Reliable coarse-grained turbulent simulations through combined offline learning and neural emulation

Christian Pedersen [1 2] **Laure Zanna** [1 2] **Joan Bruna** [1 2] **Pavel Perezhogin** [1]

Abstract

Integration of machine learning (ML) models of unresolved dynamics into numerical simulations of fluid dynamics has been demonstrated to improve the accuracy of coarse resolution simulations. However, when trained in a purely offline mode, integrating ML models into the numerical scheme can lead to instabilities. In the context of a 2D, quasi-geostrophic turbulent system, we demonstrate that including an additional network in the loss function, which emulates the state of the system into the future, produces offline-trained ML models that capture important subgrid processes, with improved stability properties.

1. Introduction

The modelling of turbulent flows is a ubiquitous challenge across many areas of science and engineering. Numerical simulations are always limited to some extent by computational cost. This is of particular concern in the case of turbulent fluids, in which there is significant interaction between length scales. Therefore coarse, but computationally affordable simulations, are often missing important processes that occur below the gridscale.

A widely used approach to tackle this problem is the Large Eddy Simulation (LES) methodology (Lesieur et al., 2005; Zhiyin, 2015). In this framework, the partial differential equation (PDE) describing the system is smoothed and coarse-grained to a computationally affordable resolution. An additional term referred to as a *subgrid forcing* is added to the coarse resolution PDE which intends to represent the influence of the missing, subgrid dynamics on the resolved field. Models that provide this subgrid forcing term as a function of the resolved field are known as *parameterizations*.

We focus on the LES formulation in the context of modelling turbulent ocean eddies (*mesoscale* eddies) that play a vital role in climate dynamics (Ferrari & Wunsch, 2009). At current computational limitations, these eddies are only partially resolved in coupled climate models (Haarsma et al., 2016), and so including representations of their effects on the resolved field is an essential part of modern climate simulations (Fox-Kemper & Menemenlis, 2008). Historically, parameterizations have been constructed via a physical understanding of the unresolved dynamics, such as energy dissipation (Smagorinsky, 1963; Leith, 1996) and energy transfer from small to large scales (Jansen & Held, 2014).

Recent years have seen significant work applying machine learning (ML) methods to the problem of turbulence modelling, and more generally solving high-dimensional PDEs. This can take many forms: one setup is to fully learn the system in a purely data-driven approach (Sanchez-Gonzalez et al., 2020; Stachenfeld et al., 2021). Alternatively, the classical numerical solver can be augmented (Bar-Sinai et al., 2019; Kochkov et al., 2021; Sirignano et al., 2020) or replaced by ML (Han et al., 2018; Sirignano & Spiliopoulos, 2018). In the context of the LES framework, ML methods have been used to construct parameterizations turbulent fluids, both in a supervised (Maulik et al., 2018; Beck et al., 2019) and reinforcement learning setting (Novati et al., 2021). For the modelling of ocean turbulence, parameterizations using convolutional neural networks (CNNs) (Lecun et al., 1998) have been trained and successfully incorporated into simulations, forming a hybrid physical + ML model (Bolton & Zanna, 2019; Perezhogin et al., 2023).

ML parameterizations can be treated in two categories. In an offline learning setting, a dataset is constructed by running a high-resolution simulation that resolves the target dynamics. These high-resolution fields are smoothed and coarse-grained to obtain low-resolution fields. A subgrid forcing term relating the high- and low-resolution fields can then be calculated (see appendix A for details). The ML task is then to model the relationship between

[*]Equal contribution [1]Courant Institute of Mathematical Sciences, New York University [2]Center for Data Science, New York University. Correspondence to: Christian Pedersen <c.pedersen@nyu.edu>.

Accepted after peer-review at the 1st workshop on Synergy of Scientific and Machine Learning Modeling, SynS & ML ICML, Honolulu, Hawaii, USA. July, 2023.

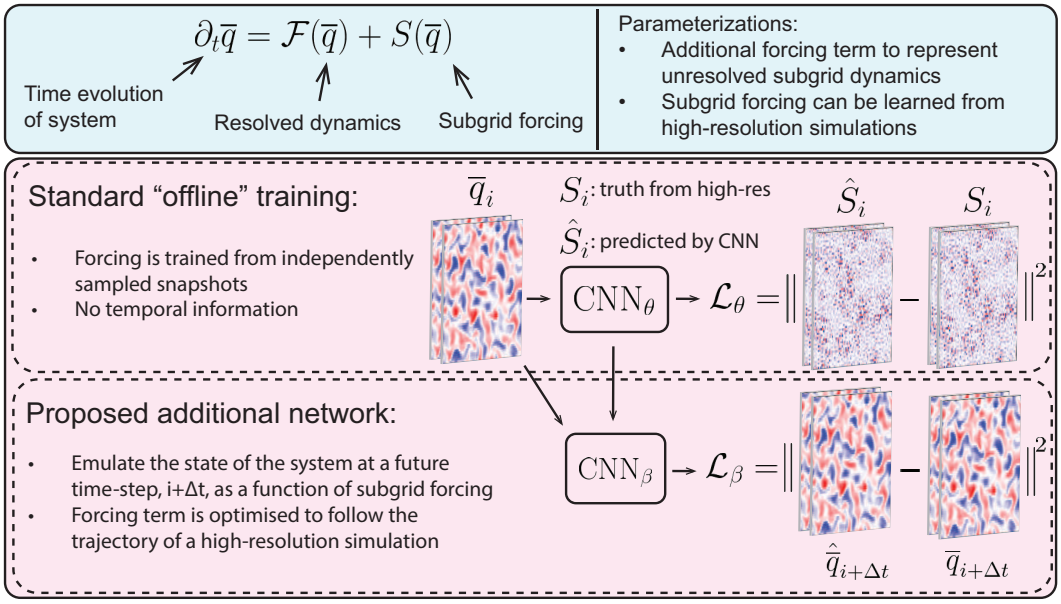

Figure 1. Overview of the optimisation scheme. A subgrid forcing term, S_i, is calculated from a dataset of high-resolution simulations. The standard offline learning framework is to train a CNN to predict this term as a function of the resolved field, $\bar{q}_i$. However this framework contains no information about the temporal evolution of the system. We include an additional network that projects the system forward in time by Δt. During optimisation of the subgrid model, Φ_θ, we include this network in the loss function as a regularisation term, and which constrains the system to follow the trajectory taken by a high-resolution simulation.

a low-resolution field, and the subgrid forcing obtained from high-resolution simulations. However, it has been shown that offline parameterizations often lead to stability issues when implemented in coarse resolution simulations (Maulik et al., 2018; Frezat et al., 2022; Guan et al., 2023). Some approaches to improving stability of purely offline parameterizations are the inclusion of physics-informed regularisations (Guan et al., 2023), or stochasticity (Guillaumin & Zanna, 2021; Perezhogin et al., 2023). In the case of fully-learned systems, stochasticity is also required to produce stable rollouts (Stachenfeld et al., 2021).

An alternative approach is to integrate the ML model within the numerical scheme as a learned correction to each timestep, and train the model over multiple timesteps. This approach is known as online learning, and has been demonstrated to produce better stability properties than purely offline trained models, as well as improved fidelity metrics when compared with high-resolution simulations (Rasp, 2020; Frezat et al., 2022). This increased performance comes from the fact that the online model is optimised along a trajectory of snapshots in time, rather than learned instantaneously as in the offline setting, where no temporal information is included. However, this procedure requires a differentiable numerical scheme to propagate gradients through. For many applications, including climate modelling, direct integration of the machine learning framework with the numerical scheme is not possible. Therefore, there is motivation to explore approaches that include temporal information in the construction of ML parameterisations of turbulent systems, but in an offline setting (i.e. without requiring interaction with the numerical scheme).

We propose a framework to construct an offline trained model that replicates some of the success of online training. Instead of evolving the system in time using a numerical scheme, we instead add an additional network to the loss function to learn the time dynamics of the system (a process often referred to as *emulation*). By including this network in the loss function and evaluating it on snapshots of high-resolution simulations, we can regularise the ML parameterization during training to follow the trajectory of a high-resolution system. In this way, we have included information about the temporal evolution of the system in the optimisation, without requiring interaction with the numerical scheme (Nonnenmacher & Greenberg, 2021). This approach is akin to offline reinforcement learning (Levine et al., 2020), where a policy must be trained from existing datasets, without interacting with the system directly. In such a setting, the learning task includes some learning of the underlying dynamics of the system from the offline dataset. Below we outline the simulation scheme, model architecture, and evaluation metrics.

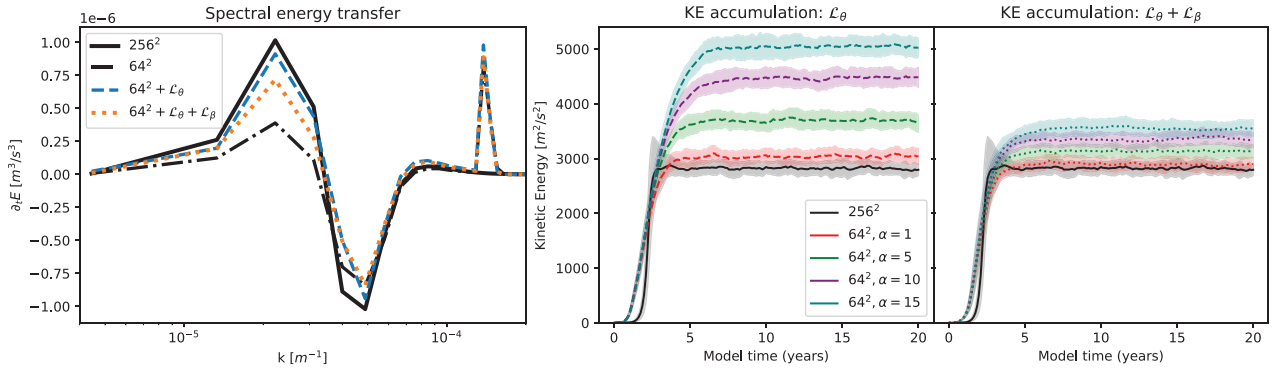

Figure 2. Online tests comparing the performance of parameterizations constructed in a purely offline mode ($\mathcal{L}_\theta$), and when the additional emulator loss is included ($\mathcal{L}_\theta + \mathcal{L}_\beta$). *Left:* The spectral energy transfer from small to large scales that exists in the high-resolution, 256^2 simulation, is significantly weaker in the low resolution, unparameterised system (64^2). Purely offline parameterizations, capture the positive energy transfer ("backscatter") in the coarse resolution simulation (blue dashed line). Including the emulator loss component (orange dotted lines) significantly improves the backscatter with respect to the unparameterised, coarse system, but not as well as a purely offline loss. *Right:* In order to test the stability of the system, we decrease the diffusivity of the system by increasing α, and plot the accumulation of kinetic energy (KE). At $\alpha = 1$, the jointly optimised system better reproduces the KE of the high-resolution system. The jointly optimised model is significantly less sensitive to changing α, implying improved stability when compared with the pure-offline models.

2. Methodology

2.1. Simulated data

We use an idealised model of two-dimensional quasi-geostrophic equations (described in Appendix A), which assumes a stratified fluid in a rotating system. This system of equations allows for the formation of mesoscale eddies, and has been commonly used to study their parameterizations (Frezat et al., 2022; Ross et al., 2023). The prognostic variable of the system is the potential vorticity, q, and we denote variables of a coarse-resolution model by $\overline{q}$. We use a system of two fluid layers, implemented in the publicly available code `pyqg` (Abernathey et al., 2022). The system is advanced using third-order Adams Bashford scheme, and we exploit the fact that the system is in a doubly periodic square domain, by calculating derivatives in Fourier space. This procedure includes a small-scale dissipation filter, which, at each timestep, exponentially damps features below a filter cutoff scale (see appendix A). This has the purpose of both removing aliasing noise, and ensuring stability by attenuating the accumulation of energy on small scales, effectively adding diffusion into the system. In order to test stability, we introduce a scaling parameter α, which modifies the sharpness of the exponential cutoff in such a way that the diffusivity decreases. We estimate the stability of a given model by the accumulation of kinetic energy in the system, as α is increased (leading to lower diffusivity).

A training dataset can be constructed from an ensemble of high-resolution simulations, outputting a total of $N = 18,000$ snapshots. The snapshots are smoothed and downsampled to low-resolution fields, and we build a dataset of tuples: $\{\overline{q}_i, S_i, \overline{q}_{i+\Delta t}\}_{i=1}^N$, where $\overline{q}_i$ is the low-resolution, potential vorticity field, S_i is the subgrid forcing calculated from equation 10, $\overline{q}_{i+\Delta t}$ is the potential vorticity field at some small future time interval, Δ_t.

2.2. Model architecture

Following recent works on ML parameterizations of QG turbulence (Guillaumin & Zanna, 2021; Ross et al., 2023), our central ML architecture is a fully-convolutional neural network (FCNN), where we use 5 convolutional layers. We introduce two networks, which contribute to a combined loss function:

$$\mathcal{L}_{\theta,\beta} = \mathcal{L}_\theta + C_\beta \mathcal{L}_\beta;, \tag{1}$$

where

$$\mathcal{L}_\theta = \frac{1}{n}\sum_i \parallel \Phi_\theta(\bar{q}_i) - S_i \parallel^2 \tag{2}$$

is the standard offline loss term, and

$$\mathcal{L}_\beta = \frac{1}{n}\sum_i \parallel \Phi_\beta(\bar{q}_i, \Phi_\theta(\bar{q}_i)) - (\bar{q}_{i+\Delta t} - \bar{q}_i) \parallel^2 \tag{3}$$

represents the emulator loss. Φ_θ, a FCNN with parameters θ, predicts the subgrid forcing field $\hat{S}_i$ for a given coarse-resolution field $\overline{q}_i$, and n represents the number of samples in each minibatch. We introduce an additional FCNN, Φ_β, which emulates the state of the system at some future

timestep $i + \Delta t$, as a function of the both the resolved dynamics $\bar{q}_i$, and subgrid forcing S_i. The purpose of this network is to project the parameterization network, Φ_θ, forward in time, and regularize it in order to have the desired temporal dynamics. This is in contrast to pure offline learning, where the ML task is purely to predict S_i with as high accuracy as possible. Since the dynamics are not closed at the level of the coarse scales, there is an inherent uncertainty in the estimation of the subgrid forcing, irrespective of the capacity of the ML model; in that sense, the role of the emulator is to identify how estimation errors are propagated through time, and use them to steer the system into a stable regime.

We note here that two important design choices had a significant impact on the performance of the parameterization. We found the best performance when jointly optimising the loss function 1, but after pre-training the emulator network 3. Secondly, we emulate the residuals of the fields between two timesteps, $(\bar{q}_{i+\Delta t} - \bar{q}_i)$ rather than just $\bar{q}_{i+\Delta t}$. Given that for the small values of Δt we are interested in $(\mathcal{O}(10))$, the fields at time i and $i + \Delta t$ are extremely similar, and so the majority of the signal in this mapping is simply the identity. We compared using a ResNet architecture to learn this mapping, vs using a FCNN to emulate the residual quantity, and found significantly improved performance using the latter approach. All fields are standardised to have zero mean and unit variance before being passed to the neural networks. When emulating a residual quantity, we found similar performance when re-standardising the residual fields to have unit variance, versus just taking the residuals of the standardised fields. Therefore, for simplicity, we do not re-standardise the residual fields when producing the results in section 3. We include a scaling coefficient, C_β, which can be chosen to either ensure the two loss terms are balanced, or put emphasis on either the offline or emulator loss component during training. The Φ_β network predicts the state of the system at some future timestep, $\hat{\overline{q}}_{i+\Delta t}$, where training data is obtained from high-resolution simulations. Therefore this joint optimisation procedure incentivises a subgrid forcing model, $\Phi_\theta(\overline{q})$, which follows the trajectory of a high-resolution simulation.

3. Results

The key feature of 2D, quasi-geostrophic turbulence that we aim to capture with parameterizations is the inverse energy cascade (Kraichnan & Montgomery, 1980). In the left panel of figure 2, we show the spectral energy transfer for four different simulations. The solid black lines show results for a high-resolution simulation with 256^2 grid cells. The positive energy transfer around wavenumber $\kappa = 2 \times 10^{-5}\mathrm{m}^{-1}$, and the negative values around $\kappa = 5 \times 10^{-5}\mathrm{m}^{-1}$ indicate that energy is being removed at small scales, and increased at larger scales following the inverse energy cascade. We refer to the positive energy transfer as a "backscatter". The low-resolution (64^2), unparameterised system in black dash-dot lines, has severely diminished backscatter due to the coarser resolution. This effect is a primary target to be reproduced by a parameterization. In blue dashed lines, we show the spectral energy transfer for a simulation run including a parameterisation trained with a pure-offline loss, i.e. using equation 2 only. As with previous works (Ross et al., 2023), this baseline model captures the backscatter well.

In orange dashed lines, we show results for a model constructed using the joint loss function, equation 1. After experimenting with a range of different time horizons and loss coefficients, we found the best performance in terms of the online metrics shown in figure 2 were achieved using $\Delta t = 10$ and $\mathcal{C}_\beta = 40$. We note that on average, the values of these two loss functions are different - the Φ_β network is predicting the residual quantity, and as described in the previous section, we do not standarise the residual fields to have unit variance. In addition, the two networks perform differently at their respective learning tasks. We observe that, on average the loss terms $\mathcal{L}_\beta \sim 10^{-3}$, and $\mathcal{L}_\theta \sim 10^{-1}$. Given this, the $\mathcal{L}_\beta$ term, with a coefficient of $C_\beta = 40$, will be approximately an order of magnitude smaller than the $\mathcal{L}_\theta$ term, making the emulator component of the network subdominant, but not irrelevant.

In the right panels, we show the accumulation of kinetic energy over time, as we decrease the diffusivity of the system by increasing α. We consistently observed (figure not shown) that this accumulation of kinetic energy occurred around the gridscale, indicating that the parameterization accumulates instabilities. In the rightmost panel, we show results for the jointly optimised model. Firstly, the KE at $\alpha = 1$ is more consistent with the high-resolution case. Additionally, as α increases, we see significantly less energy accumulation than with the purely offline model. This indicates that the system with the jointly optimised parameterisation is more stable. We noticed a general trend that the larger the coefficient $\mathcal{C}_\beta$ on the emulator component of equation 1, the more significant the gains in stability as indicated by the right panel of figure 2. However this came at the expense of diminishing the effect of the backscatter. This implies that whilst including temporal information does indeed improve stability, the offline loss term is necessary to capture the desired energy transfer properties. Given this trade-off, we additionally experiment with simply diminishing the contribution of a purely offline trained model. We find that similar results, of diminished backscatter but improved stability, can be obtained by reducing the contribution of a purely-offline model by 20% when tested in online simulations. However this approach would require *a posteriori* tuning of the parameterization, some-

thing that our framework does not require.

4. Summary

Methods to improve the accuracy of low resolution numerical simulations of turbulent fluids are essential in a wide range of fields, particularly in the case of climate modelling, where important dynamics are only partially resolved. Approaches that require differentiable numerical simulations to include information about the time-evolution of the system (*online* training) are highly effective, but particularly in the case of climate modelling, these methods are incompatible with many key simulators. We have presented a novel ML framework for the parameterization of important subgrid dynamics in ocean simulations, which includes temporal information without requiring differentiable simulations. We demonstrate that this approach leads to improved stability when the ML architecture is included into a hybrid ML+physics simulation, and identify a trade-off between stability properties and capturing the desired energy transfers. There is significant ongoing work to be explored, to clarify questions such as whether pre-training and fixing the emulator weights, or jointly optimising the networks produces optimum performance, and understanding the balance between the time-emulation and offline components of the loss function. Additionally, the emulator network could be used over multiple timesteps, analogous to the methodologies in (Rasp, 2020) and (Frezat et al., 2022).

Broader impact

The primary motivation of this work is to improve representations of subgrid dynamics in the modelling of ocean turbulence. This is a crucial component in modelling of the climate system, and therefore of being able to produce reliable climate projections. The representation of subgrid effects has been identified as one of, if not the major source of discrepancy between different climate projections. Reliable climate modelling is of fundamental importance at the economic, societal and governmental level, as projections inform decisions made in order to prepare for changes to Earth's climate, and so advancements in this area will have significant broader impact.

References

Abernathey, R., Rochanotes, Ross, A., Jansen, M., Ziwei Li, Poulin, F. J., Constantinou, N. C., Anirban Sinha, Dhruv Balwada, SalahKouhen, Jones, S., Rocha, C. B., Wolfe, C. L. P., Chuizheng Meng, Van Kemenade, H., Bourbeau, J., Penn, J., Busecke, J., Bueti, M., and , Tobias. pyqg/pyqg: v0.7.2, 2022. URL https://zenodo.org/record/6563667.

Bar-Sinai, Y., Hoyer, S., Hickey, J., and Brenner, M. P. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, July 2019. doi: 10.1073/pnas.1814058116. URL https://doi.org/10.1073/pnas.1814058116.

Beck, A., Flad, D., and Munz, C.-D. Deep neural networks for data-driven LES closure models. *Journal of Computational Physics*, 398:108910, December 2019. doi: 10.1016/j.jcp.2019.108910. URL https://doi.org/10.1016/j.jcp.2019.108910.

Bolton, T. and Zanna, L. Applications of deep learning to ocean data inference and subgrid parameterization. *Journal of Advances in Modeling Earth Systems*, 11(1):376–399, January 2019. doi: 10.1029/2018ms001472. URL https://doi.org/10.1029/2018ms001472.

Ferrari, R. and Wunsch, C. Ocean circulation kinetic energy: Reservoirs, sources, and sinks. *Annual Review of Fluid Mechanics*, 41(1):253–282, 2009. doi: 10.1146/annurev.fluid.40.111406.102139. URL https://doi.org/10.1146/annurev.fluid.40.111406.1

Fox-Kemper, B. and Menemenlis, D. Can large eddy simulation techniques improve mesoscale rich ocean models? In *Ocean Modeling in an Eddying Regime*, pp. 319–337. American Geophysical Union, 2008. doi: 10.1029/177gm19. URL https://doi.org/10.1029/177gm19.

Frezat, H., Sommer, J. L., Fablet, R., Balarac, G., and Lguensat, R. A posteriori learning for quasi-geostrophic turbulence parametrization. *Journal of Advances in Modeling Earth Systems*, 14(11), November 2022. doi: 10.1029/2022ms003124. URL https://doi.org/10.1029/2022ms003124.

Guan, Y., Subel, A., Chattopadhyay, A., and Hassanzadeh, P. Learning physics-constrained subgrid-scale closures in the small-data regime for stable and accurate LES. *Physica D Nonlinear Phenomena*, 443:133568, January 2023. doi: 10.1016/j.physd.2022.133568.

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

Haarsma, R. J., Roberts, M. J., Vidale, P. L., Senior, C. A., Bellucci, A., Bao, Q., Chang, P., Corti, S., Fučkar, N. S., Guemas, V., von Hardenberg, J., Hazeleger, W., Kodama, C., Koenigk, T., Leung, L. R., Lu, J., Luo, J.-J., Mao, J., Mizielinski, M. S., Mizuta, R., Nobre, P., Satoh, M., Scoccimarro, E., Semmler, T.,

Small, J., and von Storch, J.-S. High resolution model intercomparison project (HighResMIP v1.0) for CMIP6. *Geoscientific Model Development*, 9(11):4185–4208, November 2016. doi: 10.5194/gmd-9-4185-2016. URL https://doi.org/10.5194/gmd-9-4185-2016.

Han, J., Jentzen, A., and E, W. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Science*, 115 (34):8505–8510, August 2018. doi: 10.1073/pnas.1718942115.

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

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

Kraichnan, R. H. and Montgomery, D. Two-dimensional turbulence. *Reports on Progress in Physics*, 43(5):547–619, May 1980. doi: 10.1088/0034-4885/43/5/001. URL https://doi.org/10.1088/0034-4885/43/5/001.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi: 10.1109/5.726791. URL https://doi.org/10.1109/5.726791.

Leith, C. Stochastic models of chaotic systems. *Physica D: Nonlinear Phenomena*, 98(2-4):481–491, November 1996. doi: 10.1016/0167-2789(96)00107-8. URL https://doi.org/10.1016/0167-2789(96)00107-8.

Lesieur, M., Métais, O., and Comte, P. *Large-Eddy Simulations of Turbulence*. Cambridge University Press, 2005. doi: 10.1017/CBO9780511755507.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. *arXiv e-prints*, art. arXiv:2005.01643, May 2020. doi: 10.48550/arXiv.2005.01643.

Maulik, R., San, O., Rasheed, A., and Vedula, P. Subgrid modelling for two-dimensional turbulence using neural networks. *Journal of Fluid Mechanics*, 858:122–144, November 2018. doi: 10.1017/jfm.2018.770. URL https://doi.org/10.1017/jfm.2018.770.

Nonnenmacher, M. and Greenberg, D. S. Deep emulators for differentiation, forecasting, and parametrization in earth science simulators. *Journal of Advances in Modeling Earth Systems*, 13(7), June 2021. doi: 10.1029/2021ms002554. URL https://doi.org/10.1029/2021ms002554.

Novati, G., de Laroussilhe, H. L., and Koumoutsakos, P. Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence*, 3(1):87–96, January 2021. doi: 10.1038/s42256-020-00272-0. URL https://doi.org/10.1038/s42256-020-00272-0.

Perezhogin, P., Zanna, L., and Fernandez-Granda, C. Generative data-driven approaches for stochastic subgrid parameterizations in an idealized ocean model. 2023. doi: 10.48550/ARXIV.2302.07984. URL https://arxiv.org/abs/2302.07984.

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

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

Sanchez-Gonzalez, A., Godwin, J., Pfaff, T., Ying, R., Leskovec, J., and Battaglia, P. W. Learning to simulate complex physics with graph networks. 2020. doi: 10.48550/ARXIV.2002.09405. URL https://arxiv.org/abs/2002.09405.

Sirignano, J. and Spiliopoulos, K. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, December 2018. doi: 10.1016/j.jcp.2018.08.029.

Sirignano, J., MacArt, J. F., and Freund, J. B. DPM: A deep learning PDE augmentation method with application to large-eddy simulation. *Journal of Computational Physics*, 423:109811, December 2020. doi: 10.1016/j.jcp.2020.109811. URL https://doi.org/10.1016/j.jcp.2020.109811.

Smagorinsky, J. General circulation experiments with the primitive equations. *Monthly Weather Review*, 91(3):99–164, March 1963. doi: 10.1175/1520-0493(1963)091⟨0099:gcewtp⟩2.3.co;2. URL https://doi.org/10.1175/1520-0493(1963)091<0099:g

Stachenfeld, K., Fielding, D. B., Kochkov, D., Cranmer, M., Pfaff, T., Godwin, J., Cui, C., Ho, S., Battaglia, P., and Sanchez-Gonzalez, A. Learned Coarse Models for Efficient Turbulence Simulation. *arXiv e-prints*, art. arXiv:2112.15275, December 2021. doi: 10.48550/arXiv.2112.15275.

Zhiyin, Y. Large-eddy simulation: Past, present and the future. *Chinese Journal of Aeronautics*, 28(1):11–24, 2015. ISSN 1000-9361. doi: https://doi.org/10.1016/j.cja.2014.12.007. URL https://www.sciencedirect.com/science/article/pii/S1000936114002064.

A. Quasi-geostrophic equations solved

We consider a two-layer, quasi-geostrophic system, with the prognostic variable is the potential vorticity, given by

$$q_m = \nabla^2\psi_m + (-1)^m \frac{f_0^2}{g' H_m}(\psi_1 - \psi_2), m \in \{1, 2\}, \tag{4}$$

where $m = 1$ denotes the upper layer, $m = 2$ denotes the lower layer, H_m is the depth of the layer, ψ is the streamfunction, which is related to the fluid velocity by $\mathbf{u}_m = (u_m, v_m) = (-\partial_y\psi_m, \partial_x\psi_m)$, and f_0 is the Coriolis frequency. The time evolution of the system is given by

$$\partial_t q_m + \nabla \cdot (\mathbf{u}_m q_m) + \beta_m \partial_x \psi_m + U_m \partial_x q_m = -\delta_{m,2} r_{ek} \nabla^2 \psi_m + \mathrm{ssd} \circ q_m, \tag{5}$$

where U_m is the mean flow in the x (zonal) direction, $\beta_m = \beta + (-1)^{m+1}\frac{f_0^2}{g' H_m}(U_1 - U_2)$, r_{ek} is the bottom drag coefficient, and $\delta_{m,2}$ is the Kronecker delta. These equations are solved numerically in spectral space, using 3rd order Adams-Bashford integration. The final ssd term refers to small-scale dissipation:

$$\mathrm{ssd} = \begin{cases} \mathrm{e}^{-\alpha 23.6\,(\kappa^\star - \kappa_c)^4} : & \kappa^\star \geq \kappa_c \\ 1 : & \text{otherwise}\,. \end{cases} \tag{6}$$

where the cutoff is given by $\kappa_c = (0.65\pi)/\Delta x$, and $\kappa^\star$ is a radial wavenumber:

$$\kappa^\star \equiv \sqrt{(k\,\Delta x)^2 + (l\,\Delta x)^2}\,, \tag{7}$$

with k and l being wavenumbers in the horizontal and vertical directions, and Δx is the grid size. This dissipation filter is included for the purposes of dealising, and to ensure stability of the system. We include a parameter α, by default $\alpha = 1$, in order to reduce the diffusivity of the system, and enable tests of stability presented in figure 2.

We denote a smoothed field as having been convolved with some filter kernel, $G(\mathbf{y})$:

$$\overline{\phi}(\mathbf{x}) = \int G(\mathbf{y} - \mathbf{x})\phi(\mathbf{y}) d\mathbf{y}, \tag{8}$$

where we use the same filter kernel as in equation 6, with $\alpha = 1$. Smoothing equation 5, we get an equation for the low-resolution system:

$$\partial_t \overline{q}_m + \nabla \cdot (\overline{\mathbf{u}}_m \overline{q}_m) + \beta_m \partial_x \overline{\psi}_m + U_m \partial_x \overline{q}_m = -\delta_{m,2} r_{ek} \nabla^2 \overline{\psi}_m + S + \mathrm{ssd} \circ \bar{q}_m, \tag{9}$$

where now we have an additional forcing term S, which accounts for dynamics occurring below the smoothing scale:

$$S = \nabla \cdot (\overline{\mathbf{u}}\,\overline{q} - \overline{\mathbf{u}q})\,. \tag{10}$$