Deep Learning Turbulence Closures Generalize Best With Physics-based Methods

Alex Connolly[1], Yu Cheng[2], Robin Walters[3], Rui Wang[4], Rose Yu[4], and Pierre Gentine[1]

[1]Earth and Environmental Engineering, Columbia University, New York, NY 10025, USA
[2]Department of Earth and Planetary Sciences, Harvard University, Cambridge, MA, 02138, USA
[3]Khoury College of Computer Sciences, Northeastern University, Boston, MA, 02115, USA
[4]Computer Science and Engineering, University of California San Diego, La Jolla, 92093, USA

Key Points:

- Deep learning models of subfilter-scale stress improve resolved turbulence and near-surface jets in the stable atmospheric boundary layer.
- Deep neural networks generalize to unseen scenarios through physical scaling relationships replacing statistics-based normalization.
- Physical constraints for horizontal isotropy, vertical anisotropy best a data augmentation approach to providing geometrical information.

Corresponding author: Alex Connolly, ac5006@columbia.edu

Abstract

Boundary layer turbulence necessitates parameterization in Earth system models of the subfilter-scale stress due to the unresolved turbulent flux of momentum. Here, a deep neural network (DNN) serves as this parameterization, ingesting resolved variables and predicting the subfilter-scale stress for large-eddy simulation (LES) of the stably stratified atmospheric boundary layer (ABL). The DNN models are trained on high resolution direct numerical simulation data coarse-grained to LES resolutions. The ability of a DNN model to generalize is tested in scenarios which differ from training data by Reynolds number, grid resolution, and orientation relative to the forcing winds. Physics-based nondimensionalization for greater generality is common in fluid mechanics but data normalization for training a DNN can be done without consideration of the physics, instead using the statistics of the training data. We test this statistical scaling approach against two physical scaling approaches. To address generalizability to the flow orientation, a common practice is augmenting training data with rotated samples. We compare this to a model architecture which embeds the symmetry of the physical system, horizontally isotropic but vertically anisotropic. When implemented in the microHH code for LES of ABL flow, a DNN using physical scaling leads to LES which are stable and outperform those using a conventional closure. Simulations using statistical factors for normalization often fail, even when the same DNN models succeeded in similar offline tests. On this basis, we recommend designing deep learning parameterizations based on physics-informed nondimensionalization and model architectures constrained to the geometry of the physical system.

Plain Language Summary

Turbulence in the atmospheric boundary layer, including the gusty winds we feel at the Earth's surface, must be accounted for by weather and climate forecasts. We use new machine learning techniques to account for these effects. Our neural networks learn from high resolution data, and are added to coarser resolution simulations of the atmosphere. With our new methods which inform these machine learning models of relevant physics, we improve the predictions made by these simulations. This is true even for weather that is different from that included in the data used to train the neural network. For instance, if the winds are stronger or coming from a different direction, then our deep learning model knows how to make the necessary adjustments to account for these differences.

1 Introduction

Turbulence in the atmosphere and ocean significantly impacts the evolution of weather and climate (Wyngaard, 2010; Thorpe, 2004). For studies of turbulent flow in the stably stratified atmospheric boundary, large-eddy simulation (LES) is a common technique (Beare et al., 2006; Lu & Porté-Agel, 2011; B. Zhou & Chow, 2011; Connolly et al., 2021; Chinita et al., 2022). Though the grid meshes used in LESs are finer than those of operational weather and climate models, they are too coarse to resolve the finest scales of these turbulent flows. Due to the limits of resolution, models may employ a subfilter parameterization to account for the aggregate effect of turbulent motions on the resolved fields. Such turbulence closures seek to approximate the effects of turbulence as a function of the resolved state variables. These parameterization are necessarily empirical, and a number of functional forms have been proposed (Lilly, 1962; Smagorinsky, 1963; Clark et al., 1979; Deardorff, 1980; Bardina et al., 1980; Germano et al., 1991; Chow et al., 2005; Simon & Chow, 2021).

With the advance of machine learning, it is now possible to avoid specifying a functional form for turbulence closure, and instead let these functions be learned directly from data (Z. Zhou et al., 2019; Stoffer et al., 2021; Cheng et al., 2022). Though such approaches, particularly deep learning methods, have undoubtedly shown promise for improving tur-

bulence modeling, they are not without their own challenges. A primary challenge for machine learning applied to physical problems is generalizability, *i.e.* the ability for a machine learning model to accurately make predictions in regimes that are substantially different from those for which it was trained (Zhang et al., 2021).

In part to address this generalizability issue, enforcing physical constraints in machine learning models is receiving significant attention in turbulence and Earth system modeling (Ling et al., 2016; Wang et al., 2020; Beucler et al., 2020, 2021; Willard et al., 2022). Such constraints fall into a number of categories including conservation laws, self-similarity relationships, and symmetry. Incorporating these physical constraints has been shown to reduce the amount of training data required for accurate simulations of two-dimensional turbulence (Guan et al., 2023) and to allow for more accurate prediction of turbulent scalar fluxes in regimes outside those of the training data (Frezat et al., 2021). The current work shows that enforcing physical constraints in deep learning models improves turbulent momentum flux predictions for three-dimensional turbulence in the stably stratified atmospheric boundary layer.

Previous work has applied a deep neural network (DNN) to predict turbulent fluxes of momentum for LES of the atmospheric boundary layer, but did not include physical constraints and was designed for near neutral to unstably stratified conditions (Cheng et al., 2022). Other work on momentum closures for LES focused on neutral channel flow and did not incorporate physical constraints or attempt to test the generalization of their neural network (Stoffer et al., 2021). A related body of work on machine learning applications to turbulence closures that were not designed for LES but for Reynolds-Averaged Navier-Stokes (RANS) simulation (Ling et al., 2016; Kaandorp & Dwight, 2020) did include physical invariance constraints, but their approach is less appropriate in the context of density stratification because it does not allow for vertical anisotropy. Other related work applies deep learning to the prediction of turbulent flow fields, rather than to the turbulence closure. Though some of this work incorporates physical constraints leading to improved generalizability (Wang et al., 2020), the models are specific to domains for which they were trained within relatively idealized fluid solvers so cannot be readily implemented in legacy codes with full atmospheric physics.

Our proposed approach includes a number of physical constraints through various means in the design choices regarding which variable to target and the deep learning architecture. First, we target only turbulent stresses. Other design choices could be to target the divergence of these stress, which ultimately drive the time evolution of turbulent velocity fields, or the time evolution itself. These other design choices would not have guaranteed conservation of momentum and mass, so we prefer this approach in predicting the stresses. Beyond this, our deep learning approach is designed to enforce symmetry to rotations in the horizontal plane, which the governing equations obey, invariance to translations in all 3 spatial dimensions, and Galilean invariance.

In section 2, we provide a brief overview on large-eddy simulation, deep learning, symmetry, and a method for incorporating symmetry in deep learning models. Information on the high-resolution direct numerical simulations (DNSs) and our method for coarse-graining these to LES resolutions are provided in section 3. Also in this methods section are details on our DNNs, with special attention to how symmetry is implemented and the various approaches to scaling the inputs and outputs to the deep learning model. Finally, we detail the LES configurations conducted in *a posteriori* tests of our DNN turbulence closure models. Results of offline tests presented in section 4 include the impact of different scaling on the generalizability of the deep learning model, comparison of the model architecture including symmetry constraints to the conventional data augmentation approach, the effect of buoyancy as an input to the model, and the consequences of enforcing rotational symmetry for angles finer than the grid can precisely represent. The *a posteriori* tests in section 5 revisit the impact of different scaling relationships on the generalizability of the deep learning model in an online setting. This allows a com-

parison of offline and online testing for the same deep learning models. We conclude that deep learning models utilizing physical scaling can successfully generalize for accurate and numerically stable simulations of scenarios different than those on which they were trained.

2 Background

2.1 Large-eddy simulation

The dynamical cores of atmospheric models are based on discretized versions of the governing equations for relevant physical variables: momentum, mass, energy, moisture, trace species, and so on. The effect of the grid resolution is often assumed similar to applying a spatial filter to the governing equations (Clark et al., 1979; Pope, 2000; Wyngaard, 2004). If the grid spacing is coarse enough, the associated filtering is considered equivalent to ensemble averaging and the governing equations describing momentum and mass conservation are the familiar Reynolds Averaged Navier-Stokes (RANS) equations (Reynolds, 1895), for which it is assumed no turbulence is resolved. At finer resolutions, this assumption is relaxed and the governing equations are the large-eddy simulation (LES) equations (Lilly, 1962; Smagorinsky, 1963; Deardorff, 1980), for which it is assumed that the largest, most energetic turbulent motions are resolved.

We consider three-dimensional flow in Cartesian coordinates, $x_i = [x_1, x_2, x_3] = [x, y, z]$ with x_1, the zonal direction; x_2; the meridional; and x_3 antiparallel to the force of gravity. Corresponding components of velocity, $u_i = [u_1, u_2, u_3] = [u, v, w]$, co-evolve with pressure, P; density, ρ; buoyancy, b; and subfilter-scale stress, τ, and are influenced by gravitational acceleration, g, and the Coriolis frequency, f_C. Using an overline, $\overline{\cdot}$, to represent a quantity solved on the computational grid, *i.e.* a resolved variable, the LES equations expressing conservation of resolved momentum can be written with a Boussinesq approximation as

$$\frac{\partial \overline{u_i}}{\partial t} + \overline{u_j}\frac{\partial \overline{u_i}}{\partial x_j} = -\frac{1}{\overline{\rho}}\frac{\partial \overline{P}}{\partial x_i} - \delta_{i3}(g - b) + \epsilon_{3ij} f_C \overline{u_j} - \frac{\partial \tau_{ij}}{\partial x_j} + \cdots \tag{1}$$

where the terms, from left to right, are called unsteadiness, resolved advection, resolved pressure gradient force, reduced gravity, Coriolis force, and subfilter-scale stress gradient force.

The system of governing equations is closed if the number of equations equals the number of state variables. The three components of velocity are matched by eqns. 1, the three evolution equations for velocity which express conservation of momentum. Gravitational acceleration is a known function of altitude and the Coriolis frequency is a known function of latitude. The three additional variables introduced in eqn. 1, $\overline{\rho}$, $\overline{P}$, and $\overline{b}$, are constrained by three additional equations: conservation of mass, the equation of state, and conservation of internal energy. The energy equation is often formulated in terms of another variable, potential temperature, $\overline{\theta}$, or virtual potential temperature in some approaches to moist variables not considered here. When selecting a reference temperature, θ_0, approximating the temperature of the environment, potential temperature can be expressed as an approximate vertical acceleration,

$$\overline{b} = \frac{g}{\theta_0}(\overline{\theta} - \theta_0), \tag{2}$$

referred to as buoyancy, which enters eqns. 1 in the reduced gravity term. To close the system of LES equations, we require additional equations to specify the subfilter-scale stress,

$$\tau_{ij} = \overline{u_i u_j} - \overline{u_i}\,\overline{u_j}. \tag{3}$$

It is worth noting that the Navier-Stokes equations can be manipulated to form explicit partial differential equations which govern these stresses, but these equations in-

troduce additional variables such that the total number of variables would still exceed the number of independent equations. Although it is possible to close these equations instead (Mellor & Yamada, 1974; Bogenschutz et al., 2013), we do not consider these approaches to the closure problem in this work. Rather, we seek to express the subfilter-scale stress directly in terms of the resolved, grid-scale variables already introduced. This is also a common approach that is sufficient to close the system of governing equations without introducing additional prognosticated variables and equations requiring additional parameterizations.

Eddy-diffusivity models are a common class of turbulence closures based on analogies to Fickian diffusion and molecular viscosity formulated by introducing the parameter, K_{ij}, referred to as the eddy diffusivity or eddy viscosity. A common form is

$$\left(\tau_{ij} - \frac{1}{3}\delta_{ij}\tau_{kk}\right)_{\text{Smagorinsky}} = -K_{ij} \odot \left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right) = -K_{ij} \odot D_{ij}. \tag{4}$$

where a third of the trace of the subfilter-scale stress tensor has historically been subtracted such that the eddy-diffusivity models only attempt to model the deviatoric stress. This ensures that for incompressible flow, with $\nabla \cdot \mathbf{u} = 0$, the contraction of both sides of eqn. 4 above is precisely zero. The trace, which is twice the turbulent kinetic energy, is often prognosticated and incorporated into a modified pressure so the formulation is physically consistent (Deardorff, 1970). For turbulent environmental fluid mechanics, additional physics are usually utilized to specify a variable eddy diffusivity. These formulations often introduce anisotropy, so we have represented K_{ij} as a tensor for generality. It is strictly required by incompressibility that K_{ij} be symmetric, because τ_{ij} is, though this requirement has been violated in previous implementations (see the discussion in Simon and Chow (2021) for more details).

For LES of the stable atmospheric boundary layer, a common model for the eddy viscosity is the Smagorinsky-Lilly model (Lilly, 1962), or simply Smagorinsky model for brevity. Smagorinsky is often credited with the insight of taking the grid spacing, when in the inertial subrange, as a turbulent length scale (Smagorinsky, 1963). Lilly (1962) suggested using the buoyancy or Brunt-Väissälä frequency,

$$N = \sqrt{\frac{g}{\theta_0}\frac{\partial \overline{\theta}}{\partial z}} = \sqrt{\frac{\partial \overline{b}}{\partial z}}, \tag{5}$$

which is a common measure of density stratification, in the calculation of eddy diffusivity. The form which we consider here is,

$$K_{ij} = \left(c_s \Delta_{ij}\right)^2 \max\left[0, \left(|\mathcal{D}|^2 - Pr^{-1}N^2\right)^{0.5}\right] \tag{6}$$

where the Smagorinsky constant, c_s and the Prandtl number, Pr, assume commonly used constant values of 0.25 and 1/3, respectively and the length scale, Δ_{ij}, is based on the grid spacing. This length scale has isotropic, $\Delta = (\Delta x \Delta y \Delta z)^{1/3}$, as well as various anisotropic formulations (Simon & Chow, 2021). In *a priori* comparisons to DNS data, we present results from a slightly better performing anisotropic formulation with $\Delta_{ii}(\text{no sum}) = \Delta_{12} = (\Delta x \Delta y)^{1/2}$ and $\Delta_{13} = \Delta_{23} = \Delta z$ in offline tests. The Smagorinsky model in the microHH code is implemented with an isotropic grid length scale damped by the wall (Mason & Thomson, 1992), so some LES use this formulation during *a posteriori* evaluations. Upon tuning these LES, a higher $Pr = 10$ is used in the Smagorinsky model for offline tests.

Another class of turbulence closures, scale similarity models, rely on resolved turbulence and uses the self-similarity of turbulence to define the closure. A common model of this sort is the Bardina model,

$$\tau_{ij,\text{Bardina}} = \alpha_B \left(\overline{\overline{u_i}\,\overline{u_j}} - \overline{\overline{u_i}}\,\overline{\overline{u_j}}\right). \tag{7}$$

the constant, α_B, is typically and here taken to be unity, in which case the expression is derived by direct substitution of the resolved velocity, $\overline{u_i}$, for the unknown velocity, u_i, in the definition of subfilter-scale stress. Other values of α_B are in the range of 0.9 to 1.1. The underlying assumption is that the variations of the resolved velocity will be similar to those of the unresovled velocity given the scale similarity of turbulence. For $\alpha_B \neq 1$, there is some tuned correction for slight differences at scales greater than the filter width compared to those less than the filter width.

Another popular similarity model is the velocity gradient model or the Clark model in reference to its use in Clark et al. (1979). Through Taylor expansion, we obtain an exact expression for the subfilter-scale stress in terms of the resolved velocities as an infinite series. Bedford and Yeo (1993) derived this series as

$$\overline{u_i u_j} - \overline{u_i}\,\overline{u_j} = 2\alpha \frac{\partial \overline{u_i}}{\partial x_k}\frac{\partial \overline{u_j}}{\partial x_k} + \frac{(2\alpha)^2}{2!}\frac{\partial^2 \overline{u_i}}{\partial x_k^2}\frac{\partial^2 \overline{u_j}}{\partial x_k^2} + \frac{(2\alpha)^3}{3!}\frac{\partial^3 \overline{u_i}}{\partial x_k^2}\frac{\partial^3 \overline{u_j}}{\partial x_k^2} + ... \tag{8}$$

where the factor, α, depends on the filter type and is $\alpha = \Delta^2/24$ for a box filter of width Δ on all sides. Truncating this series to retain only the first order term recovers the original Clark model. We consider an anisotropic formulation,

$$\tau_{ij,\text{Clark}} = \sum_k 2\frac{(4\Delta x_k)^2}{24}\frac{\partial \overline{u_i}}{\partial x_k}\frac{\partial \overline{u_j}}{\partial x_k}. \tag{9}$$

In the current work, we use these existing closures, Smagorinsky, Bardina, and Clark, to compare to models developed through a deep learning approach. Specifically, we develop deep neural networks which take resolved variables as inputs and output the subfilter-scale stress. Such a deep neural network can be used in place of existing momentum closures in LESs. General background on deep neural networks is given in the next section, while details of the specific model architecture are include in the methods section.

2.2 Deep Neural Networks

In the previous section, we presented several conventional turbulence closures whose functional forms varied widely. An alternative approach is to use deep learning, which does not require any specific prescription of a functional form. Indeed, deep neural networks can approximate any function between input variables and their associated outputs (Hornik et al., 1989).

A neural network works through successive application of linear combination of features and nonlinear activation functions. A single layer, feed forward network with N inputs, x_q, and vector valued output, z_p, is defined by

$$z_p = f\left(\sum_{q=0}^{N} w_{p,q} x_q\right) \tag{10}$$

for either $x_0 = 1$ such that w_0 is the bias term or $x_0 = 0$ for no bias, and f a nonlinear activation function.

A deep neural network (DNN) stacks multiple layers like those above and z_p are the 'neurons' of the intermediary, hidden layers. If there are L layers and the lth layer has N_l neurons with the same activation function applied for all neurons at all layers except the final layer for which no non-linearity is applied, the output, y_k^{NN}, is computed as

$$y_k^{NN} = \sum_0^{N_L} w_{l,m}^L f\left(\sum_0^{N_{L-1}} w_{m,n}^{L-1} f\left(...\sum_0^{N_1} w_{p,q}^1 x_q\right)...\right) \tag{11}$$

During the training of a DNN, the weights, $w^{\bullet}_{\_,\star}$, are updated based on the gradients with respect to these weights of an objective function that measures the error of the network's predictions, commonly with the mean squared error, $a = 2$ below, or mean absolute error, a=1. These objective functions are defined as

$$\mathcal{J} = \frac{1}{S}\sum_{s=1}^{S}\left(\sum_{k}\omega_k|y_k - y_k^{NN}|^a\right)_s \tag{12}$$

where y_k is a vector of ground truth labeled data which are multiplied by specific loss weights, ω_k, often to balance the contributions from different output variables to the loss function, taken from S samples of training data. Thanks to the development of automatic differentiation, new optimization techniques, and the explosion in data availability, DNNs are increasingly deployed with great success in a wide range of applications (LeCun et al., 2015).

2.3 Symmetry for generalization

Despite the successes of DNNs in numerous applications, generalization outside of the distribution of training data, i.e. out-of-distribution prediction, remains challenging. Problems of generalization occur in fluid dynamics as well, and the most common strategy to deal with such issue relies on the definition of dimensionless numbers such as the Reynolds number. This approach to generalizability is discussed in more detail in section 3.4. Now, we first turn our attention to another topic related to generalization considering the symmetries of the physical system.

A principle of classical mechanics is that the choice of coordinate system should not affect the underlying physical laws. In other words, a differential equation is invariant with respect to a symmetry group when its transformed solution remains a solution. The governing equations of fluid mechanics, the Navier-Stokes as well as the RANS and LES equations (eqn. 1), are invariant to a number of transformations: space translation, time translation, uniform motion (with appropriate modification to the Coriolis pseudoforce), scaling, and rotation, at least in the horizontal plane given the presence of gravity and, in our case, vertical density stratification. Typical DNNs do not conform to these invariances, so we develop new DNN models that are constrained by these symmetries.

Each of these physical symmetry laws may be mathematically formalized as symmetry group, G. In this formalism, a G-invariant function, f, is defined as

$$f(\rho_{\text{in}}(g)x^*) = f(x^*) \tag{13}$$

for all $g \in G$, a transformation with representation, $\rho_{\text{in}}(g)$, acting on x^*, the inputs of the function. Note, juxtaposition is not to be interpreted as scalar or matrix multiplication but as a more general action. A broader notion of symmetry is that of equivariance. A G-equivariant function is defined by

$$f(\rho_{\text{in}}(g)x^*) = \rho_{\text{out}}(g)f(x^*), \tag{14}$$

where the actions ρ_{in} and ρ_{out} may differ because they act on the input and output spaces of function f, respectively. These spaces are in general different, so the inputs and outputs may be associated with different representations of the same transformation.

Informally, we understand that transforming the inputs to an equivariant function should produce an equivalent transformation of the outputs. In the case of a fluid flow, the resolved velocity vector under a linear change of coordinates, $\mathcal{R}$, such as rotation, transform by the rule

$$\overline{\mathbf{u}}^r = \mathcal{R}\overline{\mathbf{u}}. \tag{15}$$

In this example, the action of ρ_{in} is left multiplication by R, a matrix. The subfilter-scale stress, a rank-2 tensor, transforms slightly differently as

$$\boldsymbol{\tau}^r = \mathcal{R}\boldsymbol{\tau}\mathcal{R}^{-1}. \tag{16}$$

In this case, the action of ρ_{out} is both left multiplication by $\mathcal{R}$ and right multiplication by $\mathcal{R}^{-1}$.

We consider equivariance to rotations in the horizontal (x_1−x_2) plane. We do not seek to enforce equivariance to rotations in three-dimensions because we are targeting the stable boundary layer with anisotropy in the vertical due to buoyancy stratification. If this anisotropy breaks the symmetry only slightly, then recently developed methods for approximate equivariance (Wang et al., 2022a) could be deployed as an interesting extension that we suggest for future work. For continuous data, the group corresponding to rotation in two-dimensions is SO(2), the special orthogonal group. In practice, continuous rotations cannot be expressed by the gridded simulation data considered here. Thus, we instead consider the cyclic group, C_N, with associated transformation matrix,

$$\mathcal{R} = \begin{bmatrix} \cos\left(p\frac{2\pi}{N}\right) & -\sin\left(p\frac{2\pi}{N}\right) & 0 \\ \sin\left(p\frac{2\pi}{N}\right) & \cos\left(p\frac{2\pi}{N}\right) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{17}$$

for $p \in \{0, 1, ..., N-1\}$, which can be considered a discrete approximation of the SO(2) group.

Requiring our model for subfilter-scale stress to respect these symmetries is equivalent to generalizing across flow orientations. This is important when training a DNN on data from idealized simulations, which are unlikely to represent all possible flow orientations. Indeed, the high resolution simulations used in this work, described in section 3.1, all align the forcing winds aloft with the x_1 direction. This is a natural choice for idealized simulation, but cannot be expected for more realistic simulations. For that reason, it is appropriate to require any turbulence closure to be general enough to handle arbitrary flow orientations.

2.4 Equivariant steerable convolution neural networks

The subfilter-scale stress, in unclosed form or modeled through the conventional parameterizations defined in eqns. 4 – 9, are equivariant to the groups associated with the same symmetries as the governing equations. Enforcing a deep neural network function to be equivariant to a symmetry group typically requires additional constraints on the model architecture. A notable example is a convolutional neural network (CNN), which enforces approximate translational equivariance through convolutional layers,

$$z_p(i', j') = f\left(\sum_q \sum_a \sum_b K_{p,q}(a, b) x_q(i - a, j - b)\right). \tag{18}$$

which is a special case of eqn. 10 which defines a general layer. This form is often used when inputs and outputs are images, or 2-dimensional arrays more generally. The pth channel of the output, $z_p(i', j')$, has spatial indices, i' and j', whose extents are not necessarily the same as the input. The image sizes may differ due to details in the range of indices and padding of boundaries, or as a consequence of pooling operations which are common choices for the activation function in image processing. Importantly, the weights, $w_{p,q}$ in eqn. 10, are now constrained to be shared across the spatial dimensions as $K_{p,q}(a, b)$, a kernel convolving x_q, the 2-dimensional inputs. Sharing weights in this way makes the convolutional layer equivariant to translations.

Theory and practice for other symmetries have been developed by T. Cohen and Welling (2016); Weiler and Cesa (2019a); T. S. Cohen et al. (2019), and other works which refer to the method as equivariant steerable CNNs (T. S. Cohen & Welling, 2016; Weiler & Cesa, 2019b). An implementation is available in the e2cnn python library. In this implementation, the convolutional kernel is constrained by

$$K(g\mathbf{x}) = \rho_{\text{out}}^{-1}(g) K(\mathbf{x}) \rho_{\text{in}}(g) \quad \forall\, g \in G \tag{19}$$

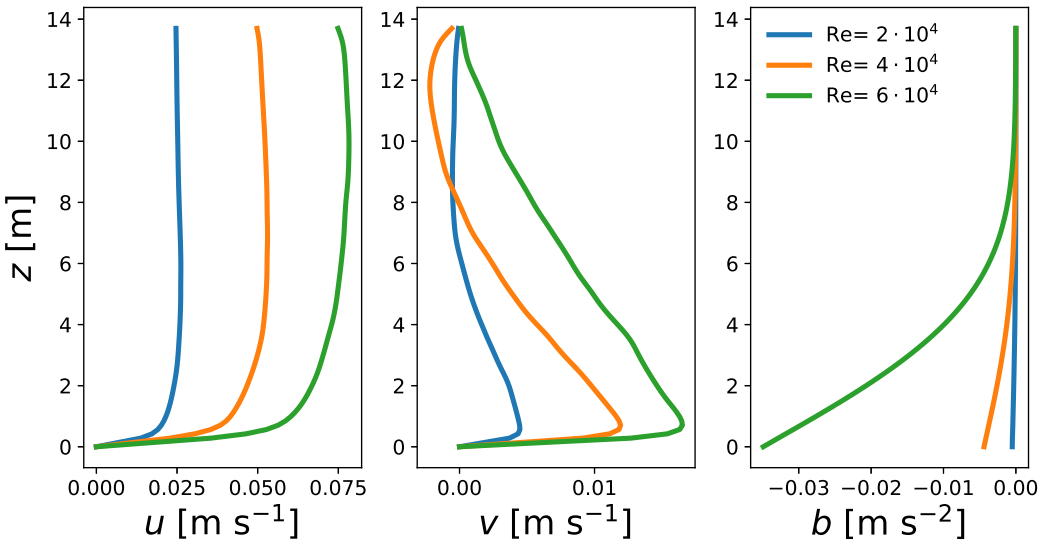

Figure 1. Profiles of zonal velocity, u (left), meridional velocity, v (middle), and buoyancy, b (right), for each of the 3 DNS, averaged in the horizontal and time.

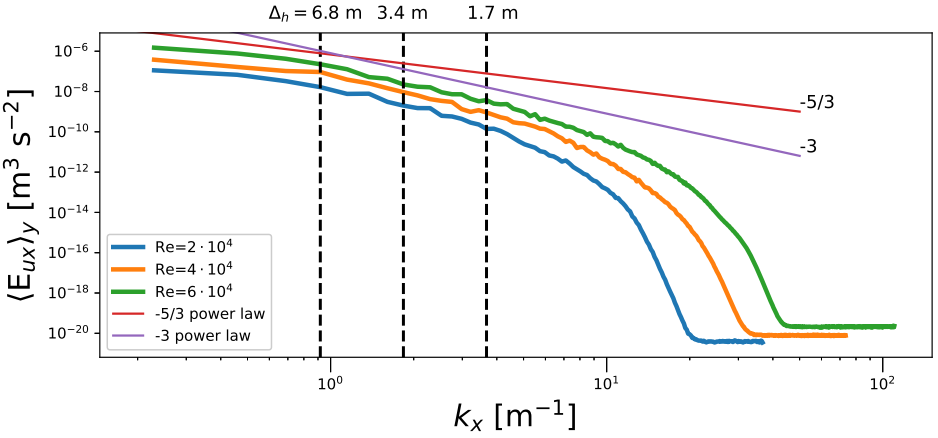

Figure 2. Power spectra of u-velocity as a function of x-direction wavenumber and averaged in the y-direction from each DNS. Dashed lines indicate the wavenumbers which correspond to the horizontal widths of the filter, $\Delta_h = 4\Delta x$, for the three LES grid resolutions considered.

where G, is a user specified subgroup of $\mathbb{E}(2)$, the Euclidean group, and g is a transformation given as a standard representation acting on $\mathbf{x}$, the spatial coordinates, which replace the spatial indices used previously for brevity. As in the general CNN layer, weight sharing across the 2 spatial axes results from such kernel constraints. Unlike typical CNN, in this implementation for an equivariant steerable CNN, the input, output and hidden layers must be defined by specifying their irreducible representations. Sections 3.3.1 and 3.3.2 detail the specific irreducible representations used in this work.

Re	U_g [cm s^{-1}]	$\Delta x = \Delta y$	Δz	Q [K m s^{-1}]	Ri_b
$2 \cdot 10^4$	2.5	$0.1562D$	$0.0198D$	4.94e-8	7.5
$4 \cdot 10^4$	5.0	$0.0781D$	$0.0099D$	4.34e-7	15
$6 \cdot 10^4$	7.5	$0.0521D$	$0.0066D$	3.44e-6	53

Table 1. Description of the DNS configurations. Length scales are given in terms of $D = \sqrt{2\nu/f_C} = 0.55$ m, the laminar Ekman layer depth.

3 Methods

3.1 Direct numerical simulations

Three separate direct numerical simulations (DNSs) of the stably stratified atmospheric boundary layer are used to derive filtered velocity and buoyancy as well as the subfilter-scale stress. The DNSs are computed with the Boussinesq approximations of the incompressible Navier-Stokes equations, using the microHH code described in Heerwaarden et al. (2017). The prognostic variables are velocity and buoyancy, which are coupled. An imposed cooling rate is applied at the bottom surface as the buoyancy boundary condition to develop a stable boundary layer. No slip, no flow conditions are enforced for velocity at the bottom boundary. A sponge layer is applied to the top 25% of the domain to prevent gravity wave reflections (Nieuwstadt et al., 1993). Further details of the simulation setup can be found in Cheng et al. (2023).

Specifics on the current DNS configurations are given in table 1. Each of the three simulations corresponds to a different Reynolds number,

$$Re = \frac{U_g \, z_i}{\nu}, \tag{20}$$

where U_g refers to the geostrophic wind, $z_i = 12$ m is the boundary layer depth, $\nu = 1.5{\times}10^{-5}$ m^2 s^{-1} is the kinematic viscosity of air. We select the boundary layer height as characteristic length scale in the Reynolds number because it is relevant at both DNS and large-eddy simulation (LES) scales. For the DNS, another relevant length scale is $D = \sqrt{2\nu/f_C} = 0.55$ m, the laminar Ekman layer height for $f_C = 10^{-4}$ rad/s. This length scale is the same for all simulations as are the extents of the domain, $50D$ in the horizontal directions and $33D$ in the vertical. The simulations vary the magnitude of the imposed pressure gradient, whose effect is manifested in the geostrophic wind speed, U_g, shown at the top of the domain in Fig. 1, left panel. Covarying with Re is the the imposed cooling rates, Q, with larger cooling rates leading to stronger near-surface buoyancy gradients shown in Fig. 1, right panel. The gradients from 10 m to the surface can be characterized by a bulk Richardson number, which has values of 7.5, 15, and 53 for $Re = 2{\cdot}10^4$, $Re = 4{\cdot}10^4$, and $Re = 6{\cdot}10^4$, respectively. Because Richardson number is here a covariate of Reynolds number, we distinguish the DNSs by specifying only Reynolds number. Higher Reynolds number simulations necessitate finer grid mesh resolutions, so the number of grid points differs between simulations though the physical domain size is unchanged. Consequently, 44 timeframes were saved from the $Re = 2{\cdot}10^4$ DNS, while 15 timeframes were saved for $Re = 4 \cdot 10^4$ and only 3 timeframes were saved from the $Re = 6 \cdot 10^4$ simulation.

To ensure that each DNS is adequately resolved, we check the power spectra shown in figure 2. These indicate that all the DNSs are sufficiently resolved by the presence of a robust dissipative range where energy rapidly decreases with increasing wavenumber. The spectra are computed for the dominant along-stream component of velocity, u in the along-stream direction, x, and averaged in the cross-stream direction, y. Specifically, the discrete spectral density, E_{ux}, is defined as

$$E_{ux,m} = \frac{\Delta x |\hat{u}_m|^2}{2\pi N} \tag{21}$$

where $\hat{u}$ is the discrete Fourier transform of u following the normalization conventions (of numpy's fft or rfft routine) in which the forward transform is not normalized such that

$$\hat{u}_m = \sum_{j=1}^{N} u_j \exp\left[\frac{-2\pi i}{N}(m-1)(j-1)\right]. \tag{22}$$

3.2 Coarse-graining

An assumption underlying *a priori* testing of turbulence closures is that the effect of a coarse grid can be approximated by an analytic filter applied to high resolution observation or simulation data (Clark et al., 1979; Vreman et al., 1995). The limitations of such an assumption may manifest when a subgrid model is implemented in a natively coarse simulation. We successfully perform such *a posteriori* tests with models trained from coarse-grained data, as documented later (section 5). As such, we did not thoroughly address the appropriateness of the chosen analytic filter, but future work could focus on the impact of different filtering approaches in the training of deep learning turbulence closures.

Filtered quantities, including the subfilter-scale stress from eqn. 3, can be derived from the DNS data once we specify a filter to approximate the effect of a coarser grid. We choose a top-hat box filter which has widths equal to 4 times the coarsened grid spacing. That the filter width should be larger than the grid spacing is established by previous work (Clark et al., 1979; Vreman et al., 1995; Chow & Moin, 2003; Skamarock, 2004; Chow et al., 2005). The specific choice in filter to grid ratio (FGR) of 4 is within the range of this existing literature. For instance, Chow et al. (2005) discuss the appropriateness of FGR of both 2 and 4, in the context of explicit filtering for LES. Skamarock (2004) discussed a concept related to FGR, termed effective resolution. Rather than prescriptive suggestion for FGR, effective resolution is discussed in a more descriptive manner. Defined through comparison of kinetic energy spectra from simulation to observation, effective resolution describes what scale of features are actually resolved. Skamarock (2004) found, for a specific numerical weather prediction code and numerical scheme, the effective resolution was 7 times the grid spacing at the mesoscale and suggested that this factor would be smaller at finer resolutions used in LES.

Proceeding with the choice of a box filter with FGR equal to 4, we define a filtered quantity as,

$$\overline{\xi}(i\Delta x, j\Delta y, k\Delta z) = \frac{1}{4\Delta x}\int_{x=(i-2)\Delta x}^{(i+2)\Delta x}\frac{1}{4\Delta y}\int_{y=(j-2)\Delta y}^{(j+2)\Delta y}\frac{1}{4\Delta z}\int_{z=(k-2)\Delta z}^{(k+2)\Delta z}\xi(x,y,z)dxdydz \tag{23}$$

where $\xi(x,y,z)$ is either the DNS variables or products of these variables. With the simplest two-point averaging, we first destagger the DNS data so the products, u_1u_3 for example, can be taken at collocated points. The integrals above are computed on the collocated DNS grid using the midpoint rule for Riemann sums. Periodic boundary conditions are used to pad the x and y directions. We pad the vertical velocity with the no flow, $w = 0$, surface boundary condition for destaggering, but we are still restricted to $z \geq 2\Delta z$ due to filtering near the solid boundary. This combination of filtering and downsampling is what we refer to as coarse-graining.

We select three different coarse resolution grids on which we down-sample the filtered DNS data. Choice in coarse grid and the varying number of timeframes saved from DNSs of different Reynolds number both affect the number of space-time locations for

Total Possible Samples	LES Grid Resolution $\Delta x = \Delta y = 0.43$ m $\Delta z = 0.14$ m	$\Delta x = \Delta y = 0.86$ m $\Delta z = 0.28$ m	$\Delta x = \Delta y = 1.7$ m $\Delta z = 0.57$ m
Re			
$2 \cdot 10^4$	16,941,056	2,072,576	247,808
$4 \cdot 10^4$	5,775,360	706,560	84,480
$6 \cdot 10^4$	1,155,072	141,312	16,896

Table 2. Total possible number of training or testing samples per dataset. Each dataset corresponds to a DNS, identified by its Reynolds number, from which the fields are derived and resolution to which they are coarse-grained.

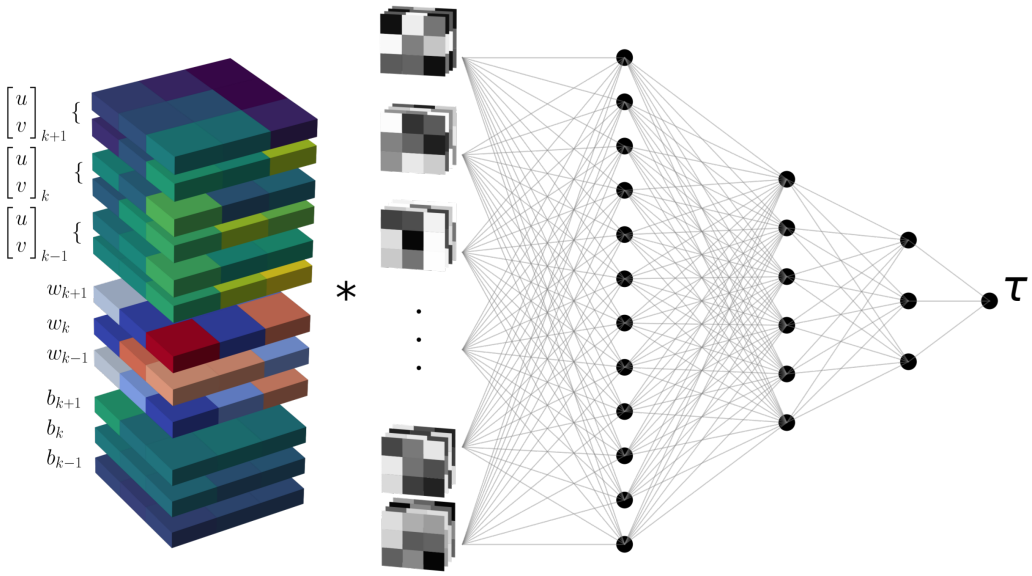

Figure 3. Visualization of the C_4-equivariant DNN architecture of the deep learning turbulence closure models.

which subfilter-scale stress can be calculated. Consequently, the total number of possible labeled samples vary substantially across the combinations of DNS and coarse grid. This number of samples, ns if given by,

$$ns = nt \times nx \times ny \times (0.75nz - 2), \tag{24}$$

a function of number of DNS timeframes saved, nt, and the number of coarse grid points in the x, y, and z directions, nx, ny, and nz, which reflects the presence of a sponge layer and the inability to predict subfilter-scale stress at the top and bottom levels given the input stencil detailed in the next section. In table 2, these calculations have been carried out for the coarse-grained datasets used in this work.

3.3 Deep neural network architecture

Illustrated in figure 3, our deep neural network (DNN) takes as input 4 variables, the coarsened 3-dimensional velocities and the coarsened buoyancy within a local box, and outputs the subfilter-scale stress at the center of the box. The input box is a $3 \times 3 \times 3$ box of grid cells except for one numerical experiment testing the sensitivity to the

input size, in which we use a $5 \times 5 \times 3$ box. Other than this case, the dimensionality of the input feature array is $3^3 \times 4 = 108$. The outputs are the 6 unique components of the subfilter-scale stress tensor, τ_{ij}, which is a symmetric tensor by definition.

This local stencil approach is similar to that of Stoffer et al. (2021) and Cheng et al. (2022) and there are several reasons for using a similar approach in the current work. First, as a practical matter, a local input stencil is easier to implement in a fluid solver. These codes are often parallelized through domain decomposition such that the processor responsible for computing subfilter-scale stress at one grid cell will not have access to input values from other parts of the domain farther away. Secondly, in the context of stable stratification, turbulent motions are more localized because there is no convection. Finally, in large-eddy simulations, the influence of the large scale forcing on the subgrid turbulence should be mediated by resolved turbulence at the scales of the grid resolution.

Our C_N-equivariant DNNs are implemented as 2-dimensional convolutional network using the e2cnn python library (Weiler & Cesa, 2019a). Since C_N represents rotations only in a plane, we reshape the input into a $3{\times}3{\times}12$ tensor with the 12 channels corresponding to the 4 variables on 3 vertical levels labeled $k{-}1, k, k{+}1$ in figure 3. We use convolutional kernel sizes equal to the 'image' size, 3 by 3 in the horizontal plane, with no padding. As such, within the model the kernel is not slid across the input but only applied in one location. Invariance to 3-dimensional translation comes from the choice to apply the identical model to every location in the domain at which the input box is centered.

The same activation function, applied after each hidden layer other than the output layer, is the commonly used rectified linear unit (ReLU) function,

$$\mathrm{ReLU}(s) = \begin{cases} s, s > 0 \\ 0, \text{else} \end{cases} . \tag{25}$$

There is no bias such that zero input, indicating no resolved turbulence, will be mapped to zero output, indicating no subgrid turbulence, which is most appropriate for large-eddy simulation considering the forward energy cascade and assuming quick equilibration of subgrid variables with resolved fields. The Adam optimizer (Kingma & Ba, 2014) is used for training and learning rates are initialized to 10^{-2} and reduced to 10% their current value for the first 2 epochs, reaching a minimum of 10^{-4} and remaining at that value. We did not find that the model performance is particularly sensitive to the choice between mean square error or mean absolute error for the loss function. Mean square error is conventionally used for greater penalty of outliers. As we did not find outliers were of particular concern for this problem, we chose the mean absolute error. Detailed in the results section are various splits of training, validation, and testing data used to test the network's ability to generalize. The models are trained until the loss measured on validation data does not decrease below its minimum value after a patience of 20 epochs. Performance is ultimately measured on testing data which is used neither for training nor for early stopping.

3.3.1 Stress representation

In order to enforce SO(2)- or C_N-equivariance of the subfilter-scale stress tensor, it is useful to utilize a change of basis, which factors the stress tensor into irreducible representations of the symmetry group. Applying the matrix $\mathcal{T}$ gives the new basis, $\boldsymbol{y_T}$, in terms of the components of a vector, $\boldsymbol{y}$, consisting of only the unique components of the

stress tensor, which is symmetric by definition. That is,

$$\mathcal{T}\boldsymbol{y} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ -1 & 2 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & -1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \tau_{11} \\ \tau_{12} \\ \tau_{13} \\ \tau_{22} \\ \tau_{23} \\ \tau_{33} \end{bmatrix} = \boldsymbol{y_T} \tag{26}$$

$$= \begin{bmatrix} \tau_{11} + \tau_{22} \\ \tau_{33} \\ \tau_{13} \\ \tau_{23} \\ -\tau_{11} + 2\tau_{12} + \tau_{22} \\ \tau_{11} + 2\tau_{12} - \tau_{22} \end{bmatrix}. \tag{27}$$

Adopting the terminology of representation theory (Lang, 2012; Weiler & Cesa, 2019a), the components of this new basis are irreducible representations, $\Psi_k^{C_N}(\theta)$, which are defined for a discrete set of rotation angles, $\theta = p\frac{2\pi}{N}, p \in \{0, 1, ..., N-1\}$. We have ordered the new basis vector such that the first two components, $\tau_{11}+\tau_{22}$ and τ_{33}, each rotate as scalars, i.e. the trivial irreducible representation, $\Psi_0^{C_N}(\theta) = 1$. The next two components, $[\tau_{13}, \tau_{23}]$, rotate as a two dimensional vector with irreducible representation:

$$\Psi_1^{C_N}(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix}, \tag{28}$$

which is also called the standard representation. The final two rows also rotate as a two-dimensional vector, but with irreducible representation,

$$\Psi_2^{C_N}(\theta) = \begin{bmatrix} \cos(2\theta) & -\sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{bmatrix}. \tag{29}$$

Above, we have provided the most general approach to enforcing C_N- or SO(2)-equivariance for completeness sake. Most of the following work will focus only on the special case of C_4-equivariance. The implementation of $\Psi_2^{C_4}$, indeed for all $\Psi_{N/2}^{C_N}$ for even N, is handled as a special case and referred to as the sign representation. This simplification follows from the realization that $\sin(2p\frac{\pi}{2}),\ p \in \mathbb{Z}$, is always zero. As such, the only non-zero component of the two-dimensional $\Psi_{N/2}^{C_N}$ is $\cos(2p\frac{\pi}{2}) = \pm 1$, and we can discard the extra information. Thus, the last two components of $\boldsymbol{y_T}$ are associated with 2 1-dimensional representations, rather than 1 2-dimensional representation.

A choice must be made regarding whether the loss is calculated with the changed basis or the original stress. If the loss is calculated from the changed basis, $\boldsymbol{y_T}$, one would first change the basis of the training data during preprocessing. In practice, this can complicate the optimization of the neural network because some components, particularly τ_{12}, are always combined with other components that will likely have different magnitudes. Indeed, in our data, the magnitude of τ_{12} is nearly 30 times smaller than both τ_{11} and τ_{22}. To avoid this issue, we instead compute the loss in terms of the original stress components. This requires the application of the inverse matrix,

$$\mathcal{T}^{-1}\boldsymbol{y_T} = \begin{bmatrix} 1/2 & 0 & 0 & 0 & -1/4 & 1/4 \\ 0 & 0 & 0 & 0 & 1/4 & 1/4 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 1/2 & 0 & 0 & 0 & 1/4 & -1/4 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \boldsymbol{y_T} = \boldsymbol{y}, \tag{30}$$

to recover the original stress vector as the last step of each forward pass of the equivariant DNN.

3.3.2 Input and hidden layer representations

The irreducible representations for the input variables, $\overline{u}, \overline{v}, \overline{w}$, and $\overline{b}$, are relatively straightforward compared to those associated with the outputs. However, it is potentially counterintuitive that, though we have three components of velocity, we only treat the horizontal components, $[\overline{u}, \overline{v}]$ as a two-dimensional vector. This is natural when recalling we are only considering rotations in the horizontal plane for which we expect rotational equivariance. As a two dimensional vector, $[\overline{u}, \overline{v}]$ is associated with the standard representation (eqn. 28). For rotations in the horizontal plane, the vertical velocity is treated as a scalar with the trivial irreducible representation. As a true scalar, buoyancy is also associated with the trivial representation.

Unlike the inputs and outputs, neurons in hidden layers are not easy to interpret geometrically. However, they too must be associated with representations to ensure equivariance. In our case, we associate the neurons with the regular representations of C_N, which are the matrices associated with the cyclic permutations of N values. This representation is compatible with the ReLU activation function unlike other irreducible representations. Since the N values are not altered by permutation, ReLU can be applied component-wise to the regular representation much as in a typical neural network layer. The networks have 4 hidden layers, with 512, 256, 128, and 64 features associated with the regular representation.

3.4 Scaling approaches

In addition to the network architecture detailed in previous sections, description of a complete machine learning workflow must specify the methods for preprocessing input and output data. It is common in machine learning to normalize the inputs and outputs during preprocessing. For numerical reasons, it is beneficial to scale variables to roughly $O(1)$ values to avoid the problem of vanishing gradients, particularly with deep neural networks (Glorot & Bengio, 2010). For physical systems modeling, normalization should also transform inputs and outputs to nondimensional quantities which prevents a dependence on the choice of units. Ideally, this physical normalization would improve the generalizability of a machine learning model if the nondimensional inputs and outputs are the physically relevant parameters, as in Buckingham Pi theorem (Buckingham, 1914; Fukami & Taira, 2021; Bakarji et al., 2022; Oppenheimer et al., 2023).

In the current work, we will compare three scaling approaches: a "statistical" approach most resembling the practices in machine learning literature (LeCun et al., 2002), including those with applications to turbulence (Wang et al., 2020; Stoffer et al., 2021; Frezat et al., 2021; Wang et al., 2022a; Cheng et al., 2022), a physical scaling based on "global" features of the flow, which has a long history in fluid dynamics (Reynolds, 1895) and is similar to previous applications of machine learning to turbulence (Thuerey et al., 2020; Obiols-Sales et al., 2020), and a final physical scaling which relies on "local" features of the flow, which has been suggested for machine learning applications to turbulence but infrequently employed (Wu et al., 2018; Prakash et al., 2022).

In typical practice, abbreviated in algorithm 1, the normalized inputs, x^*, and outputs, y^*, would be computed during preprocessing and used in the evaluation of the loss. Rather than dividing the ground truth values in this manner, the scaling can be applied as a multiplicative factor to the outputs of the neural network, as shown in algorithm 2, instead. Beyond this difference, the two algorithms also compute the loss weights differently. In the 'preprocess scaling' approach, the loss weights, ω^*, are nondimensional and $O(1)$, but still necessary to balance the different components of the output. In the loss scaling approach, the weights, ω are calculated from the unnormalized outputs, so have dimensions reciprocal of those of the outputs. They are also not $O(1)$ in this approach, and instead ensure that the product, $\mathbf{y}\cdot\omega$, is $O(1)$ and unitless. The loss scaling approach avoids any division by zero and improves performance compared to the pre-

process scaling, so loss scaling is used for training every deep learning model presented here.

Algorithm 1 Preprocess Scaling

$\mathbf{x}^{\text{scale}}, \mathbf{y}^{\text{scale}} = g(\mathbf{x}, \mathbf{y})$
$\mathbf{x}^* = \mathbf{x}/\mathbf{x}^{\text{scale}}$
$\mathbf{y}^* = \mathbf{y}/\mathbf{y}^{\text{scale}}$
$\omega^* = 1/\sigma_{\mathbf{y}^*}$
while training **do**
  Loss = $|\text{DNN}(\mathbf{x}^*) \cdot \omega^* - \mathbf{y}^* \cdot \omega^*|$
  DNN.update
end while

Algorithm 2 Loss Scaling

$\mathbf{x}^{\text{scale}}, \mathbf{y}^{\text{scale}} = g(\mathbf{x}, \mathbf{y})$
$\mathbf{x}^* = \mathbf{x}/\mathbf{x}^{\text{scale}}$
$\omega = 1/\sigma_{\mathbf{y}}$
while training **do**
  Loss = $\left|\text{DNN}(\mathbf{x}^*) \odot \mathbf{y}^{\text{scale}} \cdot \omega - \mathbf{y} \cdot \omega\right|$
  DNN.update
end while

In all three of the scaling approaches, statistical, global, and local, we first subtract the input box mean from the input variables. We will denote these difference quantities with an apostrophe, *e.g.*

$$\xi' = \overline{\xi} - \langle \overline{\xi} \rangle \tag{31}$$

where $\langle \cdot \rangle$ is the average over the 3×3×3 (or 5×5×3) grid cells which make up the input box at whose center we seek to predict the subfilter-scale stress. This shifts the distributions of the input variables to be centered around zero to improve generalizability by limiting out-of-sample inputs. Additionally, this practice awards Galilean invariance, *i.e.* switching reference frame to one moving at a constant velocity would have no effect. Galilean invariance could also be achieved by subtracting the mean of the velocities taken across the entire domain, a practice perhaps more common in machine learning applications to fluid mechanics. However, such an approach would be impractical to implement in weather and climate models which utilize domain decomposition to parallelize computation. Beyond these practical issue, taking a mean over a local input box is grounded by the same theory that motivates the local scaling approach. Indeed, subtracting the mean of only the input box prohibits flow far from where the subfilter-scale stress is predicted to have any influence on the calculation except through the statistical and global scaling factors.

3.4.1 Statistical scaling

Statistical properties of variables are often used in normalization for machine learning. While this helps to avoid the vanishing gradient issue during model training, it is unlikely to improve generalizability of the model, as unseen test data may have substantially different distributions than the data used to train the model. To test this, we use a statistical scaling in one version of our DNN momentum closures.

For variables that are associated together as components of a vector, we use the statistics of the norms,

$$|u_h'| = \sqrt{(u')^2 + (v')^2} \tag{32}$$

$$|\tau_h| = \sqrt{\tau_{11}^2 + 2\tau_{12}^2 + \tau_{22}^2} \tag{33}$$

$$|\tau_{i3}| = \sqrt{\tau_{13}^2 + \tau_{23}^2}. \tag{34}$$

This approach will not violate rotational equivariance as would a statistical approach that normalized each component by its own standard deviation.

The normalized inputs, $\vec{x}^*$, and outputs, $\vec{y}^*$, are

$$\mathbf{x}^* = \begin{bmatrix} u' \\ v' \\ w' \\ b' \end{bmatrix} \odot \begin{bmatrix} 1/\sigma(|u_h'|)_{\text{train}} \\ 1/\sigma(|u_h'|)_{\text{train}} \\ 1/\sigma(w')_{\text{train}} \\ 1/\sigma(b')_{\text{train}} \end{bmatrix}, \quad \mathbf{y}^* = \begin{bmatrix} \tau_{11} \\ \tau_{12} \\ \tau_{13} \\ \tau_{22} \\ \tau_{23} \\ \tau_{33} \end{bmatrix} \odot \begin{bmatrix} 1/\sigma(|\tau_h|)_{\text{train}} \\ 1/\sigma(|\tau_h|)_{\text{train}} \\ 1/\sigma(|\tau_{i3}|)_{\text{train}} \\ 1/\sigma(|\tau_h|)_{\text{train}} \\ 1/\sigma(|\tau_{i3}|)_{\text{train}} \\ 1/\sigma(|\tau_{33}|)_{\text{train}} \end{bmatrix}. \tag{35}$$

where $\sigma(\xi)$ is the standard deviation of a variable, ξ, calculated across the entire training sample set, as indicated by the subscript.

3.4.2 Global scaling

Due to the imposed external forcing, there are global variables that can be used to nondimensionalize the physical system. The resulting scaled inputs and outputs are candidates for nondimensional parameters in the sense of Buckingham Pi theorem, which makes them likely to improve the generalizability of the model. We term such a scaling the "global" approach with normalized inputs and outputs,

$$\mathbf{x}^* = \begin{bmatrix} u' \\ v' \\ w' \\ b' \end{bmatrix} \odot \begin{bmatrix} \frac{\sqrt{Re}}{U_g} \\ \frac{\sqrt{Re}}{U_g} \\ \frac{\sqrt{Re}}{U_g} \\ \left(b\left(z_i\right) - b_0\right)^{-1} \end{bmatrix}, \quad \mathbf{y}^* = \frac{Re}{U_g^2} \begin{bmatrix} \tau_{11} \\ \tau_{12} \\ \tau_{13} \\ \tau_{22} \\ \tau_{23} \\ \tau_{33} \end{bmatrix} \tag{36}$$

where Re is the Reynolds number, $b\left(z_i\right)$ is buoyancy at $z = z_i$, the boundary layer height, and b_0 is buoyancy at the surface. The temperature relaxes to the reference temperature, θ_0, at the boundary layer height, so $b\left(z_i\right) = 0$ from the definition in 2 used here. We give the height dependence to note that potential temperatures at the same heights could be substituted for buoyancy without affecting the nondimensional parameter if potential temperature also replaces buoyancy as an input feature.

Though these normalization factors are specific to each simulation, the factors are global in the sense that their values are constant, or nearly so, within each simulation domain. In the current work, these factors are easy to approximate from the DNS. For implementation in real-weather LES, such scaling factors would require more effort to diagnose. A global scaling could utilize values specific to each atmospheric column, in which case we may prefer to designate the approach as a "column" rather than "global" scaling. In this case, the geostrophic wind can be taken as the wind speed near the boundary layer height. The boundary layer height is often diagnosed for each column by planetary boundary layer schemes and such methods could be adapted for this purpose. Alternatively, it is possible to calculate a characteristic geostrophic wind for a domain, see Connolly et al. (2020) for example, which would be updated as the state of the simulation evolves.

As the Reynolds number Re is defined in terms of the geostrophic wind, substitution of eqn. 20 produces an equivalent expressions for the normalization factor,

$$\frac{U_g^2}{Re} = \frac{\nu U_g}{z_i}. \tag{37}$$

We prefer the first form because the square of the free stream velocity is familiar from classic theory of aerodynamic drag. Further, in this theory, the inverse of the Reynolds number may also appear in the drag coefficient, for example laminar pipe flow has this form (Moody, 1944). This connection to drag is likely not coincidental, as the above scaling drastically outperforms a simpler scaling factor, U_g^2 without Re, in unshown test cases similar to those in the following sections. This is an example of how deep learning can be used to test hypotheses. In this case, we have tested which nondimensional parameter is the physically relevant Pi group. We have chosen the form given in eqn. 36 based on the improved generalizability it affords a deep learning model.

Though a global scaling undoubtedly contains useful physical information, the use of global variables appears to conflict with principles of turbulence modeling at LES resolutions. At these resolutions, the global forcing should primarily influence the resolved turbulence and affect the subfilter scales mostly indirectly through the resolved fields. By the same logic, a global or column scaling is more theoretically grounded at RANS resolutions, for which there is no resolved turbulence to mediate the effects of large scale forcing. Despite these theoretical objections, we find the global scaling often yields the best performing models.

3.4.3 Local scaling

Following the scale similarity principles of LES, we test another scaling, still based on physical quantities, but local to the grid cells within each input box. Particularly promising scaling factors are based on estimates for turbulent kinetic energy (TKE), E_k, to scale velocities and stresses, and turbulent potential energy (TPE), E_p (Zilitinkevich et al., 2008), to scale buoyancy. To avoid introducing additional equations to prognosticate these variables, we use the algebraic expressions based on the Taylor expansions involving only grid-scale variables given in eqn. 8 (Bedford & Yeo, 1993). For isotropic box filter of width, Δ, these expression are

$$2E_k = \overline{u_i u_i} - \overline{u_i}\,\overline{u_i} = \frac{\Delta^2}{12}\frac{\partial \overline{u_i}}{\partial x_j}\frac{\partial \overline{u_i}}{\partial x_j} + \text{h.o.t.}, \tag{38}$$

$$2E_p = \frac{\overline{b^2} - \overline{b}^2}{N^2} = \frac{\Delta^2}{12N^2}\frac{\partial \overline{b}}{\partial x_j}\frac{\partial \overline{b}}{\partial x_j} + \text{h.o.t.}. \tag{39}$$

Following these approximations, we define an anisotropic formulation, which separates the contribution to TKE from the horizontal velocity components and those due to the vertical velocity. Additionally neglecting both the higher order terms and the constant coefficients in (38) and (39), we define

$$k_h = \Delta x_j^2 \left(\frac{\partial \overline{u}}{\partial x_j}\right)^2 + \Delta x_j^2 \left(\frac{\partial \overline{v}}{\partial x_j}\right)^2 \tag{40}$$

$$k_v = \Delta x_j^2 \left(\frac{\partial \overline{w}}{\partial x_j}\right)^2 \tag{41}$$

$$k = k_h + k_v \tag{42}$$

$$e_p = \frac{\Delta x_j^2}{N^2}\left(\frac{\partial \overline{b}}{\partial x_j}\right)^2. \tag{43}$$

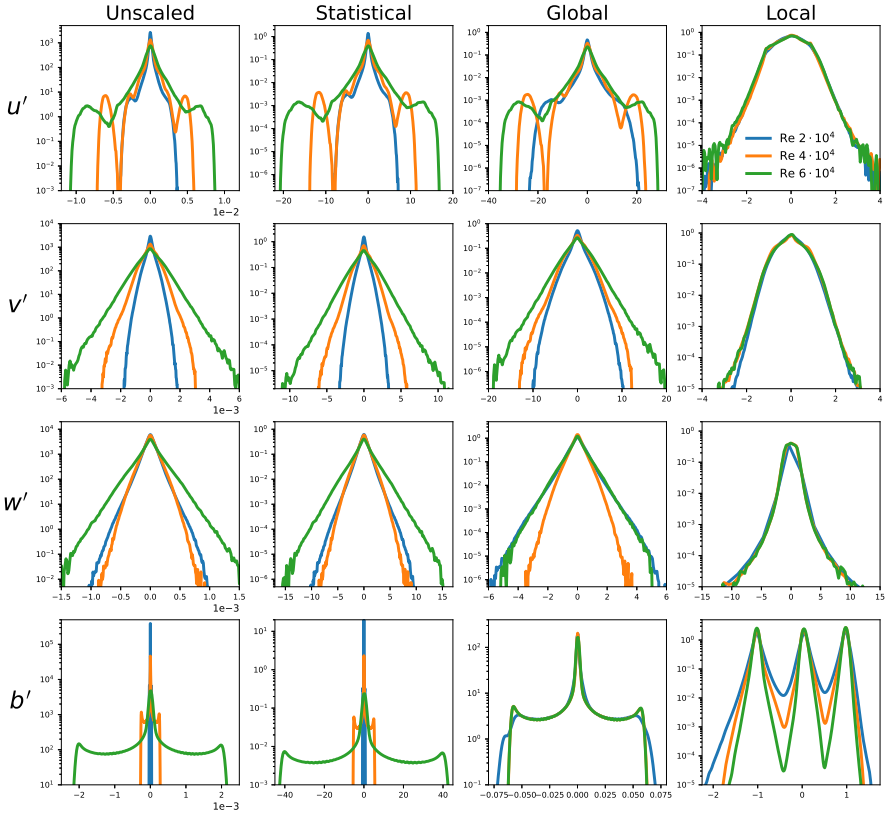

Figure 4. Distributions of input variables, velocity and buoyancy, from each of three different Reynolds numbers considered, all coarse-grained to the same intermediate resolution grid. Shifted, by subtracting the input box mean, but unscaled values shown on the left are followed in the subsequent columns by those scaled through the 3 different approaches, statistical, global, and local, described in the body text. Unscaled velocity components and buoyancy have units of [m s^{-1}] and [m s^{-2}], respectively, and scaled variables are dimensionless.

Evaluating these scaling factors at the center of each input box, the normalized inputs and outputs are

$$\mathbf{x}^* = \begin{bmatrix} u' \\ v' \\ w' \\ b' \end{bmatrix} \odot \begin{bmatrix} k^{-\frac{1}{2}} \\ k^{-\frac{1}{2}} \\ k_v^{-\frac{1}{2}} \\ \frac{\Delta z}{e_p} \end{bmatrix}, \quad \mathbf{y}^* = \begin{bmatrix} \tau_{11} \\ \tau_{12} \\ \tau_{13} \\ \tau_{22} \\ \tau_{23} \\ \tau_{33} \end{bmatrix} \odot \begin{bmatrix} (k)^{-1} \\ (k)^{-1} \\ (k \cdot k_v)^{-\frac{1}{2}} \\ (k)^{-1} \\ (k \cdot k_v)^{-\frac{1}{2}} \\ (k_v)^{-1} \end{bmatrix}. \tag{44}$$

3.4.4 Distributions of scaled variables

Before training the networks on data scaled through each of the three approaches, statistical, global, and local, we may compare the approaches through the distributions

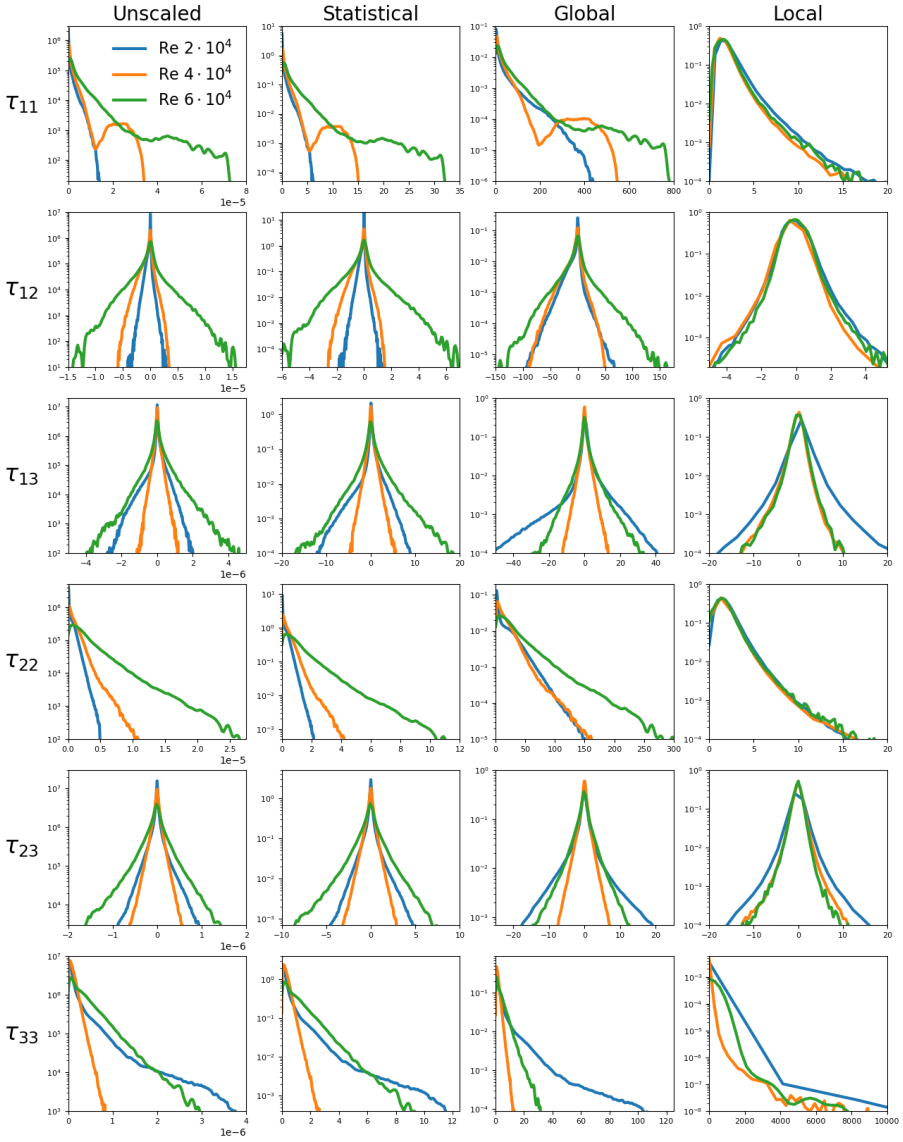

Figure 5. Distributions of subfilter-scale stresses from each of three different Reynolds numbers considered, all coarse-grained to the same intermediate resolution grid. Unscaled values shown on the left are followed in the subsequent columns by those scaled through the 3 different approaches, statistical, global, and local, described in the body text. Unscaled stress components have units of $[\mathrm{m}^2\ \mathrm{s}^{-2}]$ and scaled variables are dimensionless.

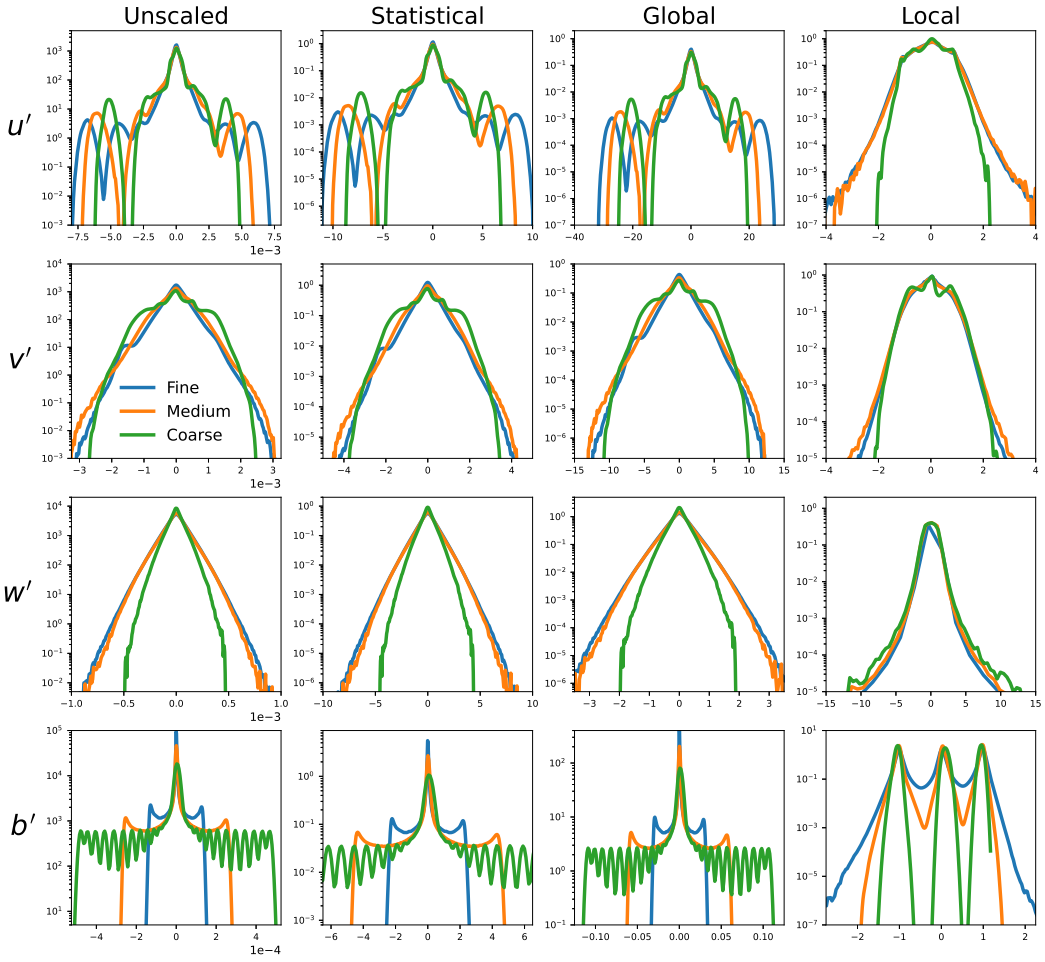

Figure 6. As in figure 4 but comparing the three different coarse-grained grid resolutions considered, all derived from the same intermediate Reynolds number DNS.

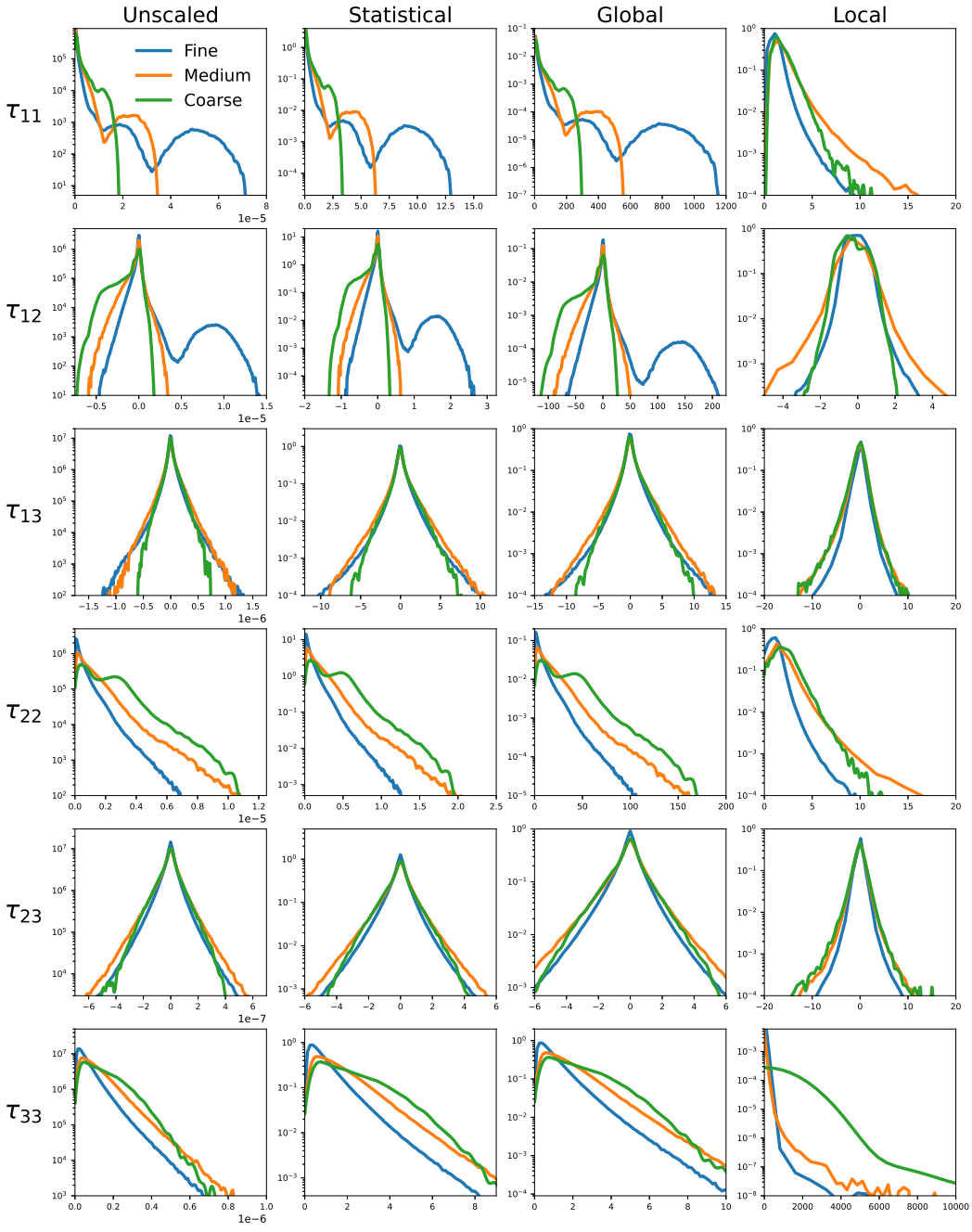

Figure 7. As in figure 5 but comparing the three different coarse-grained grid resolutions considered, all derived from the same intermediate Reynolds number DNS.

of the scaled variables. First, the case of variable Reynolds number is shown for the intermediate grid resolution, $\Delta x = \Delta y$ =0.86 m and Δz =0.28 m, in figures 4 and 5. Second, the case of variable resolution is shown for the intermediate Reynolds number, $Re = 4 \cdot 10^4$, in figures 6 and 7. While the physical scaling approaches, global and local, do not depend on the train–test split, the statistical approach does. The scaling shown in the distribution plots follow from train–test splits used for extrapolation tasks detailed in the following section. It is shown that the local scaling best collapses the distributions of both input and output variables across different resolutions and Reynolds numbers.

In figures 4 and 5, we see the distribution of an unscaled variable differ substantially across Reynolds numbers. The distributions are made most similar with the local scaling, while they continue to differ with the global scaling and differences are even exacerbated by the statistical scaling. The local scaling affects the shape of the distributions, particularly evident for buoyancy and $\overline{u}$ velocity, which are prominently multimodal in the unscaled distributions. The statistical and global scaling approaches do little to address the peculiarities of these distributions, while the local scaling results in distribution collapse near a single mode for $\overline{u}$ velocity and about three modes, one for each vertical level in the input box, for buoyancy. Retaining multiple modes in the scaled buoyancy might be expected given the appreciable vertical gradients of buoyancy in the stable boundary layer.

In the comparisons across resolutions, figures 6 and 7, there are some improvements unique to the local scaling approach, but the collapse of distributions from local scaling is not as absolute as it was across Reynolds number. In particular the normal stresses do not collapse well with any scaling. The multimodal distributions of unscaled $\overline{u}$ velocity and buoyancy are improved by local scaling in ways similar to those in the Reynolds number comparisons. In addition, the horizontal shear stress, τ_{12}, in the highest resolution data is also collapsed only by the local scaling. For this stress, contributions to the secondary mode are largely from the near surface region where shear is highest, which explains why the mode is only observed on the high resolution grid that best resolves this region. This increased shear corresponds to an increase in turbulent kinetic energy, on which the local scaling factors are based. As such, the local scaling can account for the differences of this near surface region in a way that the statistical and global scaling approaches cannot. This explains the unique capability of the local scaling to collapse the distribution around a single mode.

In every case, unscaled inputs and outputs are orders of magnitude smaller than $O(1)$. With any scaling, the magnitude of variables are brought nearer $O(1)$, which is certain to improve the optimization of the deep learning models. Indeed, deep neural networks attempting to learn from these unscaled variables will not train at all. In many cases, only the local scaling approach leads to reasonable distribution collapse, both across Reynolds number and grid resolution. This is promising for generalizability, and local scaling does indeed aid the prediction of subfilter-stress in offline comparison to test data significantly different than the training data. Results from such comparisons across a host of generalization tasks is the subject of the section 4. However, in the online setting, local scaling in the deep learning parameterization leads to numerical instability in cases for which other scaling approaches lead to stable simulation. Section 5 will analyze the use of these scaling approaches in online tests of generalizability in more details. Next, we detail the large-eddy simulation configurations used for these online tests.

3.5 Large-eddy simulation configurations

The DNN momentum closures are implemented in a fluid dynamics code, microHH, designed for both DNS and LES of the atmospheric boundary layer (Heerwaarden et al., 2017). Though a relatively newer code, microHH has already been validated on numerous case studies and even participated in an intercomparison of LESs of the stable bound-

ary layer (Couvreux et al., 2020) similar to those conducted here. Further, the more modern code of microHH interfaces well with the demands of deep learning. Critically, the code is entirely C++, as is as library for the popular machine learning library, torch. Indeed, the DNNs are trained with the python language version of torch, PyTorch. The developers also offer a torch API in C++ called libtorch. At initialization, this library is used to load a just-in-time (JIT) trace version of the DNN, generated in python beforehand. The DNN could not be defined, saved, and loaded directly with libtorch because of its reliance on the third party python library, e2cnn, used for symmetry constraints. During time integration, libtorch functions are used to conduct inference with the loaded DNN in parallel on the CPU responsible for computations of DNN inputs, which are intentionally local in part for this reason.

The LESs are forced as similarly to the DNSs as possible. At the lateral boundary, both the LES and DNS are periodic except for an imposed pressure gradient corresponding to user assigned geostrophic wind forcing. The model tops are also similar, zero Neumann aided by a sponger layer covering the top-most 25% of the domain. The bottom boundary condition for buoyancy is given as a Neumann boundary condition in the DNS. For the LES, we convert this to a constant flux boundary condition using the molecular diffusivity of heat, which we assume equal to molecular viscosity. This buoyancy flux is also an input to the surface layer scheme, based on Monin-Obukohv Similarity Theory (MOST), which diagnoses the vertical fluxes of horizontal momentum for the LES bottom boundary condition. The surface layer model also requires specification of the roughness length which is not provided by the smooth wall DNS, nor are there roughness elements from which z_0 could be approximated from their lengths. Though there have been proposed relationships linking roughness length and the surface stress for smooth wall flow (Li et al., 2016), we did not employ these. Instead, we use trial and error and conducted a number of LESs with the Smagorinsky closure to determine that $z_0 = 10^{-5}$ leads to rough agreement with the DNS at the first horizontal velocity points above ground level which are most affected by the surface drag. Similarly, no turbulent Prandtl number is provided by the DNS, but required by the LES to diagnose the thermal diffusivity from the momentum diffusivity of the Smagorinsky model. Again, we used trial and error to determine that a value of $Pr = 10$ leads to reasonable agreement between LES using Smagorinsky and the DNS in the near-surface buoyancy profiles. A relatively large value for this turbulent Prandtl number agrees with previous literature for such high Richardson number cases considered here (Katul et al., 2014). Note, the subfilter fluxes of buoyancy are diagnosed by the Smagorinsky model even when the DNN is employed for momentum fluxes.

While the DNSs are initialized with constant mean profiles, the LESs are initialized with mean profiles taken from the DNSs spun up for hours of simulated time. The LESs nonetheless need their own spin-up period. This is because the three-dimensional fields are generated by applying uniform perturbations with height-dependent amplitudes to the mean profiles. The structure of this applied noise is different than the structure of the resolved turbulence in the DNS, and additional time is required to adjust to a more physical structure in the LES. The maximum amplitude of the perturbations and the exponent of the decay of the amplitude with height can be tuned. We chose 10% of geostrophic wind for the maximum amplitude of the horizontal velocities and 1% for the vertical. The maximum perturbation amplitude for the buoyancy field is 0.5% its surface value obtained from the DNS. The exponent for the decay of these amplitudes are 0.5, 2 and 4 for horizontal velocities, vertical velocity and buoyancy, respectively.

Finite differencing, to calculate tendencies from the divergence of the subfilter-scale stress, is done with the smallest stencil possible for a centered difference. This stencil varies depending on the momentum component and the direction of the gradient, due to the staggered grid. The DNN expects inputs at grid centers, so momentum is first destaggered with two-point averaging and boundary values are sent to neighboring domain patches.

The centered variables are then filtered over a 3x3x3 cube (or 3x3 plane at the first grid level). For momentum, this is equivalent to trapezoidal-rule averaging with box length of $4\Delta s$ in the s-direction. This filter-to-grid ratio of 4 is what the DNNs were trained on, so we see the best performance with this additional filtering. Buoyancy, which is native to grid centers, is only filtered to $3\Delta s$ by this procedure. Carrying out the trapezoidal-rule averaging over 4 grid cells near the edge of the domain patches would require additional ghost points. Rather than restructuring the code for this, the mismatch between training and implementation is tolerated because we don't expect buoyancy to be as important as momentum for predictions based on the ablation studies shown later.

For the local scaling approach, the turbulent kinetic energy based scaling factors are computed from the destaggered, but not filtered, velocities and used to nondimensionalize the filtered inputs to the DNN. The scaling factors are then filtered before being used to dimensionalize the outputs of the DNN. Filtering the scaling factors themselves, rather than computing them from the filtered velocities, limits the occurrences of predicting much higher or lower energy than at neighboring cells due to anomalous high wavenumber flow features, better reflecting the energy levels over the finite difference stencil. Though we hope they will benefit the LES, these practical choices are ultimately made more from intuition than any rigorous underpinning.

Several other practical choices were made during the implementation. Importantly, we can not diagnose subfitler-scale stress at the first grid cell above the bottom surface, because our model uses a stencil with 3 vertical levels. We opted for very simple linear interpolation between the fluxes diagnosed by MOST or, if not diagnosed, assumed zero at the surface and those at the second grid cell from the DNN. Additionally, we have implemented the choice to use the deviatoric or the full subfilter-scale stress as a user-designated switch. When taking the deviatoric component, the pressure is formulated as a modified pressure which includes the mean normal subfilter-scale stress. We found choice in deviatoric or full stress made little difference in our simulations. Apparently, corrections from the pressure Poisson solver constrain the total normal stress strongly enough to render inconsequential any reasonable formulation for the normal stress in the turbulence scheme. By default, we calculate tendencies with only the deviatoric subfilter-scale stress. This aids comparisons to a conventional eddy-diffusivity closures like Smagorinsky which are designed only to predict the deviatoric stress (eqn. 4). Further, using a deviatoric subfilter-scale stress seems more appropriate when diagnosing pressure through the pressure Poisson, which can not distinguish between thermodynamic pressure and the contribution to total normal stress from subfilter-scale turbulence.

4 *A priori* tests

4.1 Generalizability to unseen Reynolds number and resolution

The coarse-grained datasets differ in two primary ways. One is the Reynolds number of the direct numerical simulation (DNS) from which the coarse fields are derived. The other is the resolution to which they are coarsened. There are also two ways to test the capabilities of a deep neural network (DNN) to generalize: interpolation and extrapolation. Given the relatively low Reynolds number and fine resolution, extrapolation to higher Reynolds numbers and coarser grids is the most likely application of deep learning turbulence closures trained from expensive even if idealized DNS data such as ours. In the following sections, we test the ability to extrapolate and interpolate of a C_4-equivariant deep neural network utilizing each of the scaling approaches detailed in sections 3.4.1 – 3.4.3.

A reasonable hypothesis is that the statistical scaling will do well for interpolation tasks, because the statistics taken across the training data should be roughly equal to the statistics if taken instead from the testing data, which is by design intermediate for

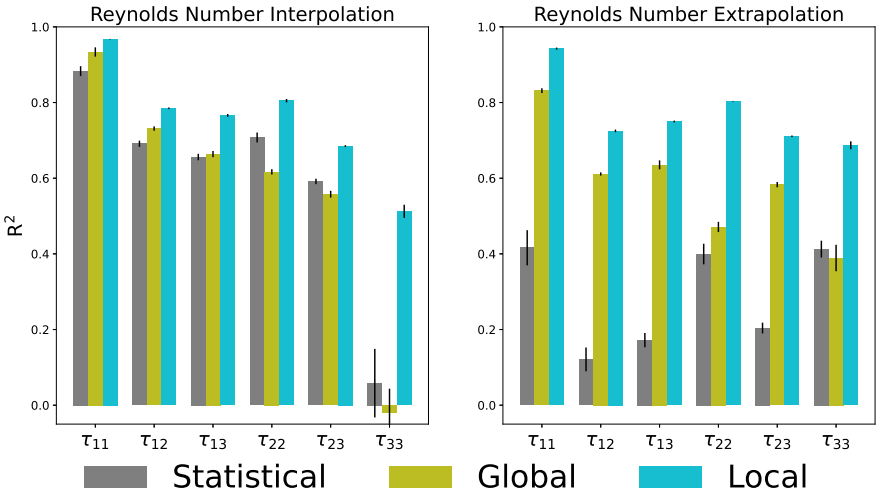

Figure 8. Coefficient of determination, R^2, values for DNN model predictions of the subfilter-scale stress. Deep neural network models are identified by which of the scaling approaches, statistical, global, or local, is used. In the Reynolds number interpolation test, the Reynolds number of the test data is unseen but bounded by those of the training datasets. For extrapolation, the Reynolds number of the test data is greater than that of the training data.

interpolation. By the same logic, we expect the statistical scaling to fail the extrapolation tasks, because the model will encounter out-of-distribution samples. Another reasonable hypothesis is that the global scaling will do well in the Reynolds number generalization tasks, because related physical variables and even the Reynolds number itself are used in this scaling approach. However, this global scaling approach does not utilize any information pertaining to the grid resolution, so we expect the global scaling to fail on the grid generalization tasks. In comparison, the local scaling does utilize information on the grid resolution, so we expect it will succeed at the grid tasks. Additionally, the local scaling has indirect information on the Reynolds number, through the turbulent kinetic energy and turbulent potential energy estimates, so we expect it will succeed at Reynolds numbers tasks. Further, we expect the local scaling to succeed not just at interpolation but also at extrapolation, because the distributions of inputs and outputs nearly collapse (Figs. 4 – 7) such that there should be few out of distribution samples.

To test these hypotheses, we run a series of experiments in which we withhold some of the DNS-derived, coarse-grained data and train on data which differs either in the Reynolds number or the coarse-grain resolution. Half of the withheld data, the validation data, will be used for early-stopping of the training loop (algorithm 2), to prevent overfitting. The other half, the test data, are used to calculate the R^2 values illustrated in figures 8 and 9 and detailed in tables A1 – A4. We report both a mean and standard deviation, as a margin of error, from 5 training runs for each of the three scaling approaches.

4.1.1 Unseen Reynolds number

We test the ability of the deep neural network to generalize to Reynolds numbers not seen during training. In the interpolation case we train on data with the lowest, $Re = 2 \cdot 10^4$, and highest, $Re = 6 \cdot 10^4$, Reynolds number data, and test the model on inter-

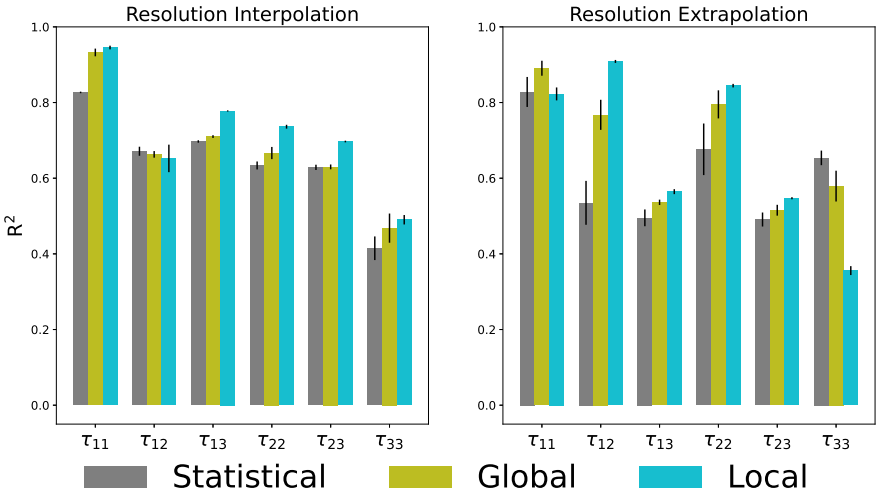

Figure 9. Coefficient of determination, R^2, values for DNN model predictions of the subfilter-scale stress. Deep neural network models are identified by which of the scaling approaches, statistical, global, or local, is used. In the resolution interpolation test, the resolution of the test data is unseen but bounded by those of the training datasets. In extrapolation, the resolution of the test data is coarser than that of the training data.

mediate, $Re = 4 \cdot 10^4$, Reynolds number data. In the extrapolation case, we train on data with the lowest and intermediate Reynolds numbers and test the model on the highest Reynolds number data. For these experiments, we hold grid resolution constant at the intermediate resolution, $\Delta x = \Delta y = 0.86$ m and $\Delta z = 0.28$ m.

Given the varying number of samples available for the different datasets (table 2), we employ a simple data balancing technique. For both the interpolation and extrapolation tests, we randomly select only 10% of possible samples from the lowest Reynolds number dataset and use all samples from either the intermediate or highest Reynolds number data. In the interpolation test, in terms of number of samples, this biases the training data towards the lower Reynolds number. However, the stresses from this simulation are lesser in magnitude than those of the DNS with highest Re and, considering how loss is calculated on re-scaled values by algorithm 2, equalizing the samples between these two datasets would result in an inductive bias favoring the higher Reynolds number data. Taking slightly more samples from the lower Reynolds number than from the higher Reynolds number data seems a good compromise between balancing the number of samples and the magnitude of the sampled stresses. Conversely, in the extrapolation test, the number of samples from the lowest Reynolds number data is less than those from the intermediate Reynolds number data. This imbalance is exacerbated by the fact that the stresses from the intermediate Reynolds number are also greater magnitude. In this case, data imbalance is actually preferred because the bias favors the data with Reynolds number closest to that of the data on which we will be testing the model.

The performance of the DNN models utilizing different scaling techniques are illustrated in figures 8 through the coefficient of determination. For interpolation to unseen Reynolds number, numerical presentation of the statistic is given in table A1 and in table A2 are the statistics for the extrapolation test. The performance of the conventional parameterizations (eqns. 4 - 9) on the same datasets is also given in the tables. For each case, local scaling leads to the best prediction of every component of the subfilter-

scale stress. For interpolation, DNN models with any scaling generally outperform the existing closures. The only exceptions are the Clark model, which predicts τ_{13} slightly better than DNNs using statistical or global scaling, and predicts τ_{23} better than the DNN with global scaling. However, in line with our hypothesis, the statistical scaling performs generally much worse for the extrapolation task than for interpolation, and is outperformed by the Clark model for most stress component predictions and the Bardina model for half the components.

Inability to extrapolate to unseen Reynolds number, as seen for the statistical scaling approach, is not the case when physical scaling is used. Indeed, there are a number of components for which predictions from the physically scaled DNN models are better on the extrapolation test data than from the same models on the interpolation test data. Though both physical scaling approaches exhibit some extrapolation capability, the local scaling is better than the other methods for this extrapolation to unseen Reynolds numbers, as it was for interpolation. This supports our hypothesis, that the local scaling will excel at extrapolation because of the collapse of distributions.

4.1.2 Unseen Resolution

We test the ability of the deep neural network to generalize to grid resolutions not seen during training. In the interpolation case we train on data with the finest, $\Delta x = \Delta y = 0.43$ m and $\Delta z = 0.14$ m, and coarsest, $\Delta x = \Delta y = 1.7$ m and $\Delta z = 0.57$ m, grid data, and test the model on intermediate, $\Delta x = \Delta y = 0.86$ m and $\Delta z = 0.28$ m, resolution data. In the extrapolation case, we train on data with the lowest and intermediate grid resolutions and test the model on the coarsest grid data. For these experiments, we hold Reynolds number constant at the intermediate value, $Re = 4 \cdot 10^4$.

Data balancing for the grid generalization tasks is similar to the approach in the Reynolds number generalization experiments in that we take all data from the training dataset with the fewest possible samples. For the interpolation test, we take all of the coarsest resolution data but randomly select only 2.5% of the finest resolution data. This choice reflects a compromise between balancing the number of samples and the magnitude of those fluxes. In the extrapolation test, we take all of the intermediate resolution data, and about 12.2% of the finer resolution data. In this extrapolation case, our choice is to simply balance the number of samples, because the magnitude of the stresses are similar enough between the two training datasets.

The performance of the DNN models utilizing different scaling techniques are illustrated in figure 9 through the coefficient of determination. For interpolation to unseen resolution, numerical presentation of the statistic is given in table A3 and in table A4 are the statistics for the extrapolation test. Note that the test data for grid interpolation is the same as that for Reynolds number interpolation, so the performance of the conventional turbulence closures is the same as in the previous interpolation test. The results show that the DNN models do well at the interpolation task, outperforming the existing parameterizations. For interpolation, the DNN using local scaling is the best performing model for all but one component, τ_{12}, which is better predicted by the DNN with statistical scaling. However, the difference in mean R^2 for this component is within the margin of error of the DNN with local scaling. For local scaling, the standard deviation of R^2 of the τ_{12} component is notably larger than that of any other component of the subfilter-scale stress for reasons that are not immediately clear. As we hypothesize that statistical scaling should succeed at interpolation, its superior prediction of τ_{12} is not entirely counter to expectation. A result that does run counter to our expectation is the success of the DNN using global scaling, whose performance at predicting data with grid resolutions unseen during training is comparable to the other DNN and better than the existing closures. Despite the grid spacing not being incorporated into the global scaling factors, the global scaling approach even leads to the best pre-

Layer	Baseline	C_4 (3×3)	C_8 (3×3)	C_4 (5×5)	C_8 (5×5)
Input	110 592	36 864	36 864	67 584	67 584
Hidden 1	524 288	524 288	1 048 576	524 288	1 048 576
Hidden 2	131 072	131 072	262 144	131 072	262 144
Hidden 3	32 768	32 768	65 536	32 768	65 536
Output	768	384	384	384	384
Total	799 488	725 376	1 413 504	756 096	1 444 224

Table 3. Number of trainable parameters in each layer and total for the baseline, non-equivariant model architecture as well as C_N-equivariant models which take inputs from 3 vertical levels with either 3×3 or 5×5 grid cells per level.

dictions for one stress component, τ_{11}, in the resolution extrapolation test. Another result, in line with our hypothesis, the statistical approach is once again the worst performing scaling approach for the extrapolation task at least for all components but one, τ_{33}, for which it is surprisingly the best performing model. Other than these two components, the local scaling DNN produces the best predictions for extrapolation, supporting our hypothesis that the local scaling should perform well in extrapolation as well as interpolation.

4.2 Generalizability to flow orientation: comparison to data augmentation

To approximate invariance and equivariance in machine learning models, a common technique is data augmentation. In the context of rotational equivariance, data augmentation entails manually rotating the training data so that the model is exposed to a variety of data orientations during training. While there is some theoretical work comparing data augmentation and equivariant architectures (Wang et al., 2022b), we perform an experiment to compare these techniques.

For this experiment, a second model architecture is used that is in most ways similar to the architecture previously considered but does not enforce any equivariance. The number of trainable parameters cannot be identical in the equivariant and baseline models, due to specifics of their implementations. The best compromise, summarized in table 3, allows for an equal number of trainable parameters in all but the input and output layers for which the baseline architecture has a factor of 4 more trainable parameters.

We compare the predictions from this non-equivariant, baseline model on rotated data when trained with and without data augmentation. To simplify the analysis, we consider only the Reynolds number interpolation task when using local scaling, the best performing scaling approach. Example contours of stresses, ground truth values as well as model output are shown in figure 10, and statistical metrics are illustrated in figure 11 and given in table A5.

Without data augmentation, the baseline model's performance with rotated data is significantly reduced compared to its skill at predicting data with the same orientation as the training data. Data augmentation improves these statistical metrics, with the baseline model trained with augmented data performing significantly better on rotated data. Indeed, the baseline model trained with augmentation performs slightly better on rotated data than the baseline model trained only on the unrotated data does on unrotated test data, though this difference is within the margin of error. However, examining figure 10, we see the predictions from rotated inputs are not precisely what we would obtain from rotating the predictions made from inputs with the original orientation. The

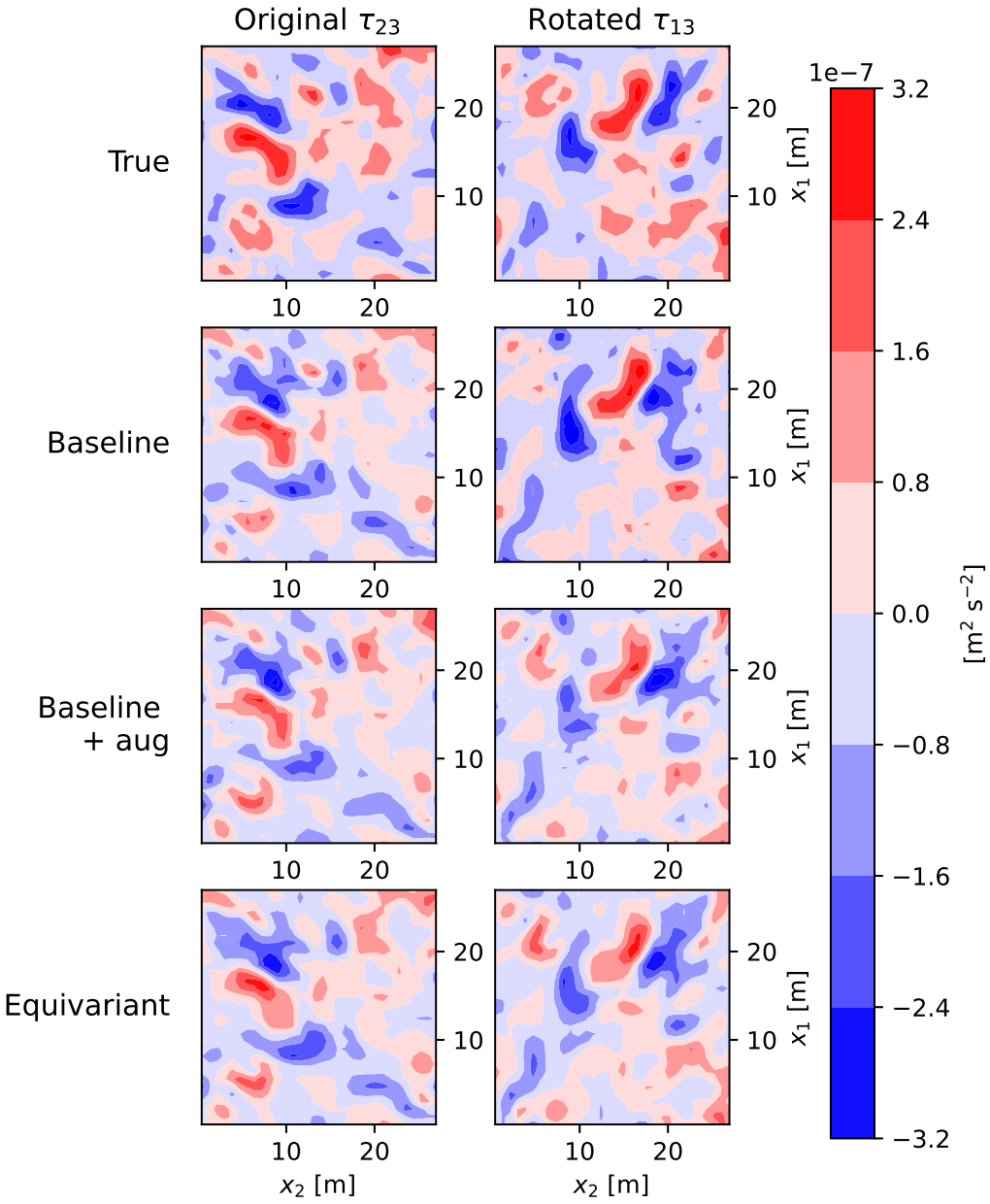

Figure 10. Horizontal cross-section of equivalent components of stress with the original orientation (left) and after rotation by 270° (right). The true DNS-derived stresses (top) are shown along with outputs from various models described in the body text.

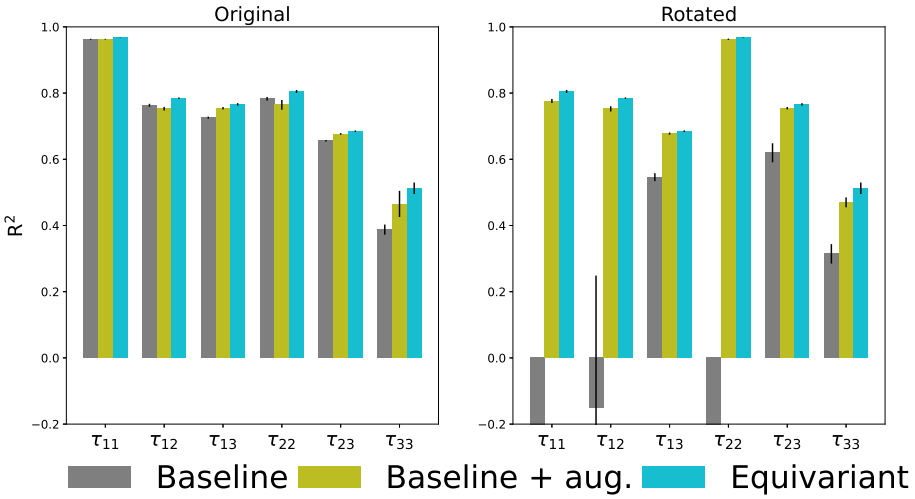

Figure 11. Coefficient of determination, R^2, values for DNN model predictions of the subfilter-scale stress. Test data either retains the original flow orientation or is rotated by 270°. Baseline models do not enforce equivariance and are trained with or without data augmentation through the inclusion of training samples randomly rotated by either 0°, 90°, 180°, or 270°. Local scaling is utilized and grid resolution is held constant while the Reynolds number of the test data is unseen but bounded by those of the training datasets.

equivariant model, on the other hand, does exhibit this precision. Further, despite the gains made through data augmentation, the predictive skill of the equivariant model is higher whether testing on unrotated or rotated data.

4.3 Buoyancy ablation

A prominent difference between the existing turbulence closures presented in section 2.1, is that the Bardina (eqn. 7) and Clark (eqn. 8) models do not involve buoyancy while the Smagorinsky-Lilly model (eqns. 4 – 6) does include buoyancy through N, the buoyancy frequency. Comparisons of these models' predictions (tables A1 – A4), show that the Clark and Bardina models outperform the Smagorinsky-Lilly model in *a priori* tests. This suggests that the information pertaining to subfilter-scale stress contained in the buoyancy field is at best redundant, because the effect of buoyancy is already felt by the resolved velocity field, likely vertical velocity primarily. At worst, inclusion of buoyancy is even deleterious to predictions.

A useful application of deep neural networks as universal approximators is to test hypotheses. Here, we test the hypothesis that including buoyancy improves prediction of subfilter-scale stress through an ablation study. To do so, we revisit the tests of Reynolds number extrapolation and retrain models with similar architecture as the C_4-equivariant model with local scaling. In some models, we replace the buoyancy input channels with noise drawn from a standard normal distribution. The comparisons are done for each grid resolution considered. We balance data between the different resolutions so that each training has roughly the same number of points as the coarse grid, which has the fewest possible samples (table 2). As in the previous Reynolds number extrapolation tests, for the lowest Reynolds number data, we take only 10% of the fraction taken from intermediate Reynolds number data. Results from this experiment are illustrated in figure 12

and given in table A6 as means and standard deviations from 10 training runs. The entries corresponding to the model, either with buoyancy or with noise, that yielded a better average coefficient of determination are denoted with an asterisk (highlighted gray in the table) if the improvements are statistically significant ($p < 0.05$).

In terms of the mean of the R^2 statistic, inclusion of buoyancy is generally beneficial. Indeed, at the finest and intermediate resolution, prediction of every component is improved in a statistically significant way by including buoyancy. Even at coarse resolution, more components of the subfilter-scale stress are improved than degraded by including buoyancy. The τ_{11}, τ_{13}, and τ_{22} are improved with buoyancy; τ_{12} and τ_{23} see no statistically significant effect of buoyancy; and τ_{33} is predicted worse with buoyancy than with noise. Even when the effects of buoyancy are statistically significant, a measure which is relative to the variability across training runs, the effect is modest in a more absolute sense. Whether or not this improvement is worth the associated computational expense is a subjective decision. Future LES users who will make this subjective choice can be better informed by the results presented in this ablation study.

4.4 Sample size and resolution effects

The numerical experiments introduced in previous sections provide other insights. For one, we see that at finer resolutions, prediction skills, with or without buoyancy, increase in figure 12 (table A6). Since we have balanced the amount of training data across resolutions, we have experimentally verified the intuitive notion that the subfilter-scale stress is inherently more predictable at finer resolutions.

As extrapolation to unseen Reynolds number on the medium resolution grid was done twice, with a little over 8 times more training data in one experiment, we get a sense of how data limited the models are. Comparing the results in figure 8 (table A2) to those in figure 12 (table A6), we see that more training data does lead to better predictions. As such, it is reasonable to expect that the model skills measured in the current work could be further improved with even more data.

4.5 C_8-equivariant models

We investigate the effect of enforcing higher order cyclic group equivariance. For these experiments we use Reynolds number interpolation as a test problem. Figure 13 and table A7 show the predictive skill of a model equivariant to the C_8 cyclic group, corresponding to rotations by multiples of 45°, compared to the model used in previous results which is equivariant for only multiples of 90°, the C_4 cyclic group.

With no further modifications to the model (figure 13; leftmost columns of table A7) we see enforcing C_8-equivariance reduces model skill compared to C_4-equivariance. Particularly for the τ_{12} and τ_{22} components, mean R^2 for model prediction drops dramatically, but no component's prediction improves with C_8-equivariance. A likely explanation for this decrease in skill is the error due to the interpolation required to represent 45° rotation on a 3×3 grid (Weiler et al., 2018; Diaconu & Worrall, 2019), the horizontal extent of the input box. To test this hypothesis, we train another set of models, still with inputs from three vertical levels, but expanding the extent of each horizontal plane to 5×5 grid cells. This does increase the predictive skill of a C_8-equivariant DNN (figure 13; table A7, right column), but only marginally. We also confirm these improvements are due to better representation of the rotations on a larger 'image' rather than from the increase of information in the inputs by increasing the input box widths for a C_4-equivariant model as well. In the C_4-equivariant case, the same increase in input information has little effect on a model prediction, and even reduces skill for some components. It is possible that a further increase in input box width would further reduce the error by better resolving finer rotation angles. However, the computational cost as-

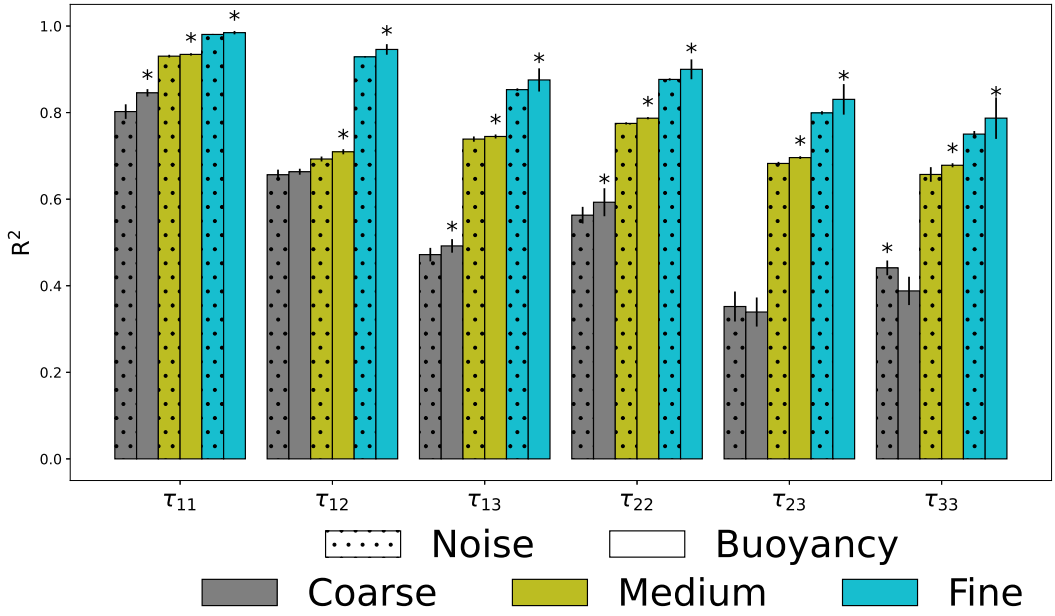

Figure 12. Coefficient of determination, R^2, values for DNN model predictions of the subfilter-scale stress. Which grid is used and whether buoyancy is input or replaced with standard normal noise are varied as indicated. The better performing model is indicated with an asterisk if the difference between buoyancy or noise is statistically significant ($p < 0.05$). Local scaling is utilized and grid resolution is held constant while the Reynolds number of the test data is greater than those of the training datasets.

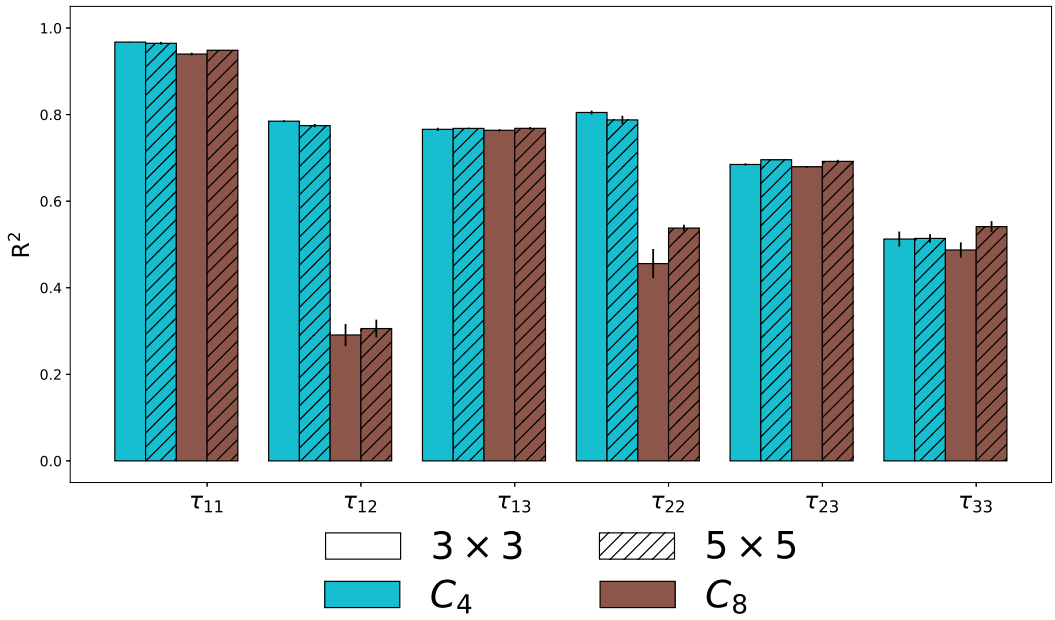

Figure 13. Coefficient of determination, R^2, values for model predictions of the subfilter-scale stress. Either C_4 or C_8 group equivariance is enforced for rotations in a horizontal plane. Inputs are taken for 3 vertical levels with horizontal widths extending either 3 or 5 grid cells in each direction and using a convolutional kernel of the same width, 3 or 5 cells, as the input box. Local scaling is utilized and grid resolution is held constant while the Reynolds number of the test data is unseen but bounded by those of the training datasets.

sociated with increased input dimension and higher order equivariance (table 3) is hard to justify. As such, the C_4-equivariant model seems a better choice than a C_8-equivariant model.

5 *A posteriori* tests

For the following *a posteriori* tests, we implement our deep neural network (DNN) momentum closures in large-eddy simulations (LESs) of the stable atmospheric boundary layer. We force an LES similarly to the direct numerical simulation (DNS) to which it will be compared. The LESs are initialized with mean profiles from these DNSs taken from the first time step available. The LES equations (Eqn. 1), with subfilter-scale stresses diagnosed by DNN models trained offline with various coarse-grained DNS data, are then integrated in time for the same duration as the DNS. This allows comparison of the DNS time evolution with that of the LES until the latest DNS time step available. Due to the high computational cost of the DNSs, which largely motivates this and all work in turbulence modeling, the durations of the DNSs are insufficient to fully spin up the LESs. To better understand the long term effects of the DNN parameterization, we also run the LESs for a longer period of time and perform an intercomparison of LESs only.

In every case we find utilizing the DNN with locally scaled inputs and outputs results in numerical instability and model blow-up in less time than the DNS duration. In section 5.4, we discuss possible explanation and recourse in more detail. Simulations using DNNs making use of the other scaling methods remain stable, at least for the duration of the DNS. This allows comparison between the DNS and LESs utilizing DNNs with either a statistical scaling (eqn. 35) or a physical scaling, the global variant based on imposed forcing (eqn. 36). In sections 5.1 – 5.3, we analyze the performance of these LESs in *a posteriori* tests which mirror the *a priori* tests presented in section 4.1. Further, we use the same DNNs trained and evaluated in those previous experiments to allow more direct comparison between offline skill and online performance. As in the *a priori* tests, we select training data which differs, by either characteristic Reynolds number or grid resolution, from the simulation for which the DNN will be used.

5.1 Interpolation to unseen Reynolds number and resolution

To test the ability of the DNN models to interpolate to unseen Reynolds number and resolution, we perform large-eddy simulations with intermediate, $Re = 4{\cdot}10^4$, Reynolds number, on a grid with spacing, $\Delta x = \Delta y = 0.86$ m and $\Delta z = 0.28$ m, the intermediate resolution. The LESs are labeled in figures 14 – 16 to indicate which scaling method was used, 'Stats.' or 'Phys.' for statistical or physical scaling. labeled 'Re interp.' is one set of simulations in which the subfilter-stress is diagnosed by a DNN trained on lower, $Re = 2{\cdot}10^4$, and higher, $Re = 6{\cdot}10^4$, Reynolds numbers data. The Re interp. simulations have the same grid spacing data as the coarse-grained data from which the DNN was trained. Another set is labeled 'res. interp.' and uses DNNs trained on finer, $\Delta x = \Delta y = 0.43$ m and $\Delta z = 0.14$ m, and coarser, $\Delta x = \Delta y = 1.7$ m and $\Delta z = 0.57$ m, resolution but the same intermediate Reynolds number data. These variations allow us to test the effect of our DNN closures when required to interpolate across either the Reynolds numbers or resolutions of their training data in simulations with the same forcing and grid.

Profiles of domain averaged variables in figure 14 show that, even over the short duration of the DNS, the LES employing the Smagorinsky closure begins to exhibit overmixing near the surface, particularly at the jet maximum in the v-velocity. The strength of this maximum is much better predicted in LES using any DNN closure, which better maintain the strong near-surface gradients present throughout the duration of the DNS. Above the jet, the velocity profiles in the LESs begin to agree with each other and with the DNS.

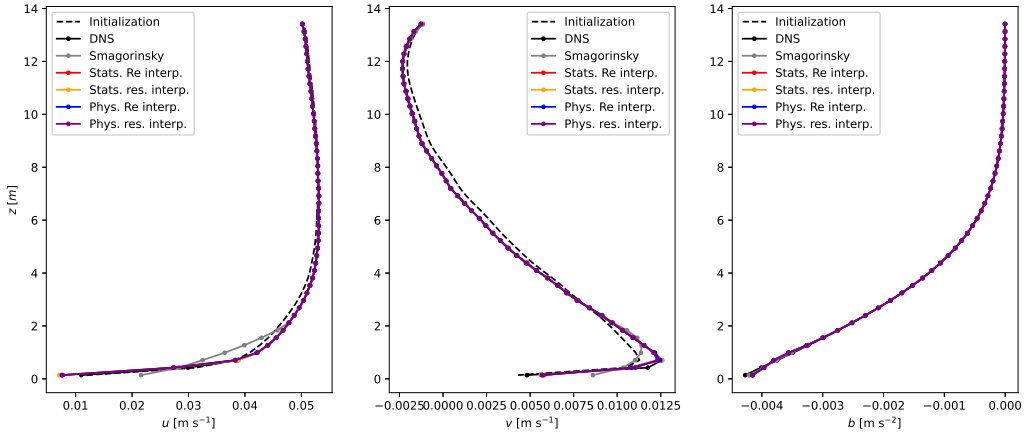

Figure 14. Mean profiles from LESs initialized by mean profiles from DNS at the earliest available time (dashed black) compared to the latest available time of the DNS (solid black). The LESs vary by which turbulence closure is used, either Smagorinsky (gray) or one of several DNN models: two use the statistical scaling (red and orange) and two use the physical scaling (blue and purple). Those labeled 'Re interp.' use training data with the same resolution as well as lower and higher but not the current Reynolds number (red and blue). Those labeled 'res. interp.' use training data with the same Reynolds number as well as finer and coarser but not the current resolution data (orange and purple).

Above the region of sharp near-surface gradients, where the mean profiles show little differences, the LESs still distinguish themselves by the structure of resolved turbulence, summarized in figures 15 and 16 showing power spectra of u- and w-velocity. In figure 15, we see the spectra from grid levels nearest 2 m and 10 m at both the start of the LES and at the last time for which DNS data are available for comparison. The LESs are all initialized with the same perturbed fields, so their spectra are identical at the initial time (left column). We also see that the random perturbation initialization procedure results in spectra which are much flatter than those from the DNS which show the negative slope indicative of a forward cascade of energy. By 23 min and 20 s after initialization, at the end of the DNS duration, the spectra of LES using Smagorinsky show notably lower energy than the DNN-enabled LES, which are largely similar. At this time, the energy at the lowest wavenumbers remains underestimated and, at the highest wavenumbers, overestimated, though there are indications that the LES spin up is ongoing. Importantly, the slopes of the LES spectra at intermediate wavenumbers have become more negative since initialization. This is most notable at 2 m and in the w-velocity spectra of LES using a DNN with physical scaling, which may be contributing to faster spin up than the statistical scaling.

We continue the LESs past the duration of the DNS from which they were initialized to present an LES intercomparison in figure 16, showing the power spectra of of u- and w-velocity at the grid levels nearest 2 m above ground level over the course of 3 h of simulated time. Eventually, the LESs using either DNN with physical scaling and the 'Stats. res. interp.' LES mostly agree. Given the constant forcing, we might expect the final spectra to cluster around a quasi-equilibrium which displays a shape that matches our physical expectations. For one, the energy at the lowest wavenumbers, which was underestimated at initialization, does increase soon after the end of the DNS duration when it was still too low (figure 15, right). Second, the slope of the spectra at highest

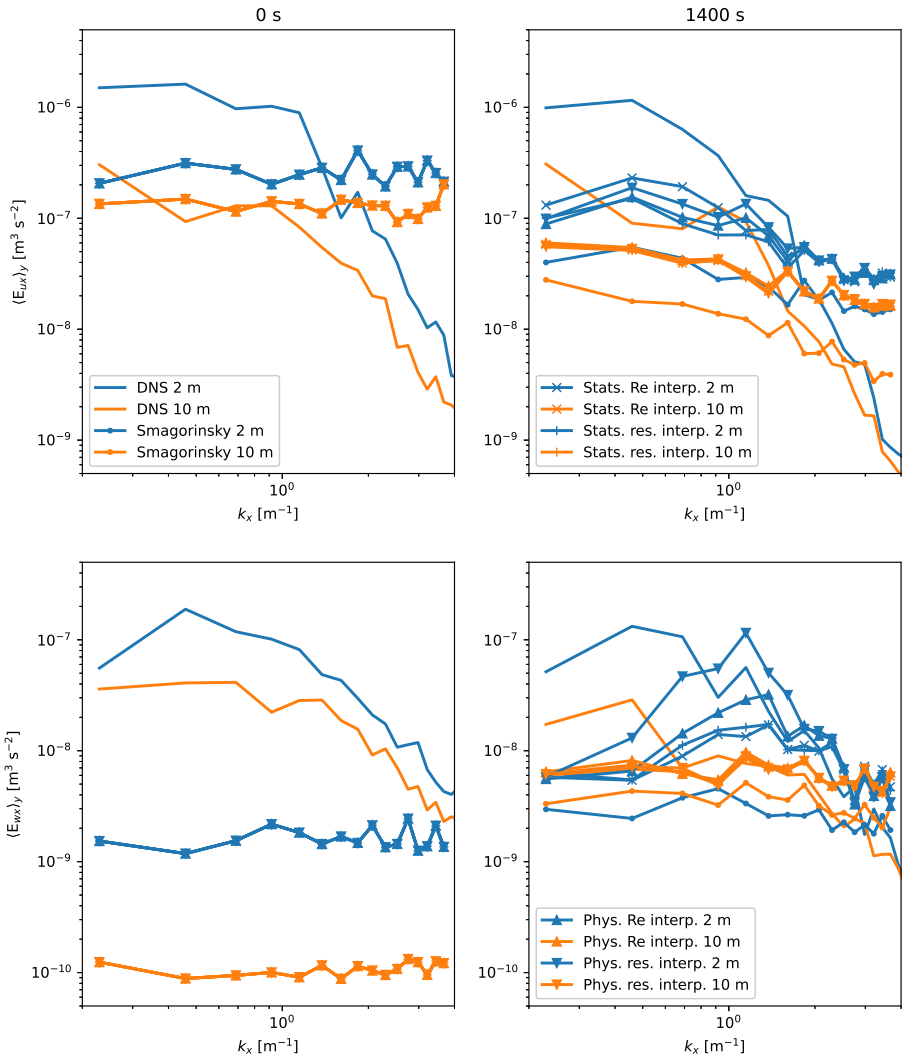

Figure 15. Power spectra of u-velocity (top) and w-velocity (bottom) as a function of x-direction wavenumber and averaged in the y-direction from the $Re = 4 \cdot 10^4$, Reynolds number DNS and various LESs forced similarly to the DNS. The LESs are the same as those in figure 14, distinguished by hash marks, and velocities are from the vertical levels nearest 2 m (blue) and 10 m (orange) at the time of initialization (left) and the final DNS time step (right).

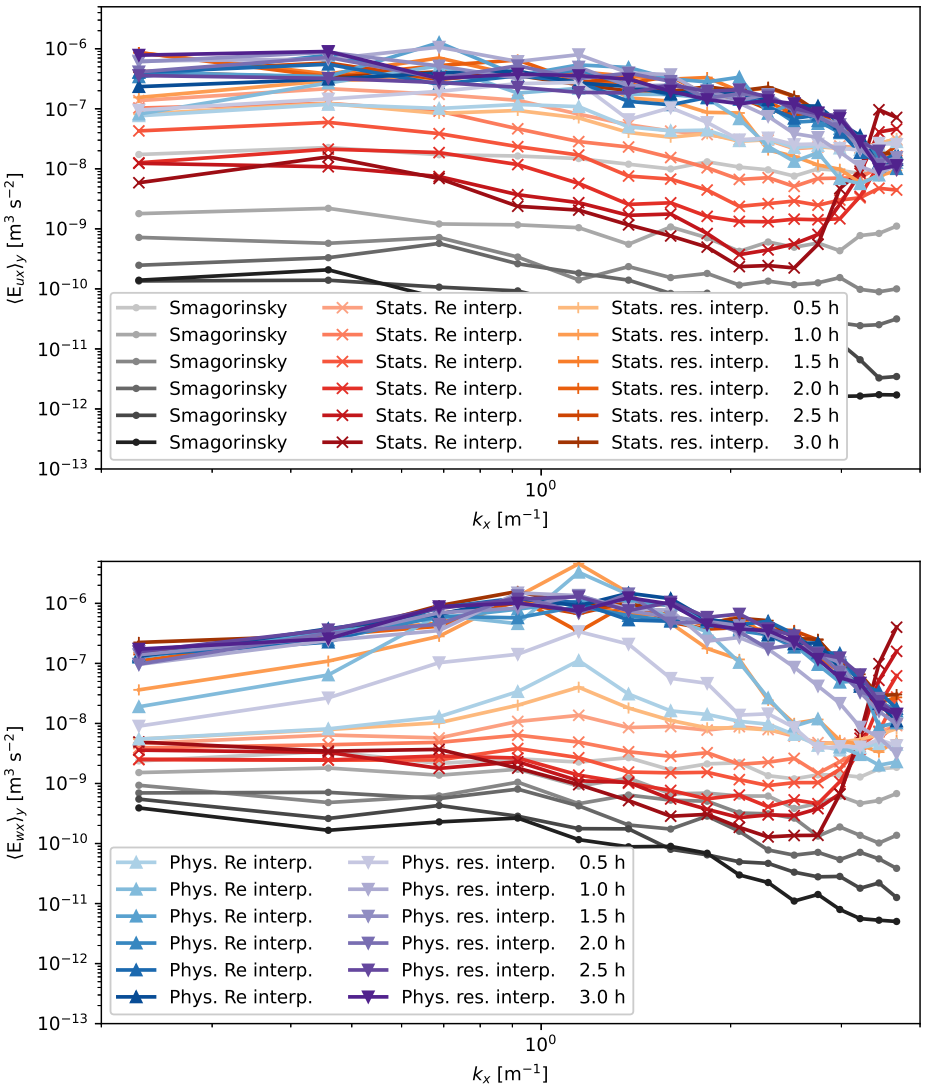

Figure 16. Power spectra of u-velocity (top) and w-velocity (bottom) as a function of x-direction wavenumber and averaged in the y-direction from the $Re = 4 \cdot 10^4$, Reynolds number LESs. The LESs vary by which turbulence closure is used, either Smagorinsky (grays) or one of several DNN models. Two use the statistical scaling (reds and oranges) and two use the physical scaling (blues and purples). Those labeled 'Re interp.' use training data with the same resolution as well as lower and higher but not the current Reynolds number (red and blue). Those labeled 'res. interp.' use training data with the same Reynolds number as well as finer and coarser but not the current resolution data (orange and purple). Successive shades are taken a half hour apart until 3 h of simulation time.

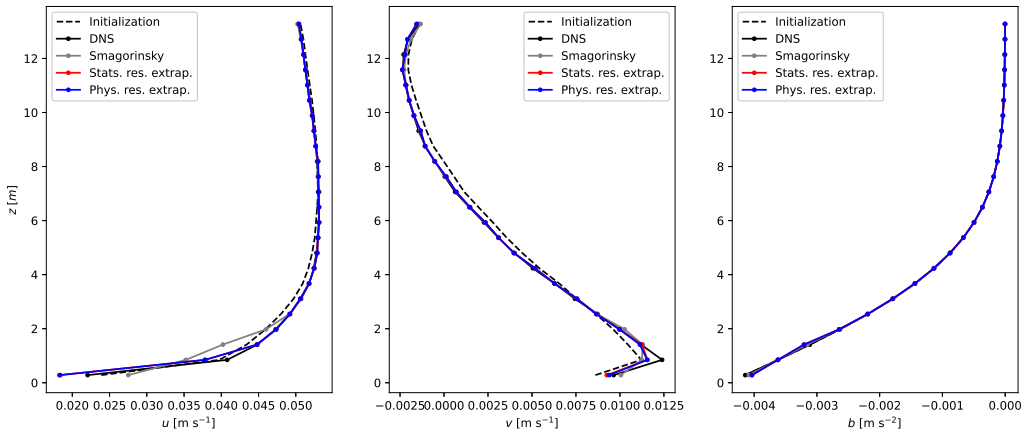

Figure 17. Mean profiles from LESs initialized by mean profiles from DNS at the earliest available time (dashed black) compared to the latest available time of the DNS (solid black). The LESs vary by which turbulence closure is used, either Smagorinsky (gray), a DNN using statistical scaling (red), or a DNN using physical scaling (blue). For these 'res. extrap.' tests, the LESs have grids coarser than the coarse-grained DNS data from which the DNNs were trained, though the Reynolds number is the same.

wavenumbers, which was still too flat at the end of the DNS duration, eventually develop the expected negative slope in the range of resolved dissipation. This is in contrast to the LES utilizing the DNN tasked with Reynolds number interpolation using statistical scaling, which develops a very unphysical spectra, with energy at the highest wavenumbers growing in time and even surpassing the energy found at the larger scales which is conversely decreasing. The LES using the Smagorinsky model appears to fail to maintain resolved turbulence at any scale by rapidly losing energy at all wavenumbers throughout the duration of the simulation.

It is notable that the LESs using the physical scaling approach agree even though they use DNNs trained with entirely different datasets. The failure of an LES with statistical scaling approach contradicts our hypothesis that statistics based normalization will suffice for interpolation tasks. Indeed, the statistical scaling did suffice for the *a priori* test, with higher R^2 than the global physical scaling DNN predictions for half the components of the subfilter-scale stress. The implications are that offline skill is not entirely predictive of online skill and that, even for interpolation tasks, there are benefits to utilizing physical scaling over statistical normalization in hybrid, physics combined with deep learning, models.

5.2 Extrapolation to coarser resolution

To test the ability of the DNN models to extrapolate to coarser resolutions than those seen during training, we perform LES with the coarsest resolution, $\Delta x = \Delta y = 1.7$ m and $\Delta z = 0.57$ m, at the intermediate Reynolds number, $Re = 4{\cdot}10^4$. The LESs are labeled in figures 17 & 18 to indicate which scaling method was used, 'Stats.' or 'Phys.' for statistical or physical scaling. These 'res. extrap.' simulations use DNNs trained on data from DNS of flow with the same intermediate Reynolds number but coarse-grained to grids of only finer, $\Delta x = \Delta y = 0.43$ m and $\Delta z = 0.14$ m as well as $\Delta x = \Delta y = 0.86$ m and $\Delta z = 0.28$ m, resolutions than the LES.

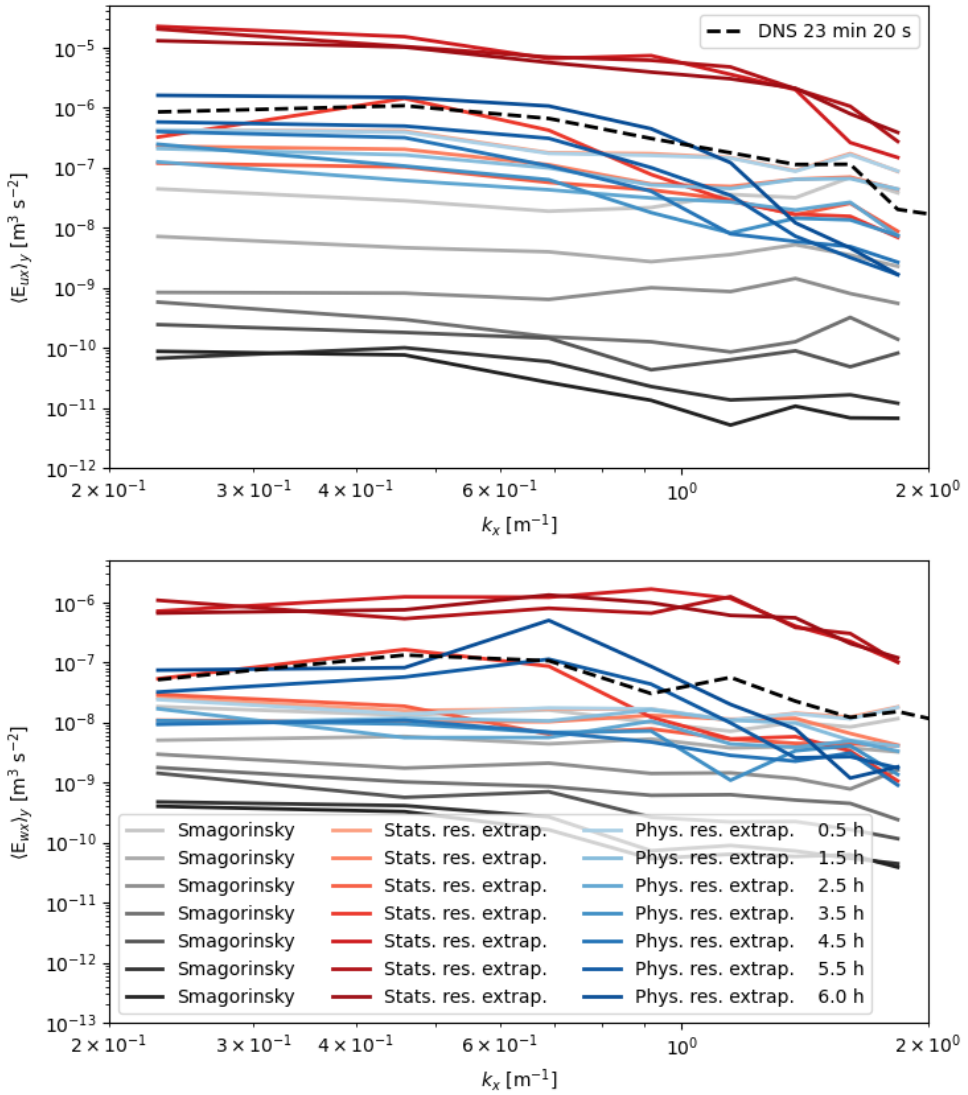

Figure 18. Power spectra of u-velocity (top) and w-velocity (bottom) as a function of x-direction wavenumber and averaged in the y-direction from the $Re = 4 \cdot 10^4$, Reynolds number LESs. The LESs vary by which turbulence closure is used, either Smagorinsky (grays), a DNN using statistical scaling (red), or a DNN using physical scaling (blue). For these 'res. extrap.' tests, the LESs have grids coarser than the coarse-grained DNS data from which the DNN were trained, though the Reynolds numbers are the same.

Profiles of domain averaged variables in figure 17 show that there is some improvement due to the DNN, but even LESs employing DNN momentum closures have shortcomings. These issues are likely related to the coarse grid itself, rather than any turbulence closure, as they are absent in the LESs on the intermediate grid considered in the previous section. In the u-velocity profiles, with no shallow jet, the DNN-enabled LESs do better maintain the strong near-surface gradient than the LES using Smagorinsky, which exhibits over-mixing. However, all LESs underpredict the strength of the near-surface jet maximum in the v-velocity profiles. To resolve such near-surface jets, an LES needs at least two grid levels below the jet maximum (Smith & Porté-Agel, 2014), and the LESs on this coarsest grid do not meet this requirement. As such, the failure of the DNN-enabled LESs in this case is traceable to intrinsic limitations of resolution, rather than a failure of the DNN models.

The LESs on the coarse grid take even longer to spin-up than those on the intermediate grid. As such, the power spectra from DNS are not the best comparison for the LES spectra. Nonetheless, we include the spectra from the latest DNS time step available in figure 18 as reference for the LES intercomparison over a longer, 6 h, duration. As at the intermediate resolution, the LES using the Smagorinsky closure loses energy at all wavenumbers throughout the duration of the simulation. When using the DNN with physical scaling, the spectra of the LES eventually agree well with that of the DNS for the largest, injection scales. At higher wavenumbers, the energy of the flow in the LES using physical scaling is less than in the DNS and also has a steeper slope, which indicates the finer resolved scales are quickly transferring energy to yet finer scales and eventually losing energy to subgrid scales. This is expected from relatively coarse simulations, whose ability to maintain fine scale flow features are hindered by numerics. The LES using the DNN with statistical scaling leads to spectra with a similar shape, but at elevated energy level compared to the DNS reference.

Though we hypothesized such failures for the statistical normalization in extrapolation tasks such as this, this particular failure is not obviously predictable on the basis of the offline test for resolution extrapolation. In the *a priori* evaluation, the statistical scaling method did reasonably well and even scored the highest R^2 of all scaling methods for the τ_{33} component (figure 9). The global physical scaling approach also did well in these *a priori* test, counter to our hypothesis that a physical scaling without information on the grid spacing would not generalize well on other grids. For the physical scaling DNN, offline skill has translated to good online performance in this case. Indeed, though there are issues related to the coarse resolution evident in the spectra as in the near-surface jet in the mean profiles, the LES using a DNN with physical scaling is comparatively successful for approximating the energy of the flow best.

5.3 Extrapolation to higher Reynolds number

To test the ability of the DNN models to extrapolate to higher Reynolds number flow than those seen during training, we perform LES with the highest Reynolds number, $Re = 6 \cdot 10^4$, at the intermediate resolution, $\Delta x = \Delta y = 0.86$ m and $\Delta z = 0.28$ m. The DNN-enabled LESs are labeled in figures 19 – 20 to indicate which scaling method was used, 'Stats.' or 'Phys.' for statistical or physical scaling. These 'Re extrap.' simulations use DNNs trained on DNS data coarse-grained to the same intermediate resolution but from simulations of flow with Reynolds numbers only lower, $Re = 2 \cdot 10^4$ and $Re = 4 \cdot 10^4$, than that of the LESs.

Profiles of domain averaged variables in figure 19 are similar to the same resolution LESs of the intermediate Reynolds number flow (figure 14) in that the LESs using a DNN closure better predict the height and strength of the v-velocity jet maximum. The DNN-enabled LESs also maintain the strong near-surface gradient in u-velocity, while

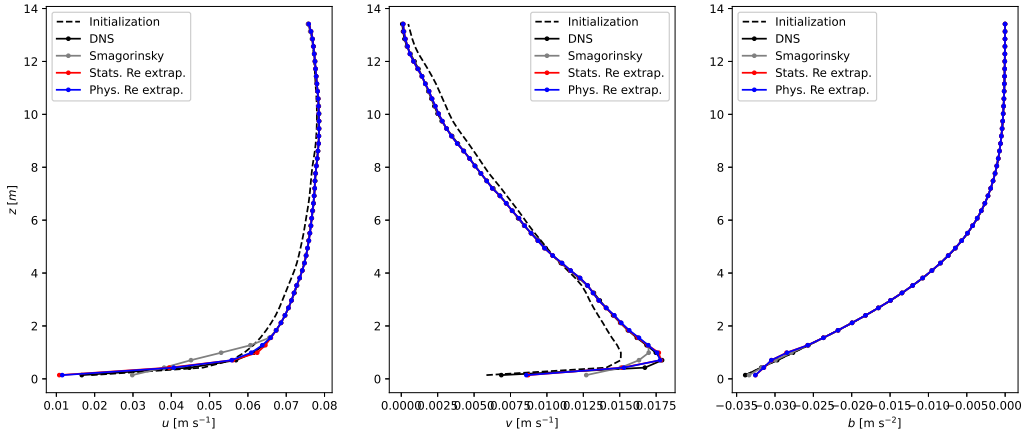

Figure 19. Mean profiles from LESs initialized by mean profiles from DNS at the earliest available time (dashed black) compared to the latest available time of the DNS (solid black). The LESs vary by which turbulence closure is used, either Smagorinsky (gray), a DNN using statistical scaling (red), or a DNN using physical scaling (blue). For these 'Re extrap.' tests, the DNNs are trained with only lower Reynolds number DNS data coarse-grained to the same intermediate grid resolution.

the LES employing the Smagorinsky closure begin to exhibit over-mixing near the surface.

As in previous cases, the power spectra from DNS are not the best comparison due to ongoing spin-up of the LESs. Despite this, we include the spectra from the latest DNS time step available in figure 20 as a reference for the LES intercomparison over a longer, 6 h, duration. The LES using the Smagorinsky closure loses energy at all wavenumbers throughout the simulation of the high Reynolds flow, as it did for the intermediate Reynolds number flow. The LESs with DNN parameterizations produce spectra which eventually converge to their quasi-equilibria, both with reasonable shapes though slightly different. The statistical scaling DNN leads to flow with greater energy, particularly in the range of resolved dissipation. Without DNS to compare at these long time horizons, we cannot conclude from the spectra alone which LES is more accurate.

However, profiles at the end of the 6 h simulation reveal the statistical scaling DNN is likely deficient (figure 21), as hypothesized for this extrapolation task. After 6 hr simulated time, the velocity profiles from the LES using a statistical scaling DNN deviate from physical expectation by the presence of additional local extrema. The LES using the physical scaling DNN does not have these additional fluctuations, and its profiles are much smoother as well. An LES using Smagorinsky has a similar and smooth shape, but predicts a deeper near-surface jet whose depth was already overestimated by the end of the DNS duration (figure 19). The LES with a DNN closure based on physical scaling produces a shallow near surface jet with physically expected profiles and spectra. These successes are more remarkable when considering the DNN was trained on data from only lower Reynolds number flow.

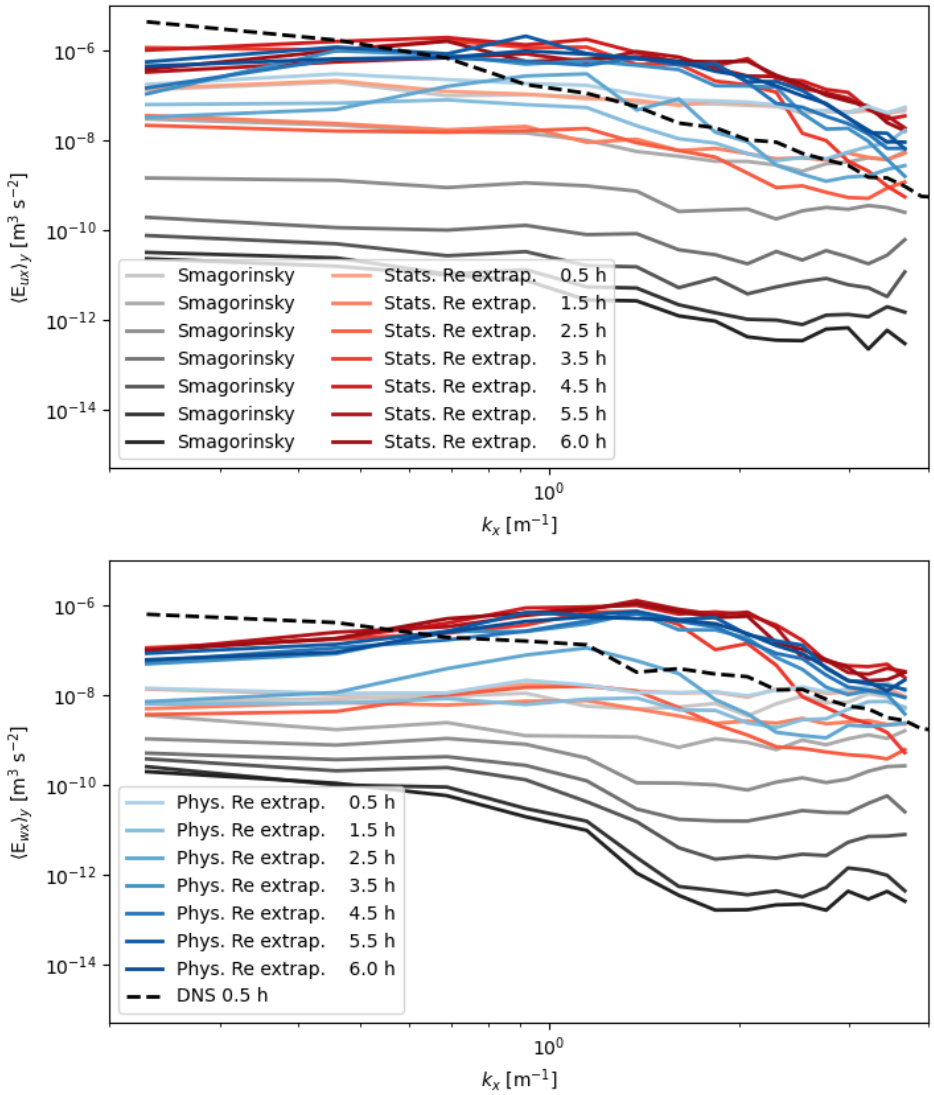

Figure 20. Power spectra of u-velocity (top) and w-velocity (bottom) as a function of x-direction wavenumber and averaged in the y-direction from the $Re = 6 \cdot 10^4$, Reynolds number LESs. The LESs vary by which turbulence closure is used, either Smagorinsky (grays), a DNN using statistical scaling (red), or a DNN using physical scaling (blue). For these 'Re extrap.' tests, the DNNs are trained with only lower Reynolds number DNS data coarse-grained to the same intermediate grid resolution.

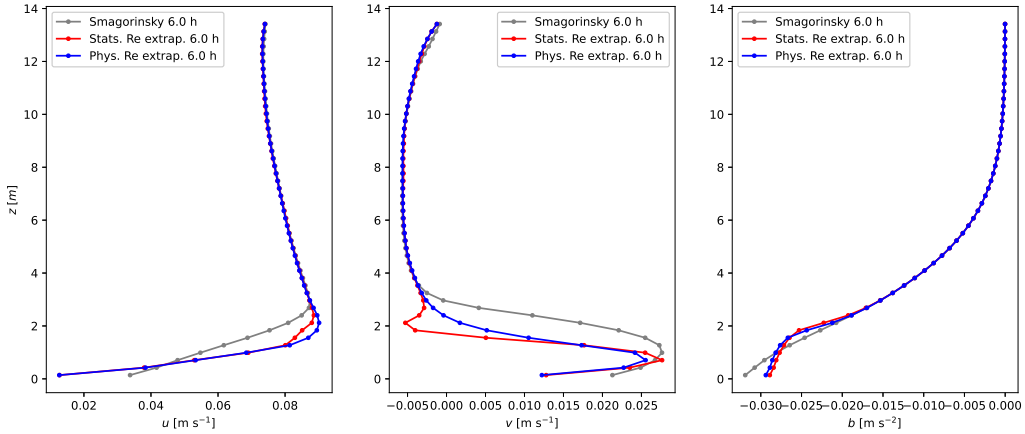

Figure 21. As in figure 19 but after 6 hr simulated time in the LESs.

5.4 LES with Local Scaling DNN

Now we analyze LES which employ a DNN turbulence parameterization with nondimensionalization scheme based on the local physics of resolved turbulence. These were excluded from previous intercomparisons because the current formulation and/or implementation of the local scaling approach leads to numerical instability. This failure is an important cautionary tale considering DNNs using this approach were the overall best performing in *a priori* tests. The nature of the model failure is similar in all cases. For analysis, we select one case corresponding to the Reynolds number interpolation task which was successfully simulated with the DNN parameterization utilizing the global, physical scaling method. During this test, an LES of the intermediate, $Re = 4{\cdot}10^4$, Reynolds number case using a DNN with local scaling trained on data with the lowest, $Re = 2{\cdot}10^4$, and highest, $Re = 6 \cdot 10^4$, Reynolds number data, failed after 743 simulation seconds and as many time steps.

5.4.1 Velocity fields and spectra

We find that domain-averaged turbulent kinetic energy is ever increasing in this and all other simulations using the local scaling DNN closures. The spectra in figure 22 reflect this and show particularly large increases in energy at the end of the simulated time. Eventually, the energy content of the flow is orders of magnitudes higher than those of the DNS and the LES using another DNN trained on the same data but using the global physics scaling approach based on imposed forcing.

Zonal velocity contours in figure 23 illustrate the overestimation of energy at the fine scales of the flow at 700 s that we saw previously in the spectra. This is the latest time for which we have DNS data for comparison before the 'Local' LES blow-up at 743 s after initialization of the LES. For comparison, the 'Global' LES using DNN with global scaling does show stronger fine scale fluctuations, attributed to perturbations at initialization in previous analysis of the model spin up, but the range of velocity is comparable to that of the coarse-grained DNS field. Effects of the initial perturbation may linger in the 'Local' LES as well, but their contribution to the energy content of the flow would be negligible by the end of the simulation compared to that of the internal dynamics.

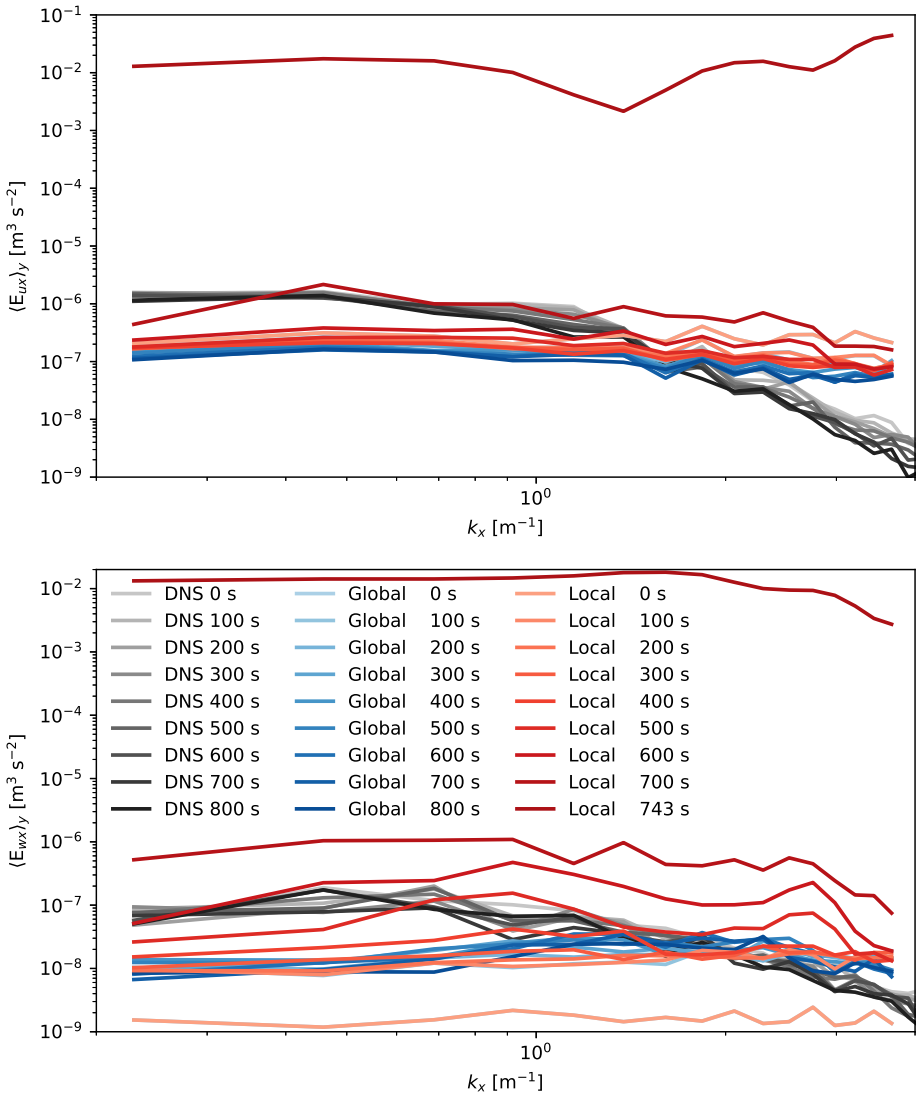

Figure 22. Power spectra of u-velocity (top) and w-velocity (bottom) as a function of x-direction wavenumber and averaged in the y-direction from the $Re = 4 \cdot 10^4$, Reynolds number DNS and various LESs forced similarly to the DNS. The 'Global' and 'Local' LESs use DNNs with scaling based on global or local physics, respectively, and trained on the same dataset, one designed to test interpolation to unseen Reynolds numbers. The 'Global' simulation remains stable while the 'Local' LES fails after 743 s simulated time.

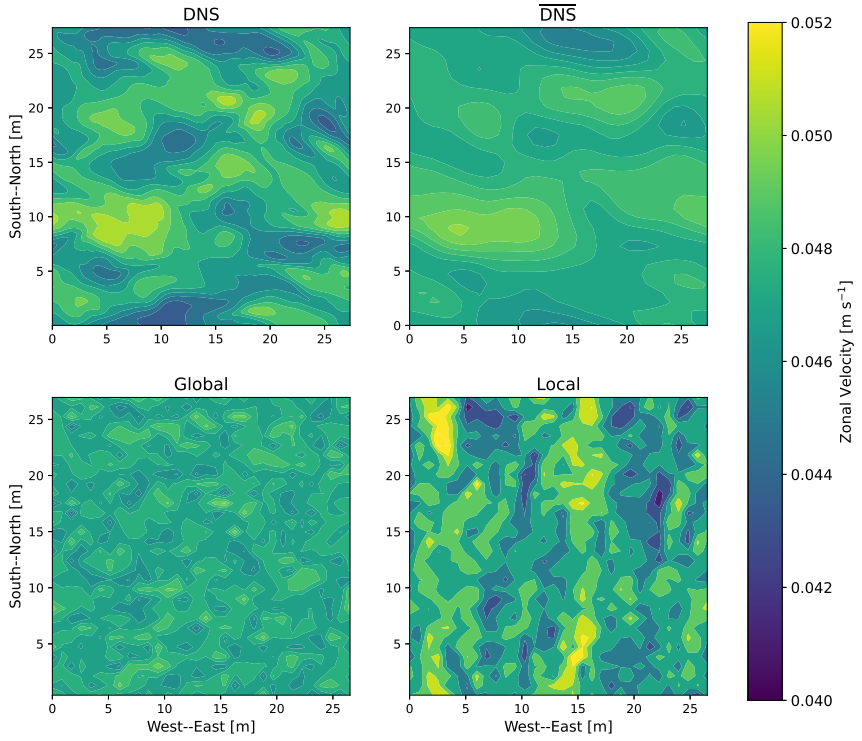

Figure 23. Contours of u-velocity in a horizontal plane near 2 m above ground from 700 s after the LES initialization, 43 s before the 'Local' LES blow-up. Compared are DNS, coarse-grained DNS, and various LESs. The 'Global' and 'Local' LESs use DNNs with scaling based on global or local physics, respectively, and trained on the same dataset, one designed to test interpolation to unseen Reynolds numbers.

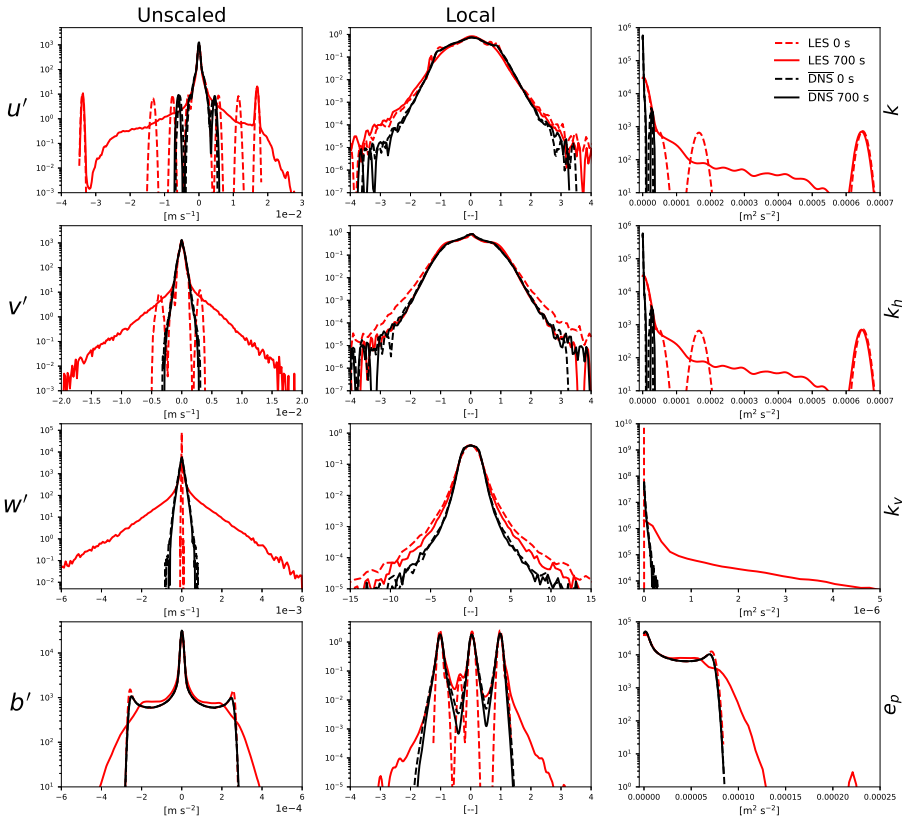

Figure 24. Distributions of input variables, velocity and buoyancy, both in raw, dimensional form (left column) as well as after the local physics normalization (middle column) by the turbulent energy based scaling factors (right column). We show both the time of LES initialization and at the latest time which we have DNS for comparison to the 'Local' LES that blows up at 743 s after initialization.

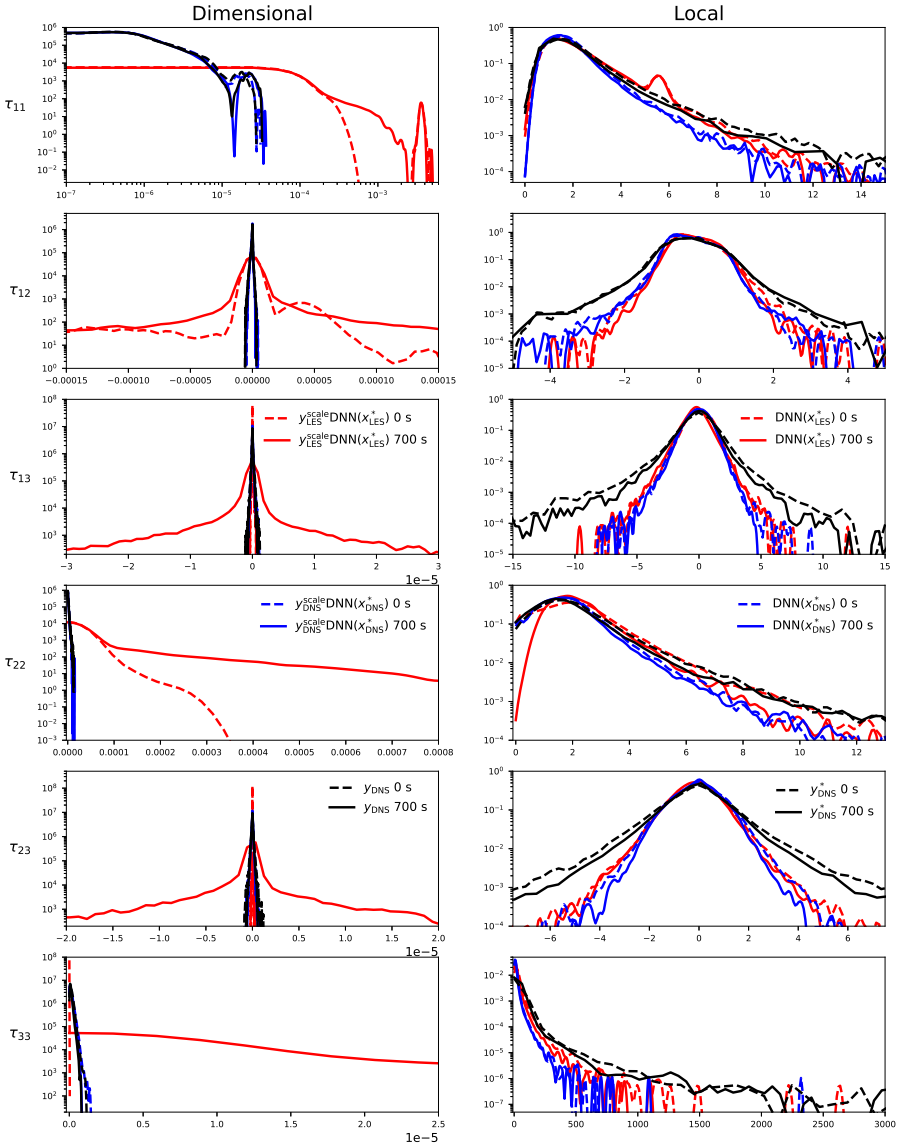

Figure 25. Distributions of subfilter-sclae stresses, both upon dimensionalization (left column) by the local scaling factors, as well as the raw, dimensionless outputs of the DNN (right column). We show the output of the DNN with LES fields as inputs (red), the values actually used online in the 'Local' LES that blows up at 743 s after initialization. We also show the output of the DNN with coarse-grained DNS inputs (blue), the predicted values used during offline testing, as well as the ground truth values used during offline testing (black).

5.4.2 Distributions

Distributions of the turbulent energy based scaling factors (figure 24, right column) at initialization and the final model time step confirm that, while there is disagreement with the DNS data due to initialization, these discrepancies are exacerbated by the LES over the course of the simulation. This is particularly notable in the vertical velocity contributions to turbulent kinetic energy. That part is proportional to k_v which is actually underestimated by the initialization, but is soon an order of magnitude larger than those of the coarse-grained DNS. These errors in scaling factors are reflected in the dimensional variables (figures 24 and 25, left columns) but the variables with local physics nondimensionalization are nonetheless surprisingly well normalized. So, while the predictions of the DNN may very well be accurate for another flow with much higher energy than ours, it is regions of the LES domain drifting into such a regime that is unphysical and ultimately leads to model failure.

5.4.3 Grid-to-subgrid transfer

One reason the energy content of the flow may be ever increasing when using the resolved turbulence based scaling for subfilter-stress is to do with grid to subgrid-scale transfer of energy. The total SGS transfer due to the subfilter-scale stress term is

$$\tau_{ij}^{\mathrm{dev}} S_{ij} \tag{45}$$

where S_{ij} is the strain rate, $S_{ij} = \frac{1}{2}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)$, and we only consider the deviatoric subfilter-stress, $\tau_{ij}^{\mathrm{dev}} = \tau_{ij} - \frac{1}{3}\delta_{ij}\tau_{kk}$, for our treatment of modified pressure. Note, the formulation of the Smagorinsky closure (eqns. 4 – 6) ensure that SGS transfer is always negative. Energy goes from grid to subgrid scales in a forward cascade of energy because S_{ij} and $-\tau_{ij}^{\mathrm{dev}}$ are always aligned with the Smagorinsky model. However, inspection of the coarse-grained DNS reveal that backscatter, energy transfer from subgrid to grid scale, does occur and is indeed not rare (figure 26). The DNN allows for this backscatter in the LES and prediction of backscatter at any given point is not itself an issue. However, in the current implementation, an unphysical feedback loop may arise during the time evolution.

Around halfway between initialization and the time of model blow-up, the SGS transfer in the LES compares quite well to that of the coarse-grained DNS (figure 26, top row). At this time, 400 s, both the range of the energy transfer and the spatial scales which characterize regions of dissipation and backscatter are similar in the LES and the coarse-grained DNS. Later, at 700 s (middle row), the coarse-grained DNS retains both the range and characteristic spatial scale. The LES continues to have similar spatial scales in most of the domain but the range of SGS transfer has grown considerably, particularly within comparatively small, local regions throughout the domain. At the height shown in the contour, one such region of exaggerated range is seen in the top left of the domain. Later, this region of exaggerated range, which contained both strong backscatter and strong dissipation, shows only backscatter with yet higher magnitude just before blow-up. We see that this region of backscatter does in fact cause too wide of a range of velocity (figure 26, bottom row). Evidently, within a similarly small locality of the flow, the range becomes too large to support with the floating point data type causing model failure.

From conservation of energy, backscatter should drain the stock the of subgrid turbulent kinetic energy. Our formulation does not consider this conservation, and its violation may lead to a positive feedback which ultimately results in numerical instability. To understand the feedback loop, we first recall that τ_{ij} represents the flux of i-direction momentum in the j-direction. Also note that S_{ij} is defined by velocity gradients so is associated with the direction in which velocities are increasing. As such, when $\tau_{ij}S_{ij}$ is positive, the subfilter-scale stress provides a flux of momentum in the directions in which the momentum is already increasing. This further increases the magnitude of the veloc-

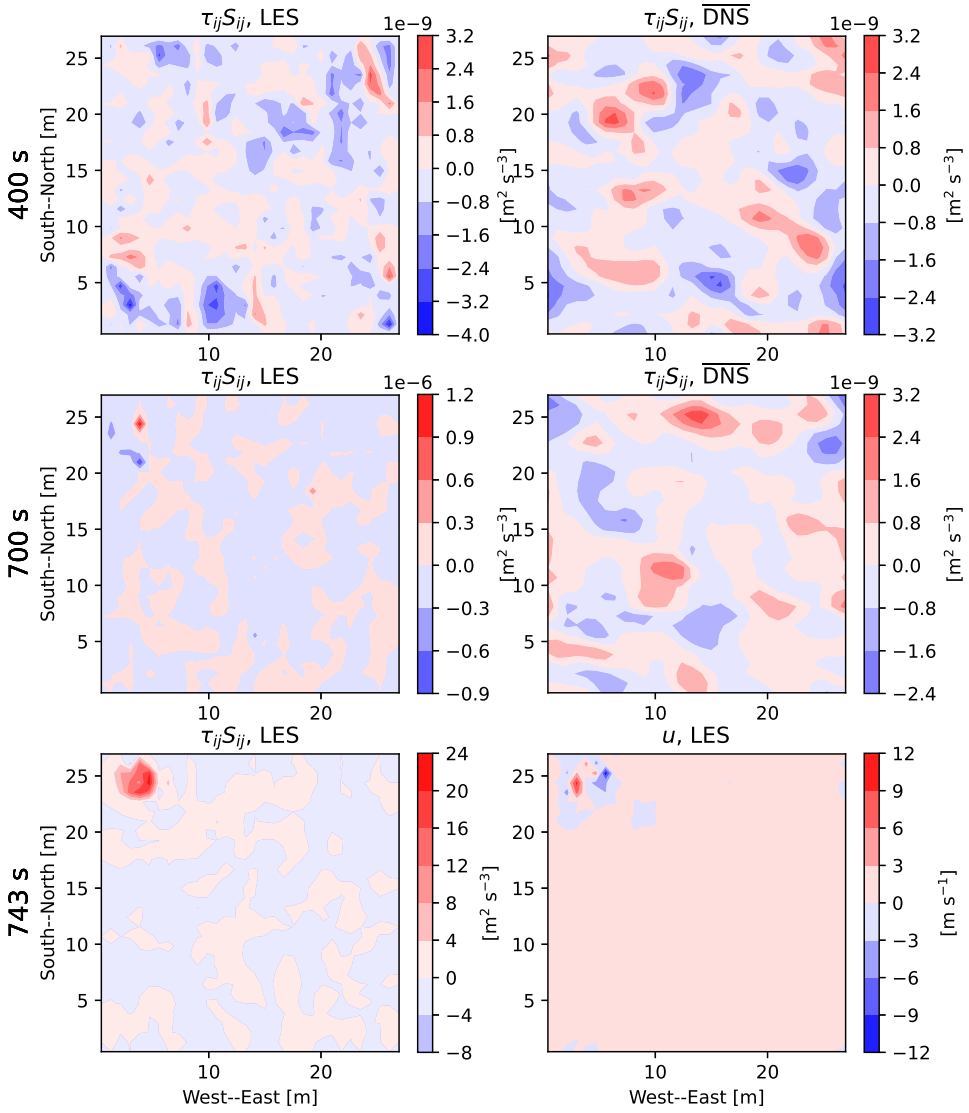

Figure 26. Contours of the SGS transfer of energy from both the LES (left column) and the coarse-grained DNS (right column, top two rows) both long before the model blowup (top) and at the last time which we have DNS data before blowup (middle). The SGS transfer is also shown for the last time step before the LES blow-up (bottom left) along with zonal velocity contours at this final time.

ity gradients. In our formulation, the subgrid turbulent kinetic energy is estimated from these resolved gradients. As such, backscatter increases subsequent estimates of subgrid energy, by increasing the magnitude of velocity gradients, when we know that backscatter should reduce this energy from conservation principles.

5.4.4 Online training for numerical stability

One way to rectify this mismatch between physics and the current local scaling approach is to explicitly prognosticate the stores of subgrid turbulent energy on which the local scaling factors are based. This methodology is common in fluid dynamics, tracing its origins to the earliest LES of the atmospheric boundary layer (Deardorff, 1980), but is beyond the scope of this paper due to machine learning challenges. The difficulty arises because the evolution equations for subgrid turbulent energy involve the subfilter-scale stress itself. Thus, capturing the feedback between the neural network and its scaling factors can only be done in the setting of the simulation itself. This is often called online learning to distinguish the approach from offline learning, during which the model components are learned from time snapshots as they were done in the current work. Development of online training procedures are much more difficult than offline, but first steps have been made in the current work by compiling the machine learning library, libtorch, with the fluids code, microHH. Further development of online learning methods for deep learning turbulence closures based on local and physical scaling factors should be encouraged by the performance of turbulent energy based scaling in the *a priori* tests shown.

6 Discussion

Though our focus has been on generalizability, we do not claim a deep learning model that can be used across all flow regimes found in the stable atmospheric boundary layer. To create such a model, a necessary first step would be to create a large dataset of direct numerical simulation (DNS) data from which to train. In contrast, we have not attempted to train any single deep neural network (DNN) with all the DNS data that we have currently available. Instead, we have deliberately excluded a significant portion of data from training, utilizing various train–test splits, to conduct controlled numerical experiments related to data normalization and network architecture. The creation of a DNS dataset from which a truly general DNN model can be trained is a monumental research task in and of itself. We hope that the results presented here will encourage the community to contribute to this effort.

A benefit to conducting such controlled numerical experiments is comparing offline skill and online performance. It is understood by both the community studying machine learning for dynamical systems and the field of turbulence modeling, that high statistical scores in offline test are not sufficient for good online performance. The most dramatic example of this presented here is the case of the local scaling approach. The local scaling most often led to highest offline statistical scores, however it always led to unstable simulations when implemented online. Another example involves the statistical scaling failing during online test for ability to interpolate across Reynolds number, again despite high offline scores. Finally, even when the models do not lead to numerical instability there are limitations that cannot be resolved by machine learning, no matter how high offline scores may be achieved. During the resolution extrapolation tests, we observed such a failure as no coarse large-eddy simulation (LES) could resolve the near-surface jet profile. This limitation, due to the natively coarse LES grid, is not apparent in the offline comparison to synthetically coarse-grained DNS data. Modelers must therefore continue to be mindful of the resolution requirements for the physical phenomena they seek to model, even when employing sophisticated, deep learning parameterizations for subgrid effects. These cautionary tales, important as they are, should not distract

from the successes of the DNN, which should encourage continued pursuit of deep learning approaches to parameterization.

A result we find particularly encouraging is establishing that deep learning parameterizations for subfilter-scale stress can be used in LESs of the stably stratified atmospheric boundary layer which remain numerically stable. We have also shown that use of physical scaling is preferred over statistical scaling, providing hope for formulating more generalizable deep learning models. The improved generalizability can partly be explained by the distribution of scaled variables, which can be made more similar through physical nondimensionalization. However, the scaling based on local physics that best collapses these distributions leads to numerically unstable simulations despite yielding the highest offline scores. It is not yet clear if there will be an elegant solution to this issue, or if such endogenous scaling factors are doomed to produce numerical instability. Along with the production of more high quality training data, investigating issues relating to numerical stability is encouraged for future work.

7 Conclusion

We have tested deep neural network (DNN) models which ingest grid-scale variables, velocity and buoyancy, and predict the subfilter-scale stress for large-eddy simulation (LES) of the stably stratified atmospheric boundary layer. Our primary focus has been generalizability, a major challenge in deep learning. In the current context, generalizability means the ability to make accurate predictions for different flow regimes and simulation configurations. The ability of the DNN to generalize is tested in scenarios which differ from training data in three main ways: Reynolds number, grid resolution, and orientation relative to the direction of the forcing winds. Conventional machine learning practices, data augmentation and statistics-based scaling, are both outperformed by the methods proposed here, enforcing strict equivariance and utilizing physics-based scaling. Further, replacing the conventional Smagorinsky model with a DNN turbulence closure leads to improved LESs which better resolve the sharp velocity gradients of a near-surface jet and maintain resolved turbulence despite the strongly stable density stratification.

During *a posteriori* evaluation, a global physics nondimensionalization based on the imposed forcing was most successful in online tests of generalizability, yielding both stable and accurate simulations. Normalization based on statistics of the training dataset occasionally shared this success, but more often failed. Surprisingly, these failures include an interpolation task, in which the Reynolds number of the test scenario was bounded by those in the training data, for which we hypothesized a statistical approach would suffice. In different tests, the mode of failure of LES using a statistical scaling DNN turbulence closure varied. Occasionally the spectra revealed issues (figure 16, red line with 'x' hashing), and elsewhere the failure was more apparent in the mean profiles (figure 21). This contrasts with LESs using the Smagorinsky model, which always failed in the same way: inability to maintain resolved turbulence and over-mixing of sharp gradients at the levels of the near-surface jets. The LES can generally resolve these near-surface jets with DNN turbulence closures using the global physics scaling. An exception occurs when the LES resolutions are too coarse to resolve these low level jets regardless of choice in subfilter model. When this was not a limitation, for the LES on the intermediate resolution grid as well as during offline testing, we were surprised to find that DNN models without any explicit information of the grid spacing were able to generalize to unseen resolutions. Presumably this is due to the inherent scale-similarity of turbulence resolved on LES grids with spacing appropriately selected to lie within the inertial range. Further, when using a physics-based nondimensionalization for the DNN turbulence closure, the LESs consistently maintain appropriate structure and intensity of turbulence in addition to better resolving these jets, even for flow with Reynolds number greater than any in the training data. These simulations support the claim that deep learning

parameterization can improve LES of the stable boundary layer, provided enough attention is paid to the underlying physics during model development.

For *a priori* comparisons, a local physics scaling relationship led to the overall best model performance in offline tests of generalizability. The local scaling factors are based on estimates of the turbulent kinetic energy and turbulent potential energy at a given grid point. The superiority of the local scaling approach is foreshadowed by its unique ability to collapse the distributions of input and output variables across a range of Reynolds numbers and grid resolutions (figures 4 – 7) and borne out in the statistics of offline predictions (figures 8 – 9, tables A1 – A4). However, simulations with deep learning parameterization using this local scaling are numerically unstable. We trace this to an unphysical feedback between the backscatter of energy from subgrid to grid scales and subsequent estimates for subgrid turbulent kinetic energy. Future work may avoid this by utilizing prognostic equations for the subgrid turbulent energy quantities used as scaling factors, provided an adequate online training procedure could be developed.

We used a model architecture which is strictly equivariant to rotations in the horizontal plane but allows for vertical anisotropy. The subfilter-scale stress tensor, a rank-2 tensor, has irreducible representations for rotations, formally transformations in SO(2) and C_N groups, that are not as well known as those for vectors and scalars. We provide here a general formulation to enforce the SO(2)- and C_N-equivariance through a change of basis of the components of the subfilter-scale stress tensor. However, for practical considerations related to interpolating rotations on discrete grids, we have focused mostly on enforcing only C_4-equivariance. This corresponds to rotations by multiples of 90° and leads to better predictions than models enforcing equivariance to a higher order cyclic groups corresponding to multiples of finer angles of rotation (figure 13, table A7). Our approach improves predictions compared to those that do not strictly enforce equivariance but approximate it through data augmentation, *i.e.* training with manually rotated samples and labels (figures 11 & 10, table A5). By using this model architecture, our deep learning parameterization generalizes to different wind directions without the effort of modifying the training data in these ways.

Handling these modes of generalizability, to Reynolds number, grid resolution, and orientation, are important prerequisites for turbulence closures in Earth system models. A scale-aware parameterization is ideal considering the impracticality of retraining a deep learning model at the resolution of each new use case. Similarly important is generalization across Reynolds number and direction of the forcing winds, which are emergent properties of atmospheric dynamics. A good parameterization should handle any condition that may emerge at any resolution a user may appropriately select. The novel architecture and nondimensionalization procedures presented here have proven to be viable approaches for deep learning turbulence models to achieve this generalizability.

8 Acknowledgments

The authors wish to express deep gratitude to Pavel Perezhogin for his useful suggestion to pursue a local scaling based on first order estimates of turbulent kinetic energy.

Alex Connolly and Pierre Gentine received M^{2}LInES research support through Schmidt Sciences, LLC. Additional support was provided by the National Science Foundation Science and Technology Center (STC) Learning the Earth with Artificial intelligence and Physics (LEAP), Award # 2019625 - STC, from European Research council grant USMILE and U.S. Department Of Energy, Office of Science grant #DE-SCO022323. Robin Walters is supported by NSF grants #2107256 and #2134178. Participation of Rui Wang and Rose Yu was supported in part by U.S. Department Of Energy, Office of Science,

Facebook Data Science Research Awards, U. S. Army Research Office under Grant W911NF-20-1-0334, and NSF Grants #2134274 and #2146343.

We acknowledge computing resources from Columbia University's Shared Research Computing Facility project, which is supported by NIH Research Facility Improvement Grant 1G20RR030893-01, and associated funds from the New York State Empire State Development, Division of Science Technology and Innovation (NYSTAR) Contract C090171, both awarded April 15, 2010. We would like to acknowledge high-performance computing support from the Derecho system (doi:10.5065/qx9a-pg09) and the Casper system (https://ncar.pub/casper), both provided by the NSF National Center for Atmospheric Research (NCAR), sponsored by the National Science Foundation.

Declaration of Interests. The authors report no conflict of interest.

9 Open Research

The repository at `https://github.com/adconnolly/SFSstressDNN` contains the software to coarse-grain the DNS data, train and JIT trace the DNN, perform *a priori* analysis, and generate accompanying plots as well as the associated job output files. The fork of the microHH model at `https://github.com/adconnolly/microHH` is modified to perform the online test. The source, LES configuration, and analysis files can be found there along with the JIT trace version of the DNNs used. The data including the DNS and LES output; the coarse-grained DNS data; and the weights of the DNNs exist on the Globally Accessible Data Environment (GLADE) file system of the National Center for Atmospheric Research (NCAR) and can be made available upon request through the Globus service.

References

Bakarji, J., Callaham, J., Brunton, S. L., & Kutz, J. N. (2022). Dimensionally consistent learning with buckingham pi. *Nature Computational Science*, 1–11.

Bardina, J., Ferziger, J., & Reynolds, W. (1980). Improved subgrid-scale models for large-eddy simulation. In *13th fluid and plasmadynamics conference* (p. 1357).

Beare, R. J., Macvean, M. K., Holtslag, A. A., Cuxart, J., Esau, I., Golaz, J.-C., ... others (2006). An intercomparison of large-eddy simulations of the stable boundary layer. *Boundary-Layer Meteorology*, *118*, 247–272.

Bedford, K., & Yeo, W. (1993). Conjunctive filtering procedures in surface water flow and transport. *Large eddy simulation of complex engineering and geophysical flows*, 513–537.

Beucler, T., Pritchard, M., Gentine, P., & Rasp, S. (2020). Towards physically-consistent, data-driven models of convection. In *Igarss 2020-2020 ieee international geoscience and remote sensing symposium* (pp. 3987–3990).

Beucler, T., Pritchard, M., Rasp, S., Ott, J., Baldi, P., & Gentine, P. (2021). Enforcing analytic constraints in neural networks emulating physical systems. *Physical Review Letters*, *126*(9), 098302.

Bogenschutz, P. A., Gettelman, A., Morrison, H., Larson, V. E., Craig, C., & Schanen, D. P. (2013). Higher-order turbulence closure and its impact on climate simulations in the community atmosphere model. *Journal of Climate*, *26*(23), 9655–9676.

Buckingham, E. (1914). On physically similar systems; illustrations of the use of dimensional equations. *Physical review*, *4*(4), 345.

Cheng, Y., Giometto, M. G., Kauffmann, P., Lin, L., Cao, C., Zupnick, C., ... others (2022). Deep learning for subgrid-scale turbulence modeling in large-eddy simulations of the convective atmospheric boundary layer. *Journal of Advances in Modeling Earth Systems*, *14*(5), e2021MS002847.

Cheng, Y., Grachev, A., & van Heerwaarden, C. (2023). Logarithmic profiles of velocity in stably stratified atmospheric boundary layers. *Physical Review Fluids*, *8*(11), 114602.

Chinita, M. J., Matheou, G., & Miranda, P. M. (2022). Large-eddy simulation of very stable boundary layers. part i: Modeling methodology. *Quarterly Journal of the Royal Meteorological Society*, *148*(745), 1805–1823.

Chow, F. K., & Moin, P. (2003). A further study of numerical errors in large-eddy simulations. *Journal of Computational Physics*, *184*(2), 366–380.

Chow, F. K., Street, R. L., Xue, M., & Ferziger, J. H. (2005). Explicit filtering and reconstruction turbulence modeling for large-eddy simulation of neutral boundary layer flow. *Journal of the atmospheric sciences*, *62*(7), 2058–2077.

Clark, R. A., Ferziger, J. H., & Reynolds, W. C. (1979). Evaluation of subgrid-scale models using an accurately simulated turbulent flow. *Journal of fluid mechanics*, *91*(1), 1–16.

Cohen, T., & Welling, M. (2016). Group equivariant convolutional networks. In *International conference on machine learning* (pp. 2990–2999).

Cohen, T. S., Geiger, M., & Weiler, M. (2019). A general theory of equivariant cnns on homogeneous spaces. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc. Retrieved from `https://proceedings.neurips.cc/paper_files/paper/2019/file/b9cfe8b6042cf759dc4c0cccb27a6737-Paper.pdf`

Cohen, T. S., & Welling, M. (2016). Steerable CNNs. *arXiv preprint arXiv:1612.08498*.

Connolly, A., Chow, F. K., & Hoch, S. W. (2021). Nested large-eddy simulations of the displacement of a cold-air pool by lee vortices. *Boundary-Layer Meteorology*, *178*(1), 91–118.

Connolly, A., van Veen, L., Neher, J., Geurts, B. J., Mirocha, J., & Chow, F. K. (2020). Efficacy of the cell perturbation method in large-eddy simulations of boundary layer flow over complex terrain. *Atmosphere*, *12*(1), 55.

Couvreux, F., Bazile, E., Rodier, Q., Maronga, B., Matheou, G., Chinita, M. J., ... others (2020). Intercomparison of large-eddy simulations of the antarctic boundary layer for very stable stratification. *Boundary-Layer Meteorology*, *176*, 369–400.

Deardorff, J. W. (1970). A numerical study of three-dimensional turbulent channel flow at large reynolds numbers. *Journal of Fluid Mechanics*, *41*(2), 453–480.

Deardorff, J. W. (1980). Stratocumulus-capped mixed layers derived from a three-dimensional model. *Boundary-layer meteorology*, *18*, 495–527.

Diaconu, N., & Worrall, D. (2019). Learning to convolve: A generalized weight-tying approach. In *International conference on machine learning* (pp. 1586–1595).

Frezat, H., Balarac, G., Le Sommer, J., Fablet, R., & Lguensat, R. (2021, Feb). Physical invariance in neural networks for subgrid-scale scalar flux modeling. *Phys. Rev. Fluids*, *6*, 024607. Retrieved from `https://link.aps.org/doi/10.1103/PhysRevFluids.6.024607` doi: 10.1103/PhysRevFluids.6.024607

Fukami, K., & Taira, K. (2021). Robust machine learning of turbulence through generalized buckingham pi-inspired pre-processing of training data. In *Aps division of fluid dynamics meeting abstracts* (pp. A31–004).

Germano, M., Piomelli, U., Moin, P., & Cabot, W. H. (1991). A dynamic subgrid-scale eddy viscosity model. *Physics of Fluids A: Fluid Dynamics*, *3*(7), 1760–1765.

Glorot, X., & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics* (pp. 249–256).

Guan, Y., Subel, A., Chattopadhyay, A., & Hassanzadeh, P. (2023). Learning physics-constrained subgrid-scale closures in the small-data regime for sta-

ble and accurate les. *Physica D: Nonlinear Phenomena*, *443*, 133568. Retrieved from https://www.sciencedirect.com/science/article/pii/S016727892200272X doi: https://doi.org/10.1016/j.physd.2022.133568

Heerwaarden, C. C. v., Van Stratum, B. J., Heus, T., Gibbs, J. A., Fedorovich, E., & Mellado, J. P. (2017). Microhh 1.0: a computational fluid dynamics code for direct numerical simulation and large-eddy simulation of atmospheric boundary layer flows. *Geosci. Model Dev.*, *10*(8), 3145–3165.

Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural networks*, *2*(5), 359–366.

Kaandorp, M. L., & Dwight, R. P. (2020). Data-driven modelling of the reynolds stress tensor using random forests with invariance. *Computers & Fluids*, *202*, 104497. Retrieved from https://www.sciencedirect.com/science/article/pii/S0045793020300700 doi: https://doi.org/10.1016/j.compfluid.2020.104497

Katul, G. G., Porporato, A., Shah, S., & Bou-Zeid, E. (2014). Two phenomenological constants explain similarity laws in stably stratified turbulence. *Physical Review E*, *89*(2), 023007.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Lang, S. (2012). *Algebra* (Vol. 211). Springer Science & Business Media.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *nature*, *521*(7553), 436–444.

LeCun, Y., Bottou, L., Orr, G. B., & Müller, K.-R. (2002). Efficient backprop. In *Neural networks: Tricks of the trade* (pp. 9–50). Springer.

Li, Q., Bou-Zeid, E., Anderson, W., Grimmond, S., & Hultmark, M. (2016). Quality and reliability of les of convective scalar transfer at high reynolds numbers. *International Journal of Heat and Mass Transfer*, *102*, 959–970.

Lilly, D. K. (1962). On the numerical simulation of buoyant convection. *Tellus*, *14*(2), 148–172.

Ling, J., Kurzawski, A., & Templeton, J. (2016). Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, *807*, 155–166.

Lu, H., & Porté-Agel, F. (2011). Large-eddy simulation of a very large wind farm in a stable atmospheric boundary layer. *Physics of Fluids*, *23*(6), 065101.

Mason, P. J., & Thomson, D. J. (1992). Stochastic backscatter in large-eddy simulations of boundary layers. *Journal of Fluid Mechanics*, *242*, 51–78.

Mellor, G. L., & Yamada, T. (1974). A hierarchy of turbulence closure models for planetary boundary layers. *Journal of the atmospheric sciences*, *31*(7), 1791–1806.

Moody, L. F. (1944). Friction factors for pipe flow. *Transactions of the American Society of Mechanical Engineers*, *66*(8), 671–678.

Nieuwstadt, F. T., Mason, P. J., Moeng, C.-H., & Schumann, U. (1993). Large-eddy simulation of the convective boundary layer: A comparison of four computer codes. In *Turbulent shear flows 8* (pp. 343–367). Springer.

Obiols-Sales, O., Vishnu, A., Malaya, N., & Chandramowliswharan, A. (2020). Cfdnet: A deep learning-based accelerator for fluid simulations. In *Proceedings of the 34th acm international conference on supercomputing* (pp. 1–12).

Oppenheimer, M. W., Doman, D. B., & Merrick, J. D. (2023). Multi-scale physics-informed machine learning using the buckingham pi theorem. *Journal of Computational Physics*, *474*, 111810.

Pope, S. B. (2000). *Turbulent flows*. Cambridge university press.

Prakash, A., Jansen, K. E., & Evans, J. A. (2022). Invariant data-driven subgrid stress modeling in the strain-rate eigenframe for large eddy simulation. *Computer Methods in Applied Mechanics and Engineering*, *399*, 115457.

Reynolds, O. (1895). On the dynamical theory of incompressible viscous fluids and

the determination of the criterion. *Philosophical transactions of the royal society of london.(a.)*(186), 123–164.

Simon, J. S., & Chow, F. K. (2021). Alternative anisotropic formulations for eddy-viscosity models in the weather research and forecasting model. *Boundary-Layer Meteorology*, *181*(1), 11–37.

Skamarock, W. C. (2004). Evaluating mesoscale nwp models using kinetic energy spectra. *Monthly weather review*, *132*(12), 3019–3032.

Smagorinsky, J. (1963). General circulation experiments with the primitive equations: I. the basic experiment. *Monthly Weather Review*, *91*(3), 99 - 164. Retrieved from `https://journals.ametsoc.org/view/journals/mwre/91/3/1520-0493_1963_091_0099_gcewtp_2_3_co_2.xml` doi: https://doi.org/10.1175/1520-0493(1963)091⟨0099:GCEWTP⟩2.3.CO;2

Smith, C. M., & Porté-Agel, F. (2014). An intercomparison of subgrid models for large-eddy simulation of katabatic flows. *Quarterly Journal of the Royal Meteorological Society*, *140*(681), 1294–1303.

Stoffer, R., Van Leeuwen, C. M., Podareanu, D., Codreanu, V., Veerman, M. A., Janssens, M., ... Van Heerwaarden, C. C. (2021). Development of a large-eddy simulation subgrid model based on artificial neural networks: a case study of turbulent channel flow. *Geoscientific Model Development*, *14*(6), 3769–3788.

Thorpe, S. (2004). Recent developments in the study of ocean turbulence. *Annu. Rev. Earth Planet. Sci.*, *32*, 91–109.

Thuerey, N., Weißenow, K., Prantl, L., & Hu, X. (2020). Deep learning methods for reynolds-averaged navier–stokes simulations of airfoil flows. *AIAA Journal*, *58*(1), 25–36.

Vreman, B., Geurts, B., & Kuerten, H. (1995). A priori tests of large eddy simulation of the compressible plane mixing layer. *Journal of engineering mathematics*, *29*(4), 299–327.

Wang, R., Walters, R., & Yu, R. (2020). Incorporating symmetry into deep dynamics models for improved generalization. *arXiv preprint arXiv:2002.03061*.

Wang, R., Walters, R., & Yu, R. (2022a). Approximately equivariant networks for imperfectly symmetric dynamics. In *International conference on machine learning* (pp. 23078–23091).

Wang, R., Walters, R., & Yu, R. (2022b). Data augmentation vs. equivariant networks: A theory of generalization on dynamics forecasting. *arXiv preprint arXiv:2206.09450*.

Weiler, M., & Cesa, G. (2019a). General $e(2)$-equivariant steerable cnns. *arXiv.org*. Retrieved from `https://arxiv.org/abs/1911.08251` doi: https://doi.org/10.48550/arXiv.1911.08251

Weiler, M., & Cesa, G. (2019b). General E(2)-equivariant steerable CNNs. In *Advances in neural information processing systems (neurips)* (pp. 14334–14345).

Weiler, M., Hamprecht, F. A., & Storath, M. (2018). Learning steerable filters for rotation equivariant cnns. In *Proceedings of the ieee conference on computer vision and pattern recognition* (pp. 849–858).

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2022). Integrating scientific knowledge with machine learning for engineering and environmental systems. *ACM Computing Surveys*, *55*(4), 1–37.

Wu, J.-L., Xiao, H., & Paterson, E. (2018, Jul). Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Phys. Rev. Fluids*, *3*, 074602. Retrieved from `https://link.aps.org/doi/10.1103/PhysRevFluids.3.074602` doi: 10.1103/PhysRevFluids.3.074602

Wyngaard, J. C. (2004). Toward numerical modeling in the "terra incognita". *Journal of the atmospheric sciences*, *61*(14), 1816–1826.

Wyngaard, J. C. (2010). *Turbulence in the atmosphere*. Cambridge University Press.

Zhang, C., Bengio, S., Hardt, M., Recht, B., & Vinyals, O. (2021). Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, *64*(3), 107–115.

Zhou, B., & Chow, F. K. (2011). Large-eddy simulation of the stable boundary layer with explicit filtering and reconstruction turbulence modeling. *Journal of the Atmospheric Sciences*, *68*(9), 2142–2155.

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.

Zilitinkevich, S. S., Elperin, T., Kleeorin, N., Rogachevskii, I., Esau, I., Mauritsen, T., & Miles, M. (2008). Turbulence energetics in stably stratified geophysical flows: Strong and weak mixing regimes. *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, *134*(633), 793–799.

Appendix A Additional Tables

Reynolds Number Interpolation						
	Statistical	Global	Local	Bardina	Clark	Smagorinsky
τ_{11}	0.88 ± 0.013	0.93 ± 0.011	0.97 ± 0.0004	-0.51	0.53	-0.11
τ_{12}	0.69 ± 0.008	0.72 ± 0.019	0.79 ± 0.003	0.55	0.14	-0.17
τ_{13}	0.66 ± 0.008	0.66 ± 0.012	0.77 ± 0.003	0.46	0.67	-0.00
τ_{22}	0.71 ± 0.013	0.61 ± 0.024	0.81 ± 0.004	0.20	0.29	-0.00
τ_{23}	0.59 ± 0.007	0.55 ± 0.027	0.69 ± 0.002	0.41	0.58	-0.01
τ_{33}	0.06 ± 0.091	-0.19 ± 0.16	0.51 ± 0.022	-0.76	-0.57	-0.42

Table A1. Numerical presentation of the left panel of figure 8 plus coefficient of determination, R^2, statistics for the predictions made by the conventional turbulence closures, Bardina, Clark and Smagorinsky, in the same test as the DNN closures.

Reynolds Number Extrapolation						
	Statistical	Global	Local	Bardina	Clark	Smagorinsky
τ_{11}	0.42 ± 0.047	0.83 ± 0.006	0.94 ± 0.003	0.09	0.45	-0.17
τ_{12}	0.12 ± 0.032	0.61 ± 0.004	0.73 ± 0.003	0.47	0.35	-0.01
τ_{13}	0.17 ± 0.019	0.64 ± 0.012	0.75 ± 0.002	0.48	0.64	-0.01
τ_{22}	0.40 ± 0.027	0.47 ± 0.013	0.80 ± 0.001	0.01	0.14	-0.02
τ_{23}	0.20 ± 0.014	0.58 ± 0.007	0.71 ± 0.002	0.44	0.61	-0.01
τ_{33}	0.41 ± 0.022	0.39 ± 0.035	0.69 ± 0.010	-0.36	-0.24	-0.62

Table A2. As in table A1 but for the right panel of figure 8.

Resolution Interpolation						
	Statistical	Global	Local	Bardina	Clark	Smagorinsky
τ_{11}	0.83 ± 0.002	0.93 ± 0.010	0.95 ± 0.005	-0.51	0.53	-0.11
τ_{12}	0.67 ± 0.012	0.66 ± 0.009	0.65 ± 0.036	0.55	0.14	-0.17
τ_{13}	0.70 ± 0.003	0.71 ± 0.004	0.78 ± 0.002	0.46	0.67	-0.00
τ_{22}	0.63 ± 0.011	0.67 ± 0.016	0.74 ± 0.005	0.20	0.29	-0.00
τ_{23}	0.63 ± 0.007	0.63 ± 0.006	0.70 ± 0.002	0.41	0.58	-0.01
τ_{33}	0.41 ± 0.031	0.47 ± 0.039	0.49 ± 0.012	-0.76	-0.57	-0.42

Table A3. As in table A1 but for the left panel of figure 9.

Resolution Extrapolation						
	Statistical	Global	Local	Bardina	Clark	Smagorinsky
τ_{11}	0.83 ± 0.040	0.89 ± 0.020	0.82 ± 0.017	0.26	-0.25	-0.22
τ_{12}	0.53 ± 0.058	0.77 ± 0.040	0.91 ± 0.004	0.84	0.90	-0.38
τ_{13}	0.50 ± 0.022	0.54 ± 0.007	0.56 ± 0.007	0.26	0.46	-0.01
τ_{22}	0.68 ± 0.068	0.80 ± 0.037	0.84 ± 0.005	0.22	0.14	-0.19
τ_{23}	0.49 ± 0.019	0.52 ± 0.014	0.55 ± 0.003	0.25	0.44	-0.01
τ_{33}	0.65 ± 0.019	0.58 ± 0.041	0.36 ± 0.012	-1.8	-1.6	-0.96

Table A4. As in table A1 but for the right panel of figure 9.

	0°			270°		
	Baseline No Data Aug.	Baseline Data Aug.	Equivariant No Data Aug.	Baseline No Data Aug.	Baseline Data Aug.	Equivariant No Data Aug.
τ_{11}	0.96 ± 0.002	0.96 ± 0.001	0.97 ± 0.001	-4.2 ± 0.866	0.78 ± 0.006	0.80 ± 0.004
τ_{12}	0.76 ± 0.003	0.75 ± 0.005	0.78 ± 0.002	-0.15 ± 0.400	0.75 ± 0.008	0.78 ± 0.002
τ_{13}	0.73 ± 0.006	0.75 ± 0.003	0.77 ± 0.004	0.55 ± 0.012	0.68 ± 0.003	0.68 ± 0.002
τ_{22}	0.78 ± 0.002	0.76 ± 0.015	0.80 ± 0.004	-0.95 ± 0.232	0.96 ± 0.002	0.97 ± 0.001
τ_{23}	0.66 ± 0.015	0.68 ± 0.003	0.68 ± 0.002	0.62 ± 0.029	0.75 ± 0.004	0.77 ± 0.004
τ_{33}	0.39 ± 0.004	0.47 ± 0.040	0.51 ± 0.017	0.31 ± 0.029	0.47 ± 0.015	0.51 ± 0.017

Table A5. Numerical presentation of figure 11.

	Fine		Medium		Coarse	
	Buoyancy	Noise	Buoyancy	Noise	Buoyancy	Noise
τ_{11}	0.985 ± 0.004	0.981 ± 0.001	0.934 ± 0.003	0.930 ± 0.003	0.85 ± 0.009	0.80 ± 0.017
τ_{12}	0.95 ± 0.012	0.93 ± 0.001	0.71 ± 0.006	0.69 ± 0.006	0.66 ± 0.007	0.66 ± 0.012
τ_{13}	0.88 ± 0.027	0.85 ± 0.003	0.745 ± 0.005	0.739 ± 0.006	0.49 ± 0.016	0.47 ± 0.016
τ_{22}	0.90 ± 0.023	0.88 ± 0.003	0.79 ± 0.003	0.78 ± 0.003	0.59 ± 0.003	0.56 ± 0.019
τ_{23}	0.83 ± 0.035	0.80 ± 0.004	0.70 ± 0.003	0.68 ± 0.004	0.34 ± 0.034	0.35 ± 0.035
τ_{33}	0.79 ± 0.048	0.75 ± 0.007	0.68 ± 0.005	0.66 ± 0.017	0.39 ± 0.033	0.44 ± 0.017

Table A6. Numerical presentation of figure 12. The greater mean R^2 values are highlighted gray only if the difference is statistically significant ($p < 0.05$).

	Equivariance group and horizontal width of input box			
	C_4 3×3	C_8 3×3	C_4 5×5	C_8 5×5
τ_{11}	0.97 ± 0.001	0.94 ± 0.003	0.96 ± 0.003	0.95 ± 0.0004
τ_{12}	0.78 ± 0.002	0.29 ± 0.026	0.77 ± 0.004	0.31 ± 0.021
τ_{13}	0.77 ± 0.004	0.76 ± 0.003	0.77 ± 0.002	0.77 ± 0.003
τ_{22}	0.80 ± 0.004	0.46 ± 0.033	0.79 ± 0.009	0.54 ± 0.008
τ_{23}	0.68 ± 0.002	0.68 ± 0.002	0.70 ± 0.001	0.69 ± 0.004
τ_{33}	0.51 ± 0.017	0.49 ± 0.018	0.51 ± 0.010	0.54 ± 0.013
Average	0.75 ± 0.002	0.60 ± 0.008	0.75 ± 0.002	0.63 ± 0.004

Table A7. Numerical presentation of figure 13.

Acronyms

DNN Deep neural network

DNS Direct numerical simulation

LES Large-eddy simulation