

1 **Benchmarking of machine learning ocean subgrid**
2 **parameterizations in an idealized model**

3 **Andrew Ross¹, Ziwei Li¹, Pavel Perezhogin¹, Carlos Fernandez-Granda^{1,2},**
4 **Laure Zanna¹**

5 ¹Courant Institute of Mathematical Sciences, New York University, New York, NY, USA

6 ²Center for Data Science, New York University, New York, NY, USA

7 **Key Points:**

- 8 • We develop 19 physical and climatological metrics to evaluate 148 subgrid closures
9 online in an open-source idealized ocean model.
10 • Neural network closures perform well online for some filtering operators under these
11 new metrics, but are not robust to distribution shift.
12 • We develop a new hybrid genetic programming and linear regression algorithm and
13 find a symbolic closure with more robust online performance.

Corresponding author: Laure Zanna, laure.zanna@nyu.edu

Abstract

Recently, a growing number of studies have used machine learning (ML) models to parameterize computationally intensive subgrid-scale processes in ocean models. Such studies typically train ML models with filtered and coarse-grained high-resolution data and evaluate their predictive performance offline, before implementing them in a coarse resolution model and assessing their online performance. In this work, we systematically benchmark the online performance of such models, their generalization to domains not encountered during training, and their sensitivity to dataset design choices. We apply this proposed framework to compare a large number of physical and neural network (NN)-based parameterizations. We find that the choice of filtering and coarse-graining operator is particularly critical and this choice should be guided by the application. We also show that all of our physics-constrained NNs are stable and perform well when implemented online, but generalize poorly to new regimes. To improve generalization and also interpretability, we propose a novel equation-discovery approach combining linear regression and genetic programming with spatial derivatives. We find this approach performs on par with neural networks on the training domain but generalizes better beyond it. We release code and data to reproduce our results and provide the research community with easy-to-use resources to develop and evaluate additional parameterizations.

Plain Language Summary

Accurately predicting climate change requires running intensive computer simulations called climate models. Climate models divide the world into grid cells, solving an approximation of continuous equations that model the true dynamics. For accurate predictions, these cells must be small, or equivalently models must be high-resolution. However, even with modern supercomputers, running many high-resolution simulations is prohibitively expensive.

One solution is to run climate models at coarser resolution, but include “subgrid parameterizations” to account for physical processes occurring at finer scales and correct bias. Parameterizations are usually developed by analyzing the continuous equations and empirically determining formulae to predict unresolved effects. However, recent studies have applied machine learning (ML) methods to learn parameterizations automatically from limited high-resolution data.

This approach has shown promise, but also introduced new challenges with dataset preparation, evaluation, interpretability, and implementation. We provide an open-source framework for learning and evaluating parameterizations in a simplified model of the ocean. We use this framework to evaluate numerous ML methods and analyze how best to prepare datasets. We also develop a method of learning equation-based parameterizations which can be more easily interpreted and implemented. Our approach performs comparably to the best ML parameterizations, but generalizes better to oceanic conditions unseen during training.

1 Introduction

Current state-of-the-art climate models solve geophysical fluid equations on horizontal grids of size 25km and coarser. Models at this resolution are not able to accurately and sufficiently resolve processes with physical length scales smaller than the model grid, for example, convection in the atmosphere and mesoscale eddies in the ocean. Since increases in computational power will likely not enable climate models to resolve these processes before the effects of climate change ensue (Fox-Kemper et al., 2014; Schneider et al., 2017), we must represent subgrid-scale (SGS) processes with closure models, also known as parameterizations. Yet, these SGS models are some of the largest sources of bias and uncertainties in climate simulations: e.g., insufficient representations of tran-

63 sient eddies cause biases in modeled currents and sea surface temperature in the ocean
 64 (Griffies et al., 2015; Hewitt et al., 2020), and the precipitation pattern is strongly sen-
 65 sitive to the different subgrid cloud closures, thereby causing significant errors in climate
 66 projections (Stevens & Bony, 2013). Therefore, developing robust parameterizations re-
 67 mains an important task towards reliable climate projections.

68 In ocean circulation models, stratified turbulent processes are one of the primary
 69 targets of SGS closures (Pope, 2000; Vallis, 2017). Classic SGS models are typically de-
 70 signed with specific goals in mind; for example, to dissipate small-scale enstrophy (Smagorinsky,
 71 1963), to reinject energy at larger scales via backscattering (Jansen & Held, 2014; Jansen
 72 et al., 2015), or to improve the representation of heat and tracer transport in the ocean
 73 interior (Redi, 1982; Gent & McWilliams, 1990; Gent et al., 1995). However, human choices
 74 in the design, formulation, and tuning of these SGS models sometimes lead to poor cor-
 75 relation between parameterized SGS forcing and true SGS forcing as diagnosed from high
 76 resolution simulations (Khani & Porté-Agel, 2017). This can result in unrealistic large-
 77 scale simulations despite recent progress in the representation of resolved processes (Griffies
 78 et al., 2009; Fox-Kemper et al., 2019). These shortcomings call for complementary, more
 79 systematic and data-driven approaches.

80 Recently, an increase in high-resolution observations and simulations combined with
 81 advances in machine-learning (ML) methods has propelled a surge in the development
 82 of data-driven SGS parameterizations in climate models (Krasnopolsky et al., 2010; Bolton
 83 & Zanna, 2019; Rasp et al., 2018; O’Gorman & Dwyer, 2018; Guillaumin & Zanna, 2021;
 84 Zanna & Bolton, 2021; Beucler et al., 2021; Guan et al., 2022; Yuval & O’Gorman, 2021;
 85 Frezat et al., 2022; Subel et al., 2022). Directly learning from data, ML methods auto-
 86 matically extract relevant information from observations and high-resolution simulations
 87 to improve coarse-resolution models at a reduced computational cost. Despite their uni-
 88 versal approximation properties (Hornik et al., 1989), popular ML models such as neu-
 89 ral networks are often opaque to interpretation and can extrapolate poorly to conditions
 90 unseen during training (Recht et al., 2018; O’Gorman & Dwyer, 2018; Bolton & Zanna,
 91 2019; Subel et al., 2022).

92 The performance of data-driven approaches is greatly influenced by choices that
 93 must be made in dataset preparation. The formulation of the subgrid forcing term, ei-
 94 ther in terms of tendency or subgrid-scale fluxes, can affect the stability of paramete-
 95 rized models (Yuval et al., 2021). Different filtering schemes also have significant effects
 96 on the online performance of subgrid parameterizations (Piomelli et al., 1988; Zhou et
 97 al., 2019; Frezat et al., 2022).

98 There is currently a vast number of possible choices in terms of ML models, train-
 99 ing target formulation, and filtering and coarse-graining methods. However, few stud-
 100 ies offer a direct and adequate comparison between data-driven ML methods and physical-
 101 based parameterizations. Moreover, well-defined quantitative (rather than qualitative)
 102 online metrics are lacking. In this paper, we introduce a family of datasets (Sections 2
 103 and 3) and quantitative metrics (Section 4) for learning and evaluating ocean eddy sub-
 104 grid parameterizations, both offline and online, using a quasi-geostrophic (QG) simula-
 105 tion (datasets and code are available open-source; see Appendix D). Our online metrics
 106 quantify to what extent the time-averaged spectral and distributional properties of pa-
 107 rameterized simulations match those of ground-truth high-resolution simulations, as well
 108 as whether they improve the accuracy of more predictable short-term dynamics. These
 109 metrics make it possible to comprehensively compare numerous parameterizations, and
 110 the effects of dataset design choices on their performance on the physics of the simula-
 111 tions (e.g., spectral properties), climate (e.g., distributional of variables such as PV), and
 112 weather (e.g., the evolution of short-term forecast).

113 In Section 5, we perform such a study for fully convolutional neural network pa-
 114 rameterizations, evaluating how offline and online performance change with different de-

115 signs of inputs (i.e. types of feature variables – velocity or potential vorticity) and out-
 116 puts (i.e. formulations of subgrid-scale forcing). Even for the best-performing neural net-
 117 works, we find poor generalization to flow regimes unseen during training, consistent with
 118 previous literature (Recht et al., 2018; O’Gorman & Dwyer, 2018; Bolton & Zanna, 2019;
 119 Subel et al., 2021; Guan et al., 2022; Subel et al., 2022).

120 Motivated by these generalization issues, as well as the lack of interpretability of
 121 neural networks, there has been increasing interest in the physical sciences community
 122 in symbolic regression (also known as equation discovery). In symbolic regression, in-
 123 stead of an opaque model, the final output of training is a transparent equation, which
 124 often generalizes better (Rudy et al., 2017; Zhang & Lin, 2018; Champion et al., 2019;
 125 Zanna & Bolton, 2020; Mojgani et al., 2021). Many of these studies perform symbolic
 126 regression using sparse linear regression on top of a manually constructed basis of terms
 127 representing various operations (e.g. derivatives or multiples) of base features. Although
 128 powerful for small numbers of terms, this approach quickly becomes prohibitive because
 129 the space and time requirements grow exponentially if we consider higher-order oper-
 130 ations.

131 To address these challenges, in Section 6 we introduce a novel algorithm for equa-
 132 tion discovery based on genetic programming, an alternative form of symbolic regression
 133 that is stochastic but can more efficiently explore higher-order operations (Koza, 1994;
 134 Schmidt & Lipson, 2009; Xing et al., 2022). We adapt this algorithm to search over spa-
 135 tial differential operators, and combine it with linear regression and residual-fitting to
 136 more efficiently and accurately fit constants. We find that the discovered expression of
 137 symbolic parameterization includes features discovered in prior works, is superior to tra-
 138 ditional physics-informed turbulence SGS closures, has similar performance to neural net-
 139 works in both offline and online metrics, and generalizes better to unseen flow regimes
 140 than neural networks and baseline physical parameterizations.

141 2 Numerical Simulations

142 This section describes the simulations we use to generate our datasets, which are
 143 based on `pyqg` (Abernathy et al., 2022), a Python library that models quasi-geostrophic
 144 (QG) systems using pseudo-spectral methods. QG systems are able to capture the gener-
 145 ation of ocean mesoscale eddies, the key process we parameterize in this study, and are
 146 often used to develop and test physic-based parameterizations (P. S. Berloff, 2005; Porta Mana
 147 & Zanna, 2014; Jansen & Held, 2014). In addition, QG systems are a reasonable approx-
 148 imation to the equations of motion in more realistic ocean models in the limit of strong
 149 stratification and rotation. Importantly for this study, which tests numerous parame-
 150 terizations online, they can be simulated much more efficiently than full-fledged ocean
 151 models or GCMs.

152 2.1 Idealized two-layer QG model

153 We use a two-layer version of the QG model from `pyqg`. The model’s prognostic
 154 variable is potential vorticity (PV), denoted as q_1 in the upper and q_2 in the lower layer:

$$q_m = \nabla^2 \psi_m + (-1)^m \frac{f_0^2}{g' H_m} \Delta \psi, \quad m \in \{1, 2\}, \quad (1)$$

155 where ψ_m is the streamfunction with depth H_m , $\Delta \psi = (\psi_1 - \psi_2)$, and $\nabla = \langle \frac{\partial}{\partial x}, \frac{\partial}{\partial y} \rangle$
 156 is the horizontal gradient operator. Zonal and meridional velocities are obtained from
 157 the streamfunction by the relations $u_m = -\partial_y \psi_m$ and $v_m = \partial_x \psi_m$, for each layer with
 158 $m \in \{1, 2\}$. We express the horizontal velocity as a single vector $\mathbf{u}_m = \langle u_m, v_m \rangle$. We
 159 use the beta-plane approximation, such that the Coriolis acceleration is a linear func-
 160 tion of latitude (y) with slope β , such that $f = f_0 + \beta y$, and g' is the reduced gravity.

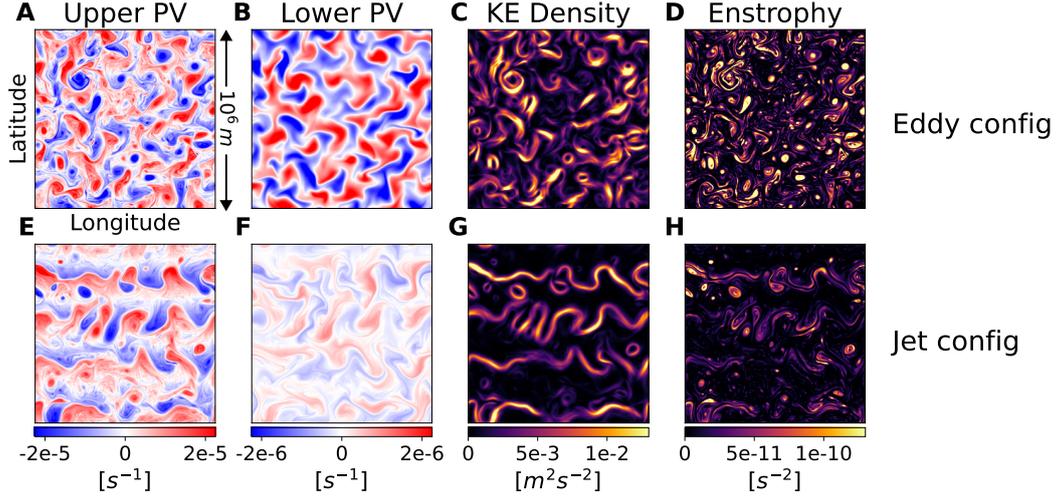


Figure 1. Snapshots of upper (A,E) and lower (B,F) potential vorticity (PV), barotropic kinetic energy (C,G), and barotropic enstrophy (D,H) for simulations run for 10 years in eddy (A-D) and jet (E-H) configurations over a square, doubly-periodic domain of length $10^6 m$. Eddy configuration results in an approximately isotropic distribution of vortices, while jet configuration results in the formation of stable, long-lived jets with more coherent latitudinal structure.

161 The prognostic equations, solved in spectral space, are:

$$\frac{\partial \hat{q}_m}{\partial t} = -\hat{J}(\psi_m, q_m) - ik\beta_m \hat{\psi}_m - ikU_m \hat{q}_m + \delta_{m,2} r_{ek} \kappa^2 \hat{\psi}_2 + \widehat{\text{ssd}}, \quad (2)$$

162 where ∂_t is the Eulerian time derivative, $\widehat{(\)}$ denotes taking the Fourier transform, and
 163 $\kappa = \sqrt{k^2 + l^2}$ is the radial wavenumber, where k and l are zonal and meridional wavenum-
 164 bers, respectively. $J(A, B) = A_x B_y - A_y B_x$ is the horizontal Jacobian. The mean PV
 165 gradient in each layer is $\beta_m = \beta + (-1)^{m+1} \frac{f_0^2}{g'H_m} \Delta U$, where $\Delta U = U_1 - U_2$ is a fixed
 166 mean zonal velocity shear between the two fluid layers. The Dirac delta function, $\delta_{m,2}$,
 167 indicates that the bottom drag with coefficient r_{ek} is only applied to the second and bot-
 168 tom layer. \hat{q} and $\hat{\psi}$ are related to each other via

$$(\mathbf{M} - \kappa^2 \mathbf{I}) \cdot \begin{bmatrix} \hat{\psi}_1 \\ \hat{\psi}_2 \end{bmatrix} = \begin{bmatrix} \hat{q}_1 \\ \hat{q}_2 \end{bmatrix}, \text{ where } \mathbf{M} = \begin{bmatrix} -\frac{f_0^2}{g'H_1} & \frac{f_0^2}{g'H_1} \\ \frac{f_0^2}{g'H_2} & -\frac{f_0^2}{g'H_2} \end{bmatrix}, \quad (3)$$

169 such that either q or ψ can independently identify the state of the system.

170 The model is solved pseudospectrally (Fox & Orszag, 1973) through inverting the
 171 velocity field and PV to real space, calculating the Jacobian using real-space PV fluxes,
 172 and transforming back to spectral space. The scale-selective dissipation (ssd), written
 173 as an additive term in Equation 2, is defined as a highly-scale selective operator, which
 174 attenuates the last 1/3 of the spatial frequencies of the spatial frequencies of all terms
 175 on the right-hand side of Equation 2. More precisely, the operator takes the form of an
 176 exponential filter, $F_c(\kappa)$, such that

$$F_c(\kappa^*) = \begin{cases} 1, & \kappa^* < \kappa_c \\ e^{-23.6(\kappa^* - \kappa_c)^4}, & \kappa^* \geq \kappa_c \end{cases}, \quad (4)$$

177 where κ^* is the non-dimensional radial wavenumber and $\kappa_c = 0.65\pi$, the cut-off
 178 wavenumber. After each time step, $\hat{q}_m(\kappa^*)$ values are multiplied by $F_c(\kappa^*)$. Similar to

179 the 2/3 dealiasing rule (Orszag, 1971), this filtering scheme reduces aliasing errors in the
 180 same range of scales, but additionally provides numerical dissipation necessary for sta-
 181 ble simulations. The energetic contribution from the ssd term is relatively small (see Fig-
 182 ure D11; energy fluxes are an order of magnitude lower than those shown in Figure 2D-
 183 G, and only nonzero over a narrow range of wavenumbers), which is important for sim-
 184 ulations of quasi-2D turbulence (Thuburn et al., 2014).

185 2.2 Model setup

186 We configure the model with a doubly-periodic square domain with a size of $L =$
 187 $10^6 m$, a flat topography, and a total depth of $H = H_1 + H_2$, a fixed mean zonal ve-
 188 locity shear, ΔU with $U_2 = 0$. We set a fixed deformation radius r_d , which is the char-
 189 acteristic scale for baroclinic instability and mesoscale turbulence, using $r_d^2 = \frac{g'}{f_0^2} \frac{H_1 H_2}{H}$
 190 (see Table 1 for parameter values).

191 We select the model’s grid size, Δx , in relation to the deformation radius. To re-
 192 solve mesoscale eddies, one needs to ensure that $r_d/\Delta x$ is greater than 2 (Hallberg, 2013).
 193 With $r_d = 15000 m$, if we choose a 256×256 grid where $\Delta x_{\text{hires}} = L/256 = 3906.25 m$,
 194 then $r_d/\Delta x_{\text{hires}} = 3.84$, so mesoscale turbulence should be well-resolved; if instead we
 195 choose $\Delta x_{\text{lores}} = L/64 = 15625 m$ such that $r_d/\Delta x_{\text{lores}} = 0.96$, we expect that the
 196 simulation is unrealistic with a lack of mesoscale eddies. In such configuration, we would
 197 need to find a parameterization that acts at that resolution to replace the missing tur-
 198 bulent physics. We hereby refer simulations with a grid of 256×256 as “Highres”, and
 199 simulations with a grid of 64×64 as “Lores”. All simulations are run with a numerical
 200 timestep $\Delta t = 1 \text{ hour}$.

201 We consider two distinguishable flow regimes on which generalization properties
 202 of parameterizations can be tested: **eddy configuration**, which leads to the formation
 203 of isotropically distributed eddies, and **jet configuration**, which leads to the formation
 204 of anisotropic jets. These configurations exemplify the two primary scaling regimes of
 205 meridional heat transport (Gallet & Ferrari, 2021), and we will test whether parameter-
 206 izations learned with data from one generalize to the other. Snapshots from each are vi-
 207 sualized in Figure 1, and the `pyqg` parameters used to generate them are given in Ta-
 208 ble 1.

Config.	β [$\frac{1}{ms}$]	r_{ek} [$\frac{1}{s}$]	H_1 [m]	H_2 [m]	ΔU [$\frac{m}{s}$]	g' [$\frac{m}{s^2}$]	r_d [m]
Eddy	1.5e-11	5.787e-07	500	2000	0.025	9.81	15000
Jet	1.0e-11	7.0e-08	500	5000	0.025	9.81	15000

Table 1. Table of parameters used in eddy and jet configuration.

209 2.3 Diagnostics

210 The physical characteristics of QG systems can be qualitatively represented by var-
 211 ious diagnostics such as energy and enstrophy spectra, total kinetic energy and enstro-
 212 phy, and a spectral energy budget (Marques et al., 2022; Yankovsky et al., 2022). Fur-
 213 ther, we also use these diagnostics quantitatively to define difference and similarity met-
 214 rics and compare the performance across different SGS models implemented in low res-
 215 olution simulations in section 4.

216 Resolution has a strong impact on these diagnostics. In Figure 2, we show quasi-
 217 steady state statistics (spectra, kinetic energy timeseries, probability density function)
 218 from simulations run at multiple resolutions (48×48 , 64×64 , 128×128 , and 256×256 grids).
 219 The two higher-resolution simulations ($L/\Delta x \geq 128$) show similar behavior, indicat-

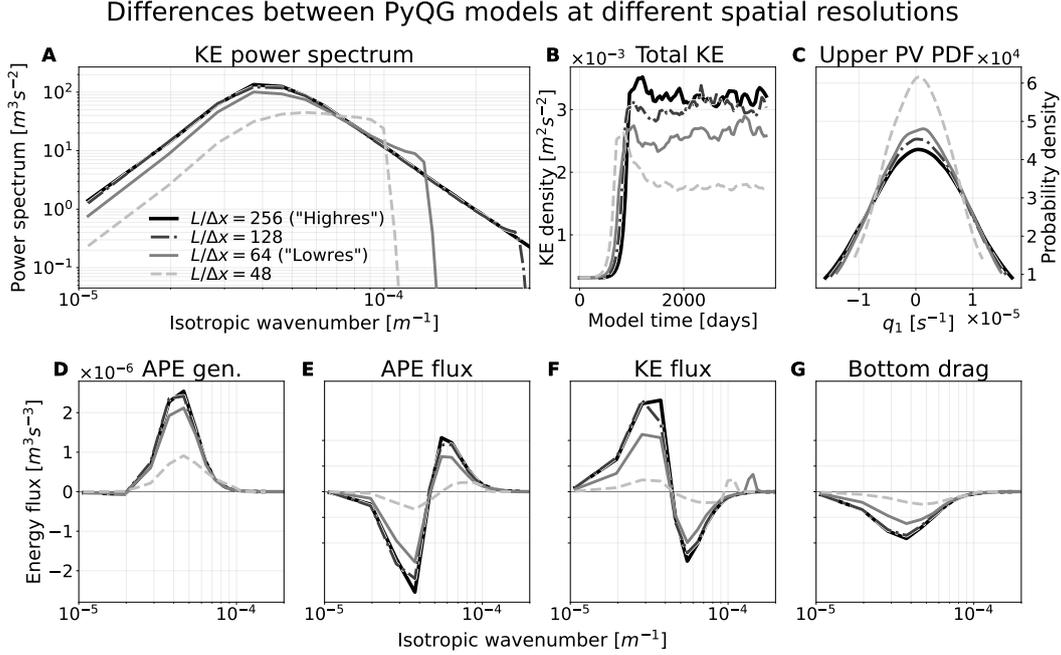


Figure 2. Comparison of time-averaged kinetic energy power spectra summed over fluid layers (A), time-series of total kinetic energy, (B), spatially flattened probability distribution of upper layer PV (C), and spectral energy flux terms (D-G) for eddy configuration simulations at multiple horizontal resolutions: $L/\Delta x=256$, $L/\Delta x=128$, $L/\Delta x=64$, and $L/\Delta x=48$. Higher-resolution simulations ($L/\Delta x \geq 128$) converge, while lower-resolution simulations ($L/\Delta x \leq 64$) differ from each other and from the higher-resolution simulations.

220 ing near-convergence of the statistical characteristics over the wavenumber band contain-
 221 ing most of the kinetic energy of mesoscale eddies. The two lower-resolution simulations
 222 ($L/\Delta x \leq 64$) show significant differences due to insufficiently-resolved turbulent fea-
 223 tures which affect the flow at all scales.

224 To identify the energy pathway of the flow, we evaluate spectral fluxes of different
 225 terms in the two-layer QG system. Let $E(k, l)$ denote the total spectral energy density
 226 of the two-layer system, we have

$$\begin{aligned} \frac{\partial E(k, l)}{\partial t} = & \sum_{m=1}^2 \frac{H_m}{H} \Re \left[\hat{\psi}_m^* \hat{J}(\psi_m, \nabla^2 \psi_m) \right] + \frac{f_0^2}{g'H} \Re \left[(\hat{\psi}_1^* - \hat{\psi}_2^*) \hat{J}(\psi_1, \psi_2) \right] \\ & + \frac{f_0^2}{g'H} k \Delta U \Re \left[j \hat{\psi}_1^* \hat{\psi}_2 \right] - \frac{H_2}{H} r_{ek} \kappa^2 |\hat{\psi}_2|^2, \end{aligned} \quad (5)$$

227 where $*$ denotes complex conjugate, \Re denotes real part, j is the imaginary unit, and the
 228 terms on the right-hand side are the spectral contributions from kinetic energy flux (KE
 229 flux), available potential energy flux (APE flux), available potential energy generation
 230 (APE gen), and bottom drag, respectively.

231 The lower row of Figure 2 demonstrates the typical energy cycle in QG turbulence
 232 (Salmon, 1980; Vallis, 2017): the potential energy of fluctuations is extracted from the
 233 prescribed mean flow (APE gen) and cascades toward small scales up to the deforma-
 234 tion radius (APE flux) where it is converted to kinetic energy due to baroclinic insta-
 235 bility (not shown). The kinetic energy then flows back to large scales following the in-
 236 verse energy cascade (KE flux), where it is ultimately dissipated by friction (bottom drag).

237 The coarse resolution models ($L/\Delta x \leq 64$) poorly resolve the formation of mesoscale
 238 eddies due to baroclinic instability and their enlargement due to the inverse energy cas-
 239 cade (Zanna et al., 2020), leading to underestimated extraction of energy from the mean
 240 flow and a breakdown in the energy cycle.

241 A promising approach to avoid this breakdown is to supplement the resolved ki-
 242 netic energy flux with a so-called “backscatter” parameterization (Jansen & Held, 2014;
 243 Porta Mana & Zanna, 2014) which energize eddies. We believe that efficient subgrid pa-
 244 rameterizations should simulate backscatter at eddy permitting resolution, but also other
 245 processes that may matter, such as dissipation. In this paper we do not study precisely
 246 which physics are parameterized with data-driven subgrid models, but instead quantify
 247 how they influence the resolved energy cycle.

248 3 Diagnosing Subgrid Forcing

249 The goal of our work is to learn models that, given only low-resolution inputs, can
 250 predict the subgrid forcing, S , missing from a low-resolution QG model (Eq. 6). To do
 251 that, we need to first quantify subgrid forcing, which is generally done by filtering and
 252 coarse-graining high-resolution simulations. This is done sometimes under the implicit
 253 assumption that coarsened high-resolution data will have a similar enough distribution
 254 to low-resolution data that the same data-driven parameterizations will work for both.
 255 We use $(\overline{\quad})$ to denote a generic filtering and coarse-graining operator. However, the choice
 256 of filtering and coarse-graining and the choice of subgrid forcing terms to learn are not
 257 uniquely defined.

258 In Sections 3.1 and 3.2, we present several options for how to define subgrid forc-
 259 ing terms in their continuous forms. Specifically, we consider a forcing S in the PV equa-
 260 tion that can be added to a coarse resolution simulation to improve its physics such that

$$\frac{\partial \hat{q}_m}{\partial t} = -\hat{J}(\overline{\psi_m}, \overline{q_m}) - ik\beta_m \hat{\psi}_m - ikU_m \hat{q}_m + \delta_{m,2} r_{ek} \kappa^2 \hat{\psi}_2 + \widehat{\text{ssd}} + \hat{S}, \quad (6)$$

261 where S can take the form of $\{S_{q_{tot}}, S_q, \overline{\nabla} \cdot \phi_q, \text{curl}(S_u, S_v), \text{curl}(\overline{\nabla} \cdot \Phi_{\mathbf{u}})\}$, (detailed
 262 in Sections 3.1 and 3.2).

263 In Section 3.3 we discuss the contribution of the forcing term to the energy bud-
 264 get. Finally, in Section 3.4, we describe three different filtering and coarse-graining op-
 265 tions applied in this work.

266 3.1 Subgrid forcing of potential vorticity

267 We consider three different definitions of subgrid PV forcing for each fluid layer of
 268 the QG model: a total tendency, $S_{q_{tot}}$, which is computed online as the residual between
 269 the low-res and high-res simulation (e.g., P. S. Berloff (2005) and Brenowitz and Brether-
 270 ton (2018)); the subgrid forcing due to nonlinear advection, S_q ; and the subgrid flux di-
 271 vergence forcing, $\overline{\nabla} \cdot \phi_q$.

272 3.1.1 Total tendency (nonlinear advection and numerical dissipation)

273 Let ∂_t^H and ∂_t^L denote tendency functions from the high- and low-resolution mod-
 274 els, respectively (dropping subscripts referring to the model layer for simplicity). For any
 275 given high-resolution q , we can express its total subgrid forcing (Porta Mana & Zanna,
 276 2014; Kent et al., 2016; P. Berloff et al., 2021; Shevchenko & Berloff, 2021) due to the
 277 differences between the high- and low-resolution models with respect to $(\overline{\quad})$ as

$$S_{q_{tot}} = \overline{\partial_t^H q} - \partial_t^L \overline{q}. \quad (7)$$

278 We compute this quantity by setting the initial conditions of the high- and low-resolution
 279 models to be q and \overline{q} , respectively, taking a single step forward with equal Δt , and sub-

tract the tendency of the low-resolution model from the filtered and coarse-grained tendency of the high-resolution model.

3.1.2 Subgrid tendency due to nonlinear advection

Another commonly used definition of subgrid forcing considers the unresolved nonlinear advection (Bolton & Zanna, 2019; Beck et al., 2019; Maulik et al., 2019; Xie et al., 2020; Zanna & Bolton, 2020; Guillaumin & Zanna, 2021; Guan et al., 2022), which can be expressed as

$$S_q = \overline{(\mathbf{u} \cdot \nabla)q} - (\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{q}, \quad (8)$$

where $(\bar{\mathbf{u}} \cdot \bar{\nabla})$ denotes the advection operator defined on the coarse grid. Note that following Grooms et al. (2013) and Porta Mana and Zanna (2014), we define the filtered and coarsened velocity $\bar{\mathbf{u}}$ by inverting the filtered and coarsened PV \hat{q} to $\hat{\psi}$ using Eq. 3, multiplying $\hat{\psi}$ by ik and il , and applying an inverse Fast Fourier Transform (FFT) to obtain \hat{u} and \hat{v} , respectively.

3.1.3 Flux divergence subgrid tendency.

One difficulty in parameterizing subgrid forcing is that naive ML parameterizations may not obey conservation laws, e.g. for momentum and vorticity. Many physical parameterizations are formulated as divergences of fluxes to satisfy conservation laws by the divergence theorem. Ideally, we want to learn ML parameterizations which behave similarly. One approach is to train ML models to predict subgrid forcing (e.g. S_q) but incorporate a numerical divergence operation into their architectures (e.g. as the final layer of a neural network, see Zanna and Bolton (2020)). Another is to diagnose a different quantity whose divergence equals the subgrid forcing (Pawar et al., 2020; Stoffer et al., 2021; Yuval et al., 2021), train ML models to predict this quantity (i.e., the subgrid flux) directly, and compute divergences outside the learned model as part of the implementation of parameterization.

To enable experimentation with this second approach, we define a “subgrid flux” that will be predicted by the FCNN

$$\phi_q = \bar{\mathbf{u}}\bar{q} - \bar{\mathbf{u}}\bar{q}. \quad (9)$$

Under the assumption that the flow is incompressible (i.e. that $\nabla \cdot \mathbf{u} \approx \bar{\nabla} \cdot \bar{\mathbf{u}} \approx 0$) and that differentiation commutes with filtering and coarsening, we can show that

$$\bar{\nabla} \cdot \phi_q = \bar{\nabla} \cdot (\bar{\mathbf{u}}\bar{q} - \bar{\mathbf{u}}\bar{q}) \approx S_q.$$

These three formulations (S_q , $S_{q_{tot}}$, and $\bar{\nabla} \cdot \phi_q$) are always highly correlated and often nearly identical, but the exact value of this correlation (especially for $\bar{\nabla} \cdot \phi_q$ vs. the others) can range from 0.75 to $1 - 10^{-14}$, depending on the layer, timestep, configuration, and especially the filtering and coarse-graining operator (Section 3.4).

3.2 Subgrid forcing of velocity

Realistic ocean models use velocity as their prognostic variable with temperature and salinity, rather than PV. Many studies have focused on momentum subgrid closures (Zanna & Bolton, 2020; Guillaumin & Zanna, 2021). Here, we define momentum forcing which is later related to the subgrid PV forcing.

The first definition involves the advection-based subgrid momentum forcing given by

$$\mathbf{S}_u = \overline{(\mathbf{u} \cdot \nabla)\mathbf{u}} - (\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{\mathbf{u}}. \quad (10)$$

319 Analogous to Equation 9, we also define momentum subgrid flux terms as

$$\begin{aligned}\phi_u &= \overline{\mathbf{u}u} - \overline{\mathbf{u}}\overline{u}, \\ \phi_v &= \overline{\mathbf{u}v} - \overline{\mathbf{u}}\overline{v}.\end{aligned}\tag{11}$$

320 We use $\Phi_{\mathbf{u}} = (\phi_u, \phi_v)$ to denote the matrix of all four terms of the stress tensor, with
321 a total forcing $\overline{\nabla \cdot \Phi_{\mathbf{u}}}$; where the y -component of ϕ_u is equal to the x -component of ϕ_v .

322 Given that the PV flux is composed of two parts: the relative vorticity flux and
323 the buoyancy or thickness flux, we note that the relative vorticity flux is related to the
324 momentum flux via the curl operator (Vallis, 2017; Killworth, 1997). In our simulations,
325 we update the PV tendency with the curl of subgrid momentum forcing, e.g., $\text{curl}(S_u, S_v) =$
326 $\partial_x S_v - \partial_y S_u$, which serves as a momentum parameterization in QG equations (similarly
327 for $\Phi_{\mathbf{u}}$). Note that $\text{curl}(S_u, S_v)$ is different from S_q when obtained from the respective
328 coarse-grained fluxes: correlations between the two terms range from +0.2 to -0.4 de-
329 pending on the filtering and coarse-graining operator.

330 3.3 Contribution of forcing to diagnostics

331 Similar to Eq. 5, we derive the spectral contribution of subgrid-scale forcing towards
332 total energy

$$\left(\frac{\partial E(k, l)}{\partial t}\right)^{\text{sub}} = -\frac{1}{H} \sum_{m=1}^2 H_m \mathbb{R} \left[\hat{\psi}_m^* \hat{S}_m \right],\tag{12}$$

333 where \hat{S}_m denotes the spectral PV tendency induced by the SGS model in the m th layer.
334 This equation states that the total tendency induced by the subgrid term can be writ-
335 ten as the projection of subgrid tendency onto the streamfunction in each layer.

336 3.4 Coarse-graining and filtering

337 We are using a combination of filtering and coarse-graining to diagnose the sub-
338 grid forcing. There are a number of possible ways to filter and coarse-grain simulations.
339 Filtering can be identified by various convolutional kernels (top-hat, Gaussian, e.g., Sagaut
340 (2006)), which can be approximated on a given mesh with quadrature rules (Xie et al.,
341 2020; Guillaumin & Zanna, 2021), polynomials based on Laplacian operator (Sagaut &
342 Grohens, 1999; Grooms et al., 2021) or applied in spectral space (Guan et al., 2022). Coarse-
343 graining methods include spectral truncation (Thuburn et al., 2014), averages over boxes
344 (Porta Mana & Zanna, 2014; Beck & Kurz, 2021) or subsampling (Xie et al., 2020, i.e.
345 selection of every K 's point). The combination of filtering and coarse-graining has also
346 been shown to reduce aliasing in the computation of subgrid forcing (Zanna & Bolton,
347 2021). Here, rather than focusing on one method for filtering and coarse-graining, we
348 examine the sensitivity of our results to three different operators for diagnosing the sub-
349 grid forcing in our simulations: two different filters in spectral space (referred to as “Op-
350 erator 1” and “Operator 2”) , and one filter in real space (and “Operator 3”).

351 For Operator 1 and Operator 2, given that `pyqg` is a pseudo-spectral model, it is
352 natural to use spectral methods to perform coarse-graining and filtering. For data gen-
353 eration, we first coarse-grain and then filter , which are commutative for elementwise spec-
354 tral filtering operators, so can be done in whichever order is most convenient. Coarse-
355 graining is done by coarse-graining the simulation by a factor of K , or more precisely,
356 truncating the set of spatial modes of \hat{q} by only keeping the first $1/K$. For example, in
357 the case of going from resolution 256×256 to 64×64 , we start with a \hat{q} with 128 modes
358 and only keep the first 32. Spectral filtering generally consists of applying selective de-
359 cay that reduces the strength of the highest frequencies, whereas low-frequency compo-
360 nents are mostly retained after truncation. Here, we use two different filtering methods
361 (Sections 3.4.1 and 3.4.2).

362 Finally, to mimic the procedure necessary for ocean models which are not run in
 363 spectral space, we convert our output to a Cartesian grid and applying filtering and coarse-
 364 graining in real space (“Operator 3”, Section 3.4.3).

365 **3.4.1 Operator 1: spectral truncation, sharp filter**

366 The first option is implemented by simply applying the same quadruple-exponential
 367 filter used by `pyqg` to implement small-scale dissipation in Equation 4. This filter leaves
 368 small wavenumbers unchanged but attenuates wavenumbers above a cutoff threshold $\kappa^c \equiv$
 369 $2/3$ of the low-resolution model’s Nyquist frequency:

$$\hat{q}_\kappa = \begin{cases} \hat{q}_\kappa, & \kappa < \kappa^c \\ \hat{q}_\kappa * e^{-23.6(\kappa - \kappa^c)^4 \Delta x_{\text{lores}}^4}, & \kappa \geq \kappa^c. \end{cases} \quad (13)$$

370 In some sense, this is the most conservative choice of filter possible (i.e. closest to not
 371 filtering at all), since it will already be applied within the ocean model. We use “Op-
 372 erator 1” to refer to spectral truncation followed by the application of this filter.

373 **3.4.2 Operator 2: spectral truncation, softer Gaussian filter**

374 The second spectral filtering option considered (“Operator 2”) is to instead apply
 375 the following Gaussian filter to all remaining modes:

$$\hat{q}_\kappa = \hat{q}_\kappa * e^{-\kappa^2 (2\Delta x_{\text{lores}})^2 / 24} \quad (14)$$

376 This choice of filter is based on Guan et al. (2022) and Pope (2000). According to the
 377 definition of the filter width given by Lund (1997), this filter is twice as large as the grid
 378 size of the coarse model.

379 **3.4.3 Operator 3: diffusion-based filtering, real-space coarsening**

380 Finally, we consider a procedure which is closer to the procedure needed for ocean
 381 models. We apply `GCM-Filters` (Grooms et al., 2021; Loose et al., 2022), a recent fil-
 382 tering method which approximates the spectral transfer function of Gaussian filter with
 383 polynomials based on the Laplacian diffusion operator, converting our `pyqg` output to
 384 a Cartesian grid. We then coarse-grain the filtered output in real space. To reduce the
 385 resolution by a factor of K , we average the input field over non-overlapping boxes of $K \times K$
 386 points. We call this procedure “Operator 3.”

387 A comparison of the effects of these different filtering and coarse-graining opera-
 388 tors on PV and its subgrid forcing is shown in Figures 3 and 4.

389 **3.5 Comments regarding notation**

390 We use superscripts (1), (2), and (3) (for Operators 1, 2, and 3, respectively) to
 391 describe subgrid forcing computed with each operator. For example, $S_q^{(1)}$ signifies the
 392 subgrid tendency due to nonlinear advection diagnosed by Equation 8 and computed with
 393 the operator from Section 3.4.1, while $\Phi_{\mathbf{u}}^{(3)}$ signifies the tensor of velocity subgrid fluxes
 394 diagnosed by Equation 11 and computed with the operator from Section 3.4.3.

395 In addition, we use $\overline{(\)}$ to refer to all low-resolution variables, whether coarse-grained
 396 from a high-resolution simulation or natively from a low-resolution simulation. The rea-
 397 son is that we evaluate parameterizations offline over filtered and coarse-grained high res-
 398 olution variables, but evaluate parameterizations online over low-resolution variables. Al-
 399 though these variables can be different in various respects (e.g., may be differently dis-
 400 tributed), when learning data-driven parameterizations from subgrid forcing data col-
 401 lected offline, we necessarily assume that one will generalize to the other. Though this
 402 is an assumption we explicitly test in online evaluation.

403
404

In the remaining text, we also simplify $\bar{\nabla}$ to just ∇ for conciseness. It should be treated as $\bar{\nabla}$ when applied to a coarsened or low-resolution variable.

Comparing effects of three filtering and coarse-graining operators on potential vorticity forcing

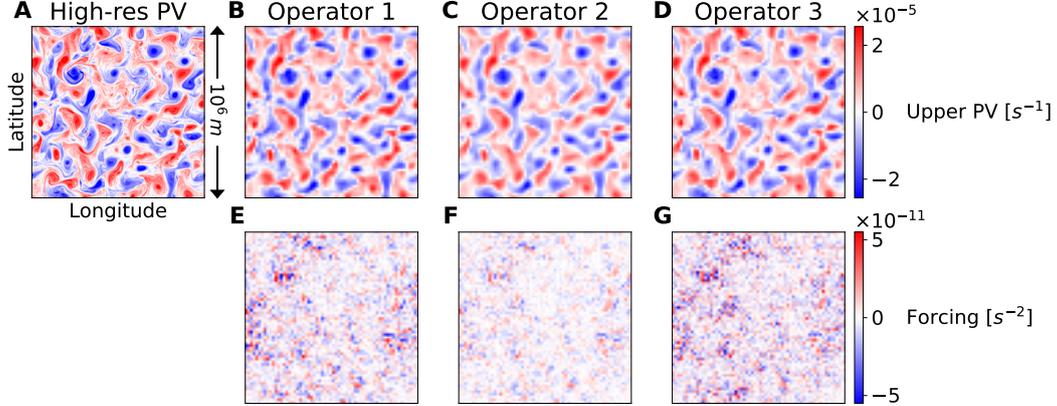


Figure 3. Comparison of the effects of three different methods of filtering and coarse-graining 256×256 eddy configuration initial states (A) to 64×64 (B-D), along with resulting forcing terms S_q , defined in Eq. 8 (E-G). Operators 1 (B,E) and 2 (C,F) truncate Fourier modes and apply sharp and soft spectral filters, respectively, while Operator 3 (D,G) applies diffusion-based filtering and averaging in real space. See Section 3.4 for operator definitions and Figure 4 for comparisons of associated spectral properties.

405

4 Metrics

406
407
408
409
410
411
412
413
414
415
416

In the sections that follow, we evaluate a large number of parameterizations on data generated with different operators and forcing formulations, as well as different inputs and architectures. Given that the models are too numerous to manually inspect, we define several levels of metrics to quantify their performance. In Section 4.1, we define metrics which can be evaluated offline, i.e. on held-out testing sets of subgrid forcing data. In Section 4.2, we define online metrics that measure the similarity of low-resolution simulations run with the parameterization to high-resolution simulations. These metrics account for (1) aspects of the model physics (e.g., kinetic energy flux at different scales), (2) the climatological biases and characteristics of key variables (e.g, distributions of potential vorticity), and (3) the forecast skill of the simulation (e.g., decorrelation timescales of short term forecasts).

417

4.1 Offline

418
419

Offline metrics quantify the parameterization’s skill at predicting its intended target. For each fluid layer, we consider:

420
421
422
423
424

1. The coefficient of determination (R^2), $1 - \frac{\mathbb{E}[(S - \hat{S})^2]}{\mathbb{E}[(S - \mathbb{E}[S])^2]}$, which is 1 when predictions are perfect, 0 when predictions are no better than always predicting the mean, and negative when worse than always predicting the mean.
2. Pearson correlation (ρ), $\frac{\text{Cov}(S, \hat{S})}{\sigma_S \sigma_{\hat{S}}}$, where σ denotes the empirical standard deviation of a quantity over the dataset. This quantity is between -1 and 1 and can

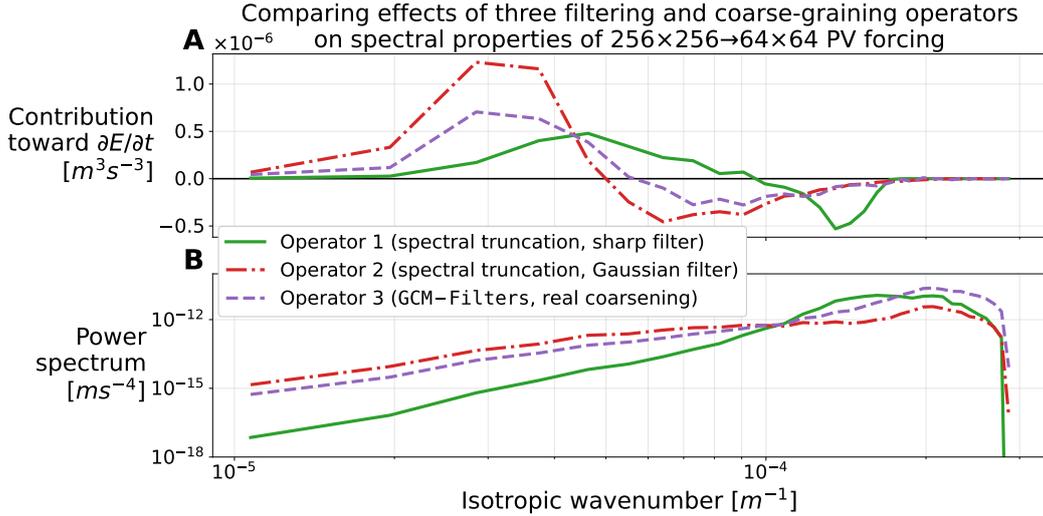


Figure 4. Energy redistribution, Eq. 12 (A) and power spectra (B) of S_q by filtering and coarse-graining operator (computed on eddy configuration data, averaged over time, and summed across layers). Each operator produces forcing which redistributes energy differently across scales with different spatial spectra. See Figure 3 for comparisons of forcing snapshots.

425 remain high even when R^2 is negative, e.g. if predictions are wrong by a large but
 426 consistent scaling factor.

427 These metrics are evaluated on held-out datasets of filtered and coarse-grained high-resolution
 428 simulations from both eddy and jet configurations. They can either be aggregated over
 429 time and space or expressed as functions of time or space. In addition, we visualize the
 430 power and energy redistribution spectra of the predicted subgrid forcing and compare
 431 them to the corresponding quantities for the ground-truth forcing.

432 4.2 Online

433 In contrast to offline metrics, we evaluate online metrics by initializing a new QG
 434 simulation at low resolution and, at every time step, passing its state to the parameter-
 435 ization and adding the parameterization's output to the PV tendency. The distribution
 436 of these low-resolution states may therefore be different, but by analyzing the ultimate
 437 results and testing for various forms of consistency with high-resolution results (i.e. on-
 438 line metrics), we can evaluate whether the parameterization is effective at improving the
 439 model physics and/or the climatological or forecast skill.

440 To compute such online metrics, we first run 5 parameterized low-resolution sim-
 441 ulations for 10 years in both eddy and jet configurations initialized from different ran-
 442 dom states, saving all state variables and diagnostics described in Section 2.3. We then
 443 compute distance metrics between the (statistical and spectral) distributions of these vari-
 444 ables from the parameterized low-resolution simulation and those from 5 simulations run
 445 at high resolution in corresponding configurations. Finally, we normalize these distances
 446 by the corresponding metrics for *unparameterized* low-resolution simulations to obtain
 447 more interpretable similarity scores.

448

4.2.1 Differences between time-averaged power spectra and fluxes

449

450

451

452

Some of the most important characteristics of simulations are how energy and enstrophy distribute and flow across scales, which we measure using power spectra and the spectral flux diagnostics described in Section 2.3. Ideally, a parameterized simulation should match a high-resolution simulation with respect to all such quantities.

453

454

For both power spectra and fluxes, we compute a total root mean squared difference between curves f :

$$\text{spectral_diff}(\text{sim1}, \text{sim2}; f) \equiv \sqrt{\frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} (f_{\text{sim1}}(k) - f_{\text{sim2}}(k))^2} \quad (15)$$

455

456

457

458

459

460

461

462

where \mathcal{K} is a suitably chosen set of isotropic wavenumbers common to both simulations. In our case, \mathcal{K} is evenly distributed in log space and is up to 2/3 of the Nyquist frequency of the low-resolution simulation ($\approx 1.07 \times 10^{-5} m^{-1}$). We compute this metric for the energy and enstrophy power spectra in each layer and for the spectral energy fluxes (KE flux, APE flux, APE generation, and bottom drag), yielding a total of 8 metrics. The contribution of parameterizations towards total energy (Eq. 12) is added onto the KE flux term for parameterized low-resolution simulations. An illustration of this kind of distance metric is shown in Figure 5A.

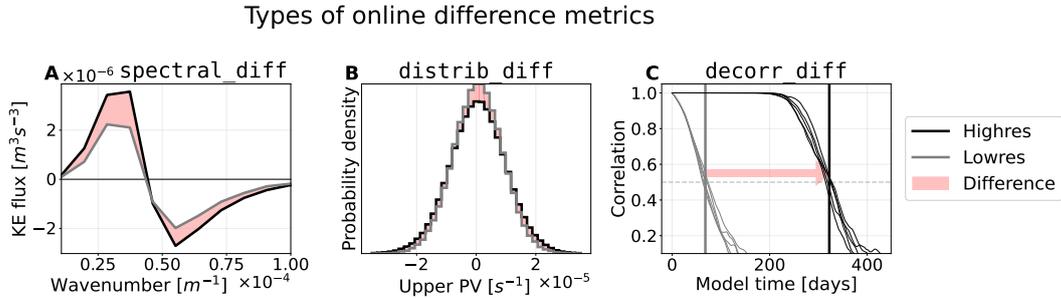


Figure 5. Illustration of the three types of difference metrics defined in Section 4.2.

`spectral_diffs` (A) compute the RMSE between different quantities summed over isotropic wavenumber κ . `distrib_diffs` (B) compute the earth mover’s distance between the marginal distributions of variables at the end of the simulation. `decorr_diff` (C) estimates how much faster a given simulation diverges from a high-resolution simulation when starting from the same random initial state.

463

4.2.2 Differences between spatially flattened probability distributions

464

465

466

We also consider differences between the empirical distributions of various quantities in different simulations at the end of the simulation, which we measure with earth mover’s distance or Wasserstein distance (Monge, 1781; Rubner et al., 2000):

$$\text{distrib_diff}(\text{sim1}, \text{sim2}; f) \equiv \int_{-\infty}^{\infty} |P_{\text{sim1}}(f \leq x) - P_{\text{sim2}}(f \leq x)| dx, \quad (16)$$

467

468

469

470

471

472

where $P_{\text{sim}}(f \leq x)$ is a cumulative distribution function of quantity f in a given simulation. If we imagine the two probability density functions as mounds of earth, this metric corresponds to the minimum amount of work required to move all the mass from one mound to the other. For 1-dimensional distributions, it reduces to the integral of the difference in each cumulative distribution function, which we approximate empirically. We compute these differences for the quasi-steady-state distributions (marginalized over space

473 and at the final timestep) of u , v , q , the kinetic energy density $(u^2+v^2)/2$, and enstro-
 474 phy $\text{curl}^2(\mathbf{u})/2$ at each layer. This leads to 10 total metrics for each simulation.

475 Note that when comparing low-resolution to high-resolution metrics, we are compar-
 476 ing the distributions of, e.g., u and \bar{u} , so histograms are appropriately normalized.
 477 An illustration of this kind of comparison is shown in Figure 5B, though for brevity we
 478 show only the difference in the integrals of PDFs rather than the integrals of the cor-
 479 responding CDFs (which gives the exact value of `distrib.diff`).

480 *4.2.3 Differences in decorrelation times*

481 The previous metrics consider whether aggregate, long-term simulation statistics
 482 (i.e. “climate”) match those of high-resolution simulations. Arguably, though, param-
 483 eterizations should also improve the similarity of short-term trajectories (i.e. “weather”)
 484 between low- and high-resolution simulations – or at least not significantly worsen it.

485 We measure this short-term similarity by defining a “decorrelation time” metric,
 486 i.e. minimum time t to achieve correlation δ from above, averaged over ensemble of ini-
 487 tial conditions q_0 and their perturbations ϵ

$$\text{decorr_time}(\text{sim1}, \text{sim2}) \equiv \mathbb{E}_{q_0, \epsilon} \left[\min_t \left\{ t : \text{Corr} \left(q_{\text{sim1}}^{(t)}(q_0), q_{\text{sim2}}^{(t)}(q_0 + \epsilon) \right) \leq \delta \right\} \right] \quad (17)$$

488 where each $q_{\text{sim}}^{(t)}(q_0)$ denotes a snapshot of the PV for the given `simulation` integrated for
 489 time t starting from an initial condition q_0 sampled from the quasi-steady state, ϵ is a
 490 small independent Gaussian perturbation with standard deviation 10^{-10} , and $\delta = 0.5$.
 491 When `sim1` and `sim2` have the same dimensionality, `Corr` denotes the simple Pearson
 492 correlation; when they have different dimensionalities (i.e. if `sim1` is higher resolution
 493 than `sim2`), we compute the correlation after filtering and coarse-graining the higher-
 494 resolution simulation to the resolution of the other simulation using Operator 1 (Equa-
 495 tion 13). We approximate expectations $\mathbb{E}_{q_0, \epsilon}$ using empirical averages over 5 random sam-
 496 ples of q_0, ϵ , and we use the same random high-resolution q_0 for all low-resolution mod-
 497 els so that correlation trends for different low-resolution models can be paired.

498 With a decorrelation time now defined, we can compare the expected time it takes
 499 for one type of simulation to fall out of sync with another, vs. the expected time it takes
 500 for one simulation to fall out of sync with a perturbed version of itself:

$$\text{decorr_diff}(\text{sim1}, \text{sim2}) \equiv \text{decorr_time}(\text{sim1}, \text{sim1}) - \text{decorr_time}(\text{sim1}, \text{sim2}), \quad (18)$$

501 In our study, `sim1` is a high-resolution simulation, which stays correlated with a perturbed
 502 version of itself for a relatively long time, while `sim2` will be a low-resolution simulation.
 503 With the eddy configuration, the correlation of high-resolution simulations stays above
 504 0.5 for about 1 year (black vertical line in Figure 5C; this roughly quantifies the limit
 505 of predictability of the system), whereas unparameterized low-resolution simulations remain
 506 >0.5 correlated for about 2 months (gray vertical line in Figure 5C), leading to a
 507 `decorr_diff` of 10 months (red arrow in Figure 5C). For a parameterized low-resolution
 508 simulation, we might hope that its `decorr_diff` is lower than that of the low-resolution
 509 model (e.g., 8 months), indicating that its short-term evolution is more consistent with
 510 that of the high-resolution model.

511 *4.2.4 From difference to similarity*

512 One issue with defining such a variety of distance metrics is that they become dif-
 513 ficult to compare especially when they have different units. However, for any particu-
 514 lar metric, what we care about is not its actual value but whether it is smaller for pa-
 515 rameterized simulations (vis-à-vis Highres simulations) than for low-res simulations. To
 516 that end, we re-express our distance metrics as similarity scores that quantify how much

517 closer parameterized models are to high-res than to low-res:

$$\text{Similarity}(\text{param, high-res; diff}) \equiv 1 - \frac{\text{diff}(\text{param, high-res})}{\text{diff}(\text{low-res, high-res})}. \quad (19)$$

518 This similarity score is approximately 1 if the parameterized model’s distance to the high-
 519 res model is much smaller than that of the low-res model (and exactly 1 for the high-
 520 res model); it is approximately 0 if this distance is approximately equal to that of the
 521 low-res model (and exactly 0 for the low-res model), and is less than 0 if the distance
 522 is larger than that of the low-res model. An example is shown in Figure 6. We also in-
 523 clude a validation of the consistency of these scores with respect to high- and low-resolution
 524 simulations generated with different random initial conditions in Figure D10. In general,
 525 we evaluate our online results using similarity scores.

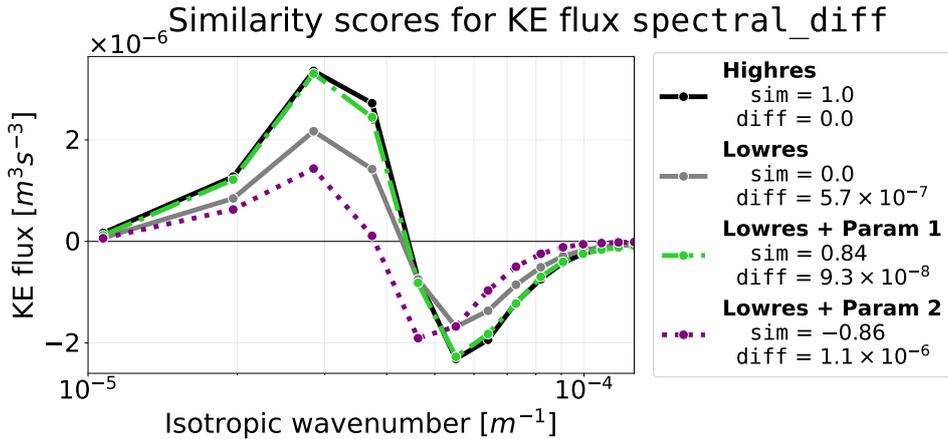


Figure 6. Example online similarity scores for two parameterizations corresponding to the `spectral_diff` of their KE flux terms with respect to high-res (as compared to low-res). In this example, the first parameterized model’s KE flux curve is much closer to that of the high-res model than the low-res model, so its similarity is positive and close to 1 (though slightly lower than it might seem from visual inspection due to the logarithmic x -scale). The second parameterized model, on the other hand, is further away than low-res, so it receives a negative score.

526 Note that there are many alternative metrics that could have been selected (e.g.,
 527 RMSE for decorrelation timescales, Kullback-Leibler for probability distributions (Kullback
 528 & Leibler, 1951), absolute error for differences in spectra, etc), that may augment the
 529 set defined here to focus on other aspects of the simulations (e.g. extreme events).

530 4.3 Experimental Setup

531 With our datasets and metrics now defined, we now describe our experiments to
 532 learn and evaluate parameterizations. In total, we test 148 parameterizations—105 fully
 533 convolutional neural networks (FCNNs) trained with different dataset design decisions
 534 (described in Section 5), a hybrid linear and symbolic regression method using genetic
 535 programming (described in Section 6), and 42 different parameter settings spread over
 536 three baseline physical parameterizations: symbolic regression from Zanna and Bolton
 537 (2020), backscatter from Jansen and Held (2014), and Smagorinsky (1963), all three de-
 538 scribed in Appendix A.

539 The trained parameterizations are evaluated offline and also implemented into the
 540 coarse resolution simulation with 64×64 horizontal resolution for the online evaluation.

To simplify the discussion, we begin by describing these categories of parameterizations individually, along with some of the experimental results specific to those categories. We then compare performance across parameterization categories in Section 7.1.

Because we have multiple categories of parameterization (FCNN, genetic programming, and baselines) and multiple categories of online metric (spectral, distributional, and decorrelation time) with numerous individual parameterizations (148) and metrics (19) within each category, we will often simplify as follows. For each category of online metric, we summarize individual parameterization’s scores by taking means (or medians and percentiles if visualizing variation). For each category of parameterization, we either show a distribution of these mean scores, or select individual parameterizations to highlight from the Pareto frontier of mean scores within each category, i.e. the set of parameterizations which maximize some linear combination of mean scores.

5 Convolutional Neural Network Parameterizations

We consider parameterizations implemented as fully convolutional neural networks (FCNNs) which output predictions for all x, y points simultaneously. Models receive input data at all points x, y in both layers (though we train separate models for each fluid layer to reduce memory cost during training), which allows them to be maximally flexible, and therefore useful for studying the effects of changing attributes of the dataset on best-case performance.

5.1 Dataset design choices

For our FCNN experiments, we are interested in how the structure of the dataset affects the offline and online performance. We train FCNNs to predict subgrid forcing diagnosed with each of the five forcing formulations ($\{S_{q_{tot}}, S_q, \bar{\nabla} \cdot \phi_q, \text{curl}(S_u, S_v), \text{curl}(\bar{\nabla} \cdot \Phi_{\mathbf{u}})\}$, Section 3), and for each forcing formulation, we generate three FCNNs trained on datasets generated by each filtering and coarse-graining operator (Section 3.4). Finally, we also investigate the effect of the choice of input variables we pass to the FCNN by testing every non-empty element of the power set of $\{\bar{q}, \bar{\mathbf{u}}, \nabla \bar{\mathbf{u}} = (\partial_x \bar{u}, \partial_x \bar{v}, \partial_y \bar{u}, \partial_y \bar{v})\}$, which is 7 options in total. This gives us $5 \times 3 \times 7 = 105$ total options for constructing FCNN parameterizations.

Notation-wise, we refer to models trained on each option as, e.g., $\text{FCNN}(\bar{q}, \bar{\mathbf{u}} \rightarrow S_{\mathbf{u}}^{(2)})$, which signifies an FCNN trained on the values of PV and velocity to estimate subgrid momentum forcing (Eq. 10), computed with Operator 2 (spectral truncation + Gaussian filter, Section 3.4.2).

For each operator and configuration, we use data from 250 independent high-resolution simulations started from random noise and run for 10 simulation years (generally reaching the quasi-steady state by 3-5 simulation years depending on the configuration; we also include data from the transient spin-up state in the dataset). We sample subgrid forcing formulations (i.e. potential prediction targets) and coarsened model state variables (i.e. potential input variables) every 1000 simulation hours, to remove almost all correlation between successive samples. This gives us 6 datasets (2 simulation configurations, jet and eddy, \times 3 operators) each with 21,750 snapshots of input and target variables (each of which is a $64 \times 64 \times 2$ array).

5.2 Architectural details and constraints

Following Guillaumin and Zanna (2021), we train FCNNs with 8 fully convolutional layers (128 and 64 filters for the first two layers, respectively and 32 thereafter), ReLU activations, batch normalization after all intermediate layers, and circular padding due to the periodicity of the domain. Each input variable at each fluid layer is passed in a

588 separate input channel. The loss function is mean squared error (MSE), defined as $\mathbb{E}[(S - \hat{S})^2]$,
 589 where \mathbb{E} denotes the expected value over a dataset and S is a generic prediction target.
 590 The FCNNs are trained for 50 epochs on a MSE loss evaluated over minibatches of 64
 591 samples.

592 In preliminary experiments, we found that constraining the FCNNs' final output
 593 layers to have zero spatial mean when predicting S_q and S_u was necessary for online nu-
 594 merical stability (as otherwise, q can continually increase, leading to Courant-Friedrichs-
 595 Lewy (CFL) condition violations). This is done within the FCNN architecture and not
 596 as a post-processing step. The constraint ensures that at each timestep, parameteriza-
 597 tions redistribute but not increase or decrease the total PV. However, when predicting
 598 ϕ_q and Φ_u , we leave FCNNs unconstrained because we only apply predictions after tak-
 599 ing their divergence.

600 Although the chosen architecture could be improved, e.g., by adopting the U-Net
 601 model of Ronneberger et al. (2015), our goal is not to maximize the performance but to
 602 study its relationship with dataset design choices.

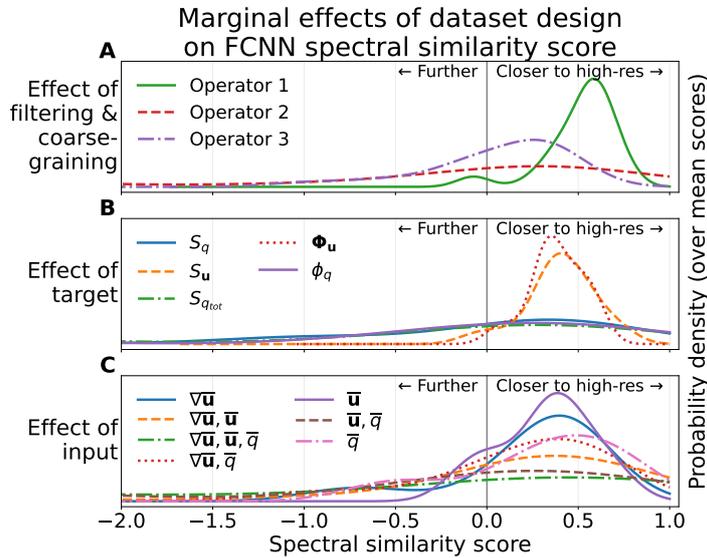


Figure 7. Visualizing the effects of different dataset design choices: A) filtering and coarse-graining operator, B) forcing formulation, C) input. We use the probability distributions of mean spectral similarity scores, conditioned on each design choice, and smoothed using kernel density estimation for visual clarity. Similarity score probability mass further to the right (past the 0 line, and towards 1) indicate that the corresponding difference metric was low compared to low-res, therefore indicating good online performance. The results suggest that marginally, similarity was highest along most metrics for parameterizations trained to use velocity (Panel C) to predict velocity-based subgrid forcing (Panel B) calculated with a sharp spectral filter (Panel A).

603 5.3 Sensitivity of FCNN performance to dataset design

604 We now present FCNN-specific results of how online performance varies with the
 605 dataset design choices described in Section 5.1. For each design choice, we constructed
 606 the corresponding eddy configuration training data, trained an FCNN parameterization,
 607 and evaluated it in both eddy and jet configuration, both offline and online. In each case,
 608 all simulations were numerically stable (the CFL condition was not violated). The sta-

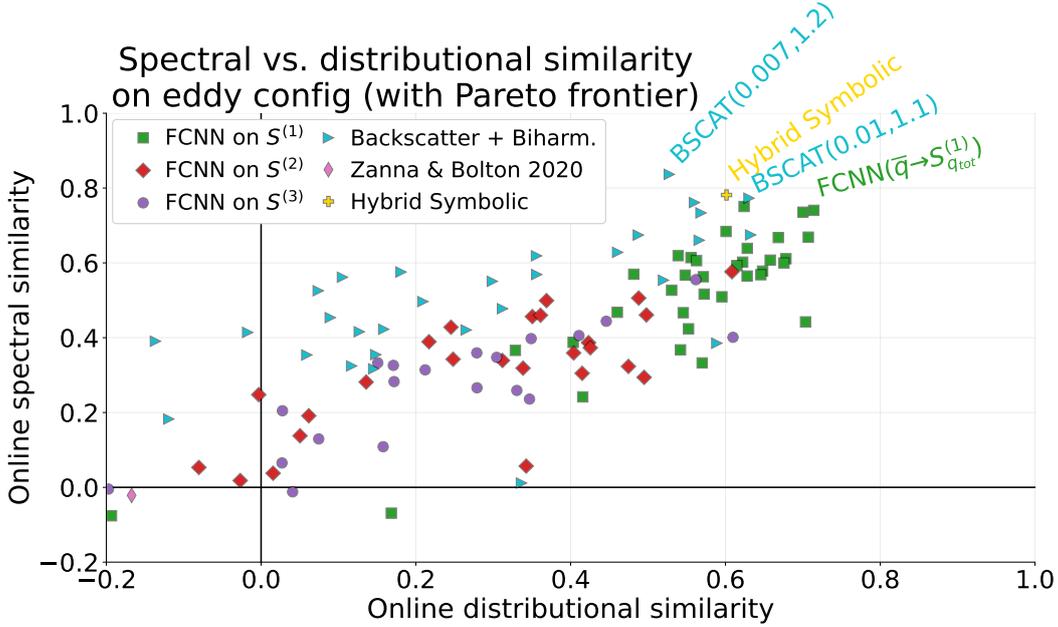


Figure 8. Mean eddy-configuration distributional and spectral similarity scores for many of the 148 parameterizations tested, with those defining the Pareto frontier shown with text (the runs with remaining parameterizations, including all Smagorinsky runs, have scores to the lower-left of the plot range.)

609 bility is likely due to our architectural constraints (as discussed above) and perhaps the
 610 spectral numerical dissipation scheme of `pyqg`. However, performance in terms of similarity
 611 metrics varied greatly.

612 To visualize this variation, Figure 7 shows the kernel density estimates (Rosenblatt,
 613 1956) of conditional probability distributions of the mean `spectral_diff` similarity score
 614 (substituting Eq. 15 into Eq. 19) for different dataset design choices of filtering and coarse-
 615 graining operator, forcing formulation, and input variables. Specifically, these plots show
 616 the distribution of the average similarity score across KE power spectra, enstrophy power
 617 spectra, and energy budget terms over isotropic wavenumber, conditioned on different
 618 choices of filter and coarse-graining (Fig. 7A), targeted forcing formulation (Fig. 7B),
 619 and input variables (Fig. 7C). Probability density closer to 1 indicates better performance.
 620 Overall, we see higher spectral similarity scores for FCNNs trained on data generated
 621 with Operator 1 (spectral truncation with sharp filter) (Fig. 7A) and predicting momen-
 622 tum forcing rather than PV forcing (Fig. 7B). The choice of input has a weaker impact
 623 on these scores (Fig. 7C), though simpler terms ($\bar{\mathbf{u}}$, $\nabla\bar{\mathbf{u}}$, or \bar{q} alone) do slightly better,
 624 consistent with (Dresdner et al., 2022). The same results hold for `distrib_diff` simi-
 625 larity (not shown), which is strongly correlated with `spectral_diff` (Figs 8).

626 In addition, we can gain insights through analyzing specific models. If we look at
 627 the Pareto frontier of eddy-configuration distributional and spectral similarity across all
 628 our experiments (Fig. 8), we find that the only Pareto-optimal FCNN predicts $S_{q_{tot}}^{(1)}$, which
 629 is computed with Operator 1 but formulated in terms of PV rather than velocity. If we
 630 compare this FCNN to others which are identical except for the filtering and coarse-graining
 631 operator (Fig. D4) or forcing formulation (Fig. D5), we find again that the choice of op-
 632 erator continues to matter, but that the forcing formulation has much less effect as FC-
 633 NNs predicting other forcing formulations with the same filtering and coarse-graining

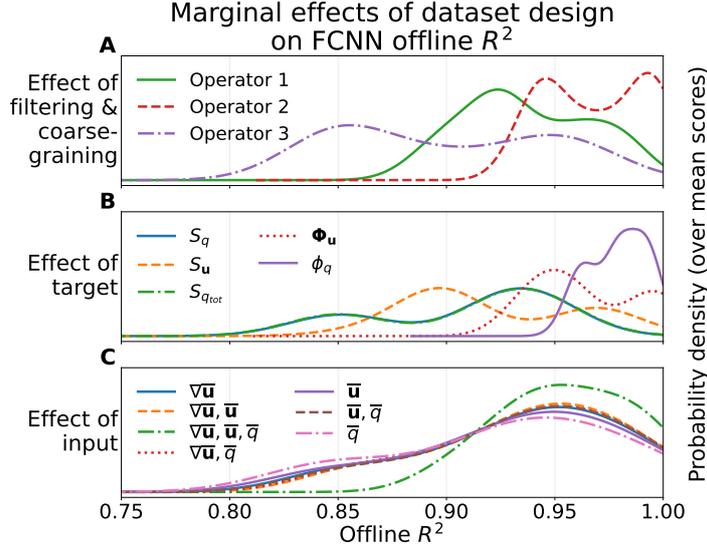


Figure 9. Offline R^2 scores by dataset design choice as in Figure 7, almost all of which achieve an R^2 of above 0.8 regardless of condition. The best models by offline R^2 are different from those in Figure 7.

operator all have near-identical effects. Combining these individual results with the aggregate results of Fig. 7, our overall interpretation is that a) the choice of operator is the most important for online performance, and b) predicting velocity forcing (S_u or Φ_u) rather than PV forcing (S_q , $S_{q_{tot}}$, or ϕ_q) is not necessary for optimal performance, but may be more robust to variations in other suboptimal design choices (e.g. picking Operators 2 or 3). Operator 1 is more faithful to the numerics of the coarse-resolution model that we are using in the online evaluation (this is further supported by the lack of backscatter generated using ZB2020, see conclusions).

5.4 Relationship between offline and online FCNN metrics

Offline performance, measured using R^2 , is strong for all design choices (see Fig. 9), though its relationship with online performance depends on the filtering and coarse-graining operator. For Operator 1, we see positive correlations between offline and online performance (Fig. 10A, D), meaning that higher R^2 parameterizations generally performed better online. However, for Operators 2 and 3, we see low or negative correlations, meaning that improved offline performance was associated with *worse* rather than better online performance. This result underscores the importance of not focusing too much on improving the offline performance of subgrid parameterizations without first demonstrating that such improvements lead to improvements in physical realism online.

5.5 Varying the evaluation target

Some studies measure the online performance of parameterized low-resolution models with respect to the filtered and coarse-grained version of high-resolution data (Beck et al., 2019; Xie et al., 2020; Guan et al., 2022), rather than the high-resolution simulation. Calculating similarity scores for coarse-resolution parameterized models relative to a coarsened and filtered high-resolution simulation increases the scores of top-performing FCNNs, if the parameterization and the target high-resolution models use the same operator (Figure 11). However, even when the target is coarsened with Operator 2 or 3,

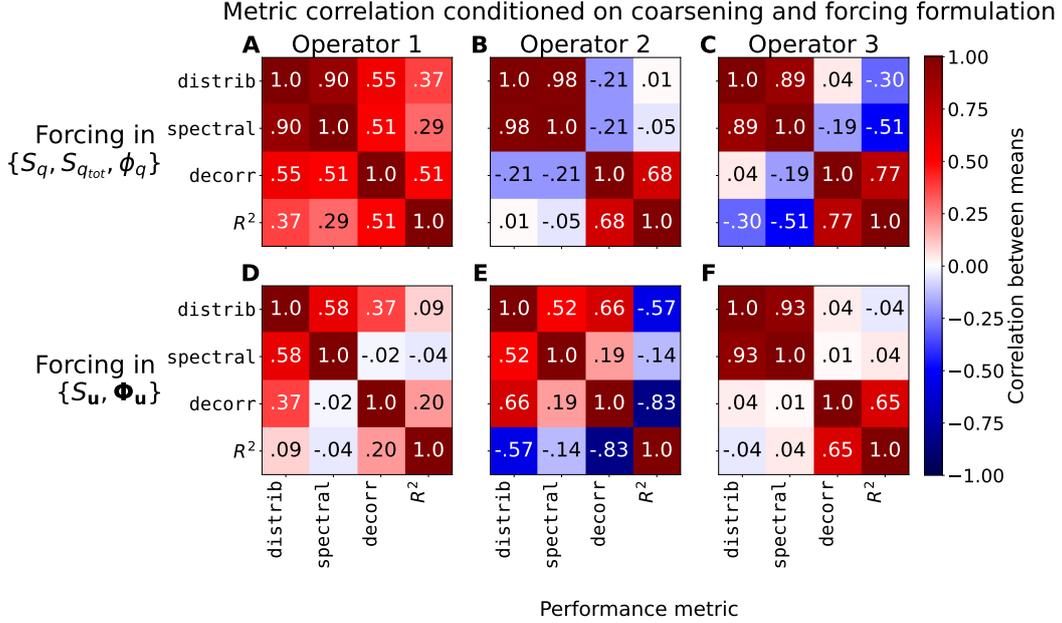


Figure 10. Correlation between FCNNs’ mean scores in each metric group conditioned on the filtering and coarse graining operator (columns) and forcing formulation (rows) used to generate their training data. In most cases, distributional and spectral similarity are closely correlated. Correlations with offline R^2 tend to be negative or small, except for FCNNs trained to predict PV forcing variants computed with Operator 1 (A).

660 the actual scores of these models are significantly lower than in the case where the FCNN
 661 is trained on data generated by Operator 1. The FCNN trained on data generated by
 662 Operator 1 has the best overall spectral similarity score whether we perform the eval-
 663 uation using the original high-resolution data or data coarsened with Operator 1. This
 664 result suggests that Operator 1 is more appropriate for computing subgrid forcing in this
 665 dataset in an absolute sense.

666 5.6 Feature importance for FCNNs

667 To explore the importance of individual features to our FCNN predictions, we look
 668 at snapshots of input gradients, or the partial derivatives of the model’s output with re-
 669 spect to its inputs (Baehrens et al., 2010). Note that although there are many proposed
 670 methods for quantifying neural network input saliency (Springenberg et al., 2014; Bach
 671 et al., 2015), input gradients consistently pass sanity checks that have been developed
 672 to validate these methods, while many alternatives do not (Adebayo et al., 2018; Kin-
 673 dermans et al., 2019).

674 In Fig. 12, we show a snapshot of input gradients for the FCNN($\bar{q} \rightarrow S_{q_{tot}}^{(1)}$) at the
 675 center of the domain, which quantifies the sensitivity of its predictions at this particu-
 676 lar location to its inputs. Although our FCNN architecture allows changes in the upper
 677 layer \bar{q} to influence the lower layer prediction and vice-versa, this particular FCNN’s in-
 678 put gradients are only large in magnitude for \bar{q} in the same layer as the output, suggest-
 679 ing that it largely operates layer-wise.

680 Additionally, gradients were largest in magnitude around the spatial location of the
 681 output, suggesting that the model operates locally in the horizontal plane. However, we

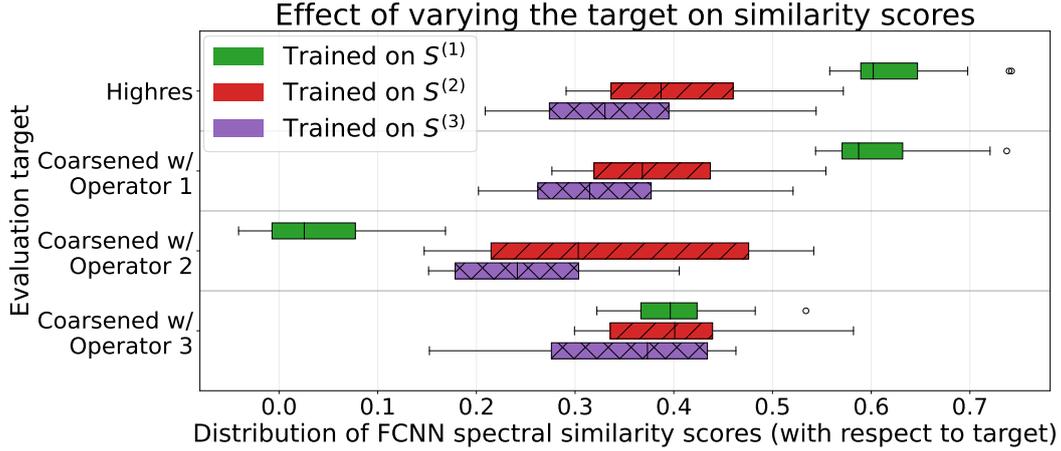


Figure 11. Boxplots showing distribution of `spectral_diff` similarity scores (across all FCNNs trained with each filtering and coarse-graining operator, e.g. $S^{(1)}$ for Operator 1, with any forcing formulation) when the definition of similarity is changed to be relative to a filtered and coarsened version of the high-resolution simulation (bottom three rows), rather than the original high resolution simulation (top row). Center line shows medians, colored bars show the interquartile range (middle 50% of the data), whiskers show positions of nearest points outside twice the interquartile range, and dots show outliers. In general, the relative performance of FCNNs improves when evaluating them against simulations coarsened with the same operator used in their training data. However, absolute performance is only high for FCNNs trained on data from Operator 1 (spectral truncation + maximally sharp filter).

682 find that a radius of 5 pixels (4th-order operations) is needed to explain 50% of the gra-
 683 dients, and a radius of 9 pixels (8th-order operations) is needed to reach 95% (Figure 12
 684 I). This suggests that symbolic parameterizations may need to be fairly non-local and
 685 high-order to mimic the behavior of FCNNs. We explore this in the next section.

686 6 Hybrid Linear and Symbolic Regression and Genetic Programming

687 In addition to opaque models such as neural networks and random forests, it is also
 688 possible to learn equations from data directly with symbolic regression (Koza, 1994).

689 Symbolic-regression based on running sparse linear regression on top of a manu-
 690 ally constructed feature library has become popular and achieved impressive results in
 691 a number of applications (Brunton et al., 2016; Li et al., 2021). Zanna and Bolton (2020)
 692 (ZB2020 hereafter) learned an expression for the subgrid momentum forcing \mathbf{S}_u with sparse
 693 Bayesian regression (see Eq. A7 in Appendix). They used data generated from an ide-
 694 alized primitive equation model, with Gaussian filtering (similar to Operator 2 defined
 695 here). Using data from `pyqg` and Operator 2 to calculate the same basis features as in
 696 ZB2020 (i.e., divergence, vorticity, stretching and deformation, their x - and y -derivatives,
 697 and all cross-multiples), we are able to re-discover Eq. A7 with a simple sparse linear re-
 698 gression algorithm.

699 However, sparse linear regression entails trade-offs between the size and expressive-
 700 ness of the feature library and the complexity and cost of sparse regression, as discussed
 701 in Zanna and Bolton (2020). In the example above, our feature library has the initial
 702 basis features (4 elements), their first spatial derivatives (8 elements), and all cross-multiples
 703 of those initial features (144 elements). If we want to expand this library to consider suc-

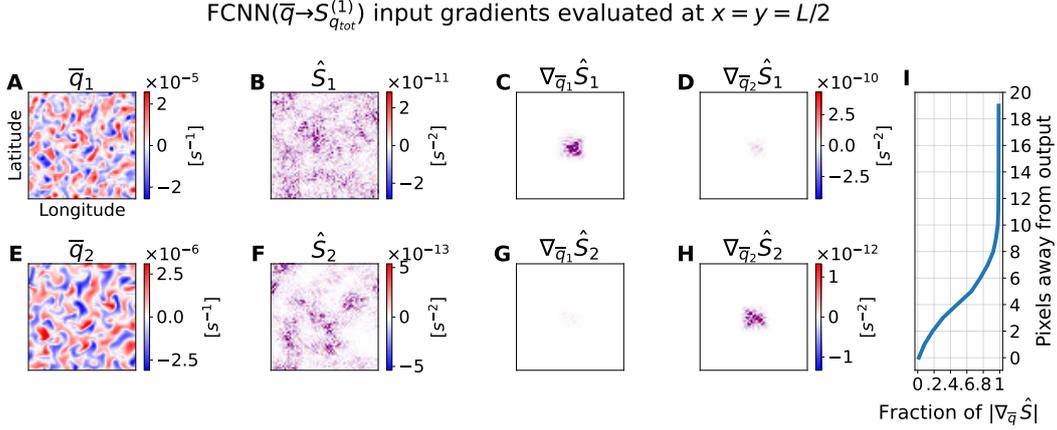


Figure 12. Input gradients of an FCNN mapping \bar{q} (A,E) to $S_{q_{tot}}^{(1)}$ (B,F), evaluated at the center of the domain. Gradient magnitudes are largest around the x, y -position corresponding to the prediction (C,H) in a given layer. However, they still extend relatively far horizontally, needing 9 pixels to reach $>95\%$ of their full magnitude (I).

cessively higher-order derivatives (or more than just linear and quadratic multiples), then the number of different expressions we must evaluate for the whole dataset will grow exponentially. Additionally, many expressions will be highly correlated, which can prevent many sparse regression algorithms from converging (Hastie et al., 2015).

6.1 Hybrid genetic programming (GP)

An alternative approach for symbolic regression is genetic programming (GP), a classic approach in AI (Turing, 1950; Koza, 1994). In contrast to sparse regression, GP algorithms do not require an explicit feature library, simply a set of atomic features and a set of operations for combining them. The GP algorithm then constructs arbitrarily deep expressions by successively applying operators to combine atomic and/or composite features in a randomized fashion, using evolutionary principles to guide a parallel search for an expression that parsimoniously fits the data.

More concretely, GP algorithms begin with a “population” of initially short and randomly-constructed programs. At each iteration (“generation”), programs are randomly culled, with probability inversely related to their relative performance on a “fitness” metric (see Algorithm 1). Programs that survive can then be randomly modified (“mutated”) in a variety of ways, which can either lengthen or shorten them. This procedure is repeated for a configurable number of generations, after which the GP algorithm returns the best-performing program.

To implement genetic programming, we used the `gplearn` Python library (Stephens, 2019). We ran into several difficulties with its default implementation, primarily in its difficulty discovering linear combinations of terms with different orders of magnitude in the weights (constant ranges must be chosen beforehand, and are sampled randomly rather than optimized), as well as the lack of built-in support for spatial differential operators in program evolution. We defined custom `gplearn` functions for differential operators ($\partial/\partial x_i$, ∇^2 , and $\bar{\mathbf{u}} \cdot \nabla$) and combined genetic programming and linear regression in an iterative, residual-fitting procedure described in Algorithm 1. Crucially, in each genetic programming step, we define fitness in terms of correlation rather than absolute error, making fitting the outermost constants unnecessary. We run genetic programming with \bar{q} , \bar{u} , and \bar{v} as our base features. Arbitrary powers or cross-multiples of these features

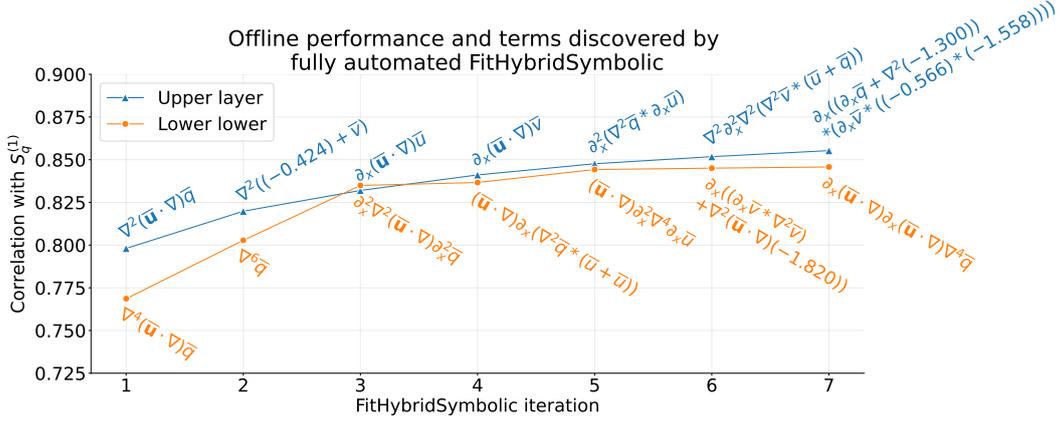


Figure 13. Offline correlation and sequence of terms discovered by Algorithm 1 – Hybrid Symbolic regression without any human-in-the-loop intervention (terms learned for upper/lower layers in blue/orange respectively). Terms learned in initial iterations tended to be physically meaningful, relatively simple, and related to parameterizations in the literature, while terms learned in later iterations tended to be complex or unphysical (e.g., adding \bar{u} and \bar{q} despite incompatible units in iteration 6).

734 can be discovered since the operator set includes multiplication. This approach allows
 735 us to discover all the same terms which appear in the feature library used for ZB2020,
 736 but is not limited to them.

737 Based on results obtained from the FCNNs (Section 5.4), we chose to run our GP
 738 method on PV subgrid forcing, S_q , computed with Operator 1 (Section 3.4.1) to sim-
 739 plify learning. Running Algorithm 1 without any manual experimenter intervention leads
 740 to a formula of the forcing with an expression for each iteration given in Fig. 13. We saw
 741 offline performance increase significantly, with many of the discovered features seemingly
 742 physically-relevant, based on previous published parameterizations. In particular, we note
 743 that $\nabla^2(\bar{u} \cdot \nabla)\bar{q}$ and $\nabla^2\bar{v}$ in the upper layer, together approximate a parameterization
 744 proposed by Porta Mana and Zanna (2014), though missing a Eulerian time derivative
 745 of PV which is not provided to the algorithm. However, terms discovered *after* the first
 746 two iterations tend to vary significantly on random restarts. Terms after the fourth it-
 747 eration are also significantly harder to interpret (see right end of Figure 13). More im-
 748 portantly, we find that some combinations of terms include additions of terms with dif-
 749 ferent units (e.g., q and u). Finally, implementations of parameterizations using the hy-
 750 brid expressions found after the sixth iteration were numerically unstable; in addition,
 751 although the online performance of runs with the first 4-6 terms from the symbolic pa-
 752 rameterizations did improve over low-resolution models, there were still significant dif-
 753 ferences with respect to many high-resolution diagnostics. To address these issues, we
 754 added a human-in-the-loop guidance step described below.

755 6.2 Human-in-the-loop guidance

756 Some manual intervention can be introduced during the learning procedure to im-
 757 prove interpretability and stability. We added a human-in-the-loop guidance step in each
 758 iteration (gray lines in Algorithm 1), where we edited or removed terms that seemed un-
 759 physical and sometimes added what seemed like natural extensions of existing terms. In
 760 our final OptionalUserEdits step, we attempted to prune the set of terms as much as pos-

761 sible by removing those whose removal did not worsen online performance or adding some
 762 that may improve it. We provide an account of our specific actions in Appendix C.

Algorithm 1 “Hybrid” linear and genetic programming-based symbolic regression (with optional human-in-the-loop interventions in light gray).

```

1: procedure FITGENETICPROGRAM( $x, y$ )
2:   Run gplearn (Stephens, 2019) with operators  $\{\partial_x, \partial_y, \nabla^2, (\mathbf{u} \cdot \nabla), *, +\}$ , and
      Fitness(term) =  $|\text{Corr}(\mathbf{term}(x), y)| - 0.001 * \text{Length}(\mathbf{term})$ 
3: end procedure
4:
5: procedure FITLINEARREGRESSION( $x, y$ )
6:   Find  $w$  to minimize  $\|w \cdot x - y\|_2^2$ 
7: end procedure
8:
9: procedure FITHYBRIDSYMBOLIC( $x, y$ )
10:  terms  $\leftarrow \emptyset$  ▷ set of symbolic expressions
11:   $w \leftarrow \emptyset$  ▷ weights of those expressions
12:   $\tilde{y} \leftarrow y$  ▷ residual forcing to predict
13:
14:  repeat
15:    for all layers  $z$  do
16:      terms  $\leftarrow \mathbf{terms} \cup \text{FITGENETICPROGRAM}(x_z, \tilde{y}_z)$  ▷ learn the next term
17:    end for
18:    terms  $\leftarrow \text{OPTIONALUSEREDITS}(\mathbf{terms})$ 
19:    for all layers  $z$  do
20:       $w_z \leftarrow \text{FITLINEARREGRESSION}(\mathbf{terms}(x_z), y_z)$  ▷ reweight terms
21:       $\tilde{y}_z \leftarrow w_z \cdot \mathbf{terms}(x_z) - y_z$  ▷ update residuals
22:    end for
23:  until convergence or user decision
24:
25:  return terms,  $w$ 
26: end procedure

```

763 This procedure left us with a final parameterization of the form:

$$\begin{aligned}
 S_q^{\text{CP}} = & (w_1 \nabla^2 + w_2 \nabla^4 + w_3 \nabla^6)(\bar{\mathbf{u}} \cdot \nabla) \bar{q} \\
 & + (w_4 \nabla^4 + w_5 \nabla^6) \bar{q} \\
 & + (\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2 (w_6 \bar{v}_x + w_7 \bar{u}_y).
 \end{aligned} \tag{20}$$

764 Here w_i signify the linear weights. Evaluating this parameterization against FCNNs and
 765 traditional physics-based models, we find its performance competitive with neural net-
 766 works in the eddy configuration (Figs. 16 and 17) and near-dominant in the jet config-
 767 uration (18 and 19). We discuss its performance further in Section 7.1 where we com-
 768 pare and contrast different categories of parameterizations.

769 6.3 Symbolic regression feature importance

770 As in Section 5.6 for FCNNs, it is useful to quantify the relative importance of the
 771 different symbolic terms. One way to do this is by examining the weights w_i . These are
 772 visualized in Fig. 14 in two ways: (A) as raw values (on a log scale), and (B) normal-
 773 ized after dividing by the standard deviations of the corresponding features (on a lin-
 774 ear scale), which makes them directly comparable despite each w_i having different units.

775 In normalized form (Fig. 14B), the largest coefficients in both layers are for $\nabla^4(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$,
 776 and the absolute magnitudes of these coefficients (Fig. 14A) are somewhat close.
 777 In contrast, the next-largest normalized coefficients in Fig. 14B disagree between layers;
 778 for the upper layer, the next-largest coefficient is for $\nabla^6(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$, while the correspond-
 779 ing value in the lower layer is near zero. Instead, the next-largest coefficients in the lower
 780 layer are for $\nabla^4 q$ and $\nabla^6 q$, which receive much more weight relative to their magnitudes
 781 in the dataset. However, despite the difference in relative weight across layers, the ab-
 782 solute magnitudes of the $\nabla^4 q$ and $\nabla^6 q$ coefficients in Fig. 14A are almost equal. Over-
 783 all, these results suggest that the parameterization learns to behave in reasonably sim-
 784 ilar ways in both layers, but with a few crucial differences, particularly in how they han-
 785 dle $\nabla^6(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$. The final two terms, $(\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2 \bar{v}_x$ and $(\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2 \bar{u}_y$, receive relatively
 786 little (normalized) weight in either layer.

787 Another way to estimate feature importance is by removing each term, re-fitting
 788 the linear regression coefficients, and re-evaluating online performance (Fig. 15). If we
 789 consider the performance decrease after removal of each feature as a measure of its im-
 790 portance, we reach similar conclusions: the ∇^4 and ∇^6 terms (for both \bar{q} and $(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$)
 791 are most important, the $\nabla^2(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$ term is somewhat important, and the $(\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2$ terms
 792 are relatively unimportant.

793 6.4 Interpretation of the learned expression

794 Note that the goal of the paper is not to focus on interpretability but to introduce
 795 methods for learning and evaluating parameterizations from data. Therefore, we are not
 796 claiming that this parameterization is more physical than anti-viscosity backscatter (Jansen
 797 & Held, 2014) or deformation-based parameterizations (Anstey & Zanna, 2017). Nev-
 798 ertheless, we will discuss briefly how the discovered terms compared to other subgrid pa-
 799 rameterizations and leave further analysis of their contribution to model physics to fu-
 800 ture studies.

801 The components of the proposed model were discovered in the following order. In
 802 the first few iterations, quadratic expressions, proportional to $(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$, were discovered.
 803 Quadratic models are often found to be highly-correlated with subgrid forcing (Meneveau
 804 & Katz, 2000; Layton & Rebholz, 2012; Porta Mana & Zanna, 2014; Anstey & Zanna,
 805 2017), but often cannot be used as standalone parameterizations. The next few itera-
 806 tions led to eddy-viscosity models, $\nabla^4 \bar{q}$ and $\nabla^6 \bar{q}$. Particularly, both weights w_4 and w_5
 807 being positive implies that there is dissipation of energy in small scales and redistribu-
 808 tion to larger scales, i.e. backscattering (Jansen & Held, 2014). The final terms discov-
 809 ered are cubic in model variables and contains double-advection operator, $(\bar{\mathbf{u}} \cdot \nabla)^2$. The
 810 terms resemble the anticipated PV method from Vallis and Hua (1988). This method
 811 allows to preserve properties inherent to geostrophic turbulence such as conservation of
 812 energy and dissipation of enstrophy (Marshall & Adcroft, 2010), but it suffers from in-
 813 accurate representation of spectral fluxes (Thuburn et al., 2014). In summary, our dis-
 814 covered closure contains elements of existing subgrid parameterizations, which have pros
 815 and cons when used as standalone ones.

816 This symbolic parameterization includes up to the seventh spatial derivative of \bar{q} ,
 817 which may be unrealistic to implement into a climate model. However, it might be more
 818 realistic than a fully non-local approach such as the convolutional neural network pa-
 819 rameterizations considered in Section 5 or extremely local physics-based parameteriza-
 820 tions (such as anti-viscosity).

821 7 Discussion and Conclusion

822 We will finally compare our top parameterizations and then summarize our key find-
 823 ings in this section.

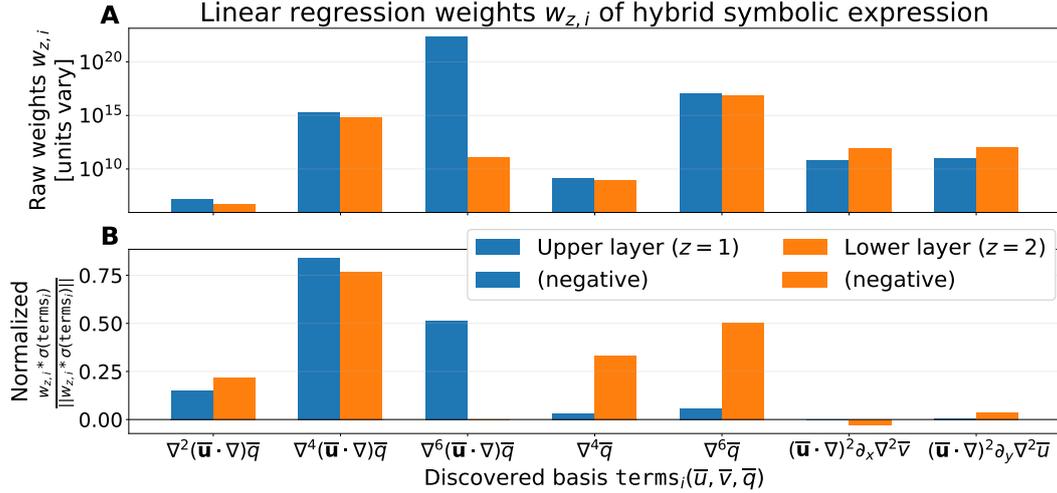


Figure 14. Linear regression-derived weights w for the human-in-the-loop genetic programming-derived basis terms of Equation 20, both as raw values (A, negative values shown with hatching) and normalized (B) after multiplying by the standard deviations of the terms over the training set (giving them consistent units). The absolute magnitudes of many terms are somewhat similar across layers, but their effective contributions to the output differ.

824

7.1 Comparing top parameterizations

825

826

827

828

829

830

831

To conclude our analysis, we focus on the top-performing models of different categories. The Pareto frontier of distributional and spectral similarity (Fig. 8) conveniently includes one FCNN, our symbolic parameterization, and two backscatter parameterizations (we select the one with higher spectral similarity). Note that the Smagorinsky parameterizations have very poor performance online (not surprisingly since they are dissipative) and we strongly encourage the community to choose better physics baselines when evaluating the performance of data-driven parameterizations.

832

833

834

835

836

837

838

839

Offline on eddy configuration (Fig. 16), FCNN performance (A-E) is strongest overall, though power spectra diverge slightly at large scales (E). The symbolic regression model (F-J) performs slightly worse offline than the FCNN, but matches the power spectrum at all scales reasonably well. The backscatter model (K-O) performs much worse offline than the data-driven models, using R^2 as a metric. However, all three selected models perform well online (Fig. 17), with the FCNNs showing better distributional performance than the other models (Fig. 17C). However, the FCNN models seem to spin up the large scale faster than the other models (Fig. 17B).

840

841

842

843

844

845

846

847

On jet configuration, the offline performance remains similar for all models, except for the R^2 of the FCNN in the lower layer which is significantly lower than for the eddy configuration (Fig. 18B). However, online FCNN's performance degrades to significantly worse than the low-resolution without parameterization (Figs. 19 and 20). In addition, the backscatter model does not have a significant impact on the low-resolution simulation, though this depends on which metric we consider (e.g., Fig. 19). On the other hand, the symbolic model remains fairly robust - without retraining or tuning in this new configuration.

848

849

850

FCNNs with different forcing formulations degraded slightly less when transferring to jet configuration (Fig D6). However, their average similarity scores were still low compared to the hybrid symbolic model (Fig. D9), and they disrupted the characteristic jet

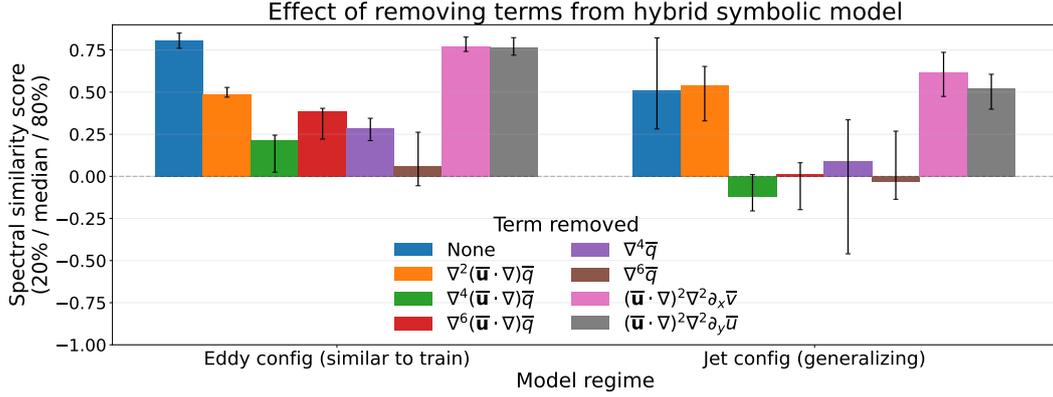


Figure 15. Effect of removing individual terms from the symbolic expression of Equation 20 (using human-in-the-loop guidance) on spectral similarity (median scores within groups, with error bars showing the 20th and 80th percentiles). From left to right, removing $\nabla^2(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$ reduced performance in eddy configuration, but not jet configuration. Removing ∇^4 and ∇^6 terms (for both \bar{q} and $(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$) drastically reduced performance in both configurations, which suggests these terms are crucial. Removing the $(\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2$ terms had small effects, suggesting they could be dropped for future experiments.

851 features, causing the flow to more closely resemble the eddy configuration on which they
 852 were trained (Fig. D3).

853 Even in the eddy configuration, `decorr.times` for the best-performing models are
 854 only modestly closer to those from the high-resolution compared to those of the low-resolution
 855 simulation. In the case of the FCNN, the decorrelation times are actually worse (Fig. 21)
 856 than the low resolution. Using the decorrelation metric, Smagorinsky parameterizations
 857 actually performed best (slightly ahead of certain backscatter settings), even though they
 858 performed near the worst by all other metrics (see also Fig. D8). As expected, the data-
 859 driven parameterizations are doing well at representing the averaged statistics at coarse
 860 resolution (i.e., the climate) but do not improve the short-term trajectories (i.e., the “weather”).

861 7.2 Conclusion

862 We introduced a framework and a set of datasets for learning and evaluating ocean
 863 subgrid forcing parameterizations in a quasi-geostrophic setup, with a focus on a set of
 864 well-defined quantitative offline and online metrics. We used this framework to train and
 865 test physics-based and data-driven parameterizations under a variety of conditions, namely
 866 the different training datasets and definitions of subgrid forcing.

867 Several conclusions stand out as particularly relevant for developing subgrid pa-
 868 rameterizations from high-resolution simulations for climate models, even though some
 869 of the parameterizations developed here cannot easily be implemented in climate mod-
 870 els. We summarize our key points as follows

- 871 • **Metrics:** performance offline and online needs to be rigorously evaluated, rather
 872 than eyeballing improvement over a few selected diagnostics, to determine the ac-
 873 curacy and reliability of a given parameterization or simulation. Here, we designed
 874 multiple level of metrics: offline metrics that captures the statistics of the subgrid
 875 forcing; online metrics that captures the physics of parameterized simulation (e.g.,
 876 kinetic energy flux) or the climatological and short-term performance of the model

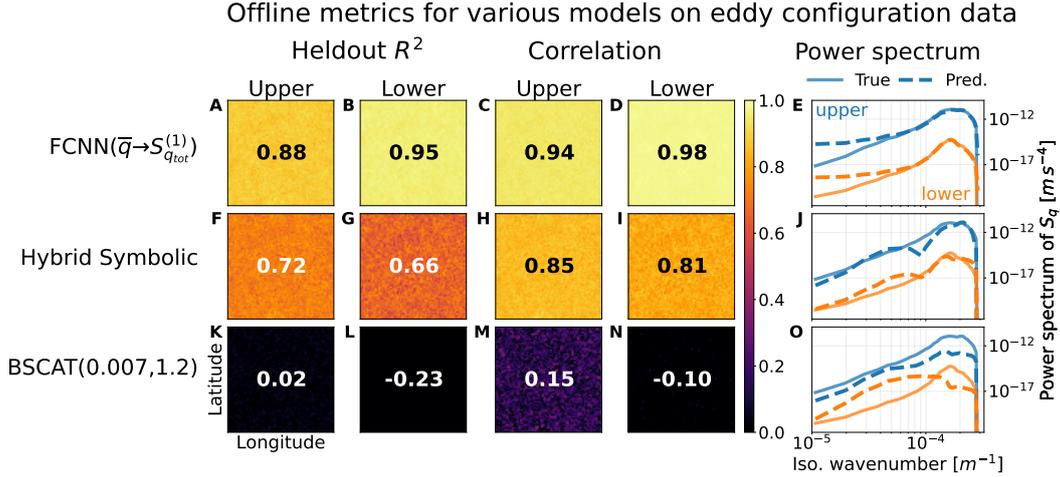


Figure 16. Offline performance for selected subgrid parameterizations on a heldout eddy configuration dataset computed with Operator 1, with means shown in spatial plots. FCNN performance (A-E) is strongest overall, though subgrid power spectra diverge slightly at large scales (E). The symbolic regression (F-J) model performs slightly worse, but matches the power spectrum at all scales reasonably well. The backscatter model (K-O) perform much worse offline (though all three perform well online, Fig. 17).

877 (e.g, climatological PDF of potential vorticity, or decorrelation timescales of short
878 term forecasts, respectively). Our open-source framework (Appendix D) will hope-
879 fully encourage the research community to find easy-to-use resources for such eval-
880 uation and facilitate the development of new parameterizations that more faith-
881 fully capture the effects of subgrid-scale processes.

- 882 • **Data design choice:** the filtering and coarse-graining operator is key, consistent
883 with Zanna and Bolton (2021) and Frezat et al. (2022). The online results for a
884 given FCNN architecture are highly sensitive to filtering choice; here the best per-
885 formance was obtained with a filtering that most closely follow the numerics of
886 the model. Therefore, we encourage testing multiple operators for data prepara-
887 tion guided by the target target application rather than varying hyperparameters
888 or neural network architectures.
- 889 • **Stability:** Our architecturally-constrained FCNNs remained numerically stable
890 in any configuration (as shown in Guillaumin and Zanna (2021) for different model
891 configurations), which is likely further aided by the spectral truncation of high-
892 frequency modes in `pyqg`.
- 893 • **Generalization:** symbolic expressions, found using a new algorithm that we de-
894 veloped, were more interpretable with fewer parameters and generalized better to
895 new domains than neural networks, which are infamously sensitive to even minor
896 distributional shifts (Recht et al., 2018).

897 There are many possible directions we did not explore for NN optimization, includ-
898 ing online learning (Kochkov et al., 2021; Frezat et al., 2022; Sirignano et al., 2020; Um
899 et al., 2020; Dresdner et al., 2022), or training on multiple datasets (Bolton & Zanna,
900 2019; O’Gorman & Dwyer, 2018). New approaches to remain more faithful to the physics
901 of the problem that could be explored as well which include non-dimensionalizing input
902 variables (Beucler et al., 2021), modeling subgrid-scale organization (Shamekh et al., 2022),
903 or finding a better latent space for our input (and eliminating spurious correlation with
904 causal inference). There are also opportunities for improving our symbolic regression pro-

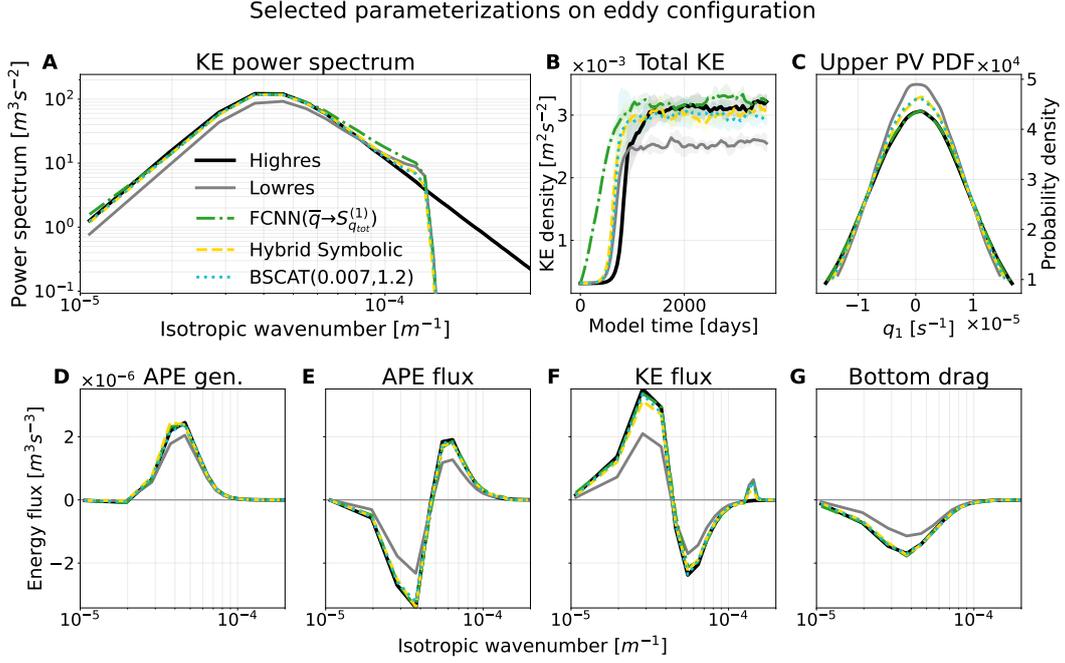


Figure 17. Sample of online performance diagnostics for symbolic regression and best FCNN/backscatter parameterizations by eddy-config `spectral_diff` (taken from the Pareto frontier of top models by spectral and distributional similarity, and averaged across five independent runs). Shading in KE time-series shows standard deviation over runs. All parameterizations improve significantly over the low-resolution model.

905 cedure, including more intelligently interweaving continuous optimization with genetic
 906 programming (Cranmer, 2020), initializing symbolic regression with terms from exist-
 907 ing physical parameterizations, or directly learning residuals on top of them. For both
 908 neural networks and symbolic regression, finding better metrics for offline learning or test-
 909 ing might help ensure more robust results for online implementation in existing legacy
 910 climate models.

911 Appendix A Baseline Local Physical Parameterizations

912 A1 Smagorinsky

913 A common baseline for physical parameterizations was proposed by Smagorinsky
 914 (1963) as scale-selective dissipation. Given the strain-rate tensor, T ,

$$T = \begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix} = \frac{1}{2} \begin{pmatrix} 2\bar{u}_x & \bar{u}_y + \bar{v}_x \\ \bar{u}_y + \bar{v}_x & 2\bar{v}_y \end{pmatrix}, \quad (\text{A1})$$

915 the Smagorinsky parameterization predicts the subgrid forcing of u and v , denoted as
 916 S_{smag} , such that

$$S_{\text{smag}} = \begin{pmatrix} S_{u,\text{smag}} \\ S_{v,\text{smag}} \end{pmatrix} = 2 \begin{pmatrix} (\nu_{\text{smag}} T_{11})_x + (\nu_{\text{smag}} T_{12})_y \\ (\nu_{\text{smag}} T_{21})_x + (\nu_{\text{smag}} T_{22})_y \end{pmatrix}, \quad (\text{A2})$$

917 where the short-hands $()_{x,y} \equiv \frac{\partial}{\partial x,y}$ are used for low-resolution spatial derivatives,

$$\nu_{\text{smag}} = (C_S \Delta x)^2 \sqrt{T_{11}^2 + T_{12}^2 + T_{21}^2 + T_{22}^2}, \quad (\text{A3})$$

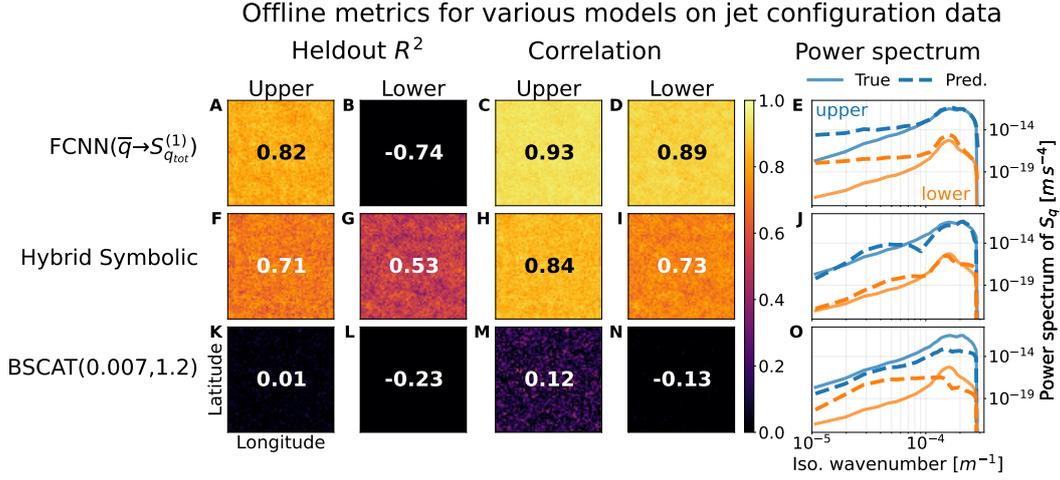


Figure 18. Offline performance as in Figure 18, but testing for generalization to jet configuration. For FCNNs (A-E), R^2 is lower in the upper layer and actually negative in the lower layer. However, correlation remains fairly high, suggesting that performance might improve with rescaling. For our symbolic regression model (F-J) and backscatter (K-O), offline performance remains similar to eddy configuration, though only the hybrid symbolic model generalizes online (Figure 19).

918 and C_S is a tunable parameter. Here we will use $C_S \in \{0.075, 0.15, 0.3\}$.

919 Smagorinsky is a parameterization of small-scale dissipation, which can correct the
 920 tendency of low-resolution models to concentrate too much energy at small scales. How-
 921 ever, the parameterization does not redistribute this energy back up to larger scales via
 922 backscatter, as show in theoretical analysis and simulations of quasi-2D turbulence (Kraichnan,
 923 1976; Thuburn et al., 2014; Natale & Cotter, 2017).

924 A2 Backscatter and Biharmonic Dissipation

925 Different parameterizations that can potentially address backscatter include the
 926 parameterization suggested by Jansen and Held (2014); Jansen et al. (2015), which con-
 927 sists of scale-selective dissipative operator and an additional negative viscosity part rein-
 928 jecting energy at larger scales. The magnitude of the negative viscosity part is chosen
 929 such that resulting model approximately conserves energy.

930 We adapt this parameterizations for use in `pyqg`. The small-scale dissipation of en-
 931 tropy is parameterized with biharmonic Smagorinsky model (see Eq. A3)

$$F_{\text{smag}} = -\nabla^2 [\nu_{\text{smag}} \nabla^4 \bar{\psi}]. \quad (\text{A4})$$

932 The negative viscosity backscatter is parameterized with less scale-selective Laplacian
 933 viscosity operator:

$$F_{\text{bscat}} = -\nu_{\text{bscat}} \nabla^4 \bar{\psi}, \quad (\text{A5})$$

934 and total contribution to PV equation is given as $S_{\text{bscat}} = F_{\text{smag}} + F_{\text{bscat}}$. The nega-
 935 tive viscosity backscatter re-injects the C_B fraction of the total energy dissipated by the
 936 biharmonic model. As such, the negative viscosity coefficient is given by:

$$\nu_{\text{bscat}} = C_B \frac{\sum_{i=1}^2 H_i \iint \bar{\psi}_i F_{\text{smag},i} d\bar{x} d\bar{y}}{\sum_{i=1}^2 H_i \iint \bar{\psi}_i \nabla^4 \bar{\psi}_i d\bar{x} d\bar{y}} \quad (\text{A6})$$

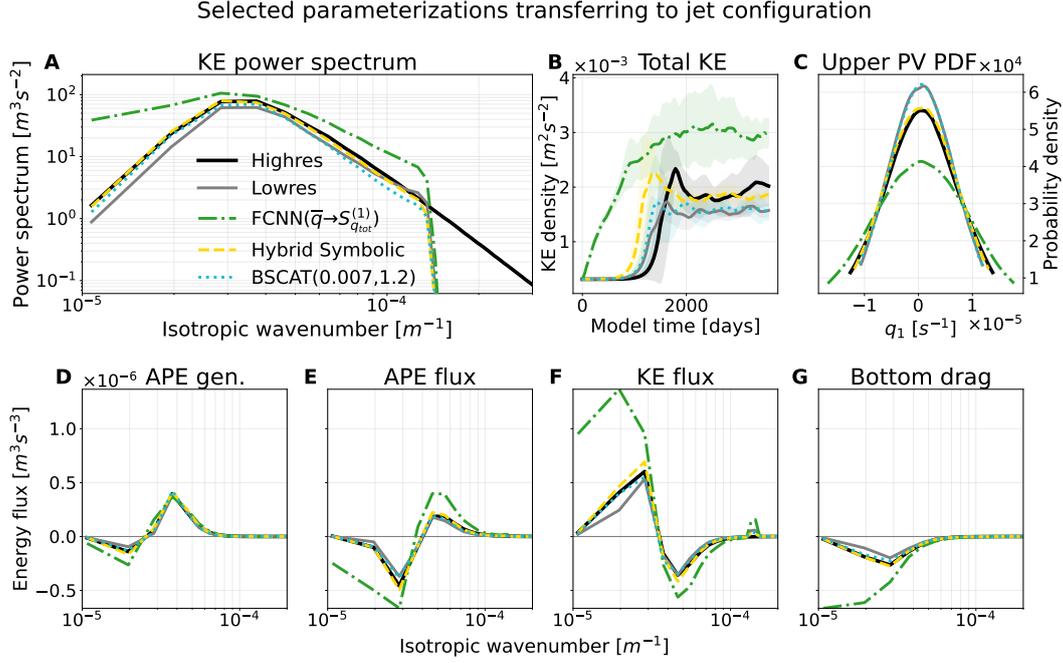


Figure 19. Similar to Figure 17, but evaluated on jet rather than eddy configuration (without retuning). The hybrid symbolic parameterization still improves significantly over low-resolution model, while backscatter has no discernible effect and FCNNs degrade significantly.

937 where $F_{\text{smag},i}$ is the value of Eq. A4 at a particular layer. We run this parameterization
 938 at 36 parameter settings corresponding to every combination of $C_B \in \{.7, .8, .9, 1.0, 1.1, 1.2\}$
 939 and $C_S^2 \in \{.003, .005, .007, .01, .02, .04\}$ (the use of C_S^2 is for convenience).

940 **A3 Zanna Bolton Data-Driven Equation-Discovery parameterization**

941 Using data from an idealized primitive equation model and relevance vector ma-
 942 chine, Zanna and Bolton (2020) learned an expression for the subgrid momentum forc-
 943 ing. They use both barotropic and baroclinic simulated data, and apply Gaussian filter-
 944 ing with coarse-graining to diagnose the subgrid forcing. The form of the parameteri-
 945 zation is given by

$$\hat{\mathbf{S}}_{\mathbf{u}}^{\text{ZB2020}} \approx \kappa^{\text{ZB2020}} \nabla \cdot \begin{pmatrix} -\zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta D \end{pmatrix} + \mathbf{I} \frac{1}{2} \kappa^{\text{ZB2020}} \nabla (\zeta^2 + D^2 + \tilde{D}^2), \quad (\text{A7})$$

946 for each vertical layer, with

$$\zeta = \bar{v}_x - \bar{u}_y, \quad \sigma = \bar{u}_x + \bar{v}_y, \quad (\text{A8a})$$

$$D = \bar{u}_y + \bar{v}_x, \quad \tilde{D} = \bar{u}_x - \bar{v}_y, \quad (\text{A8b})$$

947 where ζ is the relative vorticity, σ is the divergence, and D and \tilde{D} are the shearing and
 948 stretching deformation of the low-resolution flow field, respectively.

949 For online tests, rather than using the value of κ^{ZB2020} diagnosed in Zanna and Bolton
 950 (2020), we fit the parameter empirically to achieve maximal offline R^2 on the training
 951 set (equivalent to that generated using Operator 2). For online simulations, we also test
 952 at $\kappa^{\text{ZB2020}} = 2$ and $1/2$ times the empirically fit value.

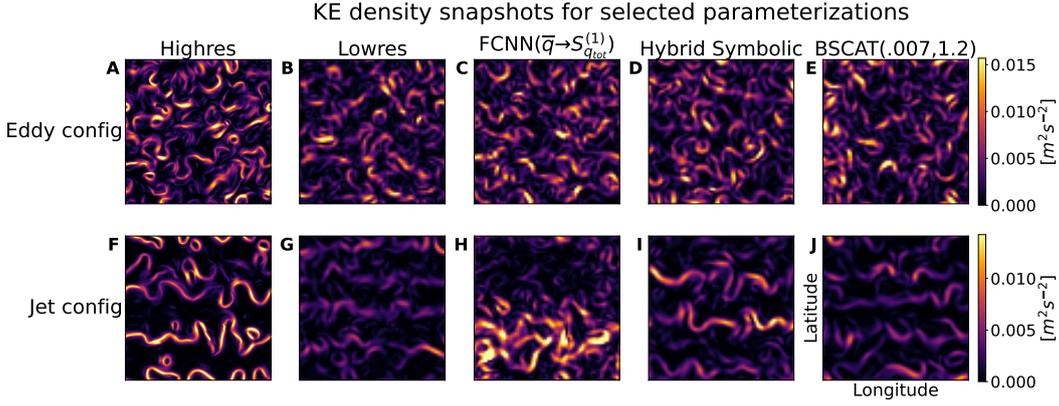


Figure 20. Randomly chosen snapshots of kinetic energy density for selected parameterizations on eddy (A-E) and jet configuration (F-J). On jet configuration, the symbolic parameterization (J) matches high-resolution (F) reasonably well, while the FCNN (I) deviates significantly and backscatter (H) does not appear to have any effect or modify the low-resolution (G). See Figures D1 and D2 for more.

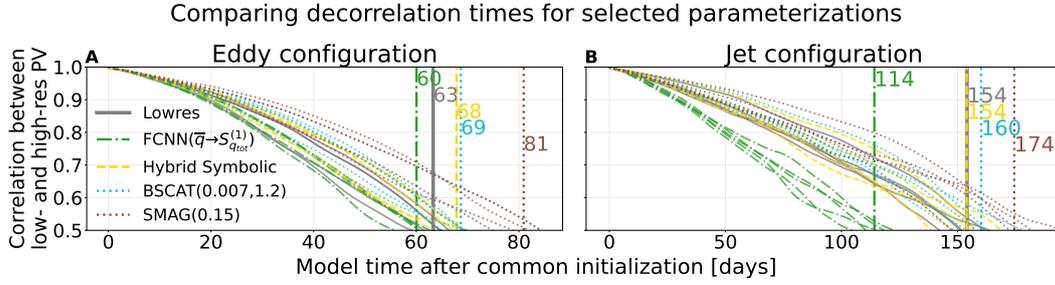


Figure 21. `decorr_time` results for selected parameterizations on eddy (A) and jet (B) configuration. For each parameterization type, vertical lines show average time for five pairs of high- and low-resolution simulations to reach 0.5 correlation after starting at different randomly sampled initial conditions q_0 . FCNNs diverged from high resolution models faster than unparameterized models, while backscatter and hybrid symbolic parameterizations stayed correlated for similar durations. Smagorinsky parameterizations stayed correlated significantly longer.

Appendix B Decomposition of subgrid forcing

We can further decompose subgrid contribution into the contribution towards kinetic energy and the contribution towards potential energy. Let S_ψ be the tendency in the streamfunction induced by subgrid forcing, we use Eq. 3 to rewrite Eq. 12 as

$$\begin{aligned} \left(\frac{\partial E(k, l)}{\partial t}\right)^{\text{sub}} &= -\frac{1}{H} \sum_{m=1}^2 H_m \Re \left[\hat{\psi}_m^* [(-\kappa^2 \mathbf{I} + \mathbf{M}) \hat{\mathbf{S}}_\psi] \right] \\ &= \frac{1}{H} \kappa^2 \sum_{m=1}^2 H_m \Re \left[\hat{\psi}_m^* \left(\mathbf{A}_\kappa \hat{\mathbf{S}}_q \right)_m \right] - \frac{1}{H} \sum_{m=1}^2 H_m \Re \left[\hat{\psi}_m^* \left(\mathbf{M} \mathbf{A}_\kappa \hat{\mathbf{S}}_q \right)_m \right], \end{aligned} \quad (\text{B1})$$

where $\mathbf{A}_\kappa = (-\kappa^2 \mathbf{I} + \mathbf{M})^{-1}$. On the right-hand side of Eq. B1, the first term matches the definition of the contribution towards kinetic energy, and we regard the second term as the contribution towards potential energy. This decomposition is used in calculating the spectral similarity scores.

Appendix C Human-in-the-Loop Symbolic Regression Steps

In this section, we describe the specific “OptionalUserEdits” steps we took in applying Algorithm 1 to obtain Equation 20.

In the first `gplearn` step, we discovered $\nabla^2(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$ (in the upper layer) and $\nabla^4(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$ (in the lower layer), which gave us training set correlations of 0.80 (upper) and 0.77 (lower) after fitting models with both terms to each layer. To this, we added $\nabla^6(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$ to extend the pattern, which brought the same correlations to 0.84 and 0.82. We then ran the next `gplearn` step, which outputted $\nabla^4\bar{q}$ (upper) and $\nabla^6\bar{q}$ (lower). This brought correlations up to 0.845 (upper) and 0.836 (lower). We kept both these terms, and experimented with adding $\nabla^8\bar{q}$, but correlations actually decreased in the lower layer. We then ran the next `gplearn` step, which returned $(\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2 \partial_x \bar{v}$ and $\partial_x \nabla^8 \bar{q}$. This nudged correlations to 0.846 (upper) and 0.838 (lower), which nudged very slightly higher to 0.846 and 0.840 when further adding the counterparts of these terms obtained by switching x and y , $(\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2 \partial_y \bar{u}$ and $\partial_y \nabla^8 \bar{q}$.

From this set of terms (which includes all terms in Equation 20 with the addition of two ninth-order $\partial_i \nabla^8 \bar{q}$ terms), we began a final OptionalUserEdits step using online performance as a guide (removing each term individually, but pairing up the removals of the terms with natural x and y counterparts). In this step, we found that the $\partial_i \nabla^8 \bar{q}$ terms were actually hampering online performance (i.e. performance rose without them), while the others all appeared to help (i.e. performance fell without them)—though our results in Figure 15 later showed that the slight improvement we saw from the $(\bar{\mathbf{u}} \cdot \nabla)^2 \nabla^2 \partial_i \bar{u}_i$ terms was not significant. We then accepted the expression of Equation 20 as our final output, saving its weights (learned with respect to eddy-config $S_q^{(1)}$).

Note that because the genetic programming steps are stochastic, re-running this procedure with a different random seed might produce different results. For example, in Figure 13, we discovered a $\nabla^2 \bar{v}$ term in the second step, but in this case such a term was never learned (though this could be alternately explained by the manual addition of $\nabla^6(\bar{\mathbf{u}} \cdot \nabla)\bar{q}$, which may have accounted for its contribution).

Appendix D Supplementary Figures

This section includes additional result figures.

Open Research

Version 1.0.2 of the Python repository used for training and evaluating parameterizations is preserved at <https://doi.org/10.5281/zenodo.7222704>, available via the MIT license and developed openly at https://github.com/m2lines/pyqg_parameterization_benchmarks (Ross et al., 2022).

The baseline high- and low-resolution datasets used for evaluating parameterizations, as well as the subgrid forcing datasets used for training them, are available at Zenodo via <https://doi.org/10.5281/zenodo.6609034> under a Creative Commons Attribution 4.0 International license (Ross, 2022).

Acknowledgments

This research is supported by the generosity of Eric and Wendy Schmidt by recommendation of Schmidt Futures, as part of its Virtual Earth System Research Institute (VESRI). C.F.G. was partially supported by the NSF DMS grant 2009752. This research was also supported in part through the NYU IT High Performance Computing resources, services, and staff expertise. The authors would like to thank Elizabeth Yankovsky, Ryan Aber-

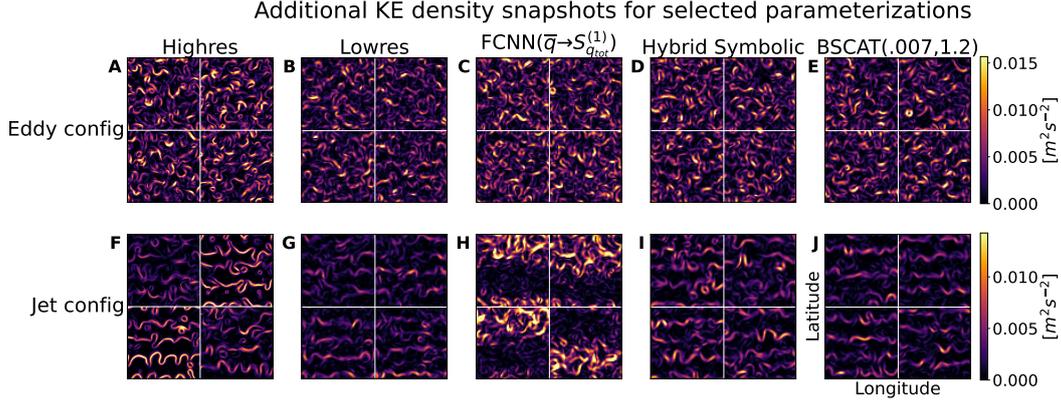


Figure D1. Like Figure 20, but showing additional randomly chosen snapshots.

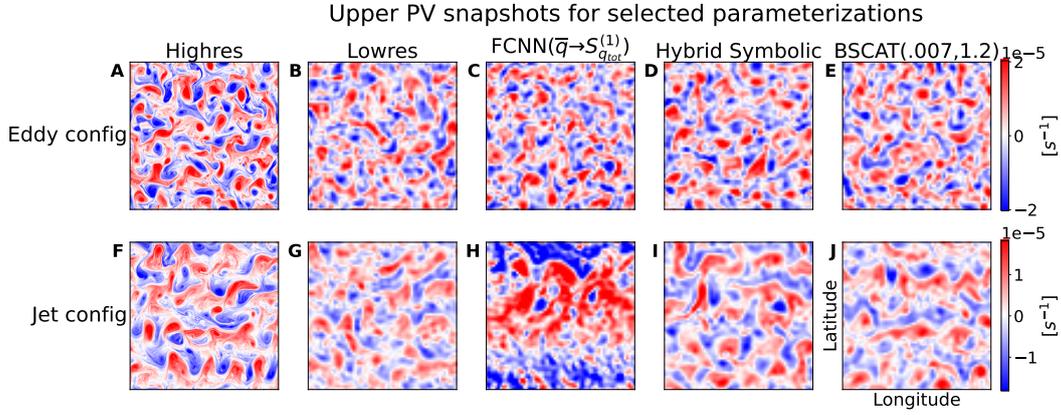


Figure D2. Like Figure 20, but showing upper PV q_1 rather than KE density.

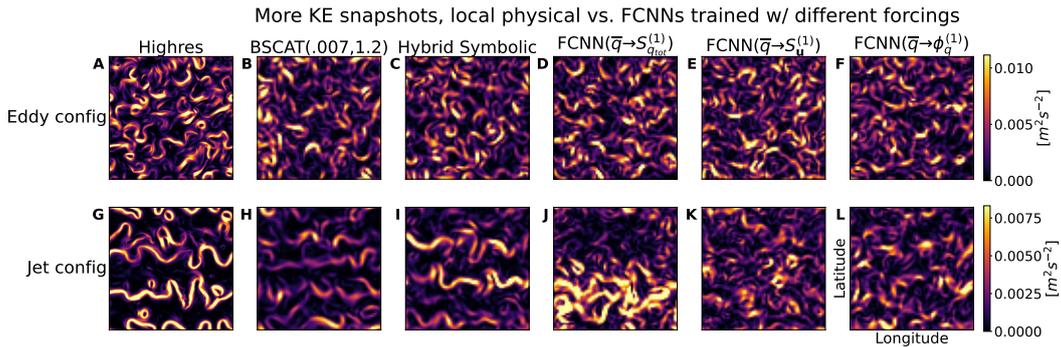


Figure D3. Like Figure 20, but additionally showing KE snapshots for FCNNs trained on eddy configuration data with different forcing formulations (see Figures D5 and D6). All FCNNs produce reasonable results on eddy configuration (D-F), but on jet configuration (J-L), the snapshots do not resemble high-resolution (G), with either latitude-specific increases in energy (J) or disruption of jets in favor of isotropic eddies (K-L), resembling FCNN training conditions.

Comparing FCNNs trained on different operators on eddy configuration

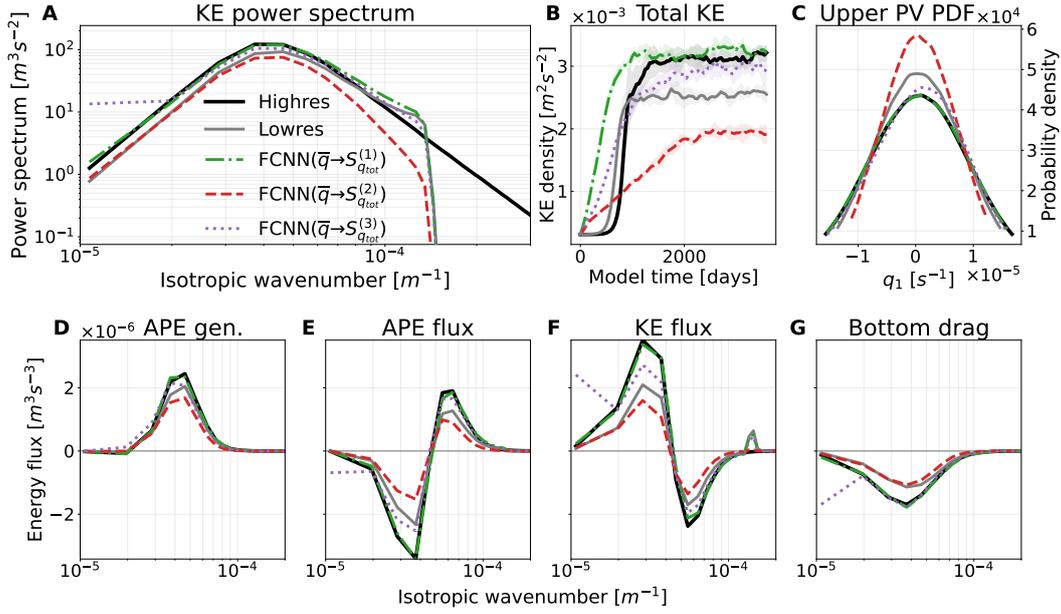


Figure D4. Like Figure 17, but comparing FCNNs trained to predict $S_{q_{tot}}$ computed with each filtering and coarse-graining operator. Only the model trained with Operator 1 (Section 3.4.1) performs near-optimally, though the model trained with Operator 3 (Section 3.4.3) does well except for deviations in spectral metrics at large scales (A,F,G). These results suggest the filtering and coarse-graining operator is important for parameterization performance.

Comparing FCNNs predicting different forcing formulations on eddy configuration

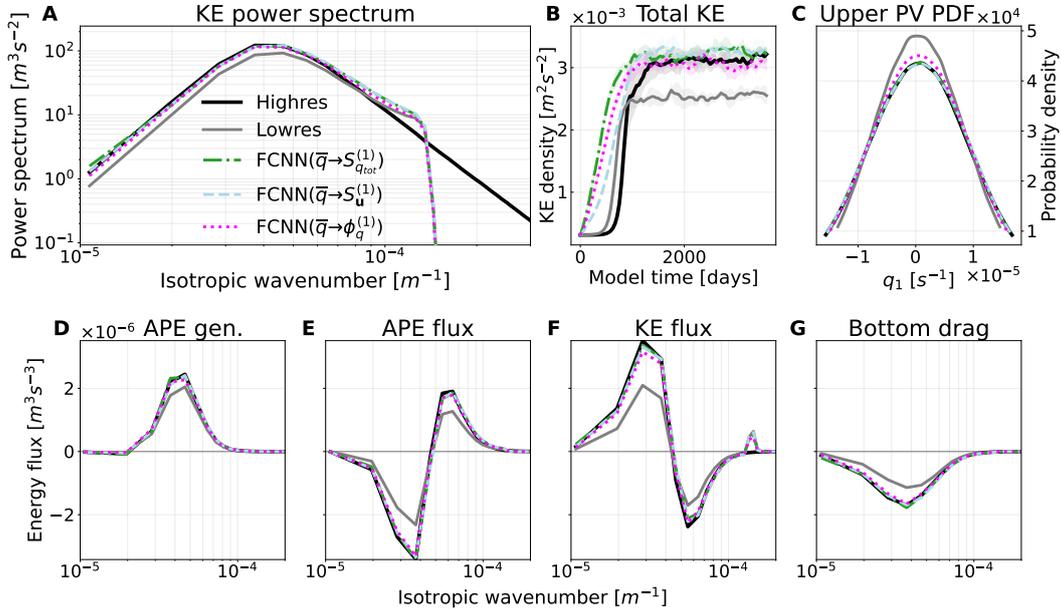


Figure D5. Like Figure 17, but comparing the online eddy configuration performance of FCNNs trained to predict different subgrid forcing formulations ($S_{q_{tot}}$, S_u , and ϕ_q) computed with Operator 1 (Equation 13). All perform almost equally well, suggesting that the forcing formulation may matter much less than the filtering and coarse-graining operator (Figure D4).

Comparing FCNNs predicting different forcing formulations on jet configuration

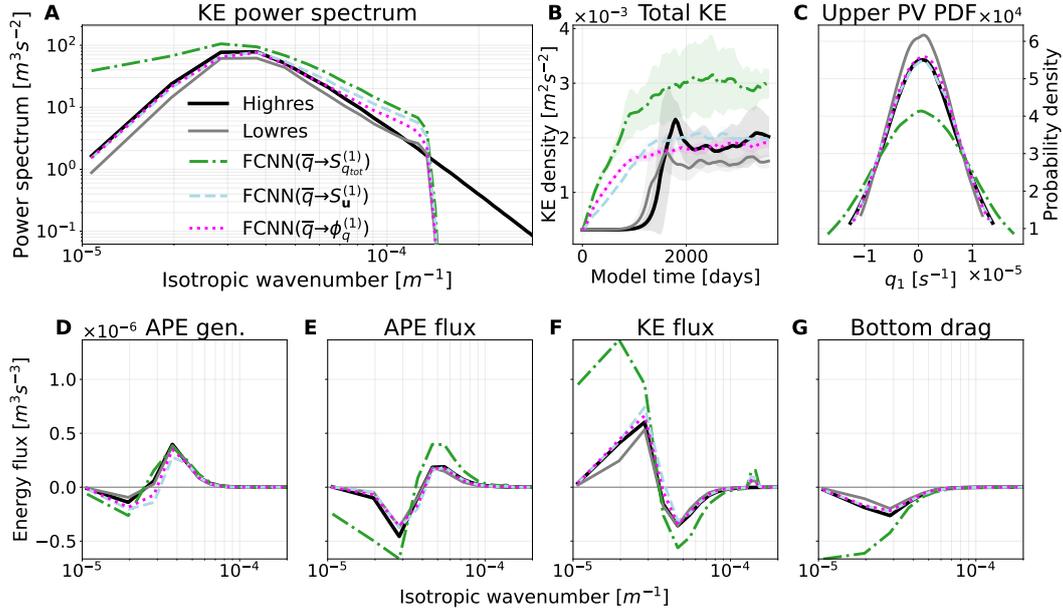


Figure D6. Like Figure 19, but comparing the online jet configuration performance of FCNNs trained to predict different subgrid forcing formulations ($S_{q_{tot}}$, S_u , and ϕ_q) computed with Operator 1 (Equation 13). In this case, the models trained to predict $S_u^{(1)}$ and $\phi_q^{(1)}$ appear to generalize better. However, their average scores across the full set of metrics (e.g. Figure D9) remain low, and in KE snapshots from these FCNNs (Figure D3), the characteristic jet behavior we see in high-resolution is absent.

Offline metrics for FCNNs predicting different outputs on eddy configuration

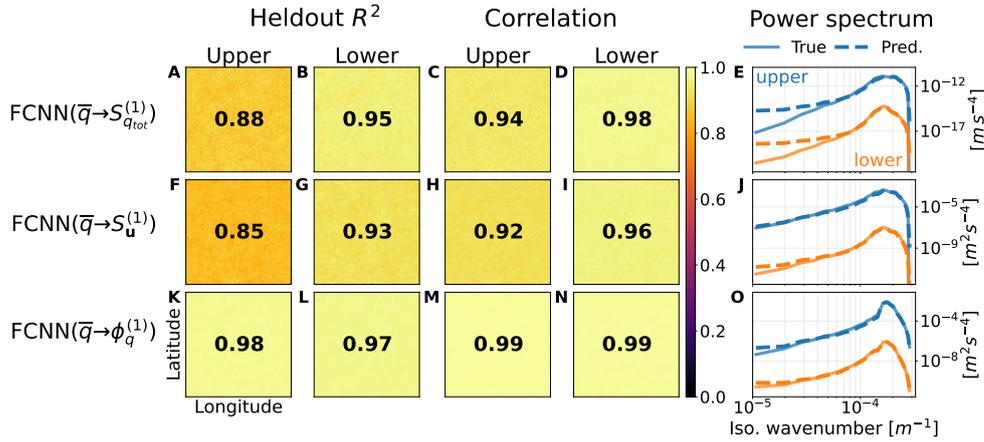


Figure D7. Offline results for more forcing formulations ($S_u^{(1)}$ and $\phi_q^{(1)}$ results show averages over u and v terms). Many performance metrics are generally higher for models trained to predict subgrid fluxes (K-N), but this difference disappears if we compute them with respect to the implied subgrid forcing (i.e. by taking the divergence of the predicted quantities and comparing that to the true subgrid forcing, rather than comparing predicted to true subgrid fluxes).

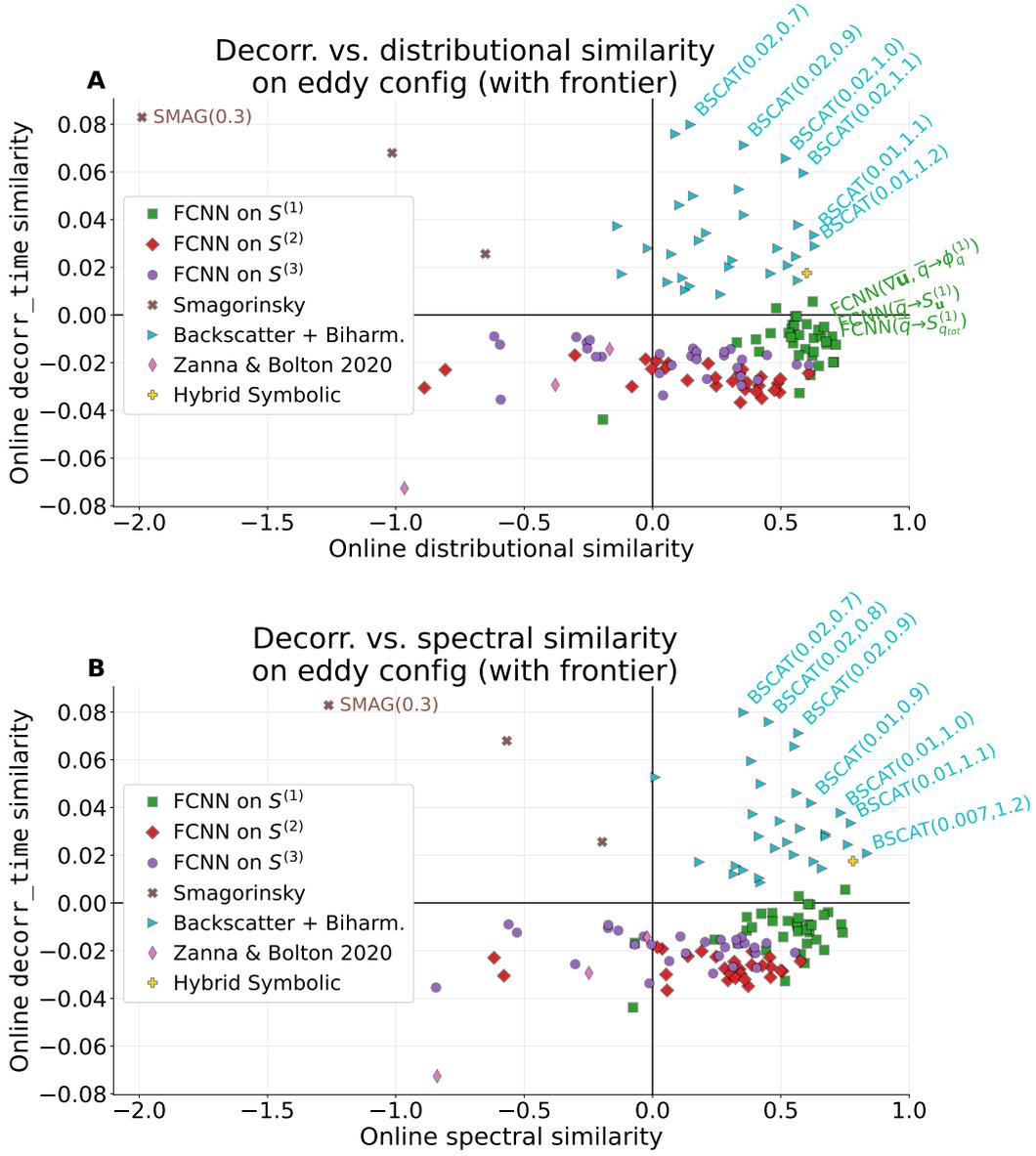


Figure D8. Like Figure 8, but comparing distributional similarity from Section 4.2.2 (A) and spectral similarity from Section 4.2.1 (B) with decorrelation time similarity from Section 4.2.3. Smagorinsky and backscatter parameterizations (which form most of the Pareto frontier in both plots) increase decorrelation time, though only by about 8% of the gap between low- and high-resolution decorrelation times (which is what the y -axis signifies). Neural networks almost universally reduce it, while the hybrid symbolic parameterization modestly increases it.

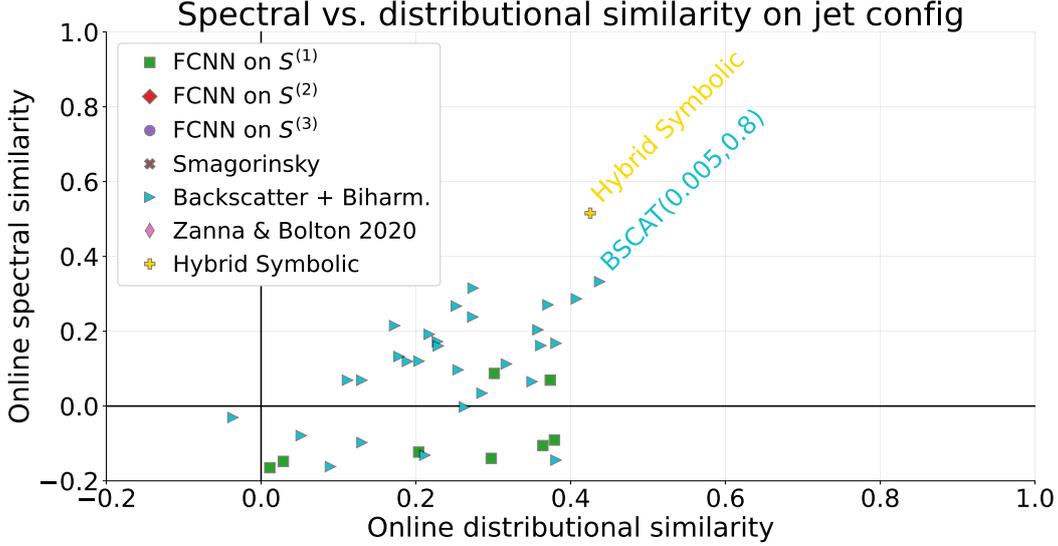


Figure D9. Like Figure 8, but evaluated on jet configuration. In this regime, the only models which appear on the Pareto frontier (highlighted in text) are the hybrid symbolic model and one parameter setting of the backscatter parameterization, which differs significantly from the eddy-configuration Pareto-optimal settings shown in Figures 8 and D8.

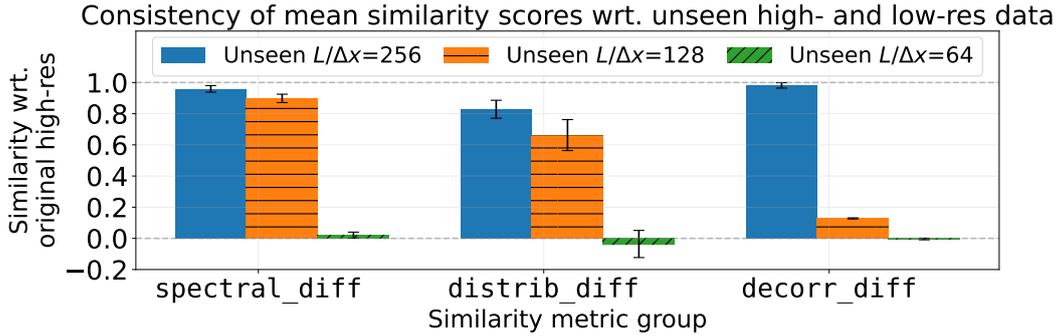


Figure D10. Mean similarity scores for *unseen* high-resolution ($L/\Delta x=256$), low-resolution ($L/\Delta x=64$), and intermediate-resolution ($L/\Delta x=128$) simulations with respect to the actual high- and low-resolution datasets used to evaluate parameterizations. Error-bars show means and standard deviations over 10 random samples of 5 simulations from a set of 25 unseen simulations. Spectral and decorrelation time similarity scores between different randomly re-run high-res simulations are >0.95 on average (and ≤ 0.01 on average for unseen low-resolution simulations), indicating they are fairly reliable (they should be near 1 for high-res and near 0 for low-res). End-of-simulation distributional similarity scores are a bit noisier, averaging 0.83 for unseen high-resolution simulations (so such scores in our results of above ≈ 0.8 are potentially near-optimal). Although distributional similarity scores are still precise enough to provide meaningful insight into parameterization performance, future experiments could improve their precision by increasing the size of ensembles, or by comparing distributions marginalized over more than just the final timestep. Finally, $L/\Delta x=128$ simulations score highly (closer to $L/\Delta x=256$) on distributional and spectral similarity, indicating convergence on long-term “climate” predictions. However, they score much worse (closer to $L/\Delta x=64$) on decorrelation time similarity, suggesting that short-term “weather” predictions are more sensitive to changes in resolution.

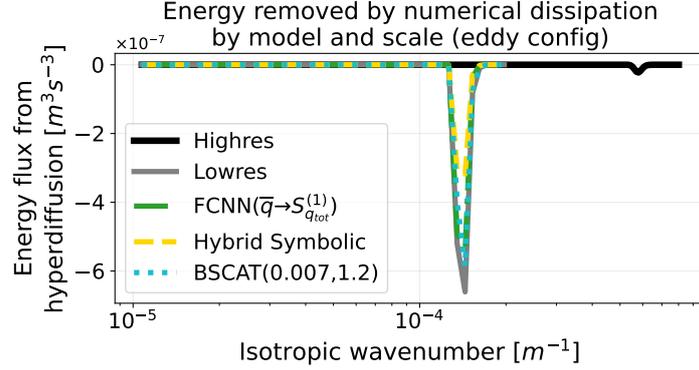


Figure D11. Comparison of the energy density removed via numerical dissipation at each scale for different parameterizations and resolutions (on eddy configuration, with spectra averaged over 5 simulations). Although the definition of the dissipative term is identical at each resolution, the actual amount of energy dissipated varies in practice due to how parameterizations change the distribution of quantities across scales. In this case, parameterized models lose less energy to numerical dissipation than unparameterized models at the same resolution, likely because the purpose of those parameterizations is to transport energy to larger scales.

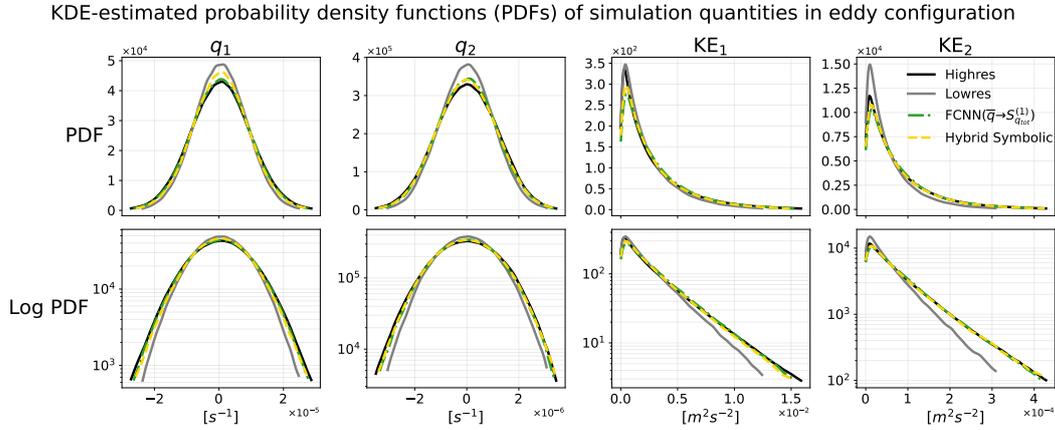


Figure D12. Probability density functions (PDFs), calculated using kernel density estimates (KDE), in both real (top) and logarithmic (bottom) space of upper and lower PV and KE for selected parameterizations (as compared to unparameterized baselines, and computed via kernel density estimation). The selected parameterizations cause these quantities to match the high-resolution simulation much more closely, even in the tails of the distribution (e.g. far right sides of log PDF plots).

Simulation Type	Train Time	Runtime
Neural Network (CPU)	—	2hrs 53min
Neural Network (GPU)	52min	13min 4s
High-res	N/A	5min 57s
Hybrid Symbolic	35min	2min 22s
Backscatter + Biharmonic	N/A	59s
Low-res	N/A	22s

Table D1. Wall clock times to train parameterizations (center) and run simulations (right) for different simulation types; i.e., for single runs on a Tesla V100 for GPUs and an M1 MacBook Pro for CPUs. The FCNN slows down the low-res simulations (Zanna & Bolton, 2020), even when utilizing a GPU. The low-res simulations with FCNN are twice as slow as the high-res; the slow down is primarily due to the depth of the neural network. The symbolic regression-parameterized simulations are more than twice as fast as the high-res simulations. Lower-order backscatter parameterizations are $>2x$ as fast again (though still $<\frac{1}{2}x$ the speed of unparameterized low-res simulations). This is consistent with Zanna & Bolton, 2020.

Model	Eddy configuration		Jet configuration	
	Rank by Σ	Rank by Π	Rank by Σ	Rank by Π
FCNN($\bar{q} \rightarrow S_{tot}^{(1)}$)	1	—	129	—
BSCAT(0.02,1.0)	28	1	31	—
Hybrid Symbolic	4	11	1	—
BSCAT(0.005,0.8)	74	23	2	1

Table D2. “Leaderboard” of models with the highest arithmetic means (Σ) and geometric means (Π) over our three score categories (average distributional, spectral, and decorrelation time similarity; i.e. the axes of the Pareto frontier plots in Figures 8, D8, and D9) in both eddy and jet configuration. Off-diagonal elements show each model’s ranking by score reductions where it is not optimal, and “—” indicates that a negative similarity score was found (so the geometric mean is not meaningful). The hybrid symbolic model ranks highly by all score reductions except its jet configuration geometric mean (where its decorrelation time similarity is slightly negative).

1006 nathey, Adam Subel, and Tom Beucler for helpful conversations and suggestions; and
1007 three reviewers for their comments which helped improved the manuscript.

1008 References

- 1009 Abernathey, R., Rocha, C. B., Ross, A., Jansen, M., Li, Z., Poulin, F. J., ...
1010 Tobias (2022, May). *pyqg/pyqg: v0.7.2*. Zenodo. Retrieved from
1011 <https://doi.org/10.5281/zenodo.6563667> doi: 10.5281/zenodo.6563667
1012 Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., & Kim, B. (2018).
1013 Sanity checks for saliency maps. *Advances in neural information processing*
1014 *systems*, 31.
1015 Anstey, J. A., & Zanna, L. (2017). A deformation-based parametrization of ocean
1016 mesoscale eddy reynolds stresses. *Ocean Modelling*, 112, 99–111.
1017 Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., & Samek, W.
1018 (2015). On pixel-wise explanations for non-linear classifier decisions by layer-
1019 wise relevance propagation. *PLoS one*, 10(7), e0130140.
1020 Baehrens, D., Schroeter, T., Harmeling, S., Kawanabe, M., Hansen, K., & Müller,

- 1021 K.-R. (2010). How to explain individual classification decisions. *The Journal*
1022 *of Machine Learning Research*, 11, 1803–1831.
- 1023 Beck, A., Flad, D., & Munz, C.-D. (2019). Deep neural networks for data-driven les
1024 closure models. *Journal of Computational Physics*, 398, 108910.
- 1025 Beck, A., & Kurz, M. (2021). A perspective on machine learning methods in turbu-
1026 lence modeling. *GAMM-Mitteilungen*, 44(1), e202100002.
- 1027 Berloff, P., Ryzhov, E., & Shevchenko, I. (2021). On dynamically unresolved oceanic
1028 mesoscale motions. *Journal of Fluid Mechanics*, 920.
- 1029 Berloff, P. S. (2005). Random-forcing model of the mesoscale oceanic eddies. *Journal*
1030 *of Fluid Mechanics*, 529, 71–95. doi: 10.1017/S0022112005003393
- 1031 Beucler, T., Pritchard, M. S., Yuval, J., Gupta, A., Peng, L., Rasp, S., . . . Gentine,
1032 P. (2021). Climate-invariant machine learning. *CoRR*, abs/2112.08440.
- 1033 Bolton, T., & Zanna, L. (2019). Applications of Deep Learning to Ocean Data In-
1034 ference and Subgrid Parameterization. *Journal of Advances in Modeling Earth*
1035 *Systems*, 11(1), 376–399. doi: 10.1029/2018MS001472
- 1036 Brenowitz, N. D., & Bretherton, C. S. (2018). Prognostic validation of a neural net-
1037 work unified physics parameterization. *Geophysical Research Letters*, 45(12),
1038 6289–6298. doi: <https://doi.org/10.1029/2018GL078510>
- 1039 Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2016). Sparse identification of nonlin-
1040 ear dynamics with control (sindyc). *IFAC-PapersOnLine*, 49(18), 710–715.
- 1041 Champion, K., Lusch, B., Kutz, J. N., & Brunton, S. L. (2019). Data-driven
1042 discovery of coordinates and governing equations. *Proceedings of the*
1043 *National Academy of Sciences*, 116(45), 22445–22451. Retrieved from
1044 <https://www.pnas.org/doi/abs/10.1073/pnas.1906995116> doi:
1045 10.1073/pnas.1906995116
- 1046 Cranmer, M. (2020, September). *Pysr: Fast & parallelized symbolic regression in*
1047 *python/julia*. Zenodo. Retrieved from [http://doi.org/10.5281/zenodo](http://doi.org/10.5281/zenodo.4041459)
1048 [.4041459](http://doi.org/10.5281/zenodo.4041459) doi: 10.5281/zenodo.4041459
- 1049 Dresdner, G., Kochkov, D., Norgaard, P., Zepeda-Núñez, L., Smith, J. A., Brenner,
1050 M. P., & Hoyer, S. (2022). Learning to correct spectral methods for simulating
1051 turbulent flows. *arXiv preprint arXiv:2207.00556*.
- 1052 Fox, D. G., & Orszag, S. A. (1973). Pseudospectral approximation to two-
1053 dimensional turbulence. *Journal of Computational Physics*, 11(4), 612–619.
- 1054 Fox-Kemper, B., Adcroft, A., Böning, C. W., Chassignet, E. P., Curchitser, E.,
1055 Danabasoglu, G., . . . Yeager, S. G. (2019). Challenges and prospects in
1056 ocean circulation models. *Frontiers in Marine Science*, 6. doi: 10.3389/
1057 fmars.2019.00065
- 1058 Fox-Kemper, B., Bachman, S. D., Pearson, B. C., & Reckinger, S. J. (2014). Princi-
1059 ples and advances in subgrid modelling for eddy- rich simulations..
- 1060 Frezat, H., Sommer, J. L., Fablet, R., Balarac, G., & Lguensat, R. (2022). A poste-
1061 riori learning for quasi-geostrophic turbulence parametrization. *arXiv preprint*
1062 *arXiv:2204.03911*.
- 1063 Gallet, B., & Ferrari, R. (2021). A quantitative scaling theory for meridional
1064 heat transport in planetary atmospheres and oceans. *AGU Advances*, 2(3),
1065 e2020AV000362.
- 1066 Gent, P. R., & McWilliams, J. C. (1990). Isopycnal mixing in ocean circulation mod-
1067 els. *Journal of Physical Oceanography*, 20(1), 150 - 155. doi: 10.1175/1520-
1068 -0485(1990)020(0150:IMIOCM)2.0.CO;2
- 1069 Gent, P. R., Willebrand, J., McDougall, T. J., & McWilliams, J. C. (1995). Pa-
1070 rameterizing eddy-induced tracer transports in ocean circulation models. *Jour-
1071 nal of Physical Oceanography*, 25(4), 463 - 474. doi: 10.1175/1520-0485(1995)
1072 025(0463:PEITTI)2.0.CO;2
- 1073 Griffies, S., Adcroft, A., Hewitt, H., Oning, C., Chassignet, E., Danabasoglu, G., . . .
1074 Lab, D. (2009, 01). Problems and prospects in large-scale ocean circulation
1075 models. *OceanObs09 Proceedings*.

- 1076 Griffies, S., Winton, M., Anderson, W. G., Benson, R., Delworth, T. L., Dufour,
 1077 C. O., ... Zhang, R. (2015). Impacts on ocean heat from transient mesoscale
 1078 eddies in a hierarchy of climate models. *Journal of Climate*, 28(3), 952 - 977.
 1079 Retrieved from [https://journals.ametsoc.org/view/journals/clim/28/3/](https://journals.ametsoc.org/view/journals/clim/28/3/jcli-d-14-00353.1.xml)
 1080 [jcli-d-14-00353.1.xml](https://journals.ametsoc.org/view/journals/clim/28/3/jcli-d-14-00353.1.xml) doi: 10.1175/JCLI-D-14-00353.1
- 1081 Grooms, I., Loose, N., Abernathy, R., Steinberg, J., Bachman, S. D., Marques, G.,
 1082 ... Yankovsky, E. (2021). Diffusion-based smoothers for spatial filtering of
 1083 gridded geophysical data. *Journal of Advances in Modeling Earth Systems*,
 1084 13(9), e2021MS002552.
- 1085 Grooms, I., Nadeau, L.-P., & Smith, K. S. (2013). Mesoscale eddy energy locality in
 1086 an idealized ocean model. *Journal of physical oceanography*, 43(9), 1911–1923.
- 1087 Guan, Y., Chattopadhyay, A., Subel, A., & Hassanzadeh, P. (2022). Stable a poste-
 1088 riori les of 2d turbulence using convolutional neural networks: Backscattering
 1089 analysis and generalization to higher re via transfer learning. *Journal of Com-
 1090 putational Physics*, 458, 111090.
- 1091 Guillaumin, A. P., & Zanna, L. (2021). Stochastic-deep learning parameterization
 1092 of ocean momentum forcing. *Journal of Advances in Modeling Earth Systems*,
 1093 n/a(n/a), e2021MS002534. Retrieved from [https://agupubs.onlinelibrary](https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021MS002534)
 1094 [.wiley.com/doi/abs/10.1029/2021MS002534](https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021MS002534) (e2021MS002534
 1095 2021MS002534) doi: <https://doi.org/10.1029/2021MS002534>
- 1096 Hallberg, R. (2013). Using a resolution function to regulate parameterizations of
 1097 oceanic mesoscale eddy effects. *Ocean Modelling*, 72, 92–103.
- 1098 Hastie, T., Tibshirani, R., & Wainwright, M. (2015). Statistical learning with spar-
 1099 sity. *Monographs on statistics and applied probability*, 143, 143.
- 1100 Hewitt, H. T., Roberts, M., Mathiot, P., Biastoch, A., Blockley, E., Chassignet,
 1101 E. P., ... others (2020). Resolving and parameterising the ocean mesoscale in
 1102 earth system models. *Current Climate Change Reports*, 6(4), 137–152.
- 1103 Hornik, K., Stinchcombe, M., & White, H. (1989). Multilayer feedforward net-
 1104 works are universal approximators. *Neural Networks*, 2, 359–366. (arXiv:
 1105 1011.1669v3 ISBN: 08936080 (ISSN)) doi: 10.1016/0893-6080(89)90020-8
- 1106 Jansen, M. F., & Held, I. M. (2014). Parameterizing subgrid-scale eddy effects using
 1107 energetically consistent backscatter. *Ocean Modelling*, 80, 36–48.
- 1108 Jansen, M. F., Held, I. M., Adcroft, A., & Hallberg, R. (2015). Energy budget-based
 1109 backscatter in an eddy permitting primitive equation model. *Ocean Modelling*,
 1110 94, 15–26.
- 1111 Kent, J., Jablonowski, C., Thuburn, J., & Wood, N. (2016). An energy-conserving
 1112 restoration scheme for the shallow-water equations. *Quarterly Journal of the
 1113 Royal Meteorological Society*, 142(695), 1100–1110.
- 1114 Khani, S., & Porté-Agel, F. (2017). Evaluation of non-eddy viscosity subgrid-scale
 1115 models in stratified turbulence using direct numerical simulations. *European
 1116 Journal of Mechanics - B/Fluids*, 65, 168-178. doi: [https://doi.org/10.1016/j](https://doi.org/10.1016/j.euromechflu.2017.03.009)
 1117 [.euromechflu.2017.03.009](https://doi.org/10.1016/j.euromechflu.2017.03.009)
- 1118 Killworth, P. D. (1997). On the parameterization of eddy transfer part i. theory.
 1119 *Journal of Marine Research*, 55(6), 1171–1197.
- 1120 Kindermans, P.-J., Hooker, S., Adebayo, J., Alber, M., Schütt, K. T., Dähne, S., ...
 1121 Kim, B. (2019). The (un) reliability of saliency methods. In *Explainable ai:
 1122 Interpreting, explaining and visualizing deep learning* (pp. 267–280). Springer.
- 1123 Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., & Hoyer, S.
 1124 (2021). Machine learning–accelerated computational fluid dynamics. *Pro-
 1125 ceedings of the National Academy of Sciences*, 118(21), e2101784118.
- 1126 Koza, J. R. (1994). Genetic programming as a means for programming computers by
 1127 natural selection. *Statistics and computing*, 4(2), 87–112.
- 1128 Kraichnan, R. H. (1976). Eddy viscosity in two and three dimensions. *Journal of At-
 1129 mospheric Sciences*, 33(8), 1521–1536.
- 1130 Krasnopolsky, V. M., Fox-Rabinovitz, M. S., Hou, Y. T., Lord, S. J., & Belochit-

- 1131 ski, A. A. (2010). Accurate and fast neural network emulations of model
 1132 radiation for the ncep coupled climate forecast system: Climate simulations
 1133 and seasonal predictions. *Monthly Weather Review*, *138*(5), 1822 - 1842. Re-
 1134 trieved from [https://journals.ametsoc.org/view/journals/mwre/138/5/](https://journals.ametsoc.org/view/journals/mwre/138/5/2009mwr3149.1.xml)
 1135 [2009mwr3149.1.xml](https://journals.ametsoc.org/view/journals/mwre/138/5/2009mwr3149.1.xml) doi: 10.1175/2009MWR3149.1
- 1136 Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The annals of*
 1137 *mathematical statistics*, *22*(1), 79–86.
- 1138 Layton, W. J., & Rebholz, L. G. (2012). *Approximate deconvolution models of tur-*
 1139 *bulence: analysis, phenomenology and numerical analysis* (Vol. 2042). Springer
 1140 Science & Business Media.
- 1141 Li, H., Zhao, Y., Wang, J., & Sandberg, R. D. (2021). Data-driven model develop-
 1142 ment for large-eddy simulation of turbulence using gene-expression program-
 1143 ing. *Physics of Fluids*, *33*(12), 125127.
- 1144 Loose, N., Abernathy, R., Grooms, I., Busecke, J., Guillaumin, A., Yankovsky,
 1145 E., . . . Martin, P. (2022). Gcm-filters: A python package for diffusion-
 1146 based spatial filtering of gridded data. *Journal of Open Source Software*,
 1147 *7*(70), 3947. Retrieved from <https://doi.org/10.21105/joss.03947> doi:
 1148 [10.21105/joss.03947](https://doi.org/10.21105/joss.03947)
- 1149 Lund, T. (1997). On the use of discrete filters for large eddy simulation. *Annual Re-*
 1150 *search Briefs*, 83–95.
- 1151 Marques, G. M., Loose, N., Yankovsky, E., Steinberg, J. M., Chang, C.-Y., Bhamidi-
 1152 pati, N., . . . others (2022). Neverworld2: An idealized model hierarchy to
 1153 investigate ocean mesoscale eddies across resolutions. *Geoscientific Model*
 1154 *Development*, *15*(17), 6567–6579.
- 1155 Marshall, D. P., & Adcroft, A. J. (2010). Parameterization of ocean eddies: Po-
 1156 tential vorticity mixing, energetics and arnold’s first stability theorem. *Ocean*
 1157 *Modelling*, *32*(3-4), 188–204.
- 1158 Maulik, R., San, O., Rasheed, A., & Vedula, P. (2019). Subgrid modelling for two-
 1159 dimensional turbulence using neural networks. *Journal of Fluid Mechanics*,
 1160 *858*, 122–144.
- 1161 Meneveau, C., & Katz, J. (2000). Scale-invariance and turbulence models for large-
 1162 eddy simulation. *Annual Review of Fluid Mechanics*, *32*(1), 1–32.
- 1163 Mojgani, R., Chattopadhyay, A., & Hassanzadeh, P. (2021). Closed-form discovery
 1164 of structural errors in models of chaotic systems by integrating bayesian sparse
 1165 regression and data assimilation. *arXiv preprint arXiv:2110.00546*.
- 1166 Monge, G. (1781). Mémoire sur la théorie des déblais et des remblais. *Mem. Math.*
 1167 *Phys. Acad. Royale Sci.*, 666–704.
- 1168 Natale, A., & Cotter, C. J. (2017). Scale-selective dissipation in energy-conserving
 1169 finite-element schemes for two-dimensional turbulence. *Quarterly Journal of*
 1170 *the Royal Meteorological Society*, *143*(705), 1734–1745.
- 1171 O’Gorman, P. A., & Dwyer, J. G. (2018). Using Machine Learning to Parameterize
 1172 Moist Convection: Potential for Modeling of Climate, Climate Change, and
 1173 Extreme Events. *Journal of Advances in Modeling Earth Systems*, *10*(10),
 1174 2548–2563. doi: 10.1029/2018MS001351
- 1175 Orszag, S. A. (1971). On the elimination of aliasing in finite-difference schemes
 1176 by filtering high-wavenumber components. *Journal of Atmospheric Sciences*,
 1177 *28*(6), 1074–1074.
- 1178 Pawar, S., San, O., Rasheed, A., & Vedula, P. (2020). A priori analysis on deep
 1179 learning of subgrid-scale parameterizations for kraichnan turbulence. *Theoretic-*
 1180 *ical and Computational Fluid Dynamics*, *34*(4), 429–455.
- 1181 Piomelli, U., Moin, P., & Ferziger, J. H. (1988). Model consistency in large eddy
 1182 simulation of turbulent channel flows. *The Physics of Fluids*, *31*(7), 1884-1891.
 1183 doi: 10.1063/1.866635
- 1184 Pope, S. B. (2000). *Turbulent flows*. Cambridge university press.
- 1185 Porta Mana, P., & Zanna, L. (2014). Toward a stochastic parameterization of ocean

- 1186 mesoscale eddies. *Ocean Modelling*, 79, 1-20. doi: 10.1016/j.ocemod.2014.04
 1187 .002
- 1188 Rasp, S., Pritchard, M. S., & Gentine, P. (2018). Deep learning to represent subgrid
 1189 processes in climate models. *Proceedings of the National Academy of Sciences*,
 1190 115(39), 9684–9689. doi: 10.1073/pnas.1810286115
- 1191 Recht, B., Roelofs, R., Schmidt, L., & Shankar, V. (2018). Do cifar-10 classifiers
 1192 generalize to cifar-10? *arXiv preprint arXiv:1806.00451*.
- 1193 Redi, M. H. (1982). Oceanic isopycnal mixing by coordinate rotation. *Journal of*
 1194 *Physical Oceanography*, 12(10), 1154–1158.
- 1195 Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for
 1196 biomedical image segmentation. In *International conference on medical image*
 1197 *computing and computer-assisted intervention* (pp. 234–241).
- 1198 Rosenblatt, M. (1956). Remarks on some nonparametric estimates of a density func-
 1199 tion. *The Annals of Mathematical Statistics*, 27(3), 832–837.
- 1200 Ross, A. S. (2022, June). *pygg subgrid forcing datasets*. Zenodo. Retrieved from
 1201 <https://doi.org/10.5281/zenodo.6609035> doi: 10.5281/zenodo.6609035
- 1202 Ross, A. S., Zanna, L., & Perezhugin, P. (2022, October).
 1203 *m2lines/pygg-parameterization-benchmarks: v1.0.2*. Zenodo. Retrieved from
 1204 <https://doi.org/10.5281/zenodo.7222704> doi: 10.5281/zenodo.7222704
- 1205 Rubner, Y., Tomasi, C., & Guibas, L. J. (2000). The earth mover’s distance as a
 1206 metric for image retrieval. *International journal of computer vision*, 40(2), 99–
 1207 121.
- 1208 Rudy, S. H., Brunton, S. L., Proctor, J. L., & Kutz, J. N. (2017). Data-driven dis-
 1209 covery of partial differential equations. *Science Advances*, 3(4), e1602614. Re-
 1210 trieved from <https://www.science.org/doi/abs/10.1126/sciadv.1602614>
 1211 doi: 10.1126/sciadv.1602614
- 1212 Sagaut, P. (2006). *Large eddy simulation for incompressible flows: an introduction*.
 1213 Springer Science & Business Media.
- 1214 Sagaut, P., & Grohens, R. (1999). Discrete filters for large eddy simulation. *Interna-*
 1215 *tional Journal for Numerical Methods in Fluids*, 31(8), 1195–1220.
- 1216 Salmon, R. (1980). Baroclinic instability and geostrophic turbulence. *Geo-*
 1217 *physical & Astrophysical Fluid Dynamics*, 15(1), 167-211. doi: 10.1080/
 1218 03091928008241178
- 1219 Schmidt, M., & Lipson, H. (2009). Distilling free-form natural laws from experimen-
 1220 tal data. *Science*, 324(5923), 81-85. doi: 10.1126/science.1165893
- 1221 Schneider, T., Teixeira, J., Bretherton, C. S., Brient, F., Pressel, K. G., Schär, C., &
 1222 Siebesma, A. P. (2017). Climate goals and computing the future of clouds. *Na-*
 1223 *ture Climate Change*, 7(1), 3–5. Retrieved from [https://doi.org/10.1038/
 1224 nclimate3190](https://doi.org/10.1038/nclimate3190) doi: 10.1038/nclimate3190
- 1225 Shamekh, S., Lamb, K. D., Huang, Y., & Gentine, P. (2022). Implicit learning of
 1226 convective organization explains precipitation stochasticity. *Earth and Space*
 1227 *Science Open Archive*, 16. Retrieved from [https://doi.org/10.1002/essoar](https://doi.org/10.1002/essoar.10512517.1)
 1228 .10512517.1 doi: 10.1002/essoar.10512517.1
- 1229 Shevchenko, I., & Berloff, P. (2021). On a minimum set of equations for parame-
 1230 terisations in comprehensive ocean circulation models. *Ocean Modelling*, 168,
 1231 101913.
- 1232 Sirignano, J., MacArt, J. F., & Freund, J. B. (2020). Dpm: A deep learning pde
 1233 augmentation method with application to large-eddy simulation. *Journal of*
 1234 *Computational Physics*, 423, 109811.
- 1235 Smagorinsky, J. (1963). General circulation experiments with the primitive equa-
 1236 tions: I. the basic experiment. *Monthly weather review*, 91(3), 99–164.
- 1237 Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2014). Striving for
 1238 simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.
- 1239 Stephens, T. (2019). *Genetic Programming in Python, with a scikit-learn inspired*
 1240 *API*. Retrieved from <https://gplearn.readthedocs.io/en/stable>

- 1241 Stevens, B., & Bony, S. (2013). What are climate models missing? *Science*, *340*,
1242 1053–1054. doi: 10.1126/science.1237554
- 1243 Stoffer, R., Van Leeuwen, C. M., Podareanu, D., Codreanu, V., Veerman, M. A.,
1244 Janssens, M., . . . Van Heerwaarden, C. C. (2021). Development of a large-
1245 eddy simulation subgrid model based on artificial neural networks: a case
1246 study of turbulent channel flow. *Geoscientific Model Development*, *14*(6),
1247 3769–3788.
- 1248 Subel, A., Chattopadhyay, A., Guan, Y., & Hassanzadeh, P. (2021). Data-driven
1249 subgrid-scale modeling of forced burgers turbulence using deep learning with
1250 generalization to higher reynolds numbers via transfer learning. *Physics of*
1251 *Fluids*, *33*(3), 031702.
- 1252 Subel, A., Guan, Y., Chattopadhyay, A., & Hassanzadeh, P. (2022). Explaining the
1253 physics of transfer learning a data-driven subgrid-scale closure to a different
1254 turbulent flow. *arXiv preprint arXiv:2206.03198*.
- 1255 Thuburn, J., Kent, J., & Wood, N. (2014). Cascades, backscatter and conserva-
1256 tion in numerical models of two-dimensional turbulence. *Quarterly Journal of*
1257 *the Royal Meteorological Society*, *140*(679), 626–638.
- 1258 Turing, A. (1950, 10). I.—COMPUTING MACHINERY AND INTELLIGENCE.
1259 *Mind*, *LIX*(236), 433-460. Retrieved from [https://doi.org/10.1093/mind/](https://doi.org/10.1093/mind/LIX.236.433)
1260 [LIX.236.433](https://doi.org/10.1093/mind/LIX.236.433) doi: 10.1093/mind/LIX.236.433
- 1261 Um, K., Brand, R., Fei, Y. R., Holl, P., & Thuerey, N. (2020). Solver-in-the-loop:
1262 Learning from differentiable physics to interact with iterative pde-solvers. *Ad-*
1263 *vances in Neural Information Processing Systems*, *33*, 6111–6122.
- 1264 Vallis, G. K. (2017). *Atmospheric and oceanic fluid dynamics*. Cambridge University
1265 Press.
- 1266 Vallis, G. K., & Hua, B.-l. (1988). Eddy viscosity of the anticipated potential vortic-
1267 ity method. *Journal of Atmospheric Sciences*, *45*(4), 617–627.
- 1268 Xie, C., Wang, J., & Weinan, E. (2020). Modeling subgrid-scale forces by spatial ar-
1269 tificial neural networks in large eddy simulation of turbulence. *Physical Review*
1270 *Fluids*, *5*(5), 054606.
- 1271 Xing, H., Zhang, J., Ma, W., & Wen, D. (2022). Using gene expression programming
1272 to discover macroscopic governing equations hidden in the data of molecular
1273 simulations. *Physics of Fluids*, *34*(5), 057109. doi: 10.1063/5.0090134
- 1274 Yankovsky, E., Zanna, L., & Smith, K. S. (2022). Influences of mesoscale ocean ed-
1275 dies on flow vertical structure in a resolution-based model hierarchy. *Journal of*
1276 *Advances in Modeling Earth Systems*, e2022MS003203.
- 1277 Yuval, J., O’Gorman, P. A., & Hill, C. N. (2021). Use of neural networks for
1278 stable, accurate and physically consistent parameterization of subgrid at-
1279 mospheric processes with good performance at reduced precision. *Geophys-*
1280 *ical Research Letters*, *48*(6), e2020GL091363. Retrieved from [https://](https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020GL091363)
1281 agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2020GL091363
1282 (e2020GL091363 2020GL091363) doi: <https://doi.org/10.1029/2020GL091363>
- 1283 Yuval, J., & O’Gorman, P. A. (2021). Neural-network parameterization of sub-
1284 grid momentum transport in the atmosphere. *Earth and Space Science Open*
1285 *Archive*, *15*. doi: 10.1002/essoar.10507557.1
- 1286 Zanna, L., Bachman, S., & Jansen, M. (2020). Energizing turbulence closures in
1287 ocean models. *CLIVAR Exchanges/US CLIVAR Variations*, *18*(1), 3–8. doi:
1288 10.5065/g8w0-fy32.
- 1289 Zanna, L., & Bolton, T. (2020). Data-driven equation discovery of ocean
1290 mesoscale closures. *Geophysical Research Letters*, e2020GL088376. doi:
1291 10.1029/2020GL088376
- 1292 Zanna, L., & Bolton, T. (2021). Deep learning of unresolved turbulent ocean pro-
1293 cesses in climate models. In *Deep learning for the earth sciences* (p. 298-306).
1294 John Wiley & Sons, Ltd. doi: <https://doi.org/10.1002/9781119646181.ch20>
- 1295 Zhang, S., & Lin, G. (2018). Robust data-driven discovery of governing physical

1296 laws with error bars. *Proceedings of the Royal Society A: Mathematical, Physical*
1297 *and Engineering Sciences*, 474(2217), 20180305. Retrieved from [https://](https://royalsocietypublishing.org/doi/abs/10.1098/rspa.2018.0305)
1298 royalsocietypublishing.org/doi/abs/10.1098/rspa.2018.0305 doi: 10
1299 .1098/rspa.2018.0305
1300 Zhou, Z., He, G., Wang, S., & Jin, G. (2019). Subgrid-scale model for large-eddy
1301 simulation of isotropic turbulent flows using an artificial neural network. *Com-*
1302 *puters & Fluids*, 195, 104319. doi: <https://doi.org/10.1016/j.compfluid.2019>
1303 .104319