocol, the longer our LES runs remain correlated in time with the ground truth FDNS, the better we consider the forecast accuracy of the LES to be. We use standard Pearson's correlation computed at each time between the FDNS and the LES $\bar{\omega}$ snapshots, and visualize these as a function of time represented as multiples of Δt, recalling that $\Delta t_{LES} = 16\Delta t$.

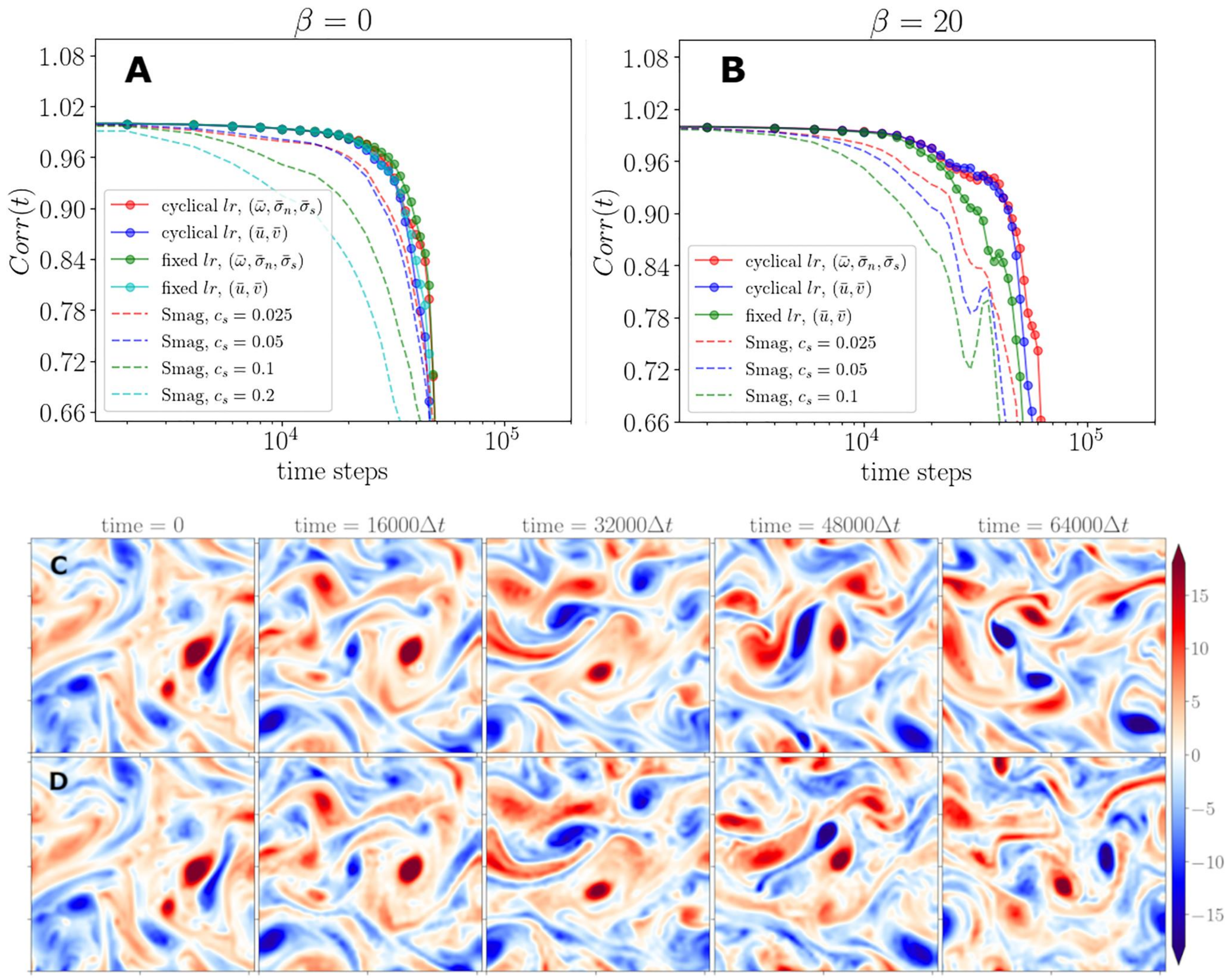

Figure 6. Panels (a, b) Temporal evolution of the correlation coefficient between the FDNS and multiple configurations of the CNN-LES models (each initialized with the same FDNS snapshot) as a function of time-steps of the direct numerical simulations model (solid lines with markers); to get the corresponding number of time steps for the CNN-LES, divide by 16. Also shown are the correlations of the Smagorinsky-parameterized runs with the FDNS for different Smagorinsky constants as indicated. *Row* (d) Example of CNN-LES skills in producing successive snapshots respecting the patterns and coherent structures displayed by the original FDNS simulation for the case $\beta = 0$ shown in row (c) The CNN-LES cases shown here correspond to the best long-term online error based on a large hyperparameter search for the choice of inputs or learning rate choice; see Sections 4.2.2 and 4.2.3 below.

We then define a single metric for forecast accuracy, the decorrelation time, as the time after which the correlation between FDNS and LES drops below 0.96 (Dresdner et al., 2022); thus a longer decorrelation time implies a better forecast accuracy. Given that we are dealing with chaotic turbulent flows, one can also define an *intrinsic* decorrelation time of the FDNS itself, evaluated by perturbing the initial DNS snapshot with noise and then by measuring how fast the flow decorrelates from the unperturbed DNS. Ross et al. (2023) found that Smagorinsky models had close to the best forecast accuracy among all their LES models in spite of poorer long-term online performance. Therefore, we believe it suffices to compare the relative forecast performance of our CNN-LES runs with Smagorinsky/Leith LES runs serving as a strong baseline, without concerning ourself with the intrinsic decorrelation of the DNS itself.

Figure 6 shows the resulting forecast results. First, we note that all four types of CNN-LES solutions (solid curves with solid markers) corresponding to cases shown in the offline section (Section 4.1) have remarkably similar correlation curves. Furthermore, the CNN-LES solutions outperform the Smagorinsky cases (dashed lines) substantially taking almost twice as long to decorrelate from the FDNS runs (note that the time axis is logarithmic). A decrease of the Smagorinsky constant c_s, results into an improvement in the forecast performance. However, as discussed in subsequent sections, decreasing the value of c_s below 0.025 (purple curve in Figure 6) causes a small-wavenumber pile up of energy leading eventually to a deterioration of the long-term statistics compared to FDNS.

Similar observations hold for the case with $\beta = 20$, but unlike the $\beta = 0$ case, the forecast accuracy of CNN-LES models trained through cyclical lr is demonstrably better than for the CNN-LES models trained with fixed lr. In each case, the CNN-LES runs outperforms the Smagorinsky and Leith runs in terms of forecasting accuracy; with Leith's results very similar to the Smagorinsky's ones (not shown).

4.2.2. Long-Term Accuracy

Short term forecasts are of direct interest to problems like weather forecasting but long term forecasts are more relevant for climate studies. There is no a priori reason to expect that high accuracy in the former implies the same for the latter, especially when concerning data-driven models whose training can be highly task specific. In fact, Dresdner et al. (2022) find that their turbulence parameterized solutions are matched in forecast accuracy by a purely data-driven auto-regressive NN model through the Encoder-Process-Decoder framework used in (Stachenfeld et al., 2021), a model that is ultimately unstable over long times. Our objective is to validate our CNN-LES setup for numerical stability and fidelity over long time scales, with the forecast accuracy being a mere side-effect. Given our aggressive choices regarding the size of the CNN we run our models for substantially longer times than in other recent works concerned with neural turbulent closures. Guan, Subel, et al. (2022) compute their online CNN-LES runs for (Note that our computational setup is identical to theirs, except our Re value is marginally larger) $2 \times 10^6\ \Delta t$ but we compute to $T = 6 \times 10^6\ \Delta t$ corresponding to 375,000 Δt_{LES}. As explained in Section 3.5, this also ensures convergence of metrics like the probability distribution function, which can now be computed directly without resorting to KDE.

In this section, we highlight the long term accuracy of the four specific CNN models described in Sections 4.1 and 4.2.1 for the six quantities chosen in Section 3.5 as diagnostics for measuring solution fidelity; these are the 1-dimensional time-averaged energy spectrum, $E(k)$, the SGS divergence spectrum, $|\Pi(k)|^2$, the cross-scale energy and enstrophy fluxes, $E_{\mathrm{flux}}(k)$ and $Z_{\mathrm{flux}}(k)$ and the 1-D probability distribution functions, $P(\bar{\omega})$ and $P(\Pi)$ computed over the course of the entire online run. A comparison of the FDNS runs with the corresponding four CNN-LES runs is shown in Figure 7. We find that all four cases have good agreement across the six shown metrics, with the highest overall accuracy being observed in the two cases with cyclical learning rate but different inputs. The SGS spectrum, $|\Pi(k)|^2$ and PDF, $P(\Pi)$ has as high a degree of accuracy in the online runs as in the offline tests (Figure 5) which also ties in with the high accuracy of the obtained structural flow metrics like $E(k)$ and $P(\bar{\omega})$ and the dynamical metrics of the cross-scale fluxes. An implication of this result is that Galilean invariance is, evidently, not a particular difficult physical symmetry to learn in these cases.

However, these results do not translate to the case with $\beta = 20$ (Figure 8) when the run corresponding to cyclical annealed lr and $(\bar{u}, \bar{v})$ input (blue curves in Figure 8) substantially outperforms on all metrics except for $P(\Pi)$. While the role of the cyclical annealing in model robustness and fidelity is broadly expected, it is surprising that choosing Galilean invariance a priori in our modeling (through using $(\bar{\omega}, \bar{\sigma}_n, \bar{\sigma}_s)$ as inputs) actually *hinders* online accuracy. The precise reason is unclear but it evidently has to do with the presence of strong eddy-driven zonal jets in the $\beta = 20$ case that are absent when $\beta = 0$. This observation seems to imply that for the case of geophysical turbulence, it might be better to choose $(\bar{u}, \bar{v})$ as inputs to the CNNs. It is also interesting to note that while $\Pi(k)$ is accurately predicted as is the vorticity distribution, $P(\bar{\omega})$, the tails of the distribution $P(\Pi)$ are missed.

Interestingly, we observe that our CNN solutions have here high online accuracy across both short and long-time scales (i.e., across both "weather" and "climatic" regimes). Previous results do not report on such solutions. While Dresdner et al. (2022) do not examine long-time fidelity, Frezat et al. (2022) find that their best long term accurate solutions (trained using their fully online methodology) actually have poor forecast accuracy and lose out to an offline-trained CNN model that is numerically unstable at long times. Consistent with this result, Ross et al. (2023) find that their forecast accuracy does not in general correlate with long-term fidelity. To underscore this result further, we evaluate the online performance of the best CNN-LES solutions for the $\beta = 0$ and $\beta = 20$ cases (among the cases shown in Figures 7 and 8) in Figure 9 with reference to various Smagorinsky cases shown earlier in Figure 6. Note that the Smagorinsky solution with the best accuracy in capturing the distribution, $P(\bar{\omega})$, (corresponding to $c_s = 0.025$) has a pile up of energy at small scales (Figures 9a and 9b). Furthermore, the

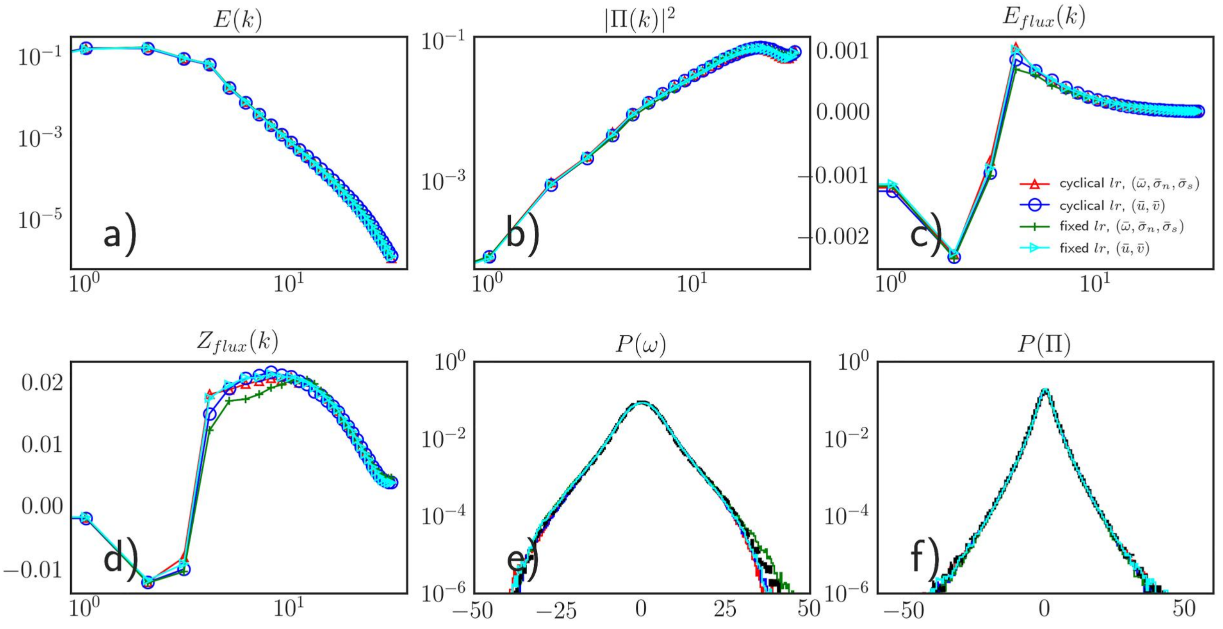

Figure 7. Comparison of FDNS data (dashed black line) with online runs across four different cases shown in Figures 6 and 5 as indicated on the legend, for $\beta = 0$ for the diagnostic quantities indicated. Note that the different curves are barely discernible in panels (a, b, f) and to a certain extent (e). The real points of difference emerge in the two cross-scale energy fluxes, (c, d); here the two cases with $(\bar{u}, \bar{v})$ are the most accurate. The markers have been removed from (e, f) to reduce clutter. Observe the inverse energy transfer ($E_{flux}(k) < 0$) for scales smaller than the forcing wavenumber ($k_f = 4$) and a corresponding forward enstrophy transfer ($Z_{flux}(k) > 0$) for $k > k_f$.

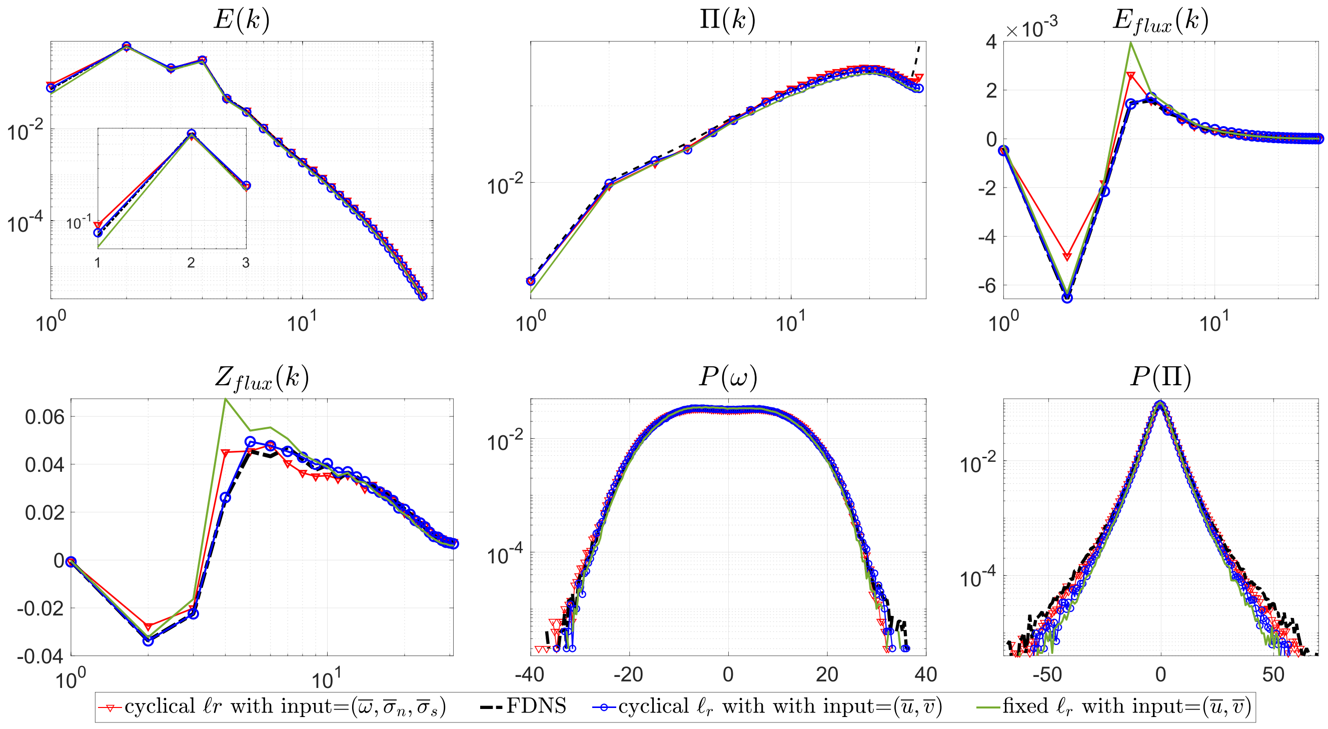

Figure 8. Same as Figure 7 but for $\beta = 20$. The case corresponding to a fixed lr and $(\bar{\omega}, \bar{\sigma}_s, \bar{\sigma}_n)$ inputs is not shown because of poor accuracy. The inset in a) shows a blow-up of the low-wavenumber energy spectrum of each CNN-LES model. Note that unlike in Figure 7, the CNN-LES model learned through cyclic lr with $(\bar{u}, \bar{v})$ inputs (blue curve with circles) substantially outperforms in accuracy the other CNN-LES models.

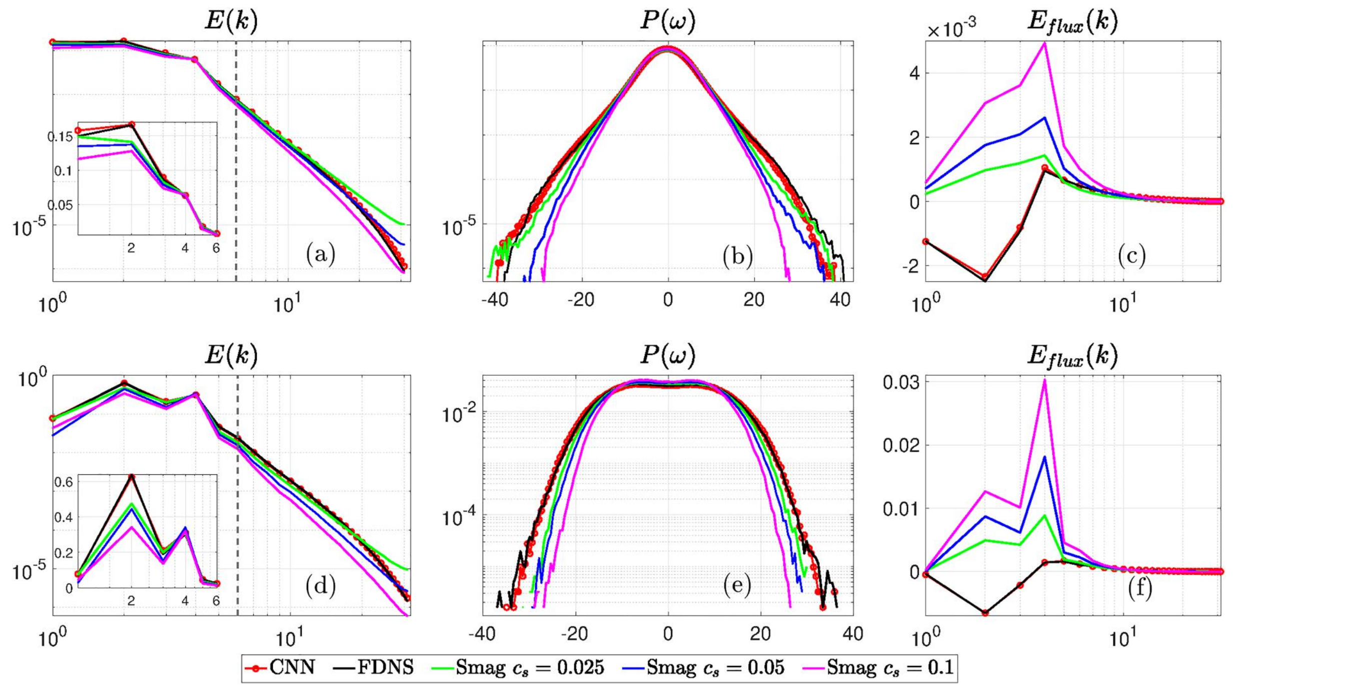

Figure 9. Comparison of long-term online performance of the best Convolutional Neural Networks models (both cyclic *lr* cases) from Figures 7 and 8 with the respective Smagorinsky runs which are run for identical times as the CNN-LES cases. Only a few of the key metrics from the earlier figures are shown for brevity. Insets highlight the large-wavenumber comparisons between CNN-LES and Smagorinsky spectra; note that these are shown with a linear *y*-axis instead of logarithmic to highlight the differences further. The vertical dashed line in panels (a, d) shows the rightmost extent of the inset. Note that, compared to the online CNN-LES solutions, the Smagorinsky ones fail to capture the inverse cascade.

Smagorinsky cases fail to capture the tails of the vorticity distribution that the CNN-LES solutions do consistently; this is especially true for the jets cases (Figure 9e).

Our online accuracy for the jets case is comparable to that obtained by Frezat et al. (2022) using their online-learning framework. The study by Ross et al. (2023) employs a similar offline-learning approach as ours for parameterizing their corresponding jets but they instead focus on examining the robustness of CNN parameterizations. To this end, they train their CNN models on their eddy case ($\beta = 0$) and evaluate this model online on their jets case ($\beta > 0$). They find that the CNN models do not transfer well without training on jets examples. Since our present study is primarily focused on studying scale locality in CNN-based parameterizations, we leave our explorations in obtaining robust CNN parameterizations to future work.

Finally, our best models, shown here and the preceding two sections are cherry picked from a large range of models trained over a range of hyperparameters. In the next section we examine the behavior of our CNN-LES models across this hyperparameter space depending on the training methodology or input choices.

4.2.3. Hyperparameter Dependence of Model Fidelity

In preceding sections we have demonstrated that by searching in the (lr, wd)-hyperparameter space we can obtain high-fidelity solutions with shallow CNNs. However, it is important to know how common or rare such high-fidelity solutions might be to find, and to in turn to be able to find the most accurate online solutions easily given such a large number of runs. Such an evaluation, though seemingly mundane and somewhat tedious, is especially relevant for foreshadowing the challenges in parameterizing more complex flow configurations observed in the climate system when such large hyperparameter searches might not be viable.

In Section 3.5, we define six different difference metrics based on the six quantities shown in Figures 7 and 8 that compare errors in the online CNN-LES solutions relative to the FDNS solutions. Here, we construct a metric that consists of simply summing up the six metrics, defined thus as

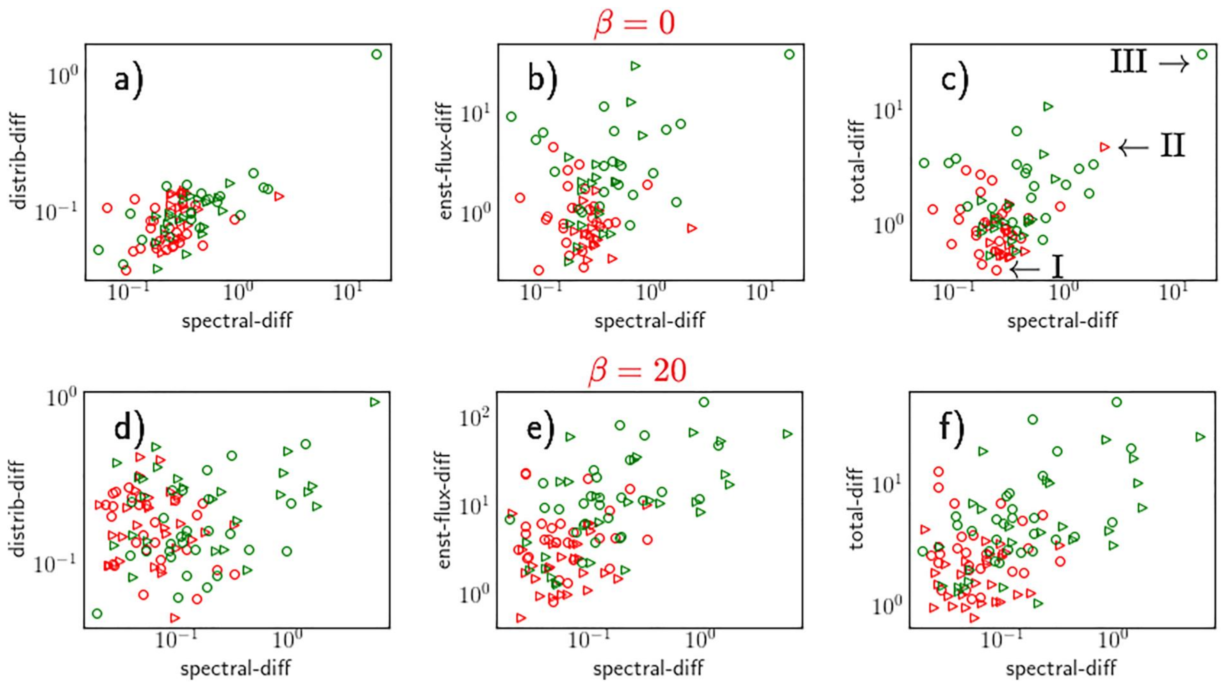

Figure 10. A comparison of three of six key difference metrics defined in Section 3.5 against the sixth online metric, the spectral-diff that measures the difference between the energy spectra for online runs based on CNNs trained across a range of (lr, wd)-values in the hyperparameters space. (a–c) $\beta = 0$ and (d–f) $\beta = 20$. The red markers denote online CNN-LES models trained with cyclical lr while green ones with fixed lr. The circles use $(\bar{\omega},\bar{\sigma}_n,\bar{\sigma}_s)$ as inputs while the triangles use $(\bar{u},\bar{v})$. The total-diff metric is defined in Equation 37. The cases I, II and III marked (c) are visualized in Figure 12.

$$\texttt{total}-\texttt{diff} = (\texttt{spectral}-\texttt{diff}+\texttt{spectral}-\texttt{sgs}-\texttt{diff}+\texttt{enst}-\texttt{flux}-\texttt{diff} \\ +\texttt{energy}-\texttt{flux}-\texttt{dim}+\texttt{distrib}-\texttt{sgs}-\texttt{diff})/5. \tag{37}$$

To understand how these metrics vary in the hyperparameter space, we display three of the most important metrics, *other than* spectral − diff (which measures the accuracy in estimating the energy spectrum) relative to spectral − diff. This helps us assess how the various metrics relate (implicitly) to each other by observing their variation against spectral − diff Figure 10 show the results for $\beta = 0$ and $\beta = 20$. We examine each of these two cases as the results show us variations of the metrics across the (lr, wd)-space. We start by examining the total − diff metric (Figures 10c–10f) which highlights two important points. First, we observe the cyclical annealed cases (red markers) are more tightly clustered and closer to the origin (i.e., lower errors) while the fixed lr cases (green markers) have a much larger spread in performance. This feature indicates that not only does the cyclic lr procedure lead to more robust solutions overall by producing a narrower range of behaviors as the hyperparameters are changed, they are also overall more accurate as well. This is evident in the $\beta = 0$ and even more so, the $\beta = 20$ case. The cyclic lr cases also have substantially lower flux errors (Figures 10b and 10e) compared to the fixed lr cases though the other metrics are harder to distinguish (as demonstrated in the Supporting Information S1). The enhanced accuracy and robustness of the cyclical lr cases are even better captured visually using box-filter plots (Figures 11a–11c) that show the median (solid black line in the middle of each box) and the various quantile ranges. Clearly, the cyclic lr cases have overall lowest total − diff (i.e., better accuracy), lower median error across hyperparameters, narrower quantile ranges and smaller outliers (diamonds above each box.) A narrower range of errors across hyperparameters means a greater robustness in results. Furthermore, the outliers (the diamonds on top of each plot) are substantially worse for the fixed lr cases.

Second, within the cyclic lr cases (red markers), solutions that use $(\bar{\omega},\bar{\sigma}_n,\bar{\sigma}_s)$ as inputs (red circles) perform best when $\beta = 0$ though not by a lot. However, when $\beta = 20$ almost all the best solutions are for the cases which use $(\bar{u},\bar{v})$ as inputs (red triangles). Overall, across both $\beta = 0$ and $\beta = 20$ the differences between the different input choices are a lot harder to discern from the box-whisker plots (Figures 11b and 11d). Curiously for both $\beta = 0$ and $\beta = 20$, the fidelity of $P(\Pi)$ (see Supporting Information S1) seems to have the opposite relationship with the flow

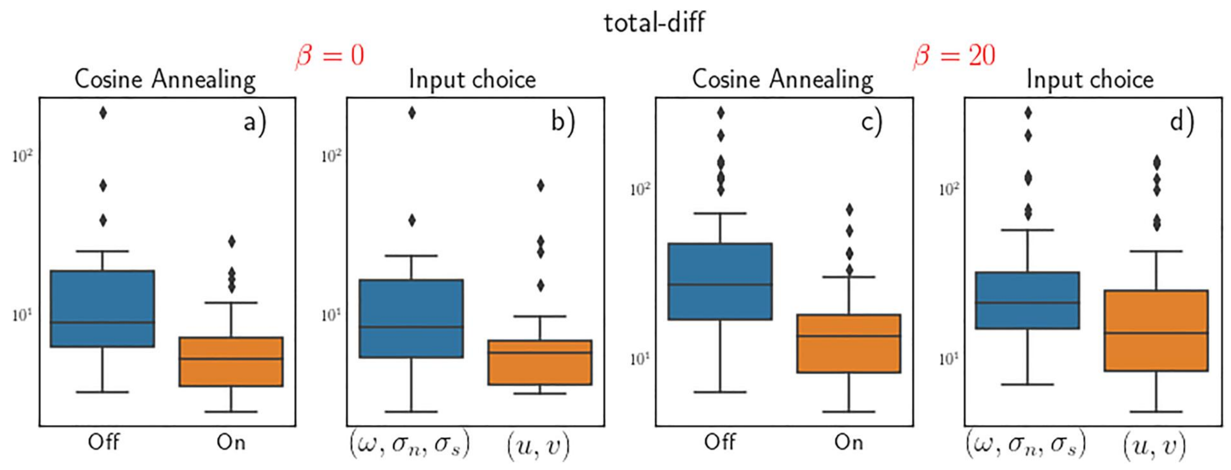

Figure 11. Box-whisker plot of the total-diff metric, Equation 37, showing the median and quantile ranges for different choices of learning rate annealing (cyclical or fixed) or inputs, $(\bar{\omega}, \bar{\sigma}_n, \bar{\sigma}_s)$ or $(\bar{u}, \bar{v})$ for both $\beta = 0$ and $\beta = 20$.

metrics with the fixed lr cases having higher accuracy in this metric. We believe this is not a significant issue as tail errors in $P(\Pi)$ are not insignificant even for the offline cases (Figures 5k and 5l).

Our exhaustive parameter search allows us to highlight a specific case with extremely large online error, an order of magnitude larger than that in the best case, while being numerically stable for long times (marked as Case III in Figure 10f). To understand this anomalous solution better, we visualize snapshots of $\bar{\omega}$ and Π for three cases corresponding to the CNN-LES runs with best and middling online runs along with this worst online run. A surprising fact is that both I (the best case) and II (the "middle" case) have very similar flow structure while III looks noisy and unphysical. Curiously though, the three cases have similar offline accuracy, again highlighting the challenges in predicting online accuracy from offline accuracy.

A final point of concern that is not specifically a hyperparameter is the random seed of the pseudo-random number generator used by Pytorch. In our foray across hyperparameter space, we do not set the seed so that Pytorch assigns a different value each time. While the results of cyclical versus fixed lr are found seem to be statistically robust across different solutions, there might be some concern that some of these are related to the specific choices of the seed. In Appendix A1, we examine the role of the random seed in detail and find that even across variations of seed, the cyclical lr cases maintain the twin advantages of fidelity and robustness.

The findings above raise questions about the relationship between the large number of online-stable CNN parameterizations that we find as part of our hyperparameter searches. The close structural similarity of the SGS error observed in Figure 5 across different input choices and optimization methods indicates that we may be learning a common family of parameterizations. While one might hope to find a single unique parameterization that is globally optimal under some accuracy metrics for complex turbulent flows, this can be challenging, if at all possible, to infer from finite data sets. One might wonder how these results change for a fixed NN architecture but in the infinite data limit. Such questions are tied to the study of ergodicity properties. If ergodicity applies, the uniqueness (and existence) of a parameterization that averages out optimally the small scales is guaranteed in the infinite data limit (Chekroun et al., 2020, Theorem 4) although the use of finitely sampled data can lead to various admissible solutions with different scoring as reported in Figures 10 and 11. A detailed study of this phenomenon under the lens of a possible ergodicity of the underlying forced flows (Chekroun & Glatt-Holtz, 2012; Foias et al., 2001; Hairer & Mattingly, 2006, 2011) goes beyond the scope of the present work.

4.2.4. Offline Versus Online Predictions

The ultimate goal of any data-driven parameterization is to reach high-quality online fidelity and accuracy. To achieve such a feat requires typically to run the model over long times and evaluating the online solutions across various metrics such as reported the sections above. Such an approach becomes particularly expensive as one tests over a wide swaths in the hyperparameter space even when using shallow neural networks like in this study. Ideally one would hope that one could simply compute offline accuracy on the held-out test set and use that to

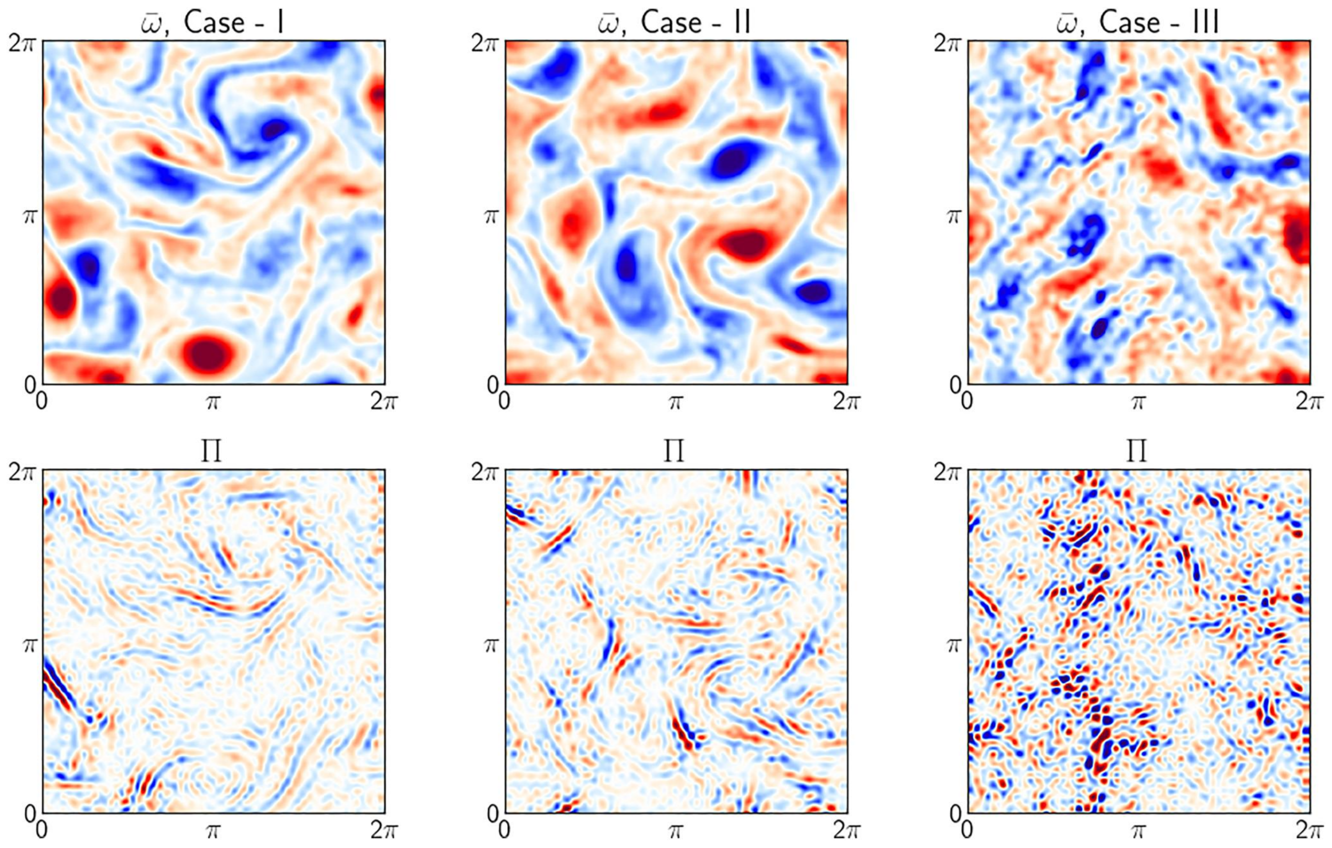

Figure 12. Snapshots of $\bar{\omega}$ and corresponding Π for three online corresponding to cases marked I, II and III found in Figure 10f.

estimate online accuracy. But this remains idealistic. In that respect, Figure 13 compares offline accuracy with the total-diff online accuracy measure defined in Equation 37 across the (lr, wd)-hyperparameter space. The results are shown for $\beta = 0$ in Figures 13a–13c and for $\beta = 20$ in Figures 13d–13f. Unlike in the previous section, we separate the results for different filter sizes of the hidden layers. It can be observed that as the number N_f of convolutional filters is increased corresponding to an increase in the number of CNN parameters, the offline error (the x-axis) systematically decreases with increasing N_f, as manifested by the shift of the cloud of points toward the origin of the x-axis. However, the corresponding decrease in online error (the y-axis) is actually rather small and consequently difficult to discern. This observation highlights the crucial point that a large decrease in offline error might lead to only small gains in online CNN-LES performance. It also foreshadows the result that deeper CNNs that would inevitably improve offline accuracy do not necessarily lead to more accurate online solutions.

Concurrently, we note that the correlation between online and offline performance remains weak implying that diagnosing the latter is insufficient for knowledge of the former, our actual objective. One reason is that our metrics for offline performance, the mean square error, calculated over 400 snapshots and spatially averaged through the norm involved therein, does not reflect necessarily the dynamical differences in the data produced by the different CNN-LES models. It is conceivable that more physically grounded metrics of offline performance might in fact improve online skills without running more expensive CNN-LES runs for long times; this remains an area of active investigation.

5. Discussion and Concluding Remarks

5.1. Physical and Computational Implications of SGS Near-Locality

The primary result in this work is the effective learning of accurate parameterizations of two-dimensional and geostrophic turbulent flows with shallow two-layer CNNs. This accomplishment essentially implies that the SGS stresses for these problems have a spatially nearly local (or weakly non-local) dependence on the coarse fields. In

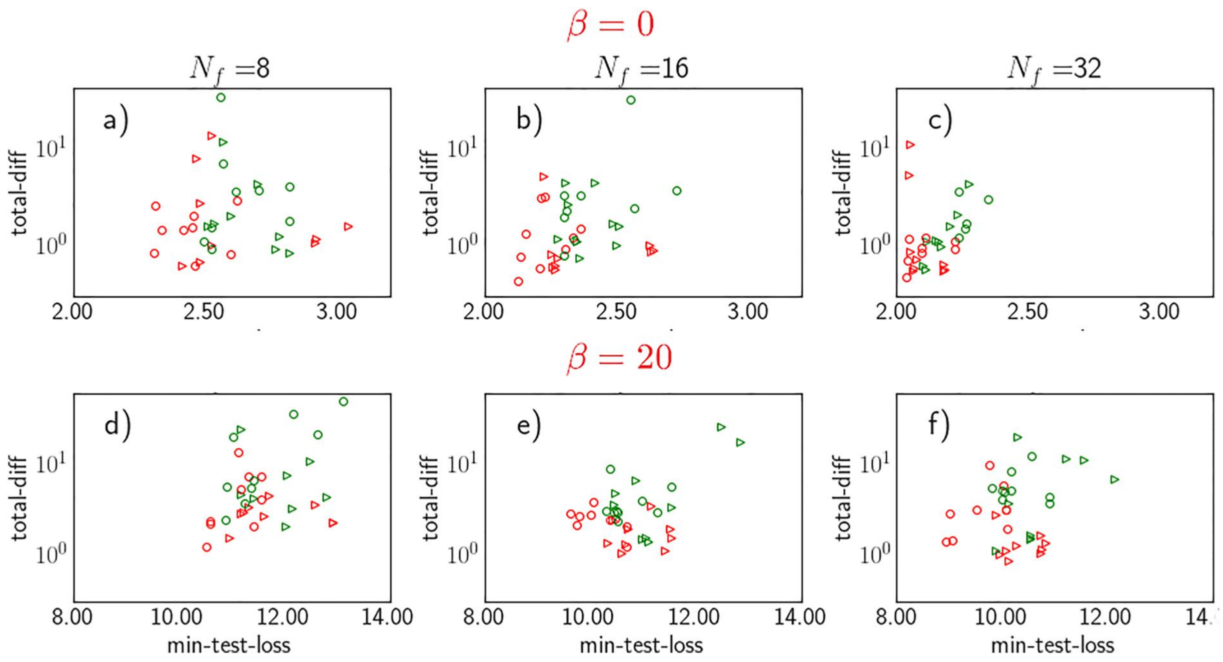

Figure 13. Comparison of online total-diff metric versus offline test-error over the (lr, wd)-hyperparameter space for (a–c) $\beta = 0$ and (d–f) $\beta = 20$. The results are grouped by the different values of the convolutional filters used, given by N_f; details of the Convolutional Neural Networks architecture are given in Section 3.1. Markers color-coding is the same as in Figure 10.

other words we have inferred the physics of this class of problems through direct construction of a spatial nearly local parameterization. This is not in general true for all classes turbulent flows; for example, convective motions in the oceanic surface-mixed layer and broadly in the Earth's troposphere (processes that also result in cloud formation and subsequent precipitation) can be strongly correlated across a significant fraction of the vertical extent within which they take place. Recent ML models that successfully parameterize vertical fluxes in the atmosphere use NNs that span the most of the air column (Yuval & O'Gorman, 2020; Yuval et al., 2021). Similarly, traditional empirical physics-based parameterizations in the oceanic surface mixed layer employ a simple vertically non-local flux to represent convective turbulence. While current class of atmospheric ML-based parameterizations do not represent the horizontal eddy-fluxes, evidently for reasons of simplicty, our work here suggests that more complete representations would likely be spatially (nearly) local in the horizontal and have a greater degree of non-locality in the vertical; in the ocean the non-locality should be limited to the extent of the surface-mixed layer where air-sea fluxes are actively felt.

A secondary implication of this work is for purposes of numerical computation of oceanic, atmospheric and climate models. These models often rely on domain decomposition in the horizontal for numerical solution over large clusters of compute nodes, with inter-process communications typically limited to a small number of points on the boundaries of the sub-domains. Thus employing data-driven parameterizations precludes a high degree of spatial non-locality which would make the inter-process exchanges prohibitively expensive; note that this is not an issue in the vertical direction (i.e., along the direction of gravity). Our results above demonstrate that the horizontal SGS fluxes can be modeled through shallow CNN models and are thus easier to incorporate into existing pipelines than by using more cumbersome deep CNNs (Partee et al., 2022; Sane et al., 2023; Zhang et al., 2023). Consequently, our findings highlight the delicate balance between enhancing forecasting accuracy and managing computational costs, underscoring the potential of smaller, optimized CNNs in effectively advancing atmospheric and oceanic modeling without overwhelming computational resources. This equilibrium between forecast skill and efficiency is crucial for the practical application of data-driven parameterizations in large-scale climate models.

5.2. Theoretical Consequences and Toward Interpretability

Our neural closure results with shallow CNNs presented in this study are valid for cutoffs within the inertial range and for high Reynolds numbers. This problem is known to be difficult as small errors at the level of the SGS typically amplify the errors at the large scales due to the inverse cascade (Jansen & Held, 2014; Piomelli et al., 1991). To dispose of SGS parameterizations at low cutoff levels for such turbulent flows with a controlled error is thus one of the challenges to resolve. The accuracy and stability of our closure results have an interesting mathematical interpretation. These results hint for example, at the existence of a nonlinear function Φ_{CNN} such that the SGS, Π, satisfies, after spin up, a relation of the form

$$\Pi = \Pi_{CNN}(\overline{u},\overline{v}) + \epsilon, \tag{38}$$

where the residual ϵ is a spatio-temporal function whose fluctuations are controlled and small in a mean square sense. In Equation 38, Π_{CNN} denotes the function found by means of shallow CNNs trained by minimizing our loss function through cyclical annealing. Actually, Equation 38 is a consequence of the very construction of Π_{CNN} obtained by minimization of our loss function Equation 28 along with its regularization terms.

We have shown that with the appropriate hyperparameter searching in the learning rate and weight decay coefficient space, as well as usage of cyclical learning rate, small residuals can be reached offline—with shallow and weakly local CNNs—while leading to stable and accurate online solutions. As Π_{CNN} is a nonlinear functional of the coarse-grained variables only, our good closure skills suggest that the knowledge of Π_{CNN} is amply sufficient to achieve good performances in particular in the β-case when compared to other recent neural closures (Ross et al., 2023), for a range of physically and computationally interesting cutoffs within the inertial range.

By interpreting Π_{CNN} as a nonlinear functional linking the coarse-grained (resolved) variables to the (unresolved) SGS, and ϵ in Equation 38 as a least-square error, our neural turbulent closure results together with related recent studies restore some credentials to ideas proposed in the late 80s by Foias et al. (1988, 1991). In these works, turbulent solutions to the 2D Navier–Stokes equations were pursued as trajectories evolving in the phase space, within some thin neighborhood (in a mean square sense) of a finite-dimensional manifold parameterizing the small scales in terms of the large ones. However, attempts to construct these parameterizations based on analytical formulas derived from the model's equations were successful only for cutoff wavenumbers within or close to the dissipation range (Pascal & Basdevant, 1992). Only recently, it has been shown that such ideas can handle much more interesting situations by allowing for data-informed optimization of the underlying analytic parameterizations. Then, relevant approximate parameterizations in a mean-square sense can be derived in a systematic fashion for a diversity of applications (Chekroun et al., 2017, 2020, 2021, 2023) although this approach has not been yet tested for 2D turbulence.

The use of neural networks thus opens up new perspectives on this old problem. The aforementioned results show indeed that data-driven formulas of such parameterizations are accessible for much lower and computationally relevant cutoffs within the inertial range for 2D turbulence. However, note that lowering the cutoff scale is inevitably prone to the emergence of memory and/or stochastic terms at some point (Lucarini & Chekroun, 2023).

The discovery of nearly-local shallow CNN-parameterizations to achieve this feat is likely to be interpretable and generalizable because of its intrinsic low dimensionality. We hope thus to reconcile the previous failures in analytic attempts with the recent empirical successes by seeking for new analytic formulas for closure that would exploit the discovery of our nearly-local shallow CNNs.

5.3. Machine Learning Versus Physical Design Choices

In their recent work, Ross et al. (2023) suggest that the focus of ML-based parameterizations should be less on ML details like NN architectures or optimization techniques but should instead be focused on physical design choices, in this case on the choice of the inputs/outputs or the coarse-graining filter. Based on our results, we suggest that one should in fact focus on both aspects, and in some cases these are closely related for example, our choice of the 2-layer CNN architecture based on the nearly local character of the SGS coarse-field components interacting with the small scales. This study shows that the choice of optimization techniques like cyclical annealed *lr* can make the learning task substantially easier and generally more robust; this would be even more relevant when the problems being considered are not simplified turbulent models like the ones dealt with here and in the studies referred to as in this study. Furthermore, hyperparameter searches, while tedious, can lead to substantial improvements in task accuracy and efficiency, and therefore should not be ignored.

Appendix A: The Random Seed and How It Interacts With Learning Rate Annealing

The training of a NN is often done tabula rasa, that is, from a blank state wherein the NN weights are assigned from some form of random distribution, either a scaled uniform or Gaussian distribution. The scaling factor depends on the choice of the activation function; we use the He initialization (He et al., 2015) that is the default CNN initialization in Pytorch. However, independent of the distribution of the weights, tha actual initial values assigned to the weights depend on the random seed value of the pseudo-random generator employed by Pytorch. Here we examine how the seed value affects training when cyclical learning rate annealing is employed as opposed to for the case of fixed learning rate. For this purpose we pick a specific choice of (lr, wd) hyperparameters and input choice which for this section we choose to be $(\bar{u}, \bar{v})$. The (lr, wd) cases are those with the best CNN-LES runs shown in Figure 7 for the fixed and cyclical lr cases. After we pick these hyperparameters, we pick 18 different values of the random seed (we simply choose values from 10 to 180 with increments of 10 but these are arbitrary and others can be chosen) and run the fixed and cyclical lr cases and compare offline and long-time online accuracy as before. First, we found that all the 18 cyclical lr cases were unstable but 4 of the 18 fixed lr cases were numerically unstable in online runs, thus showing the greater sensitivity not just of fidelity but also of stability with fixed lr.

Of the stable cases, we show box-whisker plots that record the median and quantile ranges of online and offline error across these seed values for the two learning rate choices. The results are shown in Figure A1. The offline accuracy changes are pretty marginal in the best models for both cases (within 5%) though on average the cosine annealing cases ("On") actually have marginally higher offline errors though in a narrower range. Next, we find that the variations in online fidelity (measured by the total-diff metric) have a much greater spread in the fixed lr cases compared to the cyclical lr cases. Thus the cyclical lr cases are substantially more robust in online performance and have better accuracy overall even across variations of the random seed supporting the other results in the paper. Typically the intra-seed online error variations (Figure A1a) are smaller than those observed when varying (lr, wd) (Figures 11a and 11c).

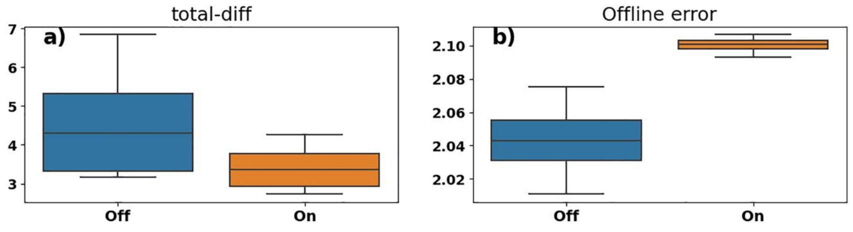

Figure A1. Box-whisker plot of the total-diff metric, Equation 37, showing the median and quantile ranges across variations of the random seed for different choices of learning rate annealing (cyclical or fixed) for $(\bar{u}, \bar{v})$ inputs and specific $\beta = 0$ cases corresponding to the best online runs in Figure 7.

Data Availability Statement

The code for the entire pipeline including simulated data generation for training and the training, testing and validation of the NN-parameterization can be downloaded at Srinivasan (2023).

Acknowledgments
This work is supported by the Office of Naval Research (ONR) Multidisciplinary University Research Initiative (MURI) Grant N00014-20-1-2023. It benefitted greatly through discussions with Pavel Perezhogin; the critiques and insighful comments from Adam Subel, Dhruv Balwada and the two other anonymous reviewers are greatly appreciated.

References

Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. arXiv preprint arXiv:1607.06450.
Bachman, S. D., Fox-Kemper, B., & Pearson, B. (2017). A scale-aware subgrid model for quasi-geostrophic turbulence. *Journal of Geophysical Research: Oceans*, *122*(2), 1529–1554. https://doi.org/10.1002/2016jc012265
Bakas, N. A., Constantinou, N. C., & Ioannou, P. J. (2015). S3t stability of the homogeneous state of barotropic beta-plane turbulence. *Journal of the Atmospheric Sciences*, *72*(5), 1689–1712. https://doi.org/10.1175/jas-d-14-0213.1
Bengio, Y., & Delalleau, O. (2011). On the expressive power of deep architectures. In *Algorithmic learning theory: 22nd international conference, alt 2011, ESPOO, Finland, October 5-7, 2011. Proceedings 22* (pp. 18–36).
Botev, Z. I., Grotowski, J. F., & Kroese, D. P. (2010). Kernel density estimation via diffusion. *The Annals of Statistics*, 2916–2957.

Brachet, M., Meneguzzi, M., Politano, H., & Sulem, P. (1988). The dynamics of freely decaying two-dimensional turbulence. *Journal of Fluid Mechanics*, *194*(1), 333–349. https://doi.org/10.1017/s0022112088003015

Brenowitz, N. D., Henn, B., McGibbon, J., Clark, S. K., Kwa, A., Perkins, W. A., et al. (2020). Machine learning climate model dynamics: Offline versus online performance. arXiv preprint arXiv:2011.03081.

Carnevale, G., McWilliams, J., Pomeau, Y., Weiss, J., & Young, W. (1991). Evolution of vortex statistics in two-dimensional turbulence. *Physical Review Letters*, *66*(21), 2735–2737. https://doi.org/10.1103/physrevlett.66.2735

Carnevale, G., McWilliams, J., Pomeau, Y., Weiss, J., & Young, W. (1992). Rates, pathways, and end states of nonlinear evolution in decaying two-dimensional turbulence: Scaling theory versus selective decay. *Physics of Fluids A: Fluid Dynamics*, *4*(6), 1314–1316. https://doi.org/10.1063/1.858251

Chekroun, M. D., & Glatt-Holtz, N. E. (2012). Invariant measures for dissipative dynamical systems: Abstract results and applications. *Communications in Mathematical Physics*, *316*(3), 723–761. https://doi.org/10.1007/s00220-012-1515-y

Chekroun, M. D., Liu, H., & McWilliams, J. (2020). Variational approach to closure of nonlinear dynamical systems: Autonomous case. *Journal of Statistical Physics*, *179*(5–6), 1073–1160. https://doi.org/10.1007/s10955-019-02458-2

Chekroun, M. D., Liu, H., & McWilliams, J. (2021). Stochastic rectification of fast oscillations on slow manifold closures. *Proceedings of the National Academy of Sciences*, *118*(48), e2113650118. https://doi.org/10.1073/pnas.2113650118

Chekroun, M. D., Liu, H., & McWilliams, J. C. (2017). The emergence of fast oscillations in a reduced primitive equation model and its implications for closure theories. *Computers & Fluids*, *151*, 3–22. https://doi.org/10.1016/j.compfluid.2016.07.005

Chekroun, M. D., Liu, H., & McWilliams, J. C. (2023). Optimal parameterizing manifolds for anticipating tipping points and higher-order critical transitions. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *33*(9). https://doi.org/10.1063/5.0167419

Curcic, M. (2019). A parallel Fortran framework for neural networks and deep learning. *ACM SIGPLAN Fortran Forum*, *38*(1), 4–21. https://doi.org/10.1145/3323057.3323059

Dresdner, G., Kochkov, D., Norgaard, P., Zepeda-Núñez, L., Smith, J. A., Brenner, M. P., & Hoyer, S. (2022). Learning to correct spectral methods for simulating turbulent flows. arXiv preprint arXiv:2207.00556.

Eyink, G. L. (2005). Locality of turbulent cascades. *Physica D: Nonlinear Phenomena*, *207*(1–2), 91–116. https://doi.org/10.1016/j.physd.2005.05.018

Eyink, G. L., & Aluie, H. (2009). Localness of energy cascade in hydrodynamic turbulence. I. Smooth coarse graining. *Physics of Fluids*, *21*(11), 115107. https://doi.org/10.1063/1.3266883

Farrell, B. F., & Ioannou, P. J. (2007). Structure and spacing of jets in barotropic turbulence. *Journal of the Atmospheric Sciences*, *64*(10), 3652–3665. https://doi.org/10.1175/jas4016.1

Fisher, B. (2008). The cross-correlation and wiener-khinchin theorems. *Journal of Neuroscience*, 8107–8115.

Foias, C., Manley, O., Rosa, R., & Temam, R. (2001). Navier-Stokes equations and turbulence. In *Encyclopedia of mathematics and its applications* (Vol. 83). Cambridge University Press.

Foias, C., Manley, O., & Temam, R. (1988). Modeling of the interaction of small and large eddies in two-dimensional turbulent flows. *ESAIM: Mathematical Modelling and Numerical Analysis*, *22*(1), 93–118. https://doi.org/10.1051/m2an/1988220100931

Foias, C., Manley, O. P., & Temam, R. (1991). Approximate inertial manifolds and effective viscosity in turbulent flows. *Physics of Fluids A: Fluid Dynamics*, *3*(5), 898–911. https://doi.org/10.1063/1.858212

Frezat, H., Sommer, J. L., Fablet, R., Balarac, G., & Lguensat, R. (2022). A posteriori learning for quasi-geostrophic turbulence parametrization. arXiv preprint arXiv:2204.03911.

Guan, Y., Chattopadhyay, A., Subel, A., & Hassanzadeh, P. (2022). Stable a posteriori LES of 2D turbulence using convolutional neural networks: Backscattering analysis and generalization to higher Re via transfer learning. *Journal of Computational Physics*, *458*, 111090. https://doi.org/10.1016/j.jcp.2022.111090

Guan, Y., Subel, A., Chattopadhyay, A., & Hassanzadeh, P. (2022). Learning physics-constrained subgrid-scale closures in the small-data regime for stable and accurate LES. arXiv preprint arXiv:2201.07347.

Guillaumin, A. P., & Zanna, L. (2021). Stochastic-deep learning parameterization of ocean momentum forcing. *Journal of Advances in Modeling Earth Systems*, *13*(9), e2021MS002534. https://doi.org/10.1029/2021ms002534

Hairer, M., & Mattingly, J. (2006). Ergodicity of the 2D Navier-Stokes equations with degenerate stochastic forcing. *Annals of Mathematics*, *164*(3), 993–1032. https://doi.org/10.4007/annals.2006.164.993

Hairer, M., & Mattingly, J. (2011). A theory of hypoellipticity and unique ergodicity for semilinear stochastic PDEs. *Electronic Journal of Probability*, *16*(none), 658–738. https://doi.org/10.1214/ejp.v16-875

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision* (pp. 1026–1034).

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, *2*(5), 359–366. https://doi.org/10.1016/0893-6080(89)90020-8

Hussein, A., Gaber, M. M., Elyan, E., & Jayne, C. (2017). Imitation learning: A survey of learning methods. *ACM Computing Surveys*, *50*(2), 1–35. https://doi.org/10.1145/3054912

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning* (pp. 448–456).

Jansen, M. F., & Held, I. M. (2014). Parameterizing subgrid-scale eddy effects using energetically consistent backscatter. *Ocean Modelling*, *80*, 36–48. https://doi.org/10.1016/j.ocemod.2014.06.002

Keisler, R. (2022). Forecasting global weather with graph neural networks. arXiv preprint arXiv:2202.07575.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., & Hoyer, S. (2021). Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, *118*(21), e2101784118. https://doi.org/10.1073/pnas.2101784118

Kraichnan, R. H. (1971). Inertial-range transfer in two-and three-dimensional turbulence. *Journal of Fluid Mechanics*, *47*(3), 525–535. https://doi.org/10.1017/s0022112071001216

Large, W., McWilliams, J. C., & Doney, S. C. (1994). Oceanic vertical mixing: A review and a model with a nonlocal boundary layer parameterization. *Reviews of Geophysics*, *32*(4), 363–403. https://doi.org/10.1029/94RG01872

Lele, S. K. (1992). Compact finite difference schemes with spectral-like resolution. *Journal of Computational Physics*, *103*(1), 16–42. https://doi.org/10.1016/0021-9991(92)90324-R

List, B., Chen, L.-W., & Thuerey, N. (2022). Learned turbulence modelling with differentiable fluid solvers: Physics-based loss functions and optimisation horizons. *Journal of Fluid Mechanics*, *949*, A25. https://doi.org/10.1017/jfm.2022.738

Loshchilov, I., & Hutter, F. (2016). Sgdr: Stochastic gradient descent with warm restarts. arXiv preprint arXiv:1608.03983.

Lucarini, V., & Chekroun, M. D. (2023). Theoretical tools for understanding the climate crisis from Hasselmann's programme and beyond. *Nature Reviews Physics*, *5*(12), 1–22. https://doi.org/10.1038/s42254-023-00650-8

Ma, C., Wang, J., & Weinan, E. (2019). Model reduction with memory and the machine learning of dynamical systems. *Communications in Computational Physics*, *25*(4), 947-962. https://doi.org/10.4208/cicp.oa-2018-0269

Marston, J., Conover, E., & Schneider, T. (2008). Statistics of an unstable barotropic jet from a cumulant expansion. *Journal of the Atmospheric Sciences*, *65*(6), 1955–1966. https://doi.org/10.1175/2007jas2510.1

Maulik, R., San, O., Rasheed, A., & Vedula, P. (2019). Subgrid modelling for two-dimensional turbulence using neural networks. *Journal of Fluid Mechanics*, *858*, 122–144. https://doi.org/10.1017/jfm.2018.770

McWilliams, J. C. (1984). The emergence of isolated coherent vortices in turbulent flow. *Journal of Fluid Mechanics*, *146*, 21–43. https://doi.org/10.1017/s0022112084001750

Orszag, S. A., & Israeli, M. (1974). Numerical simulation of viscous incompressible flows. *Annual Review of Fluid Mechanics*, *6*(1), 281–318. https://doi.org/10.1146/annurev.fl.06.010174.001433

Ott, J., Pritchard, M., Best, N., Linstead, E., Curcic, M., & Baldi, P. (2020). *A Fortran-Keras deep learning bridge for scientific computing*. Scientific Programming.

Partee, S., Ellis, M., Rigazzi, A., Shao, A. E., Bachman, S., Marques, G., & Robbins, B. (2022). Using machine learning at scale in numerical simulations with smartsim: An application to ocean climate modeling. *Journal of Computational Science*, *62*, 101707. https://doi.org/10.1016/j.jocs.2022.101707

Pascal, F., & Basdevant, C. (1992). Nonlinear Galerkin method and subgrid-scale model for two-dimensional turbulent flows. *Theoretical and Computational Fluid Dynamics*, *3*(5), 267–284. https://doi.org/10.1007/bf00717644

Pearson, B., Fox-Kemper, B., Bachman, S., & Bryan, F. (2017). Evaluation of scale-aware subgrid mesoscale eddy models in a global eddy-rich model. *Ocean Modelling*, *115*, 42–58. https://doi.org/10.1016/j.ocemod.2017.05.007

Perezhogin, P., Zanna, L., & Fernandez-Granda, C. (2023). Generative data-driven approaches for stochastic subgrid parameterizations in an idealized ocean model. arXiv preprint arXiv:2302.07984.

Piomelli, U., Cabot, W. H., Moin, P., & Lee, S. (1991). Subgrid-scale backscatter in turbulent and transitional flows. *Physics of Fluids A: Fluid Dynamics*, *3*(7), 1766–1771. https://doi.org/10.1063/1.857956

Rahaman, N., Baratin, A., Arpit, D., Draxler, F., Lin, M., Hamprecht, F., et al. (2019). On the spectral bias of neural networks. In *International conference on machine learning* (pp. 5301–5310).

Ramachandran, P., Zoph, B., & Le, Q. V. (2017). Searching for activation functions. arXiv preprint arXiv:1710.05941.

Rasp, S. (2020). Coupled online learning as a way to tackle instabilities and biases in neural network parameterizations: General algorithms and Lorenz 96 case study (v1. 0). *Geoscientific Model Development*, *13*(5), 2185–2196. https://doi.org/10.5194/gmd-13-2185-2020

Ross, A., Li, Z., Perezhogin, P., Fernandez-Granda, C., & Zanna, L. (2023). Benchmarking of machine learning ocean subgrid parameterizations in an idealized model. *Journal of Advances in Modeling Earth Systems*, *15*(1), e2022MS003258. https://doi.org/10.1029/2022ms003258

Sane, A., Reichl, B. G., Adcroft, A., & Zanna, L. (2023). Parameterizing vertical mixing coefficients in the ocean surface boundary layer using neural networks. arXiv preprint arXiv:2306.09045.

Souza, A. N., Wagner, G., Ramadhan, A., Allen, B., Churavy, V., Schloss, J., et al. (2020). Uncertainty quantification of ocean parameterizations: Application to the k-profile-parameterization for penetrative convection. *Journal of Advances in Modeling Earth Systems*, *12*(12), e2020MS002108. https://doi.org/10.1029/2020ms002108

Srinivasan, K. (2023). Turbulence closure with small, local neural networks: Forced two-dimensional and beta-plane flows [Software]. Zenodo. https://doi.org/10.5281/zenodo.8129236

Srinivasan, K., & Young, W. R. (2012). Zonostrophic instability. *Journal of the Atmospheric Sciences*, *69*(5), 1633–1656. https://doi.org/10.1175/JAS-D-13-0246.1

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(1), 1929–1958.

Stachenfeld, K., Fielding, D. B., Kochkov, D., Cranmer, M., Pfaff, T., Godwin, J., et al. (2021). Learned coarse models for efficient turbulence simulation. arXiv preprint arXiv:2112.15275.

Wang, P., Li, X., Yaras, C., Zhu, Z., Balzano, L., Hu, W., & Qu, Q. (2023). Understanding deep representation learning via layerwise feature compression and discrimination. arXiv preprint arXiv:2311.02960.

Xiao, Z., Wan, M., Chen, S., & Eyink, G. (2009). Physical mechanism of the inverse energy cascade of two-dimensional turbulence: A numerical investigation. *Journal of Fluid Mechanics*, *619*, 1–44. https://doi.org/10.1017/s0022112008004266

Yuval, J., & O'Gorman, P. A. (2020). Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature Communications*, *11*(1), 1–10. https://doi.org/10.1038/s41467-020-17142-3

Yuval, J., O'Gorman, P. A., & Hill, C. N. (2021). Use of neural networks for stable, accurate and physically consistent parameterization of subgrid atmospheric processes with good performance at reduced precision. *Geophysical Research Letters*, *48*(6), e2020GL091363. https://doi.org/10.1029/2020gl091363

Zanna, L., & Bolton, T. (2020). Data-driven equation discovery of ocean mesoscale closures. *Geophysical Research Letters*, *47*(17), e2020GL088376. https://doi.org/10.1029/2020gl088376

Zhang, C., Perezhogin, P., Gultekin, C., Adcroft, A., Fernandez-Granda, C., & Zanna, L. (2023). Implementation and evaluation of a machine learned mesoscale eddy parameterization into a numerical ocean circulation model. *Journal of Advances in Modeling Earth Systems*, *15*(10), e2023MS003697. https://doi.org/10.1029/2023ms003697

Zhou, D.-X. (2020). Universality of deep convolutional neural networks. *Applied and Computational Harmonic Analysis*, *48*(2), 787–794. https://doi.org/10.1016/j.acha.2019.06.004

Zhou, Z., He, G., Wang, S., & Jin, G. (2019). Subgrid-scale model for large-eddy simulation of isotropic turbulent flows using an artificial neural network. *Computers & Fluids*, *195*, 104319. https://doi.org/10.1016/j.compfluid.2019.104319