

A Data-Driven Investigation into the Behaviour and Parameterisation of Mesoscale Eddies



Thomas Bolton
Oriel College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Michaelmas 2019

To Caz.
For her unwavering support.

Life is like riding a bicycle. To keep
your balance you must keep moving.

Albert Einstein

Anyone who values truth should stop
worshipping reason.

Jonathan Haidt

Acknowledgements

Throughout my undergraduate and postgraduate studies at Oxford, I sometimes struggled to see myself ever getting to the end. However, thanks to numerous people along the way, it seems the end has arrived.

My first and largest thanks go to my supervisor Laure Zanna. We started working with each other during my MPhys final year project, where she initially took a chance with me. In the following four years of my DPhil, Laure's guidance improved my ability to conduct scientific research immeasurably - those skills will prove invaluable in my career. In addition to receiving world-class supervision, I am also extremely grateful for her support when life gave me a few lemons.

I would like to thank Andrew Wells, David Marshall, and Tim Palmer, for their discussions and advice. In particular, Andrew Wells, who acted as my internal examiner, as well as examining my transfer and confirmation - his feedback has been much appreciated. I would also like to thank Hannah Christensen for her guidance, who, in a summer research internship during my undergraduate studies, sparked my interest in joining AOPP. This thesis would not have been possible without the computing help from Man-Suen Chan and the administrative help from Lucy Li.

The Ocean-Climate group has been more than welcoming during my time in AOPP. I am grateful to Robert Fraser, for the numerous coffee breaks and lending an ear to my moans. To Tomos David, for the entertainment and *occasional* pub outing. To Jamie Parkinson, Chris MacMackin, and Chris O'Reilly, for adding to my AOPP experience. And to Matthias Aengenheyster, for putting up with me as an office mate.

I would like to thank my friends and family for their support. Friends for the fun times, and family for keeping me grounded during the hard times. And finally, to Caz, for sharing this long winding path with me.

Abstract

Mesoscale eddies, turbulent oceanic features of length-scales 10-100km, are an important component of the climate system. They impact the large-scale circulation of both the ocean and atmosphere, as well as the transport of tracers such as heat and carbon. Due to the computational costs of running ocean models at high-resolution, climate models generally do not fully resolve mesoscale eddies, and will not be able to for the foreseeable future. We, therefore, need to parameterise the effects of mesoscale eddies on the large-scale flow. Eddy parameterisations are typically derived from physical theories and mechanisms; this approach can sometimes over-simplify the underlying dynamical process. The primary aim of this thesis is to uncover new ways of capturing the complex non-linear behaviour of mesoscale eddies.

Using a hierarchy of numerical models, we take a data-driven approach to studying and parameterising mesoscale eddies. Using an observation-driven experiment, we investigate how the spatio-temporal variability of real mesoscale eddies affect tracer fluxes in the North Atlantic. In an idealised model, we demonstrate that modern machine learning algorithms, namely convolutional neural networks, can accurately represent eddy momentum fluxes. We also demonstrate that machine learning can construct expressions for eddy momentum and temperature fluxes while retaining interpretability. We find that, when implemented, data-driven parameterisations can emulate the scale-interaction between mesoscale eddies and the resolved flow, increasing the kinetic energy and improving the higher-order statistics of the velocity field.

Our results show that data-driven algorithms, while respecting physical principles, can be used to parameterise the effects of mesoscale eddies, with implementation in more realistic models now a possibility. The predictive skill of data-driven algorithms is not the only advantage: our use of interpretable machine learning opens up the door for more knowledge discovery studies about eddies and turbulence in general.

Contents

1	Introduction	1
1.1	The role of mesoscale eddies in the ocean and atmosphere	4
1.2	Thesis aims and structure	9
2	Background	11
2.1	Equations of motion of a geophysical fluid	11
2.2	Parameterisations of turbulent ocean eddies	13
2.2.1	Physics-driven parameterisations	17
2.2.2	Data-driven parameterisations	22
3	Methods	26
3.1	Numerical ocean models	26
3.1.1	Shallow water model	26
3.1.2	Quasi-geostrophic model	30
3.1.3	Primitive equation model	32
3.2	Defining the sub-grid effects of eddies	38
3.3	Supervised machine learning for regression	44
3.3.1	Data-driven prediction: convolutional neural networks	50
3.3.2	Data-driven inference: closed-form equation discovery	53
4	Observed eddy tracer fluxes in the North Atlantic	61
4.1	Introduction	61
4.2	Data and methods	63
4.2.1	AVISO satellite altimetry data	63
4.2.2	Passive tracer model	63
4.2.3	Up-stream and down-stream regions	64

4.3	North Atlantic lateral tracer mixing	65
4.3.1	Tracer statistical moments	65
4.3.2	Effective diffusivity and tracer Flux	67
4.3.3	Mixing length diffusivity expression	72
4.4	Up-stream Gulf Stream jet dynamics	77
4.5	Discussion and summary	79
5	Predicting eddy momentum fluxes with machine learning	83
5.1	Introduction	83
5.2	Data and methods	85
5.2.1	Eddy momentum forcing	85
5.2.2	Predictive algorithm: convolutional neural networks	86
5.3	Neural network generalisation and sensitivity	92
5.3.1	Non-local predictions	92
5.3.2	Generalising to different Reynolds numbers	96
5.3.3	Sensitivity of neural networks to under-sampling	97
5.4	Physically-constrained neural networks	101
5.4.1	Momentum biases	101
5.4.2	Towards momentum-conserving neural networks	102
5.5	Summary and discussion	106
5.5.1	Summary	106
5.5.2	Implications for leveraging observations	108
5.5.3	Implications for parameterisations	109
6	Inferring expressions for eddy fluxes with machine learning	111
6.1	Introduction	111
6.2	Data and methods	113
6.3	Results: eddy momentum flux expressions	117
6.3.1	Barotropic model	117
6.3.2	Baroclinic model	122
6.4	Results: eddy temperature flux expressions	128
6.4.1	Baroclinic model	128
6.5	Results: prognostic eddy energy equation	133

6.6	Physical interpretation of eddy flux expressions	135
6.6.1	Eddy momentum forcing expressions	135
6.6.2	Eddy temperature forcing expression	138
6.7	Summary and discussion	141
6.7.1	Summary	141
6.7.2	Discussion	142
7	Prognostic evaluation of eddy momentum parameterisations	145
7.1	Introduction	145
7.2	Data and methods	147
7.2.1	Candidate momentum parameterisations	147
7.2.2	Ocean model to parameterise	152
7.3	Diagnostic evaluation	153
7.4	Prognostic evaluation	157
7.4.1	Statistical moments of velocity fields	158
7.4.2	Impact of parameterisations on model energetics	164
7.5	Summary and discussion	171
7.5.1	Summary	171
7.5.2	Discussion	174
8	Summary and discussion	178
8.1	Summary	178
8.2	Discussion	182
	Bibliography	187

List of Figures

1.1	From the study of Klocker and Abernathey (2014): Panel (a) shows an observational snapshot of the sea surface height anomaly in a sector of the Pacific (180°-130°W), and Panel (b) shows the corresponding zonally-averaged zonal velocity (black curve), eddy velocity (red curve), and eddy phase-speed (blue curve).	2
1.2	The two-regime description of eddy-mean flow interaction from the idealised study of Waterman and Jayne (2010), with the figure adapted from Waterman and Hoskins (2013). The unstable-jet and wave-maker regimes, of the up- and down-stream regions respectively, are illustrated, highlighting the patterns of the large-scale absolute vorticity gradients. Solid black arrows denote the direction of the divergent time-mean eddy vorticity fluxes and their patterns of convergence/divergence. Regarding the solid-filled contours, light gray shading indicates positive values (an eddy vorticity flux convergence) and dark gray shading indicates negative values (an eddy vorticity flux divergence).	6
1.3	A schematic of the feedback between mesoscale eddies and the atmosphere. The interaction regulates the strength of the Kuroshio through exchanges of energy between various reservoirs.	8
2.1	An illustration of the spectrum of approaches to constructing parameterisations, from the traditional physics-driven approach on the left-hand side, moving to purely data-driven approaches at the far right-hand side.	16

3.1	Snapshots and time-means of the three prognostic variables of a high-resolution 3.75km shallow water model. Panels (a), (b), and (c) show snapshots of u , v , and h respectively, while Panels (d), (e), and (f) show the time-means over a 10 year period.	27
3.2	Panels (a), (b) and (c) show snapshots of the three prognostic variables u , v , and η respectively, of the high-resolution 3.75km barotropic primitive equation model in MITgcm. Panels (d), (e) and (f) show the time-mean of the three prognostic variables u , v , and η respectively .	34
3.3	Snapshots and time-means of the depth-averaged prognostic variables in the 7.5km resolution baroclinic primitive equation model with 15 vertical layers. Panels (a), (b), and (c) show snapshots of the depth-averaged u , v , and T respectively. Panels (d), (e), and (f) show the time-means of the depth-averaged u , v , and T . The visible numerical grid-scale noise, primarily at the northern boundary, could be removed without significantly increasing the dissipation and over-dampening of the circulation. The time step was also reduced to try and remove the numerical noise to no avail.	36
3.4	Snapshots and time-means of the zonally-averaged prognostic variables in the baroclinic primitive equation model. Panels (a), (b), and (c) show snapshots of the zonally-averaged u , v , and T respectively. Panels (d), (e), and (f) show the time-means of the zonally-averaged u , v , and T	37
3.5	An illustration of a 4×4 grid box spatial coarse-graining procedure of a quantity c to $\bar{c}$	39
3.6	A synthetically generated streamfunction, Panel (a), based on a elliptical Gaussian function. The resulting velocities $u = -\frac{\partial \psi}{\partial y}$ and $v = \frac{\partial \psi}{\partial x}$ are shown in Panel (b) and Panel (c) respectively.	42
3.7	Illustrations of three averaging procedures: Panel (a) shows coarse-graining, Panel (b) for a spatial filter, and Panel (c) shows spatial filtering and then coarse-graining. The top row shows the averaged zonal velocity $\bar{u}$ and the bottom row shows the zonal component of the resulting eddy momentum forcing S_x	43

3.8	An illustration of a simple feed forward neural network with four input variables x_i , a single hidden layer of five nodes z_j , leading to a single output node for the predicted target variable $\hat{y}$	47
3.9	An illustration of local versus global minima in non-convex optimisation. Gradient descent aims to iteratively find the minimum of a function by considering the gradient of that function with respect to the parameters.	48
3.10	An illustration of a convolution operation of a 7×7 matrix q with a 3×3 filter f , showing the final 5×5 result $q \otimes f$	51
3.11	Solutions to the one dimensional Burger's equation, which comprises an advection and diffusion term, at various time-steps with a sine function initialisation.	57
3.12	The iterative relevance vector machine algorithm applied to data from a one-dimensional Burger's equation. Nine basis functions comprise the library, of which three are selected by the algorithm after three iterations. At each iteration, basis functions with a regression coefficient with uncertainty α_i higher than the threshold δ are pruned from the library. Note that the uncertainties α_i are re-calculated from the data after each iteration.	59
4.1	The first four statistical moments of SST and the passive tracer q by sampling values at each point in space. The rows from top to bottom represent the four statistical moments: mean, standard deviation, skewness and kurtosis. The first column is for 22 years of passive tracer data, and the second column is for SST observations over the same period. SST data is from the ARMOR3D dataset (Guinehut et al., 2012). The mean and standard deviation have units of tracer concentration (or degrees Celsius for SST), while the skewness and kurtosis are dimensionless. The up- and down- stream regions (80-50W and 50-10W respectively), are illustrated in Panels (a) and (b).	66

4.2	The effective diffusivity, and the corresponding tracer flux, calculated for two different regions of the North Atlantic: up-stream, defined as 80-50W, and down-stream, defined as 50-10W. The boundary between up- and down-stream is shown by the black-dotted line in Panel (a). Panels (b) and (c) show the effective diffusivity for the up- and down-stream regions respectively, while Panels (d) and (e) show the corresponding tracer flux. The shading in Panels (b)-(e) represents the interannual variability, defined as the standard deviation of the annual means. The blacked-dotted lines in Panels (b)-(e) illustrate the calculation over the full region.	68
4.3	Panels (a) and (b) show the time evolution of the effective diffusivity as a function of tracer value (y-axis), for the up- and down-stream regions respectively. Panels (c) and (d) show the individual time series at the tracer value $q = 0.15$, which is approximately co-located with the 25 cm SSH contour in the jet core. The black-dotted lines in Panels (c) and (d) are linear fits in time.	71
4.4	The various components of the Ferrari and Nikurashin (2010) diffusivity expression, calculated from satellite altimetry data. Panel (a) shows the time-mean zonal velocity; Panel (b) shows the root-mean square (RMS) eddy velocity, where eddy is defined as the deviation from the time-mean; Panel (c) shows the decorrelation time-scale in days, using the formula $\tau = 2\Gamma L_e / u_{rms}$; Panel (d) is the eddy phase speed data, provided by C. Hughes (personal communication). Panel (e) shows the final product: the time-mean diffusivity expression K_{FN10}	72
4.5	Panel (a) displays the time-mean SSH field from satellite altimetry data, illustrating where each SSH value is approximately located in the North Atlantic. Panels (b) and (c) show the K_{FN10} diffusivity expression calculated along SSH contours. The shading represents the interannual variability, defined as the standard deviation of the annual means. The black-dotted lines show the FN10 diffusivity calculated over the full region.	74

4.6	Panels (a) and (b) show linear perturbations to the time-mean K_{FN10} diffusivity expression for both the up- and down-stream regions. Panels (c) and (d) compare time series of the K_{FN10} diffusivity with the effective diffusivity in the jet core (at the 25 cm SSH contour for K_{FN10} and $q = 0.15$ for K_{eff}).	76
4.7	Comparisons of the up-stream effective diffusivity and tracer flux with a measure of mean-flow: along-jet speed. Panel (a) compares the along-jet speed with the effective diffusivity, while Panel (b) compares the along-jet speed with the tracer flux - both at the tracer value $q=0.15$. Note that a more negative tracer flux indicates an increased across-jet exchange of the passive tracer.	78
5.1	Panel (a) illustrates the upper-layer filtered-streamfunction $\bar{\psi}$ of the QG model, including the three regions in which we train the neural networks: region 1 (white-dashed) is on the western boundary, region 2 (black-solid) is on the eastern boundary, and region 3 (grey-dash-dotted) is centred on the southern gyre. Panel (b) shows a close-up of the filtered-streamfunction $\bar{\psi}$ within training region 1 while Panel (c) illustrates how training region 1 is split into 16 40×40 grid point sub-regions - the size of the input and output arrays of the neural network is 40×40 grid points. The input variable of each neural network is the filtered streamfunction $\bar{\psi}$, and the output variable is either the zonal component $\tilde{S}_x$ or meridional component $\tilde{S}_y$ of the sub-filter eddy momentum forcing. The architecture of the convolutional neural network, with an example input $\bar{\psi}$ and output $\tilde{S}_x$, is illustrated underneath Panels (a), (b), and (c).	88

5.2	Examining the non-local prediction ability. Comparisons of the true zonal component of the sub-filter momentum forcing S_x , with the neural networks trained using data from three different regions. The first three rows compare snapshots, time-means, and the standard deviation respectively, while the bottom row shows the correlation between the true S_x and the predictions $\tilde{S}_x$. The first column contains the diagnostics using the true zonal sub-filter momentum forcing S_x , while columns two, three, and four use predictions $\tilde{S}_x$ from the neural networks $f_x(\bar{\psi}, \mathbf{w}_1)$, $f_x(\bar{\psi}, \mathbf{w}_2)$, and $f_x(\bar{\psi}, \mathbf{w}_3)$ respectively. All diagnostics were produced using the validation data.	93
5.3	The same diagnostics as Figure 5.2, but for the meridional component of the sub-filter momentum forcing: the true S_y and the predictions $\tilde{S}_y$ from the neural networks $f_y(\bar{\psi}, \mathbf{w}_1)$, $f_y(\bar{\psi}, \mathbf{w}_2)$, and $f_y(\bar{\psi}, \mathbf{w}_3)$	95
5.4	Examining the ability to generalise to new regimes: using the trained neural network $f_x(\bar{\psi}, \mathbf{w}_1)$, we make predictions for model runs of different viscosities and wind forcings. From each model run, we use one year of the upper-layer filtered streamfunction to generate predictions $\tilde{S}_x$ from $f_x(\bar{\psi}, \mathbf{w}_1)$ to see how they compare to the true S_x . We study a run of higher viscosity $\nu = 200 \text{ m}^2\text{s}^{-2}$, and runs with wind stress amplitude $\tau_0 = 0.3, 0.6, 0.8$, and 0.9 Nm^{-2} . Note that $f_x(\bar{\psi}, \mathbf{w}_1)$ was trained on a run with $\nu = 75 \text{ m}^2\text{s}^{-2}$ and $\tau_0 = 0.8 \text{ Nm}^{-2}$, the standard deviation and correlation maps of which are included again here in Panels (j), (k), and (l).	98
5.5	Determining how under-sampling of the training data impacts neural network error. Panel (a) shows the RMSE of the neural network $f_x(\bar{\psi}, \mathbf{w}_1)$ trained with dense (un-altered) training data, while Panel (b) shows the RMSE of the neural network trained with sub-sampled (18.75%) data. Panel (c) shows the RMSE as a function of the percentage of spatial points sampled at each time-slice of the training data. Note that the RMSE is calculated over the full-domain during the validation period (the final year of data).	99

5.6	Panels (a) and (b) show time series of the zonal and meridional components of the sub-filter momentum forcing respectively, at a single point near the middle of the domain. Panels (c) and (d) also show time series of the zonal and meridional components of the sub-filter momentum forcing, but this time spatially-averaged over the entire domain. . . .	103
5.7	The standard deviation and spatial-average time series of the predictions $\tilde{S}_x$ and $\tilde{S}_y$ of the momentum conserving approaches A, B, and C. Panels (a), (b), and (c) show the standard deviation of $\tilde{S}_x$ from $f_x(\bar{\psi}, \mathbf{w}_1^A)$, $f_x(\bar{\psi}, \mathbf{w}_1^B)$, and $f_x(\bar{\psi}, \mathbf{w}_1^C)$ respectively, while Panels (e), (f), and (g) show the standard deviation of $\tilde{S}_y$ from $f_y(\bar{\psi}, \mathbf{w}_1^A)$, $f_y(\bar{\psi}, \mathbf{w}_1^B)$, and $f_y(\bar{\psi}, \mathbf{w}_1^C)$ respectively. The spatial-averages of these predictions $\tilde{S}_x$ and $\tilde{S}_y$ are shown in Panels (d) and (h).	105
6.1	An illustration from start-to-finish of the method of discovering equations from data. The top-row provides the general concepts of the procedure: (1) a spatio-temporal dataset is used; (2) a library of basis functions is constructed from the dataset; (3) an iterative sparse regression prunes the library of basis functions; (4) when the iterative regression has converged, the remaining basis functions form the final equation. The bottom row shows specifically how we have adapted this technique for discovering eddy momentum parameterisations from high-resolution ocean models.	115
6.2	The impact of the pruning threshold on the results of the RVM algorithm applied to data from the barotropic model. Panel (a) and (b) show how the R^2 coefficient of determination (red) and number of selected basis functions (blue) changes with the pruning threshold, when applied to reveal expressions for the eddy momentum forcings S_x and S_y respectively.	119
6.3	Within the validation data of the barotropic model, comparisons of the true eddy momentum forcing S_x and S_y with the predicted values $\hat{S}_x$ and $\hat{S}_y$ from the expression revealed by the RVM algorithm, defined by Equation 6.4.	121

6.4	The impact of the pruning threshold on the results of the RVM algorithm applied to data from the baroclinic model. Panel (a) and (b) show how the R^2 coefficient of determination (red) and number of selected basis functions (blue) changes with the pruning threshold, when applied to reveal expressions for the eddy momentum forcings S_x and S_y respectively.	124
6.5	Within the validation data of the baroclinic model, a depth-averaged comparison of the true eddy momentum forcing S_x and S_y with the predicted values $\hat{S}_x$ and $\hat{S}_y$ from the expression revealed by the RVM algorithm, defined by Equation 6.7. The spatial maps here are a function of longitude (x-axis) and latitude (y-axis).	126
6.6	Within the validation data of the baroclinic model, a zonally-averaged comparison of the true eddy momentum forcing S_x and S_y with the predicted values $\hat{S}_x$ and $\hat{S}_y$ from the expression revealed by the RVM algorithm, defined by Equation 6.7. The spatial maps here are a function of latitude (x-axis) and depth (y-axis).	127
6.7	The impact of the pruning threshold on the results of the RVM-algorithm applied to the eddy temperature forcing using data from the baroclinic model. The R^2 coefficient of determination (red) and number of selected basis functions (blue) is plotted against the pruning threshold.	129
6.8	Within the validation data of the baroclinic model, a depth-averaged comparison of the true eddy temperature forcing S_T with the predicted values $\hat{S}_T$ from the expression revealed by the RVM algorithm, defined by Equation 6.10. The spatial maps here are a function of longitude (x-axis) and latitude (y-axis).	131
6.9	Within the validation data of the baroclinic model, a zonally-averaged comparison of the true eddy temperature forcing S_T with the predicted values $\hat{S}_T$ from the expression revealed by the RVM algorithm, defined by Equation 6.10. The spatial maps here are a function of latitude (x-axis) and depth (y-axis).	132

7.1	A schematic of the architecture of the fully-convolutional neural network, which takes the velocity fields u and v as multi-channel inputs, and simultaneously predicts the two components of the eddy momentum forcing S_x and S_y . The architecture is physically-constrained such that the final result is always the divergence of a symmetric eddy stress tensor.	149
7.2	Within the validation data of the barotropic MITgcm model, comparisons of the diagnostic predictions of three momentum parameterisations: the deformation parameterisation of Anstey and Zanna (2017), the barotropic RVM-algorithm expression, and the fully-convolutional neural network described in Section 7.2.1. For each parameterisation, we compare the mean, standard deviation, skewness, and kurtosis, of the zonal eddy momentum forcing $\hat{S}_x$ with the true forcing S_x	154
7.3	The same as in Figure 7.2, but comparing the meridional component of the eddy momentum forcing $\hat{S}_y$ with the true S_y	156
7.4	The mean, standard deviation, skewness and kurtosis of the zonal velocity field u for five different shallow water models. Three of the models contain either the AZ17, RVM or FCNN parameterisation. The two remaining models are run at 30km and 3.75km resolution with no parameterisation, representing the coarse- and high-resolution cases for comparison. The spatial maps are a function of longitude (x-axis) and latitude (y-axis).	160
7.5	The mean, standard deviation, skewness, and kurtosis, the same as Figure 7.4, but for the meridional component v of the velocity field. The spatial maps are a function of longitude (x-axis) and latitude (y-axis).	161
7.6	The probability distribution functions (PDFs) of the zonal velocity field, for three different regions of the model - these regions are illustrated in Panel (a). The PDFs for regions 1, 2, and 3 are shown in Panels (b), (c), and (d) respectively. Each panel includes PDFs of the high-resolution, the low-resolution, and the three parameterised models.	163

7.7	Time-series of the globally-integrated kinetic energy of four shallow water models during a 10 year spin-up. One model has no momentum parameterisation, while the other models have one of the following: deformation-based parameterisation of Anstey and Zanna (2017), the expression revealed by the relevance vector machine (RVM) algorithm, and a fully-convolutional neural network.	165
7.8	Sensitivity of each momentum parameterisation to the amount of prescribed attenuation. The global kinetic energy is average over the final 5 years of the spin-up, and plotted against the percentage by which the parameterisation is attenuated. Note that the parameterisations from the RVM algorithm and Anstey and Zanna (2017) diverge to infinity at 70% attenuation and higher, while the convolutional neural network produces un-physical flow fields at 90% attenuation and higher. . . .	166
7.9	An illustration of the instabilities that develop when the fully-convolutional neural network is implemented at 100% strength. The zonal velocity is shown at days 120, 160, and 220 of the spin-up. On day 120, instabilities develop in the bottom left corner of the domain, which then spread across the entire domain.	167
7.10	The probability distribution functions (PDFs) of the kinetic energy, for three different regions of the model - these regions are illustrated in Panel (a). The PDFs for regions 1, 2, and 3 are shown in Panels (b), (c), and (d) respectively. The kernel width parameter used in the PDF estimation scheme is the same for all regions, which is why the curves in Panel (c) and (d) are less smooth, as the kernel width (0.01) is of the same magnitude as the width of the peaks.	169
7.11	The kinetic energy spectra of the three parameterised 30km models, in addition to the 30km model with no parameterisation, and the high-resolution 3.75km model. The spectra are calculated over a 10 year period.	170
7.12	A comparison of the computational time to run the shallow water model for 10 years for various resolutions and parameterisations. . . .	173

Chapter 1

Introduction

Mesoscale eddies, turbulent features of length-scale 10-100km, are ubiquitous across the global ocean. They are the "weather" of the ocean and appear as a mixture of coherent vortices, filaments, and spirals. The dynamics of mesoscale eddies are nonlinear and chaotic, evolve over time-scales on the order of a few weeks, and are in approximate geostrophic balance. Satellite observations are often used to show the rich 'texture' of the mesoscale eddy field at the surface, but mesoscale eddies are three-dimensional features that impact the vertical motions of the ocean, down to depths of approximately 500-1000m.

Mesoscale eddies are particularly vigorous in regions of strong western boundary currents, such as the Gulf Stream and Kuroshio Extension, as well as the Antarctic circumpolar current. The strength and variability of these ocean jets are enhanced by the momentum fluxes of mesoscale eddies (Starr, 1966; Greatbatch et al., 2010a). Mesoscale eddies also have a significant impact on the ocean stratification and residual circulation (Farneti et al., 2010), and the transport of important climatic tracers such as heat and carbon (Gnanadesikan et al., 2015). The interactions between mesoscale eddies and the atmosphere also have a significant impact on both oceanic and atmospheric dynamics (Ma et al., 2016; Woollings et al., 2010; O'Reilly et al., 2017).

Although mesoscale eddies exist on spatial-scales that are small relative to the scales of ocean basins, $\mathcal{O}(1000\text{km})$, they dominate the oceanic kinetic energy (Ferrari and Wunsch, 2009; Wortham and Wunsch, 2014). Their energy is derived through a combination of baroclinic and barotropic instability, processes that extract available potential energy and kinetic energy respectively from the large-scale ocean state. The

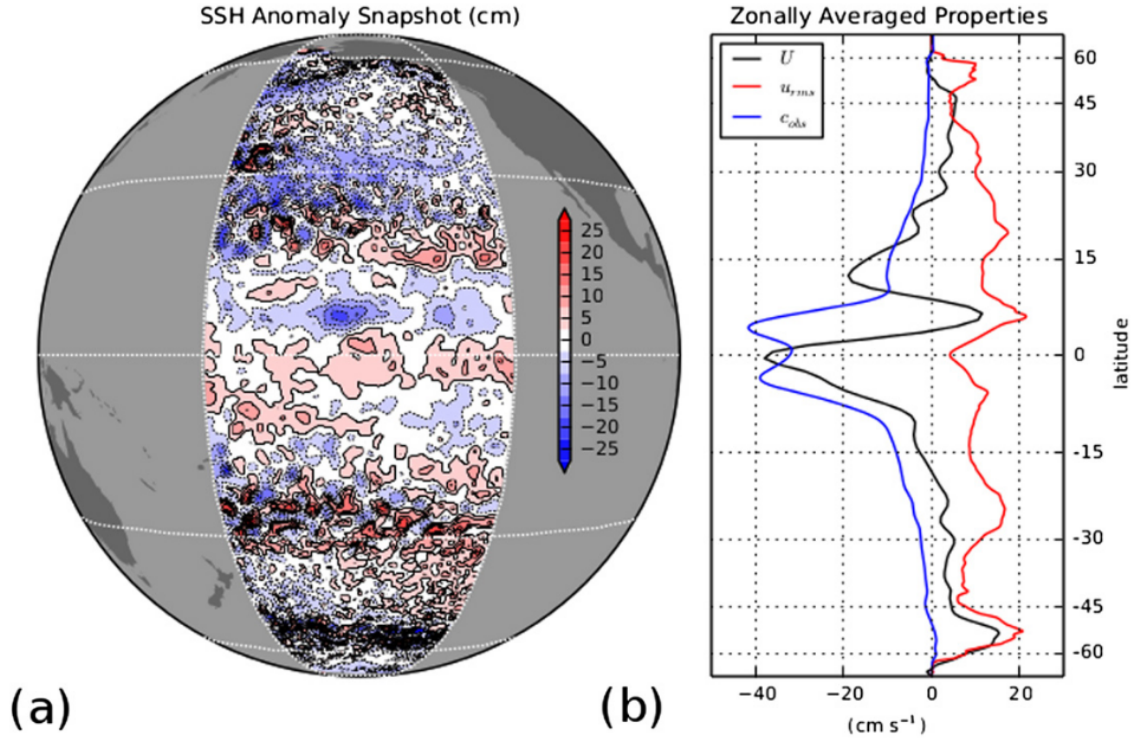


Figure 1.1: From the study of Klocker and Abernathey (2014): Panel (a) shows an observational snapshot of the sea surface height anomaly in a sector of the Pacific (180°-130°W), and Panel (b) shows the corresponding zonally-averaged zonal velocity (black curve), eddy velocity (red curve), and eddy phase-speed (blue curve).

resulting mesoscale eddy fields are complex, as illustrated by the sea surface height snapshot in Figure 1.1a.

Currently, climate models do not fully resolve mesoscale eddies. For example, in the most recent phase of the Coupled Model Intercomparison Project, CMIP5, the spatial-resolutions of the ocean components ranged from 0.2° to 2° (Taylor et al., 2012). At a resolution of 1°, virtually no mesoscale eddies are resolved. At a resolution of 0.2°, the mesoscale eddy field is only partially resolved, such that only some turbulent features of the flow field begin to emerge - resolutions at which eddies are partially resolved are known as ‘eddy-permitting’. The models of the CMIP5 project made important progress in understanding the future of our climate, as well as being included in the fifth assessment of the Intergovernmental Panel on Climate Change (IPCC). However, these models are not fully capturing the physical effects of mesoscale eddies; we discuss the importance of such effects in Section 1.1.

To resolve mesoscale eddies, the spatial resolution of these climate models could

simply be increased, but the accompanying computational costs prevent us from doing so. For example, assume we would like to double the horizontal resolution of an ocean model from 0.2° to 0.1° - what is the corresponding increase in the computational cost? Halving the grid size leads to four times the number of points on the horizontal grid, therefore quadrupling the computational cost. For numerical stability, the time-step is also reduced by approximately a half when halving the grid size, so the increase in computational cost is closer to a factor of $\times 8$. This is a large increase in computational cost, considering that increasing the resolution from 0.2° to 0.1° still does not fully resolve mesoscale eddies at all latitudes (Hallberg, 2013).

Is there a chance that high-performance computing could sufficiently improve, such that mesoscale eddies are resolved in the near future? The rate at which computational resources have increased over the past few decades has been dramatic, leading to Moore's Law (Moore et al., 1965; Schaller, 1997), the observation that processing power roughly doubles every two years. Even though Moore's Law has started to slow in recent years (Waldrop, 2016), the processing power of computers will likely continue to increase. Exascale computing, i.e., supercomputers that conduct 10^{18} floating-point operations per second, is expected to become operational in the next few years, becoming affordable by individual institutions in the next decade (Palmer, 2014). Hence, it is possible that progress in technology, and not physics, could improve simulations of climate.

There are two main issues with relying on technological progress alone to improve the representation of mesoscale eddies in climate models. First, if exascale computing is achieved, the additional computational power will have to be used to increase the resolution of both the oceanic and atmospheric components of the climate model, as unresolved cloud physics are also a large source of uncertainty in climate projections (Schneider et al., 2017b). Even with the advent of exascale computing, there might not be enough additional computing power to simultaneously resolve mesoscale eddies and cloud processes.

Secondly, assume that climate models will soon be able to fully resolve mesoscale eddies. Will the ocean component of coupled models become a negligible source of uncertainty in climate projections? It depends, as submesoscale eddies will remain unresolved for the foreseeable future. Submesoscale eddies, turbulent features of the

ocean occurring at length scales smaller than 10km, play an important role in the energy budget of the ocean (Su et al., 2018b). It is not yet clear exactly how much uncertainty submesoscale eddies produce in climate projections, but it suggests that despite technological progress in high-performance computing, the issue of resolution will remain.

Therefore, we need to improve how we capture the effects of unresolved eddies in climate models. This is known as the parameterisation or the closure problem, and relates more generally to turbulent systems numerically solved on a finite-grid or in truncated Fourier space, where convergence of the solutions cannot be demonstrated. In both cases, there are physical processes at small-scales not captured by the model that must be parameterised. We discuss the concept of parameterisation in more detail, in addition to reviewing the literature, in Section 2.2. But first, exactly why do we need to represent the effects of eddies in climate models?

1.1 The role of mesoscale eddies in the ocean and atmosphere

Mesoscale eddies derive their energy through barotropic and baroclinic instability (Vallis, 2017; Marshall and Plumb, 2016). Barotropic instability occurs in the presence of strong lateral shears, where eddies extract kinetic energy (KE) from the large-scale state. Baroclinic instability begins with atmospheric wind forcing, which acts to steepen surfaces of constant potential density in the ocean, known as isopycnals. Mesoscale eddies tend to flatten isopycnals and in the process convert available potential energy (APE) to KE. The size of this effect is proportional to the baroclinicity, $\nabla\rho \times \nabla p$, where ρ is the density, p is the pressure, and ∇ is the gradient operator. In other words, baroclinic instability occurs when surfaces of pressures and density are misaligned (Holton, 1973).

Baroclinic instability occurs primarily at the Rossby deformation radius, $L_R = \frac{NH}{\pi f}$, where N is the Brunt-Väisälä frequency, H is the vertical height of the fluid layer, and f is the Coriolis frequency. The Brunt-Väisälä frequency $N = \sqrt{-\frac{g}{\rho_0} \frac{\partial \rho(z)}{\partial z}}$ (Vallis, 2017) is a measure of the stability of a fluid due to vertical variations in density and is used as a measure of stratification. The Rossby deformation radius is the length-scale at which rotational effects become as important as buoyancy effects

and sets the characteristic length-scale of the mesoscale eddy field. The largest eddy length-scales of more than 100km occur near the equator, decreasing to below 10km at high latitudes; the latitudinal dependence of the eddy length-scale can be seen in the surface height anomaly snapshot in Figure 1.1a.

So baroclinic and barotropic instabilities transfer APE and KE respectively from the mean-state to the smaller-scale mesoscale eddy field. The stores of APE and KE in the ocean are referred to as reservoirs, with the APE reservoir three orders of magnitude larger than the KE reservoir (Ferrari and Wunsch, 2009). The exchange of energy between various reservoirs of a geophysical system was first conceived in the Lorenz Energy Cycle (Lorenz, 1955; Holton, 1973), which mathematically described how reservoirs of mean and eddy kinetic and potential energy interacted with one another. Through nonlinear eddy-eddy and eddy-mean interactions, the KE is transferred between various spatio-temporal scales. Mesoscale eddies therefore play a key role in the exchange of energy between reservoirs and across scales of the ocean.

The study of Charney (1971) suggested that the turbulence of the oceans and atmosphere resemble that of two-dimensional turbulence, which exhibits an inverse cascade of energy to larger scales (Fjørtoft, 1953) and a forward cascade of enstrophy to small scales. These two-dimensional cascades can be identified by their slopes in spectral space (Kraichnan, 1967). The two-dimensional perspective of the energy cycle was then advanced by Rhines (1977) and Salmon (1978) to include baroclinic effects. The quasi-2D Rhines-Salmon framework suggests that energy is introduced at large-scales into the baroclinic mode by wind forcing, which then cascades to smaller scales. The forward cascade of energy in the baroclinic mode continues until the Rossby deformation radius is reached, where energy is converted to the barotropic mode. The Rhines-Salmon framework suggests that an inverse energy cascade occurs in the barotropic mode. The inverse cascade produces increasingly larger eddies, until they are arrested by the beta effect of the Coriolis force (Rhines, 1975) or bottom friction dissipates their kinetic energy. The beta effect arresting the inverse cascade is related to the conservation of potential vorticity and Rossby wave generation.

An inverse energy cascade has been observed in the ocean using satellite altimetry data (Scott and Wang, 2005; Arbic et al., 2013), as well as in models (Kjellsson and Zanna, 2017). However, there is evidence that surface measurements of ocean currents

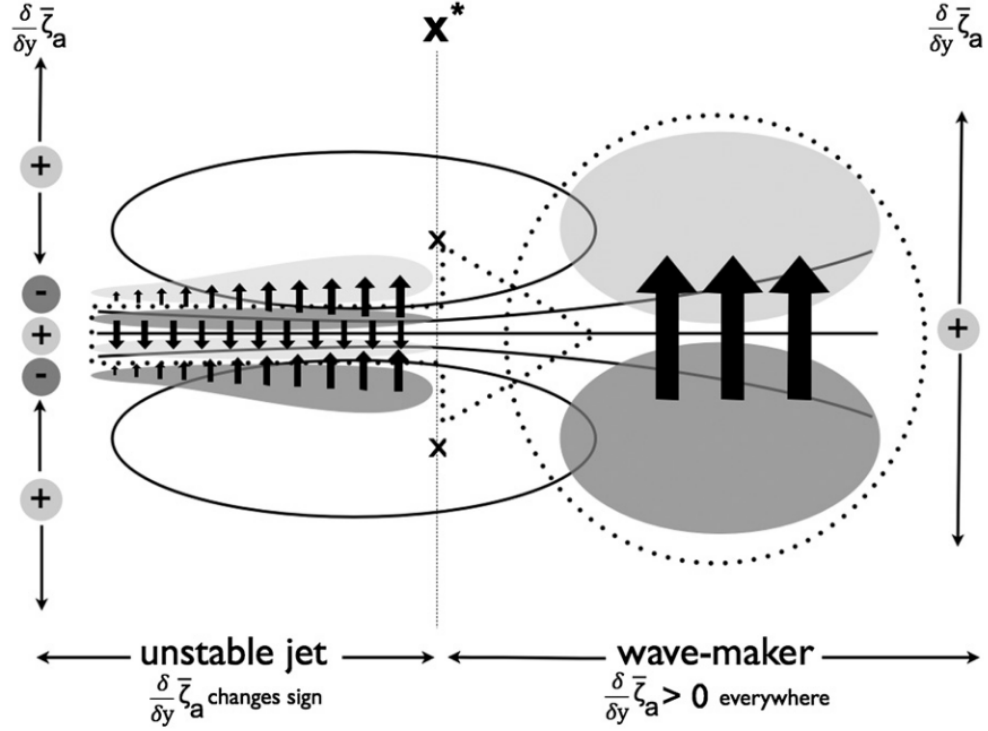


Figure 1.2: The two-regime description of eddy-mean flow interaction from the idealised study of Waterman and Jayne (2010), with the figure adapted from Waterman and Hoskins (2013). The unstable-jet and wave-maker regimes, of the up- and downstream regions respectively, are illustrated, highlighting the patterns of the large-scale absolute vorticity gradients. Solid black arrows denote the direction of the divergent time-mean eddy vorticity fluxes and their patterns of convergence/divergence. Regarding the solid-filled contours, light gray shading indicates positive values (an eddy vorticity flux convergence) and dark gray shading indicates negative values (an eddy vorticity flux divergence).

primarily reflect the first baroclinic mode (Wunsch, 1997), contradicting the theory that only the barotropic mode exhibits an inverse cascade of energy. Nonetheless, the energy cycle of the ocean is a combination of forward and inverse cascades of energy, in which mesoscale eddies play an important role.

Regions of the global ocean with strong westerly winds, and therefore large reservoirs of APE, have particularly energetic mesoscale eddy fields. For example, the Gulf Stream in the North Atlantic, the Kuroshio Extension in the North Pacific, and the Southern Ocean. At the surface, these regions exhibit large-scale zonal meandering jets with current speeds of over 1ms^{-1} (Ducet and Le Traon, 2001). How do mesoscale eddies interact with strong zonal jets?

The answer depends on exactly where along the jet you are considering. In an idealised ocean model, Waterman and Jayne (2010); Waterman and Hoskins (2013) showed that eddy fluxes stabilise the unstable jet in the up-stream region, whereas down-stream, eddies drive the time-mean recirculation through wave radiation. Evidence of this along-jet variation in eddy effects was found in observations of the Kuroshio Extension (Waterman et al., 2011). An illustration of the two-regime model of eddy-mean flow interaction is shown in Figure 1.2, where the nature (i.e. divergence and convergence) of the eddy vorticity fluxes changes significantly between the up- and down-stream regions. Eddy-mean flow interaction is therefore not as simple as stating that eddies exclusively weaken or strengthen large-scale jets.

Using satellite altimetry data, the study of Greatbatch et al. (2010b) showed that eddy momentum fluxes exhibit significant along-jet variation in both the Gulf Stream and Kuroshio Extension. The time-mean maps (Figure 2 of Greatbatch et al., 2010b) show that, for one of the eddy momentum fluxes, the sign rapidly alternates between positive and negative moving along the core of the jet. The diagnosed eddy momentum fluxes of the Gulf Stream in Greatbatch et al. (2010b) were then used to drive a shallow-water model of the North Atlantic (Greatbatch et al., 2010a), and found that the eddy fluxes alone could reproduce a significant amount of the true Gulf Stream transport, including the recirculation gyres.

Within western boundary currents, mesoscale eddies also interact with the atmosphere, producing intense air-sea fluxes (Villas Bôas et al., 2015). The mechanical forcing, i.e., momentum exchange, of atmospheric winds leads to a systematic dampening of mesoscale eddies (Wilson, 2016; Xu et al., 2016), reducing their kinetic energy. Suppressing the interaction between mesoscale eddies and the atmosphere was found to weaken the Kuroshio Extension jet by 20-40% (Ma et al., 2016); a schematic of this mechanism is shown in Figure 1.3, which involves the exchange of energy between the large-scale jet, the eddies, and the atmosphere. Ma et al. (2016) found that the eddy-atmosphere feedback dominates the destruction of eddy potential energy, and when this feedback is removed, the energy budgets re-balance such that the mean kinetic energy decreases.

The mesoscale eddy field contributes to the transport and mixing of climatic tracers, such as heat (e.g. Wunsch, 1999; Jayne and Marotzke, 2002; Danabasoglu and

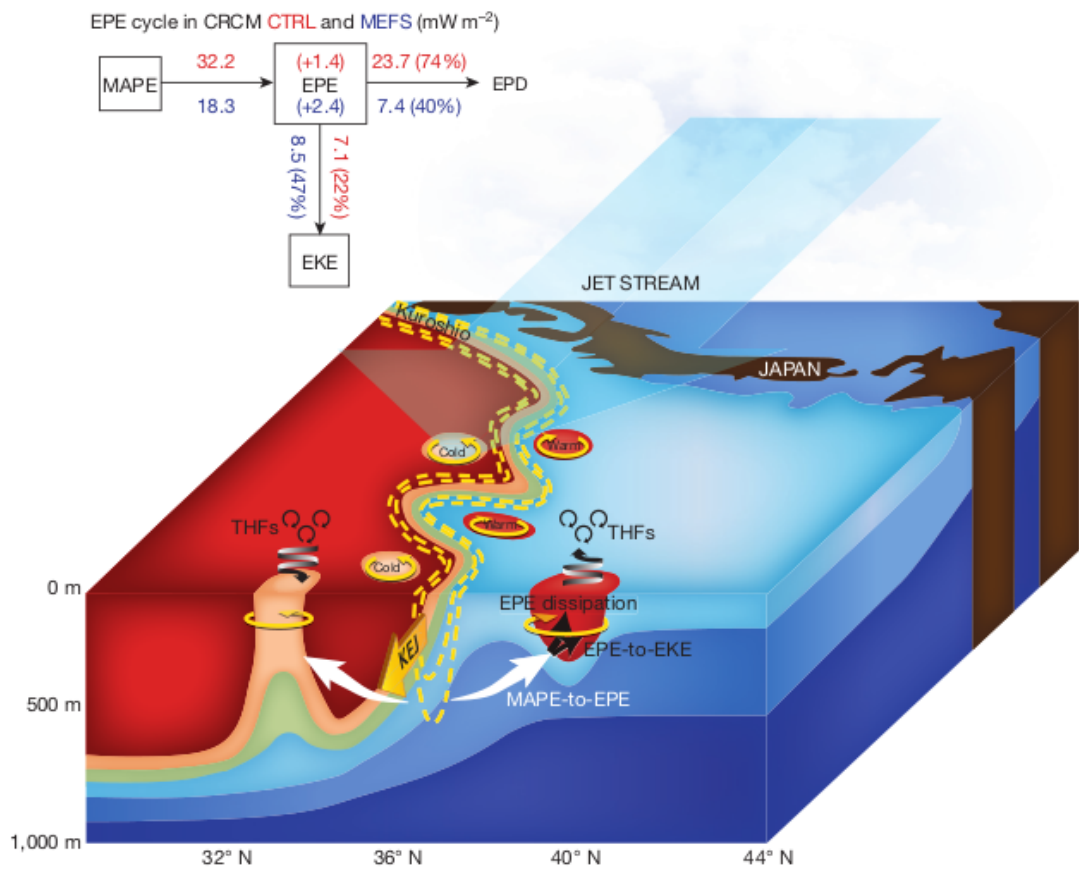


Figure 1.3: A schematic of the feedback between mesoscale eddies and the atmosphere. The interaction regulates the strength of the Kuroshio through exchanges of energy between various reservoirs.

Marshall, 2007; Griffies et al., 2015). Within jets, the strong mean flow suppresses mixing of tracers (Ferrari and Nikurashin, 2010), whereas eddies in general act to enhance mixing, with significant spatial variability (e.g. Abernathey and Marshall, 2013). In the Gulf Stream, mixing is suppressed at the surface but not at depth (Bower et al., 1985; Bower and Rossby, 1989). The mixing by mesoscale eddies along isopycnals has a significant impact on the carbon uptake of the ocean (Gnanadesikan et al., 2015).

So what are the consequences of not resolving mesoscale eddies in an ocean model? Increasing the resolution from approximately 1° to 0.1° improves mean sea surface temperatures, increases the amplitude of El Niño-Southern Oscillation (ENSO), and significantly modifies local air-sea feedbacks (Kirtman et al., 2012). The spatio-temporal variability of sea level anomalies improves at eddy-permitting resolutions, including fluctuations on longer interannual time-scales (Penduff et al., 2010).

Resolving mesoscale features within the Gulf Stream has a significant impact on models of atmospheric dynamics across Europe (Minobe et al., 2008; O’Reilly et al., 2017; Haarsma et al., 2019). At eddy-permitting resolutions, air-sea interactions more closely resemble observations, but further increasing the resolution to eddy-resolving reduces long-standing temperature biases in the North Atlantic (Roberts et al., 2016).

1.2 Thesis aims and structure

Capturing the effects of unresolved eddies is crucial for improving both the oceanic and atmospheric components of climate models. The difficulty lies in ‘capturing’ these processes accurately. As we will see in Section 2.2, traditional approaches to eddy parameterisations can sometimes over-simplify the underlying dynamics. To accurately capture the complex spatio-temporal variability of mesoscale eddies, we need to move beyond ‘bulk formulas’. We should use every tool at our disposal to construct eddy parameterisations, which include modern data-driven approaches.

Specifically, we want to use data-driven methods to study and parameterise eddy momentum and tracer fluxes. The aims of this thesis are as follows:

- to quantify the regional and temporal variability of tracer fluxes driven by observations of mesoscale eddies, focusing on the North Atlantic;

- to construct parameterisations of eddy momentum fluxes that more accurately capture the underlying dynamics, using modern machine learning techniques;
- to use data-driven algorithms to parameterise eddy fluxes, but in a physically interpretable manner, such that the underlying dynamical processes are not obscured;
- to determine the physical and numerical impact of data-driven eddy parameterisations when implemented back into an ocean model.

In Chapter 2, we present the equations of motion that govern ocean dynamics, and review existing eddy parameterisations from the literature. In Chapter 3, we provide details of each numerical ocean model used in this thesis, as well as how we define and quantify the effects of eddies. Chapter 3 also includes an introduction to supervised machine learning and provides details on the specific algorithms used in the results chapters of this thesis. Chapter 4 is the first of the results chapters, where we investigate eddy fluxes in the North Atlantic using an observation-driven passive tracer experiment. In Chapter 5, we train convolutional neural networks to represent eddy momentum fluxes in an idealised ocean model, while testing the generalisation ability of the networks and how to enforce physical constraints. We use machine learning to ‘discover equations’ from data to construct physically interpretable eddy parameterisations in Chapter 6. In Chapter 7, we test our new data-driven eddy parameterisations by implementing them back into an ocean model. We end this thesis in Chapter 8, where we summarise our results and discuss the implications of our findings.

The term ‘we’ is used throughout this thesis, with all the work being conducted by myself under the supervision of Professor Laure Zanna. If a piece of analysis or an experiment was not conducted by myself, it will be explicitly stated, with only the work of Chapter 4 containing an external collaborative element. If the results of a chapter have been published in some form, the details of that publication will be provided in the introduction of that chapter. One final note regarding the electronic version of this thesis: all sections, subsections, tables, equations, and images are hyperlinked, to allow easy access from the table of contents and the list of figures.

Chapter 2

Background

2.1 Equations of motion of a geophysical fluid

We briefly review the equations of motion that govern ocean dynamics, starting with the Navier-Stokes equations for a generic Newtonian fluid, and then approximating those equations based on balances of the global ocean. The Navier-Stokes equations, for a continuous fluid in three-dimensions on a rotating planet, are given by

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\mathbf{f} \times \mathbf{u} - \frac{1}{\rho} \nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{F}, \quad (2.1)$$

where $\mathbf{u} = (u, v, w)$ is the three-dimensional velocity vector, ∇ is the three-dimensional gradient operator, $\mathbf{f}$ is the Coriolis parameter, ρ is the density (mass per volume), ν is the kinematic viscosity, $\mathbf{F}$ represents all external forces, and we have neglected the centrifugal force. The effect of molecular viscosity, i.e., the frictional force within the fluid, is approximated by the term $\nu \nabla^2 \mathbf{u}$. The momentum field $\rho \mathbf{u}$ is proportional to $\mathbf{u}$, so Equation 2.1 determines how the momentum of a fluid evolves in time. The left-hand side of Equation 2.1 can be written using the material derivative $\frac{D}{Dt} = \frac{\partial}{\partial t} + (\mathbf{u} \cdot \nabla)$, which represents the rate of change following a fluid parcel, with the second term $(\mathbf{u} \cdot \nabla)$ representing advection by the flow. Therefore, $\frac{D\mathbf{u}}{Dt}$ represents the acceleration following a fluid parcel.

However, the Navier-Stokes equations do not exist in isolation. We have five unknown quantities - the velocity components, density, and pressure - but only three equations, one for each velocity component. Two other equations are needed to close the system. The first is derived from the fact that mass is conserved. Given a fluid parcel of variable density ρ and volume ΔV , the Lagrangian rate of change of mass

must be zero, $\frac{D(\rho\Delta V)}{Dt} = 0$. From this assumption, one can derive the mass continuity equation, given by

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0, \quad (2.2)$$

which can be interpreted as the following: any Eulerian change in the density must be balanced by a net flux of density into (or out of) the volume. If time fluctuations of the density field are also assumed to be small, i.e., $\frac{D\rho}{Dt} = -\rho(\nabla \cdot \mathbf{u}) \approx 0$, then the equation for a non-divergent velocity field $\nabla \cdot \mathbf{u} = 0$ is recovered. Within Equation 2.1, the Boussinesq approximation can be applied such that density variations are ignored, except where multiplied by the gravitational acceleration g .

The second equation is the equation of state. For the atmosphere, the ideal gas equation is used. For the ocean, however, we cannot derive an equation of state from first principles. We know that density will be a function of pressure, temperature, and salinity, so we approximate the equation and estimate any coefficients empirically. Most ocean models use a non-linear equation of state, but we will only concern ourselves with the first-order approximation, which assumes density changes are linear perturbations to the mean state, in other words

$$\rho = \rho_0(1 - \beta_p(p - p_0) - \beta_T(T - T_0) - \beta_S(S - S_0)), \quad (2.3)$$

where p is pressure, T is temperature, S is salinity, and ρ_0 , p_0 , T_0 , S_0 are the reference density, pressure, temperature, and salinity values respectively. The variables β_p , β_T , and β_S are empirical parameters to be estimated from observations.

We now have an additional equation but introduced two new variables: S and T . Therefore, for the ocean, we have to introduce two more prognostic equations for temperature and salinity. These prognostic equations typically take the form of an advection-diffusion equation with additional sources and sinks. In particular, the prognostic equation for temperature can be derived by considering the first law of thermodynamics, which represents conservation of energy. The first law implies that the change of internal energy of a fluid parcel must be equal to all heat fluxes in and out of the parcel, in addition to any sources or sinks of heat. The result is the following equation containing, advection, diffusion, and sources,

$$\frac{\partial T}{\partial t} + (\mathbf{u} \cdot \nabla)T = \nu_T \nabla^2 T + J, \quad (2.4)$$

where ν_T is the thermal diffusivity and J is any sources or sinks of heat. A similar prognostic equation exists for the salinity, S . Note, the work done on a fluid parcel is proportional to $p\partial V$, where V is the volume, but this term does not contribute to Equation 2.4 when the fluid is incompressible ($\nabla \cdot \mathbf{u} = 0$).

Temperature and salinity are active tracers, such that the flow field impacts the tracer distribution and the tracer then impacts the flow via changes in density. Passive tracers, however, are simply transported (advected and diffused) by the flow and have zero effect on the underlying flow field. Some active tracers have such a small effect on the flow that they are regarded as passive tracers (e.g. ocean chlorophyll).

2.2 Parameterisations of turbulent ocean eddies

As mentioned in the introduction, the equations of motion of an ocean model are numerically integrated at a finite spatial and temporal resolution. The resolution at which a model is run is severely restricted by the computational cost, such that the smallest scales are truncated. The ocean components of some climate models are approaching a resolution of 30km in CMIP6 (Eyring et al., 2016), which is comparable to the first baroclinic deformation radius at mid-latitudes. However, many of the models of CMIP5 have a resolution of approximately 100km, which resolves virtually none of the mesoscale eddy field. The Rossby deformation radius (200km at the equator, less than 10km at high latitudes) determines the length at which mesoscale eddies proliferate. Therefore, ocean models should aim to have a horizontal resolution smaller than the Rossby deformation radius, to start capturing the non-linear effects of mesoscale effects.

The dynamical effects of unresolved mesoscale eddies on the large-scale flow must, therefore, be parameterised in some way, i.e., introducing additional terms into the equations of motion which represent those unresolved dynamical effects. As an illustrative example, take the Navier-Stokes equations for the time-evolution of the velocity field $\mathbf{u}$, defined in Equation 2.1. We cannot solve Equation 2.1 analytically, so we instead numerically integrate the equations on a finite spatial grid (or in truncated spectral space). To make the distinction between the exact dynamical quantities and those quantities solved on a finite grid, we denote the quantities on the finite grid using an over-bar ($\overline{..}$). Although climate models solve a modified set

of equations called the primitive equations, for illustrative purposes, let us consider numerically solving the full Navier-Stokes

$$\frac{\delta \bar{\mathbf{u}}}{\delta t} + (\bar{\mathbf{u}} \cdot \bar{\nabla}) \bar{\mathbf{u}} = -\mathbf{f} \times \bar{\mathbf{u}} - \frac{1}{\bar{\rho}} \bar{\nabla} \bar{p} + \bar{\nu} \bar{\nabla}^2 \bar{\mathbf{u}} + \bar{\mathbf{F}}, \quad (2.5)$$

where the infinitesimal change ∂t is now a finite time-step δt . By numerically solving on a finite grid, we have introduced structural errors into the model, such that $\frac{\partial \mathbf{u}}{\partial t} \neq \frac{\delta \bar{\mathbf{u}}}{\delta t}$. The structural error makes the model less physically representative of reality and originates from not resolving all spatio-temporal scales of the physical system, i.e., the smallest scales of the model are truncated. For an ocean model, the truncation of scales leads to not fully resolving mesoscale eddies - and many other processes.

The truncation of scales also has a significant impact on the energy cycle of ocean models. Models that do not fully resolve mesoscale eddies have an overall lack of energy at all scales and weaker inverse cascade of energy to larger scales (Kjellsson and Zanna, 2017); as discussed in the introduction, the inverse cascade of energy is a key component of the energy cycle and is present in observations (Scott and Wang, 2005; Scott and Arbic, 2007). The inverse cascade can be non-existent if the model does not resolve the Rossby deformation radius. Even though only the smallest scales of models are truncated, the resulting dynamics at all scales become less realistic, making eddy parameterisations even more important.

Parameterising unresolved mesoscale eddies involves adding additional terms to Equation 2.5, to account for their effects on the resolved state $\bar{\mathbf{u}}$, making the model reflect reality more accurately. Let the over-bar $(\overline{\cdot})$ now represent some form of average (as opposed to a variable on finite grid). The average $(\overline{\cdot})$ can be a spatial, temporal, or ensemble mean. We can write a variable as the combination of its mean and the fluctuation from that mean $\mathbf{u} = \bar{\mathbf{u}} + \mathbf{u}'$. The Reynolds average is a specific type of average, in either space or time (or both, as in Large Eddy Simulations, e.g., Pope, 2001) such that $\overline{\mathbf{u}'} = 0$. The two-dimensional Reynolds-averaged Navier-Stokes equations, i.e., the equation for $\frac{\partial \bar{\mathbf{u}}}{\partial t}$, has an additional term, given by

$$\nabla \cdot (\overline{\mathbf{u}' \mathbf{u}'}) \equiv \nabla \cdot \begin{pmatrix} \overline{u' u'} & \overline{u' v'} \\ \overline{v' u'} & \overline{v' v'} \end{pmatrix}. \quad (2.6)$$

This term represents the interaction between $\bar{\mathbf{u}}$ and $\mathbf{u}'$, and the tensor on the right-hand side of Equation 2.6 is known as the Reynolds stress tensor. We cannot explicitly

calculate the Reynolds stress tensor, so a dynamical model is needed for the relationship between $\bar{\mathbf{u}}$ and $\mathbf{u}'$, to ‘close’ the equations. In other words, we can attempt to write down equations for these second-order terms $\overline{u'u'}$, $\overline{u'v'}$, and $\overline{v'v'}$, but then triple-order terms would appear, e.g., $\overline{u'v'v'}$. Writing down an equation for $\overline{u'v'v'}$ would then yield fourth-order terms and so on. This is the turbulence closure problem (Boussinesq, 1872; Pope, 2001), and is more general in that it applies to any averaging procedure - not just Reynolds averaging. Parameterising mesoscale eddy fluxes in climate models is therefore a form of the turbulence closure problem.

In this section, we provide a non-exhaustive review of existing eddy parameterisations. We begin with the traditional physics-driven approach and then contrast with the now-popular data-driven approach to constructing parameterisations. By data-driven, we mean employing an algorithm to learn the parameterisation directly from data - with minimal human input. We show that most parameterisations - for any geophysical process, not just oceanic - exist on a spectrum, ranging from purely physics-driven at one end, to purely data-driven approaches at the other.

The crux of parameterising physical processes within models is how to represent the effects of some unresolved quantity x' on the resolved state $\bar{x}$. Models cannot explicitly calculate these effects as, by definition, the model does not have access to the unresolved quantities x' . We then have no choice to use the resolved state $\bar{x}$ to approximate the effects of x' , using some function $f(\bar{x})$.

The challenge lies in constructing the function $f(\bar{x})$ which, in an ideal world, achieves the following: accurately captures the dynamical processes being parameterised, respects physical principles such as conservation laws, is numerically stable when implemented into a model, and generalises to new dynamical regimes - while only using the resolved state $\bar{x}$. On top of this, the effect being parameterised is typically a complex non-linear dynamical process. This is why parameterising unresolved processes in climate models can be such a difficult task.

Physics-driven and data-driven parameterisations differ in how this task is tackled, in particular how the function $f(\bar{x})$ is constructed, with both approaches having advantages and disadvantages. We now discuss these differing approaches and introduce the parameterisation spectrum lying between data-driven and physics-driven.

The parameterisation spectrum

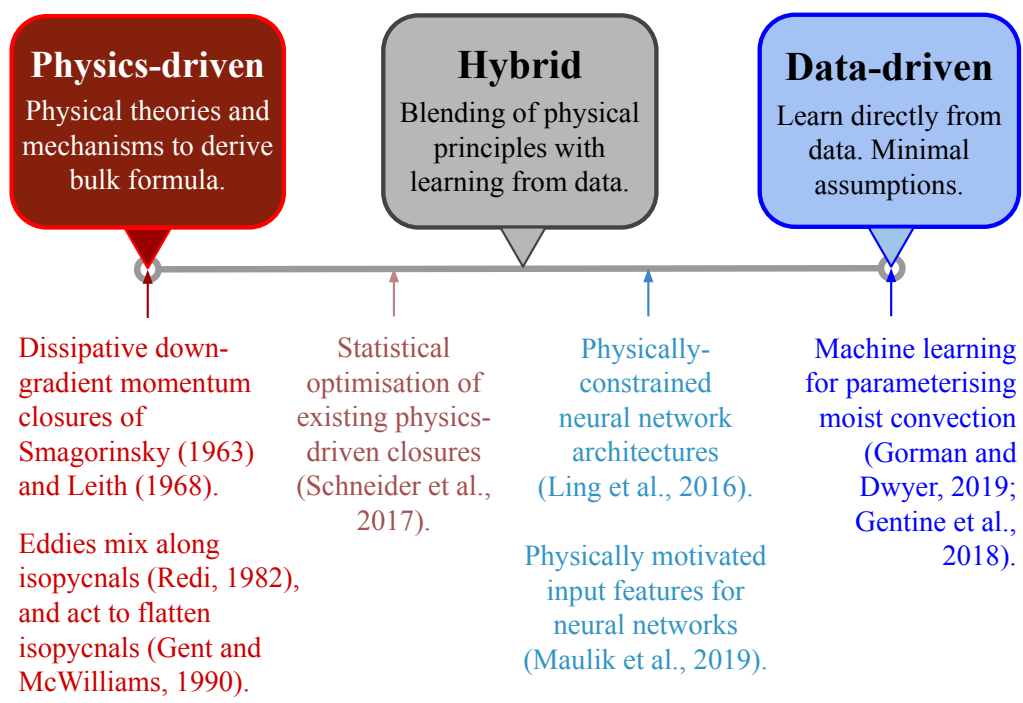


Figure 2.1: An illustration of the spectrum of approaches to constructing parameterisations, from the traditional physics-driven approach on the left-hand side, moving to purely data-driven approaches at the far right-hand side.

The parameterisation spectrum is illustrated in Figure 2.1, with specific examples at different stages - these examples are discussed in more detail below. The far left-hand side represents the purely physics-driven approach, and this has been the prevailing approach for the past few decades. The physics-driven approach starts with some physical principle, mechanism, or theory, related to the dynamical process to be parameterised. From the physical knowledge, a bulk formula is derived to approximate the effect of that process. In other words, we assume that the unresolved state x' impacts $\bar{x}$ through mechanism A , which leads us to the functional form $f_A(\bar{x})$.

At the other end of the parameterisation spectrum, the purely data-driven approach makes zero physical assumptions about the process to be parameterised. Instead, it is data that guides the construction of the parameterisation, where the data can be from high-resolution models or observations. An algorithm is employed to directly learn the parameterisation from the data. Note that we use the term algorithm loosely here, as the ‘algorithm’ could be simple linear regression, or more complex statistical or machine learning techniques. The choice of algorithm will implicitly make assumptions regarding the parameterisation, but in general, the functional form and parameters are empirically estimated from the data.

Below, we present specific examples of parameterisations from the literature that lie more towards the physics-driven or data-driven side of the spectrum. We also discuss some examples which are hybrid approaches, blending empirical estimation from data with some physically-motivated component. The blending could involve estimating the parameters of a physics-driven parameterisation from data, or applying physical constraints to a data-driven algorithm.

2.2.1 Physics-driven parameterisations

We start with the simplest and most common type of momentum parameterisation, known as eddy viscosity, first introduced conceptually by Boussinesq (1903). In Equation 2.1, molecular diffusion is represented by the term $\nu \nabla^2 \mathbf{u}$ for an incompressible fluid and can be derived from Kinetic Theory (Landahl et al., 1989; Pope, 2001). In molecular diffusion, the random motion of molecules leads to a flux of molecules from high- to low-concentrations (Fick, 1855). Molecular diffusion occurs across all length-scales but becomes small at length-scales large than the Kolmogorov scale; below the

Kolmogorov scale, turbulent kinetic energy is dissipated into heat. The coefficient ν is the molecular viscosity and determines the ‘strength’ the internal friction of the fluid.

With regards to eddy viscosity models, the dynamics of eddies are assumed to be ‘random’, such that their motions re-distribute momentum in a manner similar to molecular diffusion, i.e., the flux of momentum is proportional to the large-scale gradient of momentum. In other words, eddy fluxes are assumed to be on-average down-gradient and act to homogenise the resolved momentum field. Assuming mesoscale eddies diffuse momentum leads to the following Laplacian parameterisation

$$\hat{\mathbf{S}}_{\mathbf{u}} = \nu_e \nabla^2 \bar{\mathbf{u}}, \quad (2.7)$$

with ν_e now the *eddy* viscosity, and has the same units m^2s^{-1} as the molecular viscosity. Note that we denote the true eddy momentum forcing as $\mathbf{S}_{\mathbf{u}}$ and the parameterisation as $\hat{\mathbf{S}}_{\mathbf{u}}$. The key difference between eddy viscosity and molecular viscosity is that the eddy viscosity coefficient is orders of magnitude larger than that of molecular viscosity. An eddy viscosity parameterisation improves the numerical stability of ocean models and dissipates enstrophy and energy at the grid-scale.

The parameterisations of Smagorinsky (1963) and Leith (1968) build upon the eddy viscosity concept by constructing the eddy viscosity coefficient to have a functional dependence on the resolved flow. The Smagorinsky (1963) viscosity coefficient is based on the energetics of 3D isotropic turbulence and is based on the energy flux at any particular length scale being constant. This length-scale independent energy flux is constructed using the expression for the Kolmogorov scale and is given by

$$\nu_e = \left(\frac{\alpha}{\pi}\right)^2 \Delta x^2 \sqrt{D^2 + \tilde{D}^2}, \quad (2.8)$$

where Δx is the horizontal resolution, and α is a parameter to be prescribed, which relates to the proportionality of the expression with the Kolmogorov length scale. The terms $D = \frac{\partial \bar{u}}{\partial y} + \frac{\partial \bar{v}}{\partial x}$ and $\tilde{D} = \frac{\partial \bar{u}}{\partial x} - \frac{\partial \bar{v}}{\partial y}$ are the shearing and stretching deformations respectively.

The Leith viscosity coefficient is based on the enstrophy flux of 2D turbulence, and in a manner similar to the Smagorinsky expression, the expression is constructed such that the flux of enstrophy is equal to the grid scale dissipation rate of enstrophy.

The enstrophy dissipation rate length scale, and the resulting eddy viscosity, can be estimated from dimensional analysis and leads to

$$\nu_e = \left(\frac{\alpha}{\pi}\right)^2 \Delta x^3 \sqrt{\left(\frac{\partial \tilde{D}}{\partial \tilde{x}}\right)^2 + \left(\frac{\partial \tilde{D}}{\partial \tilde{y}}\right)^2}, \quad (2.9)$$

where α again is a parameter to be prescribed.

The study of Holland (1978) suggested that the differential operator in the eddy viscosity should be more scale selective, and should act less on the larger-scales as they are better resolved by the model. A biharmonic operator was suggested to replace the Laplacian, leading to the following momentum parameterisation

$$\hat{\mathbf{S}}_{\mathbf{u}} = -\nu_e \nabla^4 \bar{\mathbf{u}}, \quad (2.10)$$

where ν_e now has units of m^4s^{-1} . In spectral space, the Laplacian and biharmonic operators are proportional to K^2 and K^4 respectively, where K is the total wavenumber, showing that the biharmonic term will have a larger impact at high-wavenumbers and hence smaller scales. Similarly to the Laplacian operator, the coefficient of the biharmonic operator can be made flow-dependent with analogous Smagorinsky and Leith expressions.

Not all ocean models are written in terms of the velocity fields u and v . Another theoretically important quantity is the potential vorticity (PV), which is conserved following a fluid parcel in the absence of frictional or diabatic processes (Vallis, 2017). For example, quasi-geostrophic models assume that a fluid is *almost* in geostrophic balance, such that an ageostrophic component of the flow is permitted. More formally, quasi-geostrophic equations can be derived through a first-order Rossby number expansion of the Navier-Stokes equations. The key prognostic variable in quasi-geostrophic models is the quasi-geostrophic PV, defined as

$$q = \nabla^2 \psi + \beta y + \frac{\partial}{\partial z} \left(\frac{f_0^2}{N^2} \frac{\partial \psi}{\partial z} \right), \quad (2.11)$$

where f_0 is the Coriolis parameter value at a particular latitude, β is the Rossby parameter, $N = \left(-\frac{g}{\rho} \frac{d\rho}{dz}\right)^{\frac{1}{2}}$ is the Brunt-Väisälä frequency and ψ is the streamfunction for the non-divergent horizontal velocity $\mathbf{u} = \left(-\frac{\partial \psi}{\partial y}, \frac{\partial \psi}{\partial x}\right)$. The parameters f_0 and β arise from the beta plane approximation, which is a linear Taylor series approximation to the latitude-dependent Coriolis parameter $f \approx f_0 + \beta y$. Parameterising eddies

in the PV framework is not distinct from parameterising eddies in the momentum framework - parameterising one indirectly parameterises the other.

Just as eddy momentum fluxes can be parameterised with an eddy viscosity, so can eddy PV fluxes. A down-gradient PV closure was suggested by Rhines and Young (1982), where PV was found to be homogenised by mesoscale eddies in the gyres of both models and observations. This closure is also known as the ‘PV mixing’ closure and takes the following functional form

$$\hat{S}_q = \nu_e \bar{\nabla}^2 \bar{q}. \quad (2.12)$$

However, not all parameterisations assume that the eddy flux is directed down the mean gradient. A more sophisticated parameterisation is that of Mana and Zanna (2014), which assumes the sub-grid transport of PV can be modelled as a non-Newtonian fluid. Specifically, the eddy stress tensor is assumed to relate to a second-grade Rivlin-Ericksen fluid (Rivlin and Ericksen, 1955); the full-details of Rivlin-Ericksen fluids are beyond the scope of this thesis, but it leads to the eddy parameterisation containing time-derivatives and therefore considers the memory of the fluid. In the quasi-geostrophic equations, the second-grade Rivlin-Ericksen eddy stress leads to the following functional form in the PV equation

$$\hat{S}_q = \kappa \bar{\nabla}^2 \frac{D\bar{q}}{Dt}, \quad (2.13)$$

which captures up-gradient eddy fluxes and dissipates enstrophy. Note that in Mana and Zanna (2014), a stochastic version of the parameterisation is implemented to better capture the variability of the sub-grid eddy flux - a stochastic parameterisation predicts the parameters of a distribution, from which values are then sampled from. In this thesis, we focus on the deterministic case, where there is a one-to-one mapping from the resolved state Y to the value of the parameterisation $f(Y)$.

The time-independent component of Equation 2.13 has been studied as a momentum parameterisation in Anstey and Zanna (2017). This time-independent component takes the following form in the momentum equations

$$\hat{\mathbf{S}}_{\mathbf{u}} = \kappa \bar{\nabla} \cdot \begin{pmatrix} -\zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta D \end{pmatrix}, \quad (2.14)$$

where ζ is the relative vorticity. The tensor is one of the Galilean-invariant tensors of Pope (1975). The Anstey and Zanna (2017) expression relates to how a larger-scale flow deforms a smaller-scale vorticity field. The expression captures up-gradient momentum fluxes, conserves energy, and is a net sink of enstrophy.

Parameterisations that are purely down-gradient produce a few issues when implemented. First, they do not capture the full extent of the complex interaction of unresolved eddies with the resolved flow (Wilson and Williams, 2004). Secondly, purely down-gradient closures cause a spurious loss of energy in ocean models (Jansen and Held, 2014; Jansen et al., 2015); Laplacian and biharmonic momentum parameterisations are a net sink in the global energy budget. Kinetic energy is dissipated to heat within the ocean and such energy dissipation within the Navier-Stokes is permitted. However, within ocean models, the energy dissipated by parameterisations is excessive and produces under-energetic flows.

Eddy parameterisations can be constructed while correcting for this spurious loss of energy. For example, introducing a separate prognostic equation for the eddy energy. A closed system should conserve total energy, so it is physically-appropriate to have an additional equation to keep track of the sources and sinks of energy due to unresolved eddies. A separate eddy energy equation can be used when parameterising PV fluxes (Eden and Greatbatch, 2008; Marshall and Adcroft, 2010) or momentum fluxes (Jansen and Held, 2014; Jansen et al., 2015), in addition to buoyancy fluxes (Cessi, 2008; Mak et al., 2018).

We have not yet discussed buoyancy, the acceleration of a fluid parcel caused by being above (below) fluid of lower (higher) density, defined as $b = -g \frac{\rho'}{\rho_0}$. Fluid parcels move along surfaces of constant potential density, known as isopycnals. The potential density of a fluid parcel is the density that parcel would acquire if adiabatically brought to a reference pressure; the potential density of a fluid parcel is useful as it is conserved under changes in pressure. As previously mentioned, wind forcing acts to increase stratification, i.e., produce steeper isopycnal slopes, whereas eddies typically act to flatten isopycnals. The flattening of isopycnals is the exact process represented by the Gent and McWilliams (1990) parameterisation, where eddies are assumed to reduce stratification by extracting APE from the resolved flow. Redi (1982) is also a popular buoyancy parameterisation, that assumes eddies enhance mixing *along*

isopycnals, and can be used in conjunction with the Gent and McWilliams (1990) parameterisation.

Parameterising eddy buoyancy fluxes, or any other tracer fluxes, with the Gent and McWilliams (1990) scheme leads to an additional advective flux, $-\overline{\nabla} \cdot (\bar{b} \mathbf{u}^*)$, in the prognostic buoyancy equation. The velocity $\mathbf{u}^*$ is referred to as the bolus velocity; it is non-divergent and represents the advective transport of sub-grid eddies that flattens isopycnals. The Redi (1982) parameterisation leads to an additional diffusive flux, $\overline{\nabla} (K \overline{\nabla} b)$, where K is a tensor that projects the gradient $\overline{\nabla} b$ onto isopycnal surfaces.

The Gent and McWilliams (1990) parameterisation suffers the same problem as the dissipative momentum closures: they lead to a spurious loss of energy when implemented. As previously mentioned, the Gent and McWilliams (1990) parameterisation extracts APE to flatten isopycnals, but this energy is lost from the model. Hence buoyancy parameterisations are being constructed with a prognostic eddy energy equation (Eden and Greatbatch, 2008; Mak et al., 2018), or with a back-scatter momentum parameterisation to re-introduce the lost energy (Bachman, 2019).

2.2.2 Data-driven parameterisations

All the previous parameterisations are predominantly physics-driven, i.e., they were constructed by considering physical principles or mechanisms, from which the functional form of the parameterisation can be derived. We now shift to the data-driven side of the parameterisation spectrum, where instead an algorithm is employed to learn the parameterisation directly from the data, from either observations or a high-resolution model.

It should be noted that there are two types of errors associated with a parameterisation: parameter errors and structural errors. The former concerned with uncertainties of the values used for the parameters, and the latter concerned with uncertainties of the functional form of the parameterisation. Structural errors in models include the numerical errors, the biases, and all the processes not included in the model (Zanna et al., 2018). Data-driven algorithms are being used to address parameter errors (e.g. Schneider et al., 2017a), but we are more focused on using data-driven algorithms to reduce structural errors. We present what these algorithms are and how they learn

from data in Section 3.3, but for now, we review some key examples of data-driven parameterisations from the literature.

This thesis is concerned with ocean eddies, but there have been similar developments in the parameterisation of atmospheric processes. For example, neural networks have been used to parameterise long-wave radiation (Krasnopolsky et al., 2005), moist-convection (Gentine et al., 2018; O’Gorman and Dwyer, 2018), sources of heat and moisture (Brenowitz and Bretherton, 2018; Rasp et al., 2018; Brenowitz and Bretherton, 2019), and precipitation (Larraondo et al., 2019). The goal of these data-driven parameterisations is to either improve the representation of unresolved physical processes in climate models, or to replace existing parameterisations that are computationally expensive. Work has also been conducted to improve the numerical stability of data-driven parameterisations of atmospheric processes when implemented (Rasp, 2019), as well as enforcing physical constraints such as conservation of energy (Beucler et al., 2019a,b).

Now for data-driven parameterisations of oceanic turbulence. The study of Berloff (2005b) constructed a data-driven parameterisation of mesoscale eddies by using a random forcing based on the spatial and temporal correlations within a high-resolution model. In Berloff (2005b), a second-order auto-regressive model is used to capture the temporal variability of the parameterisation, while the horizontal spatial variability is represented by Gaussian function. Similarly, Cooper and Zanna (2015) and Cooper (2017) optimised the lag-covariances of a coarse-resolution model to construct the optimal eddy momentum forcing. The statistical parameterisation of Cooper and Zanna (2015) used a linear stochastic process with no spatial correlation, but a spatially-varying amplitude. The statistical moments of the stochastic potential vorticity parameterisation of Mana and Zanna (2014) were diagnosed in a data-driven manner from a high-resolution simulation.

Data-driven methods have also been used to estimate the diffusivities of mesoscale eddy tracer fluxes, for example, the gauge-invariant statistical optimisation of Mak et al. (2016), or the tracer-based inversion technique of Bachman et al. (2015). These tracer parameterisations are not purely data-driven, as the eddy tracer flux is still assumed to be proportional to the resolved tracer gradient.

Data-driven parameterisations of turbulence that specifically use machine learning appeared in studies of direct numerical simulation (Tracey et al., 2015; Ling et al., 2016a,b), which concern spatial scales of tens of meters, orders of magnitude smaller than the spatial scales of climate models. In particular, Ling et al. (2016b) used a deep neural network to learn an eddy momentum parameterisation of the anisotropic stress tensor using Direct Numerical Simulation (DNS) data (a smaller-scale turbulence study compared to parameterising mesoscale eddies). Interestingly, the study of Ling et al. (2016b) physically-constrained their neural network using the Galilean-invariant tensor basis of Pope (1975). The momentum parameterisation of Ling et al. (2016b) takes the form

$$\hat{\mathbf{S}}_{\mathbf{u}} = \overline{\nabla} \cdot \left(\sum_n g_n \mathbf{T}^{(n)} \right), \quad (2.15)$$

where the $\mathbf{T}^{(n)}$ are the invariant tensors of Pope (1975), and the coefficients g_n are predicted by the deep neural network using only Galilean-invariant inputs. Only after applying these physical constraints did the deep neural network outperform a more simple linear regression model. Another study which used DNS data is that of (Salehipour and Peltier, 2018), which used a deep convolutional neural network to predict the vertical mixing efficiency of both Kelvin-Helmholtz and Holmboe wave instability, which out-performed the other non-data-driven methods such as the Osborn-Cox.

Additional studies have used machine learning algorithms - namely neural networks - to parameterise eddy momentum fluxes in models of freely-decaying 2D turbulence (Maulik and San, 2017; San and Maulik, 2018; Maulik et al., 2019; Cruz et al., 2019) or in large-eddy simulations (Zhou et al., 2019). In the study of Maulik et al. (2019), neural networks were used to parameterise eddy vorticity fluxes, which were then implemented back into the model. An interesting approach was to use the Smagorinsky and Leith eddy viscosity functions of Equations 2.8 and 2.9 as input features to one of the neural networks. They found that using these physically-motivated input features did not improve the predictive skill, but did improve how the neural network behaved once implemented into the model. Studies like these show that incorporating physical principles into data-driven algorithms is important, not only for making parameterisations physically-consistent but also for improving the performance of the parameterisation.

As of yet, there have been no published studies using more modern machine learning techniques, such as deep neural networks, to parameterise *mesoscale* eddy fluxes for use in climate models. As will be shown in this thesis, there are numerous data-driven methods which can, and should, be exploited for this very task.

Chapter 3

Methods

3.1 Numerical ocean models

Here we describe the ocean models used throughout this thesis. Each model numerically solves some form of the Navier-Stokes equations, while making specific assumptions about the nature of the geophysical fluid being simulated. We introduce a single-layered shallow water model, a multi-layered quasi-geostrophic model, and finally barotropic and baroclinic primitive equation models. For each model we briefly discuss the assumptions made to derive the equations of motion, and then present the technical details of the numerical integration and configuration.

3.1.1 Shallow water model

The shallow water equations provide a simple but useful model to study geophysical fluid flows (Vallis, 2017). Despite their simplicity, they capture the main characteristics of double-gyre circulations of the ocean. The shallow water system is characterised by a layer of constant density with a free surface at the top. As the name suggests, the shallow water system has a horizontal length-scale much larger than the vertical length-scale. The vertical distance from the bottom to the free surface is denoted by $h(x, y)$, with mean height H , and the deviation of the free surface height from the mean as $\eta = h - H$.

We obtain the shallow water equations by first assuming hydrostatic balance, $\frac{\partial p}{\partial z} = -\rho_0 g$, for the vertical component of the momentum in Equation 2.1. The density is also assumed constant, such that vertically integrating the hydrostatic balance leads to $p = -\rho_0 g(\eta - z)$. This implies that the horizontal gradient of

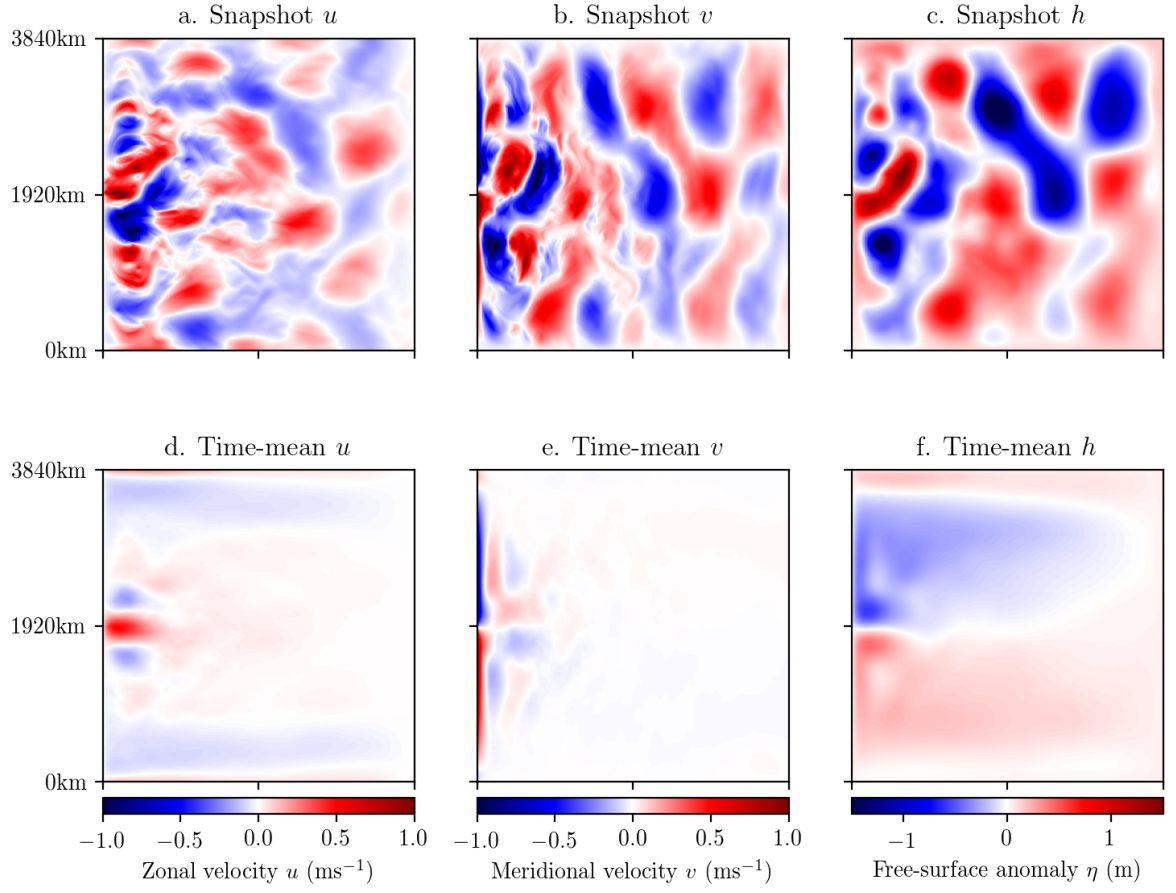


Figure 3.1: Snapshots and time-means of the three prognostic variables of a high-resolution 3.75km shallow water model. Panels (a), (b), and (c) show snapshots of u , v , and h respectively, while Panels (d), (e), and (f) show the time-means over a 10 year period.

pressure does not vary vertically. We also assume that the pressure at the free surface from the overlying fluid is negligible. The horizontal velocity equation can then be written as

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} = -\mathbf{f} \times \mathbf{u} - g \nabla \eta + \nu \nabla^2 \mathbf{u} + \mathbf{F}, \quad (3.1)$$

where the momentum vector $\mathbf{u} = (u, v)$ and gradient operator $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ are now two dimensional. The difference being that the pressure gradient forces can now be written as the gradient of the free surface height multiplied by gravity: $g \nabla \eta$.

The derivation is omitted here, but the mass continuity equation in the shallow water system becomes

$$\frac{\partial h}{\partial t} + \nabla \cdot (h \mathbf{u}) = 0, \quad (3.2)$$

which, due to the constant density, states that a net flux into a column must be balanced by an increase in the free surface height. In the absence of external forcing and dissipation, the sum of the kinetic energy $\frac{1}{2} h \mathbf{u}^2$ and potential energy $\frac{1}{2} g h^2$ densities respects the following conservation equation

$$\frac{\partial E}{\partial t} + \nabla \cdot \mathbf{e} = 0, \quad (3.3)$$

where $e = \frac{1}{2}(h \mathbf{u}^2 + g h^2)$ is the total energy density and $\mathbf{e} = \mathbf{u}(\frac{1}{2} h \mathbf{u}^2 + g h^2)$ is the energy flux. Spatially integrating Equation 3.3, and applying the divergence theorem (Pfeffer, 1986) with no normal flow at the boundaries, implies that the total energy of system is conserved: $\frac{\partial}{\partial t} \int_R E dx dy = 0$.

The numerical integration of the shallow water system is conducted in Python, adapting a model created by Milan Kloewer (the code is publicly available at github.com/milankl/swm). Although Python is computationally slow relative to other programming languages, it is easy to modify and customise. Thus, there are fewer technical obstacles to experimenting with parameterisation schemes within the model. The specific configuration of the shallow water model is as follows:

- The momentum dissipation is a combination of a Laplacian (∇^2) and biharmonic (∇^4) operators, including a quadratic bottom drag term.
- The external forcing is a constant wind forcing, to represent westerly winds at mid-latitudes. We use the square of the sine of latitude to emulate the time-mean jet in the atmosphere.

- We use the beta-plane approximation (Vallis, 2017) for the Coriolis parameter: $f \approx f_0 + \beta y$, with $f_0 = 2\omega \sin(\theta_0)$ and $\beta = \frac{2\omega}{R} \cos(\theta_0)$.
- The equations are solved on an Arakawa C-grid (Arakawa and Lamb, 1977), with an energy and enstrophy conserving advection scheme. The 4th order Runge-Kutta scheme is used for the time integration.

In the above configuration, the three prognostic equations for u , v and h of the shallow water system become

$$\frac{\partial u}{\partial t} + (\mathbf{u} \cdot \nabla)u = fv - g \frac{\partial \eta}{\partial x} + \nu_0 \nabla^2 u + \nu_1 \nabla^4 u - c_D |\mathbf{u}|u + \frac{F_0}{\rho_0 H} \sin^2\left(\frac{2\pi y}{L_y}\right), \quad (3.4a)$$

$$\frac{\partial v}{\partial t} + (\mathbf{u} \cdot \nabla)v = -fu - g \frac{\partial \eta}{\partial y} + \nu_0 \nabla^2 v + \nu_1 \nabla^4 v - c_D |\mathbf{u}|v, \quad (3.4b)$$

$$\frac{\partial h}{\partial t} + \nabla \cdot (h\mathbf{u}) = 0, \quad (3.4c)$$

where $\theta_0 = 35^\circ\text{N}$, $L_y = 3840\text{km}$, $g = 9.81\text{ms}^{-2}$, $H = 500\text{m}$, $\rho_0 = 1000\text{kgm}^{-3}$, $R = 6371\text{km}$, $\omega = \frac{2\pi}{86400}\text{s}^{-1}$, $c_D = 1 \times 10^{-5}$, and $F_0 = 0.1\text{Nm}^{-2}$. The Laplacian viscosity coefficient scales linearly with resolution, such that $\nu_0 = 540\text{m}^2\text{s}^{-1}$ for a 30km grid size, which resolves the Munk boundary layer; the Munk boundary layer is the characteristic horizontal length scale of western boundary currents, when the dominant balance is between frictional terms and the beta-effect (Stommel, 1948; Munk, 1950). The biharmonic coefficient depends on the Laplacian coefficient and also scales with resolution, through $\nu_1 = \nu_0 \frac{\Delta x^3}{\Delta x_0}$, where $\Delta x_0 = 30\text{km}$. The time step is chosen to resolve gravity waves and respect the CFL stability criterion (Lewy et al., 1928), which is a necessary condition for the convergence when numerically solving the equations of motion. Free-slip boundary conditions are used on each wall of the square domain. We spin-up each model for 10 years, before running for an additional 10 years to collect data. The 10 year spin-up allows ample time for the global kinetic energy of the model to equilibrate.

An illustration of the shallow water model, run at 3.75km resolution, is shown in Figure 3.1. The model exhibits western intensification of flow, as seen in the time-mean zonal velocity in Figure 3.1d; the jet does not extend far from the western boundary, typical of barotropic models. Westward propagating wave-like features are also present, particularly at the northern boundary, as seen in the snapshot of the

free-surface height anomaly in Figure 3.1. The shallow water model run at 30km (not shown here) has a similar but less vigorous large-scale flow. We use this shallow water model for testing new eddy momentum parameterisations in Chapter 7.

3.1.2 Quasi-geostrophic model

The quasi-geostrophic system has a single prognostic variable: the quasi-geostrophic potential vorticity q , defined as

$$q = \nabla^2 \psi + \beta y + \frac{\partial}{\partial z} \left(\frac{f_0^2}{N^2} \frac{\partial \psi}{\partial z} \right), \quad (3.5)$$

where the beta-plane approximation $f = f_0 + \beta y$ is used, $N = (-\frac{g}{\rho} \frac{d\rho}{dz})^{\frac{1}{2}}$ is the Brunt-Väisälä frequency, g is gravity, ρ is density, $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ is the horizontal gradient operator, and ψ is the streamfunction for the non-divergent horizontal velocity $\mathbf{u} = (-\frac{\partial \psi}{\partial y}, \frac{\partial \psi}{\partial x})$. For a given value of the potential vorticity q , Equation 2.11 is inverted to calculate the streamfunction ψ , providing a way to calculate the velocity field $\mathbf{u}$.

Quasi-geostrophic systems can have multiple vertical layers, and for each layer the following prognostic equation is solved

$$\frac{\partial q}{\partial t} + (\mathbf{u} \cdot \nabla) q = D + F, \quad (3.6)$$

where D and F are dissipation and external forcing terms respectively. The quasi-geostrophic model is derived by assuming the following: the Rossby number $Ro = \frac{U}{Lf}$ is small $Ro \ll 1$, the length scales of motion are of the same order as the deformation radius $L \sim L_d$, and variations in the Coriolis parameter are small $|\beta y| \ll f_0$.

Despite comprising of a single prognostic variable, the quasi-geostrophic system provides a useful model of large-scale flows in the ocean and atmosphere. Intuitively, a quasi-geostrophic model emulates a fluid that is *almost* in geostrophic balance. If the quasi-geostrophic model is comprised of a single layer, then it can be viewed as a small Rossby number approximation to the previously described shallow water equations.

To numerically integrate the quasi-geostrophic system, we use the PEQUOD model in Fortran, which solves the three-dimensional baroclinic quasi-geostrophic (QG) potential vorticity equation. We assume a constant wind forcing on a beta plane (e.g. Berloff, 2005a). The PEQUOD model numerically integrates the prognostic equation for the PV (Equation 3.6) explicitly with a purely finite-difference

scheme. Specifically, the model uses a centered-leapfrog with RAW filter and a modified Arakawa advective scheme which conserves energy but not enstrophy. The diagnostic equation for the PV (Equation 3.5) requires an inversion, which is an elliptic Helmholtz operator. The Helmholtz solver uses a discrete sine transform such that the inversion can be parallelised (link to PEQUOD page).

The model has a bounded-square domain with a flat bottom. The model has three layers ($m = 1$ upper, $m = 2$ middle, $m = 3$ upper), with thicknesses H_m of 250 m, 750 m, 3000 m, respectively. The dissipation and forcing terms are given explicitly by $D = \nu \nabla^4 \psi - r \nabla^2 \psi \delta_{m,3}$ and $F = (\nabla \times \tau)_z \delta_{m,1} / \rho_0 H_1$; F is the applied wind stress curl forcing, where $\delta_{i,j}$ is the Kronecker delta function. The potential vorticity q_m for a given vertical layer becomes $q_m = \nabla^2 \psi_m + \beta y + \frac{f_0^2}{N^2 \Delta z^2} (\psi_{m+1} - \psi_m)$ (Vallis, 2017), where δz is the distance between the two vertical layers m and $m + 1$.

The horizontal resolution of the model is 7.5 km, such that the model is (approaching) eddy resolving. The first term in the dissipation is a fourth-order term equivalent to a Laplacian momentum viscosity, with viscosity coefficient ν . The second dissipation term parameterises the presence of an Ekman layer with bottom drag coefficient r (and therefore only acts on the bottom $m = 3$ layer). The wind stress forcing applied to the upper $m = 1$ layer is given explicitly by

$$F(x, y) = \begin{cases} -\tau_0 \frac{0.9 \times 2\pi}{L \rho_0 H_1} \sin\left(\frac{\pi y}{g(x)}\right) & y \leq g(x), \\ \tau_0 \frac{2\pi}{0.9 L \rho_0 H_1} \sin\left(\frac{\pi[2y - g(x)]}{L - g(x)}\right) & y > g(x), \end{cases} \quad (3.7)$$

where $g(x) = L/2 + 0.2(x - L/2)$, $L = 3840$ km is the domain length, and ρ_0 is the reference density. The wind forcing profile is chosen to represent westerly winds at mid-latitudes that produce a realistic double-gyre circulation. After the model has been integrated from rest to a statistically steady state, we save 10 years of model output at daily resolution of the turbulent double-gyre circulation. For further details on the QG model, see Mana and Zanna (2014); Zanna et al. (2017), and for a list of the model parameters see Table 5.1.

The configuration of this model leads to two large-scale circulation gyres separated latitudinally by a strong meandering zonal jet. The model is configured to represent an idealised version of current systems such as the Gulf Stream in the North Atlantic or the Kuroshio Extension in the North Pacific; both these current systems exhibit vigorous eddies interacting with a strong mean-flow. The time-mean streamfunction,

which illustrates the double-gyre flow structure, can be seen in Figure 1a of Mana and Zanna (2014).

After we spin-up the model for 100 years at 7.5km resolution, we store 10 years of daily time-slices of all fields. After post-processing, the data is then used to train convolutional neural networks to represent eddy momentum fluxes in Chapter 5.

3.1.3 Primitive equation model

Primitive equation models, first derived by Vilhelm Bjerknes (Friedman, 1993), make fewer assumptions than the shallow water or quasi-geostrophic models, representing a more complex but realistic dynamical system (Smagorinsky, 1963; Vallis, 2017). As such, the primitive equations produce better representations of large-scale oceanic and atmospheric flows. The primitive equations can be derived from the Navier-Stokes by applying the Boussinesq approximation ($|\rho'| \ll \rho_0$) and assuming hydrostatic balance for the vertical component of momentum, which is a valid scaling for shallow fluids. Note, there are primitive equation models that do not assume hydrostatic balance (Marshall et al., 1998) but we do not discuss those here.

Assuming a free-surface of height h , the temperature T being the only active tracer, and a linear equation of state, the corresponding primitive equations are given by

$$\frac{\partial u}{\partial t} + (\mathbf{u} \cdot \nabla)u = fv - \frac{1}{\rho} \frac{\partial p}{\partial x} + D_u + F_x \quad (3.8a)$$

$$\frac{\partial v}{\partial t} + (\mathbf{u} \cdot \nabla)v = -fu - \frac{1}{\rho} \frac{\partial p}{\partial y} + D_v + F_y \quad (3.8b)$$

$$\frac{\partial h}{\partial t} + \nabla \cdot (h\mathbf{u}) = 0, \quad (3.8c)$$

$$\frac{\partial T}{\partial t} + (\mathbf{u} \cdot \nabla)T = D_T, \quad (3.8d)$$

$$p' = g\rho_0\eta + \int_{-z}^0 \rho' dz, \quad (3.8e)$$

$$\rho' = -\alpha_T \rho_0 T' \quad (3.8f)$$

where $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y})$ is the horizontal gradient operator. The terms D_u , D_v , and D_T represent horizontal and vertical dissipation terms for the zonal momentum, meridional momentum and temperature, respectively. The terms F_x and F_y represent external forcing for the zonal and meridional momentum.

We use the MITgcm general circulation model (Marshall et al., 1997) in Fortran to numerically solve Equations 3.8a-f. On an Arakawa C-grid, the MITgcm model uses the semi-implicit pressure method and Adam-Bashforth time-stepping to integrate the equations of motion. The semi-implicit pressure method first integrates the horizontal velocity equations forward in time, to produce a prediction of the horizontal velocities; however, these predicted velocities can be divergent, so an elliptical inversion involving the pressure field is used to correct the velocities to be non-divergent at each time-step. We use MITgcm to run two separate configurations of the model: a barotropic (single-layer) model and baroclinic (multi-layer) model. Data from both models are used in Chapter 6 to construct new eddy momentum and temperature flux parameterisations.

The barotropic and baroclinic models have the following in common:

- A constant zonal wind forcing of the form $F_x = F_0 \sin^4(\frac{2\pi y}{L_y})$, with $F_0 = 0.3 \text{Nm}^{-2}$, where the forcing is only applied to the surface in the baroclinic model (and $F_y = 0$).
- The beta-plane approximation $f \approx f_0 + \beta y$, with $f_0 = 1 \times 10^{-4} \text{s}^{-1}$ and $\beta = 1 \times 10^{-11} \text{m}^{-1} \text{s}^{-1}$ corresponding to a latitude of approximately 35N.
- Free-slip (no tangential stress) boundary conditions on lateral boundaries; no-slip boundary condition on the bottom.
- A time-step of 300s for all prognostic variables, in order to satisfy the advective Courant-Friedrichs-Lewy (CFL) (Courant et al., 1928) condition for numerical stability.

We now highlight the differences between the barotropic and baroclinic models, as well as describing the general features of the large-scale circulation.

Barotropic primitive equation model

The barotropic model has a single layer of depth 500m, and a square domain of length 3840km - the same as the previously described shallow water model. A Laplacian viscosity is used for the momentum dissipation terms $D_u = \nu \nabla^2 u$ and $D_v = \nu \nabla^2 v$, with $\nu = 25 \text{m}^2 \text{s}^{-1}$. A linear bottom drag is applied with coefficient of $c_D = 2 \times 10^{-5}$. We spin-up the model from rest for 10 years, at a spatial resolution of 3.75km.

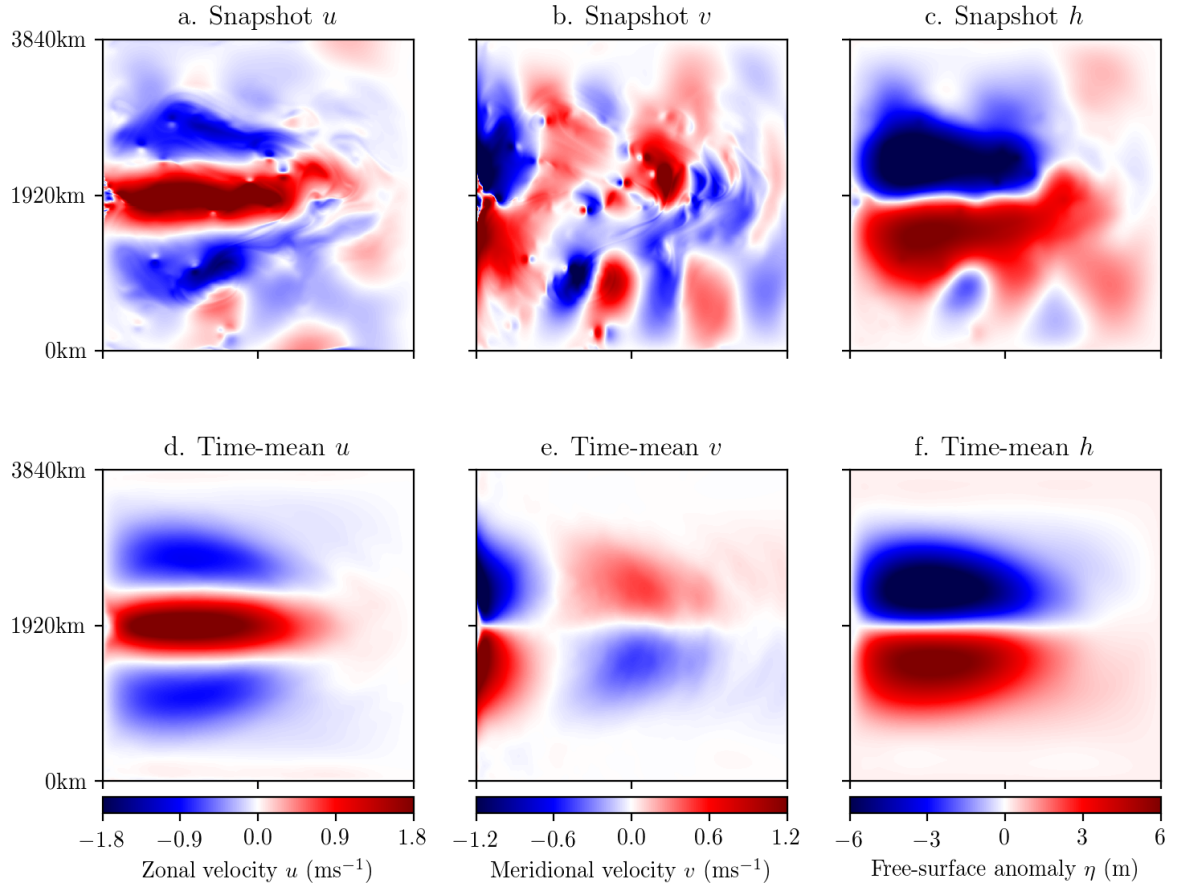


Figure 3.2: Panels (a), (b) and (c) show snapshots of the three prognostic variables u , v , and h respectively, of the high-resolution 3.75km barotropic primitive equation model in MITgcm. Panels (d), (e) and (f) show the time-mean of the three prognostic variables u , v , and h respectively

The resulting flow field of the barotropic model is illustrated in Figure 3.2. The top-row (Figure 3.2a, 3.2b, and 3.2c) shows snapshots of the three prognostic variables u , v , and h respectively. The narrow profile of the $\sin^4()$ wind forcing produces a western boundary current and jet extension that almost reaches the eastern boundary (Figure 3.2d). The double-gyre configuration can be seen clearly in the time-mean free-surface height (Figure 3.2f). In addition to the large-scale flow, eddies of length-scales 10-50km are ubiquitous across the domain (Figure 3.2b). Westward-propagating waves are also present (Figure 3.2c), similar to the shallow water model.

Baroclinic primitive equation model

The baroclinic model comprises of 15 vertical levels, for a total depth of 3600m. The surface layer has a depth of 30m, with the depth of each subsequent vertical layer increasing by 30m, leading to the bottom layer having a depth of 450m. Due to the increased computational cost of running a baroclinic model with multiple layers, we decreased the square domain size from 3840km in length to 1920km, and ran at a spatial resolution of 7.5km.

To emulate and maintain a meridional equator-pole temperature gradient, a restoring function is applied to the surface. The restoring function relaxes the temperature field to a linear profile of 25°C at $y = 0$ km to 5°C at $y = 3840$ km. The restoring time-scale is set to one month.

The lateral dissipation of temperature and horizontal velocities within the equations of motion is represented by a combination of Laplacian (∇^2) and biharmonic (∇^4) operators. The coefficients of these operators are the same for momentum and temperature, with a Laplacian coefficient of $\nu_0 = 100\text{m}^2\text{s}^{-1}$ and biharmonic coefficient of $\nu_1 = 5 \times 10^{10}\text{m}^4\text{s}^{-1}$. The vertical dissipation consists of just a Laplacian operator with coefficient of $1 \times 10^{-3}\text{m}^2\text{s}^{-1}$. Linear bottom drag is applied to the bottom layer with a coefficient of 2×10^{-4} .

We spin-up the baroclinic model for 100 years and then run for a further 10 years for data collection. The resulting large-scale flow is illustrated in Figures 3.3 and 3.4. Snapshots and time-means of the *depth*-averaged u , v , and T are shown in Figure 3.3. The double-gyre configuration is visible in the time-mean depth-averaged-

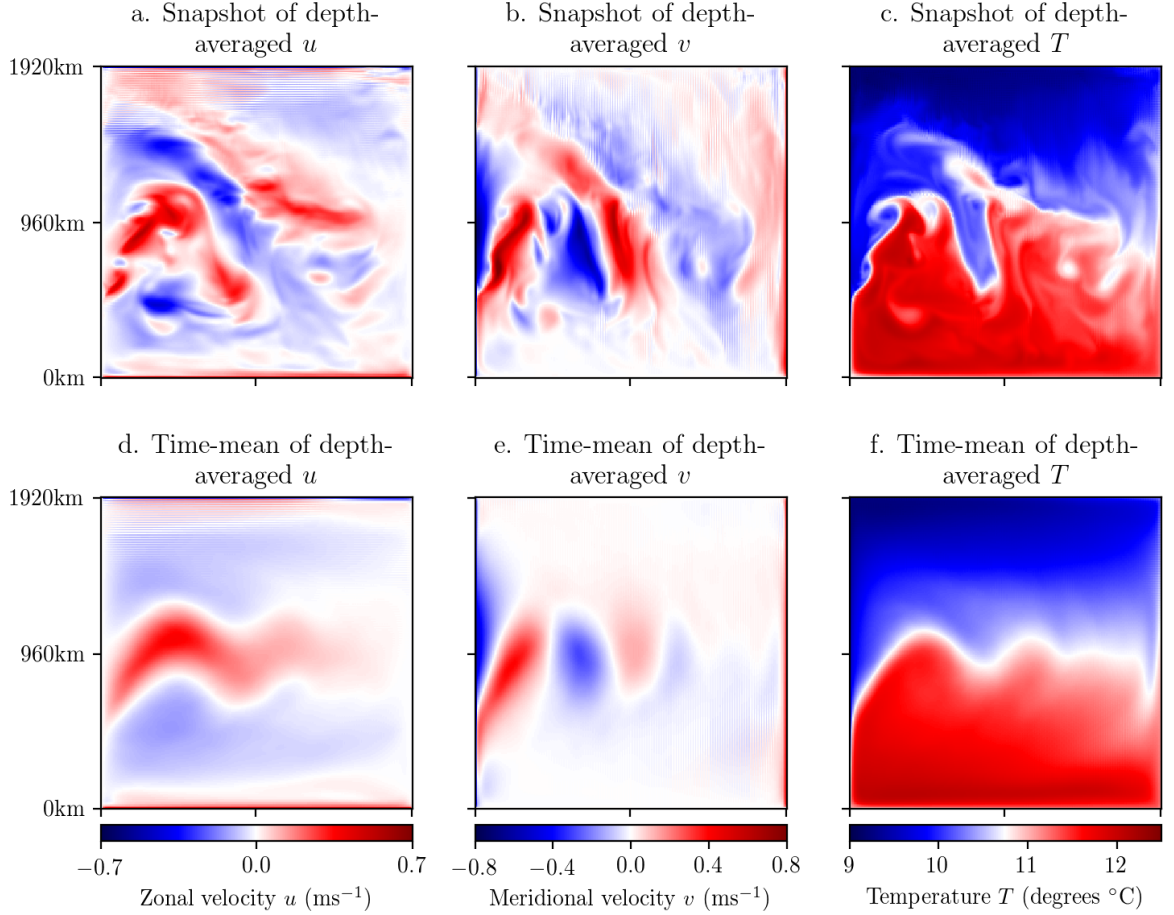


Figure 3.3: Snapshots and time-means of the depth-averaged prognostic variables in the 7.5km resolution baroclinic primitive equation model with 15 vertical layers. Panels (a), (b), and (c) show snapshots of the depth-averaged u , v , and T respectively. Panels (d), (e), and (f) show the time-means of the depth-averaged u , v , and T . The visible numerical grid-scale noise, primarily at the northern boundary, could be removed without significantly increasing the dissipation and over-dampening of the circulation. The time step was also reduced to try and remove the numerical noise to no avail.

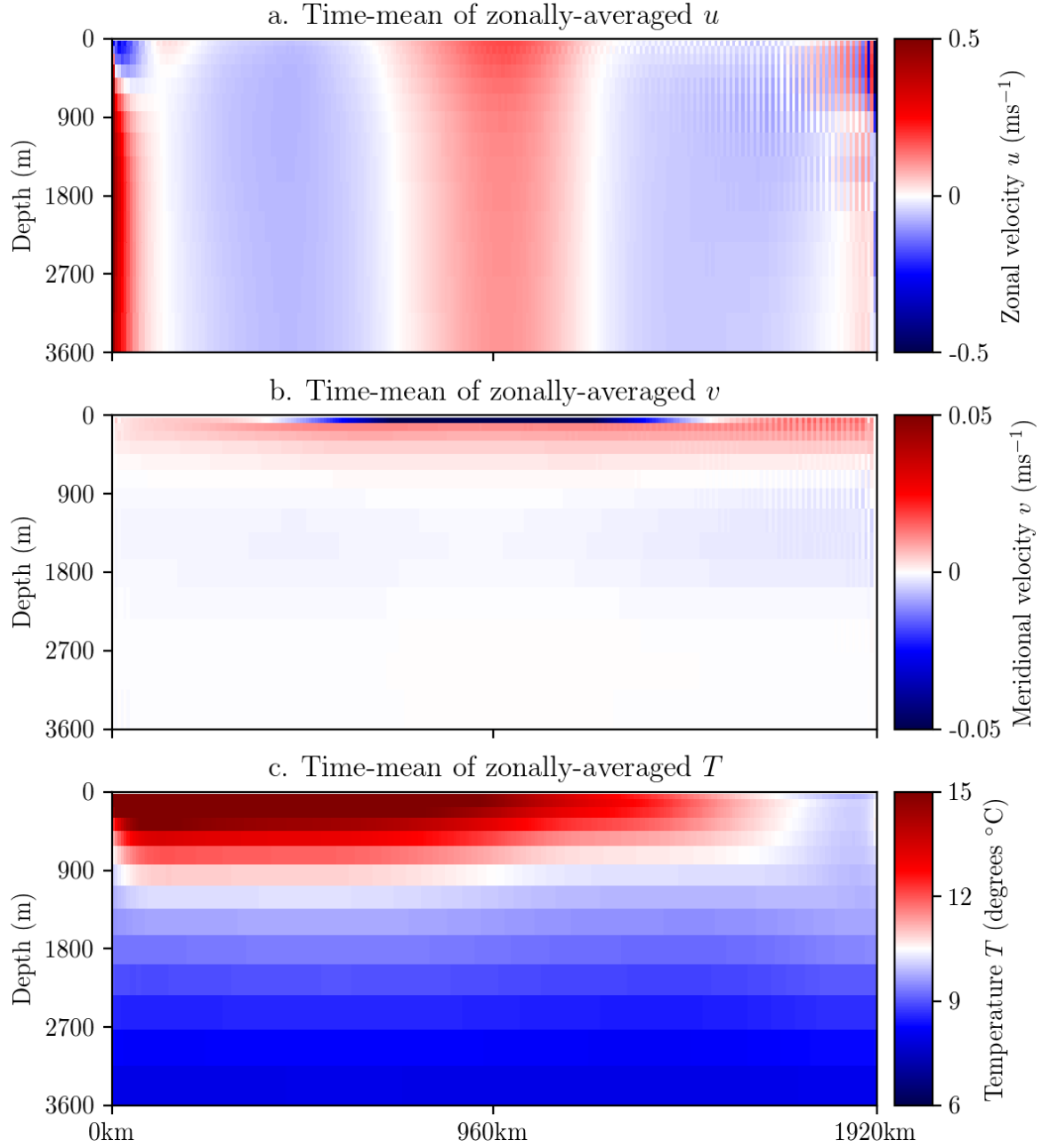


Figure 3.4: Snapshots and time-means of the zonally-averaged prognostic variables in the baroclinic primitive equation model. Panels (a), (b), and (c) show snapshots of the zonally-averaged u , v , and T respectively. Panels (d), (e), and (f) show the time-means of the zonally-averaged u , v , and T .

velocity (Figure 3.3d), with a sharp meridional front in the temperature field (Figure 3.3c and 3.3f).

There is some visible numerical noise at the surface, particularly at the Northern boundary, which can be seen in Figure 3.3a and 3.3b. The noise could be reduced by increasing the horizontal dissipation, but this led to over-damping of the flow. The results of the this model run may therefore be inaccurate and unreliable as a result of this noise.

3.2 Defining the sub-grid effects of eddies

Eddy parameterisations aim to represent the interaction of eddies with the resolved flow, but how do we quantitatively define an ‘eddy’ in the first place? The eddy component of a dynamical field c is typically defined using the deviation from some average, i.e., $c' = c - \bar{c}$, where the quantity $\bar{c}$ can be a spatial or temporal average. The aim of the averaging procedure $\overline{(\cdot)}$ is to reduce the variability - at small spatial scales or high temporal frequencies - from the data, producing a flow field more dominated by the larger scales of motion. The choice of averaging procedure has a significant impact on the nature of the resulting ‘eddy forcing’. Here we discuss the various approaches to defining the eddy component.

Using a simple arithmetic spatial or temporal mean to define the averaged quantity $\bar{c}$ results in a Reynolds decomposition, for which $\overline{c'} = 0$. A temporal Reynolds decomposition assumes that *any* time fluctuation in the quantity c contributes to the eddy component c' , i.e., $\frac{\partial}{\partial t} \bar{c} = 0$. Therefore, a problem with the temporal Reynolds decomposition is that the mean state $\bar{c}$ does not vary in the averaging period, and thus cannot include Lagrangian motion. For example, if the quantity of interest was the zonal momentum u in the Gulf Stream, then the mean state $\bar{u}$ would not incorporate the latitudinal meandering of the large-scale jet. Also, if the time period of averaging is over many years, then the eddy component u' can exhibit large spatial-scales of 500-1000 km, which may not be an appropriate definition of ‘eddy’ for a subgrid process in an ocean model.

A spatial Reynolds decomposition takes the spatial-mean over a subset of grid points within a box, $\bar{c} = \frac{1}{N} \sum_{ij} c_{ij}$, as illustrated in Figure 3.5. This is known as coarse-graining, and the size of the coarse-graining box defines the spatial scale of

2	1	0	0	→	2.6 $\bar{c} = \frac{1}{16} \sum_{ij} c_{ij}$
3	2	1	1		
4	3	3	3		
5	5	4	4		

c_{ij}

Figure 3.5: An illustration of a 4×4 grid box spatial coarse-graining procedure of a quantity c to $\bar{c}$.

the resolved state $\bar{c}$. Coarse-graining produces a more local definition of eddy, where quantities outside of the coarse-graining box have zero effect on the coarse-grained variable $\bar{c}$.

Note that each point within the coarse-graining box can be associated with a weight w_{ij} , to regulate the relative contributions of each point to the averaging. In this case the coarse-graining takes the form $\bar{c} = \frac{1}{N} \sum_{ij} w_{ij} c_{ij}$. The weighted average is necessary in models with varying grid spacing, where area-weighted coarse-graining is needed. However, in this thesis, we only consider the equal-weighting case of $w_{ij} = 1$.

Spatial coarse-graining is relatively simple, and the resolved large-scale quantity $\bar{c}$ can evolve in time. However, one issue with spatial coarse-graining is that the procedure does not commute with spatial derivatives, i.e., $\frac{\partial}{\partial x} \bar{c} \neq \overline{\frac{\partial c}{\partial x}}$. If applied to the momentum field, geostrophic balance for the original quantities u and v may not be satisfied in the coarse-grained fields $\bar{u}$ and $\bar{v}$. In other words, the geostrophic relation $\mathbf{u}_g = \frac{1}{\rho f} \mathbf{k} \times \nabla p$, where $\mathbf{k}$ is a unit vector in the vertical and p is the pressure, does not necessarily hold for the coarse-grained fields, i.e., $\bar{\mathbf{u}}_g \neq \frac{1}{\rho f} \mathbf{k} \times \nabla \bar{p}$.

An alternative to coarse-graining is to use a low-pass spatial filter,

$$\bar{c}(x_0, y_0) = \iint_R c(x, y) g(x - x_0, y - y_0) dx dy, \quad (3.9)$$

where $k(x, y)$ is the filter and the two-dimensional spatial integral is evaluated over the entire domain. Spatial filters, temporal filters, or both simultaneously, are used extensively in Large Eddy Simulation (LES) (Smagorinsky, 1963; Pope, 2001; Sagaut, 2006), where the smallest wavenumbers (highest frequencies) of systems are ignored, i.e., truncated, to avoid the computational cost of Direct Numerical Simulation (DNS) of the full Navier-Stokes equations.

Spatial filters do commute with spatial derivatives, preserving relations such as geostrophy in the subgrid and resolved quantities. If the filter g is normalised such that the global spatial integral evaluates to one, then the filter globally conserves c . In other words, $\int \bar{c} dx dy = \int c dx dy$, if $\int g(x, y) dx dy = 1$. In the case of a tracer like salinity, the total salinity in the system remains unchanged after the filtering, preserving the global integral.

Therefore, the filtering process redistributes the quantity c , rather than producing sources and sinks of c . If the filter is a low-pass filter, i.e., a filter that removes high wavenumbers from a signal, the process will smooth the field c ; the smoothing reduces variability at spatial-scales smaller than the pre-defined scale characterising the filter. Spatial filtering produces a more non-local definition of eddy, as all grid points, irrespective of location, impact the large-scale field $\bar{c}$ through the integral $\bar{c}(x_0, y_0) = \iint_R c(x, y) g(x - x_0, y - y_0) dx dy$, whereas, in spatial coarse-graining, it is only the grid points within the coarse-graining box with impact the value of $\bar{c}$.

As the spatial-scales of ocean dynamics are linked to the temporal-scales, then applying a spatial filter indirectly applies a temporal filtering. For example, applying a low-pass filter to extract large-scale features of the momentum field u will automatically select longer time-scales within the data. Thus, spatial and temporal filtering are not distinct and will lead to similar definitions of eddies. We will only consider spatial filtering and coarse-graining from hereon.

Spatial filters, unlike coarse-graining, do not change the underlying grid size. For example, spatially filtering a field on a 7.5 km grid results in a new field on a 7.5 km grid. Whereas, a 4×4 coarse-graining on a 7.5 km grid results in a new field on a 30 km grid. For spatial filtering, this can cause issues when attempting to implement a parameterisation back into a coarse-resolution model. For example, a parameterisation diagnosed on a 7.5 km grid using spatial filtering may not work on a 30 km grid.

A solution is to combine spatial filtering and coarse-graining. For example, define the mean $\bar{c}$ as spatially filtering and then coarse-graining to a new grid size. The initial spatial filtering will remove information at small scales, producing a non-local definition of eddy, and then the coarse-graining changes the grid size to a more coarse

grid. We now have three possible definitions of mean and deviation to define the ‘eddy’ field with:

- **Coarse-grain:** area-weighted arithmetic mean over a sub-region. Produces local definition of eddy. Changes grid size.
- **Spatial filter:** convolve field with loss-pass filter. Commutes with spatial derivatives. More non-local definition of eddy. Grid size unchanged after filtering.
- **Spatial filter and coarse-grain:** convolve field with loss-pass filter, and then coarse-grain. Quasi-local definition of eddy which does change grid size.

For any of the above definitions, how do we go on to define the eddy forcing related to some quantity c ? First, assume the following prognostic equation for the quantity c

$$\frac{\partial c}{\partial t} + (\mathbf{u} \cdot \nabla)c = F + D, \quad (3.10)$$

where F and D represent all forcing and dissipation terms. Now average the above equation, leading to

$$\frac{\partial \bar{c}}{\partial t} + \overline{(\mathbf{u} \cdot \nabla)c} = \bar{F} + \bar{D}, \quad (3.11)$$

where $\overline{\frac{\partial c}{\partial t}} = \frac{\partial \bar{c}}{\partial t}$, as we are only considering spatial averages. Now add $(\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{c}$ to each side of Equations 3.11 and rearrange to give

$$\frac{\partial \bar{c}}{\partial t} + (\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{c} = \bar{F} + \bar{D} + S_c. \quad (3.12)$$

We now have a prognostic equation for the coarse quantity $\bar{c}$, which differs from the original prognostic equation for c by the term S_c , given by

$$S_c = (\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{c} - \overline{(\mathbf{u} \cdot \nabla)c}, \quad (3.13)$$

which we will refer to as the eddy forcing associated with the quantity c . Note that, for a non-divergent flow ($\nabla \cdot \mathbf{u} = 0$) and Reynolds average ($\overline{c'} = 0$), we recover the divergence of the eddy flux of c : $S_c = -\bar{\nabla} \cdot (\overline{\mathbf{u}'c'})$.

What does S_c actually look like? How do the different averaging procedures impact S_c ? As an illustrative example, consider the zonal momentum u as the quantity of

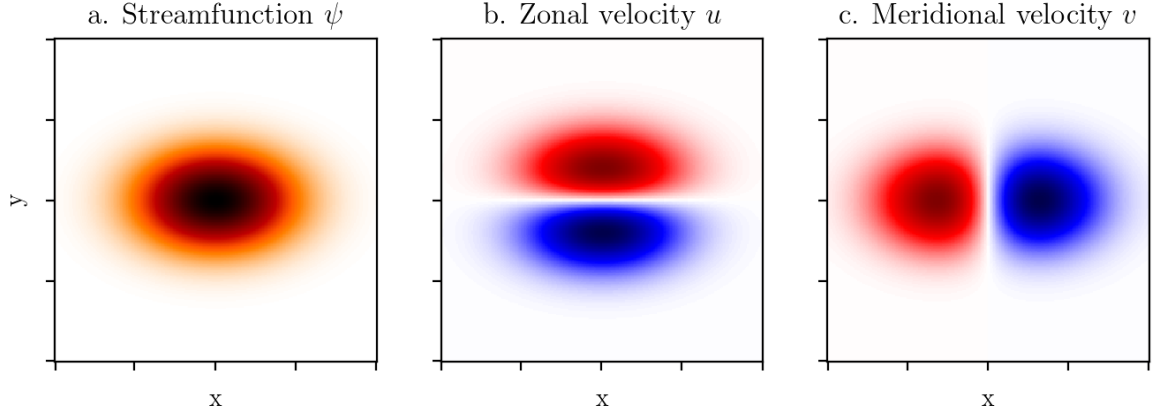


Figure 3.6: A synthetically generated streamfunction, Panel (a), based on a elliptical Gaussian function. The resulting velocities $u = -\frac{\partial\psi}{\partial y}$ and $v = \frac{\partial\psi}{\partial x}$ are shown in Panel (b) and Panel (c) respectively.

interest, and assume a Gaussian ellipse streamfunction from which to calculate u . The streamfunction $\psi \propto e^{-(ax^2+by^2)}$, and the velocities $u = -\frac{\partial\psi}{\partial y}$ and $v = \frac{\partial\psi}{\partial x}$, are shown in Figure 3.6. The Gaussian ellipse is used to emulate a coherent vortex, leading to dipole patterns for each component of the velocity. The differences caused by each eddy definition will depend on the resolution of the underlying high-resolution field, the length-scale of the prescribed flow field, and the length-scale of the averaging procedures. However, we can use this toy model to examine the more qualitative differences between the eddy momentum forcing fields from each eddy definition.

The zonal eddy momentum forcing is given by $S_x = (\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{u} - \overline{(\mathbf{u} \cdot \nabla)u}$. Figure 3.7 shows S_x calculated using the three different averaging procedures: (a) coarse-graining, (b) spatial filtering and (c) both. A radially-symmetric normalised-Gaussian function is used as the low-pass filter, and the width of the Gaussian filter is set equal to the size of the coarse-graining sub-region, to fairly compare each averaging procedure.

The averaged zonal momentum fields $\bar{u}$ (top row of Figure 3.7) appear almost identical. However, the resulting eddy momentum forcings S_x show significant differences between the averaging procedures. In Panel (b), the spatial filtering results in a smooth dipole pattern in S_x , whereas the S_x from coarse-graining exhibits more small-scale variability in the meridional direction. The S_x using both a spatial filter and coarse-graining, Panel (c), produces a eddy forcing pattern similar to the spatial filtering but without the small-scale variability of the coarse-graining in Panel (c).

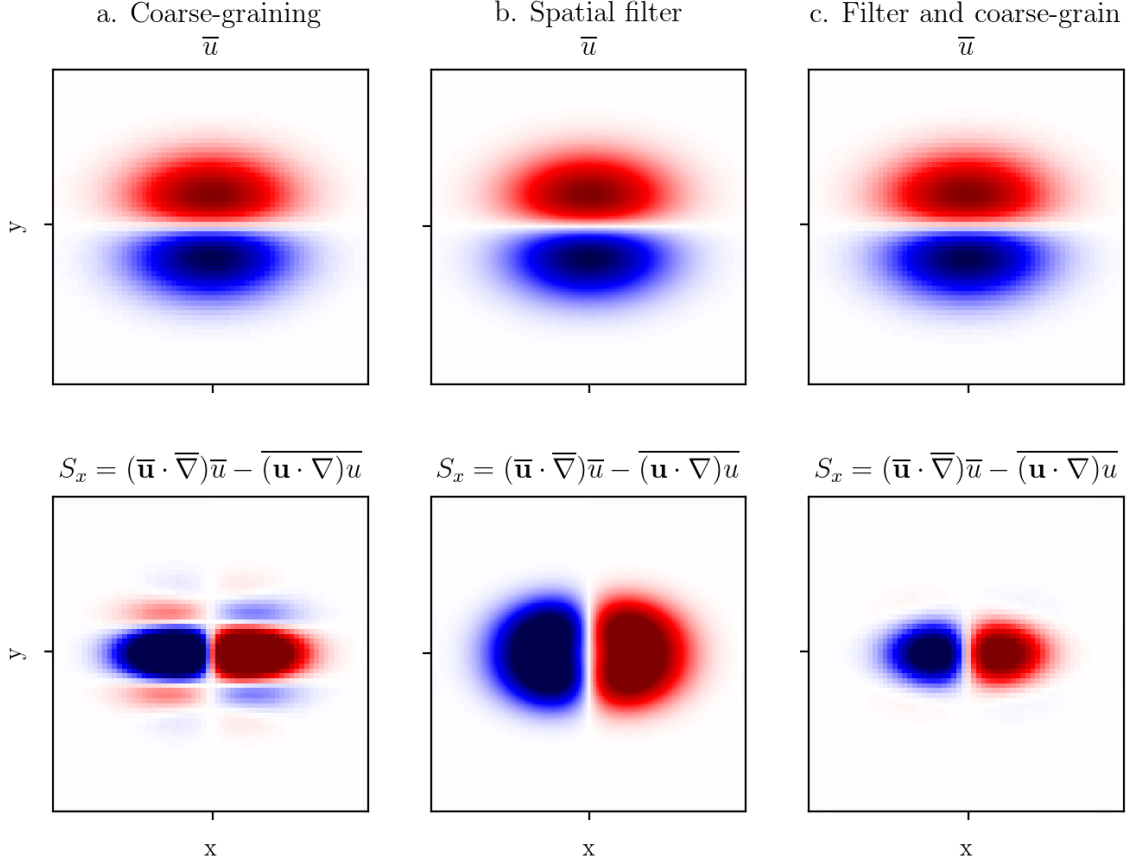


Figure 3.7: Illustrations of three averaging procedures: Panel (a) shows coarse-graining, Panel (b) for a spatial filter, and Panel (c) shows spatial filtering and then coarse-graining. The top row shows the averaged zonal velocity $\bar{u}$ and the bottom row shows the zonal component of the resulting eddy momentum forcing S_x .

Note that the combined approach in Panel (c) changes the grid size in the same way as Panel (a).

Therefore, spatial filtering produces a smoother forcing field compared to coarse-graining, but does not change the grid size. Spatial filtering is commonly used for defining closures in Large Eddy Simulation (LES), where the Navier-Stokes equations are truncated at small-scales using a low-pass filter. Coarse-graining is commonly used when constructing non-LES parameterisations (e.g. Mana and Zanna, 2014), but the resulting forcing fields can become noisy at small-scales. An approach which combines filtering and coarse-graining changes the grid size, while not exhibiting producing the small-scale noise. No approach is perfect, but each approach can be useful as long as the effects of the averaging procedure are acknowledged and understood.

3.3 Supervised machine learning for regression

Machine learning approaches to scientific problems are becoming commonplace (Rich, 2019). The popularity of machine learning can be attributed to the following factors: the increase in computing power available to researchers (Schaller, 1997; Waldrop, 2016), the availability of easy-to-use implementations in languages such as Python (e.g. Pedregosa et al., 2011), and the recent advent of deep learning (Goodfellow et al., 2016) - we will introduce deep learning in Section 3.3.1. Despite the sometimes excessive attention placed on machine learning, there has been substantial progress in data-driven algorithms in recent years (e.g. Krizhevsky et al., 2012), which should certainly be taken advantage of by the weather and climate community. Here, we provide an introduction to the general concept of machine learning, and then describe the specific supervised regression algorithms used in this thesis.

The two main types of machine learning algorithms (ignoring reinforcement learning) are supervised and unsupervised algorithms. Unsupervised learning is when there are features x , and the aim is to discover some underlying structure in the data, i.e., $f(x)$. Clustering (e.g. K-means) and dimensionality reduction (e.g. principle component analysis) are commonly used unsupervised algorithms.

Supervised learning on the other hand predicts a target variable y using the features x . Supervised learning, therefore, aims to construct a function from the data that accurately maps from x to y . Note, this is the deterministic case - there is also a probabilistic definition of supervised learning where the aim is to learn the conditional probability distribution function $p(y|x)$. The form of the function, $f(x)$, is determined by the choice of algorithm, and we denote predictions of the function as $\hat{y} = f(x)$. Each function $f(x)$ will have a set of parameters, $\mathbf{w} = (w_1, w_2, \dots)$, which impact the predictions, $\hat{y}$, of the algorithm. It is the parameters $\mathbf{w}$ of $f(x)$ that are learnt from data, and the learning involves modifying the values of $\mathbf{w}$ to produce the most accurate predictions of the target y . Therefore, the dependence of the predictions on the parameters should be technically be included as: $\hat{y} = f(x, \mathbf{w})$.

The learning of the parameters $\mathbf{w}$ is carried by an optimisation procedure, involving the minimisation of a loss function L . The form of the loss function will vary substantially between classification or regression tasks. Classification is concerned with predicting discrete categories, such as image detection in computer vision tasks.

For example, imagine a classification model attempting to determine the binary outcome of whether an image contains a human face. If the image does in fact contain a human face, then predicting probabilities close to one should yield a low loss value. This is exactly how the cross entropy loss behaves, which quantifies the difference between the predicted probability distribution and the true distribution.

In this thesis however, we only use regression methods, which predict target variables which take on continuous values. The mean-square error (MSE) is a commonly used regression loss function, defined by

$$L \propto \sum_i^N (y_i - \hat{y}_i)^2, \quad (3.14)$$

where the sum is over all N samples in the training data, with a sample being a single instance of the target y_i and features x_i . The loss function quantifies the difference between the predictions and the true target variables, and the choice of loss function is just as important as the choice of the functional form of $f(x, \mathbf{w})$. From a probabilistic perspective, the specific choice of loss function represents how the errors are assumed to be distributed.

There are a plethora of supervised regression models to choose from, ranging in complexity from linear regressions, which model the target as a linear combination of the inputs; to random forests, which average predictions from a multitude of decision trees, with each decision tree using a random subset of inputs. Model selection can be challenging, and there are methods of automatic model selection, normally involving information criterion. But every prediction problem will require a different model, even if the differences between the model are minimal. This statement is more formally known as the “no free lunch theorem” (Wolpert et al., 1997), which states that no single optimisation algorithm can perform better than blind search/random selection, when the performance is averaged across all possible problems.

However, despite the no free lunch theorem, one particular form of algorithm has stood out in recent years: neural networks. Neural networks can be thought of (loosely) as representing the firing of neurons in the brain, and indeed the original perceptron algorithm (Rosenblatt, 1957) was based on modelling a single neuron firing. An illustrative example of a simple neural network, shown in Figure 3.8, forms predictions in the following way: features x_i are introduced to the network through

the input layer, with a node for each feature. The input values x_i are then passed to the hidden layer (there may be more than one hidden layer) with weights w_{ij}^0 , with the j^{th} hidden layer node z_j taking the following value

$$z_j = \Phi\left(\sum_i^4 w_{ij}^0 x_i\right), \quad (3.15)$$

where w_{ij}^0 is referred to as a weight of the connection between the input x_i and the hidden node z_j , with w_{ij} being learnt from the data. The function $\Phi()$ is called the activation function, and maps the real-valued sum of inputs multiplied by the connection weights $\sum_i w_{ij} x_i$ (which can be positive or negative), to values between zero and one. The activation function was originally introduced to mimic a neuron which, depending on whether it receives enough of a signal from neighbouring neurons, will turn on and ‘fire’ or ‘activate’; hence, if the activation function produces a value close to one, that loosely resembles a single neuron firing.

Currently, the activation function is used to introduce non-linearity into the network and there are many to choose from depending on the task at hand. One of the most widely-known activation functions is the ‘rectified linear unit’ (ReLU), given by

$$\Phi(z) = \max(0, z), \quad (3.16)$$

where in the original perceptron model (Rosenblatt, 1957), a negative value of z leads to a value of zero, whereas as positive value of z ‘activates’ the ‘neuron’. In recent years, the *Leaky* ReLU has proved one the most effective (Maas et al., 2013).

Each hidden node value z_j is referred to as the activation value. In the case of a single hidden layer, as illustrated in Figure 3.8, the final output variable $\hat{y}$ is the sum of the activation values of the hidden layer (and also assuming no activation function for the final prediction). In other words, $\hat{y} = \sum_j w_j^1 z_j$, where w_j^1 are the weights of the connections between the hidden layer nodes and the output variable. The full function mapping from x to y of the neural network in Figure 3.8 can then be written as

$$\hat{y} = \sum_j^5 w_j^1 \Phi\left(\sum_i^4 w_{ij}^0 x_i\right),$$

which is already quite complex for what is a simple neural network architecture. The process of the inputs x_i being transformed by the hidden layers to the final output

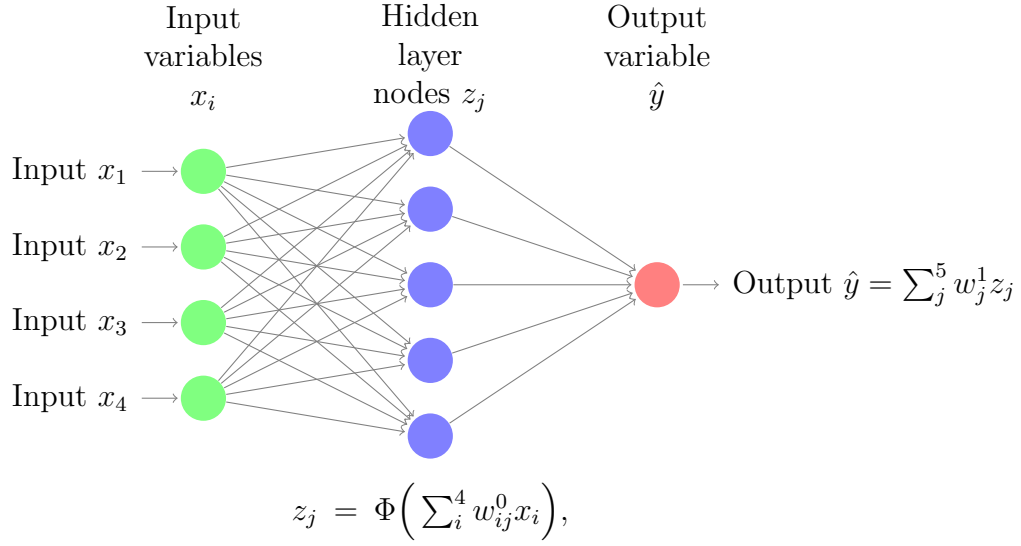


Figure 3.8: An illustration of a simple feed forward neural network with four input variables x_i , a single hidden layer of five nodes z_j , leading to a single output node for the predicted target variable $\hat{y}$.

variable $\hat{y}$, is known as a *feed-forward* neural network, as the values of each layer are ‘fed’ to the next layer. Note that if we remove the hidden layer, and retain only an input and output layer, our function reduces to

$$\hat{y} = \sum_i^4 w_i x_i,$$

which is plain linear regression onto four input variables. Therefore, multiple linear regression can be thought of as a special type of neural network with no hidden layers and no activation function on the output layer.

We have described how a feed-forward neural network makes predictions, but how are the weights of the connections w_{ij}^0 and w_j^1 learnt from data? We previously stated that a loss function L is minimised, with the MSE loss of Equation 3.14 being a common choice for supervised regression tasks. How is the minimisation achieved in the case of a neural network? For regression algorithms such as linear regression, the minimisation of the MSE loss can be achieved exactly with a closed-form formula. However, even simple neural network architectures, leading to functions like that of Equation 3.17, do not have a closed-form solution for minimising the MSE loss.

We need an optimisation algorithm which numerically finds the values of the parameters w_{ij}^0 and w_j^1 that minimise the loss function. Optimising the weights of neural

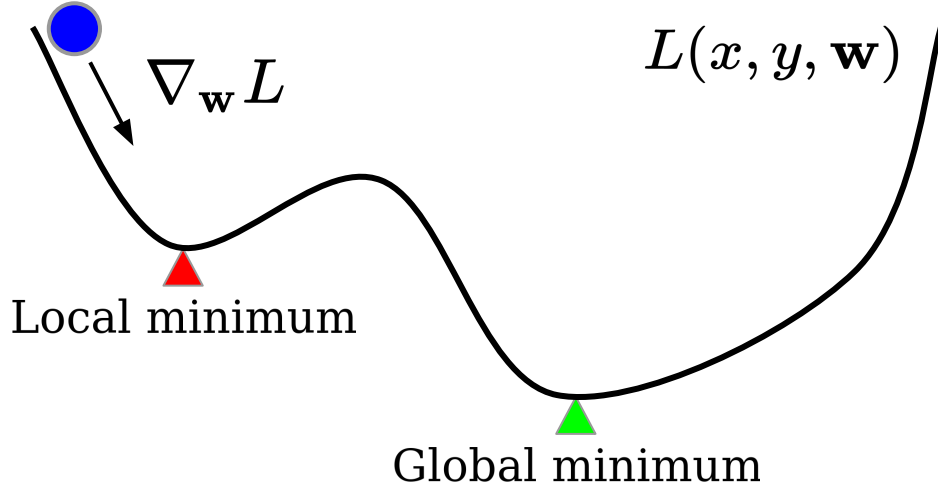


Figure 3.9: An illustration of local versus global minima in non-convex optimisation. Gradient descent aims to iteratively find the minimum of a function by considering the gradient of that function with respect to the parameters.

networks is part of the more general field of non-convex optimisation. The defining feature of non-convex optimisation is that a local minimum does not necessarily correspond to a global minimum. This property makes non-convex optimisation problems difficult, as the optimisation algorithm must navigate the parameter space without prematurely stopping at a local minimum. In the case of the neural network shown in Figure 3.8, prematurely stopping in a local minimum of the loss function will lead to weight values w_{ij}^0 and w_j^1 that do not predict the target value y as well as they should. Improved weight values, and therefore higher predictive skill, can be found if the optimisation algorithm can avoid getting stuck in local minima.

This is where gradient descent comes in. Gradient descent is an algorithm for finding a minimum of a loss function L by examining the gradient of that function with respect to the parameter values. Assuming all the parameters of the function - w_{ij}^0 and w_j^1 in the case of the above neural network - are represented as a single one-dimensional vector $\mathbf{w}$, then the gradient descent algorithm is defined as

$$\mathbf{w}_{n+1} = \mathbf{w}_n - \eta \nabla_{\mathbf{w}} L(x, y, \mathbf{w}), \quad (3.17)$$

where η is the learning rate and n is the iteration number of the algorithm. The gradient of the function with respect to the parameter values is averaged over all

samples in the training data,

$$\nabla_{\mathbf{w}}L(x, y, \mathbf{w}) = \frac{1}{N} \sum_i^N \frac{\partial L(x_i, y_i, \mathbf{w})}{\partial \mathbf{w}}, \quad (3.18)$$

where N is the number of samples in the training data. In the case of a neural network, the hidden layers are a sequence of individual functions, meaning that the chain-rule can be used to evaluate the partial derivatives of the loss function. The weights of the output layer are updated first, according to Equation 3.17, and then subsequent layers until the input layer is reached. This is why the chain-rule within gradient descent is referred to as back-propagation: the input is fed *forward* through the network to predict the target variable, but the error (quantified by the loss function) is propagated *backwards* through the network, updating the weights along the way.

Gradient descent defined by Equation 3.17 can be thought of (approximately) as a ball rolling down a hill under gravity, with the ball rolling in the direction where the hill is steepest. If this rolling ball analogy is accurate, then the ball can get stuck in a small dip before reaching the bottom of the larger hill. How does gradient descent deal with the difficulties of getting stuck in a small dip, i.e., the problem of non-convex optimisation, illustrated in Figure 3.9?

This is where *stochastic* gradient descent (SGD) comes in. Assuming the loss is accurately evaluated by samples in the training data, then $\nabla_{\mathbf{w}}L(x, y, \mathbf{w}) = 0$ in a local minimum. However, instead of using all samples to evaluate the gradient of the loss function, SGD uses a random subset of the training data, i.e.

$$\nabla_{\mathbf{w}}L(x, y, \mathbf{w})_{\text{SGD}} = \frac{1}{M} \sum_j^{M < N} \frac{\partial L(x_j, y_j, \mathbf{w})}{\partial \mathbf{w}}, \quad (3.19)$$

where the reduced number of samples M is referred to as the batch-size of the training process. Stochastic approximation of the gradient of the loss function allows SGD to more effectively navigate the parameter space, leading to better convergence (i.e. more optimal minima) for many machine learning algorithms. SGD also reduces the computational cost of evaluating the gradient of the loss function, as not all training samples need loading into memory and less floating point operations are required per application of the back-propagation algorithm. There are many variants of SGD descent now in use, with the Adam algorithm (Kingma and Ba, 2014) proving effective at optimising neural networks across a wide range of tasks.

Note that algorithms such as gradient descent optimise for parameters of the neural network, after the architecture of the neural network has already been specified. The number of hidden layers, the number of nodes in each hidden layer, the activation functions, etc, are all chosen before training but nonetheless impact performance. These are known as the hyperparameters, as they are not directly optimised by gradient descent. However, the hyperparameters of any neural network must be investigated for their impact on the final prediction, and can in fact be optimised using other packages, e.g., Keras.

Many of the general concepts mentioned so far - mapping features x to targets y , loss functions, optimisers - apply to not only neural networks but to almost any supervised regression algorithm. These techniques have been known for decades; a form of stochastic gradient descent has been known about since at least the 1950's (Robbins and Monro, 1951). However, neural networks involving many hidden layers, grouped under the general term 'deep learning', have arisen only recently. It was not only the depth of these neural networks which proved effective, but some consisted of entirely new types of architectures.

3.3.1 Data-driven prediction: convolutional neural networks

Deep learning is a broad term describing a subset of machine learning methods based on neural networks. The general defining feature of deep learning - and hence what makes it distinct from neural networks of previous decades - is the large architecture, specifically the increased number of layers forming the network. Only in recent years has computational power reached the levels required to train large neural networks, and also manage the gigabytes or petabytes of data needed to perform said training. Examples of some deep learning architectures are deep fully-connected networks, deep belief networks, recurrent neural networks, and convolutional neural networks - for more details, see Goodfellow et al. (2016).

One type of deep neural network, convolutional neural networks, began to dominate the field of two-dimensional computer vision tasks. ImageNet (Deng et al., 2009) is a database of images and is used as a testing ground for image classification algorithms. The goal is to accurately determine whether or not an image contains a particular object, with 1000 different classes and millions of images. The now famous

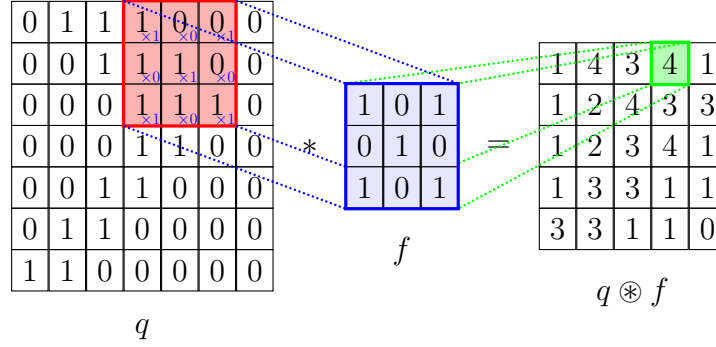


Figure 3.10: An illustration of a convolution operation of a 7×7 matrix q with a 3×3 filter f , showing the final 5×5 result $q \otimes f$.

paper by Krizhevsky et al. (2012) used a convolutional neural network called AlexNet, which achieved a state of the art error rate of 15.3%. To put this achievement in perspective, at the time the next best was achieving an error rate of 26.1%. The machine learning community was well aware of convolutional neural networks by 2012, but the paper of Krizhevsky et al. (2012) began drawing attention from the wider scientific community. Ever since, convolutional neural networks have become the go-to deep learning algorithm for almost any task involving images.

So what, specifically, is a convolutional neural network? They are defined by convolution layers, that aim to take advantage of the spatial covariance of structured data. Spatial correlations allow the sharing of parameters at different locations of a field, which leads to the concept of a learnt convolution operation.

All the simple neural networks discussed previously contained fully-connected layers, which connect every input node with every hidden node, with each connection having a weight to be learnt from the data. Convolution layers are different: they take a 2D input field which is convolved with a filter (also known as a kernel), where the values of the filter are the same at each location. A discrete convolution of a two-dimensional field q with a filter f is defined as

$$(q \otimes f)(x, y) := \sum_{n=-\infty}^{\infty} \sum_{m=-\infty}^{\infty} q(n, m) f(x - n, y - m). \quad (3.20)$$

If the filter f is an Gaussian function $f \propto e^{-(\mathbf{x}-\mathbf{x}_0)^2/2\sigma}$, then the convolution operation becomes equivalent to a low-pass Gaussian filter, as used in Section 5.2.1 to smooth fields and remove information at small scales. However, for a convolution layer within

a neural network, the weights of the filter f are parameters to be *learnt from data*. An illustrative convolution operation is shown in Figure 3.10, where a 3×3 filter f is convolved with a 7×7 matrix q .

The purpose of the convolution layer is to extract information from the input field. For a neural network with many convolution layers trained for facial recognition, the initial convolution layer may extract local features from the input such as edges. The next convolution layer may then build on those simple features by constructing shapes such as circles. Subsequent convolution layers will extract features of increasing complexity, integrating the knowledge of the previously extracted local features. The extracted features will tend toward the abstract notion of a face, until a final prediction can be made as to whether the original image did in fact contain a face.

This is the key concept of convolutional neural networks: the filters of each convolution layer are optimised to extract the most important information for the task at hand. For instance, convolutional neural networks trained to predict blood pressure from retinal images will extract information from the blood vessels of the eye (Poplin et al., 2018). Note that predicting blood pressure is a regression task, as convolutional neural networks are not limited to classification.

Each convolution layer has multiple filters - the exact number of filters of each layer is a hyperparameter to be specified prior to training. This allows multiple pieces of potentially separate information to be extracted. The size of the filters is also a hyperparameter, and determines how large an area each filter can ‘see’. Filter sizes such as 2×2 or 3×3 will only extract very local features, whereas increasingly large filters will capture information over a larger area of the field. After the convolution has finished, each value is then passed through an activation function, the same as in fully-connected layers.

The power of convolutional neural networks comes from the fact that the filters are learnt from the data. This allows the convolution operation to efficiently extract the most important information from the input. A disadvantage of convolution layers is the computational cost of forming a prediction during implementation, relative to common physics-driven parameterisations (or even some simpler data-driven algorithms). The computational cost of complex using data-driven algorithms must be

considered when attempting to parameterise eddies in coarse-resolution models, as the cost should not approach that of running a higher resolution model.

3.3.2 Data-driven inference: closed-form equation discovery

The previously described convolutional neural networks are useful when predictive skill is valued above other factors such as interpretability. In general, it is difficult to understand how deep learning methods transform an input into the target variable. Yes, we can talk broadly about how convolution layers automate feature extraction, but identifying exactly what features are being extracted by the many learnt filters is cumbersome. Interpretability is particularly hard to extract in large networks with $10^5 - 10^6$ parameters.

It is worth distinguishing between prediction and inference at this point. A regression task is a prediction task if getting the right answer, or producing the lowest error rate possible, is the primary aim. For example, when using a data-driven algorithm to diagnose potentially life threatening diseases, this is a prediction task where lowering the error rate is of the utmost importance.

Inference is more concerned with revealing knowledge. Just like with prediction tasks, a data-driven algorithm is used to map from the inputs to the target variable, but with inference the emphasis is shifted towards understanding how the inputs relate to the target variable. This is why interpretability of the data-driven algorithm is key for inference; 99% accuracy is not useful to inference if the algorithm cannot be deciphered and understood.

An example of inference in medical diagnosis would be to identify the most important factors leading to an accurate diagnosis. For inference, the regression algorithm still needs some level of predictive skill - an algorithm that captures 0% of the variance is useless to both prediction and inference - but the specific accuracy rate is not as important. For example, a 5% improvement in diagnosis accuracy is substantial for a purely predictive task, but it may not be useful for inference.

In this section we introduce a particular inference method for discovering closed-form equations from data. Given some spatio-temporal dataset, this inference method aims to uncover an equation that could have produced that dataset. Algorithms for the data-driven discovery of equations typically begin by constructing a library of

basis functions, which can comprise of any function such as products or derivatives of relevant quantities. A sparse linear regression is applied iteratively to the library of - linear and non-linear - basis functions, which prunes/discards functions which are deemed unimportant according to some condition. If successful, the end result is a handful of basis functions that form the final equation, while capturing a significant proportion of the variance of the dataset.

The aim of a sparse linear regression is to model the target as a linear function of the inputs, while being strict in choosing which inputs are used, i.e., feature selection. Mathematically, a sparse linear regression causes regression coefficients of unimportant inputs to become exactly, or at least very close to, zero - hence the name ‘sparse’ regression. Sparseness is achieved by adding a term to the loss function, which penalises unnecessarily large regression coefficients in some way. The additional penalty term in the loss function is known as regularisation and reduces over-fitting. The penalty term will have a parameter that needs tuning, which controls the strength of the regularisation.

For example, the method of Rudy et al. (2017) uses Ridge regression, which is linear regression with a mean squared error loss, but with an extra penalisation term proportional to the sum of the regression coefficients squared, $\alpha \sum_i w_i^2$, where the sum is over all inputs, and the parameter α controls the strength of the regularisation. A regression with this sort of penalised loss function is known as $L2$ regularisation and is used to reduce overfitting. In the method of Rudy et al. (2017), the library function is pruned iteratively by only retaining basis functions with a regression coefficient larger than a pre-specified threshold. We experimented with the method of Rudy et al. (2017) in this thesis, but we found the method to be sensitive to the pruning threshold, the strength of the regularisation, the amount of training data, and how spatial derivatives we were calculated in forming the basis functions.

We instead used the method of Zhang and Lin (2018), which uses a sparse Bayesian linear regression. The defining feature of Bayesian methods is the use of prior distributions, so instead of setting a parameter to a single value, a distribution of possible values is assumed. For an intuitive but comprehensive introduction to the Bayesian approach, the textbook of McElreath (2018) is a fantastic resource.

A *sparse* Bayesian linear regression assumes prior distributions on the regression coefficients. The priors on the regression coefficients encourage sparse solutions, as the priors prevent the coefficients taking unnecessarily large values and deviating too far from zero. In fact, it can be shown that Bayesian priors on the regression coefficients can be equivalent to adding a regularising penalty term to the loss function.

The key difference between Bayesian priors on the regression coefficients, and adding a regularising penalty term to the loss function, is that the Bayesian priors empirically find the optimal strength of the sparseness. In other words, the parameter of the regularising penalty term has to be set manually beforehand by the practitioner, whereas the parameters of the priors can be optimally estimated from the data. Therefore, Bayesian sparse regression can potentially lead to better feature selection compared to penalised loss functions. The Bayesian prior approach however assumes you have enough data such that the choice of prior does not significantly impact the prediction.

For example, the method of Zhang and Lin (2018) uses a sparse Bayesian regression based on relevance vector machines (Tipping, 2001). Relevance vector machines are a regression that assumes Gaussian prior distributions for each regression weight. The parameters of these prior Gaussian distributions are then also assumed to have prior distributions (priors on the parameters of prior distributions are known as hyperpriors). The parameters of the Gaussian priors are estimated from the data using type-II maximum likelihood estimation (Bishop, 2006). The key point of this method is that the width of the Gaussian prior of each regression weight provides a measure of uncertainty of that regression weight.

The pruning of the library of basis functions in Zhang and Lin (2018) is carried out by discarding the basis functions with an uncertainty higher than a pre-specified threshold. This uncertainty threshold is the only parameter that requires setting in the Zhang and Lin (2018) method. The algorithm finishes when the uncertainty measures of each regression weight stop changing from iteration to iteration. We found this method to be more robust than the Ridge regression of Rudy et al. (2017) and is therefore used in this chapter.

When applied to the problem of eddy parameterisation, this inference approach using sparse regressions has the following advantages over more complex methods

such as convolutional neural networks:

- The end result is significantly easier to interpret physically.
- The computational cost of implementation is lower, as the resulting parameterisation is simply derivatives and products of the resolved fields.
- Training time of the algorithm is also lower, as there are fewer parameters to optimise.

One disadvantage of the equation discovery method is the memory requirements, in order to load in the values of all basis functions at every point in space and time in the training data. For roughly 100 basis functions, 1000 time-slices, and a 128×128 spatial grid, the memory requirements can easily reach 50-100GB.

Also, another disadvantage is that the sparse regression may not be successful at capturing the variance in the data; it may be that a sparse regression does not have the capacity to capture the nonlinear effects of eddies, whereas the flexibility of neural networks may be required instead. For parameterising mesoscale eddies, we need to be cautious of the trade-off between the interpretability of inference approaches and the predictive skill of more black-box methods.

Toy example: Burger's equation

To finish this section describing the data-driven discovery of equations, we provide an illustrative example of the Zhang and Lin (2018) method used in Chapter 6. Our toy equation is the one-dimensional Burger's equation which, for the field $u(x, t)$, evolves through the following differential equation

$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = \nu \frac{\partial^2 u}{\partial x^2}, \quad (3.21)$$

where we set the viscosity coefficient to $\nu = 0.05$. Given some spatio-temporal data on the field $u(x, t)$, our aim is to discover the equation for the time tendency $\frac{\partial u}{\partial t}$. To generate the training and validation dataset, we set $u(x, t = 0)$ as a sine function and integrate forward for 240 time-steps. The space dimension has 260 grid points, so the total number of samples in our dataset is 240×260 , which is split 50:50 for training and validation.

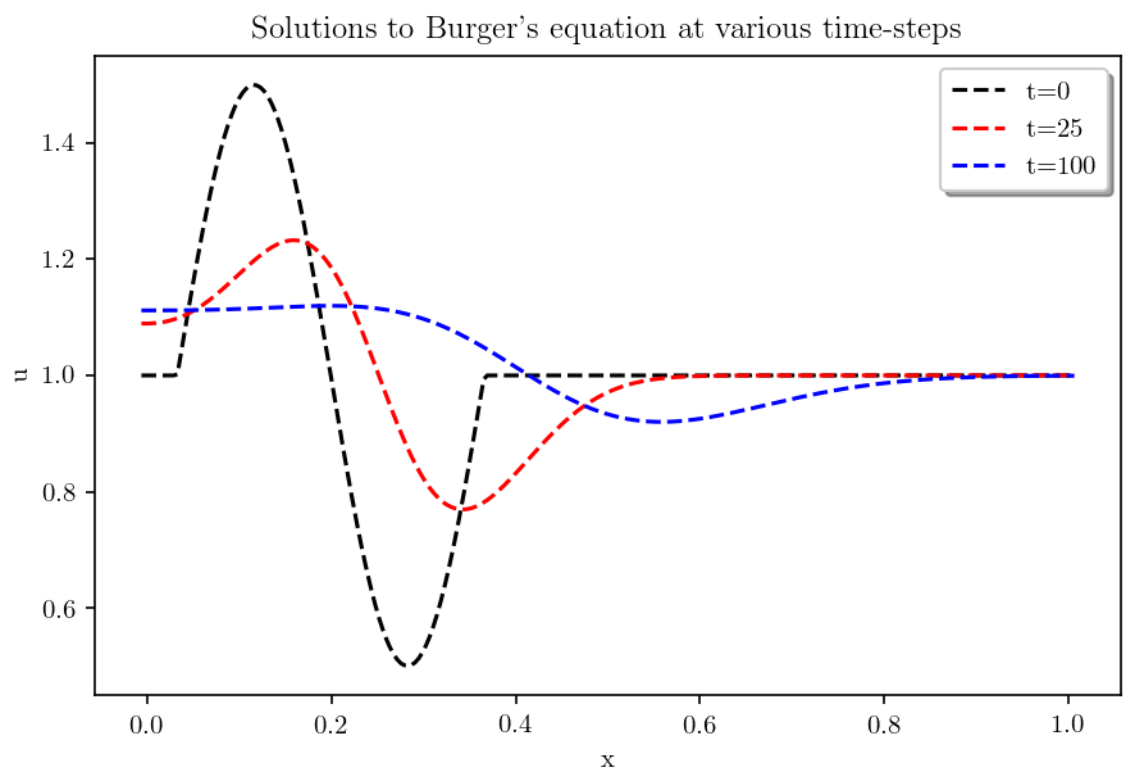


Figure 3.11: Solutions to the one dimensional Burger's equation, which comprises an advection and diffusion term, at various time-steps with a sine function initialisation.

Equation 3.21 is numerically solved with a second-order method in space and time, with the advective term evaluated explicitly and the diffusion term implicitly (Crank-Nicholson). A time-step of $\Delta t = 0.001$ was used in the numerical integration. Solutions at various time-steps are illustrated in Figure 3.11. The initial sine function (black curve) gets advected and diffused over time (red and blue curves).

We must now construct a library of basis functions which are relevant to the time-tendency $\frac{\partial u}{\partial t}$ that we are discovering an equation for. In this toy example, we use up to 2nd order spatial derivatives and 2nd order products of the field u . This produces a library of the following nine basis functions ϕ_i

$$\phi_i(u) = u, \quad \frac{\partial u}{\partial x}, \quad \frac{\partial^2 u}{\partial x^2}, \quad u^2, \quad \left(\frac{\partial u}{\partial x}\right)^2, \quad \left(\frac{\partial^2 u}{\partial x^2}\right)^2, \quad u \frac{\partial u}{\partial x}, \quad u \frac{\partial^2 u}{\partial x^2}, \quad \frac{\partial u}{\partial x} \frac{\partial^2 u}{\partial x^2}.$$

Higher order products and derivatives could have been included in the library, and in general the more basis functions the better, but for illustrative purposes we keep the number of basis functions low.

After the function library has been constructed, the iterative sparse regression is introduced to prune the library and (hopefully) converge towards the final expression. As previously mentioned, Zhang and Lin (2018) uses a relevance vector machine algorithm. At each iteration of the algorithm, the relevance vector machine predicts the time-tendency $\frac{\partial u}{\partial t}$ using a linear combination of the nine non-linear basis functions, i.e.,

$$\frac{\partial u}{\partial t} = \sum_i^9 w_i \phi_i(u), \quad (3.22)$$

where w_i are the regression coefficients of each basis function $\phi_i(u)$. Each regression coefficient has a measure of uncertainty, α_i , estimated from the training data. At the end of each iteration of the algorithm, if the uncertainty α_i is higher than the pre-specified threshold δ , then the corresponding basis function $\phi_i(u)$ is pruned from the library. The iterations of the algorithm continue until the uncertainties cease changing and no more basis functions are discarded. The remaining basis functions and their coefficient values form the final ‘discovered’ equation.

So how does this process manifest for our toy example? After testing various values of the pruning threshold, we apply the relevance vector machine algorithm with a threshold of $\delta = 35$ to the training data from the Burger’s equation. The

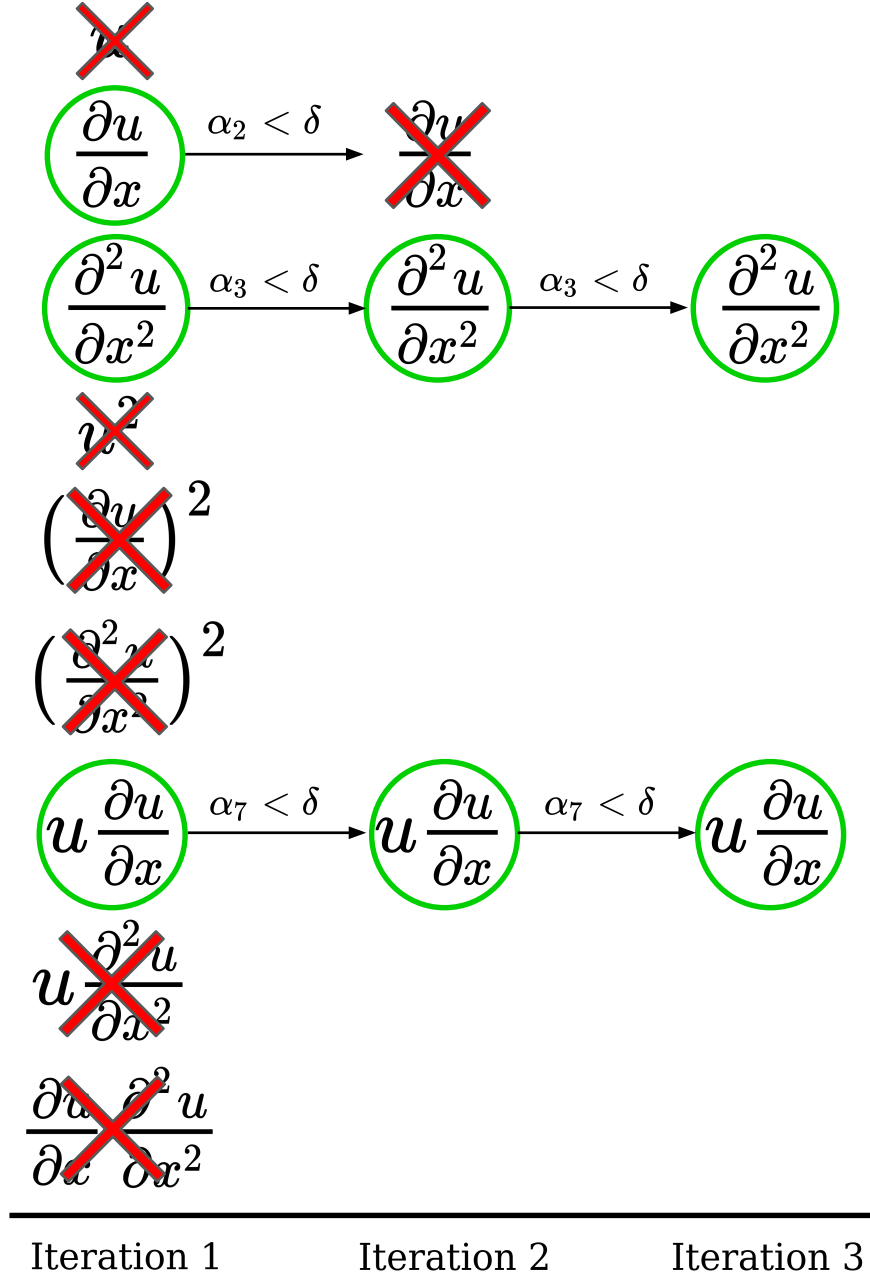


Figure 3.12: The iterative relevance vector machine algorithm applied to data from a one-dimensional Burger's equation. Nine basis functions comprise the library, of which three are selected by the algorithm after three iterations. At each iteration, basis functions with a regression coefficient with uncertainty α_i higher than the threshold δ are pruned from the library. Note that the uncertainties α_i are re-calculated from the data after each iteration.

algorithm takes three iterations to converge, with each intermediate result of each iteration illustrated in Figure 3.12. The remaining two basis functions, combined with their regression coefficients estimated from the data, leads to the following equation

$$\frac{\partial u}{\partial t} \approx w_3 \frac{\partial^2 u}{\partial x^2} + w_7 u \frac{\partial u}{\partial x}, \quad (3.23)$$

$$= 0.034 \frac{\partial^2 u}{\partial x^2} - 0.734 u \frac{\partial u}{\partial x}, \quad (3.24)$$

which captures 99.6% of the variance in the validation data. Note that the regression coefficients $w_3 = 0.034$ and $w_7 = -0.734$ are not exactly the true values of 0.05 and -1 respectively; this discrepancy is typically caused by co-linearity between basis functions, i.e., the algorithm may not be able to exactly separate the effects of diffusion and advection in this particular case. The relevance vector machine algorithm is nonetheless apt at revealing the dominant closed-form expressions underlying a spatio-temporal dataset.

Chapter 4

Observed eddy tracer fluxes in the North Atlantic

4.1 Introduction

Mesoscale eddies play an important role in the transport and mixing of tracers, including those relevant to climate, such as heat and carbon (e.g. Gnanadesikan et al., 2015). As discussed previously, we do not fully resolve these physical processes due to computational constraints. The unresolved transport and mixing of tracers, i.e., the eddy tracer fluxes, therefore requires parameterising, to account for their effects on the large-scale flow.

To accurately parameterise tracer fluxes, we need to understand how actual mesoscale eddies in the ocean impact tracer fluxes. In particular, how the spatial and temporal variability of mesoscale eddies impacts the resulting tracer fluxes. A creative approach to studying the effects of actual mesoscale eddies is to use satellite observations of ocean currents to drive a passive tracer experiment (Marshall et al., 2006; Abernathey and Marshall, 2013; Klocker and Abernathey, 2014; Busecke et al., 2017; Busecke and Abernathey, 2019). The observed ocean velocities evolve a virtual passive tracer field - the fluctuations in the tracer field are then solely due to the observed eddies.

Mesoscale eddies modify lateral tracer fluxes through stirring (i.e. advection which conserves tracer concentration), and the subsequent modification of tracer gradients. The along-gradient component of the tracer flux is typically quantified by an eddy diffusivity, which measures the amount of tracer flux per unit tracer gradient. This general process is known as eddy mixing. However, a more energetic mesoscale eddy field does not necessarily lead to increased eddy mixing, due to the phenomena of

‘mixing barriers’. While there is no strict definition of mixing barriers, they can be characterised by a significant suppression of the eddy diffusivity (e.g. Nakamura, 1996; Klocker et al., 2012).

Mixing barriers are typically caused by the presence of zonal jets, where the strong mean-flow produces a sharp coherent front, reducing or suppressing mixing despite the presence of turbulent eddies. It has been suggested Ferrari and Nikurashin (2010); Klocker et al. (2012) that mixing barriers reduce the across-jet diffusivity by effectively reducing the mixing length associated with the mixing length theory diffusivity of Prandtl (1925). An illustrative example is the Antarctic Circumpolar Current (ACC) in the Southern Ocean. The zonal jets of the ACC cause the across-jet diffusivity to drop significantly (Shuckburgh et al., 2009; Ferrari and Nikurashin, 2010; Klocker et al., 2012) despite the energetic mesoscale eddy field.

Within the North Atlantic, the presence of strong-mean flows and mesoscale eddies lead to competing effects, similar to the ACC: eddies generally act to increase lateral mixing through enhanced tracer gradients, whereas strong mean-flows lead to mixing barriers that suppress diffusivities. Bower et al. (1985) showed that within the Gulf Stream region, cross-frontal mixing is indeed suppressed at the surface. At the surface, the amplitude of the jet meandering impacts the amount of cross-frontal exchange (Bower and Rossby, 1989; Bower, 1991; Samelson, 1992), where jet meandering affects the mixing barrier suppression of the mean-flow. Reduced inter-gyre exchange across the Gulf Stream is observed for sea surface temperature (SST) anomalies (Foukal and Lozier, 2016).

The aim of this chapter is to better understand eddy tracer fluxes, in particular their spatio-temporal variability. We focus on the North Atlantic, as it exhibits a diverse range of dynamical processes, due to the energetic mesoscale eddy field interacting with the Gulf Stream, Azores, and North Atlantic currents. By seeing how observed mesoscale eddies impact tracer fluxes, we hope that we can learn how to parameterise them in the coarse-resolution ocean models.

Section 4.2 describes the satellite altimetry data, the passive tracer simulations, and the definitions of the up-stream and down-stream regions that we use in this chapter. In Section 4.3, we investigate the regional and temporal variability of the effective diffusivity, including the analysis of the mixing length diffusivity expression

of Ferrari and Nikurashin (2010). In Section 4.4, we link the up-stream effective diffusivity to the dynamics of the Gulf Stream jet. Finally, in Section 4.5, we discuss the implications of our results.

It should be noted that this chapter is a consequence of a collaboration with Professor Ryan Abernathey at Columbia University. The primary contribution of Professor Abernathey was conducting the global passive tracer experiment, driven by observations within MITgcm. The results of this chapter have been published in the Journal of Physical Oceanography (<https://doi.org/10.1175/JPO-D-19-0042.1>).

4.2 Data and methods

4.2.1 AVISO satellite altimetry data

We use two “level 4” gridded products from AVISO altimetry data¹: absolute-dynamic topography (ADT), and surface geostrophic velocities derived from ADT. ADT is formed by combining the mean dynamic topography (MDT) with sea level anomalies (SLA); MDT is the ocean topography due to the mean currents, relative to the geoid, and SLA are the satellite measurements of the oceans anomalous surface height. The surface velocities are derived from sea surface height data by assuming geostrophy. After an optimal interpolation procedure (Le Traon et al., 1998), the products have $\frac{1}{4}^\circ \times \frac{1}{4}^\circ$ spatial resolution on a Mercator grid, with daily data beginning from 1993.

4.2.2 Passive tracer model

To conduct a numerical passive tracer experiment driven by observations, we use MITgcm (Marshall et al., 1997; Adcroft et al., 2008) in “offline” mode. The AVISO geostrophic velocities only measure velocities at the surface, so it is a single layer experiment run globally. The model has coastlines, at which no normal flow boundary conditions are applied to globally conserve the passive tracer. Before running the model, the AVISO geostrophic surface velocities $\mathbf{u} = (u, v)$ are adjusted to be non-divergent, using an elliptical inversion to estimate an effective streamfunction, which corresponds to a 1% correction. A non-divergent velocity field is necessary in order to conserve the passive tracer. Weekly time-slices of the surface velocities, at each

¹<https://www.aviso.altimetry.fr/en/data.html>

time-step, are linearly interpolated to the model time, and also spatially interpolated to a $\frac{1}{10}^\circ$ grid. The following two-dimensional advection-diffusion equation is then solved for the passive tracer field q

$$\frac{\partial q}{\partial t} + (\mathbf{u} \cdot \nabla)q = \kappa \nabla^2 q, \quad (4.1)$$

where t is time, κ is a prescribed horizontal diffusivity, and $\nabla = (\partial/\partial x, \partial/\partial y)$ is the horizontal gradient. Note that here, q denotes the passive tracer field, and not the potential vorticity. The passive tracer experiment depends on the initial tracer distribution $q(t = 0)$ and the value of the horizontal diffusivity κ ; the sensitivity to both these factors are discussed in Abernathey and Marshall (2013), where various initialisations are tested, as well as values of κ ranging from 25 to 150 m^2s^{-1} .

Based on the results of Abernathey and Marshall (2013), we initialise the passive tracer on the time-mean streamfunction (averaged over the ≈ 20 year period of available data) such that the initial field is as aligned as possible with the mean-flow. Any subsequent induced tracer variance by the flow will, therefore, be due to transient motions. Initialising on the time-mean streamfunction also reduces the initial ‘shock’ of the passive tracer to the underlying velocities, and within the Gulf Stream the adjustment takes less than four weeks.

After many eddy lifetimes, the global tracer variance will decrease as the time-mean tracer flux will be directed down-gradient. The tracer is therefore re-initialised every year to avoid complete homogenisation. Re-initialising the passive tracer at the beginning of each year also improves the study of the interannual variability, as the initial adjustment then occurs within each year.

4.2.3 Up-stream and down-stream regions

To determine how the dynamics within the North Atlantic impact lateral mixing, and how the lateral mixing varies along-stream, we define two regions: up-stream as 80-50W, and down-stream as 50-10W, with both between 30-60N in latitude. The two regions are shown in Figure 4.1a and b, with the grey-dotted line illustrating the boundary between the up- and down-stream regions.

The up-stream is characterised by the presence of the Gulf Stream jet, a coherent current which separates from the coast at Cape Hatteras at 35N, after which the jet

meanders eastward. The strong mean-flow in the up-stream region produces a sharp front, also known as the Gulf Stream North Wall, argued to act as a barrier to mixing at the surface (Bower et al., 1985).

In the down-stream region, the Gulf Stream jet evolves into the North Atlantic Current and the Azores Current (Rossby, 1996), the former transporting warm tropical waters to high latitudes, and the latter flowing in a south easterly direction contributing to transport within the sub-tropical gyre. The North Atlantic Current and the Azores Current do not produce as sharp a front as the Gulf Stream jet, but both the up- and down-stream regions exhibit an energetic mesoscale eddy field, which impacts the lateral mixing of tracers.

4.3 North Atlantic lateral tracer mixing

4.3.1 Tracer statistical moments

To illustrate the time-mean behaviour of the passive tracer in the North Atlantic, we calculate the first four statistical moments of the passive tracer field and compare them to observations of an active tracer, namely sea surface temperature (SST) (Figure 4.1). We use SST observations from the ARMOR3D dataset (Guinehut et al., 2012) and remove the seasonal cycle at each grid point - the seasonal cycle was estimated by calculating the average SST value for each week of the year. Each statistical moment helps describe properties of the tracer probability distribution function (PDF), and the first four moments are the following: mean, standard deviation, skewness, and kurtosis. The standard deviation measures the width of the PDF, and is therefore a measure of the temporal variability; skewness is a measure of the asymmetry of the PDF; kurtosis indicates how probability is distributed between the peak and tails of the PDF and relates to the frequency of extreme values.

The mean of both the SST and passive tracer exhibit a sharp front, characterised by large meridional gradients, in the up-stream region, where there is a strong coherent jet. The mean fields of SST and the passive tracer in the down-stream region are more diffuse, as the Gulf Stream jet evolves into the less-vigorous North Atlantic Current and Azores Current. The standard deviation is dominated by the Gulf Stream jet in the up-stream region, with an order of magnitude more variance than in the down-stream region. The skewness of both the SST and passive tracer undergo a sign

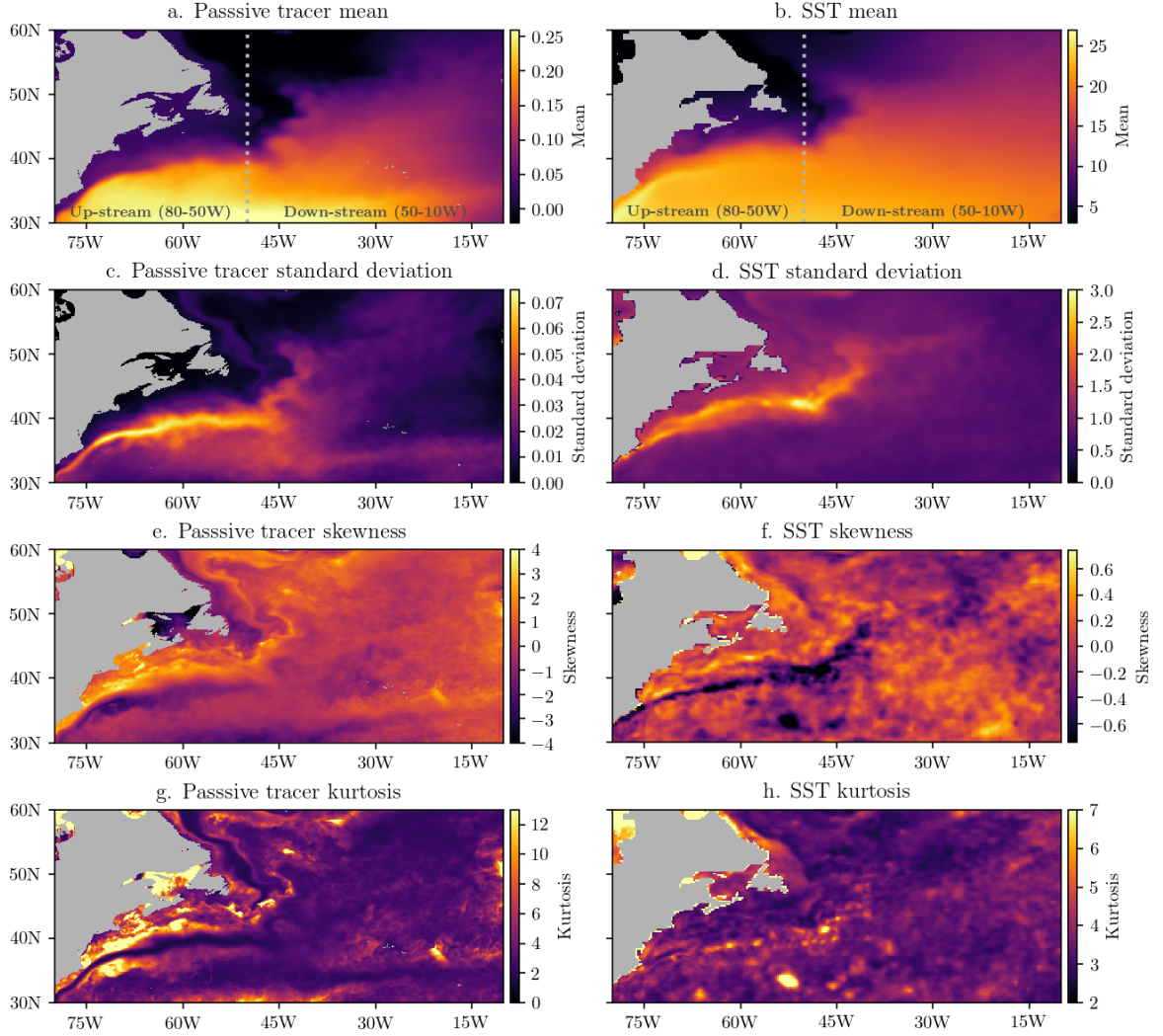


Figure 4.1: The first four statistical moments of SST and the passive tracer q by sampling values at each point in space. The rows from top to bottom represent the four statistical moments: mean, standard deviation, skewness and kurtosis. The first column is for 22 years of passive tracer data, and the second column is for SST observations over the same period. SST data is from the ARMOR3D dataset (Guinehut et al., 2012). The mean and standard deviation have units of tracer concentration (or degrees Celsius for SST), while the skewness and kurtosis are dimensionless. The up- and down- stream regions (80-50W and 50-10W respectively), are illustrated in Panels (a) and (b).

change moving across the jet in the up-stream region. The skewness sign change is indicative of a mixing barrier, as discussed in Hughes et al. (2010); David et al. (2017); the latitude of zero skewness represents the mean-jet position, as it is the latitude at which the jet core is equally likely to be above or below that point. The passive tracer kurtosis displays a high-low-high pattern moving across the jet, also indicative of a mixing barrier. However the SST kurtosis does not show this high-low-high pattern, potentially due to interactions unique to an active tracer like SST, which is not captured by the passive tracer field.

The statistical moments indicate that, at least in the time-mean evolution of SST and the passive tracer, there are important dynamical differences between the up- and down-stream regions. In addition, except for the kurtosis, the passive tracer exhibits qualitatively similar statistics to the SST field. Any insight gained by studying passive tracer mixing can then hopefully generalise to climatic tracers in the North Atlantic.

4.3.2 Effective diffusivity and tracer Flux

Eddy mixing involves the enhancement of tracer gradients due to the conservative stirring of the eddy advection. The conservative stirring deforms tracer contours into finer spatial-structures, from which the background horizontal diffusivity κ becomes significant and homogenises the tracer. The mixing produced through this mechanism can be represented with an eddy diffusivity K_{eddy} . One measure of eddy mixing by enhanced tracer gradients is the Osborn-Cox diffusivity (Osborn and Cox, 1972), which measures the enhancement of tracer gradients relative to the background gradients $K_{OC} = \frac{|\overline{\nabla q'}|^2}{|\overline{\nabla q}|^2}$, where $\overline{x}$ denotes the time-mean and x' denotes the deviation from the time-mean.

However, the Osborn-Cox diffusivity K_{OC} calculates the tracer gradients in an Eulerian manner at each point in space. When applied to a system such as the Gulf Stream, large anomalous tracer gradients $|\nabla q'|^2$ produced simply from the latitudinal shifts in the meandering jet can strongly affect the estimates of diffusivity, despite not contributing to irreversible mixing.

We therefore use the effective diffusivity K_{eff} of Nakamura (1996), a quasi-Lagrangian diagnostic of mixing that also considers the magnitude of tracer gradients, but using the tracer area A as a coordinate system. The tracer area $A = A(q)$ is the

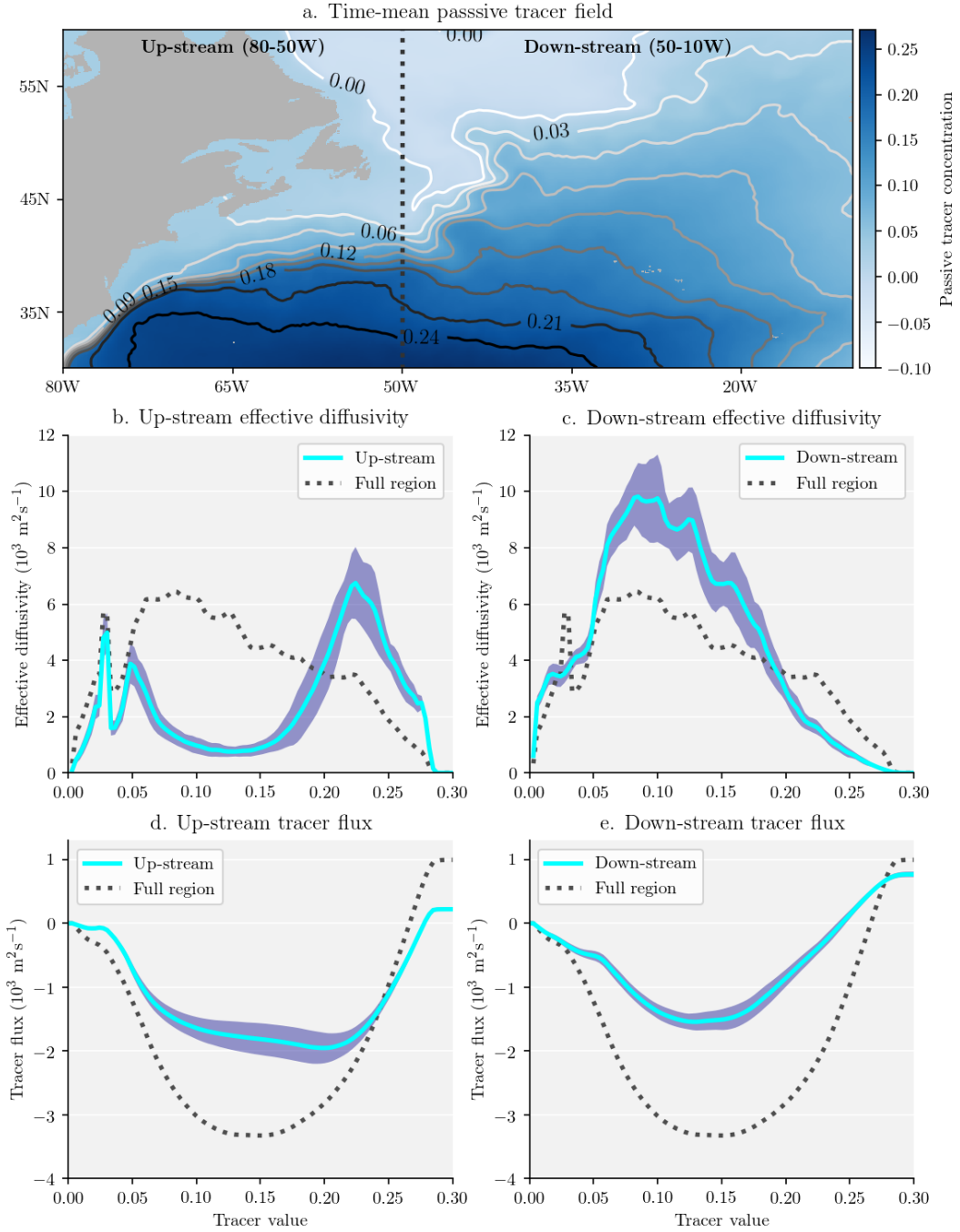


Figure 4.2: The effective diffusivity, and the corresponding tracer flux, calculated for two different regions of the North Atlantic: up-stream, defined as 80-50W, and down-stream, defined as 50-10W. The boundary between up- and down-stream is shown by the black-dotted line in Panel (a). Panels (b) and (c) show the effective diffusivity for the up- and down-stream regions respectively, while Panels (d) and (e) show the corresponding tracer flux. The shading in Panels (b)-(e) represents the interannual variability, defined as the standard deviation of the annual means. The blacked-dotted lines in Panels (b)-(e) illustrate the calculation over the full region.

total area of the tracer below the contour q (a visual explanation of the area coordinate can be found in Figure 1 of Nakamura (1996)). The effective diffusivity K_{eff} and Osborn-Cox diffusivity K_{OC} are related, as average the Osborn-Cox diffusivity along tracer contours is equivalent to the effective diffusivity (Nakamura, 1996).

More formally, the effective diffusivity is derived by transforming the advection-diffusion Equation 4.1 from Eulerian spatial coordinates to the Lagrangian area coordinate A . Transforming to area coordinates (Nakamura, 1996) leads to the following diffusion-only equation

$$\frac{\partial q}{\partial t} = \frac{\partial}{\partial A} (K_{eff} \frac{\partial q}{\partial A}), \quad (4.2)$$

where the effective diffusivity K_{eff} is given by

$$K_{eff}(q, t) = \kappa \frac{\frac{d}{dA} \int_A |\nabla q(t)|^2 dA}{L_{min}^2 (\frac{\partial q(t)}{\partial A})^2}, \quad (4.3)$$

where κ is the same prescribed model diffusivity from Equation 4.1, and L_{min} is the minimum length of a tracer contour. The effective diffusivity K_{eff} is a function of the tracer value q , and the area integral $\int_A$ is computed over the region of $\tilde{q} < q$. The effective diffusivity has a corresponding across-contour flux F_{eff} , defined as (c.f. Equation 2.5b in Nakamura (1996))

$$F_{eff} = -\kappa \int_A (\nabla^2 q) dA, \quad (4.4)$$

and represents the total cross-contour transport.

Using Equation 4.3, we calculate the time-mean and annual-means of K_{eff} for the passive tracer in both the up- and down-stream regions. Note that by calculating K_{eff} in sub-regions of the North Atlantic, we are assuming that there is negligible net tracer flux into (or out of) the region. As we initialise the passive tracer on the time-mean streamfunction, the mean-flow approximately transports equal amounts of tracer in and out of the region. This would not be the case if the tracer was initialised on some other field, such as latitude - the adjustment of the tracer to the mean-flow would cause large net fluxes in and out of the region.

Figure 4.2b and c shows the time-mean effective diffusivity for the up- and down-stream regions respectively as a function of tracer value q . The time-mean passive

tracer field is shown in Figure 4.2a to illustrate where each value of the tracer q occurs in the North Atlantic; in general, lower values of q correspond to more northerly latitudes. The essential difference between the up- and down-stream regions is that up-stream exhibits a significant mixing barrier in the jet core (tracer values $q = 0.1-0.15$), with diffusivity values of $\approx 1 \times 10^3 \text{ m}^2\text{s}^{-1}$ (Figure 4.2b). The down-stream region however exhibits a single peak which reaches a value of $\approx 10 \times 10^3 \text{ m}^2\text{s}^{-1}$ (Figure 4.2c). Therefore the coherent Gulf Stream jet in the up-stream region significantly suppresses mixing, but not the North Atlantic Current and Azores Current of the down-stream region.

Despite the suppressed effective diffusivity in the up-stream region, the magnitude of the cross-contour tracer flux is larger in the up-stream region (Figure 4.2d) relative to down-stream (Figure 4.2e)—note that negative values of the tracer flux indicate a net transport from positive to negative tracer values, which is primarily a poleward transport. Therefore, despite the mixing barrier, there is more across-jet exchange of tracer in the up-stream region. This is due to the tracer gradients being significantly larger in the up-stream region, as can be seen by the tightly-packed contours in Figure 4.2a. The large tracer gradients in the up-stream region more than compensate for the decreased diffusivity when contributing to the total tracer flux.

The interannual variability of the diffusivity coefficient and flux is defined as the standard deviation of the annual means, and is illustrated by the shading in Figures 4.2b-e. For both up- and down-stream, the interannual variability is 10-20% of the time-mean. The temporal variability of the effective diffusivity is more clearly illustrated in Figure 4.3a and b, which shows the diffusivity as a function of both time and the tracer value, for the up- and down-stream regions respectively. Figure 4.3a shows that the mixing suppression effect in the up-stream region is present in every year of the data, with the single peak in the down-stream region in Figure 4.3b also present in every year. The along-stream variation in the diffusivities is therefore robust across all years of this experiment.

The individual time series of the jet core diffusivity ($q=0.15$) are shown in Figure 4.3c and d for the up- and down-stream regions respectively. The up-stream and down-stream diffusivities at $q=0.15$ vary independently of each other, with the

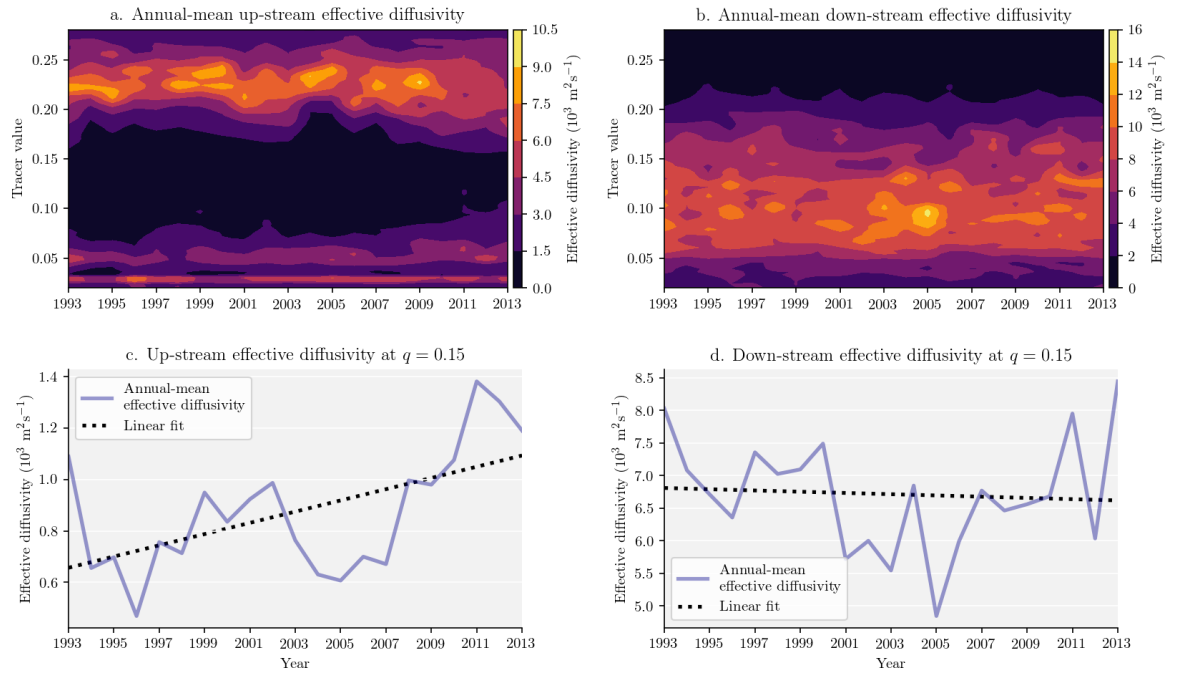


Figure 4.3: Panels (a) and (b) show the time evolution of the effective diffusivity as a function of tracer value (y-axis), for the up- and down-stream regions respectively. Panels (c) and (d) show the individual time series at the tracer value $q = 0.15$, which is approximately co-located with the 25 cm SSH contour in the jet core. The black-dotted lines in Panels (c) and (d) are linear fits in time.

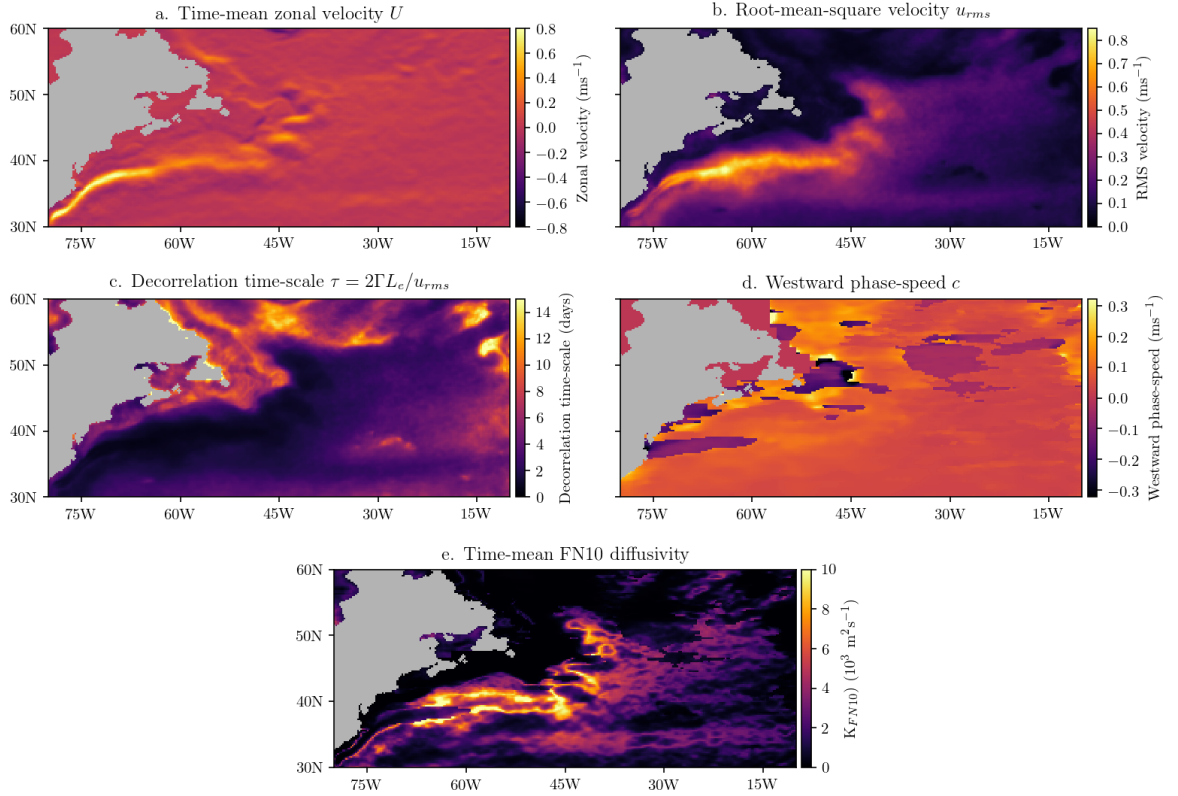


Figure 4.4: The various components of the Ferrari and Nikurashin (2010) diffusivity expression, calculated from satellite altimetry data. Panel (a) shows the time-mean zonal velocity; Panel (b) shows the root-mean square (RMS) eddy velocity, where eddy is defined as the deviation from the time-mean; Panel (c) shows the decorrelation time-scale in days, using the formula $\tau = 2\Gamma L_e/u_{rms}$; Panel (d) is the eddy phase speed data, provided by C. Hughes (personal communication). Panel (e) shows the final product: the time-mean diffusivity expression K_{FN10} .

up-stream diffusivity displaying long-term positive trend ($p < 0.05$) of $22 \text{ m}^2\text{s}^{-1}/\text{year}$ (the trend in the down-stream diffusivity was not significant).

4.3.3 Mixing length diffusivity expression

To determine the drivers of the observed temporal variability in Figure 4.3, we diagnose a physically-motivated expression for the lateral eddy diffusivity based on mixing length theory, which can be estimated from available satellite altimetry data. We employ the expression for the across-jet diffusivity derived by Ferrari and Nikurashin (2010) (referred to hereafter as FN10), allowing us to link specific changes in the flow field to changes in the lateral mixing.

The FN10 diffusivity assumes a weakly nonlinear isotropic eddy field, takes into

account the eddy phase speed c , the background zonal velocity U , and the root-mean-square (RMS) eddy velocity u_{rms} . The expression is given by

$$K_{FN10} = \frac{\Gamma L_e u_{rms}(t)}{1 + \frac{k^2}{\gamma^2} (U(t) - c)^2}, \quad (4.5)$$

where Γ is the mixing efficiency, L_e is the eddy length scale, γ is the reciprocal of the eddy decorrelation timescale, and $k = 2\pi/L_e$ is the eddy wavenumber. Equation 4.5 can be thought of a mixing length theory diffusivity (Prandtl, 1925) with a modified mixing length due to the mean-flow. The denominator in this equation acts as a mixing suppression factor, which arises due to eddy propagation relative to the mean flow. The FN10 diffusivity (Equation 4.5) has been used to reproduce mixing rates in other areas of the ocean, where both jets and turbulent eddies are present (Ferrari and Nikurashin, 2010; Klocker and Abernathey, 2014).

The components which contribute most to the temporal variability of the lateral mixing based on FN10 will therefore reveal the important underlying dynamical processes driving effective diffusivity variability (Figure 4.3). The various factors involved in the FN10 diffusivity expression are shown in Figure 4.4, which are the following: the time-mean zonal velocity U ; the RMS eddy velocity u_{rms} , with eddy defined as deviation from the time-mean; the eddy decorrelation time-scale $\tau = 2\Gamma L_e/u_{rms}$ (Klocker and Abernathey, 2014, Equation 8) with a constant eddy length scale of 30km; eddy phase speed c , where positive indicates westward propagation and is calculated from observational data using Radon transforms (Chris Hughes, pers. comm. 2018). The resulting time-mean FN10 diffusivity K_{FN10} is shown in Figure 4.4e. The time-mean FN10 diffusivity exhibits the mixing barrier effect up-stream within the jet core at approximately 36-37N, with increased values on the jet flanks. There is relatively little mixing suppression down-stream, consistent with the effective diffusivity in Figure 4.2.

To better compare the FN10 diffusivity with the effective diffusivity in Figure 4.2, we average the FN10 diffusivity along SSH contours. This provides a quasi-Lagrangian perspective, similar to the use of the area coordinate in the effective diffusivity framework (Nakamura, 1996). To do this, we calculate all the components of the FN10 diffusivity shown in Figure 4.4 along SSH contours *before* calculating the final K_{FN10} expression. Therefore, all quantities are calculated as a function of SSH and not Eulerian spatial coordinates. Given the approximate nature of the formula, together with

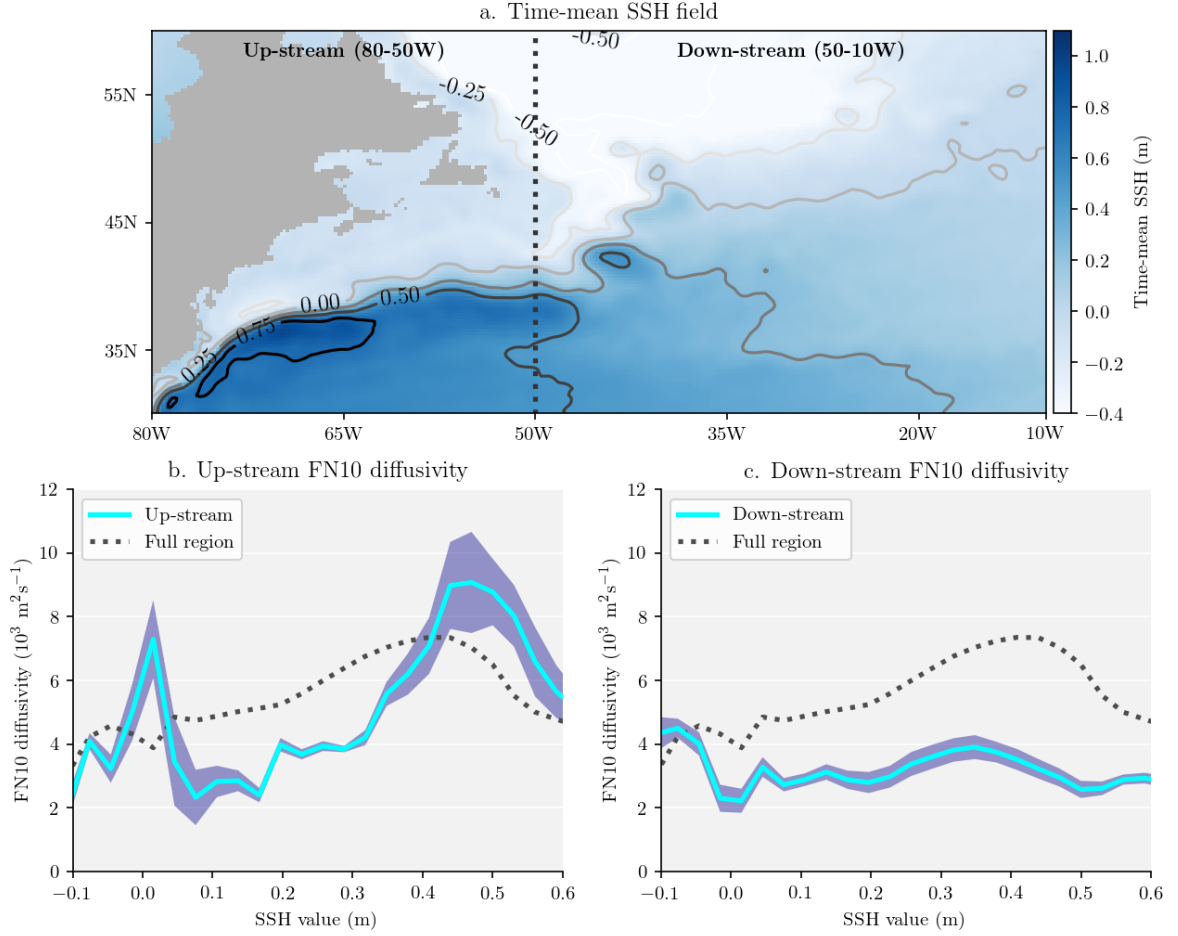


Figure 4.5: Panel (a) displays the time-mean SSH field from satellite altimetry data, illustrating where each SSH value is approximately located in the North Atlantic. Panels (b) and (c) show the K_{FN10} diffusivity expression calculated along SSH contours. The shading represents the interannual variability, defined as the standard deviation of the annual means. The black-dotted lines show the FN10 diffusivity calculated over the full region.

the spatial inhomogeneity of the input fields, we do not expect to obtain a perfect quantitative match between K_{FN10} and K_{eff} in the Gulf Stream region. Rather, as mentioned above, our goal is to use the formula to gain physical understanding of what governs mixing and its variability.

The K_{FN10} diffusivity is shown as a function of SSH in Figure 4.5b and c for the up-stream and down-stream regions respectively, with the shading representing the interannual variability. In the SSH coordinate system, the mixing suppression of the up-stream region is more evident in the K_{FN10} (although it is approximately double in the jet core compared to K_{eff}). The K_{FN10} diffusivity in the down-stream region

does not exhibit a mixing barrier, consistent with the along-stream variation observed with the effective diffusivity, but has a magnitude approximately five times smaller than K_{eff} .

Therefore, the K_{FN10} diffusivity slightly over-estimates the effective diffusivity in the up-stream region and strongly under-estimates the effective diffusivity in the down-stream region. The over-estimation of the mixing in the up-stream region could be due to the non-zonal mean flow at the west side of the up-stream region. Indeed, if the region with less zonal flow is excluded, we do see slightly better agreement between the effective and FN10 diffusivities in the up-stream region. Hence, the FN10 diffusivity is capturing the mixing barrier suppression in the up-stream region.

The mismatch between the effective diffusivity and FN10 diffusivity down-stream is more severe and could have been caused by non-local effects. By non-local effects, we mean changes to the diffusivity caused by fluxes/transport of tracer in and out of the region. The destabilised Gulf Stream jet in the up-stream region may cause significant irreversible mixing of the passive tracer, which then propagates into the down-stream region. The effective diffusivity may interpret this change as an increased diffusivity in the down-stream region, although the mixing actually occurred up-stream.

However, we tested the sensitivity of the down-stream effective diffusivity to changes in the dividing boundary separating up- and down-stream. We tested longitudes from 65W to 30W (about the original longitude of 50W). Regardless of the definition of the down-stream region, the large peak in the effective diffusivity remains, implying that non-locality does not contribute significantly to the observed peak in Figure 2c. We also do not observe any relationship between the flux of tracer variance into the down-stream region and the diffusivity. These results suggest that the effective diffusivity is a robust indicator of mixing in these region.

The mismatch may instead be due to the lack of time-scale separation between mean-flow and eddies in the down-stream region, which could cause large errors in the FN10 diffusivity. Consequently, the FN10 formula may not be capturing all the physical processes occurring in the down-stream region.

To study the interannual variability in more detail, we can linearise the FN10 expression in Equation 4.5 by assuming that only U and u_{rms} vary from year-to-

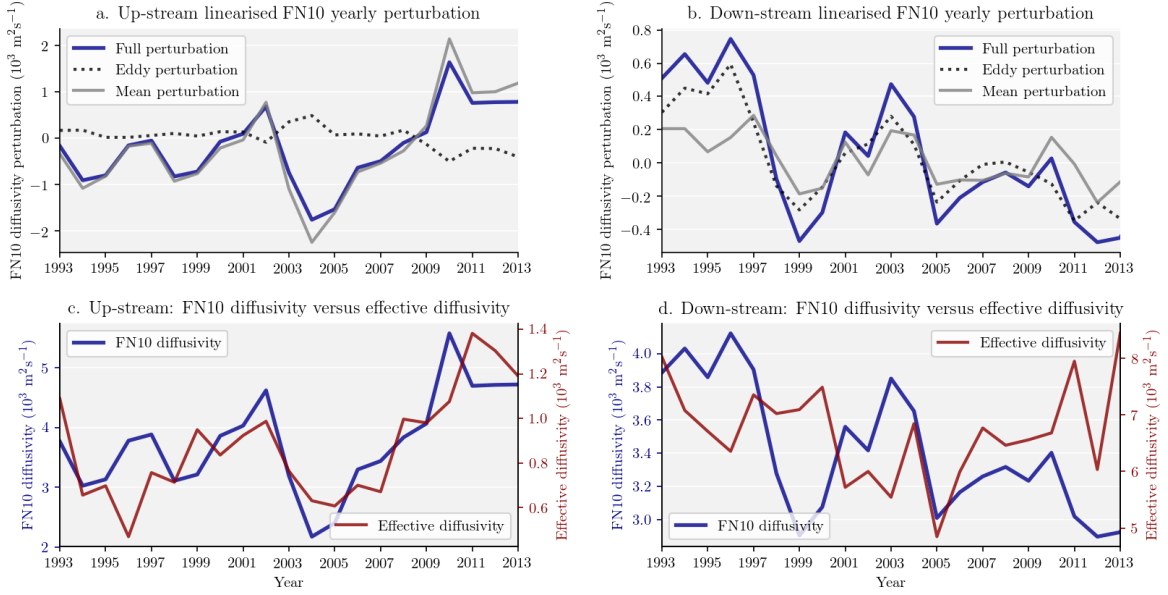


Figure 4.6: Panels (a) and (b) show linear perturbations to the time-mean K_{FN10} diffusivity expression for both the up- and down-stream regions. Panels (c) and (d) compare time series of the K_{FN10} diffusivity with the effective diffusivity in the jet core (at the 25 cm SSH contour for K_{FN10} and $q = 0.15$ for K_{eff}).

year, while all other quantities remain constant. Perturbations to K_{FN10} can then be written as the sum of contributions due to changes in the eddy velocity and the mean zonal velocity

$$\Delta K_{FN10} = \underbrace{\frac{\overline{K_{FN10}}}{\overline{u_{rms}}} \Delta u_{rms}}_{\text{Eddy perturbation}} - \underbrace{\frac{\overline{K_{FN10}}}{1 + \frac{k^2}{\gamma^2} (\overline{U} - c)^2} \Delta U}_{\text{Mean perturbation}}, \quad (4.6)$$

where we have decomposed the following variables into time-means $\bar{x}$ and annual deviations from the time-mean Δx

$$K_{FN10} = \overline{K_{FN10}} + \Delta K_{FN10}; \quad (4.7)$$

$$U = \overline{U} + \Delta U; \quad (4.8)$$

$$u_{rms} = \overline{u_{rms}} + \Delta u_{rms}. \quad (4.9)$$

The yearly linearised perturbations to the FN10 diffusivity for the up- and down-stream regions are shown in Figure 4.6a and b respectively. We focus on the FN10 diffusivity values along the 25cm SSH contour (which represents the core of the jet in up-stream region). The total perturbation to the FN10 diffusivity in the up-stream region is dominated by perturbation from the mean-flow ΔU . Indeed, the

perturbation due to ΔU accounts for 98% of the total perturbation in the up-stream region.

Down-stream on the other hand shows more equal contributions to the total perturbation by the mean-flow ΔU and eddy RMS velocity Δu_{rms} . As the currents are relatively weaker in the down-stream region, the mean-flow is more comparable in magnitude to the eddy velocity, both of which are also more comparable to the eddy phase speed. Therefore the temporal variability of the FN10 diffusivity in the down-stream region seems to be a more complex interplay between the mean-flow, eddy velocity, and eddy phase speed.

Does the temporal variability of the FN10 diffusivity match that of the effective diffusivity? We compare the individual time series of the effective diffusivity and FN10 diffusivity in the up- and down-stream regions in Figure 4.6c and d. We consider the 25 cm SSH contour for K_{FN10} , and the $q=0.15$ tracer contour of K_{eff} ; both the 25 cm SSH contour and the $q=0.15$ contour correspond to the jet core. The FN10 diffusivity tracks the longer-term decadal variations in the effective diffusivity ($r=0.75$, $p=0.02$) for the up-stream region, implying approximate agreement.

There is no relationship between the interannual variations in the FN10 diffusivity and effective diffusivity in the down-stream region ($r=-0.01$, $p=0.5$). As discussed previously, non-local contributions to the effective diffusivity appear minimal. Therefore it may be the case that the FN10 diffusivity expression does not accurately capture the correct physical processes in this region.

One caveat should be mentioned: when linearising the FN10 diffusivity, the functional dependence of the eddy decorrelation time-scale $\tau = 2\Gamma L_e / u_{rms}$ was *not* taken into account, i.e., the decorrelation time-scale was assumed constant. In hindsight, this should have been taken into account and could alter the contribution of the eddy velocity to variations in the FN10 diffusivity.

4.4 Up-stream Gulf Stream jet dynamics

In Figure 4.6, the linearised perturbations to the FN10 diffusivity imply the mean-flow is the dominant driver of lateral mixing variability in the up-stream region. The dynamics of the up-stream region are dominated by a strong and coherent jet—do

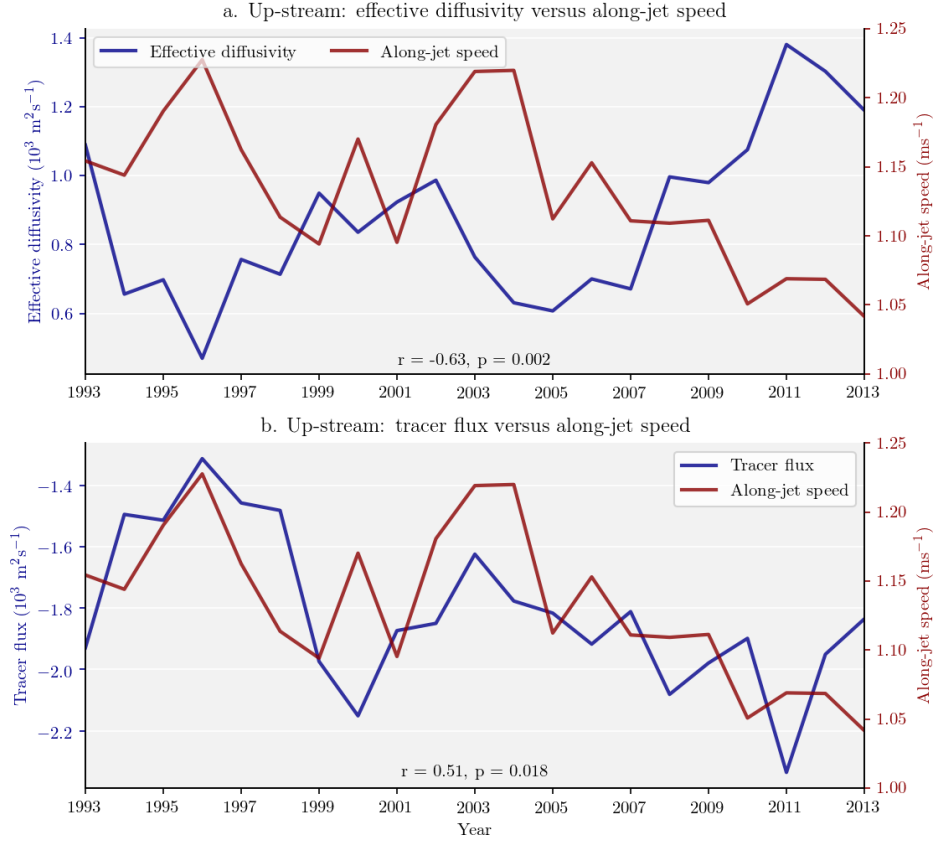


Figure 4.7: Comparisons of the up-stream effective diffusivity and tracer flux with a measure of mean-flow: along-jet speed. Panel (a) compares the along-jet speed with the effective diffusivity, while Panel (b) compares the along-jet speed with the tracer flux - both at the tracer value $q=0.15$. Note that a more negative tracer flux indicates an increased across-jet exchange of the passive tracer.

we see direct links between the dynamical properties of the jet and the diagnosed diffusivities?

We diagnose a quantity relevant to the mean-flow for each year in the up-stream region: the along-jet speed, defined as the average current speed along the 25 cm SSH contour, which provides a Lagrangian measure of ‘strength’ in the jet core. Annual values of the along-jet speed are compared with annual values of the effective diffusivity and flux in Figure 4.7.

Figure 4.7a shows there is a negative correlation ($r=-0.63, p < 0.01$) between the up-stream effective diffusivity K_{eff} and along-jet speed. Figure 4.7b shows a positive correlation ($r=0.51, p < 0.05$) between the up-stream tracer flux and the along-jet speed (with a more negative tracer flux corresponding to an increase in the across-jet exchange of tracer). Overall, these relationships show that, within the up-stream

region, a reduction in Gulf Stream mean-flow is linked to increased across-jet tracer transport. We find similar results when using the area-average mean-kinetic energy (MKE) as a diagnostic of mean-flow.

No significant relationships were found between the regional-average eddy-kinetic energy (EKE), and the effective diffusivity and tracer flux in the up-stream region. The lack of relationship with EKE implies it is specifically the suppression of mixing by the mean-flow which is the primary driver of across-jet transport variability in the up-stream region. The result is consistent with the analysis of linearised perturbations of the FN10 diffusivity in Figure 4.6.

4.5 Discussion and summary

We investigated the interannual variability of the lateral mixing of a passive tracer in the North Atlantic, using an observation-driven passive tracer experiment in MITgcm. The passive tracer behaves similarly to observed SST (Figure 4.1). Using the effective diffusivity of Nakamura (1996), we showed that lateral mixing at the surface exhibits significant along-stream variation in the North Atlantic (Figure 4.2). In the up-stream region (80-50W), the Gulf Stream jet exhibits a suppressed diffusivity at the surface due to the strong mean-flow (Figure 4.2b), consistent with Bower et al. (1985). The North Atlantic Current and Azores current in the down-stream region (50-10W) however, do not show any signs of reducing the effective diffusivity, which would occur if they were in fact suppressing mixing (Figure 4.2c).

The interannual variability of the effective diffusivity is 10-20% of the time-mean (Figure 4.2) for both the up- and down-stream regions. To determine the drivers of the temporal variability of the effective diffusivity, we used mixing length theory of Ferrari and Nikurashin (2010) to understand the physical controls on the across-jet diffusivity. The FN10 diffusivity successfully reproduces the mixing barrier in the up-stream region (Figure 4.5). In terms of the interannual variability, the FN10 diffusivity tracks the effective diffusivity in the up-stream region ($r=0.75$, $p=0.02$) (Figure 4.6c), whereas the FN10 diffusivity in the down-stream region does not match the effective diffusivity ($r=-0.01$, $p=0.5$) (Figure 4.6d).

By assuming the mean-flow and eddy velocity can vary from year-to-year, and linearly perturbing the FN10 diffusivity about the time-mean, we determine what

dynamical properties of the flow contribute to the interannual variability. We find that the perturbations due to the mean-flow dominate the interannual variability in the up-stream region (Figure 4.6a), whereas down-stream the mean-flow and eddy velocity are more balanced in their contributions to the FN10 diffusivity (Figure 4.6b).

The dominant role of the mean flow for the up-stream FN10 diffusivity variability led us to look for direct links between Gulf Stream mean-flow and the effective diffusivity. Using a measure of mean-flow, the along-jet speed, we find an inverse relationship between jet strength and the amount of across-jet mixing: reduced mean-flow leads to less suppression and more across-jet exchange of tracer on interannual time-scales (Figure 4.7).

The distinct mixing effects of the up-stream and down-stream regions show that a parameterisation of eddy mixing must account for this along-stream variation. Yes, the FN10 diffusivity successfully accounts for the mean-flow suppression in the up-stream region, but it is still unclear what the primary drivers of mixing are in the down-stream region. There is not a clear separation of mean-flow and eddies in the down-stream region, which may contribute to the issue, as scale-separation is an assumption of the FN10 diffusivity. An eddy parameterisation must therefore be sensitive to the dynamical intricacies of each region.

Along-stream variation in the interactions of eddies has previously been studied in an idealised model (Waterman and Jayne, 2010) and using observations of the Kuroshio Extension (Waterman et al., 2011): the up-stream region is characterised by an unstable jet generating eddies, whereas down-stream the jet radiates energy, known as the wave-maker regime. Liu et al. (2018) studied Lagrangian structure dynamics in the Gulf Stream using satellite altimetry data, in order to quantify transport and mixing. Lobes and eddy pinch-off were found to contribute to the across-jet transport, and this behavior was interpreted in terms of the unstable jet and wave-maker regimes of Waterman and Jayne (2010). These studies consider along-stream variation on the scale of $\mathcal{O}(100 \text{ km})$, whereas here we have considered up- and down-stream variations of order $\mathcal{O}(1000 \text{ km})$. Nonetheless our results and the aforementioned studies show that the along-stream variations in eddy interactions are an important component of the large-scale circulation, particularly in Western boundary currents.

The discrepancy between the effective diffusivity and the FN10 diffusivity in the down-stream region—in both the time-mean and interannual variability—requires further study. The linearised analysis of Figure 4.6 indicates a more complex interaction between the mean-flow, eddy velocity, and eddy phase speed, with regards to the temporal variability of mixing in the down-stream region.

Our results only concern lateral mixing at the surface, whereas sub-surface mixing is also an important component of tracer transport in the North Atlantic. The study of Bower et al. (1985) suggested that the mixing-barrier effect of the Gulf Stream may diminish with depth. However, along-stream variation in sub-surface mixing requires further study - how does the sub-surface mixing vary between the up- and down-stream regions? How does the sub-surface mixing vary on interannual time-scales? The Thorpe and Ozmidov length-scales (Thorpe, 1977; Dillon, 1982) are measures of turbulent overturning events, which depend on the kinetic energy dissipation and buoyancy frequency - these length-scale may exhibit important differences between the up- and down-stream regions.

These questions cannot be answered with satellite altimetry data alone. Indeed, Foukal and Lozier (2016) studied Lagrangian trajectories in the North Atlantic and suggested that it is the sub-surface transport which is important for connecting the Gulf Stream with the sub-polar gyre, and not transport at the surface - this was originally argued by Bower et al. (1985). For the Kuroshio Extension, whether the cross-jet transport increased with depth was dependent on along-stream position (Chen et al., 2014), and in the Southern Ocean, the local maximum of cross-jet mixing occurred between 1000-1500m (Griesel et al., 2014).

The mean-flow is the primary driver of mixing in the up-stream region, but what causes the variability in the mean-flow to begin with? We know that temporal variability in the surface-transport of the Gulf Stream and Kuroshio Extension are inversely related to a quantity called “path stability” (Kelly et al., 2010, Figure 14). Path stability is defined as the standard deviation of jet path latitude, over a pre-defined time period. The path stability is therefore related to the amplitude of jet meandering and the instability mechanisms that generate the meandering. Recent work by Chen et al. (2017) showed that the transitioning from a stable to unstable

state (i.e. increased jet latitude variance) of the Kuroshio Extension led to increased across-jet mixing.

We examined links between path stability and lateral mixing in the up-stream region. We defined the path stability of the Gulf Stream using the standard deviation of the latitude of the 25cm SSH contour, similar to Andres (2016). We found positive correlations between measures of path stability and the effective diffusivity ($r \approx 0.5$), but none were statistically significant ($p > 0.1$). We do, however, observe the inverse relationship between path stability and along-jet speed between 70-65W on interannual time-scales ($r = -0.73$, $p = 0.01$). There may indeed be a dynamical mechanism linking path stability and across-jet mixing in the Gulf Stream, but more data—from observations or eddy-resolving models—is needed to robustly determine the relationship.

Overall, we have shown that the regional variations in the dynamics of eddies has a significant impact on the transport and mixing of tracers at the oceans surface. In some regions, mixing can be parameterised as a simple function of mean-flow strength. However, in other regions the picture is more complex, and parameterisations based solely on mean-flow and EKE may not capture the full mixing effects of mesoscale eddies. This regional variability must be investigated further if we are to understand how distributions of climatic tracers will change in the future, and how to parameterise them accurately within climate models.

Chapter 5

Predicting eddy momentum fluxes with machine learning

5.1 Introduction

The current resolution of Ocean General Circulation Models (OGCM) is not high enough to fully resolve the first baroclinic deformation radius at mid-latitudes (Hallberg, 2013). Due to their finite resolution, they require large viscosity and diffusion coefficients to remain numerically stable (Jochum et al., 2008). The truncation of small-scales and artificially high viscosities smooths out features such as jets and mesoscale eddies (Hewitt et al., 2016; Kjellsson and Zanna, 2017), leading to models that are under-energetic.

Parameterising unresolved eddy momentum fluxes can help re-introduce the spuriously lost energy by viscosity, and generally counter the reduced energy caused by truncating the small-scales of the model. However, as we saw in the previous chapter, mesoscale eddies exhibit significant spatio-temporal variability, and the eddy fluxes cannot always be parameterised as a simple function of the large-scale flow. We, therefore, need new ways of constructing eddy flux parameterisations, that better capture the complex dynamics of unresolved turbulence.

In this chapter, we test whether modern machine learning algorithms can be used to represent eddy momentum fluxes. The concept of machine learning was introduced in Section 3.3, which can be summed up as learning a specific task from data without explicit instruction. We want to use these data-driven algorithms to exploit and leverage data from high-resolution models and observations. Observational datasets

can sometimes be spatially sparse or only contain data for a particular region; it is important to test the impact of such hindrances on data-driven algorithms.

Using data-driven methods to leverage ocean observations is not new. For example, using satellite altimetry data, Keating et al. (2012) constructed a stochastic model to ‘super-resolve’ the velocity field and predict the velocity at depth. Similarly, Keating and Smith (2015) used a stochastic model to produce a super-resolved sea-surface temperature (SST) field, given a low-resolution observation of SST. Chapman and Charantonis (2017) trained a form of neural network, known as a self-organising map, to reconstruct sub-surface velocities in the Southern Ocean using satellite altimetry data and Argo floats. Other studies have used random forests to predict sub-surface temperature anomalies (Su et al., 2018a) and Southern Ocean oxygen content (Giglio et al., 2018).

In this chapter, we use a modern machine learning algorithm which has not yet been utilised within oceanography: convolutional neural networks. They were introduced in Section 3.3.1 and are a form of deep learning that is well suited to extracting information from two-dimensional fields. We aim to test whether convolutional neural networks can be used to reveal information on unresolved turbulent processes, i.e., eddy momentum fluxes. We also want to test how convolutional neural networks respond to situations faced with observational data, such as sparseness in the data. To move towards these aims, as a proof-of-concept, we address the following questions:

1. Can convolutional neural networks represent the spatio-temporal variability of the sub-grid eddy momentum forcing?
2. How sensitive are the neural networks to the physical processes occurring within different regions of the data?
3. How well do they generalise to ocean models in different configurations?
4. Is it possible to physically-constrain neural networks to respect global momentum conservation?
5. How does under-sampling in the training data impact the predictive skill of the neural networks?

The paper is organised as follows. The quasi-geostrophic ocean model, the degrading of model data, and the convolutional neural network, are introduced in Section 5.2. Performance diagnostics of the neural networks, in terms of non-local predictions and generalising to different model configurations, are presented in Section 5.3. The impact of spatial under-sampling in the training data is determined in Section 5.3.3. We explore methods of physically-constraining the neural networks in Section 5.4. We summarise and discuss our results in Section 5.5.

The results of this chapter have been published in the Journal of Advances in Modeling Earth Systems (<https://doi.org/10.1029/2018MS001472>) with Professor Laure Zanna. The published article contains results regarding predicting sub-surface flow fields, using only surface data. These results are not included in this chapter, as they are not relevant to the primary aim of the thesis.

5.2 Data and methods

5.2.1 Eddy momentum forcing

We used data from the high-resolution QG model described in Section 3.1.2. We smooth the fields from the QG model using a spatial 2D low-pass filter, to produce data that is similar to satellite altimetry or a model with a large numerical dissipation. From the filtering of the model data, we can then calculate the forcing from unresolved small-scale turbulent processes. As discussed in Section 3.2, defining the eddy component using a spatial filter produces a more non-local definition of the eddy component compared to coarse-graining. Spatio-temporal filters are also commonly used in Large Eddy Simulations (LES) (Smagorinsky, 1963; Pope, 2001; Sagaut, 2006).

At every time slice in the data, we take a high-resolution variable a at a particular layer, and apply a two-dimensional spatial Gaussian filter. We denote filtered variables as $\bar{a}$, and sub-filter variables as the deviation from the filtered variable $a' = a - \bar{a}$. The value of a function $a(x, y)$, after the Gaussian low-pass filtering operation at a point (x_0, y_0) , is given by

$$\begin{aligned}\bar{a}(x_0, y_0) &= \iint a(x, y) G(x_0, y_0, x, y) dx dy \\ &= \frac{1}{2\pi\sigma^2} \iint a(x, y) e^{-[(x-x_0)^2 + (y-y_0)^2]/2\sigma^2} dx dy,\end{aligned}\tag{5.1}$$

where $\sigma = 30$ km is the standard deviation of the Gaussian filter, which determines the length-scale at which information (below that length-scale) is removed. Therefore the filter acts to remove information on dynamical processes at spatial scales smaller than 30 km.

Using the low-pass filter defined in Equation 5.1, we can now express the effects of the unresolved (sub-filter) variables onto the resolved (filtered) variables.

$$\mathbf{S} = (\bar{\mathbf{u}} \cdot \nabla) \bar{\mathbf{u}} - \overline{(\mathbf{u} \cdot \nabla) \mathbf{u}}, \quad (5.2)$$

which is referred to as the sub-filter eddy momentum forcing (Section 3.2 discusses in more detail how this expression originates). Eddy viscosity models only parameterise one component of Equation 5.2, namely the deviatoric component of the eddy stress tensor (Pope, 2001). The vector $\mathbf{S} = (S_x, S_y)$ represents the effects of the sub-filter momentum field on the filtered momentum field, i.e., the interaction between small-scale eddies and the large-scale flow. As the sub-filter eddy forcing $\mathbf{S}$ depends on the sub-filter variables, it requires parameterising.

5.2.2 Predictive algorithm: convolutional neural networks

Convolutional Neural Networks (CNNs) have proven successful in many areas of computer vision (Krizhevsky et al., 2012; Simonyan and Zisserman, 2014; Dong et al., 2016), where the primary objective is to extract information from an image, to perform a particular task. CNNs work by applying successive layers of convolutions (a form of spatial filtering) to the input; the complexity of the extracted information increases with the number of convolution layers. The powerful property of CNNs is that the filters of each convolution are learnt as part of the training process - they are not specified a priori. Therefore CNNs learn to extract the most ‘useful’ information from the input variable, given training on a particular dataset.

We chose to use CNNs, as opposed to a deep neural network of multiple fully-connected layers, due to their superior performance in computer vision tasks where the inputs have a two-dimensional structure (Krizhevsky et al., 2012). We wanted a machine learning algorithm that could exploit the two-dimensional lateral structure of turbulent fluids. Spatial filtering of the equations of motion of turbulent fluids is not new and is used in Large Eddy Simulation (LES) (Moeng, 1984; Sagaut, 2006).

Therefore, the learnt-filtering operations of a CNN appeared to be a natural choice of data-driven algorithm to apply to geophysical flows.

The training process involves the minimisation of an appropriately defined loss function, which measures the difference between the output of the CNN, and the desired targets. If the optimisation procedure was successful, such that the loss function on previously unseen data converges, the CNN will have learnt to extract the most important information from the input. The CNN then uses the information to predict continuous values. The CNN constructs the final prediction through a linear regression layer, which regresses the desired output onto the final feature maps (feature maps are the intermediate results of each convolution layer).

Here we use CNNs to represent the sub-filter eddy momentum forcing. The input is the filtered-streamfunction $\bar{\psi}$ of the upper vertical layer, which represents our resolved variable that the neural networks will extract information from. The output variables are the zonal S_x and meridional S_y components of the sub-filter momentum forcing $\mathbf{S}$, defined by Equation 5.2. An example input and output is shown in Figure 5.1. Separate CNNs are trained for each component of the sub-filter momentum forcing S_x and S_y . We only consider data from the upper-layer of the model; this is because the flow is surface-intensified, and we are assuming that our filtered quantities are similar to satellite altimetry data, which only provide information at the surface.

In addition to testing whether it is possible to train a neural network to predict S_x and S_y , from $\bar{\psi}$, we explore how a neural network trained on one region performs on another previously unseen region, i.e. how important local vs non-local information is for different regions. Hence, we construct three different datasets from the QG model data, one for each region being studied. We choose regions which differ most in their dynamical behaviour and are shown in Figure 5.1a: Region 1 is near the jet-separation point of the western boundary, where there is a strong, inertial zonal jet. Region 2 is near the eastern boundary downstream of the jet extension, where the dynamics are more wave-like. Region 3 is in the centre of the southern gyre, which is energetically less active than regions 1 and 2.

Data from the three regions are split temporally into training and validation datasets. The 10-years of daily data (3650 days) are split into the first ~ 9 years (3300 days) to train the neural networks, and the final year (350 days) is set aside for

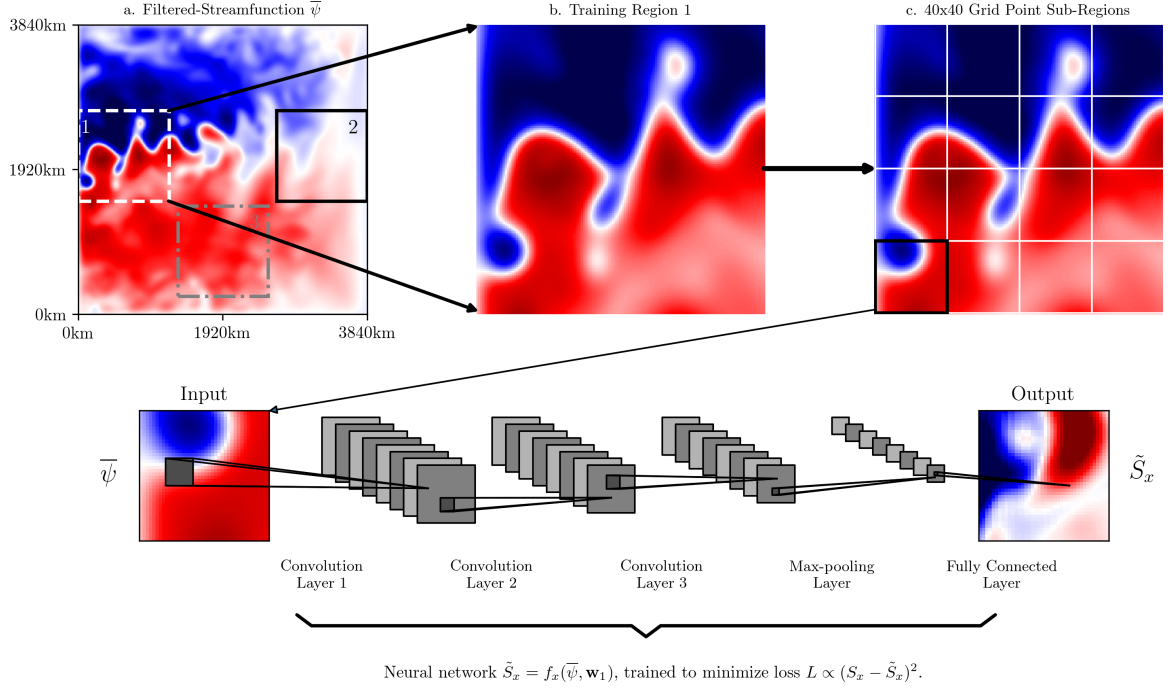


Figure 5.1: Panel (a) illustrates the upper-layer filtered-streamfunction $\bar{\psi}$ of the QG model, including the three regions in which we train the neural networks: region 1 (white-dashed) is on the western boundary, region 2 (black-solid) is on the eastern boundary, and region 3 (grey-dash-dotted) is centred on the southern gyre. Panel (b) shows a close-up of the filtered-streamfunction $\bar{\psi}$ within training region 1 while Panel (c) illustrates how training region 1 is split into 16 40×40 grid point sub-regions - the size of the input and output arrays of the neural network is 40×40 grid points. The input variable of each neural network is the filtered streamfunction $\bar{\psi}$, and the output variable is either the zonal component $\tilde{S}_x$ or meridional component $\tilde{S}_y$ of the sub-filter eddy momentum forcing. The architecture of the convolutional neural network, with an example input $\bar{\psi}$ and output $\tilde{S}_x$, is illustrated underneath Panels (a), (b), and (c).

validation. To reduce the computational cost, and the number of parameters of each CNN, we split each region spatially from the initial 160×160 grid points to sixteen 40×40 grid point sub-regions, as depicted in Figure 5.1c. Reducing the input and output size of the neural network from 160×160 to 40×40 significantly decreases the number of trainable weights, and therefore the computational cost (we attempted to make predictions for the full 160×160 of each training region, but this led to a neural network with over 250,000,000 parameters, which was computationally impractical).

Making predictions for a 40×40 area instead of a 160×160 area also increases the amount of training and validation data by a factor of sixteen, from 3300 and 350 samples to 52800 and 5600 respectively, where a sample is defined as a single input-output pair of the neural network. We end up with 52800 spatial maps (size 40×40 grid points) of input-output pairs to train the neural networks, and 5600 spatial maps of input-output pairs set aside for validation.

We train CNNs to separately predict S_x and S_y , using data from three different regions of the model; this gives a total of 6 neural networks. Each neural network is denoted by $f_i(\bar{\psi}, \mathbf{w}_R)$, where $i = (x, y)$ refers to the component of $\mathbf{S}$ being predicted, $\mathbf{w}_R$ are the trained weights of the neural network, and $R = 1, 2, 3$ refers to the region on which the neural network has been trained. For example, the neural network trained on region 2 to predict the meridional component S_y is denoted by $f_y(\bar{\psi}, \mathbf{w}_2)$.

To distinguish predictions from the true values, we label neural network predictions as $\tilde{S}_x = f_x(\bar{\psi}, \mathbf{w}_R)$, and $\tilde{S}_y = f_y(\bar{\psi}, \mathbf{w}_R)$, while the true values of the sub-filter momentum forcing remain as S_x, S_y . We use the mean-squared error as the loss function,

$$L = \sum (S_x - \tilde{S}_x)^2, \text{ or } \sum (S_y - \tilde{S}_y)^2, \quad (5.3)$$

which quantifies the difference between the neural network predictions and the truth, and where the summation is over all samples. The neural networks are trained (i.e. optimised) using a form of stochastic gradient descent, namely the Adam optimisation algorithm (Kingma and Ba, 2014), which minimises the loss function L defined in Equation 5.3. The training of each neural network $f_i(\bar{\psi}, \mathbf{w}_R)$, iteratively adjusts the values of the weights $\mathbf{w}_R$, such that the loss function in Equation 5.3 is minimised. Therefore each neural network has a different set of weights $\mathbf{w}_R$; it is these weights which determine how each neural network extracts information and makes predictions.

The architecture used for each $f_i(\bar{\psi}, \mathbf{w}_R)$ contains three convolution layers, a max pooling layer, and a final fully-connected layer (Figure 5.1). The max pooling layer reduces the dimensionality of the previous layer, by selecting the maximum value within a 2×2 grid point area - max pooling is effective when there is significant correlation between points in the feature maps. The 2×2 is the kernel size of the max pooling layer and determines the size the area over which to perform the maximum value selection. To give the neural networks the ability to learn non-linear functions, activation functions are added between layers. Here we use the scaled exponential linear unit (SELU) (Klambauer et al., 2017). The SELU activation function is given by

$$\Phi(z) = \lambda \begin{cases} z & \text{if } z > 0 \\ \alpha(e^z - 1) & \text{if } z \leq 0, \end{cases} \quad (5.4)$$

where z is the scalar input value to the activation function, and $\lambda \approx 1.0507$ and $\alpha \approx 11.6732$. SELU activation functions scale the data towards zero mean and unit variance, removing the need for batch normalisation - batch normalisation enforces zero mean and unit variance at each stage of the network but requires additional training.

The specific architecture was constructed by performing a manual systematic search, i.e., adjusting all parameters and observing which configuration most effectively minimises the loss function on the validation data. See Table 5.1 for more details of the architecture and training procedure. The total number of parameters of each neural network is 325,728.

We train and implement each neural network using Keras (Chollet et al., 2015), with the Tensorflow backend (Abadi et al., 2016). Before training, all datasets are separately normalised to zero mean and unit variance. Each CNN is trained for 200 epochs (1 epoch = 1 full pass of all the training data through the optimisation algorithm), taking approximately 10 CPU hours, after which there is negligible change in the loss function of the validation data.

Once all six neural networks are trained, we make the predictions $\tilde{S}_x$ and $\tilde{S}_y$ using the filtered-stream function $\bar{\psi}$ from the validation dataset, i.e., the final year of withheld data. We make predictions for the full-domain to determine how each neural network generalises to unseen, dynamically-distinct, regions. As the input and

Quasi-Geostrophic Model Parameters	
Domain size (grid points)	512×512
Domain length (L)	3840 km
Resolution (Δx)	7.5 km
Viscosity (ν)	75 m ² s ⁻¹
Rossby deformation radii (L_{Ro})	40, 23 km
Velocity scale ($\sqrt{EKE}$)	0.21 ms ⁻¹
Planetary vorticity (f_0)	10 ⁻⁴ s ⁻¹
Rossby parameter (β)	2 * 10 ⁻¹¹ m ⁻¹ s ⁻¹
Gravity (g)	9.8 ms ⁻²
Reduced gravity (g')	0.034, 0.018 ms ⁻²
Bottom drag coefficient (r)	4 * 10 ⁻⁸ s ⁻¹
Wind stress amplitude (τ_0)	0.8 Nm ⁻²
Reference density (ρ_0)	10 ³ kgm ⁻³
Neural Network Data Details	
Data source	Quasi-geostrophic ocean model
Input variable (feature)	Filtered streamfunction $\bar{\psi}$
Output variables (targets)	Sub-filter momentum forcing S_x, S_y
Training Region 1	Western boundary
Training Region 2	Eastern boundary
Training Region 3	Southern gyre
Number of training samples	52800 (years 1-9)
Number of validation samples	5600 (year 10)
Standardisation method	Zero mean, unit variance
Neural Network Architecture	
Input size	40×40
Number of convolution layers	3
Number of filters for each convolution layer	16, 16*8, 8*8
Size of filter for each convolution layer	8×8, 4×4, 4×4
Filter stride for each convolution layer	2, 1, 1
Activation function for each convolution layer	SELU, SELU, SELU
Max pooling kernel size	2
Output layer activation function	None/Linear
Output size	40×40
Neural Network Training Parameters	
Loss function	Mean-square error
Optimiser	Adam
Learning rate	0.001
Momentum	0.9
Batch size	16
Training epochs	200

Table 5.1: Details on the following: the quasi-geostrophic ocean model parameters, the datasets used to train the neural networks, the architecture parameters, and the optimisation parameters.

output size of each neural network is 40×40 grid points, we tile together predictions for the full domain of size 512×512 ; the tiling leads to errors at the boundaries of each tile, where discontinuities can emerge. To reduce the tiling error, we make predictions using overlapping tiles, and then average the results at each grid point.

5.3 Neural network generalisation and sensitivity

5.3.1 Non-local predictions

The filtered streamfunction represents, for example, observational measurements from satellite altimetry or coarse-resolution model data. The sub-filter eddy momentum forcing represents unresolved turbulent processes. Our goal is to replicate the complex spatio-temporal variability of S_x and S_y using neural networks $f_i(\bar{\psi}, \mathbf{w}_R)$. However observational data such as moorings (Hogg, 1992) or gliders (Rudnick et al., 2004; Davis et al., 2008), may only be available for a particular region; we therefore only train the neural networks using data from specific regions of the full domain, as described in Section 5.2.2. Our aims are to both successfully train the neural networks, and to study how they generalise to previously un-seen regions.

We study the spatio-temporal variability of S_x and $\tilde{S}_x$ by examining snapshots of the time-mean and the standard deviation, shown in Figure 5.2. Diagnostics are calculated over the full 512×512 domain, using the final year of withheld data. Both the spatial and temporal variability of the true S_x are dominated by the jet dynamics (Figure 5.2a, e, and i). In particular, strong meanders which extend eastward from the western boundary are visible. The amplitude of the spatio-temporal variability of S_x ($1.4 \times 10^{-6} \text{ms}^{-2}$) is of similar magnitude to the time-mean ($1.5 \times 10^{-6} \text{ms}^{-2}$).

All neural networks trained on three different regions, shown in Figure 5.1a and described in Section 5.2.2, successfully reproduce the spatial patterns of the true S_x , as shown by snapshots of the predictions $\tilde{S}_x$ (Figure 5.2b, c, and d). Their magnitudes, however, vary significantly. The predictions of $f_x(\bar{\psi}, \mathbf{w}_1)$, trained on data from the western boundary, are almost identical to the true S_x , and successfully reproduces the correct amplitude and variability (Figure 5.2b, f, j). The neural network $f_x(\bar{\psi}, \mathbf{w}_2)$, trained on data from the eastern boundary, underestimates the magnitude of the true S_x by approximately 50%, despite reproducing the correct spatial patterns. The

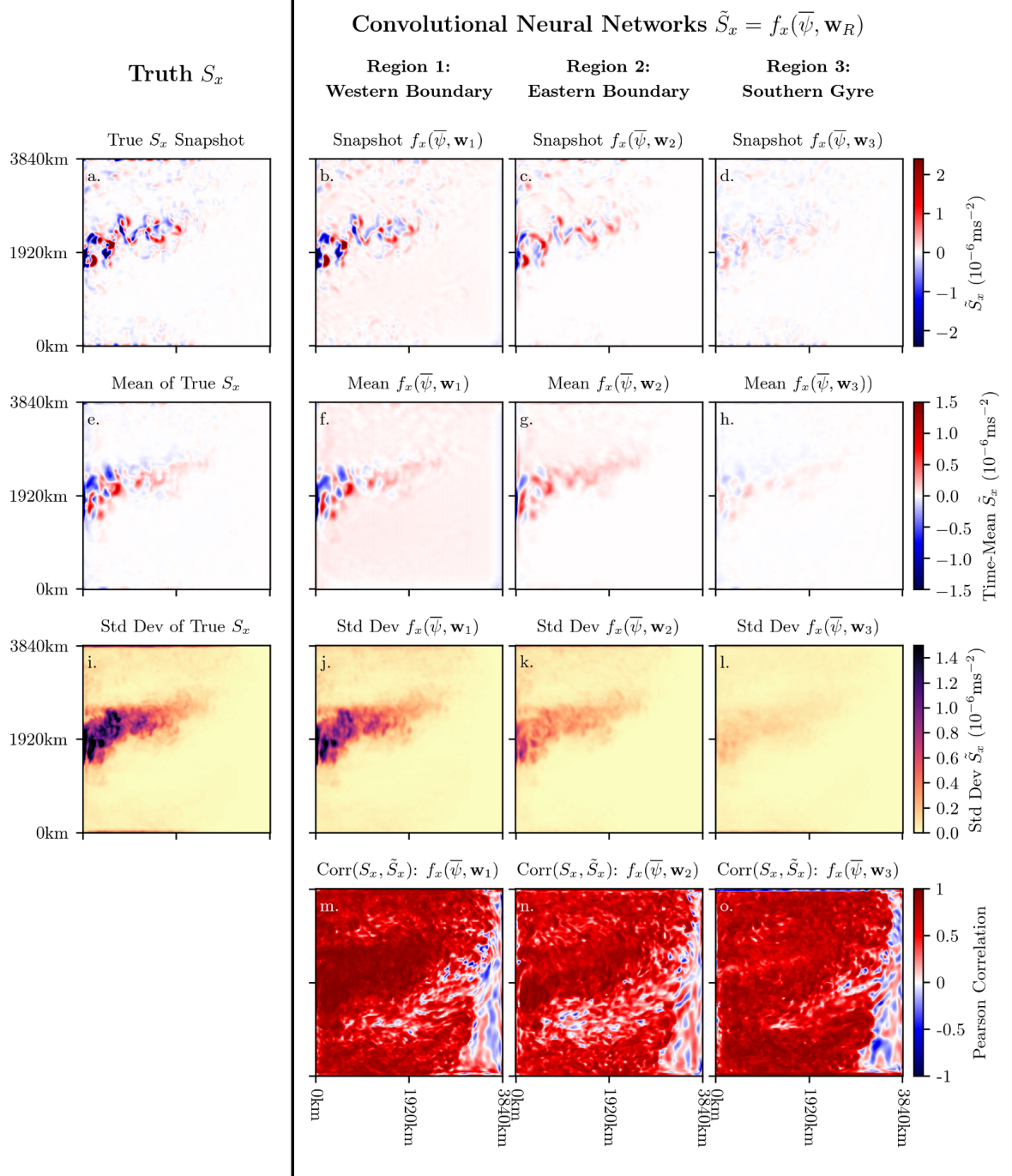


Figure 5.2: Examining the non-local prediction ability. Comparisons of the true zonal component of the sub-filter momentum forcing S_x , with the neural networks trained using data from three different regions. The first three rows compare snapshots, time-means, and the standard deviation respectively, while the bottom row shows the correlation between the true S_x and the predictions $\tilde{S}_x$. The first column contains the diagnostics using the true zonal sub-filter momentum forcing S_x , while columns two, three, and four use predictions $\tilde{S}_x$ from the neural networks $f_x(\bar{\psi}, \mathbf{w}_1)$, $f_x(\bar{\psi}, \mathbf{w}_2)$, and $f_x(\bar{\psi}, \mathbf{w}_3)$ respectively. All diagnostics were produced using the validation data.

predictions of $f_x(\bar{\psi}, \mathbf{w}_3)$, trained on the southern gyre, underestimates the true S_x by an order of magnitude (Figure 5.2d, h, l).

As the variability of S_x is dominated by the jet, it is difficult to assess the accuracy of the neural network predictions $\tilde{S}_x$ in quiescent regions such as the eastern boundary or within the gyres. We therefore calculate the Pearson correlation, a dimensionless quantity, between the true S_x and the predictions $\tilde{S}_x$. It should be noted that high-correlations do not guarantee good performance during implementation. The predictions of $f_x(\bar{\psi}, \mathbf{w}_1)$ and $f_x(\bar{\psi}, \mathbf{w}_2)$ are highly correlated with the truth ($r > 0.9$) within the jet, but tend towards zero or negative correlation near the eastern boundary (Figure 5.2m and 5.2n). The predictions of $f_x(\bar{\psi}, \mathbf{w}_3)$ have a more consistent positive correlation across the gyres and other more quiescent regions, (Figure 5.2o).

We observe similar results for the spatial and temporal variability of S_y , shown in Figure 5.3: the variability within the jet dominates, with an amplitude ($1 \times 10^{-6} \text{ms}^{-2}$) similar to S_x . The meandering of the jet again produces complex spatial patterns in S_y , which when averaged in time, produce a distinct sign change moving across the jet latitudinally. For the predictions $\tilde{S}_y$, the neural network trained on the western boundary, $f_y(\bar{\psi}, \mathbf{w}_1)$, most effectively reproduces the true S_y . However, the time-mean of $f_y(\bar{\psi}, \mathbf{w}_1)$ (Figure 5.3f) has a positive bias everywhere in the domain, whereas the time-means of $f_y(\bar{\psi}, \mathbf{w}_2)$ and $f_y(\bar{\psi}, \mathbf{w}_3)$ (Figure 5.3g and 5.3h respectively) do not.

The correlations between S_y and $\tilde{S}_y$ are similar to the zonal component: $f_y(\bar{\psi}, \mathbf{w}_1)$ and $f_y(\bar{\psi}, \mathbf{w}_2)$ are highly correlated ($r > 0.8$) within the jet, but not in the gyres. In contrast, $f_y(\bar{\psi}, \mathbf{w}_3)$ has a consistently positive correlation across the full domain, despite failing to reproduce the amplitude within the jet. In fact, the correlation of $f_y(\bar{\psi}, \mathbf{w}_3)$ within the jet (Figure 5.3o) is negative ($r \approx -0.3$). The negative correlation implies that the dynamical processes occurring within region 3, the southern gyre, have an opposite effect to the eddy momentum forcing occurring within region 1. The opposing effects of eddies could be an example of regional variation in the eddy forcing, as in Waterman and Jayne (2010), who found that whether eddies were driving the large-scale flow or not, depended critically on along-stream position.

Across all neural networks, the correlation decreases at the eastern boundary, which is partly caused by the sub-filter momentum forcing being orders of magnitude

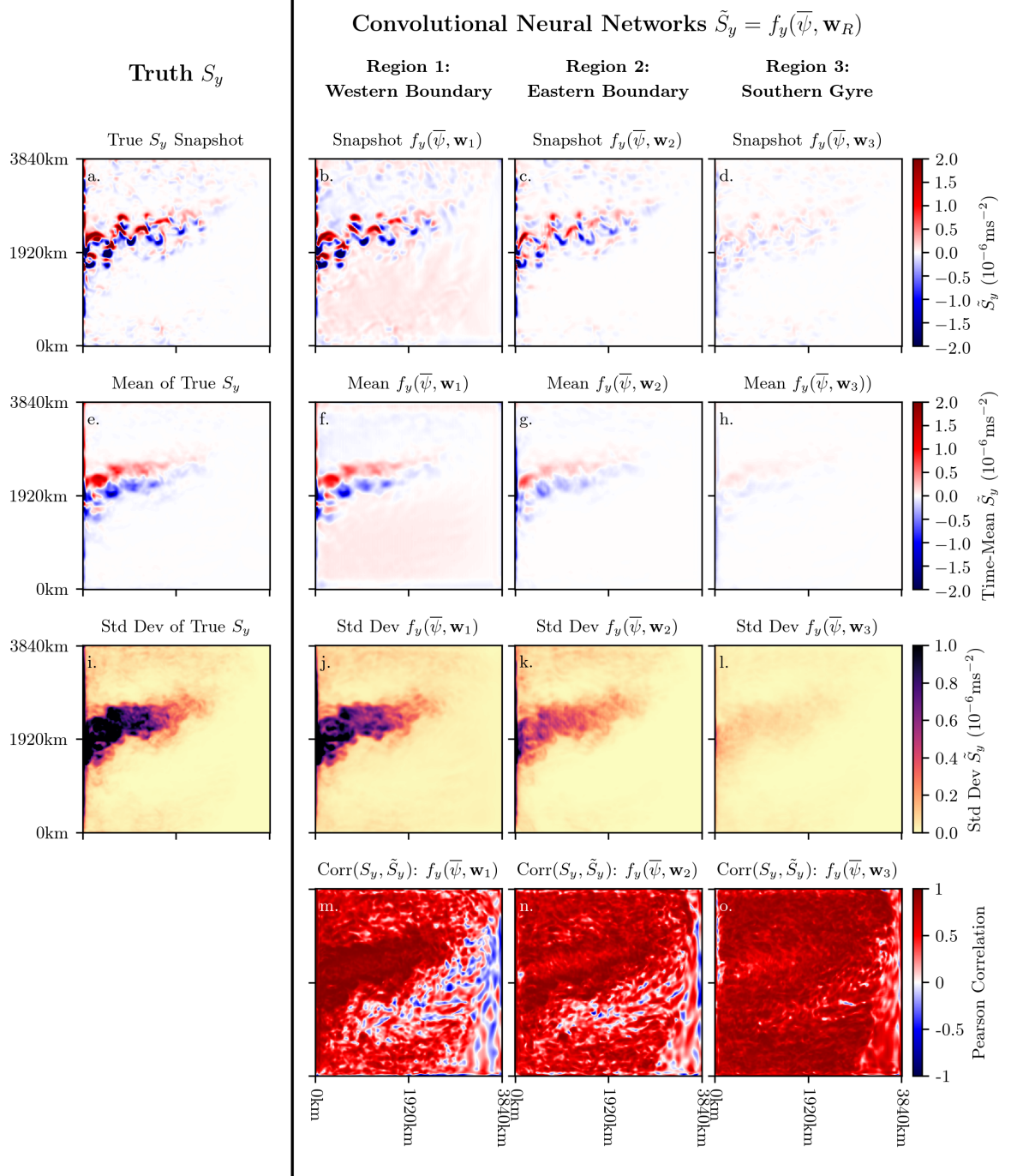


Figure 5.3: The same diagnostics as Figure 5.2, but for the meridional component of the sub-filter momentum forcing: the true S_y and the predictions $\tilde{S}_y$ from the neural networks $f_y(\bar{\psi}, \mathbf{w}_1)$, $f_y(\bar{\psi}, \mathbf{w}_2)$, and $f_y(\bar{\psi}, \mathbf{w}_3)$.

lower than elsewhere in the domain. The low magnitude of S_x and S_y is due to the wave-like behaviour of the flow having a larger spatial-scale. The larger spatial-scale at the eastern boundary leads to little variability at small scales, reducing the eddy momentum forcing to almost zero, and therefore causing the performance of neural networks to deteriorate.

Overall, we see that training neural networks on the western boundary is most successful when generalising to other areas of the domain (in terms of correlations and reproducing the variability). Training on the eastern boundary produced good correlations in the western boundary but underestimated the magnitude of the eddy forcing by approximately 50%. Training on the southern gyre did not correlate well within the western boundary and underestimated the truth by an order of magnitude.

Hence to successfully reproduce the correct amplitude and variability across the domain, the training data must contain a diverse range of scale interactions, which here corresponds to training on the most turbulent region. However, training on the turbulent regions can lead to significant net biases in the predictions, as seen in Figure 5.3f. How to correct for such biases will be discussed in Section 5.4.

It should be noted that we did train additional neural networks by forming training data from all three regions. The performance of the resulting neural networks appeared to be an 'average' of the results of each region, i.e., the overall predictive skill was not as good as training on the western boundary, but better than training on the eastern or southern regions. Therefore, training on all three regions at once prevented the neural network from specialising/catering to the dynamics of any one region, and learns an 'average' of the dynamics from each region.

5.3.2 Generalising to different Reynolds numbers

In Section 5.3.1, we investigated how neural networks trained on different regions of the domain generalise to other previously unseen regions. We now test how the neural networks generalise to different regimes, in particular, different Reynolds numbers. In Section 5.3.1, we found that the neural networks trained on region 1, the western boundary, successfully generalised to different regions; we therefore apply $f_x(\bar{\psi}, \mathbf{w}_1)$ to new model data with different wind stress amplitudes and viscosity coefficients to test its performance. We use models with higher and lower wind forcings, to test

regimes which are both more and less turbulent than the original model, which had a wind stress amplitude of $\tau_0 = 0.8 \text{ Nm}^{-2}$ and viscosity $\nu = 75 \text{ m}^2\text{s}^{-2}$.

We use the low-pass on filter the upper-layer streamfunction from each different model run, with the following: $\nu = 200 \text{ m}^2\text{s}^{-2}$, and $\tau_0 = 0.3, 0.6$, and 0.9 Nm^{-2} , and then apply the already-trained neural network $f_x(\bar{\psi}, \mathbf{w}_1)$ to generate predictions $\tilde{S}_x$. The standard deviation of the true S_x , the standard deviation of the $f_x(\bar{\psi}, \mathbf{w}_1)$ predictions $\tilde{S}_x$, and the correlation between them, are shown in Figure 5.4.

The neural network $f_x(\bar{\psi}, \mathbf{w}_1)$ reproduces the variability within the jet almost exactly, across all runs, as can be seen by comparing the standard deviations in the first and second columns, which represent the standard deviation of the true S_x and predicted $\tilde{S}_x$ respectively. The correlation within the jet remains high ($r > 0.9$) in all runs, including the model with an increased wind forcing ($\tau_0 = 0.9 \text{ Nm}^{-2}$) in Figure 5.4o. The correlations weaken at the eastern boundary for the lowest wind forcing ($\tau_0 = 0.3 \text{ Nm}^{-2}$), shown in Figure 5.4f; this may be caused by an increase in the wave-like behaviour at the eastern boundary, which is not well captured by the neural networks. In general, the higher the Reynolds number, the better the correlations, i.e., more dark red areas of $r > 0.8$.

The mean biases of the predictions of the new models are similar in magnitude to the biases of the original model configuration. These biases showed no relationship with the Reynolds number and are not discussed further.

5.3.3 Sensitivity of neural networks to under-sampling

We have so far trained the neural networks with densely sampled data, i.e., we have data at each grid point for both the input and output variables. However, most observational datasets are spatially sparse, e.g. Argo floats (Roemmich et al., 2009). We explore the impact of under-sampling with a new collection of neural networks trained on region 1 to predict S_x , but with the training data sub-sampled. At each time-slice of the training data, we randomly sample (without replacement) N points of the 40×40 input variables, $\bar{\psi}$, and output variables S_x . Using these N randomly sampled values, we use a cubic interpolation to reconstruct the full 40×40 grid point input and output (with a nearest-neighbour interpolation for grid points that fall outside the convex hull of the cubic interpolation).

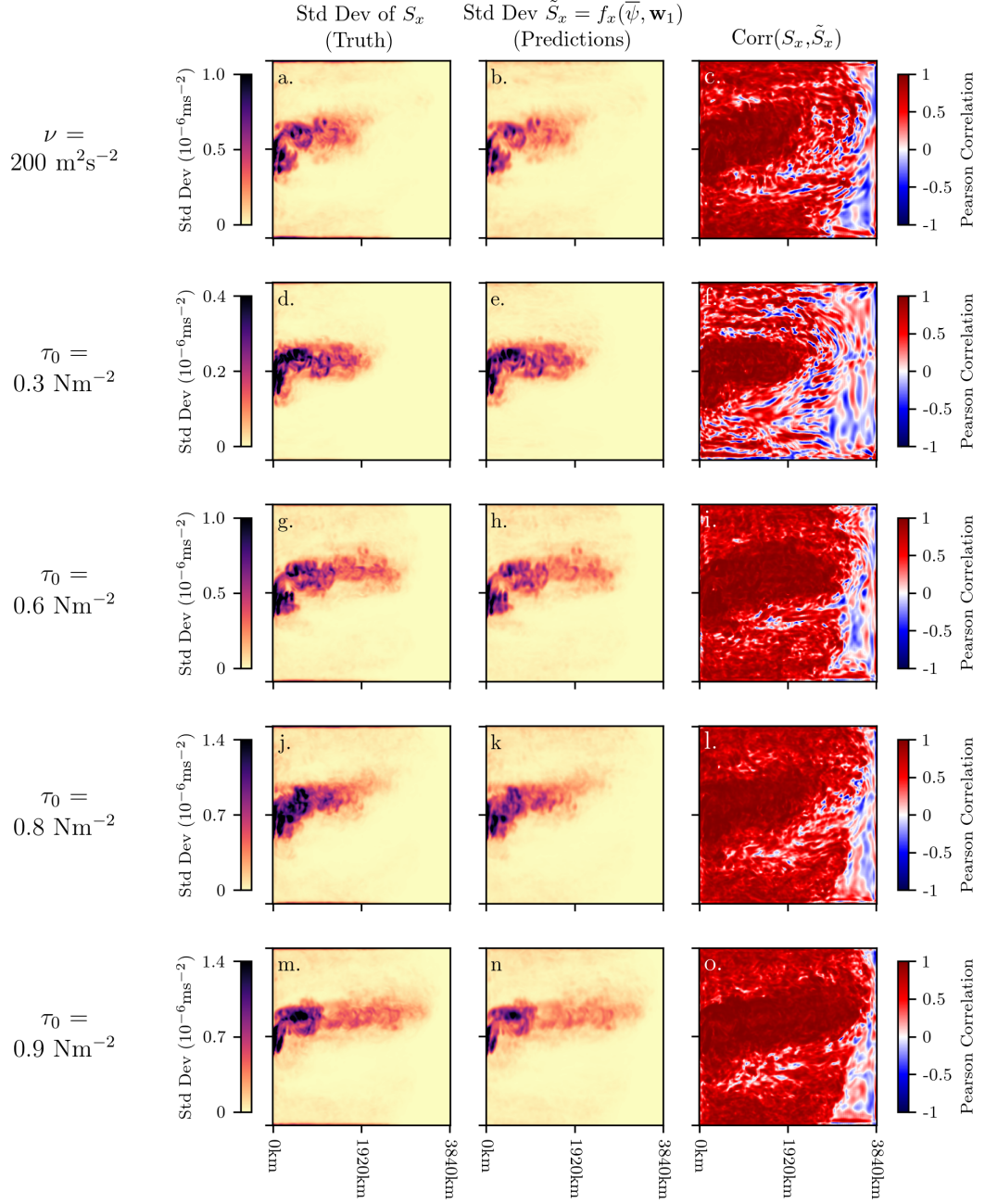


Figure 5.4: Examining the ability to generalise to new regimes: using the trained neural network $f_x(\bar{\psi}, \mathbf{w}_1)$, we make predictions for model runs of different viscosities and wind forcings. From each model run, we use one year of the upper-layer filtered streamfunction to generate predictions $\tilde{S}_x$ from $f_x(\bar{\psi}, \mathbf{w}_1)$ to see how they compare to the true S_x . We study a run of higher viscosity $\nu = 200 \text{ m}^2\text{s}^{-2}$, and runs with wind stress amplitude $\tau_0 = 0.3, 0.6, 0.8$, and 0.9 Nm^{-2} . Note that $f_x(\bar{\psi}, \mathbf{w}_1)$ was trained on a run with $\nu = 75 \text{ m}^2\text{s}^{-2}$ and $\tau_0 = 0.8 \text{ Nm}^{-2}$, the standard deviation and correlation maps of which are included again here in Panels (j), (k), and (l).

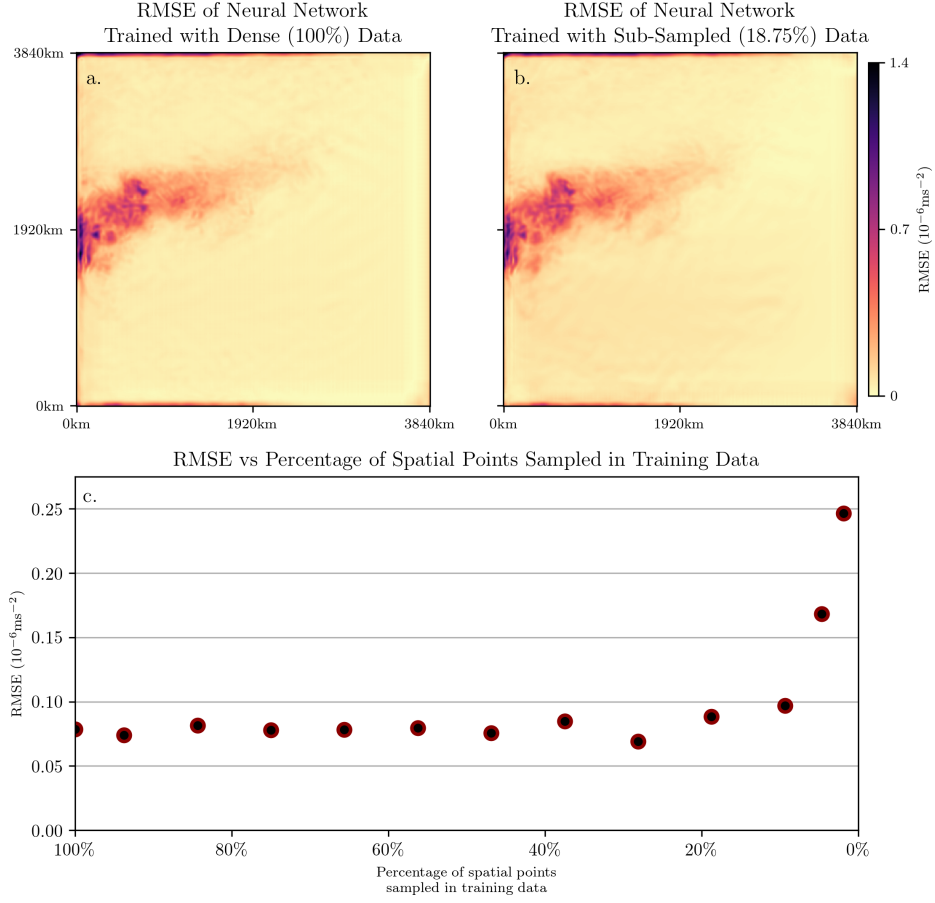


Figure 5.5: Determining how under-sampling of the training data impacts neural network error. Panel (a) shows the RMSE of the neural network $f_x(\bar{\psi}, \mathbf{w}_1)$ trained with dense (un-altered) training data, while Panel (b) shows the RMSE of the neural network trained with sub-sampled (18.75%) data. Panel (c) shows the RMSE as a function of the percentage of spatial points sampled at each time-slice of the training data. Note that the RMSE is calculated over the full-domain during the validation period (the final year of data).

These reconstructed time-slices from sub-sampled data are used to train a new set of neural networks. We vary the number of points N sub-sampled from $> 90\%$ to $< 5\%$ of the original 1600 points of the input and output variables. We have a neural network for each value of N , the sub-sampling rate. Using the neural networks trained on under-sampled data, we calculate the root-mean-square error (RMSE) on the final year of validation data over the entire domain. The validation data is not sub-sampled, providing a stronger and more accurate test of the neural network's performance.

The RMSE is shown as a function of the percentage of points sampled (Fig-

ure 5.5c). We find that the RMSE increases significantly only when the percentage of spatial points sampled drops below 10% (the error doubles at a sub-sampling rate of 4.7%). Note that the RMSE is not a monotonic function of the percentage of points sampled due to the stochastic nature of the training procedure and the use of a non-linear interpolation. The spatial map of RMSE of the neural network trained with 18.75% sub-sampled data (Figure 5.5b) shows minimal changes relative to the neural network trained on the original (unaltered) training data (Figure 5.5a). The result further suggests that the use of sparse interpolated observations can be successfully used to accurately train and predict the eddy momentum forcing as shown in Sections 5.3.1 and 5.3.2.

We also tested an alternative method of under-sampling, where the 40×40 input and output grid of the neural network is spaced out over the entire domain. In other words, we sub-sample the input and output variables of the original 512×512 grid to a regularly spaced 40×40 grid. However, training a convolutional neural network with this methodology did not work and led to severe overfitting (i.e. increasing validation loss during training). The neural networks presented in Section 5.3.1 learn to take first- and second-order derivatives of the input streamfunction (see GitHub repository), which correspond to the velocities and velocity shears. Both velocities and velocity shears are important features to provide for accurate predictions of the eddy momentum forcing. By severely sub-sampling the input streamfunction, the local information relevant to estimate velocities and velocity shears is lost.

Therefore, the success of training convolutional neural networks on observational datasets will likely depend on the horizontal sampling rate of the product being considered. Our results suggest that high horizontal sampling is necessary to capture any small-scale gradients and avoid the sharp rise in error in Figure 5.5. We anticipate that the use of (interpolated) datasets with a horizontal resolution of 1 degree or less (e.g., Argo float, altimetry) is needed to train the neural networks and predict the eddy momentum forcing.

Note that for Argo floats, the spatial sampling will not be random, as the drift patterns will produce quasi-Lagrangian structures in the data. We have not considered Lagrangian spatial patterns in the sampling here, but this will also impact how neural networks learn from observational data.

5.4 Physically-constrained neural networks

We proceed to examine the net input of momentum from the neural network predictions $\tilde{S}_x$ and $\tilde{S}_y$, which should vanish. If neural networks are used to leverage the use of observational datasets and coarse-resolution models, then spurious sources of momentum would violate physical conservation laws. Thus, we need to constrain the neural networks to respect the physical properties of the system. Here we diagnose the momentum biases of the neural networks $f_i(\bar{\psi}, \mathbf{w}_R)$, and then explore different methods of imposing conservation of momentum globally.

5.4.1 Momentum biases

Each sub-region (including those used to train the neural networks) may have a non-zero spatially-integrated momentum tendency. However, globally, the true sub-filter momentum forcing $\mathbf{S}$ should re-distribute momentum, and not act as a source or sink, i.e. $\iint \mathbf{S} dx dy = 0$. We require that the neural networks do not introduce spurious sources of momentum, to respect the physical properties of the system. By training each neural network on a sub-region, we expect to have imperfect momentum conservation, which will depend upon the particular dynamical processes within each region. For example, if eddies within a particular region are driving the mean-flow, then we would expect a positive source of momentum locally - a neural network trained on such a region would likely generalise the (local) input of momentum to the rest of the domain. A net source or sink of momentum will manifest as a non-zero bias after spatial averaging.

At a single point in space, the time series of the predictions $\tilde{S}_x$ and $\tilde{S}_y$ show that the neural networks trained on regions 1 and 2 track the true S_x and S_y closely (Figure 5.6a and 5.6b), reproducing a significant proportion ($> 80\%$) of the variance. However, if at each time-step we spatially average the neural network predictions $\tilde{S}_x$ and $\tilde{S}_y$ (Figure 5.6c and 5.6d respectively) over the full domain, we observe significant non-zero biases.

Consider the zonal component of the eddy momentum forcing in Figure 5.6c: $f_x(\bar{\psi}, \mathbf{w}_1)$ has a net positive bias, implying a global positive increase of zonal momentum at all times, while both $f_x(\bar{\psi}, \mathbf{w}_2)$ and $f_x(\bar{\psi}, \mathbf{w}_3)$ have negative biases, indicating a net decrease in zonal momentum. We can estimate the magnitude of the resulting

change in zonal velocity from these net biases, over a period of a year, by assuming $\Delta u = \langle \tilde{S}_x \rangle \Delta t$, where $\langle \rangle$ denotes the spatial average over the full domain. For $f_x(\bar{\psi}, \mathbf{w}_1)$, $f_x(\bar{\psi}, \mathbf{w}_2)$, and $f_x(\bar{\psi}, \mathbf{w}_3)$, we obtain values of $\langle \tilde{S}_x \rangle = 0.03$, 0.02 , and 0.0008 (10^{-6}ms^{-2}) respectively; this leads to zonal velocity changes of $\Delta u = 0.95$, 0.63 , and 0.025 (ms^{-1}). These changes are of similar magnitude to the time-mean zonal flow, which peaks at approximately 0.9ms^{-1} within the jet core.

There are also significant biases in the predictions of the meridional component $\tilde{S}_y$, shown in Figure 5.6d. The positive bias of $f_y(\bar{\psi}, \mathbf{w}_1)$ is visible in the time-mean $\tilde{S}_y$ shown in Figure 5.3f. We can again estimate the change in meridional velocities by assuming $\Delta v = \langle \tilde{S}_y \rangle \Delta t$. Using values of $\langle \tilde{S}_y \rangle = 0.02$, -0.01 , and 0.002 (10^{-6}ms^{-2}) for $f_y(\bar{\psi}, \mathbf{w}_1)$, $f_y(\bar{\psi}, \mathbf{w}_2)$, and $f_y(\bar{\psi}, \mathbf{w}_3)$ respectively, leads to the following changes: $\Delta v = 0.63$, -0.31 , and 0.06 (ms^{-1}). Some of these changes are the same magnitude as the time-mean meridional flow.

5.4.2 Towards momentum-conserving neural networks

The predictions of neural networks $f_x(\bar{\psi}, \mathbf{w}_1)$ and $f_y(\bar{\psi}, \mathbf{w}_1)$, described in Section 5.3.1, correctly reproduce the correct amplitude and variability of the true eddy momentum forcing S_x and S_y , as seen in Figures 5.2 and 5.3. However, training on region 1 also produced some of the largest non-zero biases in $\tilde{S}_x$ and $\tilde{S}_y$ after spatial averaging at each time step. We test whether we can reduce the biases when training on region 1 while preserving the accuracy of predictions from the neural network. We trial three approaches (A, B, and C) to reduce the biases identified in Figure 5.6c and 5.6d.

- (A) Architecture Alteration: Train neural networks on region 1, but with the final fully-connected layer modified such that the spatial mean is removed from the final output. The neural networks will, therefore, be trained to reproduce the sub-filter momentum forcing, but with momentum conservation intrinsically embedded. I.e. the same training data, but altered architecture. The motivation behind this approach is that if the local source of momentum within the 40×40 output grid is zero, then this may reduce the global net source of momentum.

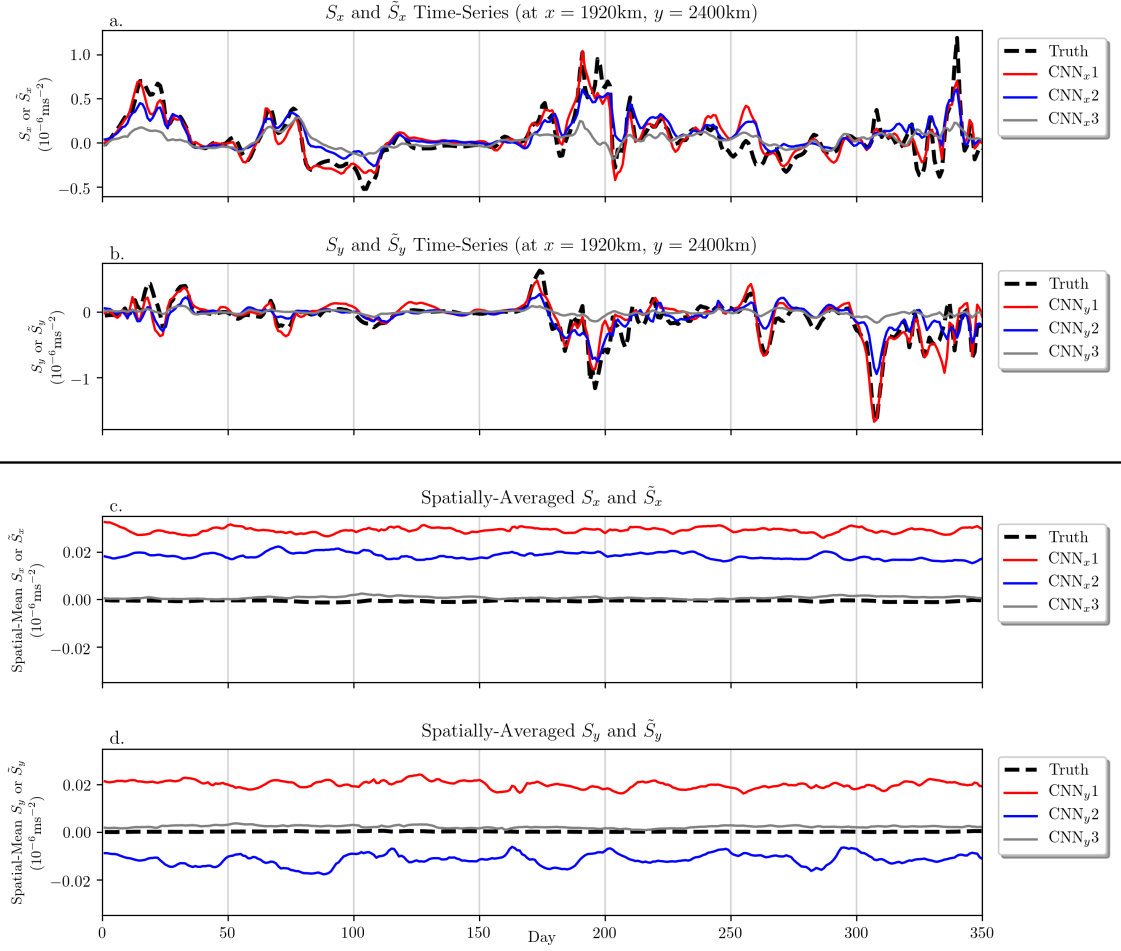


Figure 5.6: Panels (a) and (b) show time series of the zonal and meridional components of the sub-filter momentum forcing respectively, at a single point near the middle of the domain. Panels (c) and (d) also show time series of the zonal and meridional components of the sub-filter momentum forcing, but this time spatially-averaged over the entire domain.

- (B) Pre-processing of input: Train on region 1 with the original architecture described in Table 5.1 but with the spatial-mean removed from each 40×40 training snapshot of S_x . In other words, remove the local bias from each training snapshot. If the local source of momentum of each 40×40 sample is zero within the training data, then training on such data may reduce local biases when making future predictions on unseen data. However, does not guarantee that subsequent predictions will have zero local bias. This approach tries to increase the probability of *learning* local momentum conservation.
- (C) Post-processing of output: train on region 1, and enforce global momentum conservation after the predictions have been made. I.e. no changes to training data or architecture, but with additional processing of the full-domain predictions $\tilde{S}_x$ and $\tilde{S}_y$. This simply involves removing the spatial-mean after the full prediction for the 2D domain has been formed (and for each time slice).

The associated neural networks of each approach are labelled as $f_i(\bar{\psi}, \mathbf{w}_1^A)$, $f_i(\bar{\psi}, \mathbf{w}_1^B)$, and $f_i(\bar{\psi}, \mathbf{w}_1^C)$ respectively, where $i = (x, y)$ denotes either the zonal S_x or meridional S_y component being predicted.

All neural networks are optimised using the same training parameters given in Table 5.1. Approach A, which alters the architecture, and approach B, which alters the training data, are enforcing momentum conservation not just globally, but within the 40×40 sub-region being predicted. This local conservation is useful for enforcing global conservation. However, local conservation may not be desirable if there is a convergence of eddy momentum fluxes in a particular region, which can impact the large-scale flow, e.g. if eddies are fluxing momentum into the jet at a particular along-stream position, enforcing local conservation in a neural network may lead to missing these effects. Hence, caution must be taken with restricting architectures in this way.

We now explore the performance of the newly constrained neural networks and the net momentum input relative to that of the original neural networks trained on region 1: $f_x(\bar{\psi}, \mathbf{w}_1)$ and $f_y(\bar{\psi}, \mathbf{w}_1)$. The spatial-averages of neural networks based on approaches A, B, and C are shown in Figure 5.7, with the same scale axes as in Figure 5.6.

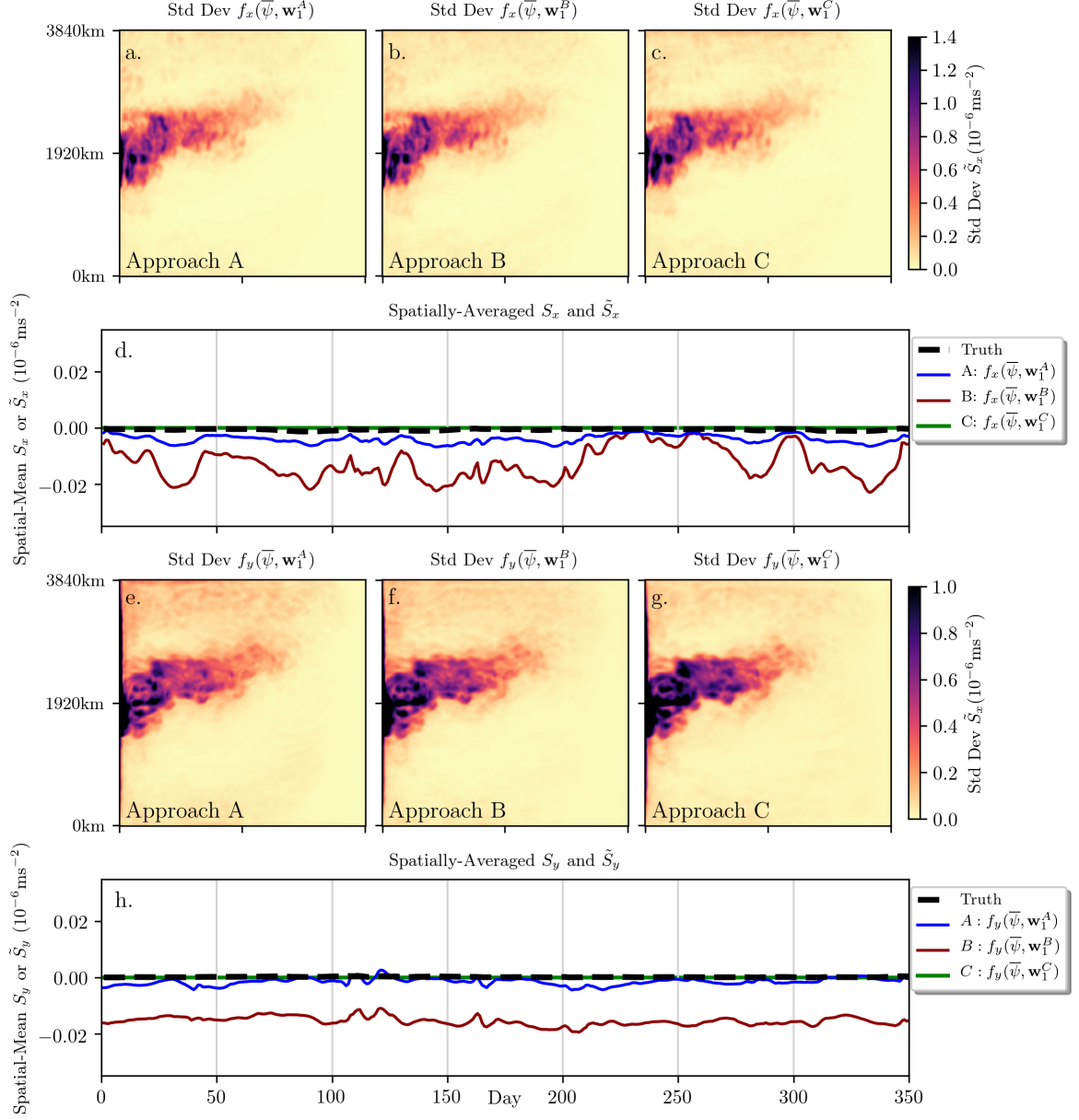


Figure 5.7: The standard deviation and spatial-average time series of the predictions $\tilde{S}_x$ and $\tilde{S}_y$ of the momentum converging approaches A, B, and C. Panels (a), (b), and (c) show the standard deviation of $\tilde{S}_x$ from $f_x(\bar{\psi}, \mathbf{w}_1^A)$, $f_x(\bar{\psi}, \mathbf{w}_1^B)$, and $f_x(\bar{\psi}, \mathbf{w}_1^C)$ respectively, while Panels (e), (f), and (g) show the standard deviation of $\tilde{S}_y$ from $f_y(\bar{\psi}, \mathbf{w}_1^A)$, $f_y(\bar{\psi}, \mathbf{w}_1^B)$, and $f_y(\bar{\psi}, \mathbf{w}_1^C)$ respectively. The spatial-averages of these predictions $\tilde{S}_x$ and $\tilde{S}_y$ are shown in Panels (d) and (h).

Approach B has significant biases of approximately -0.01 and -0.015 (10^{-6}ms^{-2}) in the zonal and meridional components respectively; the optimisation procedure aims to reproduce the *variability* in the training data, and not spatial-means, therefore pre-processing the training data does not remove the biases. Compared to the original neural networks trained on region 1, the biases of approaches A and C are 3 to 5 orders of magnitude lower, in both the zonal and meridional components. The bias for the post-processing approach is exactly zero by construction, while the altered-architecture approach A is not exactly zero due to the overlapping-tiling procedure. The biases of $f_x(\bar{\psi}, \mathbf{w}_1^A)$ and $f_y(\bar{\psi}, \mathbf{w}_1^A)$ are approximately -0.002 and -0.0005 (10^{-6}ms^{-2}) which, over the course of a year, would lead to velocity changes of $\Delta u = -0.06$ and $\Delta v = -0.01$ (ms^{-1}) respectively - now an order of magnitude smaller than the time-mean flow.

The correlation maps of all momentum-conserving approaches (not shown) change little from the original correlation maps of $f_x(\bar{\psi}, \mathbf{w}_1)$ and $f_y(\bar{\psi}, \mathbf{w}_1)$, shown in Figure 5.2m and 5.3m respectively. All approaches reproduce the correct spatial patterns of the true S_x and S_y (e.g., Figure 5.7 for standard deviations). However, approaches A and B underestimate the amplitude of S_x and S_y by approximately 20-30%, whereas there is a little difference between approach C and the truth ($< 10\%$).

In summary, approach C of post-processing successfully enforces momentum conservation, without sacrificing accuracy in the predictions of the eddy momentum forcing. Approach B, altering the training data, was not efficacious at reducing the net biases. The physically-constrained architecture of approach A successfully reduced the net bias, but at the expense of 20-30% accuracy. Though further altering of the architecture (e.g. increasing number of convolution layers and filters) or training procedure (decreasing the learning rate, with an increased number of training epochs) could reduce this drop in accuracy by countering the restriction placed on the architecture.

5.5 Summary and discussion

5.5.1 Summary

We have demonstrated, as a proof-of-concept, that machine learning algorithms can provide information on unresolved turbulent processes when given a smoothed-view

of the dynamics (i.e. the filtered-streamfunction). We smooth data from a high-resolution eddy-resolving QG model using a spatial low-pass filter, and train convolutional neural networks to predict the relationship between turbulent processes and their effect on the large-scale flow, i.e. the eddy momentum forcing. Our results show that convolutional neural networks can successfully represent both the spatial and temporal variability of the eddy momentum forcing.

We determine how neural networks trained on one area of the domain, perform in other previously-unseen areas (Figures 5.2 and 5.3), representing when observational data is limited to only particular regions, for example, mooring data (Hogg, 1992) or gliders (Rudnick et al., 2004; Davis et al., 2008). Training on a sub-region tests the sensitivity of the neural network performance to the underlying physical processes. We find that the region on which the neural network is trained significantly impacts the accuracy, as well as the mean-bias which impacts momentum conservation. In particular, training on the least energetically active region, the southern gyre, leads to the lowest accuracy; these neural networks could not reproduce the variability in more energetic regions, such as within the meandering jet. However, training on the western boundary leads to the best generalisation, in terms of reproducing the correct amplitude of the eddy momentum forcing in the rest of the domain.

The variation in performance between regions implies that training on the most turbulent region leads to the best performing neural networks for eddy momentum forcing prediction. It is possible that data from the most turbulent regions exhibit the highest variance, or contains a more diverse range of scale-interactions. However, two regions may be as turbulent or energetically active as each other, but the nature of the eddy-mean flow interactions within them may differ. For example, Waterman and Jayne (2010) showed that in an idealised model the effect of eddies on the mean-flow depended critically on along-stream position: up-stream eddies are generated by an unstable jet, while down-stream the eddies drive the time-mean circulation. Therefore training neural networks on different along-stream positions may lead to different dynamical-processes being learnt, despite both regions being energetically active.

Without further training, we show that a neural network trained on one QG model configuration generalises exceedingly well to QG models with different viscosity coef-

ficients and wind forcings (Figure 5.4). The neural network within the jet reproduces the correct spatio-temporal variability ($<10\%$ error) in all configurations, and the correlation between the predicted $\tilde{S}_x$ and the true S_x within the gyres increases with the Reynolds number of the model configuration. While the neural networks do not conserve momentum globally (Figure 5.6c and 5.6d), we show that momentum conservation can be enforced without a significant reduction in accuracy (Figure 5.7), through either a physically-constrained architecture or post-processing of the predictions.

5.5.2 Implications for leveraging observations

Our work has implications for inference from sparse observations. While previous studies have used machine learning to leverage observational datasets (Chapman and Charantonis, 2017; Su et al., 2018a; Giglio et al., 2018), the present work demonstrates that convolutional neural networks in particular are an excellent tool for such tasks. Neural networks should be further tested and exploited in the future for data inference due to

- their resilience, such that accurate predictions for the full domain can be generated by training on a sub-region;
- their generalisation to different external forcings, without any further training such that predictions outside the regime trained on can be successful;
- their ability to be successfully trained with under-sampled data. (Figure 5.5).

Collectively, these results suggest that sparse interpolated observational datasets can be leveraged by such data-driven techniques. For example, data from moorings deployed in the Drake Passage as part of the Diapycnal and Isopycnal Mixing Experiment in the Southern Ocean (DIMES) can be used to infer eddy momentum or heat flux divergences in other parts of the Southern Ocean. Besides, datasets from Argo floats (Chapman and Sallée, 2017), mooring data, ADCPs, and SSH from altimetry, could be combined to reconstruct physically- and biogeochemically important quantities such energy reservoirs, or air-sea fluxes, interior transport and/or storage of heat, carbon and oxygen in the ocean (Su et al., 2018a; Giglio et al., 2018).

However, one aspect of oceanic observational data we have not considered is Lagrangian structure in the spatial sampling of drifter data - drifter data points are not randomly distributed in space. The Lagrangian structure of drifter data may not be solved with simple interpolation, so future data-driven algorithms must take this into account when training on Argo float data.

5.5.3 Implications for parameterisations

We have shown that neural networks can successfully represent the spatio-temporal variability of the eddy momentum forcing, implying potential for data-driven oceanic turbulence closures in the future, as suggested by Zanna et al. (2018). The generalisation ability of the neural networks shows that only a limited amount of observations or high-resolution model data may be needed to successfully represent sub-grid scale processes. While the CNNs are successful at representing the relationship between the eddy momentum forcing and their effect on the resolved flow, the low-resolution climate models might have biases that are too severe (e.g., weak transport and velocity shears) to lead to a successful representation of the eddy momentum forcing from CNNs as trained here. Nonetheless, our results also show that they perform very well for varying degrees of forcing and dissipation. Therefore, until the CNNs are implemented into a coarse-resolution ocean model, their success in improving numerical simulations is speculative but deserves to be investigated.

More work into physically-constrained machine learning algorithms is crucial, and successful applications of data-driven techniques should incorporate physical knowledge. Indeed, the neural network turbulence model of Ling et al. (2016b) outperformed more simple linear models only when Galilean invariant stress tensors from Pope (1975) were used, which are also a key ingredient of the eddy parameterisation proposed by Anstey and Zanna (2017). We successfully enforce global momentum conservation in the present work, such that future implementations of data-driven parameterisations, despite being semi-empirical, can be altered to respect physical principles. Specifically, physical constraints can be incorporated into the architecture of the predictive algorithms. Enforcing global momentum conservation with post-processing worked well in this model, which contained a single strong jet with

a weaker background flow; however, in more realistic global models with multiple strong features, the post-processing approach may not perform as well.

One potential disadvantage of convolutional neural networks may be the computational cost of the matrix operations of each convolution layer to make a prediction given an input. The total time complexity (ignoring any fully-connected layers) of a CNN (He and Sun, 2015) is given by $\mathcal{O}(\sum_l^d n_{l-1} \cdot s_l^2 \cdot n_l \cdot m_l^2)$, where d is the total number of convolution layers, l is the index of a convolution layer, n_l is the number of filters, s_l is the filter size, and m_l is the size of the output feature map. The architectures of the CNNs used in this chapter were optimised to produce the best possible predictive skill. This led to relatively large models, with three convolutional layers and many filters per layer. The large models in this chapter do pose issues for future numerical implementation, as the time complexity of CNNs is larger than that of a traditional eddy closure (e.g., a simple Laplacian dissipation of momentum which only involves a few matrix additions and subtractions). One way to reduce the time complexity is to instead use depth-wise separable convolution layers (e.g Howard et al., 2017), which treat the input channels of a convolution layer more independently. This reduces the number of parameters and hence computational cost. An alternative way of reducing time complexity is to simply reduce the sizes of the input and outputs, i.e. make predictions for a region smaller than 40×40 grid points. The amount of information available to make predictions is therefore reduced. The computational cost is an area which needs addressing if CNNs are to be routinely implemented in models in the future. However, unlike other parameterisations, the training of the neural networks is only done once.

Chapter 6

Inferring expressions for eddy fluxes with machine learning

6.1 Introduction

The deep neural networks of Chapter 5 are adept at learning highly non-linear functions from data, thanks to the automated feature extraction of the convolution layers and their thousands of parameters. However, high algorithmic complexity and the ability to approximate almost any function comes at the cost of interpretability. The neural networks are implicitly learning the physics of unresolved eddies, but that knowledge remains concealed within the black-box nature of the architecture.

As mentioned in Sections 3.3.1 and 3.3.2, when choosing a data-driven algorithm for a particular task, there is generally a trade-off between predictive skill and inference. Deep neural networks have high predictive skill, but it is difficult to infer how the physical variables are used to make a prediction. That is the major strength of the traditional physics-driven approach to parameterisation: the mechanism or theory explicitly describes the physical process in which the input variable leads to the functional form of the parameterisation.

Can we use data-driven methods to explicitly reveal the functional forms of parameterisations? This approach could be viewed as reverse-engineering of a physical mechanism or theory. We learn a functional form of a parameterisation from data, and then use that functional form to understand the physical process that produced it. We attempt such a form of knowledge discovery in this chapter, using recent developments in the data-driven discovery of equations outlined in Section 3.3.2. Instead of training a black-box algorithm to learn sub-grid or sub-filter eddy forcings

from data, we aim to explicitly learn equations for the eddy forcing terms. For any successfully uncovered equation, we see if the functional form provides information on the physics of unresolved turbulent eddies.

Our target quantities to parameterise are the eddy momentum and temperature fluxes. These mesoscale eddy fluxes play a key role in the climate system, and their parameterisation can improve the transport of tracers, as well as countering the energy deficit caused by scale-truncation within coarse-resolution models. Energy is also spuriously lost through Laplacian or biharmonic momentum viscosity operators, which are necessary for numerical stability.

To further improve the energetics of the model, an additional prognostic equation for the eddy energy can be solved account for all sources and sinks of energy within the system (e.g. Eden and Greatbatch, 2008; Marshall and Adcroft, 2010; Jansen and Held, 2014; Mak et al., 2016). However, the prognostic eddy energy equation is unknown and must therefore be constructed. This can be achieved through heuristic reasoning, but this can involve over-simplification of the underlying dynamics. Therefore, we apply the data-discovery of equations to find a prognostic eddy energy equation, in addition to the momentum and temperature fluxes.

The results of this chapter have been published in *Geophysical Research Letters* (<https://doi.org/10.1029/2020GL088376> Zanna and Bolton, 2020) with Professor Laure Zanna.

This chapter is organised as follows. Section 6.2 outlines the specific ocean model used, including how the training and validation data is formed and the data-driven algorithm used. Section 6.3 presents the results of applying the data-driven equation discovery method to find expressions for the eddy momentum forcing, in both barotropic and baroclinic ocean models. Section 6.4 presents expressions revealed for the eddy temperature forcing in the baroclinic model. The data-driven equation discovery is applied to find a prognostic eddy energy equation in Section 6.5. Section 6.6 investigates the physical interpretation of the resulting equations for the eddy momentum and temperature equations. In Section 6.7, we summarise and discuss the implications of our results.

6.2 Data and methods

We use a primitive equation model, MITgcm, to provide high-resolution data for this chapter. We run idealised barotropic and baroclinic models, to emulate western boundary currents and their jet extensions at mid-latitudes. More details of MITgcm and the model configurations can be found in Section 3.1.3. After spin-up, we use 1000 time-slices of model output, with 4 days between each time-slice.

Interestingly, later investigation reveals that we could halve the number of time-slices without having a major impact on the final result. Although we did not investigate further, we could have potentially used even less data for training. The RVM algorithm (Tipping, 2001), described in Section 3.3.2, seems to require less training data compared to deep learning approaches, which could be beneficial.

With regards to data preparation, we remove information at small-scales by applying a horizontal Gaussian filter of width 30km and then coarse-grain to a 30km grid - more details on this averaging procedure can be found in Section 3.2. The combination of filtering and coarse-graining is denoted by a $\overline{(\cdot)}$. We use this averaging procedure to calculate the sub-grid eddy momentum and temperature forcing using the following definitions

$$\mathbf{S}_u = (\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{\mathbf{u}} - \overline{(\mathbf{u} \cdot \nabla)\mathbf{u}} \quad (6.1)$$

$$S_T = (\bar{\mathbf{u}} \cdot \bar{\nabla})\bar{T} - \overline{(\mathbf{u} \cdot \nabla)T}, \quad (6.2)$$

which quantifies the interaction between the sub-grid and resolved fields. We use a horizontal - and not vertical - averaging procedure for simplicity, which also allows us to compare the eddy momentum expressions between the barotropic and baroclinic models. For more details on how the averaging impacts the eddy forcing definition, please refer to Section 3.2.

We now turn our attention to the process of data-driven equation discovery. To begin with, we calculate the target variable - the eddy momentum or temperature forcing - at every grid-point for every time-slice. Then, again at every grid-point and time-slice, we construct a library of basis functions. The aim of the library is to have a diverse range of functions that are relevant to the process being parameterised. Specifically, we build the library from the resolved velocities $\bar{u}$ and $\bar{v}$, using up to 2nd order spatial derivatives and 2nd order polynomial products.

We could not go higher than 2nd order due to memory constraints. The regression part of the RVM algorithm needs to calculate the value of each basis function at every point in time or space. In the barotropic model, this amounts to a total of $N_b \times N_t \times N_x \times N_y$ data points, where N_b is the number of basis functions, and N_t , N_x , and N_y are the temporal and spatial dimensions of the data. Increasing the order of the polynomial products and derivatives, from two to three, leads to a very large increase in the number of basis functions N_b , because you are forming every possible combination of products with the new higher-order derivatives and polynomials.

We write the spatial derivatives of the velocity field using the following functions

$$\zeta = \frac{\partial \bar{v}}{\partial x} - \frac{\partial \bar{u}}{\partial y}, \quad (6.3a)$$

$$\sigma = \frac{\partial \bar{u}}{\partial x} + \frac{\partial \bar{v}}{\partial y}, \quad (6.3b)$$

$$D = \frac{\partial \bar{u}}{\partial y} + \frac{\partial \bar{v}}{\partial x}, \quad (6.3c)$$

$$\tilde{D} = \frac{\partial \bar{u}}{\partial x} - \frac{\partial \bar{v}}{\partial y}, \quad (6.3d)$$

where ζ is the relative vorticity, σ is the divergence, and D and $\tilde{D}$ are the shearing and stretching deformation of the flow field. The dynamical quantities defined by Equations 6.3a-d are relevant to parameterising turbulent eddies, which are the leading order components controlling deformation of fluid elements. We, therefore, might have anticipated this expressions to be important on physical grounds. Indeed, the deformation functions are present in the momentum parameterisation of Smagorinsky (1963).

We chose to write the library of functions using this basis because in the early development of this chapter, we found our data-driven discovery method was automatically forming Equations 6.3a-d, with no a priori knowledge. The RVM algorithm was, therefore, revealing an improved basis in which to write the library of functions. Finding the optimal physical basis is important, as it reveals the most important dynamical components from which to construct the parameterisation and provides information about the physical process itself.

Once we have calculated up to 2nd order derivatives and products of the velocities, and written them using the above basis, we then apply a physical constraint: we only use library functions that can be written as the divergence of a flux. For a

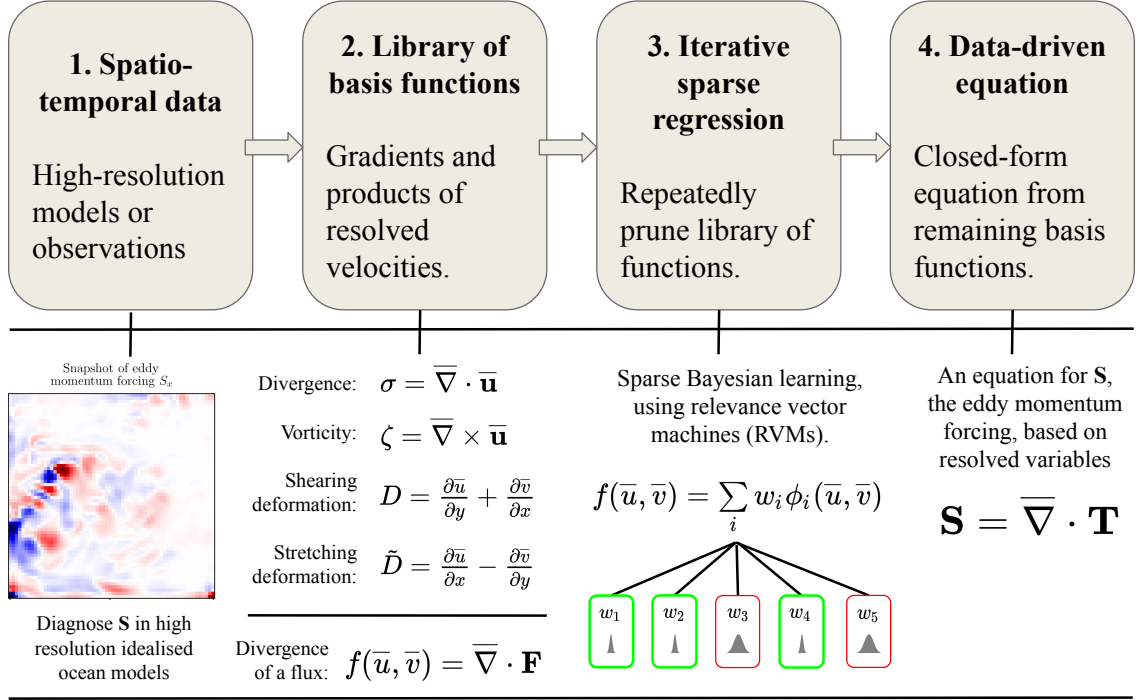


Figure 6.1: An illustration from start-to-finish of the method of discovering equations from data. The top-row provides the general concepts of the procedure: (1) a spatio-temporal dataset is used; (2) a library of basis functions is constructed from the dataset; (3) an iterative sparse regression prunes the library of basis functions; (4) when the iterative regression has converged, the remaining basis functions form the final equation. The bottom row shows specifically how we have adapted this technique for discovering eddy momentum parameterisations from high-resolution ocean models.

two-dimensional vector field $\mathbf{F}$ over an area A , the divergence theorem states that $\iint_A (\nabla \cdot \mathbf{F}) dA = \int_{C=\partial A} \mathbf{F} \cdot \mathbf{n} dl$, where C is the boundary to the region A , and $\mathbf{n}$ is a unit vector perpendicular to the boundary. In other words, the global integral of a quantity $\nabla \cdot \mathbf{F}$ can be reduced to simply evaluating $\mathbf{F}$ along the boundary.

Hence, the global integral of the quantity $\nabla \cdot \mathbf{F}$ can vanish if appropriate boundary conditions are applied to the flux $\mathbf{F}$. By constraining the eddy temperature and momentum expressions to be the divergence of a flux, global conservation can be enforced through the consideration of the boundary conditions. As the eddy momentum forcing comprises of two components (one zonal and one meridional), the constraint means that the final expression can be written as the divergence of a stress tensor: $\hat{\mathbf{S}}_u \propto \overline{\nabla} \cdot \mathbf{T}$.

In the case of the eddy momentum forcing, only using functions that can be written

as a divergence of a flux reduces the number of basis functions in the library from over 100 to 54. In the eddy temperature forcing case, even after constraining the functions to be the divergence of a flux, the library contains 261 basis functions. The higher number compared to the eddy momentum library is due to temperature being an additional field to construct functions from. Also, the vertical derivatives are possible in the baroclinic model and are relevant to the eddy temperature forcing, which further increases the number of possible functions. We include up to second-order derivatives and products of the vertical derivatives of the velocities and temperature, in the same way as the horizontal derivatives, but we do not include cross derivatives to reduce the computational costs.

Before applying the sparse Bayesian regression (Tipping, 2001) to the function library, the data needs to be normalised to zero mean and unit variance. Without normalisation, the iterative pruning does not work, as the values of each basis function have significantly different magnitudes, which means the corresponding uncertainty estimates for each regression coefficient cannot be compared. For example, consider two hypothetical regression coefficients, $c_0 \approx 10^2$ and $c_1 \approx 10^9$, that were estimated from the sparse Bayesian regression without normalisation. The uncertainty of the coefficient c_1 will likely be higher than the uncertainty of c_0 , simply due to the higher magnitude; normalising the data removes the dependence of the uncertainty estimates on the magnitude of the values.

To normalise the data, we normalise each basis function, i.e., each input to the sparse regression, individually. For a particular basis function ϕ_i , we calculate the mean μ and standard deviation σ using all points in both space and time. The normalisation is then carried out as $\phi_i \rightarrow (\phi_i - \mu)/\sigma$. In the barotropic model, the mean and standard deviation of each basis function are calculated using values across three dimensions: longitude, latitude, and time. Whereas in the baroclinic model, the mean and standard deviation are calculated using values across four dimensions: longitude, latitude, depth, and time. The targets of the sparse regression, S_x , S_y , and S_T , are also normalised to zero mean and unit variance, in the same way as each basis function. Note that, after the pruning has finished and the final equation has been revealed, we re-scale the regression coefficients back to the original magnitude of their basis function.

Once the data has been normalised, we apply the iterative relevance vector machine (RVM) algorithm to prune the library and construct the final equation. When the final equation has been constructed, the performance is estimated by calculating the R^2 coefficient of determination using the withheld validation data. In this chapter, we use 50% of the 1000 time-slices for training, and the other 50% for validation. The full process of discovery, from the data to the final equation, is illustrated in Figure 6.1. For more details on the RVM algorithm, and data-driven discovery of equations in general, please refer to Section 3.3.2.

6.3 Results: eddy momentum flux expressions

To reveal expressions for the eddy momentum forcing, we separately apply the RVM algorithm to data from a barotropic model and a baroclinic model. The RVM algorithm is also applied separately to each component of the eddy momentum forcing S_x and S_y . For each model, we first describe the sensitivity of the RVM algorithm to the uncertainty threshold, the sole hyperparameter of the method (at any iteration of the RVM algorithm, if the regression coefficient of a basis function has an uncertainty higher than the threshold, it is pruned from the function library). After the sensitivity analysis, we show the dominant expression revealed by the RVM algorithm. We compare diagnostics of the predictions of the revealed expression, with diagnostics of the true forcing. We examine the time-mean and standard deviation of each expression - we will consider higher-order statistical moments, namely skewness and kurtosis, in Chapter 7.

6.3.1 Barotropic model

To determine how the results change with respect to the uncertainty threshold, we apply the RVM algorithm to data from the barotropic model using a range of threshold values. For each threshold value, we apply the RVM algorithm to the data until convergence, i.e., when the uncertainty values for each regression coefficient remain unchanged from iteration to iteration. After convergence, the remaining basis functions and their regression coefficients are regarded as the final ‘discovered’ expression. For each threshold value, we track the number of basis functions selected and the validation R^2 coefficient of determination of the *final* expression.

Figure 6.2 shows how both the R^2 value and the number of basis functions of the final expression vary with the pruning threshold hyperparameter. Figure 6.2a and 6.2b show the sensitivity to the pruning threshold for the zonal $\hat{S}_x$ and meridional $\hat{S}_y$ eddy momentum forcing components respectively; at a particular threshold, the red curve shows the R^2 value and the blue curve shows the number of basis functions.

At low values of the pruning threshold ($\delta < 10^0$), the RVM algorithm is overly stringent, meaning that most basis functions are pruned on the first iteration and generally leads to underfitting. As the pruning threshold increases ($\delta = 10^{0.5} - 10^{1.5}$), the RVM algorithm becomes more flexible and is more free to construct different combinations of basis functions. At high threshold values ($\delta > 10^{1.5}$), the RVM algorithm becomes overly lenient and retains many basis functions which do not significantly increase the R^2 , therefore leading to overfitting.

At low threshold values, for S_x and S_y , the RVM algorithm selects a single basis function, namely the gradients of enstrophy $(\zeta^2)_x$ and $(\zeta^2)_y$ respectively. Note that we use the shorthand $()_x \equiv \frac{\partial}{\partial x}$ for spatial derivatives. The gradient of enstrophy captures $\sim 20\%$ of the variance in both S_x and S_y . As the pruning threshold increases to $\delta = 10^{0.6}$, there is a large increase from $\sim 20\%$ to $\sim 50\%$ variance captured, with the number of basis functions only increasing to 3 for both S_x and S_y . This expression is given by

$$\hat{\mathbf{S}}_u = \begin{pmatrix} \hat{S}_x \\ \hat{S}_y \end{pmatrix} = \begin{pmatrix} c_0(\zeta^2)_x - c_1(\zeta D)_x + c_2(\zeta \tilde{D})_y \\ c_3(\zeta^2)_y + c_4(\zeta D)_y + c_5(\zeta \tilde{D})_x \end{pmatrix}, \quad (6.4)$$

where the zonal and meridional components of the expression capture 55.6% and 50.6% of the variance respectively. The c_i are the individual regression coefficients of each basis function. The values of c_i are estimated by the final iteration of the sparse regression, which yields the following values

$$\begin{aligned} c_0 &= -4.096 \times 10^8, & c_1 &= -5.483 \times 10^8, & c_2 &= -4.384 \times 10^8, \\ c_3 &= -4.100 \times 10^8, & c_4 &= -6.332 \times 10^8, & c_5 &= -4.815 \times 10^8, \end{aligned}$$

where each regression coefficient has units of m^2 . To quantify the differences between the regression coefficients, we use the coefficient of variation. For a set of continuous values, the coefficient of variation is defined as the standard deviation as a percentage of the mean, therefore quantifying the spread of the values relative to the mean. A coefficient of variation close to 0% implies that values within a set deviate very little

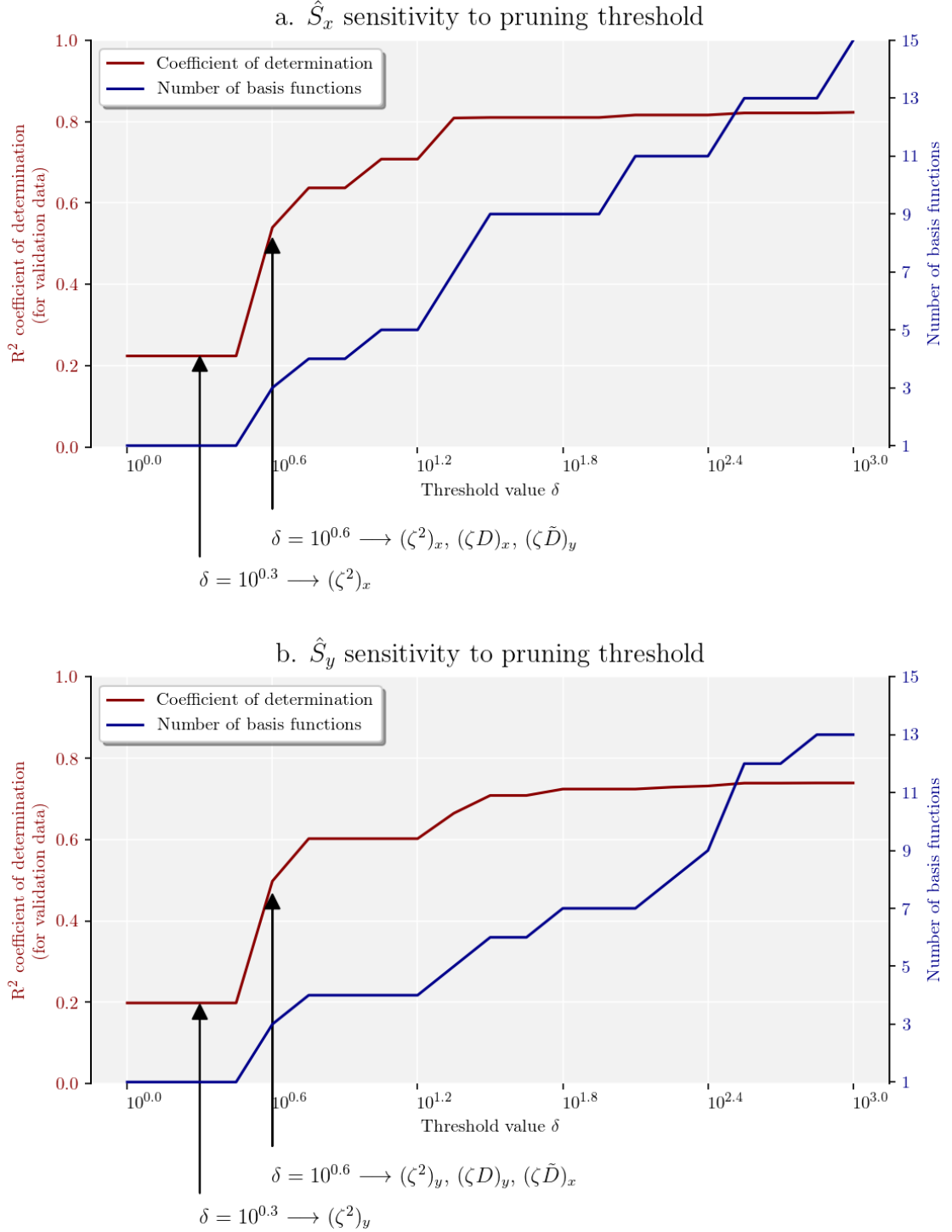


Figure 6.2: The impact of the pruning threshold on the results of the RVM algorithm applied to data from the barotropic model. Panel (a) and (b) show how the R^2 coefficient of determination (red) and number of selected basis functions (blue) changes with the pruning threshold, when applied to reveal expressions for the eddy momentum forcings S_x and S_y respectively.

from the mean; a coefficient of variation greater than 100% implies that individual variation between values is of the same order or greater than the mean itself.

For the regression coefficients c_i above, the coefficient of variation of 14.2%. We therefore decide to write the regression coefficients as approximately equal, i.e., $c_i \approx \kappa = -4.87 \times 10^8 \text{m}^2$, with an average error of 14.2%. Note that the value of $\kappa = -4.87 \times 10^8 \text{m}^2$ is the same order of magnitude as the horizontal resolution squared, which may indicate some functional dependence similar to Mana and Zanna (2014). Using this approximation, we can then re-write Equation 6.4 in the form given by

$$\hat{\mathbf{S}}_u \approx \kappa \bar{\nabla} \cdot \begin{pmatrix} \zeta^2 - \zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta^2 + \zeta D \end{pmatrix}. \quad (6.5)$$

The expression now has a single scalar as a tunable parameter, κ , which determines the ‘strength’ of the parameterisation. The expression depends only on the spatial derivatives of the vorticity and deformation terms, and therefore depends only on second order spatial derivatives of the velocities $\bar{u}$ and $\bar{v}$. From hereon we refer to the expression defined by Equation 6.5 as the barotropic momentum expression.

Note that the barotropic momentum equation contains within it the deformation-based momentum parameterisation of Anstey and Zanna (2017), defined by

$$\hat{\mathbf{S}}_{AZ17} \propto \bar{\nabla} \cdot \begin{pmatrix} -\zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta D \end{pmatrix}, \quad (6.6)$$

which is known to capture up-gradient momentum fluxes and dissipate small-scale enstrophy. The barotropic momentum expression of Equation 6.5 can therefore be interpreted as the gradient of enstrophy plus the deformation-based parameterisation of Anstey and Zanna (2017). Also, the study of Anstey and Zanna (2017) (specifically their Figures 6b and 7a) found the value of the coefficient κ , when averaged across the domain of the model, to be approximately $-5 \times 10^8 \text{m}^2$ - close to our empirical estimate.

How do the predictions of Equation 6.4 compare to the true eddy momentum forcing in the validation data? Figure 6.3 compares maps of the time-mean and standard deviation of the predicted $\hat{S}_x$ and $\hat{S}_y$ with the true S_x and S_y , including the Pearson linear correlation between them.

The time-mean of the predicted zonal component $\hat{S}_x$ (Figure 6.3a) captures the large-scale features of the true S_x (Figure 6.3b): negative in the western half of the

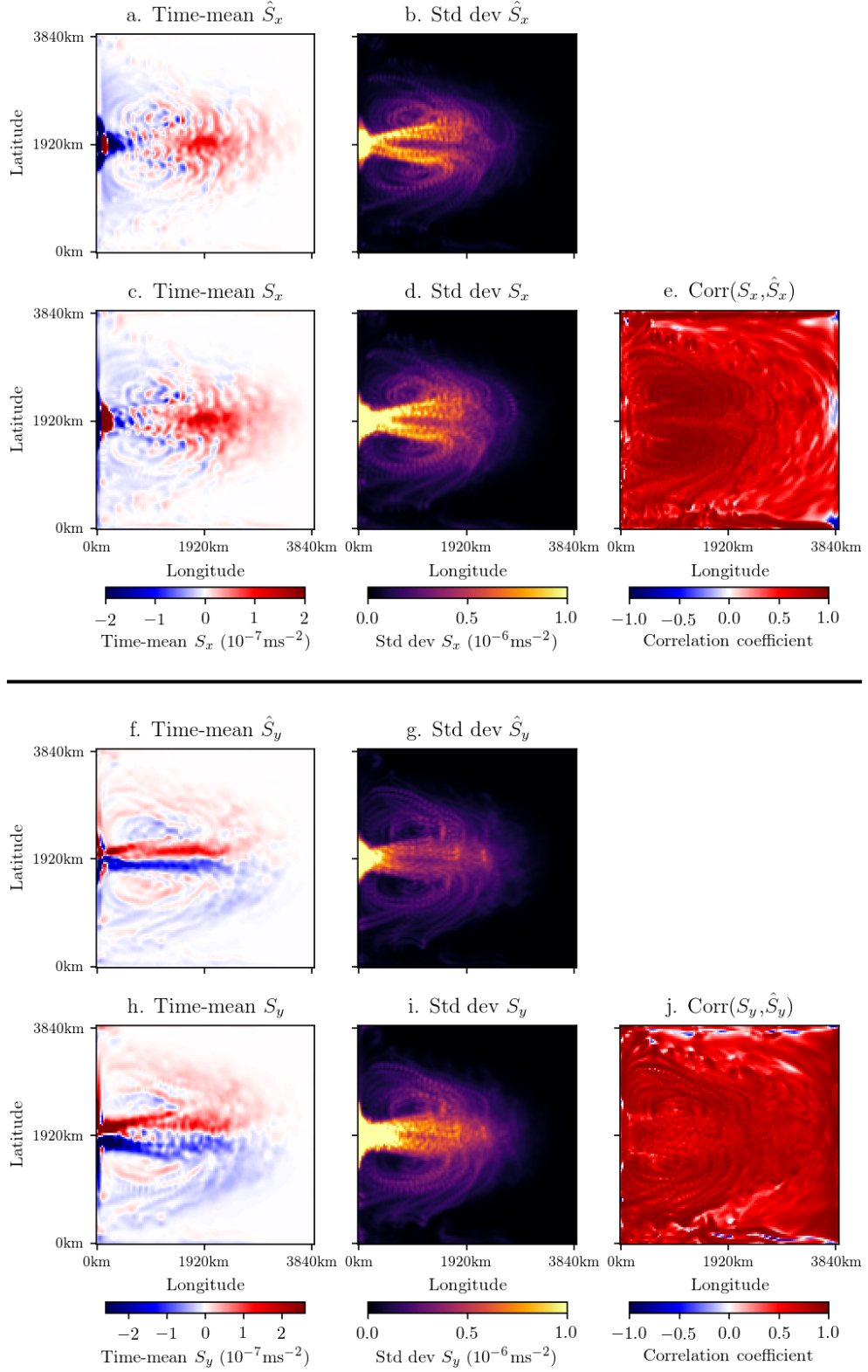


Figure 6.3: Within the validation data of the barotropic model, comparisons of the true eddy momentum forcing S_x and S_y with the predicted values $\hat{S}_x$ and $\hat{S}_y$ from the expression revealed by the RVM algorithm, defined by Equation 6.4.

domain and positive moving eastward. The presence of along-stream variation in the eddy momentum fluxes is similar to that of the along-stream variation of the eddy tracer fluxes of Chapter 4, implying different physical processes up- and down-stream. Away from the western boundary, $\hat{S}_x$ captures the majority of the small-scale spatial variability in the time-mean maps. Right at the western boundary, where the jet separates, the predicted $\hat{S}_x$ is generally too negative and does not fully capture the large positive peak where $S_x > 2 \times 10^{-7} \text{ms}^{-2}$.

The standard deviation of the predicted zonal component $\hat{S}_x$ (Figure 6.3b) captures most of the spatial variability of the true S_x (Figure 6.3d). The predicted $\hat{S}_x$ standard deviation underestimates the true standard deviation by approximately 50% at the western boundary where the jet separates. The correlation map (Figure 6.3e) shows positive correlations almost everywhere, with the highest correlations ($r > 0.6$) away from the boundaries, where the jet extends eastward into the domain. High correlations ($r > 0.6$) are also observed along the northern and southern boundaries.

We observe similar results for the predicted meridional component: $\hat{S}_y$ captures most of the large-scale pattern in the time-mean (Figure 6.3f) and standard deviation (Figure 6.3). However, the predicted component $\hat{S}_y$ is visibly more dissimilar to the true S_y time-mean (Figure 6.3h) and standard deviation (Figure 6.3i), by not capturing all the small-scale variability along the jet core where, moving northwards, the meridional forcing changes sign from negative to positive. The correlation map between $\hat{S}_y$ and S_y (Figure 6.3) is positive almost everywhere, with the highest correlations within the flanks of the jet and also along the eastern boundary.

6.3.2 Baroclinic model

In the same way as the barotropic model, we determine the sensitivity of the RVM algorithm when applied to data from the baroclinic model. The baroclinic model displayed some numerical noise at the northern boundary of the top three vertical layers (the top 180m of the model), so these were excluded from the data. At the bottom layers of the model, mesoscale features are virtually absent, leading to eddy momentum forcing values that approach zero. The bottom four layers were also excluded, to reduce the memory demands of using the RVM algorithm. Excluding the bottom layers from the training data could have reduced the accuracy of the RVM

algorithm at these layers, but the magnitude of the eddy forcings at those depths is almost negligible, so the impact will be minimal. Hence, eight of the 15 vertical layers were used in the subsequent results of the baroclinic model.

In the barotropic model, the sparse regression of the RVM algorithm uses a total of $N_b \times N_t \times N_x \times N_y$ data points, where N_b is again the number of basis functions. In the baroclinic case, the number of data points used in the sparse regression is now $N_b \times N_t \times N_x \times N_y \times N_z$, where N_z is the number of vertical layers used. We therefore use data from the selected vertical layers all at once, and do not apply the RVM algorithm separately to each vertical layer.

Figure 6.4 shows how both the R^2 value and the number of basis functions of the final expression vary with the pruning threshold hyperparameter. Figure 6.4a and 6.4b show the sensitivity to the pruning threshold for the zonal S_x and meridional S_y eddy momentum forcing components respectively.

A similar pattern emerges for the baroclinic model: low values of the pruning threshold lead to underfitting, while large values lead to overfitting, i.e., retaining more basis functions with little to no increase in R^2 . The significant increase in the R^2 occurs at $\delta \sim 10^{0.4}$, where three basis functions are retained, capturing over 40% of the variance. Here, the RVM algorithm constructs the same eddy momentum forcing from the barotropic model (Equation 6.4), albeit with different values for the regression coefficients.

A second increase in the R^2 occurs at $\delta \sim 10^{0.75}$, where five basis functions are retained, capturing nearly 70% of the variance. The resulting expression for $\hat{\mathbf{S}}_{\mathbf{u}}$ at this threshold value is given by

$$\begin{aligned} \hat{\mathbf{S}}_{\mathbf{u}} &= \begin{pmatrix} \hat{S}_x \\ \hat{S}_y \end{pmatrix} \\ &= \begin{pmatrix} \frac{1}{2}c_0(\zeta^2)_x + \frac{1}{2}c_1(D^2)_x + \frac{1}{2}c_2(\tilde{D}^2)_x - c_3(\zeta D)_x + c_4(\zeta \tilde{D})_y \\ \frac{1}{2}c_5(\zeta^2)_y + \frac{1}{2}c_6(D^2)_y + \frac{1}{2}c_7(\tilde{D}^2)_y + c_8(\zeta \tilde{D})_x + c_9(\zeta D)_y \end{pmatrix}, \end{aligned} \quad (6.7)$$

where again the c_i are the regression coefficients for each basis function, and are estimated from the training data. The zonal $\hat{S}_x$ and meridional $\hat{S}_y$ components of Equation 6.7 capture 68.3% and 68.7% of the variance respectively. The values of each c_i are the following

$$\begin{aligned} c_0 &= -10.565, & c_1 &= -7.585, & c_2 &= -8.483, & c_3 &= -9.186, & c_4 &= -9.240, \\ c_5 &= -9.884, & c_6 &= -6.970, & c_7 &= -7.833, & c_8 &= -8.503, & c_9 &= -9.018, \end{aligned}$$

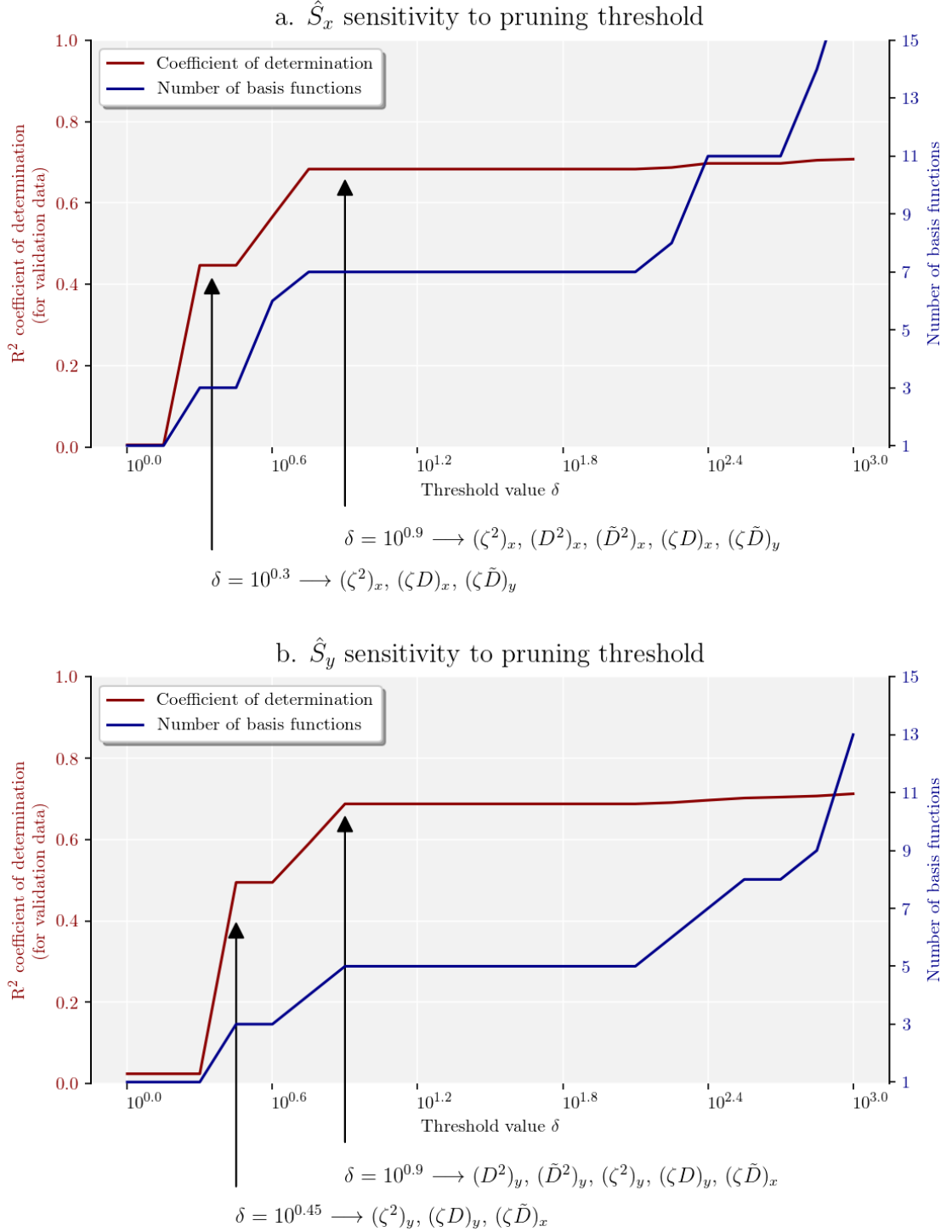


Figure 6.4: The impact of the pruning threshold on the results of the RVM algorithm applied to data from the baroclinic model. Panel (a) and (b) show how the R^2 coefficient of determination (red) and number of selected basis functions (blue) changes with the pruning threshold, when applied to reveal expressions for the eddy momentum forcings S_x and S_y respectively.

where each regression coefficient has units of 10^8m^2 . The mean value is $\kappa = -8.723 \times 10^8 \text{m}^2$, with a coefficient of variation of 9.8%. Similar to the barotropic case, due to the low coefficient of variation, we can assume that all regression coefficients are approximately equal, i.e., $c_i \approx \kappa$. Making this approximation, we can re-write Equation 6.7 in the form

$$\hat{\mathbf{S}}_{\mathbf{u}} \approx \kappa \bar{\nabla} \cdot \begin{pmatrix} \frac{1}{2}(\zeta^2 + D^2 + \tilde{D}^2) - \zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \frac{1}{2}(\zeta^2 + D^2 + \tilde{D}^2) + \zeta D \end{pmatrix}, \quad (6.8)$$

or alternatively as

$$\hat{\mathbf{S}}_{\mathbf{u}} \approx \kappa \bar{\nabla} \cdot \begin{pmatrix} -\zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta D \end{pmatrix} + \mathbf{I} \frac{1}{2} \kappa \bar{\nabla} (\zeta^2 + D^2 + \tilde{D}^2), \quad (6.9)$$

where κ is now the sole parameter of the expression. The baroclinic expression depends only on the spatial derivatives of the shearing deformation D , the stretching deformation $\tilde{D}$, and the vorticity ζ . Like the barotropic expression, the tensor is symmetric, despite separately applying the RVM algorithm to the zonal and meridional components of the eddy momentum forcing. The baroclinic expression can be written as the barotropic expression plus the gradient of the squared-deformation terms: $\hat{\mathbf{S}}_{BC} = \hat{\mathbf{S}}_{BT} + \mathbf{I} \frac{1}{2} \kappa \bar{\nabla} (D^2 + \tilde{D}^2)$.

The baroclinic model has multiple vertical layers, so to examine the predictions $\hat{S}_x$ and $\hat{S}_y$ of Equation 6.7, we depth-average the time-mean, standard deviation, and correlation coefficient, shown in Figure 6.5. The time-mean of $\hat{S}_x$ and $\hat{S}_y$ from the baroclinic expressions (Figures 6.5a and 6.5f) revealed by the RVM algorithm captures the majority of the large- and small-scale patterns of the true time-mean (Figures 6.5c and 6.8h). The spatial patterns of the standard deviations of $\hat{S}_x$ and $\hat{S}_y$ (Figures 6.5b and 6.5g) also match the spatial patterns of true standard deviations (Figures 6.5d and 6.5i), but underestimate the amplitude by approximately 40%. The correlation maps (Figures 6.5e and 6.5j) show an almost uniform coefficient of $r \approx 0.8$ across the domain, with the exception of near the northern boundary, where numerical noise is present and reduces the amount of the variance captured by $\hat{S}_x$ and $\hat{S}_y$.

We now show the same diagnostics - maps of the time-mean, standard deviation, and correlation - but this time zonally-averaged (Figure 6.6), to examine how the predictions $\hat{S}_x$ and $\hat{S}_y$ vary vertically within the validation data. The zonally-averaged

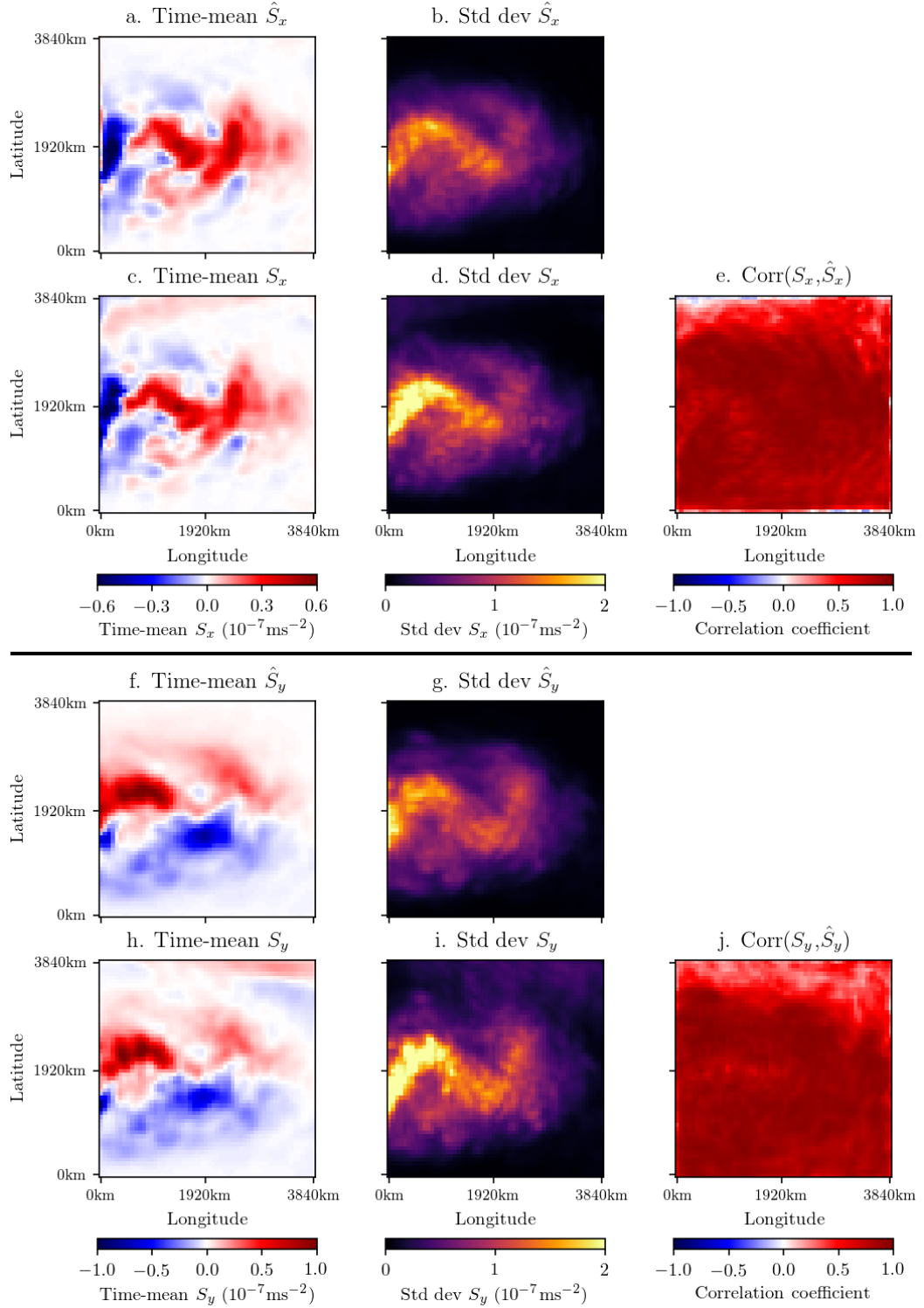


Figure 6.5: Within the validation data of the baroclinic model, a depth-averaged comparison of the true eddy momentum forcing S_x and S_y with the predicted values $\hat{S}_x$ and $\hat{S}_y$ from the expression revealed by the RVM algorithm, defined by Equation 6.7. The spatial maps here are a function of longitude (x-axis) and latitude (y-axis).

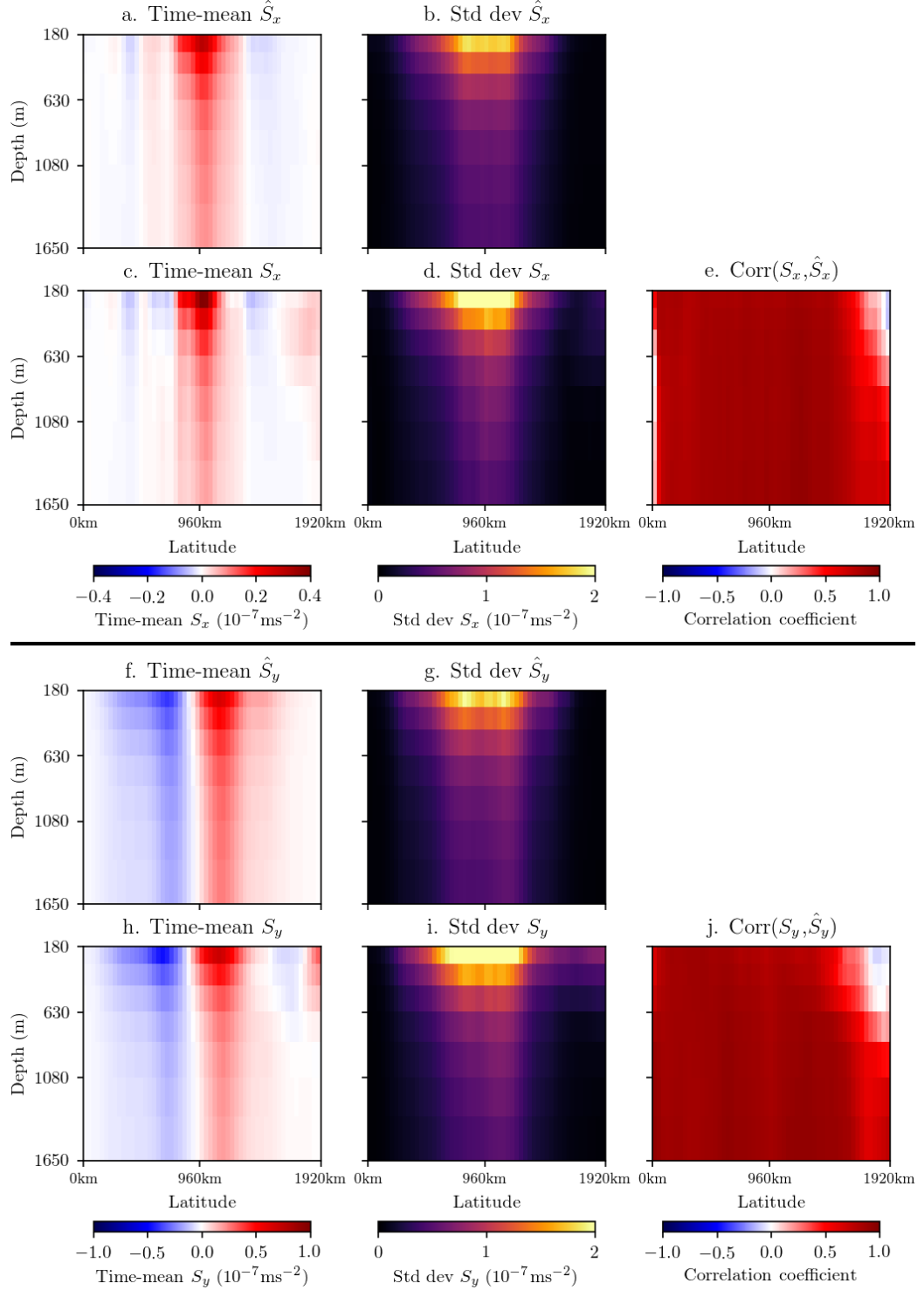


Figure 6.6: Within the validation data of the baroclinic model, a zonally-averaged comparison of the true eddy momentum forcing S_x and S_y with the predicted values $\hat{S}_x$ and $\hat{S}_y$ from the expression revealed by the RVM algorithm, defined by Equation 6.7. The spatial maps here are a function of latitude (x-axis) and depth (y-axis).

time-means of the predicted $\hat{S}_x$ and $\hat{S}_y$ (Figure 6.6a and 6.6f) reproduce the spatial patterns of the true time-means (Figure 6.6c and 6.6h), apart from near the northern boundary, where neither $\hat{S}_x$ nor $\hat{S}_y$ capture the sign change in S_x and S_y at high latitudes in the upper 700m of the model. This is confirmed by the correlation maps, which uniformly take values of $r \approx 0.8$, with the correlation coefficients degrading to zero near the north boundary at the surface; the reduction in variance captured at the northern boundary may again be due to numerical noise. Overall, with the exception of the northern boundary, the baroclinic expression of Equation 6.7 captures over half the variance across the domain, independent of horizontal or vertical position.

6.4 Results: eddy temperature flux expressions

We now present the results of applying the RVM algorithm to reveal expressions for the eddy temperature forcing, defined by Equation 6.2. The same horizontal averaging procedure is used as in the momentum forcing case, so we are trying to reveal expressions for *horizontal* eddy temperature fluxes. The barotropic primitive equation model does not have temperature as a prognostic variable, so the RVM algorithm is applied only to data from baroclinic model.

Note that, for the eddy temperature forcing, the RVM algorithm did not automatically construct quantities such as the vorticity ζ or deformations D and $\tilde{D}$, as in the momentum forcing case. We therefore did not re-write the library of functions using that basis here, and used the original basis of $\frac{\partial u}{\partial x}$, $\frac{\partial u}{\partial y}$, $\frac{\partial v}{\partial x}$, and $\frac{\partial v}{\partial y}$.

As in the momentum case, we constrain the functions of the library such that the final expression is always the divergence of a flux. Through the divergence theorem, the global integral of the temperature forcing can vanish by setting appropriate boundary conditions. This will help any revealed temperature flux parameterisation to conserve energy if subsequently implemented.

6.4.1 Baroclinic model

Figure 6.7 shows how both the R^2 value and the number of basis functions of the final eddy temperature forcing expression vary with the pruning threshold hyperparameter. Above a pruning threshold of $10^{-0.1}$, the number of basis functions increases to over 15 while increasing the R^2 only marginally, implying overfitting. At $\delta \sim 10^{-0.24}$,

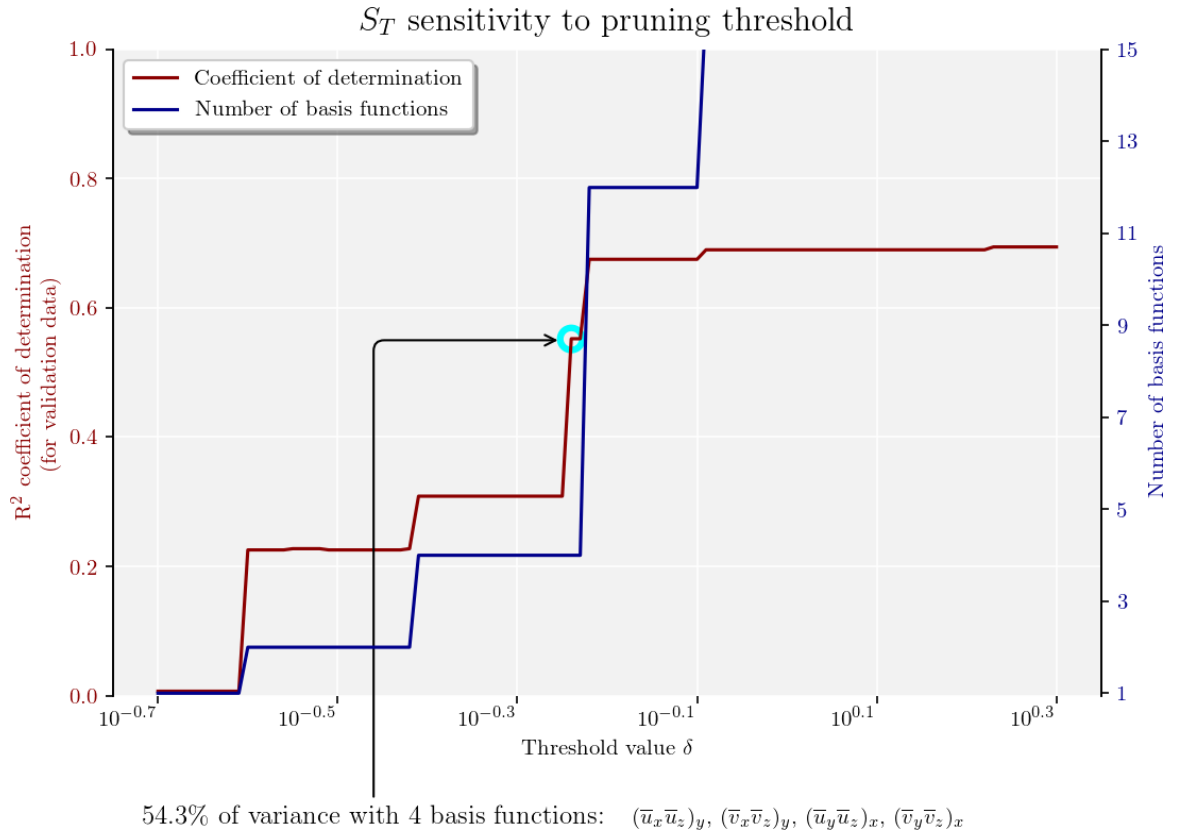


Figure 6.7: The impact of the pruning threshold on the results of the RVM-algorithm applied to the eddy temperature forcing using data from the baroclinic model. The R^2 coefficient of determination (red) and number of selected basis functions (blue) is plotted against the pruning threshold.

highlighted by the cyan circle in Figure 6.7, the R^2 jumps to 54.3% with only 4 basis functions. The expression revealed by the RVM algorithm at this R^2 value is given by

$$\hat{S}_T = c_0(\bar{u}_x\bar{u}_z)_y + c_1(\bar{v}_x\bar{v}_z)_y - c_2(\bar{u}_y\bar{u}_z)_x - c_3(\bar{v}_y\bar{v}_z)_x, \quad (6.10)$$

with the following values for the regression coefficients

$$c_0 = 1.573, \quad c_1 = 1.495, \quad c_2 = 1.518, \quad c_3 = 1.504$$

which have units of 10^8Cms . The mean coefficient value is $\kappa = 1.523 \times 10^8\text{Cms}$ with a coefficient of variation of 1.7%. We decided to approximate all the regression coefficients as being equal, with an average error of 1.7%, which yields the following expression

$$\hat{S}_T \approx \kappa \bar{\nabla} \cdot \begin{pmatrix} -\bar{u}_y\bar{u}_z - \bar{v}_y\bar{v}_z \\ \bar{u}_x\bar{u}_z + \bar{v}_x\bar{v}_z \end{pmatrix}, \quad (6.11)$$

where we have also re-written the eddy temperature forcing, to more clearly show the revealed eddy temperature flux vector. Initially, it appears counter-intuitive for the eddy temperature forcing to be independent of temperature; later in Section 6.6.2, we show how Equation 6.11 does relate to horizontal temperature gradients if thermal wind balance is assumed.

Depth-averaged diagnostics - the time-mean, standard deviation, and correlation - of the predicted eddy temperature forcing $\hat{S}_T$ are shown in Figure 6.8. The time-mean $\hat{S}_T$ (Figure 6.5a) reproduces the general large-scale pattern, but not all small-scale features. The standard deviation of $\hat{S}_T$ (Figure 6.5b) underestimates the standard deviation of the true S_t (Figure 6.5d) by approximately 50% on average. The correlation between $\hat{S}_T$ and S_T decreases at the northern boundary, but is fairly constant across the rest of the domain at $r \approx 0.6$.

Zonally-averaged diagnostics are shown in Figure 6.9. The time-mean $\hat{S}_T$ (Figure 6.5a) does not capture the true time-mean S_T at the northern boundary (Figure 6.5c), but does get the general large-scale pattern correct. The standard deviation of $\hat{S}_T$ (Figure 6.5b) also captures the general pattern of the true S_T (Figure 6.5d), but underestimates the magnitude by approximately 50%. The correlation between $\hat{S}_T$ is vertically uniform with $r \approx 0.6$, decreasing at the northern boundary.

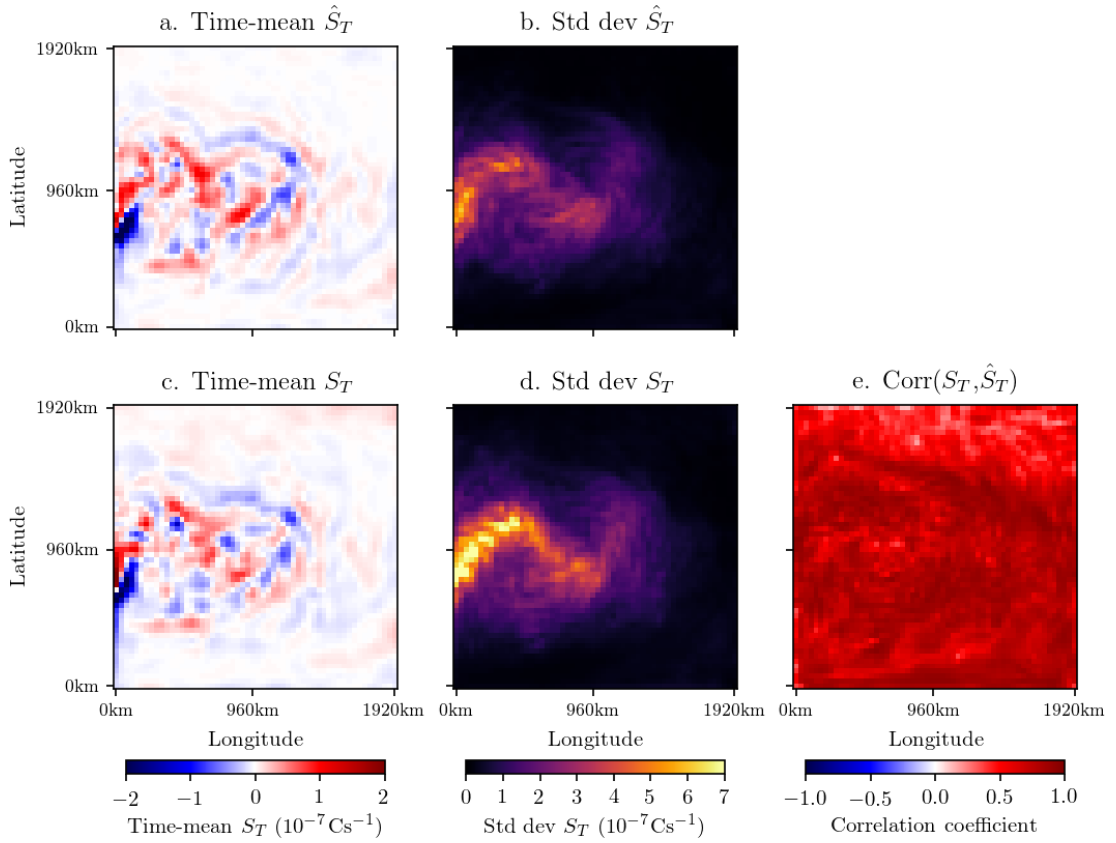


Figure 6.8: Within the validation data of the baroclinic model, a depth-averaged comparison of the true eddy temperature forcing S_T with the predicted values $\hat{S}_T$ from the expression revealed by the RVM algorithm, defined by Equation 6.10. The spatial maps here are a function of longitude (x-axis) and latitude (y-axis).

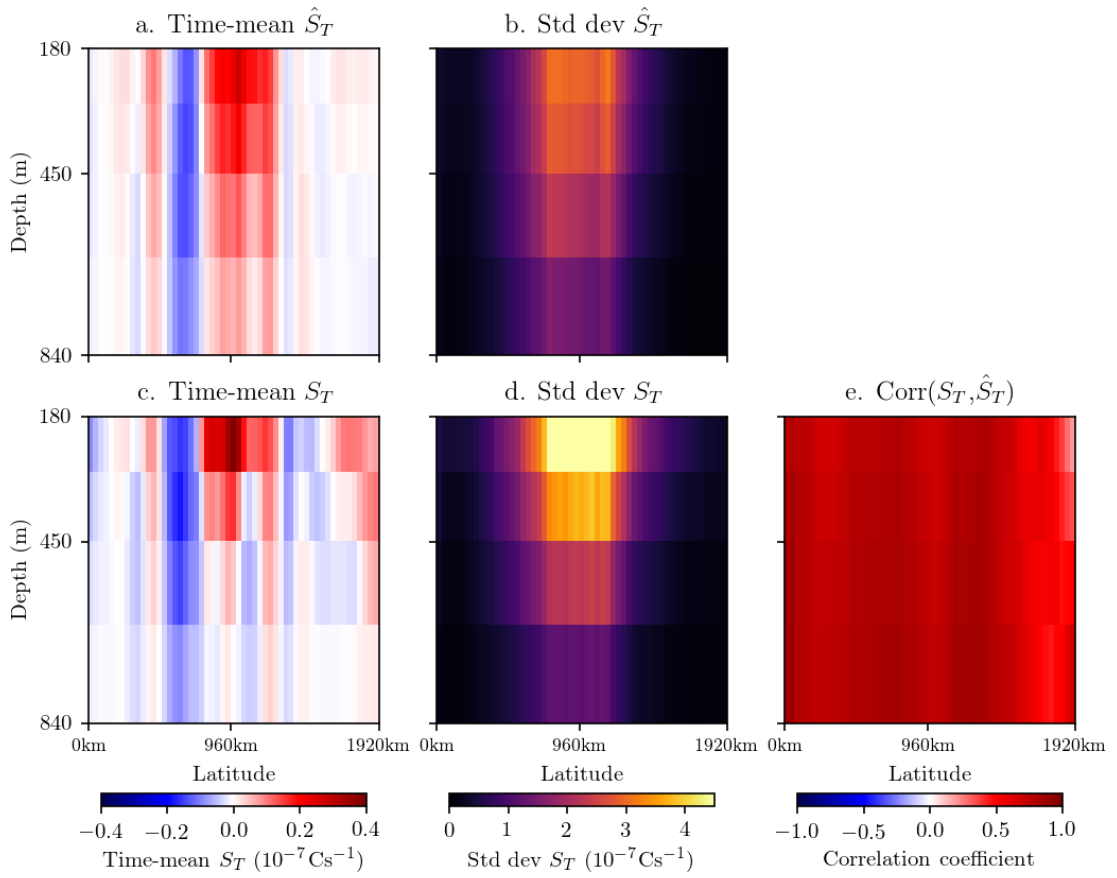


Figure 6.9: Within the validation data of the baroclinic model, a zonally-averaged comparison of the true eddy temperature forcing S_T with the predicted values $\hat{S}_T$ from the expression revealed by the RVM algorithm, defined by Equation 6.10. The spatial maps here are a function of latitude (x-axis) and depth (y-axis).

6.5 Results: prognostic eddy energy equation

The energy of a closed-system should be conserved. Coarse-resolution ocean models exhibit a spurious loss of kinetic energy due to viscous momentum closures such as a Laplacian or biharmonic operators with an enhanced eddy viscosity. The buoyancy flux parameterisation of Gent and McWilliams (1990) flattens isopycnals, through the extraction of available potential energy (APE) of the resolved state. However, this APE is lost indefinitely from the model and is not re-introduced.

To balance the energy lost to existing momentum and buoyancy parameterisations, and then re-introduce it to the model, an additional prognostic equation for the eddy energy can be solved. However, the prognostic eddy energy equation is unknown and has to be constructed (Eden and Greatbatch, 2008; Marshall and Adcroft, 2010). Here, we try and construct an equation for the eddy energy using the RVM algorithm.

Before applying the RVM algorithm, we need to calculate the eddy kinetic energy on the 30km grid with the eddy velocities $\mathbf{u}' = \mathbf{u} - \bar{\mathbf{u}}$. However, due to the coarse-graining in our averaging procedure $(\overline{\cdot})$, the averaged-velocity $\bar{\mathbf{u}}$ is defined on the 30km grid, whereas the original velocity $\mathbf{u}$ exists on the high-resolution grid. Due to this grid mismatch, we cannot compute the simple subtraction in $\mathbf{u}' = \mathbf{u} - \bar{\mathbf{u}}$.

Instead, we subtract the filtered velocity $\mathbf{u}' = \mathbf{u} - \text{filter}(\mathbf{u})$ to form the eddy velocity, and then calculate the eddy kinetic energy as $K_e = \frac{1}{2} \mathbf{u}' \cdot \mathbf{u}'$. This eddy kinetic energy is then coarse-grained to the 30km grid. Albeit not perfect, this approach provides an approximate measure of the kinetic energy of eddies of length-scales between 30km and the grid-scale.

As we want to reveal a *prognostic* equation for the eddy energy, the target quantity of the RVM algorithm is the time derivative $\frac{\partial K_e}{\partial t}$. To accurately estimate the time derivative of K_e , we need data at successive time-steps, so generate new training and validation datasets from the barotropic and baroclinic model. The time derivative is estimated using a central-difference stencil $\frac{\partial K_e}{\partial t} \approx \frac{1}{2dt}(K_e^{t+1} - K_e^{t-1})$, where the time-step of $dt = 300\text{s}$ of the models is used.

As with previous applications of the RVM algorithm, we construct the function library using up to 2nd order derivatives and up to 2nd order polynomial products of the quantities $\bar{u}$, $\bar{v}$, and K_e . The resulting library contains approximately 200 basis functions for the RVM algorithm to select from.

Now the results: for both the barotropic and baroclinic models, the RVM algorithm constructs the following equation

$$\frac{\partial K_e}{\partial t} \approx -(\bar{u} \frac{\partial K_e}{\partial x} + \bar{v} \frac{\partial K_e}{\partial y}), \quad (6.12)$$

$$= -(\bar{\mathbf{u}} \cdot \bar{\nabla}) K_e \quad (6.13)$$

which is simply advection of eddy energy and captures 50-60% of the variance in the validation data. Increasing the pruning threshold of the algorithm to allow more functions to be included in the final results leads to overfitting, as the validation R^2 increases by less than 5% but with many basis functions being added.

We varied the spatial-scale of the Gaussian filter, up to a width of 120km, to see if this would allow the RVM algorithm to reveal expressions beyond advection. Alas, the RVM algorithm still selected only local advection, even with a more non-local definition of eddy energy. We also tried including advection in the target quantity, so instead of applying the RVM algorithm to $\frac{\partial K_e}{\partial t}$, we apply it to construct an equation for the material derivative $\frac{DK_e}{Dt}$. In this case, the RVM algorithm overfitted at all pruning threshold values, and could not improve on advection as the prognostic eddy energy equation.

There are a few potential reasons why only advection was selected for the prognostic eddy kinetic equation:

- Our definition of eddy kinetic energy might not have been appropriate, i.e., how we average the fields.
- The function library may have need 3rd order derivatives and products of the velocities to accurately capture effects beyond advection.
- The time-scale for calculating the time derivative of the eddy kinetic energy might have been too short. Considering time-scales longer than the eddy turnover time may be able to better represent non-advective processes in the training data.

So although we were not successful in this Chapter in revealing an interesting prognostic eddy energy equation, there are still multiple avenues to explore.

6.6 Physical interpretation of eddy flux expressions

We have used a data-driven algorithm to construct expressions for both the eddy momentum and eddy temperature forcing. These expressions capture over 50% of variation in the validation data, using our definition of the ‘eddy’ field to define the true eddy forcings. But what do these expressions represent? Can we understand physically why these expressions appear the way they do? This chapter is motivated as needing more interpretable applications of machine learning, so here we focus on the interpreting the results of the RVM algorithm.

6.6.1 Eddy momentum forcing expressions

The eddy momentum expressions revealed by the RVM algorithm, $\hat{\mathbf{S}}_{BT}$ and $\hat{\mathbf{S}}_{BC}$ in the barotropic and baroclinic models respectively, can be written as the parameterisation of Anstey and Zanna (2017) plus the gradient of additional terms, i.e.,

$$\hat{\mathbf{S}}_{BT} = \mathbf{S}_{AZ17} + \kappa \bar{\nabla} \zeta^2 \quad (6.14)$$

$$\hat{\mathbf{S}}_{BC} = \mathbf{S}_{AZ17} + \frac{1}{2} \kappa \bar{\nabla} (\zeta^2 + D^2 + \tilde{D}^2), \quad (6.15)$$

where, as a reminder, $\mathbf{S}_{AZ17}$ is given by

$$\mathbf{S}_{AZ17} = \kappa \bar{\nabla} \cdot \begin{pmatrix} -\zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta D \end{pmatrix}. \quad (6.16)$$

When examining the contributions of $\hat{\mathbf{S}}_{BT}$ and $\hat{\mathbf{S}}_{BC}$ to the vorticity budget, the gradient terms vanish as $\bar{\nabla} \times \bar{\nabla} \phi = 0$ for any function ϕ . The resulting eddy vorticity forcings for each expression are therefore equal, i.e.,

$$\bar{\nabla} \times \hat{\mathbf{S}}_{BT} = \bar{\nabla} \times \hat{\mathbf{S}}_{BC} = \bar{\nabla} \times \mathbf{S}_{AZ17}, \quad (6.17)$$

which can be written as the divergence of a vorticity flux. The vorticity flux can be found by taking the ‘curl’ of the stress tensor $\mathbf{T}$, which is equal to $\bar{\nabla} \times \mathbf{T} \equiv \mathbf{R} \bar{\nabla} \cdot \mathbf{T} = (-\partial_y, \partial_x) \cdot \mathbf{T}$, where $\mathbf{R} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$. This leads to the following vorticity flux

$$\mathbf{F}_\zeta = \kappa (D \mathbf{R}_\times + \tilde{D} \mathbf{R}_+) \bar{\nabla} \zeta, \quad (6.18)$$

where $\mathbf{R}_+$ and $\mathbf{R}_\times$ are matrices representing reflection in the x-axis and reflection in an axis oriented at 45° to the x-axis, respectively. In Figure 1 of Anstey and Zanna

(2017), the vorticity flux of Equation 6.18 is interpreted as a larger-scale flow field acting, through $(D\mathbf{R}_\times + \tilde{D}\mathbf{R}_+)$, to deform a smaller-scale flow field, represented by $\bar{\nabla}\zeta$. This interaction can produce up-gradient momentum fluxes, sharpening jets, as well as acting as a net sink of enstrophy.

So this is an interpretation of the Anstey and Zanna (2017) expression, but what about the additional gradient terms in the right-hand side of $\hat{\mathbf{S}}_{BT}$ and $\hat{\mathbf{S}}_{BC}$? They vanish in the vorticity budget, but we can gain further insight by considering a hypothetical Reynolds stress tensor, given by

$$\mathbf{T} = \begin{pmatrix} \overline{u'u'} & \overline{u'v'} \\ \overline{v'u'} & \overline{v'v'} \end{pmatrix}, \quad (6.19)$$

where the averaging procedure $\overline{(\cdot)}$ here is a spatial Reynolds decomposition, $u = \bar{u} + u'$, where $\bar{u'} = 0$. Now, assume that the eddy velocities can be approximated by a Taylor expansion of the averaged velocities: $u' \approx \Delta x \frac{\partial \bar{u}}{\partial x} + \Delta y \frac{\partial \bar{u}}{\partial y}$ and $v' \approx \Delta x \frac{\partial \bar{v}}{\partial x} + \Delta y \frac{\partial \bar{v}}{\partial y}$. Plugging these approximations into Equation 6.19, using $\overline{\Delta x \Delta y} = 0$ (assuming Δx and Δy are randomly distributed random variables), and assuming a square coarse-graining box such that $\overline{\Delta x^2} = \overline{\Delta y^2}$, leads to the following tensor based on first-order spatial derivatives of the velocity field

$$\mathbf{T} \propto \begin{pmatrix} \bar{u}_x^2 + \bar{u}_y^2 & \bar{u}_x \bar{v}_x + \bar{u}_y \bar{v}_y \\ \bar{u}_x \bar{v}_x + \bar{u}_y \bar{v}_y & \bar{v}_x^2 + \bar{v}_y^2 \end{pmatrix}, \quad (6.20)$$

where we have used the shorthand $\bar{u}_x = \frac{\partial \bar{u}}{\partial x}$ for spatial derivatives. Let us re-write the above Taylor-expanded Reynolds tensor using the basis used to construct our library of functions for the RVM algorithm: $\zeta = \bar{v}_x - \bar{u}_y$, $\sigma = \bar{u}_x + \bar{v}_y$, $D = \bar{u}_y + \bar{v}_x$, and $\tilde{D} = \bar{u}_x - \bar{v}_y$. The stress tensor of Equation 6.20 becomes

$$\begin{aligned} \mathbf{T} \propto & \begin{pmatrix} \sigma \tilde{D} - \zeta D & \sigma D + \zeta \tilde{D} \\ \sigma D + \zeta \tilde{D} & -\sigma \tilde{D} + \zeta D \end{pmatrix}, \\ & + \frac{1}{2} \mathbf{I}(D^2 + \tilde{D}^2 + \zeta^2 + \sigma^2), \end{aligned} \quad (6.21)$$

where $\mathbf{I}$ is the identity matrix. Now, assume the velocity field is non-divergent, $\sigma = 0$, reducing the tensor to

$$\mathbf{T} \propto \begin{pmatrix} -\zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta D \end{pmatrix} + \frac{1}{2} \mathbf{I}(D^2 + \tilde{D}^2 + \zeta^2), \quad (6.22)$$

which is exactly the same tensor found by the RVM algorithm applied to the baroclinic data in Equation 6.9, including the factor of one half in the gradient terms!

Therefore, the baroclinic eddy momentum forcing $\hat{\mathbf{S}}_{BC}$ is equivalent to Taylor expanding the eddy velocities as a function of the large-scale velocities. The $\mathbf{S}_{AZ17}$ expression is partially related to a Taylor expansion of the eddy velocities, as discussed in Anstey and Zanna (2017), by using only the component of the stress tensor that contributes to the vorticity budget. The barotropic expression $\hat{\mathbf{S}}_{BC}$ also partially relates to Taylor expansion, which includes the gradient of enstrophy ζ^2 , but not the squared deformations D^2 and $\tilde{D}^2$.

A Taylor expansion of the Reynolds stress tensor is known as the non-linear gradient model (Nadiga, 2008; Nadiga and Bouchet, 2011), and assumes that the small-scale flow field of the eddies is ‘self-similar’ to the large-scale flow, and can therefore be predicted as a function of $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}$. Note, in order to recover the non-linear gradient model from our baroclinic RVM expression, we had to assume horizontal non-divergence of the flow; this was despite the training data originating from a free-surface height model with $\bar{\nabla} \cdot \bar{\mathbf{u}} \neq 0$.

We have considered the effects of $\hat{\mathbf{S}}_{BT}$ and $\hat{\mathbf{S}}_{BC}$ on the momentum and vorticity budgets, but what about the impact on the energy? In particular, what is the impact on the global kinetic energy budget? We consider the effect of only $\hat{\mathbf{S}}_{BC}$, as the barotropic momentum expression is part of the baroclinic momentum expression. To obtain the effect on the global kinetic energy, we multiply the eddy momentum forcing by $\bar{\mathbf{u}}$ and integrate over the domain, giving

$$\begin{aligned}
\iint_R \bar{\mathbf{u}} \cdot \hat{\mathbf{S}}_{BC} dx dy &= \kappa \iint_R \bar{\mathbf{u}} \cdot (\bar{\nabla} \cdot \mathbf{T}) dx dy, \\
&= \kappa \iint_R [\bar{\nabla} \cdot (\mathbf{T} \bar{\mathbf{u}}) - \frac{1}{2} \sigma (\zeta^2 + D^2 + \tilde{D}^2)] dx dy, \\
&= \kappa \int_{C=\partial R} (\mathbf{T} \bar{\mathbf{u}}) \cdot d\mathbf{l} - \frac{1}{2} \kappa \iint_R \sigma (\zeta^2 + D^2 + \tilde{D}^2) dx dy, \\
&= \frac{1}{2} \kappa \iint_R \sigma (\zeta^2 + D^2 + \tilde{D}^2) dx dy,
\end{aligned} \tag{6.23}$$

where to move from the first to the second line, since $\mathbf{T}$ is symmetric, we can write it as $\mathbf{T} = \begin{pmatrix} a & 0 \\ 0 & a \end{pmatrix} + \begin{pmatrix} b & c \\ c & -b \end{pmatrix} = a\mathbf{I} + b\mathbf{R}_+ + c\mathbf{R}_\times$, where a , b , and c are scalar functions. From this, it can be shown that $\bar{\nabla} \cdot (\mathbf{T} \bar{\mathbf{u}}) = \bar{\mathbf{u}} \cdot (\bar{\nabla} \cdot \mathbf{T}) - a\sigma - b\tilde{D} - cD$ (Anstey and Zanna, 2017). We used the divergence theorem to move from the second to the third line. The term involving $\mathbf{T} \bar{\mathbf{u}}$, originating from the Anstey and Zanna (2017) parameterisation, vanishes for both free-slip ($\zeta = 0$ causing $\mathbf{T} = 0$ at boundaries) and

no-slip ($u = v = 0$ at the boundaries) boundary conditions, conserving global kinetic energy.

The remaining term in Equation 6.23 - originating from the gradient of ζ^2 , D^2 , and $\tilde{D}^2$ - represents a net source or sink of kinetic energy from the eddy momentum forcing of $\hat{\mathbf{S}}_{BC}$. As the terms ζ^2 , D^2 , and $\tilde{D}^2$ are all positive-definite, it is the sign of the divergence $\sigma = \bar{u}_x + \bar{v}_y$ which determines whether the momentum parameterisation adds or removes global kinetic energy.

Therefore, if there is a local horizontal divergence ($\bar{\nabla} \cdot \bar{\mathbf{u}} < 0$) of the flow, then the baroclinic expression will add kinetic energy, proportional to the sum of the enstrophy and squared deformations multiplied by the divergence. In the case of a shallow water model, the horizontal divergence of the velocity field corresponds to changes in the free surface height; divergence leads to a decrease in free-surface height, and convergence corresponds to an increase in free-surface height. Changes in kinetic energy can, therefore, potentially be interpreted as interacting with the reservoir of potential energy.

The same source of kinetic energy occurs for the barotropic expression but without the deformation terms appearing in Equations 6.23. If the flow is non-divergent, both the barotropic $\hat{\mathbf{S}}_{BT}$ and baroclinic $\hat{\mathbf{S}}_{BC}$ expressions conserve global energy.

6.6.2 Eddy temperature forcing expression

As a reminder, the expression revealed by the RVM algorithm for the eddy temperature forcing is given by

$$\hat{S}_T = \kappa \bar{\nabla} \cdot \left(\frac{-\bar{u}_y \bar{u}_z - \bar{v}_y \bar{v}_z}{\bar{u}_x \bar{u}_z + \bar{v}_x \bar{v}_z} \right), \quad (6.24)$$

assuming all regression coefficients are approximately equal. If there is zero vertical variation in the velocity field, $\bar{\mathbf{u}}_z = 0$, then Equation 6.24 vanishes. The revealed expression is therefore intrinsically tied to vertical variations, which makes intuitive sense, as the expression is parameterising eddy fluxes that impact the temperature field, in turn impacting the density field.

The dependence of Equation 6.24 on vertical variability can be further examined by assuming that thermal wind balance holds, such that $\bar{u}_z \approx \frac{g}{\rho_0 f} \frac{\partial \bar{p}}{\partial y}$ and $\bar{v}_z \approx -\frac{g}{\rho_0 f} \frac{\partial \bar{p}}{\partial x}$. Here, thermal wind balance assumes the velocities $\bar{\mathbf{u}}$ and $\bar{\mathbf{v}}$ are in geostrophic balance.

This is justified, as the filtering and coarse-graining procedure removes the small-scale ($< 30\text{km}$) component of the velocity field, where the ageostrophic motion is more prominent.

Using thermal wind balance, Equation 6.24 becomes

$$\hat{S}_T = \frac{\kappa g}{\rho_0 f} \bar{\nabla} \cdot \begin{pmatrix} -\bar{u}_y \bar{\rho}_y + \bar{v}_y \bar{\rho}_x \\ \bar{u}_x \bar{\rho}_y - \bar{v}_x \bar{\rho}_x \end{pmatrix}, \quad (6.25)$$

where the eddy temperature flux is now the product of the horizontal derivatives of the velocity and density fields. If thermal wind balance is an appropriate assumption, then the revealed eddy temperature forcing relates to the interaction of lateral variations in the velocity and density fields.

Note that the coefficient κ has units of Cms , which physically is peculiar. However, after assuming thermal wind balance, the constant at the front of Equation 6.25 becomes $\frac{\kappa g}{\rho_0 f}$. If we then assume a linear equation for the density, $\rho = \rho_0 \alpha (T - T_0)$, the constant becomes $\frac{\kappa g \alpha}{f}$, which has units of m^2 . This is the same units as the coefficient of the Anstey and Zanna (2017) parameterisation and the momentum expressions revealed by the RVM algorithm. In Anstey and Zanna (2017), they found their coefficient to scale with the spatial resolution squared, Δx^2 . It may therefore be the case that the coefficient of the eddy temperature forcing - after assuming thermal wind balance and a linear equation of state - may also scale with the horizontal resolution squared Δx^2 .

We can re-write the eddy temperature flux in Equation 6.25 as the gradient of the density field $\bar{\nabla} \bar{\rho}$ being multiplied by a tensor, leading to

$$\hat{S}_T = \frac{\kappa g}{\rho_0 f} \bar{\nabla} \cdot \left[\begin{pmatrix} \bar{v}_y & -\bar{u}_y \\ -\bar{v}_x & \bar{u}_x \end{pmatrix} \bar{\nabla} \bar{\rho} \right], \quad (6.26)$$

$$= \frac{\kappa g}{2\rho_0 f} \bar{\nabla} \cdot \left[\begin{pmatrix} \sigma - \tilde{D} & \zeta - D \\ \zeta + D & \sigma + \tilde{D} \end{pmatrix} \bar{\nabla} \bar{\rho} \right], \quad (6.27)$$

$$= \frac{\kappa g}{2\rho_0 f} \bar{\nabla} \cdot \left[(s\mathbf{I} - \tilde{D}\mathbf{R}_+ + \zeta\mathbf{R}_\times + D\mathbf{R}) \bar{\nabla} \bar{\rho} \right], \quad (6.28)$$

where first we use the σ , ζ , D , and $\tilde{D}$ basis to re-write the velocity gradients within the tensor. We then obtain Equation 6.28 by re-writing the tensor itself using the following tensor basis $\mathbf{I}$, $\mathbf{R}_+$, $\mathbf{R}_\times$, and $\mathbf{R}$, given by

$$\mathbf{I} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \mathbf{R}_+ = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \mathbf{R}_\times = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \mathbf{R} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}. \quad (6.29)$$

The eddy temperature flux in Equation 6.28, in a similar way to the vorticity flux of Anstey and Zanna (2017) in Equation 6.18, can be interpreted as a larger-scale velocity field represented by $(\sigma \mathbf{I} - \tilde{D} \mathbf{R}_+ + \zeta \mathbf{R}_\times + D \mathbf{R})$, acting on the smaller-scale density field represented by $\overline{\nabla} \rho$. More work is required to further understand what the velocity gradient tensor $(\sigma \mathbf{I} - \tilde{D} \mathbf{R}_+ + \zeta \mathbf{R}_\times + D \mathbf{R})$ represents, as it is not simply the strain rate tensor, which is symmetric and given by $\mathbf{S} = \frac{1}{2}(\sigma \mathbf{I} + \tilde{D} \mathbf{R}_+ + D \mathbf{R}_\times)$.

We can gain additional information by determining the impact of $\hat{S}_T$ on the horizontal momentum equations. To do this, we can examine the horizontal momentum tendency equation in the residual-mean formulation (Ferrari and Plumb, 2003; Marshall et al., 2012). After this transformation, the effects of eddy buoyancy fluxes appear in the horizontal momentum tendency as the following

$$\frac{\partial \bar{\mathbf{u}}}{\partial t} + \dots = \frac{\partial}{\partial \bar{z}} \left(\frac{f_0}{N_0^2} \mathbf{k} \times \overline{\mathbf{u}'b'} \right), \quad (6.30)$$

where $\mathbf{k}$ is a unit vector in the vertical direction. For example, Gent and McWilliams (1990) parameterise eddy fluxes in the buoyancy equation, where the parameterisation acts to flatten isopycnals. However, in the residual-mean formulation, the parameterisation presents as an additional vertical diffusion of momentum (Gent and McWilliams, 1990; Greatbatch and Lamb, 1990). Assuming a linear equation of state, such that buoyancy is a linear function of temperature, we can go through the same procedure to see how our eddy temperature forcing in Equation 6.24 appears in the residual mean formulation.

By using our expression for the eddy temperature flux, we obtain the following term in the horizontal momentum equation, by using our discovered temperature flux in place of the eddy buoyancy flux $\overline{\mathbf{u}'b'}$ within Equation 6.30, leading to

$$\frac{\partial \bar{\mathbf{u}}}{\partial t} = -\kappa \frac{f_0 \alpha}{N_0^2} \frac{\partial}{\partial \bar{z}} \left[\begin{pmatrix} \bar{u}_x & \bar{v}_x \\ \bar{u}_y & \bar{v}_y \end{pmatrix} \begin{pmatrix} \bar{u}_z \\ \bar{v}_z \end{pmatrix} \right] \quad (6.31)$$

$$= -\kappa \frac{f_0 \alpha}{N_0^2} \frac{\partial}{\partial \bar{z}} \left[\overline{\nabla} \bar{\mathbf{u}}^T \frac{\partial \bar{\mathbf{u}}}{\partial \bar{z}} \right], \quad (6.32)$$

where $\overline{\nabla} \bar{\mathbf{u}}$ is the velocity gradient tensor, the superscript $()^T$ is the transpose, and noting that $\overline{\nabla} \bar{\mathbf{u}}^T \frac{\partial \bar{\mathbf{u}}}{\partial \bar{z}} \propto \mathbf{k} \times \overline{\mathbf{u}'b'}$. Using the residual mean formulation, our eddy temperature forcing therefore takes the form of an additional vertical diffusion of the horizontal momentum $\bar{\mathbf{u}}$, but with the diffusivity represented by the velocity gradient tensor $\overline{\nabla} \bar{\mathbf{u}}$. The values of the velocity gradient tensor are not bounded, so depending

on the flow field, Equation 6.32 could be diffusive or anti-diffusive. In other words, the vertical momentum flux can be both up- or down-gradient.

6.7 Summary and discussion

6.7.1 Summary

We have shown that eddy parameterisations can be constructed with machine learning algorithms which are interpretable, using the method of discovering equations from data, which use the fewest number of functions to accurately capture the underlying dynamics. In particular, we use a sparse Bayesian regression algorithm (Zhang and Lin, 2018) based on relevance vector machines (RVMs), that selects functions from a library of basis functions. The algorithm iteratively prunes the library by only retaining basis functions which have the most ‘certain’ regression coefficients. The algorithm has a single hyperparameter that requires tuning, namely the uncertainty threshold, above which basis functions are pruned.

The algorithm can be constrained to respect physical principles, by applying constraints to the functions that are included in the library. For the eddy momentum forcing, we constrain the function library such that the resulting equation can be written as the divergence of an eddy stress tensor $\hat{\mathbf{S}} = \overline{\nabla} \cdot \mathbf{T}$; this constraint ensures that global momentum can be conserved by setting the appropriate boundary conditions on the tensor.

The RVM algorithm reveals expressions for the eddy momentum and temperature forcing that capture over 50% of the variance in the validation data, in both barotropic and baroclinic primitive equation models. The tensors revealed by the RVM algorithm for eddy momentum forcing are symmetric, despite the algorithm being applied separately to the zonal and meridional components of the forcing. A symmetric tensor is important for vorticity conservation, as then $\nabla \times (\nabla \cdot \mathbf{T}) = \nabla \cdot (\nabla \times \mathbf{T})$, so the vorticity forcing can be written as the divergence of a vorticity flux; this implies vorticity can be globally conserved by prescribing appropriate boundary conditions on the vorticity flux through the divergence theorem. The RVM algorithm automatically tends towards functions that form the divergence of a vorticity flux.

The eddy momentum expressions, revealed by the RVM algorithm, contain the deformation-based parameterisation of Anstey and Zanna (2017), which captures up-

gradient momentum fluxes and dissipates enstrophy. The eddy momentum expressions revealed by the RVM algorithm contain additional terms, namely the gradient of enstrophy and the squared deformations squared; these terms relate to the non-linear gradient model (Nadiga, 2008; Nadiga and Bouchet, 2011), which assumes the eddy velocity can be Taylor expanded using the resolved velocities.

The revealed eddy temperature expression depends on the products of the horizontal and vertical gradients of the velocity field. Assuming thermal wind balance leads to an eddy temperature flux which depends on the product of a horizontal gradient of velocity and density. The flux can further be re-written to show a large-scale flow field acting on a smaller-scale density field. In the residual-mean formulation, the resulting eddy temperature forcing appears as an additional vertical diffusion of the horizontal velocity field, with the corresponding diffusivity tensor being proportional to the velocity gradient tensor.

6.7.2 Discussion

Using an iterative sparse regression to reveal expressions for eddy parameterisations, as opposed to training higher-complexity algorithms such as deep neural networks, has the following advantages:

- Can capture a significant proportion of the variance of the true eddy forcing while remaining interpretable.
- The interpretability makes inference possible, i.e., uncovered expressions could improve our physical understanding of turbulent mesoscale eddies.
- Fewer hyperparameters to tune within the iterative sparse regression.
- Computational cost of implementing is negligible, as resulting parameterisations comprise of functions that are simple products and derivatives.
- Closed-form expressions could more easily deal with coastlines and topography if implemented in a realistic global circulation model.

However, if the iterative sparse regression is capturing roughly 50% of the variance, then that leaves 50% not captured. It may be that certain situations require the power of higher complexity algorithms, such as the convolutional neural networks of

Chapter 5, in order to obtain 90+% captured variance, especially if interpretability is not important for the task at hand. Note that although the inner-workings of a neural network may be obscured by complexity, the architecture can still be constrained to respect physical principles such as conservation of momentum, just like with the RVM algorithm. We also found that the neural networks of Chapter 5 generalised well to models of different Reynolds numbers, whereas we have not yet conducted that same test for the expressions revealed by the RVM algorithm.

The equations we do uncover in this chapter are found using data which is close to the geostrophic limit. The resulting parameterisations may work well when the flow is predominantly in geostrophic balance. Therefore, it is not clear how well these data-driven algorithms will perform well beyond this limit.

Instead of relying on higher-complexity algorithms to improve the amount of variance captured, there are improvements that could be made to the RVM algorithm used in this chapter. For instance, we only used up to 2nd order spatial derivatives and polynomial products of the velocities $\bar{u}$ and $\bar{v}$, due to computational restrictions. Predictive skill of the RVM algorithm could be improved by including higher-order products and derivatives in the function library. In the case of the revealed eddy momentum expressions, higher-order basis functions could help move beyond the functional form of Anstey and Zanna (2017) and the non-linear gradient model.

The library of basis functions could also be improved by including memory effects, i.e., information at previous time-steps. In this chapter, the basis functions and the target quantities were all calculated at the same time-slice. We know that memory effects are important for parameterising mesoscale eddies (Mana and Zanna, 2014), so including temporal derivatives of $\bar{u}$ and $\bar{v}$ in the function library could further improve the RVM algorithm.

Then there is the RVM algorithm itself. We adapted the RVM algorithm from Zhang and Lin (2018) and found it to be more reliable than the sequential threshold ridge regression of Rudy et al. (2017). However, there may be equation-discovery algorithms that outperform the RVM algorithm. The approach of discovering equations from data is still relatively young, and as such, many new methods are being proposed. For instance, using neural networks to select basis functions (Champion

et al., 2019; Xu et al., 2019), which may utilise the high-complexity of neural networks while still producing an interpretable expression for the eddy parameterisation.

One final criticism of the RVM algorithm, which also applies to the convolutional neural networks of Chapter 5, is that all measures of success have so far been diagnostic - they have been calculated using data already produced by a high-resolution model. An important assessment of eddy parameterisations is how they behave once implemented in a model, which we have not yet tested.

Chapter 7

Prognostic evaluation of eddy momentum parameterisations

7.1 Introduction

Whether an eddy parameterisation is constructed in a physics-driven, data-driven or hybrid approach, all closures must be evaluated in a prognostic fashion. In other words, eddy parameterisations must be implemented back into a truncated or reduced-resolution ocean model. Implementation is necessary to see if the scale-interaction between the parameterised sub-grid processes and the resolved dynamics are accurately represented. The skill of the parameterisation is determined through global metrics, such as the higher-order statistical moments of the velocity field.

In addition, to assess how the physics of the model change, we also need to ascertain the impact of the eddy parameterisation on numerical stability. A parameterisation that perfectly represents the effects of eddies is of little use if it causes the model to diverge every time it is implemented. The computational cost also requires consideration, as a parameterised ocean model becomes counterproductive if the computational cost approaches that of a higher resolution model.

The convolutional neural networks and RVM algorithm of Chapters 5 and 6 learn from data in a diagnostic manner. Diagnostic means the data-driven algorithms are being applied *after* the training data has been formed, from either models of observations. We now want to test these parameterisations in a prognostic manner. A prognostic evaluation involves numerically integrating the model with the parameterisation implemented, allowing the model to interact with the parameterisation

and vice versa. A key issue of data-driven parameterisations is that diagnostic performance does not guarantee prognostic performance (Rasp, 2019), with the latter being more important for future implementation in climate models. A data-driven parameterisation may capture a significant proportion of the variance during training, but whether it actually improves a low-resolution model cannot be determined until implementation.

A note on the nomenclature within the literature: depending on the branch of science, when referring to the performance of a data-driven algorithm, the terms diagnostic and prognostic are sometimes used interchangeably with offline and online, or a-priori and a-posteriori. In this chapter, we will only use the terms diagnostic and prognostic.

Data-driven parameterisations are more frequently being evaluated in a prognostic manner, particularly in atmospheric models (e.g. Brenowitz and Bretherton, 2018; Rasp et al., 2018; O’Gorman and Dwyer, 2018). An interesting example of a neural network turbulence closure comes from Maulik et al. (2019), where using physically-motivated input features, based on the closures of Smagorinsky (1963) and Leith (1968), significantly improved prognostic performance.

Parameterisations are typically constructed with the goal of representing a physical process, with numerical stability not considered at first. The model diverging to infinity, i.e., ‘blowing up’, is therefore a common issue when attempting to implement a new parameterisation. The issue is exacerbated for some data-driven parameterisations, where the impact on stability criterion is obscured by the black-box nature of the algorithm, such as neural networks - how is the Courant-Friedrichs-Lewy (CFL) condition supposed to be evaluated for an algorithm with hundreds of thousands of parameters? There are additional considerations, such as whether an implicit or semi-implicit numerical scheme should be used with a particular parameterisation, or what order advection scheme should be used.

Numerical stability can be partially addressed by enforcing physical constraints, in particular conservation laws. With regards to parameterising eddy momentum fluxes, respecting global conservation of vorticity and momentum not only improves the representation of the physics, but may also improve the stability of the model. If a data-driven parameterisation continuously inputs global momentum at each time-

step of the model, the resulting flow field will either become physically unrealistic or numerically diverge. Respecting physical principles is therefore an important consideration during prognostic tests (Beucler et al., 2019b,a).

In this chapter, we conduct a prognostic evaluation of a suite of eddy momentum parameterisations. In Section 7.2, we introduce each parameterisation to be tested and the corresponding ocean model in which they will be implemented. Before discussing the implementation, we summarise the diagnostic performance of each parameterisation in the training data in Section 7.3. In Section 7.4, we investigate the prognostic performance of each parameterisation, by assessing the energetics and resulting flow field of each model. In Section 7.5, we summarise the results of implementing each parameterisation and the implications for future climate models.

7.2 Data and methods

In this section, we describe the three eddy momentum parameterisations to be tested and the shallow water model in which they will be implemented.

7.2.1 Candidate momentum parameterisations

Two of the candidate parameterisations are data-driven: the first being a fully-convolutional neural network, and the second being the barotropic momentum expression (Equation 6.5) revealed by the RVM algorithm from Chapter 6. We include a third parameterisation for comparison, namely the deformation-based parameterisation of Anstey and Zanna (2017), as it also appears within the barotropic and baroclinic momentum expressions revealed by the RVM algorithm. We describe each of these three parameterisations below, with regards to the parameters they use and how they are implemented.

Parameterisation 1: Fully-convolutional neural network

We initially wanted to implement the convolutional neural networks trained on the quasi-geostrophic model in Chapter 5. However, this produces the followings issues:

- We cannot conduct a fair comparison with the other data-driven parameterisation (from the RVM algorithm of Chapter 6) due to the training datasets

originating from dynamically different models, in addition to using different definitions of the eddy momentum forcing.

- The convolutional neural networks of Chapter 5 were constructed to use inputs and outputs on a 7.5km grid, which does not match the resolution of the coarse-resolution 30km grid of the model we intend to implement in.
- To form a prediction of the eddy momentum forcing with the convolutional neural networks of Chapter 5, multiple predictions have to be made and tiled together to form a coherent prediction for the full domain. This is because the convolutional neural networks of Chapter 5 only make predictions for a 40×40 grid point area at a time - predictions for overlapping 40×40 areas are combined together. This dramatically increases the computational cost of any future implementation.

These obstacles make implementing the convolutional neural networks of Chapter 5 impractical. However, we still want a prognostic test of a neural network eddy momentum parameterisation. Hence, we take what we learnt from Chapter 5 and train a whole new convolutional neural network, using the same training data from the barotropic primitive equation model used by the RVM algorithm in Chapter 6, allowing for a more rigorous comparison. We use a training-validation split of 50:50, using 1000 time-slices of model output, with time-slices taken every 4 days - the exact same as the RVM algorithm. From the lessons of Chapter 5, we can also make the implementation more practical, as well as physically-constraining the neural network architecture in an improved manner.

The architecture of the new convolutional neural network to be implemented and studied in this chapter is shown in Figure 7.1. The network takes maps of both velocity components $\bar{u}$ and $\bar{v}$ as the inputs, and then through four convolution layers simultaneously predicts both components of the eddy momentum forcing $\hat{S}_x$ and $\hat{S}_y$. The architecture contains only convolution layers and no fully-connected layers, and hence these types of networks are referred to as fully-convolutional. The advantage of using a fully-convolutional architecture is that the network can take inputs of *any* size, unlike networks with fully-connected layers which require all inputs to have the

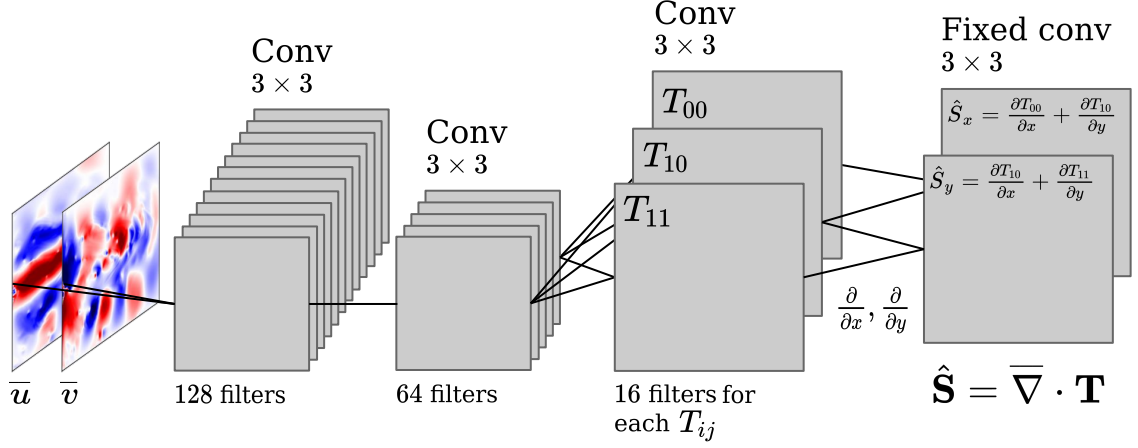


Figure 7.1: A schematic of the architecture of the fully-convolutional neural network, which takes the velocity fields u and v as multi-channel inputs, and simultaneously predicts the two components of the eddy momentum forcing S_x and S_y . The architecture is physically-constrained such that the final result is always the divergence of a symmetric eddy stress tensor.

same shape. This is powerful for training and predicting on multiple ocean models with varying domain sizes.

Note that one disadvantage of using a fully-convolutional architecture (in addition to the computational cost) is that the network can struggle to differentiate boundaries from the ocean interior. The filters of each convolution are shared across each point in space, i.e., the result of a convolution at a point in the domain (x_0, y_0) will produce the same result when applied to a different point (x_1, y_1) - this would not be the case for a fully-connected layer. Thus, if (x_0, y_0) and (x_1, y_1) represent points at the boundary and in the interior respectively, the fully-convolutional neural network will assume the dynamics are the same for both points. During initial experimentation, we found that this degraded performance at the boundary, producing spuriously large biases in the predicted eddy momentum forcing. Attempts to remove the spurious patterns failed, so we decided to force the eddy momentum forcing to be zero within four grid points of the boundaries - this is carried out within the architecture of the neural network, so no post-processing is involved.

The architecture is physically constrained such that the activation maps of the

third convolution layer represent the elements of an eddy stress tensor $\mathbf{T}$. Note that we assume the eddy stress tensor is symmetric, $T_{10} = T_{01}$, such that only three tensor elements need predicting. The final convolution layer takes the spatial derivatives of the previous tensor elements, using fixed filters representing central-difference stencils. This final fixed convolution layer, that takes spatial derivatives, forms the two outputs $\hat{S}_x = \frac{\partial T_{00}}{\partial x} + \frac{\partial T_{10}}{\partial y}$ and $\hat{S}_y = \frac{\partial T_{10}}{\partial x} + \frac{\partial T_{11}}{\partial y}$. Note that the mean-squared error loss function is a function of the final eddy momentum forcings $\hat{S}_x$ and $\hat{S}_y$, and not the eddy tensor $\mathbf{T}$, i.e.,

$$L = \frac{1}{N} \sum_{i=1}^N [(S_x^{(i)} - \hat{S}_x^{(i)})^2 + (S_y^{(i)} - \hat{S}_y^{(i)})^2], \quad (7.1)$$

where N is the total number of training samples and i represents the i th sample. By physically-constraining the architecture to form the elements of a symmetric eddy stress tensor, global momentum and vorticity conservation can be achieved by prescribing appropriate boundary conditions.

We also do not use bias parameters in any of the convolution layers. The bias parameters are trainable values that are added to each point in space after the convolution operation has taken place, but before being passed through the activation function. Having no bias parameters enforces that $\hat{S}_x = \hat{S}_y = 0$ when $\bar{u} = \bar{v} = 0$ everywhere in the domain. This must be the case physically, because if there is no large-scale flow, then there is nothing for small-scale eddies to interact and exchange momentum with.

The full details of the architecture and training are shown in Table 7.1. The hyperparameters of the architecture, such as the number of convolution layers and the number of filters, were chosen by experimenting with numerous configurations and examining the impact of the R^2 coefficient in the validation data.

Parameterisation 2: Barotropic RVM algorithm expression

We use the expression revealed by the RVM algorithm when applied to data from the barotropic primitive equation model in MITgcm. As previously presented in Chapter 6, the barotropic momentum expression takes the following form

$$\hat{\mathbf{S}}_u = \kappa \bar{\nabla} \cdot \begin{pmatrix} \zeta^2 - \zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta^2 + \zeta D \end{pmatrix}, \quad (7.2)$$

Parameter description	Parameter value
Input variables	$\bar{u}, \bar{v}$
Input size	128×128
Output variables	$\hat{S}_x, \hat{S}_y$
Output size	128×128
Number of convolution layers	3 trainable, 1 fixed
Number of filters for each convolution layer	128, 64, 48, 2
Filter stride for each convolution layer	1, 1, 1, 1
Filter size of each convolution layer	$3 \times 3, 3 \times 3, 3 \times 3, 3 \times 3$
Boolean bias specifier for each convolution layer	FALSE, FALSE, FALSE, FALSE
Activation function for each convolution layer	SELU, SELU, SELU, linear
Fixed-value central-difference filters of final convolution layer to calculate spatial derivatives	$\begin{pmatrix} 0 & 0 & 0 \\ -0.5 & 0 & 0.5 \\ 0 & 0 & 0 \end{pmatrix}, \begin{pmatrix} 0 & 0.5 & 0 \\ 0 & 0 & 0 \\ 0 & -0.5 & 0 \end{pmatrix}$
Loss function	Mean-squared error
Optimiser	Adam
Learning rate	0.0001
Batch size	1
Number of training epochs	1000

Table 7.1: Parameters of the architecture and training of the fully-convolutional neural network illustrated in Figure 7.1.

where we make the approximation that the regression coefficients are approximately equal $c_i \approx \kappa$, due to the coefficient of variation of 14.2%. The parameter κ is set to the mean value of the regression coefficients found in Chapter 6, i.e., $\kappa = -4.87 \times 10^8 \text{m}^2$.

Parameterisation 3: Deformation-based Anstey and Zanna (2017)

We compare the previous two data-driven parameterisations with the previously discussed physics-driven deformation-based parameterisation of Anstey and Zanna (2017), defined by

$$\hat{\mathbf{S}}_u = \kappa \bar{\nabla} \cdot \begin{pmatrix} -\zeta D & \zeta \tilde{D} \\ \zeta \tilde{D} & \zeta D \end{pmatrix}. \quad (7.3)$$

There are two main reasons for this choice. First, it appears within the barotropic momentum expression revealed by the RVM algorithm, so we want to determine the impact of the additional terms that appear in Equation 7.2, namely the gradient of enstrophy.

Secondly, we know from the study of Anstey and Zanna (2017) that Equation 7.3 captures up-gradient momentum fluxes, which allows for a more appropriate comparison with dynamics learnt by the two data-driven parameterisations. Eddy momentum fluxes are not always down-gradient, which traditional closures (e.g. Smagorinsky,

1963; Leith, 1968) assume. Also, coarse-resolution ocean models already include a dissipative momentum closure (e.g. a Laplacian momentum viscosity) to improve numerical stability.

We set the parameter of Equation 7.3 to $\kappa = -4.87 \times 10^8 \text{m}^2$, the exact same as the parameter of the expression from the RVM algorithm, allowing for a fair comparison. This also matches the values found empirically in Anstey and Zanna (2017), where κ was found to vary between 0 and $-8 \times 10^8 \text{m}^2$ across the domain of their primitive equation model (Figure 6b of Anstey and Zanna, 2017).

7.2.2 Ocean model to parameterise

The two data-driven parameterisations are trained on data from a high-resolution barotropic primitive equation model in MITgcm. Ideally, to evaluate the prognostic performance, we would implement the parameterisations back into the same barotropic primitive equation model, but at a coarser resolution. However, the MITgcm model numerically integrates the primitive equations in Fortran. Unfortunately, we did not have time to implement the fully-convolutional neural network parameterisation in Fortran, and therefore decided to implement using a higher-level programming language. Implementation in Fortran is a future goal however.

As Python was used to train and save the fully-convolutional neural network, we use a shallow water model in Python as our prognostic test-bed of the three candidate momentum parameterisations. Despite being significantly slower compared to Fortran, using Python allows us to more easily implement and experiment with each parameterisation.

The shallow water model, described in more detail in Section 3.1, is similar to the barotropic primitive equation model: they share the same three prognostic variables - the horizontal velocities and free-surface height - and are run in a double-gyre configuration with a constant wind forcing. The double-gyre configuration is used to emulate western boundary currents and their eastward-extending jets, such as the Gulf Stream and Kuroshio Extension (Schmitz Jr and Holland, 1986; Berloff et al., 2002; Mana and Zanna, 2014; Cooper and Zanna, 2015).

To test each parameterisation, we implement them into a 30km resolution version of the shallow water model. With the parameterisation implemented, we spin-up the

model from rest for 10 years, and then run for an additional 10 years to produce the final dataset for analysis. At each time-step, the components of the eddy momentum forcing $\hat{S}_x$ and $\hat{S}_y$ are calculated using the velocities on the coarse-grid, $\bar{u}$ and $\bar{v}$, using the prescribed parameterisation; $\hat{S}_x$ and $\hat{S}_y$ are then simply added to the prognostic equations for $\bar{u}$ and $\bar{v}$ as an additional forcing.

For all three parameterisations, to guarantee conservation of global momentum and vorticity, we set the values of the eddy stress tensor to zero at the boundaries of the domain. Due to the divergence theorem, this ensures the contribution of $\hat{S}_x$ and $\hat{S}_y$ to the global momentum and vorticity budgets vanishes at each time-step. Note that this does not prevent the parameterisations acting as a source or sink of kinetic energy.

7.3 Diagnostic evaluation

Before we proceed with the prognostic evaluation, we first summarise the performance of each parameterisation in the validation data from the barotropic primitive equation model in MITgcm. This will allow us to compare the diagnostic performance of the newly-trained fully-convolutional neural network with the true eddy momentum forcing, as well as the other parameterisations.

Figure 7.2 presents the first four statistical moments - mean, standard deviation, skewness, kurtosis - of the true S_x and the predicted $\hat{S}_x$ from each parameterisation over the validation period. Figure 7.2 also includes the Pearson correlation coefficient between each predicted $\hat{S}_x$ and the true S_x . We include the higher-order statistical moments, skewness and kurtosis, as these are important features that an eddy momentum forcing should capture (Mana and Zanna, 2014). Also, the data-driven parameterisations are trained to minimise the mean-squared error, so their predicted standard deviation should match the true standard deviation (to some extent) by construction. Comparing the skewness and kurtosis of the parameterisations is a more robust test, as those statistical moments are not directly linked to the mean-square error loss function.

For brevity, we will refer to the Anstey and Zanna (2017) deformation-based parameterisation (Equation 7.3) as AZ17; the barotropic momentum expression re-

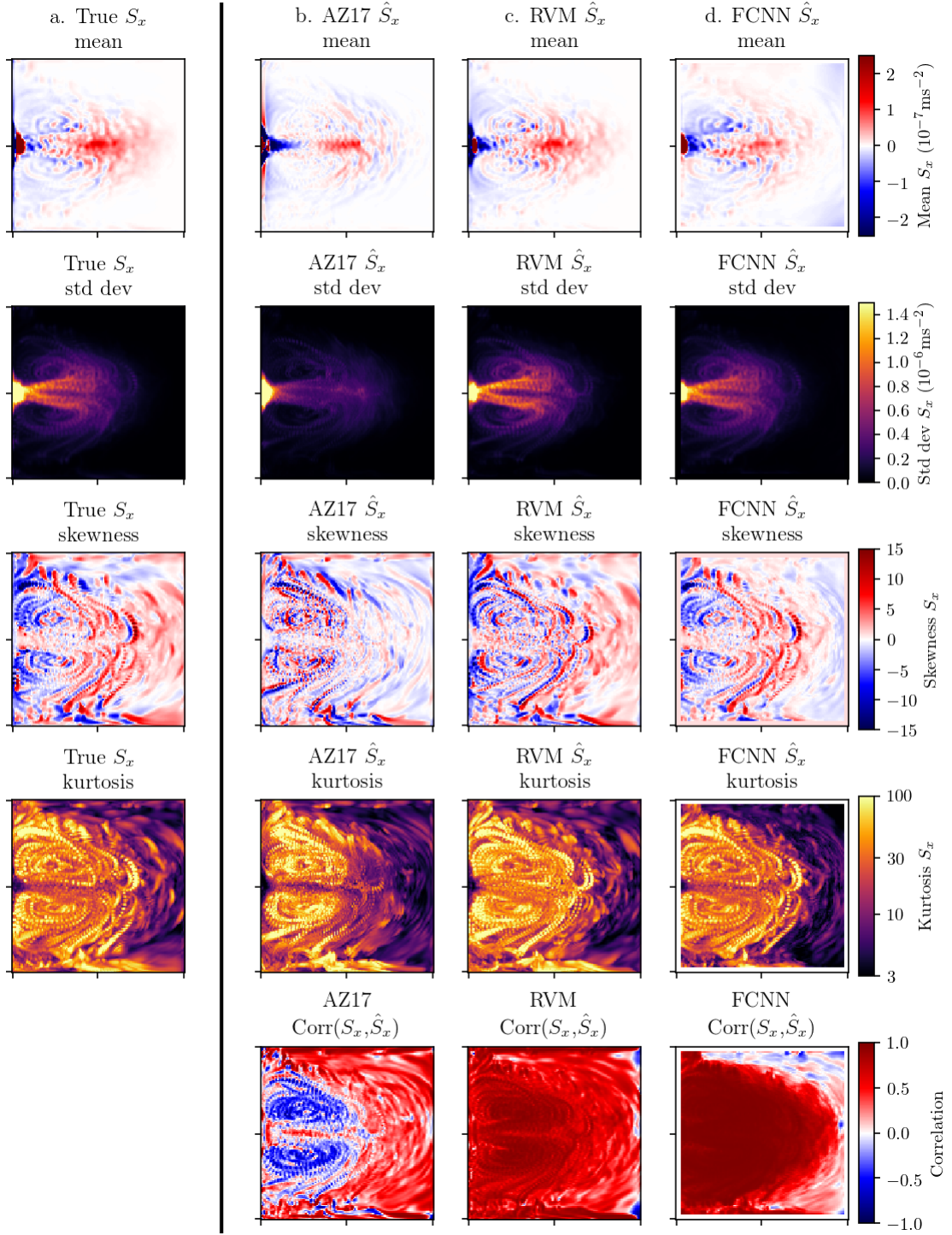


Figure 7.2: Within the validation data of the barotropic MITgcm model, comparisons of the diagnostic predictions of three momentum parameterisations: the deformation parameterisation of Anstey and Zanna (2017), the barotropic RVM-algorithm expression, and the fully-convolutional neural network described in Section 7.2.1. For each parameterisation, we compare the mean, standard deviation, skewness, and kurtosis, of the zonal eddy momentum forcing $\hat{S}_x$ with the true forcing S_x .

vealed by the RVM algorithm (Equation 7.2) as the RVM expression; and the fully-convolutional neural network parameterisation as FCNN.

In the time-mean maps of $\hat{S}_x$ (top row of Figure 7.2), the RVM expression most accurately captures the spatial patterns. The fully-convolutional neural network also captures the majority of the spatial patterns of the true time-mean, but exhibits a negative bias at the eastern boundary. The AZ17 parameterisation loosely captures the negative values at the western boundary and positive values in the interior, but struggles to capture the finer small-scale patterns of the true time-mean. Similar results hold for the standard deviation (second row of Figure 7.2): the RVM expression and FCNN reproduce the true standard deviation almost exactly, with differences only visible close to the western boundary. Whereas the AZ17 standard deviation underestimates the true standard deviation by 50% in the ocean interior.

The maps of skewness of the true S_x and predicted $\hat{S}_x$ (third row of Figure 7.2) show complex patterns with significant small-scale variability. Both the RVM and FCNN capture the majority of the small-scale variability of the true skewness, particularly in the flanks of the jet. However, the FCNN does not get the correct sign of the skewness near the eastern boundary, whereas the RVM expression does. The AZ17 parameterisation does accurately capture the true skewness at the northern and southern boundaries, but not in the flanks of the jet.

With regards to the maps of kurtosis (fourth row of Figure 7.2), again the RVM expression and FCNN do well in reproducing the small-scale variability of the true kurtosis. Despite underestimating the true standard deviation, the kurtosis of the AZ17 parameterisation kurtosis is not dissimilar to the true kurtosis. The correlations maps of each parameterisation (bottom row of Figure 7.2) are consistent with the statistical moments: the performance of AZ17 deteriorates to a negative correlation in the jet flanks, the RVM expression is positive almost everywhere in the domain with $r > 0.5$, and the FCNN performs particularly well within the jet with $r > 0.8$ but then degrades close to zero moving towards the eastern boundary.

Figure 7.3 shows the same statistical moments and correlation maps of the meridional component of eddy momentum forcing. These are the diagnostic predictions within the validation data, the same as the zonal component shown in Figure 7.2. A similar story emerges, with the RVM expression and FCNN capturing the correct

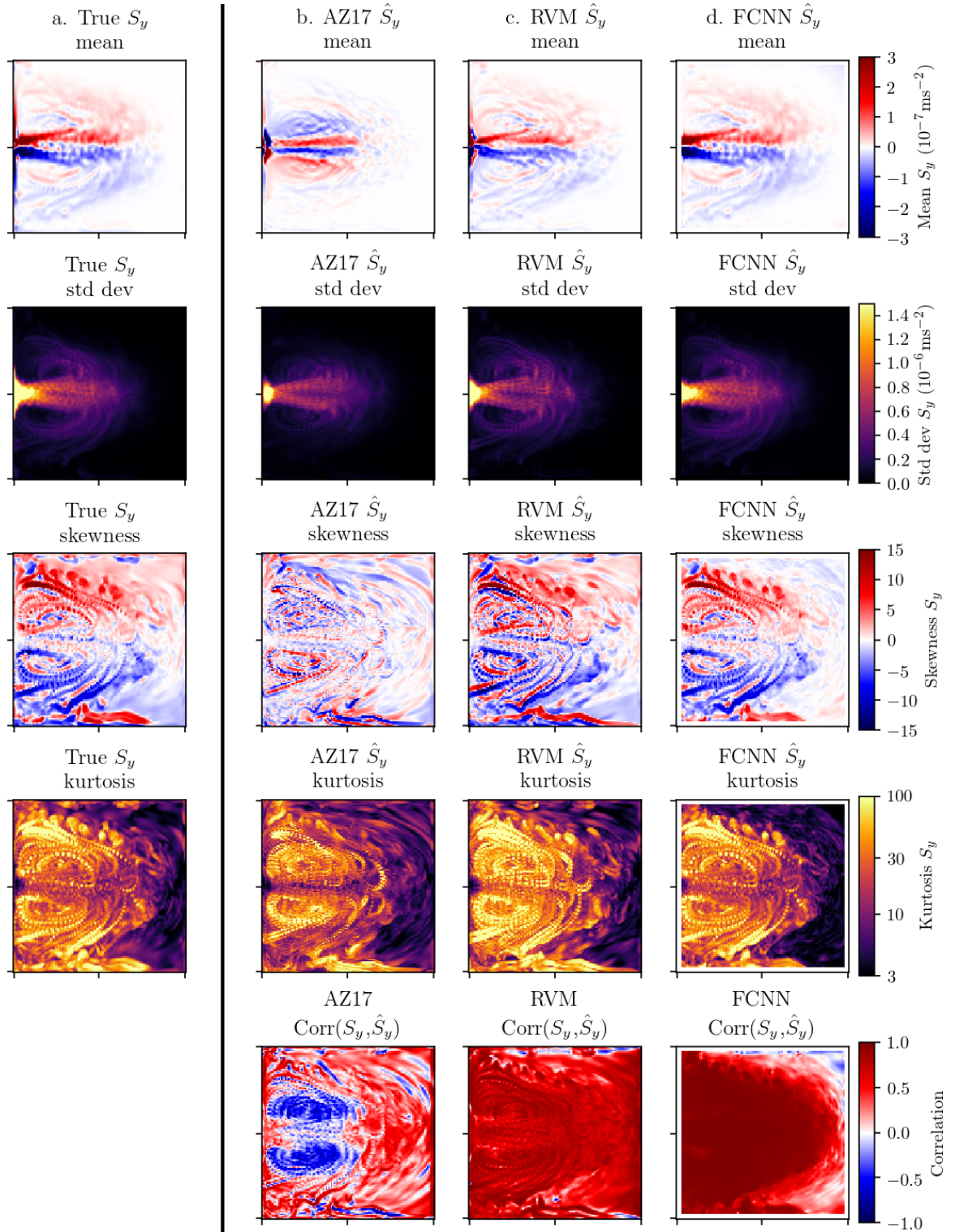


Figure 7.3: The same as in Figure 7.2, but comparing the meridional component of the eddy momentum forcing $\hat{S}_y$ with the true S_y .

spatial patterns in all the statistical moments. The reduced performance of AZ17 in the jet flanks is more visible in the meridional component, especially in the predicted time-mean and skewness, where significant differences exist compared to the truth. Again the correlation of the FCNN is high within the jet and close to the western boundary with $r > 0.8$ but degrades at the eastern boundary. The correlation of the RVM expression is more uniformly positive with $r > 0.5$. The correlation of AZ17 becomes negative in the jet flanks but moderately positive in the remainder of the domain ($r \approx 0.3$).

Overall, in a diagnostic analysis, we see that the RVM expression and FCNN capture the higher-order moments of the skewness and kurtosis, and generally outperform the AZ17 expression. In terms of predictive skill, the FCNN captures almost all of the variance in the vicinity of the jet, but this high skill is not consistent across the domain, particularly at the eastern boundary. The predictive skill of the RVM expression is not as high as the FCNN within the jet, but is significantly more consistent across the domain, with fewer patches of zero or negative correlation.

7.4 Prognostic evaluation

We now test how the diagnostic performance translates to prognostic performance when implemented into a model. As discussed in Section 7.2.2, we run a shallow water model at 30km resolution for each parameterisation. We spin-up each model from rest for 10 years and then run for an additional 10 years for data generation. In both the spin-up and data generation periods, we store time-slices of all fields at 5-day intervals.

To compare with the parameterised models, we also run models at 30km and 3.75km resolution with no parameterisation, with the latter acting as the high-resolution truth. In total there are five models, three of them containing an eddy momentum parameterisation. In this section, we compare the flow field and the energetics of all models.

Note that the parameterisations initially encountered issues when first implemented: the models with the RVM expression and AZ17 parameterisations diverged to infinity after a year, while the FCNN produced un-physical flow fields. Specifically, the FCNN over-energised the model, continually inputting energy into the large-scales

of the model through an inverse cascade. Eventually, instabilities spread over the entire domain and the current speeds approached 10ms^{-1} . So despite the FCNN not crashing the model, the dynamics were not realistic - the instabilities generated by the FCNN are discussed further in Section 7.4.2. We wondered whether the complexity of the convolutional layers allowed the FCNN parameterisation to be more resistant to numerical instabilities - at least relative to the AZ17 and RVM parameterisations.

To avoid the models diverging, or over-energising the flow, we attenuated the strength of each parameterisation, i.e., at each time-step we simply multiply $\hat{\mathbf{S}}_{\mathbf{u}}$ by an attenuation coefficient α with $0 < \alpha < 1$. We wanted to use as high an attenuation coefficient as possible for each parameterisation, to give each model the best chance of reproducing the dynamics of the high-resolution model. Through trial and error, we used attenuation levels of 50%, 60%, and 70% ($\alpha = 0.5, 0.6$, and 0.7) for the RVM, AZ17, and FCNN parameterisations respectively. These attenuation levels were found to stop the models diverging, as well as stopping the energy of the model increasing above that of the high-resolution reference model.

It could be argued that a strict comparison of the parameterisations would involve using the same attenuation level for each one. However, the end goal of a parameterisation is to implement within a model in order to improve the representation of the dynamics. Therefore, we should judge a parameterisation on its ability to improve a model, and if it can be implemented at a higher attenuation level, then that becomes an advantage of that parameterisation and should not be ignored.

In Section 7.4.2, we include a test of the sensitivity to the level of attenuation and the energetics of each model. Aside from the sensitivity test, all subsequent analysis uses attenuation levels of 50%, 60%, and 70% for the RVM, AZ17, and FCNN parameterisations respectively.

7.4.1 Statistical moments of velocity fields

When integrating forward in time, the eddy momentum forcing interacts with other terms within the model, such as advection and dissipation. Thus examining the $\hat{S}_x$ and $\hat{S}_y$ terms in isolation does not reveal how the model as a whole has changed by introducing the parameterisation. Instead, we look at the statistical moments of

the velocity field, which is impacted by all terms in the momentum time-tendency equations.

The mean, standard deviation, skewness, and kurtosis of the zonal velocity u are shown in Figure 7.4, and are calculated in an Eulerian manner over a 10 year period. The time-mean zonal velocity shows that all models exhibit an eastward jet at the western boundary, with basin-wide gyres in the lower and upper halves of the domain. There is less small-scale variability in the zonal velocity of the 30km model compared to the 3.75km model. The 30km model also has a weaker return flow at the northern and southern boundaries.

None of the three parameterisations produce drastic changes in the maps of the time-mean zonal velocity u (first column of Figure 7.4) within this model. The RVM and FCNN parameterisations slightly improve the return flow at the northern and southern boundaries, making it closer to the high-resolution model. They also appear to produce multiple distinct zonal jets in the centre of the domain, which are not present in the high-resolution run - it is not clear why this happens specifically for the RVM and FCNN parameterisations.

Despite minimal changes in the time-mean, all parameterisations have the largest impact on the standard deviation, i.e., the temporal variance, shown in the second column of Figure 7.4. All three parameterisations increase the amplitude of the standard deviation in the interior, with the spatial patterns more closely resembling the high-resolution model. The standard deviation within the jet also increases.

The skewness and kurtosis, the third and fourth columns of Figure 7.4, of the high-resolution model displays more fine-scale structures, particularly at the western boundary. The peaks of kurtosis in the high-resolution model are also larger in amplitude compared to the 30km model - this could be due to higher resolution, as the dynamics become less wave-like and extreme turbulent effects become more frequent. It is difficult to discern the exact effect of the parameterisations on the skewness and kurtosis of the velocity field. There is a visible change in the skewness is within the jet at the western boundary: the parameterisations decrease the size of the negative lobe that is present in the 30km model, but not the 3.75km model.

A similar story emerges for the statistical moments of the meridional velocity v , shown in Figure 7.5. There are some minor variations in the time-mean (first column

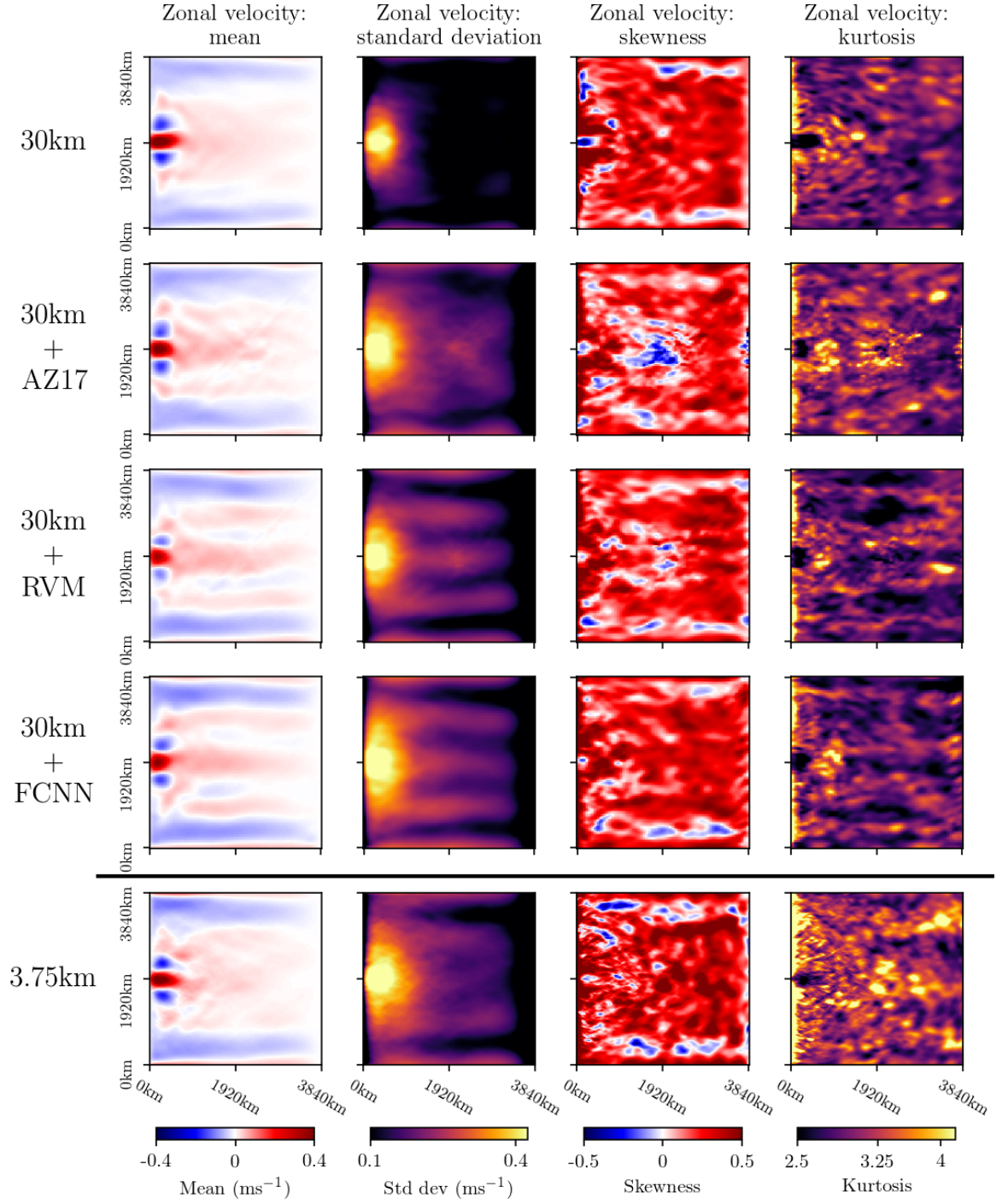


Figure 7.4: The mean, standard deviation, skewness and kurtosis of the zonal velocity field u for five different shallow water models. Three of the models contain either the AZ17, RVM or FCNN parameterisation. The two remaining models are run at 30km and 3.75km resolution with no parameterisation, representing the coarse- and high-resolution cases for comparison. The spatial maps are a function of longitude (x-axis) and latitude (y-axis).

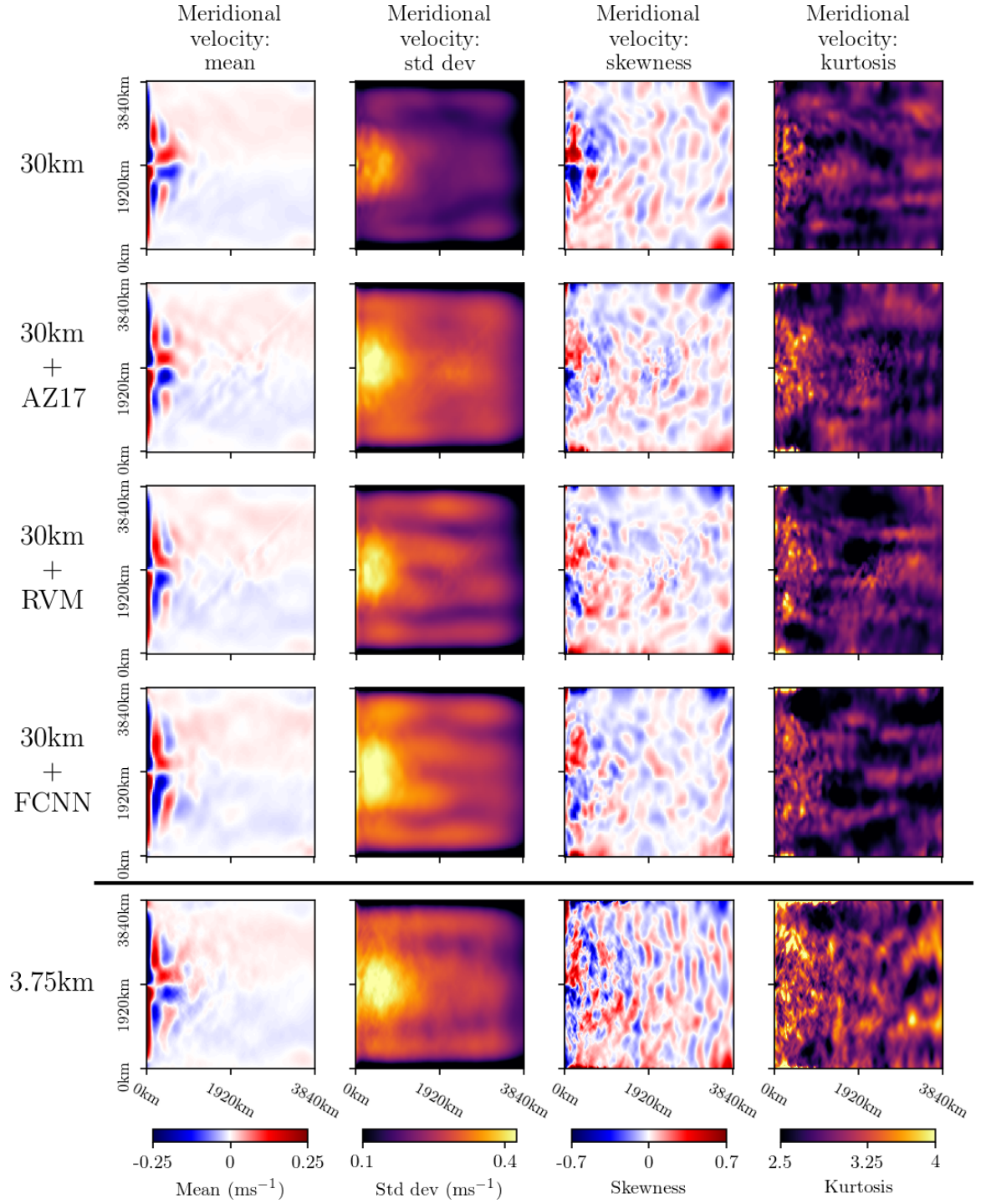


Figure 7.5: The mean, standard deviation, skewness, and kurtosis, the same as Figure 7.4, but for the meridional component v of the velocity field. The spatial maps are a function of longitude (x-axis) and latitude (y-axis).

of Figure 7.5) for the AZ17 parameterisation within the jet, but overall the parameterisations have almost no impact. Each parameterisation does however increase the amplitude of the standard deviation (column two of Figure 7.5), becoming more similar to the high-resolution model. Again, from the spatial maps, it is difficult to determine how the parameterisations are affecting the skewness and kurtosis (third and fourth columns of Figure 7.5 respectively) of the meridional velocity.

To better understand how each parameterisation is impacting higher-order statistical moments of the velocity field, we now directly examine the probability distributions functions (PDFs). Specifically, we estimate the PDFs of the zonal velocity field in three dynamically-distinct regions of the model, illustrated in Figure 7.6a. The first region is on the turbulent western boundary, the second on the more wave-like eastern boundary, and the third on the southern boundary within the lower gyre. For each region, we estimate the PDF of the zonal velocity for all five models, with the PDFs of regions 1, 2, and 3 show in Figure 7.6b, 7.6c, and 7.6d respectively.

In Figure 7.6, all the PDFs are approximately Gaussian, and hence the difficulty in discerning differences in the spatial maps of skewness and kurtosis in Figure 7.4. In all three regions, compared to the 30km model with no parameterisation (black curves), the high-resolution model (cyan curves) has more probability in the tails of the PDF, implying extreme values of the zonal velocity are more frequent.

The effect of the parameterisations is to redistribute probability from the peak to the tails; this effect is clearest in region 3 (Figure 7.6d), where the parameterisations all clearly produce a wider distribution, with the tails becoming more similar to the high-resolution. Therefore, the parameterisations are in fact improving the high-order statistics of the velocity field, despite these changes not being apparent in the skewness and kurtosis maps of Figure 7.4.

The FCNN parameterisation has the largest effect on the PDFs out of the three parameterisations, with the zonal velocity PDF of region 1 becoming very similar to the high-resolution model (Figure 7.6b). In region 3, it is more visibly clear that the FCNN is producing the largest redistribution of probability from the peak to the tails (Figure 7.6d). In region 1, the AZ parameterisation clearly outperforms the RVM algorithm, though it is not clear whether it is the functional form or attenuation level driving this difference.

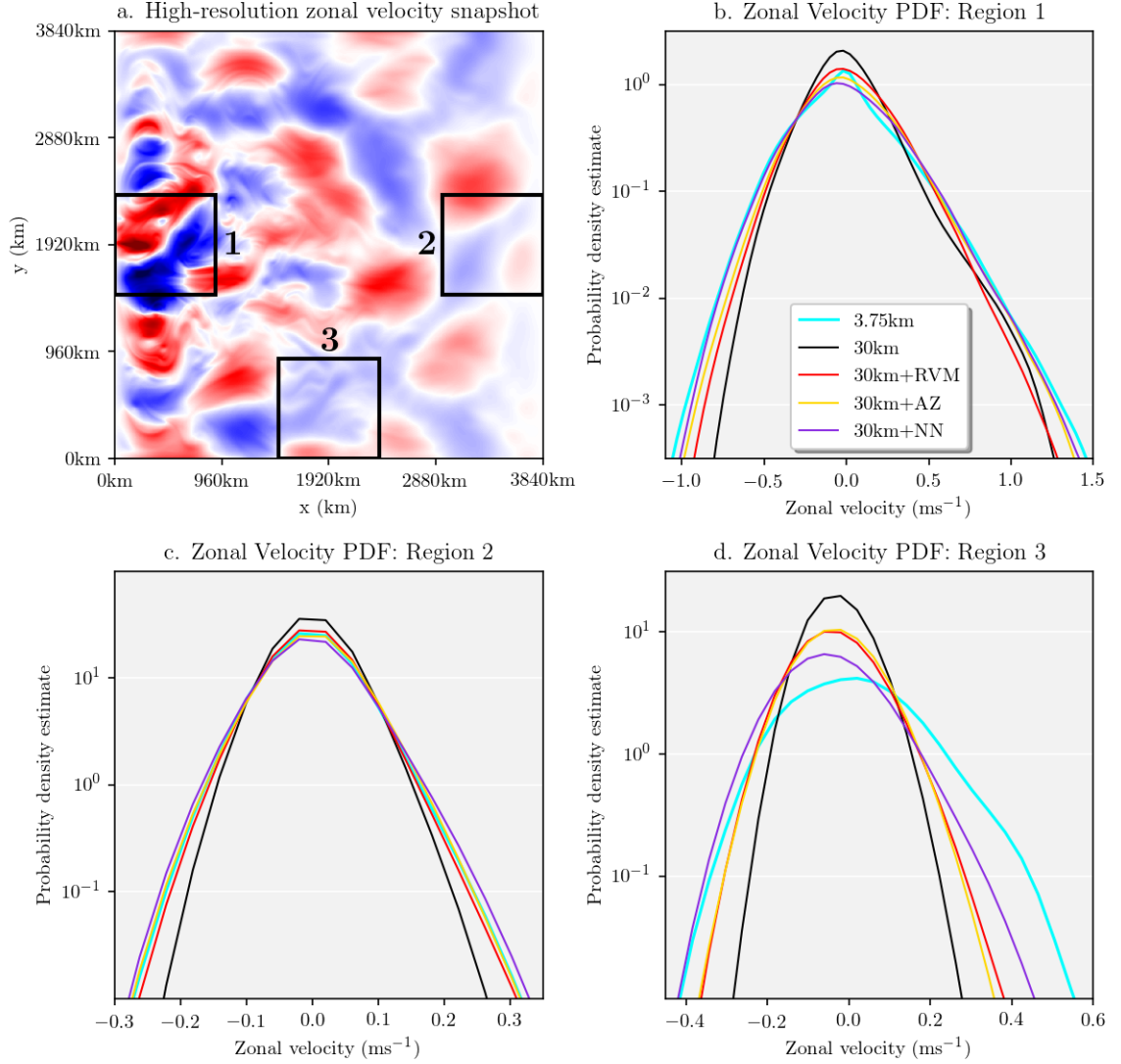


Figure 7.6: The probability distribution functions (PDFs) of the zonal velocity field, for three different regions of the model - these regions are illustrated in Panel (a). The PDFs for regions 1, 2, and 3 are shown in Panels (b), (c), and (d) respectively. Each panel includes PDFs of the high-resolution, the low-resolution, and the three parameterised models.

7.4.2 Impact of parameterisations on model energetics

Low-resolution ocean models, due to the truncation of small-scales and large viscosity coefficients, are under-energetic compared to higher-resolution models. By parameterising eddy momentum fluxes, we hope to correct some of this lack of energy. So how do the energetics of the parameterised models compare to the high-resolution model?

The spatially-averaged kinetic energy during the 10 years of spin-up is shown in Figure 7.7a. Each model takes approximately 5 years to equilibrate, in particular the 30km resolution model (dark grey) reaching an average kinetic energy of $0.024\text{m}^2\text{s}^{-2}$, and the 3.75km model (cyan) with over double with $0.056\text{m}^2\text{s}^{-2}$. Note that we do not show time series of the spatially-averaged momentum as each parameterisation conserves global momentum by construction, which was also checked manually in the model output.

All three parameterisations increase the amount of kinetic energy in the model. The RVM expression increases the kinetic energy to approximately halfway between the 30km and 3.75km models, at a value of $0.038\text{m}^2\text{s}^{-2}$. The AZ17 and FCNN parameterisations increase the kinetic energy of the model to within approximately 5% of the high-resolution model at $0.056\text{m}^2\text{s}^{-2}$.

Note that in Figure 7.7a, the AZ17 parameterisation exhibits a sharp increase in kinetic energy, surpassing that of the high-resolution model, just after the start of the fourth year of spin-up. After reviewing animations of the flow field during this spike in kinetic energy, we see patches of intense instability develop within the jet, which causes the velocities within the patches to increase dramatically. At higher attenuation levels, we found that these types of instabilities led to the model blowing up. However, in this case, the model manages to recover by redistributing and dissipating the energy within the patches. This behaviour is not limited to the AZ17 parameterisation; the FCNN and RVM parameterisations also generate the same patches of instability when the attenuation level is high enough.

The parameterisations also increase the spatially-averaged enstrophy in the 30km, shown in Figure 7.7b. Here, each parameterisation increases the enstrophy, but still underestimates the enstrophy of the high-resolution model by a significant margin.

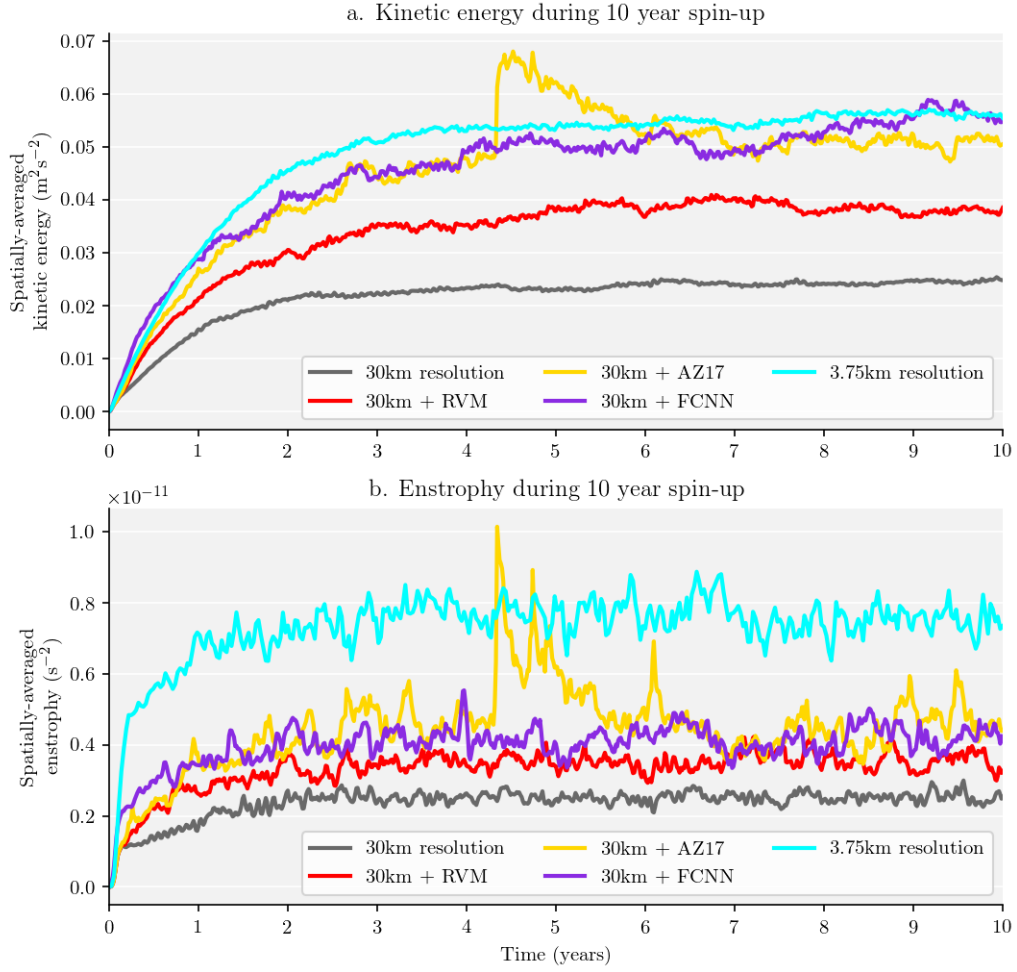


Figure 7.7: Time-series of the globally-integrated kinetic energy of four shallow water models during a 10 year spin-up. One model has no momentum parameterisation, while the other models have one of the following: deformation-based parameterisation of Anstey and Zanna (2017), the expression revealed by the relevance vector machine (RVM) algorithm, and a fully-convolutional neural network.

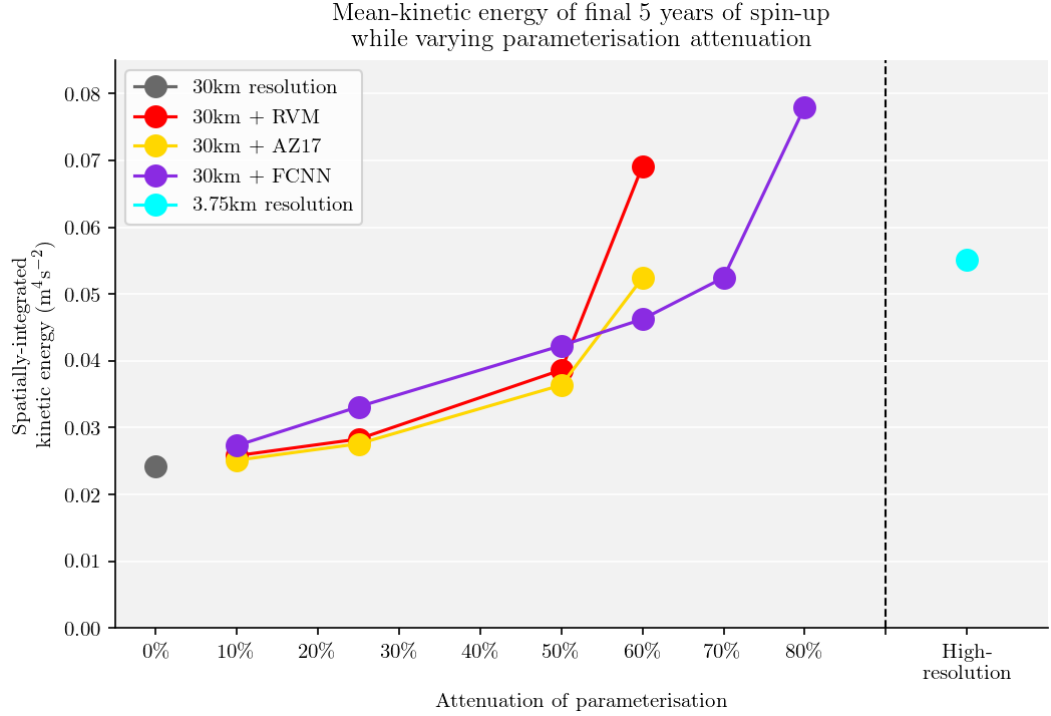


Figure 7.8: Sensitivity of each momentum parameterisation to the amount of prescribed attenuation. The global kinetic energy is average over the final 5 years of the spin-up, and plotted against the percentage by which the parameterisation is attenuated. Note that the parameterisations from the RVM algorithm and Anstey and Zanna (2017) diverge to infinity at 70% attenuation and higher, while the convolutional neural network produces un-physical flow fields at 90% attenuation and higher.

With regards to differences between the parameterisations, the RVM algorithm increases the enstrophy the least, with minimal differences between the AZ17 and FCNN parameterisations. The observed spike in kinetic energy by the AZ17 parameterisation (Figure 7.7a) also produces a large spike in the enstrophy, as can be seen during the fourth year (Figure 7.7b).

For the RVM, AZ17, and FCNN parameterisations, we have been using attenuation levels of 50%, 60%, and 70% respectively. We now show how the kinetic energy of each model varies with the attenuation level. Figure 7.8 shows the average kinetic energy of the final 5 years of the spin-up period, as a function of the level of attenuation. The 30km model with no parameterisation is included, which is equivalent to an attenuation level of 0%, i.e., all parameterisations are switched off. We also include the high-resolution model for comparison on the right-hand side of Figure 7.8.

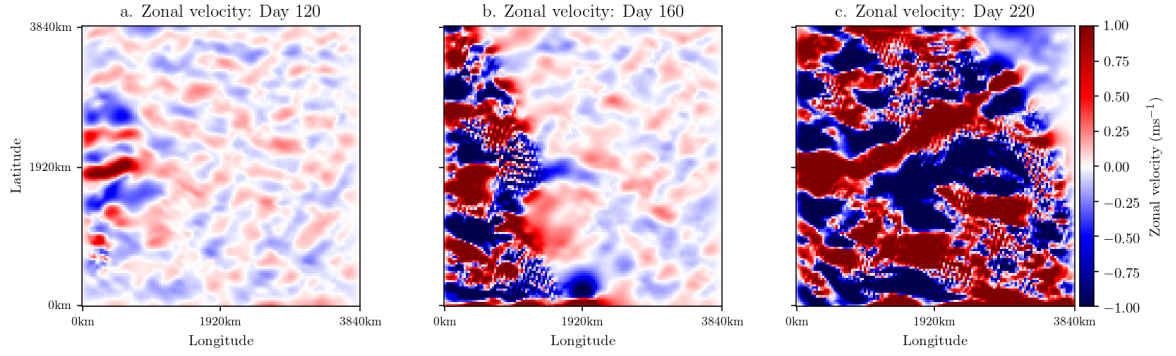


Figure 7.9: An illustration of the instabilities that develop when the fully-convolutional neural network is implemented at 100% strength. The zonal velocity is shown at days 120, 160, and 220 of the spin-up. On day 120, instabilities develop in the bottom left corner of the domain, which then spread across the entire domain.

Each parameterisation is implemented in separate models at varying degrees of attenuation, starting at 10% and increasing until either the model diverges or becomes physically un-realistic. At 10% attenuation, the FCNN, RVM and AZ17 parameterisations produce models with kinetic energy similar to the un-parameterised 30km model. The kinetic energy of the FCNN model increases almost linearly from 10% to 70% attenuation. At 80% attenuation and higher, the kinetic energy of the FCNN model increases exponentially, and at 100% the FCNN produces a un-realistic flow field with average kinetic energy $50\times$ that of the high-resolution model.

At attenuation levels higher than 50% and 60% respectively, the RVM and AZ17 parameterisations begin to dramatically increase kinetic energy within the model, with the RVM parameterisation increasing kinetic energy slightly more than the AZ17 parameterisation. At an attenuation level of 70%, both the RVM and AZ17 parameterisation cause the model to diverge to infinity.

Overall, when the energy of a parameterised 30km model approaches that of the high-resolution 3.75km model, that is where issues of numerical stability or un-physical flow fields begin to emerge. The parameterisations at this attenuation level are over-energising the flow, i.e., they are inputting too much energy into the model. This appears to be primarily a physical effect, as we do want the parameterisations to capture dynamical processes that energise the lower-resolution model; the issue is that at high attenuation levels this energising effect continues excessively and the parameterisations do not know when to stop inputting energy.

To illustrate the instabilities that over-energise the flow, the zonal velocity of the model with the FCNN implemented at 100% is shown in Figure 7.9. During the first 100 days of spin-up, the model behaves as expected, developing a zonal jet at the western boundary. At day 120 (Figure 7.9a), the very first signs of the instability - small-scale gradients that are visibly sharper - appear in the bottom-left corner of the domain. Over 10-20 weeks, the instabilities grow and spread across the entire domain (Figure 7.9b and 7.9c). The instabilities over-energise the flow, producing velocities of nearly 10ms^{-1} . Curiously, the velocities do eventually stabilise and the model does *not* blow up. The FCNN manages to input a significant amount of energy into the model, but then somehow eventually stabilises. Whereas the models with the AZ17 and RVM parameterisations diverge once the instabilities grow.

Figure 7.7a shows that the parameterisations increase the *spatially-averaged* kinetic energy of the model. But how do the parameterisations affect the PDFs of kinetic energy? Similar to the zonal velocity in Figure 7.6, we estimate the PDFs of kinetic energy of each model in Figure 7.10. We use the same three regions, illustrated in Figure 7.10a.

Each kinetic energy PDF of Figure 7.10 is approximately log-normal, i.e., the PDF comprises a large peak close to zero, which then decays as the kinetic energy increases. For all three regions, the high-resolution PDF has more probability in the tails compared to the 30km model with no parameterisation. The effect of all parameterisations is to increase the probability in the tails, with little shift in the position of the peak. Therefore, the parameterisations increase the frequency of extreme kinetic energy values, as opposed to solely increasing the mean kinetic energy.

In regions 1 and 2 (Figure 7.10b and 7.10c), the FCNN is the best performing, with the kinetic energy PDF of the FCNN parameterisation almost indistinguishable from the high-resolution model. However, in region 3, all three parameterisations cause too much probability to be redistributed in the tails, as evident by the peaks of the RVM, AZ17, and FCNN kinetic energy PDFs all being below the high-resolution peak.

The kinetic energy PDFs of Figure 7.10 suggest that the energising effects of all three parameterisations are appropriate for regions 1 and 2. But in region 3, it may be the case that the input of energy of the parameterisations is not appropriate. Indeed,

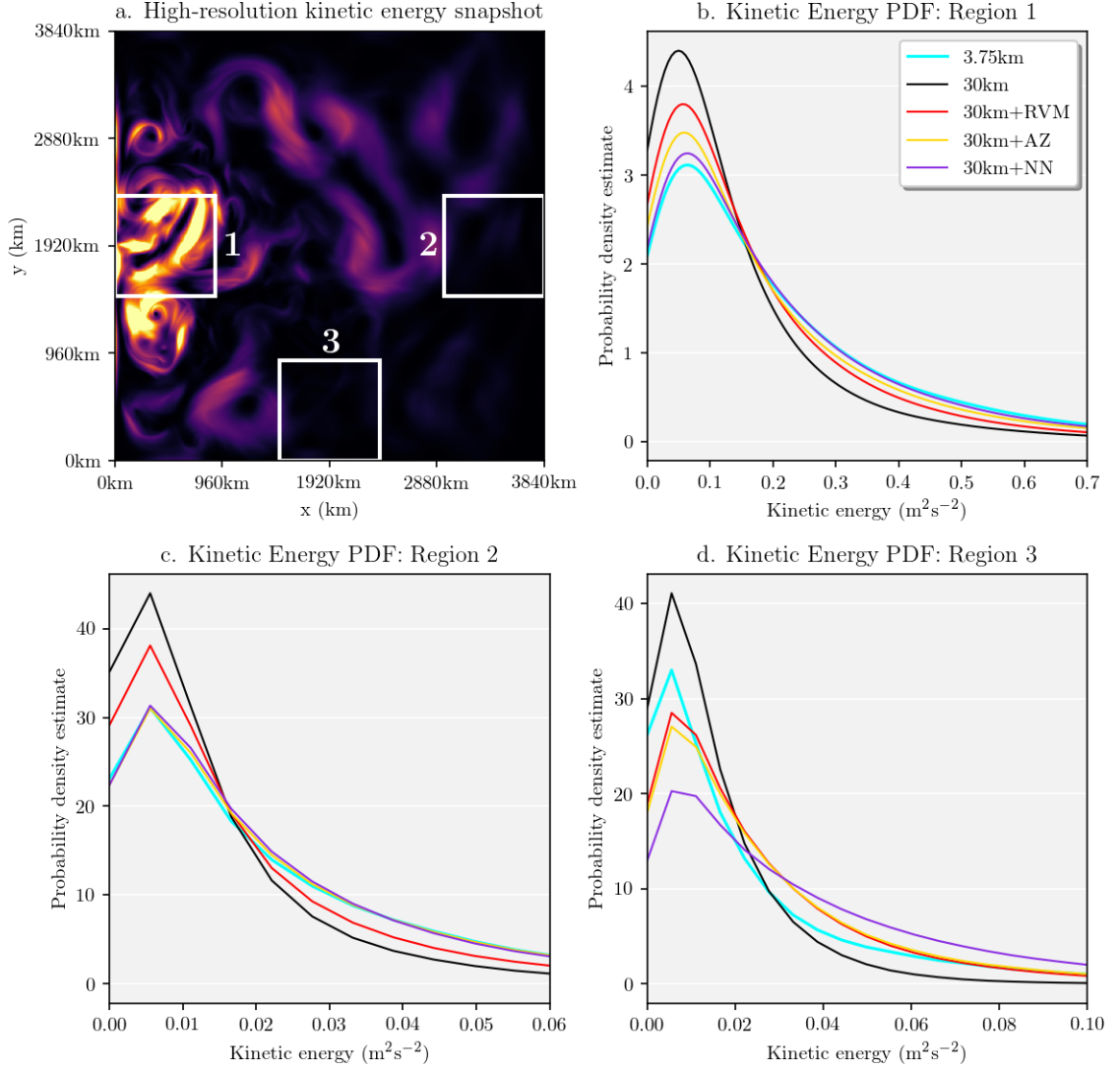


Figure 7.10: The probability distribution functions (PDFs) of the kinetic energy, for three different regions of the model - these regions are illustrated in Panel (a). The PDFs for regions 1, 2, and 3 are shown in Panels (b), (c), and (d) respectively. The kernel width parameter used in the PDF estimation scheme is the same for all regions, which is why the curves in Panel (c) and (d) are less smooth, as the kernel width (0.01) is of the same magnitude as the width of the peaks.

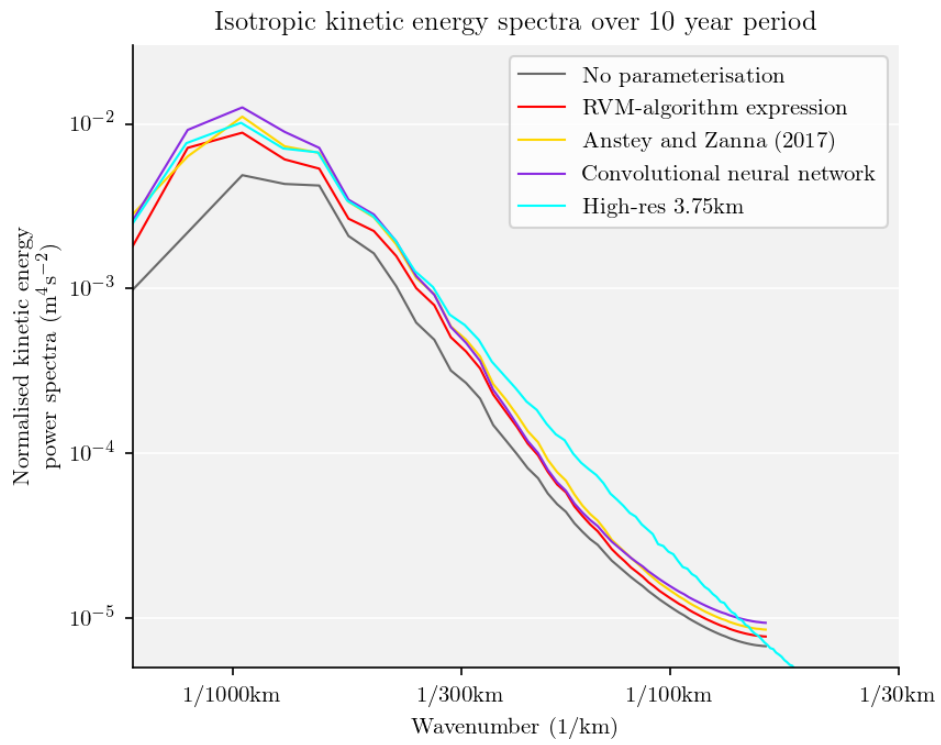


Figure 7.11: The kinetic energy spectra of the three parameterised 30km models, in addition to the 30km model with no parameterisation, and the high-resolution 3.75km model. The spectra are calculated over a 10 year period.

the variance of the eddy momentum fluxes dominates within region 1, and therefore the training of the parameterisations will be biased towards the energetics of region 1. The data-driven parameterisations may have learnt to act as a source of energy from region 1, and then during implementation assumed that the energetics of region 3 should be the same as region 1. Hence, the over-energising of region 3.

The kinetic energy power spectra of each model are shown in Figure 7.11. Across all spatial scales, the high-resolution model has more kinetic energy compared to the 30km model with no parameterisation. At spatial-scales larger than 300-400km, we see that the parameterisations increase the kinetic energy to approximately the same level as the high-resolution model. The increase in energy at larger scales implies the parameterisations are increasing the inverse cascade of energy. At large-scales, the FCNN parameterisation increases the kinetic energy to above that of the high-resolution model. At length-scales smaller than 300km, the parameterisations do increase the kinetic energy, but they all remain lower than that of the high-resolution model. We did not compare the slopes of the spectra to compare to the expected scalings for an inverse cascade, but would be an interesting avenue to explore.

7.5 Summary and discussion

7.5.1 Summary

In this chapter, we conduct a prognostic test of two data-driven and one physics-driven eddy momentum parameterisations in a shallow water model. One data-driven parameterisation is a fully-convolutional neural network with high predictive skill, while the other is an interpretable closed-form expression revealed by the RVM algorithm of Chapter 6. The physics-driven parameterisation is the deformation-based expression of Anstey and Zanna (2017).

In terms of diagnostic performance, both data-driven parameterisations accurately reproduce the spatial-patterns of the mean, standard deviation, skewness and kurtosis of the true eddy momentum forcing S_x (Figure 7.2) and S_y (Figure 7.3). The FCNN produces high correlations within the jet, whereas the RVM expression captures the variance more uniformly across the domain.

When implemented into a 30km shallow water model, all three parameterisations increase the temporal variability (second column of Figures 7.4 and 7.5), leading to a

more accurate emulation of the 3.75km high-resolution model. Although not evident in the spatial maps of skewness and kurtosis (Figure 7.4), the parameterisations do in fact increase the amount of probability in the tails of the PDFs of the zonal velocity field (Figure 7.6). The parameterisations therefore increase the frequency of extreme values of the velocity field, and do not solely increase the variability.

Each parameterisation increases the kinetic energy of the model (Figure 7.7), countering the spurious loss of energy in low-resolution ocean models. The kinetic energy of the model increases monotonically with the attenuation, or ‘strength’, of the parameterisation being implemented (Figure 7.8). When the kinetic energy of the parameterised 30km model approaches that of the high-resolution model, issues of numerical stability begin to emerge.

Note that, from Figure 7.8, we can implement the neural network parameterisation at 70% attenuation without instability issues, whereas we cannot do the same for the AZ17 and RVM parameterisations. This suggests that the complexity of a deep neural network may be more numerically stable compared to implementing a closed-form equation.

From examining the PDFs of the zonal velocity (Figure 7.6) and kinetic energy (Figure 7.10), we find that the parameterisations increase the frequency of extreme kinetic values, not just increasing the mean. The kinetic energy spectra of the parameterised models (Figure 7.11) show that the kinetic energy is increased primarily at spatial-scales larger than 300km.

One aspect of implementation we have not yet presented is the computational time taken to run each parameterised model. Figure 7.12 shows the time taken to run various instances of the shallow water model, in Python for 10 years, with the same number of CPUs. The vanilla 30km model takes approximately 5 hours to run for ten years, with the AZ17 and RVM parameterisations increasing that time by less than 2%. Implementing the FCNN parameterisation increases the run time to ~ 1 day, more than quadruple the original 30km model. This increase in time is due to the computational cost of the floating-point operations of each convolution layer. However, the FCNN model is still significantly faster to run compared to the higher resolution 7.5km and 3.75km models, which take ~ 5 and ~ 42 days respectively.

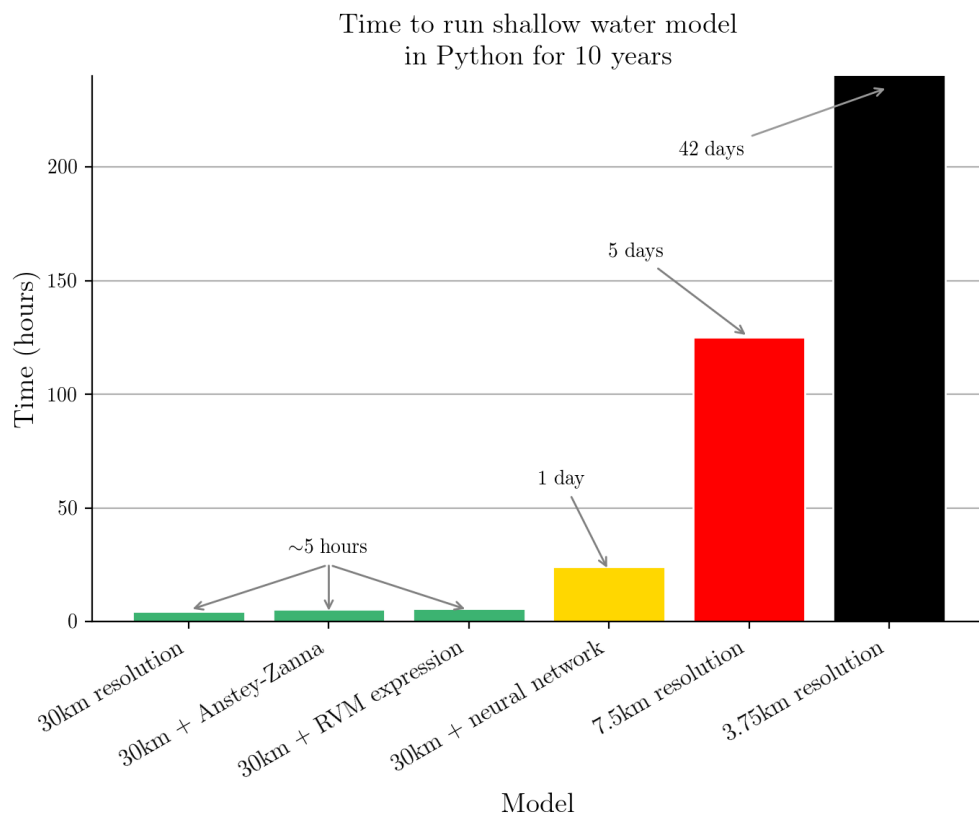


Figure 7.12: A comparison of the computational time to run the shallow water model for 10 years for various resolutions and parameterisations.

7.5.2 Discussion

We have shown that data-driven algorithms can be implemented in idealised ocean models to parameterise eddy momentum fluxes, moving a step closer to implementation in more realistic global models. The goal is not to replace physics-driven parameterisations - indeed, we should still strive to understand the theory and mechanisms of turbulent processes. In the meantime, we should use every tool at our disposal for producing more accurate projections of future climate.

We have also shown that both predictive (e.g. the FCNN) and inference-based (e.g. RVM) algorithms can be used to learn eddy parameterisations from data. Data-driven algorithms of a wide range of complexity can therefore be utilised to construct parameterisations. There will be situations where predictive skill is valued over interpretability and vice versa. For example, if the primary goal of a physical parameterisation is to reduce forecasting error within a model of the ocean or atmosphere, then predictive skill may be prioritised. On the other hand, if the goal is to also improve our physical understanding of the sub-grid or unresolved process being parameterised, then an inference-based approach of lower complexity could be preferred.

A disadvantage of using an algorithm with high predictive skill, and therefore higher model complexity, is the typically higher computational cost. Convolutional neural networks have particularly high computational costs, as we saw in Figure 7.12, due to the numerous floating point operations of each convolution layer. The computational cost of implementing a data-driven parameterisation of lower complexity, such as a sparse regression, produces lower computational overhead within the model. However, the higher complexity algorithms may be more numerically stable, as seen with the fully-convolutional neural network (Figure 7.8) - though this will depend on how the data-driven parameterisation is implemented, i.e., implicit versus an explicit numerical scheme.

An obstacle that our idealised study has not had to deal with is topography. Realistic global ocean models contain the intricate coastlines of the continents and the bathymetry of the ocean floor. This is not an issue for parameterisations constructed through the data-driven discovery of equations, as implementation means solving the existing equations of motion with the additional terms. However, the topography is

an issue for convolutional neural networks, which may not be able to discern coastlines and bathymetry from the open ocean.

One potential solution is to have what is called a locally-connected layer as the first layer of the neural network architecture. A locally-connected layer is similar to a convolution layer, except every point in space has a unique filter to be learnt, i.e., the filter of the convolution is *not* shared across each point in a locally-connected layer. Having a unique trainable filter for every point in space could allow the neural network to differentiate between land and ocean points.

Another potential solution is to somehow incorporate information about coastlines and bathymetry into the neural network, such that the architecture automatically sets the value of the parameterisation on land points to zero, and also enforces boundary conditions. This approach is exact, such that values of the parameterisation could be enforced exactly by the architecture.

Our results confirm that diagnostic performance does not directly transfer to prognostic performance within the model: the RVM and FCNN parameterisations successfully reproduce the spatial maps of skewness and kurtosis of S_x and S_y in the validation data, but does not fully reproduce the maps of skewness and kurtosis of the velocity field when implemented. Though the tails of the velocity and kinetic energy PDFs, calculated using values from particular regions, are improved by the parameterisations (Figure 7.6). It should be noted that the AZ parameterisation does outperform the RVM parameterisation, especially in the kinetic energy PDFs of region 1.

The data-driven algorithms in this chapter are deterministic, meaning that there is a one-to-one mapping from input to output. In order to further improve the higher-order moments of the velocity field in coarse-resolution models, the data-driven parameterisation may need to be stochastic, i.e., the inputs values predict the parameters of a distribution which is then sampled from. A data-driven stochastic eddy parameterisation was used in Mana and Zanna (2014) where, sampling from a distribution characterised by the first four statistical moments, improved the prognostic performance over the deterministic parameterisation. A generative adversarial network (GAN) was trained as stochastic closure in the idealised Lorenz '96 model

(Gagne et al., 2019), producing better forecasts and climate simulations compared to the baseline parameterisation.

Moving from diagnostic to prognostic tests of each parameterisation also revealed issues regarding numerical instability. If the strength of the parameterisation was too high, then it would input too much kinetic energy into the model, sometimes causing the model to diverge to infinity. This excessive source of energy is a bi-product of training a data-driven parameterisation in a diagnostic fashion: the data-driven parameterisation does not know when to stop adding energy into the model. Ideally, the parameterisation should be able to learn when to add or remove energy, depending on the amount of energy already in the model.

There are two options to achieve an eddy momentum parameterisation that correctly balances the amount of energy in the model. First, an additional prognostic equation for the eddy energy can be added to the model (e.g. Jansen and Held, 2014; Mak et al., 2016), such that energy spuriously lost can be exactly replaced by the data-driven parameterisation. This approach explicitly keeps track of energy added or removed, and is physically desirable as closed-off systems should indeed conserve energy (though the Navier-Stokes equations are not strictly closed-off, as some energy is dissipated and lost as heat). The prognostic eddy energy equation is unknown and has to be constructed from physical principles or from data, as discussed in Chapter 6.

The second approach is to train the data-driven parameterisation in a prognostic fashion, i.e., the algorithm is simultaneously implemented and trained at the same time, by comparing with a high-resolution model run concurrently. This is also known as ‘online learning’ and has already been suggested as a way to counter instability issues when implementing data-driven parameterisations (Rasp, 2019). Training an eddy momentum parameterisation in a prognostic fashion can be viewed as implicitly learning an eddy energy budget, as if the parameterisation inputs too much energy, the prognostic learning will help correct the amount of energy in the model.

In this chapter, we only implemented data-driven parameterisations of eddy momentum fluxes. We did not conduct a prognostic test of the eddy temperature forcing, nor did we solve an additional prognostic equation for the eddy energy. This was due to practical reasons, as we did not have a primitive equation model in Python that we could easily modify, and we unfortunately ran out of time to implement our pa-

parameterisations in the Fortran of MITgcm. In the future, it is hoped that all our data-driven results - eddy momentum forcing, eddy temperature forcing, and prognostic eddy energy equation - will be implemented concurrently, to fully close the equations of motion.

Finally, there is room for improvement in the algorithm which performs the data-driven discovery of equations. We initially experimented with the sequential ridge regression of Rudy et al. (2017), but found the relevance vector machine approach of Zhang and Lin (2018) to be more reliable. However, our experimentation was not exhaustive, and there are many approaches we did not test. In fact, at the time of writing, there have been more advanced algorithms published which combine convolutional neural networks with the symbolic construction of closed-form equations (Long et al., 2019). Such algorithms may prove significantly more adept at constructing parameterisations of sub-grid parameterisations, and we are only at the beginning of exploring the full potential of these tools.

Chapter 8

Summary and discussion

8.1 Summary

Mesoscale eddies play an important role in the dynamics of the ocean and atmosphere. Many aspects of the earth system are impacted by eddies: the mean and variability of the ocean circulation, the transport and mixing of climatic tracers, and the intense air-sea fluxes of western boundary currents (Section 1.1). Coupled models must capture these effects to reduce uncertainties within climate projections, but we cannot rely on brute computational force to fully resolve mesoscale eddies at all latitudes. We must, therefore, parameterise the effects of mesoscale eddies.

The overall goal of this thesis is to improve our understanding of unresolved eddies and to then construct better representations of those eddies in ocean models. To do so, we use a variety of data-driven methods within a hierarchy of numerical ocean models. These techniques and models were used to address the following questions:

- How does the spatio-temporal variability of mesoscale eddies impact tracer fluxes?
- Can we construct accurate representations of eddy momentum fluxes, that better capture the underlying dynamics, using modern machine learning techniques?
- Can we harness the power of data-driven algorithms to construct parameterisations of eddy momentum and temperature fluxes, while retaining interpretability?

- What are the physical and numerical consequences of implementing a data-driven eddy parameterisation into an ocean model?

Each results chapter, in turn, addressed the above questions. In Chapter 4, we found that eddy tracer fluxes exhibit significant regional and temporal variability within the North Atlantic, and within the Gulf Stream the across-jet mixing could be characterised as a function of jet strength. However, in the vicinity of the North Atlantic and Azores currents, the mixing was neither a simple function of mean-flow nor eddy kinetic energy, implying that a more complex parameterisation was needed.

In Chapter 5, we tested new ways of constructing parameterisations, to better capture the complex effects of unresolved eddies. In particular, we trained deep neural networks to predict unresolved eddy momentum fluxes using only resolved variables. We found that the different dynamical processes within the idealised ocean model had a significant impact on the training of each convolutional neural network. The neural network trained on the western boundary generalised well to models of different Reynolds numbers and was robust to under-sampling in the training data. The final part of Chapter 5 explored how to make neural networks respect physical principles, namely, global conservation of momentum.

We again used data-driven algorithms to parameterise eddy fluxes in Chapter 6, similar to Chapter 5, but now focusing on retaining the interpretability of the resulting eddy parameterisation. We employed a technique known as the data-driven discovery of equations (e.g Rudy et al., 2017), which uses an algorithm to reconstruct the underlying equation of some spatio-temporal dataset. The most robust algorithm was that of Zhang and Lin (2018), based on the iterative application of the relevance vector machine (RVM), a sparse Bayesian regression.

We found that the RVM algorithm could reveal expressions for the eddy momentum and temperature forcings that captured over half the variance in the validation data, with equations comprising only a handful of basis functions. Importantly, the final equation could also be constrained to respect physical principles, by controlling what basis functions could comprise the function library.

The eddy momentum expressions revealed by the RVM algorithm, in both the barotropic and baroclinic models, contained the deformation-based parameterisation of Anstey and Zanna (2017) and were also related to the non-linear gradient model

(e.g. Nadiga, 2008). The eddy temperature forcing expression depended on the vertical derivatives of the horizontal velocity field which, assuming thermal wind balance, could then be interpreted as horizontal velocity gradients acting to deform the smaller-scale temperature field. In the residual-mean formulation, the eddy temperature forcing took the form of an additional vertical diffusion in the momentum equations.

In addition to uncovering expressions for the eddy fluxes, we used the RVM algorithm to try and find a prognostic eddy energy equation. However, only advection of eddy energy was selected, accounting for half of the variance in the validation data. Several improvements could be made to try and tackle this issue: use higher-order products and derivatives in the function library, using longer time-scales to define the time derivative of the eddy kinetic energy, and improving the eddy energy definition.

The long-term goal of an eddy parameterisation is to eventually implement them in a global model. It is therefore important to evaluate any new eddy parameterisation in a prognostic manner, to assess the impact on the physics and stability of the model. In Chapter 7, we determined how data-driven parameterisations, such as neural networks and the RVM algorithm, impact a shallow-water ocean model when implemented. This prognostic evaluation included a comparison with the existing physics-driven parameterisation of Anstey and Zanna (2017). In total, we implemented three eddy momentum flux parameterisations: the parameterisation of Anstey and Zanna (2017), an expression revealed by the RVM algorithm, and a new convolutional neural network with a physics-informed architecture, trained on the same data as the RVM algorithm, which came from a high-resolution primitive equation model in MITgcm.

All three parameterisations exhibited issues with either numerical stability or over-energising the flow, such that we had to reduce the amplitude of each parameterisation to 50-70% of the original amplitude, to ensure the models did not diverge to infinity or produce un-physical flow fields. All parameterisations conserved momentum and vorticity, but energised the flow, increasing the kinetic energy within the model, leading to a large increase in the variability of the velocity fields. By examining the probability distribution functions of the velocity field and kinetic energy, we could see that the parameterisations also made extreme events more frequent, making an improvement on the low-resolution model with no parameterisation. The fully-convolutional neural

network performed the best with regards to improving the higher-order statistics of the velocity field and kinetic energy. The Anstey and Zanna (2017) parameterisation was slightly better than the relevance vector machine parameterisation across most metrics with regards to improving the velocity field and kinetic energy of the model.

Physically, the parameterisations appeared to be capturing the inverse cascade of energy predicted by theory and present in observations (Salmon, 1978; Scott and Wang, 2005; Scott and Arbic, 2007). The deep learning-based parameterisation made the largest improvements in the energetics, implying the complexity of the algorithm allowed for a better representation of the nonlinear interactions of eddies that produce the inverse cascade.

There are many avenues for where this work could be extended. These avenues will need exploring if data-driven parameterisations are to be implemented in more realistic models in the future. They are the following:

- Training and implementing data-driven parameterisations in models which have a time-varying (or at least more realistic) wind forcing profiles.
- Constructing data-driven parameterisations that utilise memory effects. Every data-driven parameterisation in this thesis only used variables at the current time step. It is unclear how incorporating memory effects would impact a data-driven parameterisation; it could cause overfitting, but it could also significantly improve the predictive skill of the data-driven algorithm.
- Oceans are not a simple rectangular domain; they have coastlines and bathymetry. A data-driven algorithm must be able to deal with these real world complex features in global models.
- Experimenting to see which numerical schemes work the best for implementing a data-driven parameterisation. In this thesis the eddy momentum forcing was numerically integrated forward in time explicitly, but at no point did we test if an implicit scheme would be better. This could potentially solve many of the instability issues faced when implementing such parameterisations.
- Adding some sort of simple stochastic element to each of the data-driven parameterisations used in this thesis.

- Applying these data-driven algorithms to systems other than quasi-geostrophic models, such as high Rossby number shear flows or convective systems.

8.2 Discussion

What are the implications of our results for the long-term improvement of ocean models? Climate models must accurately capture the turbulent physics of the ocean, with eddy parameterisations being a necessity due to the computational cost of running models at sufficiently high-resolution. If eddies are not fully resolved and their effects not parameterised, they will contribute to uncertainty in projections of future climate.

Data-driven algorithms have become a popular tool for constructing parameterisations, which attempt to learn unresolved processes directly from data. This approach contrasts to the traditional physics-driven approach of previous decades, which constructs parameterisations via consideration of physical theories or mechanisms. Physics-driven approaches can tend to over-simplify the underlying process being parameterised. Therefore, it is hoped that the power of modern data-driven algorithms can build upon the progress of physics-driven parameterisations, and better capture the intricacies of unresolved geophysical processes.

Data-driven algorithms have recently been used to parameterise processes in the atmosphere (e.g. Krasnopolsky et al., 2005; Gentine et al., 2018; O’Gorman and Dwyer, 2018; Brenowitz and Bretherton, 2018), and construct closures for turbulence models (e.g. Ling et al., 2016b; Maulik and San, 2017; San and Maulik, 2018). Regarding the parameterisation of mesoscale eddies specifically, previous studies have contained data-driven elements (Berloff, 2005a; Mana and Zanna, 2014; Cooper and Zanna, 2015; Bachman et al., 2015). However, the results of this thesis are the first systematic attempt at using machine learning algorithms to parameterise mesoscale eddies in the ocean. This includes the use of deep neural networks and the data-driven discovery of equations, to improve the representation of unresolved eddy fluxes.

As mentioned in Chapter 2, the distinction between physics-driven and data-driven parameterisations is not always clear. There are plans to use data-driven algorithms to optimise the parameters of existing physics-driven parameterisations (Schneider et al., 2017a); the neural network turbulence closure of Ling et al. (2016b)

incorporated an invariance principle into their architecture; parameterisations of cloud processes are now being constructed that automatically conserve energy (Beucler et al., 2019b). These can be viewed as physics-informed data-driven parameterisations, i.e., hybrid approaches that utilise the power of data-driven methods while respecting physical principles.

The results of this thesis confirm that data-driven algorithms can be used while respecting physical principles. For example, constraining the library of basis functions of the RVM algorithm in Chapter 6, or the convolutional neural network architecture that automatically predicts eddy stress tensor elements in Chapter 7, leading to global conservation of momentum and vorticity. So even though it is true that the inner-workings of some data-driven algorithms are difficult to interpret, that does not automatically mean the algorithm does not respect physical principles.

Interpretability is an issue of black-box algorithms, such as the convolution neural networks in Chapter 5. Due to the complexity of the algorithm, it is difficult to ascertain exactly what physical process has been learnt by the thousands of parameters of the network. Hence, the use of the data-driven discovery of equations in Chapter 6, to provide a data-driven parameterisation which is more easily interpreted. We are not yet aware of any other studies using this approach to uncover expressions for oceanic or atmospheric parameterisations. The results of Chapter 6 therefore provide a novel approach to parameterisation that uses machine learning in an interpretable manner.

Data-driven equation discovery could not only improve the representation of unresolved eddies in models but could open up more possibilities for data-driven knowledge discovery in general. Predictive skill of machine learning algorithms is powerful, but we could see machine learning being used more frequently to better understand turbulent processes. Indeed, in Chapter 6 we only tested two equation-discovery approaches, that of Rudy et al. (2017) and Zhang and Lin (2018); our experimentation was not exhaustive so there is likely a lot of scope for improvement on our methods. At the time of writing, there are already new data-driven techniques based on deep learning that could be used for inference of unresolved physics (e.g. Lguensat et al., 2019; Raissi et al., 2019; Long et al., 2019).

Regarding eddy parameterisations, whether we use an inference-based algorithm for knowledge discovery or a black-box neural network, the end goal is the same:

implementation in a coupled global climate model. All new parameterisations must, therefore, be tested in a prognostic manner, which was the focus of Chapter 7. We found that data-driven parameterisations can improve the statistics of a lower resolution model, making it behave more like the high-resolution reference model. However, all parameterisations caused issues with stability, despite having promising diagnostic performance. This mismatch between the ‘offline’ and ‘online’ performance of data-driven algorithms has been highlighted with regards to atmospheric parameterisations by Rasp (2019). Diagnostic training on observations or high-resolution models may capture a high proportion of the variance in the validation data, but that performance may not translate when the parameterisation is implemented. The study of Rasp (2019) proposed ‘coupled online learning’, such that further training is carried out by running the high-resolution model in parallel. Our results of Chapter 7 support online learning of some form, to allow the data-driven parameterisation to react to the behaviour of the model and vice versa. For example, if the parameterisation is inputting too much kinetic energy into a coarse-resolution model, online learning would allow the algorithm to compare with the high-resolution model and correct the energy input. Improvements to the stability of the model during implementation could be made by investigating which numerical schemes synergies best with the data-driven parameterisation, e.g., a semi-implicit scheme or different time-steps for the advection and forcing terms.

In this regard, online learning of a data-driven eddy parameterisation could amount to learning an intrinsic eddy energy budget, as the total energy of the system is then constrained during training to not deviate from the high-resolution reference model. However, it could be beneficial to include a separate budget for the eddy energy to begin with. Indeed, that is what recent approaches (e.g. Jansen and Held, 2014; Bachman, 2019) do, by re-introducing spuriously lost energy back into the model. Our data-driven parameterisations appeared to capture the inverse cascade of energy, but the amount of energy that each parameterisation could input into the model was not regulated.

A separate prognostic equation for the eddy energy would be desirable, as the total energy of a closed system should be conserved, but what form should it take? Previous studies make assumptions regarding the form of the energy equation (e.g. Eden and

Greatbatch, 2008; Mak et al., 2018). We attempted to limit the assumptions made and use the RVM algorithm to find a prognostic eddy equation in Chapter 6. We found that local advection dominated, and therefore should be included in an eddy energy budget. We could not, however, robustly find additional terms that comprised the eddy energy equation - this could have been due to our definition of eddy energy, or the RVM algorithm requiring improvement. Further work is needed to use data-driven discovery methods to find a prognostic eddy energy equation, which may need higher-order products and derivatives in the function library, or higher-quality data. But if it succeeds, it could provide a novel way of constructing energy budgets while making minimal assumptions.

Whether trying to construct an eddy energy equation or reveal an expression for the eddy momentum forcing, data-driven approaches suffer from an ongoing difficulty: how to define an ‘eddy’ in the first place. To quantify the effects of eddies in the data, one must decide what constitutes an eddy, with these assumptions impacting all subsequent results. For example, coarse-graining is a spatial Reynolds average which, produces a more ‘local’ definition of eddy compared to Gaussian filter, as discussed in Section 3.2.

With regards to eddy parameterisation, imagine training the same algorithm on the same dataset, but with two different definitions of eddy; what physics have these two data-driven parameterisations captured? Which one do you choose to implement in a climate model? In practice, there is no perfect definition - every definition will have its advantages and disadvantages. There will be an infinite number of definitions, as there is a continuum of spatial (or temporal) scales to select for within the data. There is also the choice of Eulerian versus Lagrangian averaging. We only discussed and used Eulerian definitions in this thesis, but it may be that Eulerian definitions of an eddy may not be optimal for describing mesoscale eddies in the ocean (Abernathey and Haller, 2018). Large-scale currents, such as the Gulf Stream and Kuroshio Extension, exhibit Lagrangian coherent structures, so it may be the case that a Lagrangian data-driven parameterisation is needed in these regions.

In sum, the freedom of defining an ‘eddy’ can lead to an infinite number of data-driven parameterisations, given the same algorithm and dataset. It is therefore vital that future work involving data-driven parameterisations are aware of the assump-

tions being made by their definition of an eddy. Data-driven parameterisations could benefit if an optimal definition of eddy was found, or alternatively, simultaneously combining parameterisations from multiple eddy definitions.

Another challenge that data-driven parameterisations face is that of generalisation, i.e., the ability to make accurate predictions outside of the training data. Generalisation is a pertinent issue for climate change projections, where significant changes may occur in the dynamical regimes of the ocean and atmosphere. Ideally, we would validate any data-driven parameterisation with observations, which is not possible for future climate change scenarios. We showed in Section 5.3.2 that deep learning may have the ability to generalise to more turbulent regimes, but this is an idealised system - more comprehensive tests of generalisation in global models are needed. O’Gorman and Dwyer (2018) found that a random forest, used to parameterise moist convection in a GCM, could not generalise to a warmer climate if trained solely on the control climate. Interestingly, the random forest could generalise to the control climate, when trained solely on the warmer climate. These results show that, despite the proficiency of some machine learning algorithms, the ability to generalise is not guaranteed. The generalisation skill of any individual data-driven parameterisation will vary significantly, and must therefore be scrutinised.

Regardless of the aforementioned difficulties that lie ahead, data-driven approaches could produce a whole new generation of parameterisations in climate models that were not previously possible. The progress of data-driven parameterisations has so far been rapid. In particular, in making algorithms respect physical principles - a common early concern within the physics community. It seems that progress in tackling instabilities with online learning will be swift as well, moving us closer to implementation in more realistic models. There is also the aspect of inter-disciplinary work with the computer science community, which is still in its early stages; collaborations between machine learning specialists and physicists are becoming more frequent, which could further improve how data-driven algorithms are used within climate models.

There is currently considerable hype surrounding machine learning, and ‘data-driven’ approaches in general, which may lead to unrealistic expectations to what these tools can achieve. Nonetheless, we should use every tool at our disposal for tackling the issue of climate change, and time will eventually help us determine the

not-so-useful applications from the useful. The few applications which do withstand the test of time could make a real impact on climate science.

References

- Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow: A system for large-scale machine learning. In *OSDI*, volume 16, pages 265–283.
- Abernathey, R. and Haller, G. (2018). Transport by lagrangian vortices in the eastern pacific. *Journal of Physical Oceanography*, 48(3):667–685.
- Abernathey, R. P. and Marshall, J. (2013). Global surface eddy diffusivities derived from satellite altimetry. *Journal of Geophysical Research: Oceans*, 118(2):901–916.
- Adcroft, A., Campin, J., Dutkiewicz, S., Evangelinos, C., Ferreira, D., Forget, G., Fox-Kemper, B., Heimbach, P., Hill, C., Hill, E., et al. (2008). Mitgcm user manual.
- Andres, M. (2016). On the recent destabilization of the Gulf Stream path downstream of Cape Hatteras. *Geophysical Research Letters*, 43(18):9836–9842.
- Anstey, J. A. and Zanna, L. (2017). A deformation-based parametrization of ocean mesoscale eddy reynolds stresses. *Ocean Modelling*, 112:99–111.
- Arakawa, A. and Lamb, V. R. (1977). Computational design of the basic dynamical processes of the ucla general circulation model. *General circulation models of the atmosphere*, 17(Supplement C):173–265.
- Arbic, B. K., Polzin, K. L., Scott, R. B., Richman, J. G., and Shriver, J. F. (2013). On eddy viscosity, energy cascades, and the horizontal resolution of gridded satellite altimeter products. *Journal of Physical Oceanography*, 43(2):283–300.
- Bachman, S. D. (2019). The gm+ e closure: A framework for coupling backscatter with the gent and mcwilliams parameterization. *Ocean Modelling*, 136:85–106.
- Bachman, S. D., Fox-Kemper, B., and Bryan, F. O. (2015). A tracer-based inversion method for diagnosing eddy-induced diffusivity and advection. *Ocean Modelling*, 86:1–14.

- Berloff, P. S. (2005a). On dynamically consistent eddy fluxes. *Dynamics of atmospheres and oceans*, 38(3-4):123–146.
- Berloff, P. S. (2005b). Random-forcing model of the mesoscale oceanic eddies. *Journal of Fluid Mechanics*, 529:71–95.
- Berloff, P. S., McWilliams, J. C., and Bracco, A. (2002). Material transport in oceanic gyres. part i: Phenomenology. *Journal of Physical Oceanography*, 32(3):764–796.
- Beucler, T., Pritchard, M., Rasp, S., Gentine, P., Ott, J., and Baldi, P. (2019a). Enforcing analytic constraints in neural-networks emulating physical systems. *arXiv preprint arXiv:1909.00912*.
- Beucler, T., Rasp, S., Pritchard, M., and Gentine, P. (2019b). Achieving conservation of energy in neural network emulators for climate modeling. *arXiv preprint arXiv:1906.06622*.
- Bishop, C. M. (2006). *Pattern recognition and machine learning*. springer.
- Boussinesq, J. (1872). Théorie des ondes et des remous qui se propagent le long d’un canal rectangulaire horizontal, en communiquant au liquide contenu dans ce canal des vitesses sensiblement pareilles de la surface au fond. *Journal de mathématiques pures et appliquées*, pages 55–108.
- Boussinesq, J. (1903). *Thōrie analytique de la chaleur mise en harmonie avec la thermodynamique et avec la thōrie mcanique de la lumi-re: Refroidissement et chauffage par rayonnement, conductibilit des tiges, lames et masses cristallines, courants de convection, thōrie mcanique de la lumi-re. 1903. xxxii, 625,[1] p*, volume 2. Gauthier-Villars.
- Bower, A. and Rossby, T. (1989). Evidence of cross-frontal exchange processes in the gulf stream based on isopycnal rafos float data. *Journal of Physical Oceanography*, 19(9):1177–1190.
- Bower, A. S. (1991). A simple kinematic mechanism for mixing fluid parcels across a meandering jet. *Journal of Physical Oceanography*, 21(1):173–180.

- Bower, A. S., Rossby, H. T., and Lillibridge, J. L. (1985). The gulf stream—barrier or blender? *Journal of Physical Oceanography*, 15(1):24–32.
- Brenowitz, N. D. and Bretherton, C. S. (2018). Prognostic validation of a neural network unified physics parameterization. *Geophysical Research Letters*.
- Brenowitz, N. D. and Bretherton, C. S. (2019). Spatially extended tests of a neural network parametrization trained by coarse-graining. *arXiv preprint arXiv:1904.03327*.
- Busecke, J., Abernathey, R. P., and Gordon, A. L. (2017). Lateral eddy mixing in the subtropical salinity maxima of the global ocean. *Journal of Physical Oceanography*, 47(4):737–754.
- Busecke, J. J. and Abernathey, R. P. (2019). Ocean mesoscale mixing linked to climate variability. *Science Advances*, 5(1):eaav5014.
- Cessi, P. (2008). An energy-constrained parameterization of eddy buoyancy flux. *Journal of Physical Oceanography*, 38(8):1807–1819.
- Champion, K., Lusch, B., Kutz, J. N., and Brunton, S. L. (2019). Data-driven discovery of coordinates and governing equations. *arXiv preprint arXiv:1904.02107*.
- Chapman, C. and Charantonis, A. A. (2017). Reconstruction of subsurface velocities from satellite observations using iterative self-organizing maps. *IEEE Geoscience and Remote Sensing Letters*, 14(5):617–620.
- Chapman, C. and Sallée, J.-B. (2017). Can we reconstruct mean and eddy fluxes from argo floats? *Ocean Modelling*, 120:83–100.
- Charney, J. G. (1971). Geostrophic turbulence. *Journal of the Atmospheric Sciences*, 28(6):1087–1095.
- Chen, R., Gille, S. T., and McClean, J. L. (2017). Isopycnal eddy mixing across the kuroshio extension: Stable versus unstable states in an eddying model. *Journal of Geophysical Research: Oceans*, 122(5):4329–4345.

- Chen, R., McClean, J. L., Gille, S. T., and Griesel, A. (2014). Isopycnal eddy diffusivities and critical layers in the kuroshio extension from an eddying ocean model. *Journal of Physical Oceanography*, 44(8):2191–2211.
- Chollet, F. et al. (2015). Keras. <https://keras.io>.
- Cooper, F. C. (2017). Optimisation of an idealised primitive equation ocean model using stochastic parameterization. *Ocean Modelling*, 113:187–200.
- Cooper, F. C. and Zanna, L. (2015). Optimisation of an idealised ocean model, stochastic parameterisation of sub-grid eddies. *Ocean Modelling*, 88:38–53.
- Courant, R., Friedrichs, K., and Lewy, H. (1928). Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische annalen*, 100(1):32–74.
- Cruz, M. A., Thompson, R. L., Sampaio, L. E., and Bacchi, R. D. (2019). The use of the reynolds force vector in a physics informed machine learning approach for predictive turbulence modeling. *Computers & Fluids*, page 104258.
- Danabasoglu, G. and Marshall, J. (2007). Effects of vertical variations of thickness diffusivity in an ocean general circulation model. *Ocean Modelling*, 18(2):122–141.
- David, T. W., Marshall, D. P., and Zanna, L. (2017). The statistical nature of turbulent barotropic ocean jets. *Ocean Modelling*, 113:34–49.
- Davis, R. E., Ohman, M. D., Rudnick, D. L., and Sherman, J. T. (2008). Glider surveillance of physics and biology in the southern california current system. *Limnology and Oceanography*, 53(5part2):2151–2168.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.
- Dillon, T. M. (1982). Vertical overturns: A comparison of thorpe and ozmidov length scales. *Journal of Geophysical Research: Oceans*, 87(C12):9601–9613.
- Dong, C., Loy, C. C., He, K., and Tang, X. (2016). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2):295–307.

- Ducet, N. and Le Traon, P.-Y. (2001). A comparison of surface eddy kinetic energy and reynolds stresses in the gulf stream and the kuroshio current systems from merged topex/poseidon and ers-1/2 altimetric data. *Journal of Geophysical Research: Oceans*, 106(C8):16603–16622.
- Eden, C. and Greatbatch, R. J. (2008). Towards a mesoscale eddy closure. *Ocean Modelling*, 20(3):223–239.
- Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., and Taylor, K. E. (2016). Overview of the coupled model intercomparison project phase 6 (cmip6) experimental design and organization. *Geoscientific Model Development (Online)*, 9(LLNL-JRNL-736881).
- Farneti, R., Delworth, T. L., Rosati, A. J., Griffies, S. M., and Zeng, F. (2010). The role of mesoscale eddies in the rectification of the southern ocean response to climate change. *Journal of Physical Oceanography*, 40(7):1539–1557.
- Ferrari, R. and Nikurashin, M. (2010). Suppression of eddy diffusivity across jets in the southern ocean. *Journal of Physical Oceanography*, 40(7):1501–1519.
- Ferrari, R. and Plumb, R. (2003). Residual circulation in the ocean. In *Near-Boundary Processes and Their Parameterization: Proc. 'Aha Huliko'a Hawaiian Winter Workshop*, pages 219–228. Citeseer.
- Ferrari, R. and Wunsch, C. (2009). Ocean circulation kinetic energy: Reservoirs, sources, and sinks. *Annual Review of Fluid Mechanics*, 41.
- Fick, A. (1855). Ueber diffusion. *Annalen der Physik*, 170(1):59–86.
- Fjørtoft, R. (1953). On the changes in the spectral distribution of kinetic energy for twodimensional, nondivergent flow. *Tellus*, 5(3):225–230.
- Foukal, N. P. and Lozier, M. S. (2016). No inter-gyre pathway for sea-surface temperature anomalies in the north atlantic. *Nature communications*, 7:11333.
- Friedman, R. M. (1993). *Appropriating the weather: Vilhelm Bjerknes and the construction of a modern meteorology*. Cornell University Press.

- Gagne, I., John, D., Christensen, H. M., Subramanian, A. C., and Monahan, A. H. (2019). Machine learning for stochastic parameterization: Generative adversarial networks in the lorenz’96 model. *arXiv preprint arXiv:1909.04711*.
- Gent, P. R. and McWilliams, J. C. (1990). Isopycnal mixing in ocean circulation models. *Journal of Physical Oceanography*, 20(1):150–155.
- Gentine, P., Pritchard, M., Rasp, S., Reinaudi, G., and Yacalis, G. (2018). Could machine learning break the convection parameterization deadlock? *Geophysical Research Letters*.
- Giglio, D., Lyubchich, V., and Mazloff, M. (2018). Estimating oxygen in the southern ocean using argo temperature and salinity. *Journal of Geophysical Research: Oceans*.
- Gnanadesikan, A., Pradal, M.-A., and Abernathey, R. (2015). Isopycnal mixing by mesoscale eddies significantly impacts oceanic anthropogenic carbon uptake. *Geophysical Research Letters*, 42(11):4249–4255.
- Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. <http://www.deeplearningbook.org>.
- Greatbatch, R., Zhai, X., Claus, M., Czeschel, L., and Rath, W. (2010a). Transport driven by eddy momentum fluxes in the gulf stream extension region. *Geophysical Research Letters*, 37(24).
- Greatbatch, R. J. and Lamb, K. G. (1990). On parameterizing vertical mixing of momentum in non-eddy resolving ocean models. *Journal of Physical Oceanography*, 20(10):1634–1637.
- Greatbatch, R. J., Zhai, X., Kohlmann, J.-D., and Czeschel, L. (2010b). Ocean eddy momentum fluxes at the latitudes of the gulf stream and the kuroshio extensions as revealed by satellite data. *Ocean Dynamics*, 60(3):617–628.
- Griesel, A., McClean, J., Gille, S., Sprintall, J., and Eden, C. (2014). Eulerian and lagrangian isopycnal eddy diffusivities in the southern ocean of an eddying model. *Journal of Physical Oceanography*, 44(2):644–661.

- Griffies, S. M., Winton, M., Anderson, W. G., Benson, R., Delworth, T. L., Dufour, C. O., Dunne, J. P., Goddard, P., Morrison, A. K., Rosati, A., et al. (2015). Impacts on ocean heat from transient mesoscale eddies in a hierarchy of climate models. *Journal of Climate*, 28(3):952–977.
- Guinehut, S., Dhomps, A., Larnicol, G., and Le Traon, P.-Y. (2012). High resolution 3-d temperature and salinity fields derived from in situ and satellite observations. *Ocean Science*, 8(5):845–857.
- Haarsma, R. J., García-Serrano, J., Prodhomme, C., Bellprat, O., Davini, P., and Drijfhout, S. (2019). Sensitivity of winter north atlantic-european climate to resolved atmosphere and ocean dynamics. *Scientific reports*, 9(1):1–8.
- Hallberg, R. (2013). Using a resolution function to regulate parameterizations of oceanic mesoscale eddy effects. *Ocean Modelling*, 72:92–103.
- He, K. and Sun, J. (2015). Convolutional neural networks at constrained time cost. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5353–5360.
- Hewitt, H. T., Roberts, M. J., Hyder, P., Graham, T., Rae, J., Belcher, S. E., Bourdallé-Badie, R., Copsey, D., Coward, A., Guiavarch, C., et al. (2016). The impact of resolving the rossby radius at mid-latitudes in the ocean: Results from a high-resolution version of the met office gc2 coupled model. *Geoscientific Model Development*, 9(10):3655–3670.
- Hogg, N. G. (1992). On the transport of the gulf stream between cape hatteras and the grand banks. *Deep Sea Research Part A. Oceanographic Research Papers*, 39(7-8):1231–1246.
- Holland, W. R. (1978). The role of mesoscale eddies in the general circulation of the ocean—numerical experiments using a wind-driven quasi-geostrophic model. *Journal of Physical Oceanography*, 8(3):363–392.
- Holton, J. R. (1973). An introduction to dynamic meteorology. *American Journal of Physics*, 41(5):752–754.

- Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.
- Hughes, C. W., Thompson, A. F., and Wilson, C. (2010). Identification of jets and mixing barriers from sea level and vorticity measurements using simple statistics. *Ocean Modelling*, 32(1):44–57.
- Jansen, M. F. and Held, I. M. (2014). Parameterizing subgrid-scale eddy effects using energetically consistent backscatter. *Ocean Modelling*, 80:36–48.
- Jansen, M. F., Held, I. M., Adcroft, A., and Hallberg, R. (2015). Energy budget-based backscatter in an eddy permitting primitive equation model. *Ocean Modelling*, 94:15–26.
- Jayne, S. R. and Marotzke, J. (2002). The oceanic eddy heat transport. *Journal of Physical Oceanography*, 32(12):3328–3345.
- Jochum, M., Danabasoglu, G., Holland, M., Kwon, Y.-O., and Large, W. (2008). Ocean viscosity and climate. *Journal of Geophysical Research: Oceans*, 113(C6).
- Keating, S. R., Majda, A. J., and Smith, K. S. (2012). New methods for estimating ocean eddy heat transport using satellite altimetry. *Monthly Weather Review*, 140(5):1703–1722.
- Keating, S. R. and Smith, K. S. (2015). Upper ocean flow statistics estimated from superresolved sea-surface temperature images. *Journal of Geophysical Research: Oceans*, 120(2):1197–1214.
- Kelly, K. A., Small, R. J., Samelson, R., Qiu, B., Joyce, T. M., Kwon, Y.-O., and Cronin, M. F. (2010). Western boundary currents and frontal air–sea interaction: Gulf stream and kuroshio extension. *Journal of Climate*, 23(21):5644–5667.
- Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

- Kirtman, B. P., Bitz, C., Bryan, F., Collins, W., Dennis, J., Hearn, N., Kinter, J. L., Loft, R., Rousset, C., Siqueira, L., et al. (2012). Impact of ocean model resolution on ccsm climate simulations. *Climate dynamics*, 39(6):1303–1328.
- Kjellsson, J. and Zanna, L. (2017). The impact of horizontal resolution on energy transfers in global ocean models. *Fluids*, 2(3):45.
- Klambauer, G., Unterthiner, T., Mayr, A., and Hochreiter, S. (2017). Self-normalizing neural networks. In *Advances in Neural Information Processing Systems*, pages 972–981.
- Klocker, A. and Abernathey, R. (2014). Global patterns of mesoscale eddy properties and diffusivities. *Journal of Physical Oceanography*, 44(3):1030–1046.
- Klocker, A., Ferrari, R., and LaCasce, J. H. (2012). Estimating suppression of eddy mixing by mean flows. *Journal of Physical Oceanography*, 42(9):1566–1576.
- Kraichnan, R. H. (1967). Inertial ranges in two-dimensional turbulence. *The Physics of Fluids*, 10(7):1417–1423.
- Krasnopolsky, V. M., Fox-Rabinovitz, M. S., and Chalikov, D. V. (2005). New approach to calculation of atmospheric model physics: Accurate and fast neural network emulation of longwave radiation in a climate model. *Monthly Weather Review*, 133(5):1370–1383.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- Landahl, M. T., Mollo-Christensen, E., and Korman, M. S. (1989). Turbulence and random processes in fluid mechanics.
- Larraondo, P. R., Renzullo, L. J., Inza, I., and Lozano, J. A. (2019). A data-driven approach to precipitation parameterizations using convolutional encoder-decoder neural networks. *arXiv preprint arXiv:1903.10274*.

- Le Traon, P., Nadal, F., and Ducet, N. (1998). An improved mapping method of multisatellite altimeter data. *Journal of atmospheric and oceanic technology*, 15(2):522–534.
- Leith, C. E. (1968). Diffusion approximation for two-dimensional turbulence. *The Physics of Fluids*, 11(3):671–672.
- Lewy, H., Friedrichs, K., and Courant, R. (1928). Über die partiellen differenzengleichungen der mathematischen physik. *Mathematische annalen*, 100:32–74.
- Lguensat, R., Sommer, J. L., Metref, S., Cosme, E., and Fablet, R. (2019). Learning generalized quasi-geostrophic models using deep neural numerical models. *arXiv preprint arXiv:1911.08856*.
- Ling, J., Jones, R., and Templeton, J. (2016a). Machine learning strategies for systems with invariance properties. *Journal of Computational Physics*, 318:22–35.
- Ling, J., Kurzawski, A., and Templeton, J. (2016b). Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166.
- Liu, Y., Wilson, C., Green, M. A., and Hughes, C. W. (2018). Gulf stream transport and mixing processes via coherent structure dynamics. *Journal of Geophysical Research: Oceans*, 123(4):3014–3037.
- Long, Z., Lu, Y., and Dong, B. (2019). Pde-net 2.0: Learning pdes from data with a numeric-symbolic hybrid deep network. *Journal of Computational Physics*, 399:108925.
- Lorenz, E. N. (1955). Available potential energy and the maintenance of the general circulation. *Tellus*, 7(2):157–167.
- Ma, X., Jing, Z., Chang, P., Liu, X., Montuoro, R., Small, R. J., Bryan, F. O., Greatbatch, R. J., Brandt, P., Wu, D., et al. (2016). Western boundary currents regulated by interaction between ocean eddies and the atmosphere. *Nature*, 535(7613):533.
- Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml*, volume 30, page 3.

- Mak, J., Maddison, J. R., and Marshall, D. P. (2016). A new gauge-invariant method for diagnosing eddy diffusivities. *Ocean Modelling*, 104:252–268.
- Mak, J., Maddison, J. R., Marshall, D. P., and Munday, D. R. (2018). Implementation of a geometrically informed and energetically constrained mesoscale eddy parameterization in an ocean circulation model. *Journal of Physical Oceanography*, 48(10):2363–2382.
- Mana, P. P. and Zanna, L. (2014). Toward a stochastic parameterization of ocean mesoscale eddies. *Ocean Modelling*, 79:1–20.
- Marshall, D. P. and Adcroft, A. J. (2010). Parameterization of ocean eddies: Potential vorticity mixing, energetics and arnold’s first stability theorem. *Ocean Modelling*, 32(3-4):188–204.
- Marshall, D. P., Maddison, J. R., and Berloff, P. S. (2012). A framework for parameterizing eddy potential vorticity fluxes. *Journal of Physical Oceanography*, 42(4):539–557.
- Marshall, J., Adcroft, A., Hill, C., Perelman, L., and Heisey, C. (1997). A finite-volume, incompressible navier stokes model for studies of the ocean on parallel computers. *Journal of Geophysical Research: Oceans*, 102(C3):5753–5766.
- Marshall, J., Jones, H., and Hill, C. (1998). Efficient ocean modeling using non-hydrostatic algorithms. *Journal of Marine systems*, 18(1-3):115–134.
- Marshall, J. and Plumb, R. A. (2016). *Atmosphere, ocean and climate dynamics: an introductory text*. Academic Press.
- Marshall, J., Shuckburgh, E., Jones, H., and Hill, C. (2006). Estimates and implications of surface eddy diffusivity in the southern ocean derived from tracer transport. *Journal of physical oceanography*, 36(9):1806–1821.
- Maulik, R. and San, O. (2017). A neural network approach for the blind deconvolution of turbulent flows. *Journal of Fluid Mechanics*, 831:151–181.

- Maulik, R., San, O., Rasheed, A., and Vedula, P. (2019). Subgrid modelling for two-dimensional turbulence using neural networks. *Journal of Fluid Mechanics*, 858:122–144.
- McElreath, R. (2018). *Statistical rethinking: A Bayesian course with examples in R and Stan*. Chapman and Hall/CRC.
- Minobe, S., Kuwano-Yoshida, A., Komori, N., Xie, S.-P., and Small, R. J. (2008). Influence of the gulf stream on the troposphere. *Nature*, 452(7184):206.
- Moeng, C.-H. (1984). A large-eddy-simulation model for the study of planetary boundary-layer turbulence. *Journal of the Atmospheric Sciences*, 41(13):2052–2062.
- Moore, G. E. et al. (1965). Cramming more components onto integrated circuits.
- Munk, W. H. (1950). On the wind-driven ocean circulation. *Journal of meteorology*, 7(2):80–93.
- Nadiga, B. (2008). Orientation of eddy fluxes in geostrophic turbulence. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1875):2489–2508.
- Nadiga, B. T. and Bouchet, F. (2011). The equivalence of the lagrangian-averaged navier-stokes- α model and the rational large eddy simulation model in two dimensions. *Physics of Fluids*, 23(9):095105.
- Nakamura, N. (1996). Two-dimensional mixing, edge formation, and permeability diagnosed in an area coordinate. *Journal of the atmospheric sciences*, 53(11):1524–1537.
- O’Gorman, P. A. and Dwyer, J. G. (2018). Using machine learning to parameterize moist convection: potential for modeling of climate, climate change and extreme events. *Journal of Advances in Modelling Earth Systems*.
- O’Reilly, C. H., Minobe, S., Kuwano-Yoshida, A., and Woollings, T. (2017). The gulf stream influence on wintertime north atlantic jet variability. *Quarterly Journal of the Royal Meteorological Society*, 143(702):173–183.

- Osborn, T. R. and Cox, C. S. (1972). Oceanic fine structure. *Geophysical & Astrophysical Fluid Dynamics*, 3(1):321–345.
- Palmer, T. (2014). Climate forecasting: Build high-resolution global climate models. *Nature News*, 515(7527):338.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.
- Penduff, T., Juza, M., Brodeau, L., Smith, G., Barnier, B., Molines, J.-M., Tréguier, A.-M., and Madec, G. (2010). Impact of global ocean model resolution on sea-level variability with emphasis on interannual time scales. *Ocean Science*, 6:269–284.
- Pfeffer, W. (1986). The divergence theorem. *Transactions of the American Mathematical Society*, 295(2):665–685.
- Pope, S. (1975). A more general effective-viscosity hypothesis. *Journal of Fluid Mechanics*, 72(2):331–340.
- Pope, S. B. (2001). Turbulent flows.
- Poplin, R., Varadarajan, A. V., Blumer, K., Liu, Y., McConnell, M. V., Corrado, G. S., Peng, L., and Webster, D. R. (2018). Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning. *Nature Biomedical Engineering*, page 1.
- Prandtl, L. (1925). 7. bericht über untersuchungen zur ausgebildeten turbulenz. *ZAMM-Journal of Applied Mathematics and Mechanics/Zeitschrift für Angewandte Mathematik und Mechanik*, 5(2):136–139.
- Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2019). Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707.

- Rasp, S. (2019). Online learning as a way to tackle instabilities and biases in neural network parameterizations. *arXiv preprint arXiv:1907.01351*.
- Rasp, S., Pritchard, M. S., and Gentine, P. (2018). Deep learning to represent sub-grid processes in climate models. *arXiv preprint arXiv:1806.04731*.
- Redi, M. H. (1982). Oceanic isopycnal mixing by coordinate rotation. *Journal of Physical Oceanography*, 12(10):1154–1158.
- Rhines, P. B. (1975). Waves and turbulence on a beta-plane. *Journal of Fluid Mechanics*, 69(3):417–443.
- Rhines, P. B. (1977). The dynamics of unsteady currents. *The sea*, 6:189–318.
- Rhines, P. B. and Young, W. R. (1982). Homogenization of potential vorticity in planetary gyres. *Journal of Fluid Mechanics*, 122:347–367.
- Rich, A. S. (2019). Ai reflections in 2019. *Machine learning*, 1:174–180.
- Rivlin, R. S. and Ericksen, J. L. (1955). Stress-deformation relations for isotropic materials. *Journal of Rational Mechanics and Analysis*, 4:323–425.
- Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.
- Roberts, M. J., Hewitt, H. T., Hyder, P., Ferreira, D., Josey, S. A., Mizielinski, M., and Shelly, A. (2016). Impact of ocean resolution on coupled air-sea fluxes and large-scale climate. *Geophysical Research Letters*, 43(19):10–430.
- Roemmich, D., Johnson, G. C., Riser, S., Davis, R., Gilson, J., Owens, W. B., Garzoli, S. L., Schmid, C., and Ignaszewski, M. (2009). The argo program: Observing the global ocean with profiling floats. *Oceanography*, 22(2):34–43.
- Rosenblatt, F. (1957). *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory.
- Rossby, T. (1996). The north atlantic current and surrounding waters: At the cross-roads. *Reviews of Geophysics*, 34(4):463–481.

- Rudnick, D. L., Davis, R. E., Eriksen, C. C., Fratantoni, D. M., and Perry, M. J. (2004). Underwater gliders for ocean research. *Marine Technology Society Journal*, 38(2):73–84.
- Rudy, S. H., Brunton, S. L., Proctor, J. L., and Kutz, J. N. (2017). Data-driven discovery of partial differential equations. *Science Advances*, 3(4):e1602614.
- Sagaut, P. (2006). *Large eddy simulation for incompressible flows: an introduction*. Springer Science & Business Media.
- Salehipour, H. and Peltier, W. R. (2018). Deep learning of mixing by two ‘atoms’ of stratified turbulence. *arXiv preprint arXiv:1809.06499*.
- Salmon, R. (1978). Two-layer quasi-geostrophic turbulence in a simple special case. *Geophysical & Astrophysical Fluid Dynamics*, 10(1):25–52.
- Samelson, R. (1992). Fluid exchange across a meandering jet. *Journal of physical oceanography*, 22(4):431–444.
- San, O. and Maulik, R. (2018). Extreme learning machine for reduced order modeling of turbulent geophysical flows. *Physical Review E*, 97(4):042322.
- Schaller, R. R. (1997). Moore’s law: past, present and future. *IEEE spectrum*, 34(6):52–59.
- Schmitz Jr, W. J. and Holland, W. R. (1986). Observed and modeled mesoscale variability near the gulf stream and kuroshio extension. *Journal of Geophysical Research: Oceans*, 91(C8):9624–9638.
- Schneider, T., Lan, S., Stuart, A., and Teixeira, J. (2017a). Earth system modeling 2.0: A blueprint for models that learn from observations and targeted high-resolution simulations. *Geophysical Research Letters*, 44(24):12–396.
- Schneider, T., Teixeira, J., Bretherton, C. S., Brient, F., Pressel, K. G., Schär, C., and Siebesma, A. P. (2017b). Climate goals and computing the future of clouds. *Nature Climate Change*, 7(1):3.

- Scott, R. B. and Arbic, B. K. (2007). Spectral energy fluxes in geostrophic turbulence: Implications for ocean energetics. *Journal of physical oceanography*, 37(3):673–688.
- Scott, R. B. and Wang, F. (2005). Direct evidence of an oceanic inverse kinetic energy cascade from satellite altimetry. *Journal of Physical Oceanography*, 35(9):1650–1666.
- Shuckburgh, E., Jones, H., Marshall, J., and Hill, C. (2009). Understanding the regional variability of eddy diffusivity in the pacific sector of the southern ocean. *Journal of Physical Oceanography*, 39(9):2011–2023.
- Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Smagorinsky, J. (1963). General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, 91(3):99–164.
- Starr, V. (1966). Physics of negative viscosity phenomena. *Earth and Planetary Science Series*, 256.
- Stommel, H. (1948). The westward intensification of wind-driven ocean currents. *Eos, Transactions American Geophysical Union*, 29(2):202–206.
- Su, H., Li, W., and Yan, X.-H. (2018a). Retrieving temperature anomaly in the global subsurface and deeper ocean from satellite observations. *Journal of Geophysical Research: Oceans*, 123(1):399–410.
- Su, Z., Wang, J., Klein, P., Thompson, A. F., and Menemenlis, D. (2018b). Ocean submesoscales as a key component of the global heat budget. *Nature communications*, 9(1):775.
- Taylor, K. E., Stouffer, R. J., and Meehl, G. A. (2012). An overview of cmip5 and the experiment design. *Bulletin of the American Meteorological Society*, 93(4):485–498.
- Thorpe, S. A. (1977). Turbulence and mixing in a scottish loch. *Philosophical Transactions of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 286(1334):125–181.

- Tipping, M. E. (2001). Sparse bayesian learning and the relevance vector machine. *Journal of machine learning research*, 1(Jun):211–244.
- Tracey, B. D., Duraisamy, K., and Alonso, J. J. (2015). A machine learning strategy to assist turbulence model development. In *53rd AIAA Aerospace Sciences Meeting*, page 1287.
- Vallis, G. K. (2017). *Atmospheric and oceanic fluid dynamics*. Cambridge University Press.
- Villas Bôas, A., Sato, O., Chaigneau, A., and Castelão, G. (2015). The signature of mesoscale eddies on the air-sea turbulent heat fluxes in the south atlantic ocean. *Geophysical Research Letters*, 42(6):1856–1862.
- Waldrop, M. M. (2016). The chips are down for moore’s law. *Nature News*, 530(7589):144.
- Waterman, S., Hogg, N. G., and Jayne, S. R. (2011). Eddy–mean flow interaction in the kuroshio extension region. *Journal of Physical Oceanography*, 41(6):1182–1208.
- Waterman, S. and Hoskins, B. J. (2013). Eddy shape, orientation, propagation, and mean flow feedback in western boundary current jets. *Journal of Physical Oceanography*, 43(8):1666–1690.
- Waterman, S. and Jayne, S. R. (2010). Eddy-mean flow interactions in the along-stream development of a western boundary current jet: An idealized model study. *Journal of Physical Oceanography*.
- Wilson, C. (2016). Does the wind systematically energize or damp ocean eddies? *Geophysical Research Letters*, 43(24):12–538.
- Wilson, C. and Williams, R. G. (2004). Why are eddy fluxes of potential vorticity difficult to parameterize? *Journal of physical oceanography*, 34(1):142–155.
- Wolpert, D. H., Macready, W. G., et al. (1997). No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82.

- Woollings, T., Hoskins, B., Blackburn, M., Hassell, D., and Hodges, K. (2010). Storm track sensitivity to sea surface temperature resolution in a regional atmosphere model. *Climate dynamics*, 35(2-3):341–353.
- Wortham, C. and Wunsch, C. (2014). A multidimensional spectral description of ocean variability. *Journal of Physical Oceanography*, 44(3):944–966.
- Wunsch, C. (1997). The vertical partition of oceanic horizontal kinetic energy. *Journal of Physical Oceanography*, 27(8):1770–1794.
- Wunsch, C. (1999). Where do ocean eddy heat fluxes matter? *Journal of Geophysical Research: Oceans*, 104(C6):13235–13249.
- Xu, C., Zhai, X., and Shang, X.-D. (2016). Work done by atmospheric winds on mesoscale ocean eddies. *Geophysical Research Letters*, 43(23):12–174.
- Xu, H., Chang, H., and Zhang, D. (2019). Dl-pde: Deep-learning based data-driven discovery of partial differential equations from discrete and noisy data. *arXiv preprint arXiv:1908.04463*.
- Zanna, L. and Bolton, T. (2020). Data-driven equation discovery of ocean mesoscale closures. *Geophysical Research Letters*, page e2020GL088376.
- Zanna, L., Brankart, J., Huber, M., Leroux, S., Penduff, T., and Williams, P. (2018). Uncertainty and scale interactions in ocean ensembles: From seasonal forecasts to multidecadal climate predictions. *Quarterly Journal of the Royal Meteorological Society*.
- Zanna, L., Mana, P. P., Anstey, J., David, T., and Bolton, T. (2017). Scale-aware deterministic and stochastic parametrizations of eddy-mean flow interaction. *Ocean Modelling*, 111:66–80.
- Zhang, S. and Lin, G. (2018). Robust data-driven discovery of governing physical laws with error bars. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2217):20180305.

Zhou, Z., He, G., Wang, S., and Jin, G. (2019). Subgrid-scale model for large-eddy simulation of isotropic turbulent flows using an artificial neural network. *Computers & Fluids*, page 104319.