High-Resolution Numerical Simulations of Barotropic Wind-Driven Gyres

by

William Ko

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Applied Mathematics

Waterloo, Ontario, Canada, 2011

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The dynamics of the world's oceans occur at a vast range of length scales. Although there are theories that aid in understanding the dynamics at planetary scales and microscales, the motions in between are still not yet well understood. This work discusses a numerical model to study barotropic wind-driven gyre flow that is capable of resolving dynamics at the synoptic, $O(10^3\,\mathrm{km})$, mesoscale, $O(10^2\,\mathrm{km})$ and submesoscales $O(10\,\mathrm{km})$. The Quasi-Geostrophic (QG) model has been used predominantly to study ocean circulations but it is limited as it can only describe motions at synoptic scales and mesoscales. The Rotating Shallow Water (SW) model that can describe dynamics at a wider range of horizontal length scales and can better describe motions at the submesoscales. Numerical methods that are capable of high-resolution simulations are discussed for both QG and SW models and the numerical results are compared. To achieve high accuracy and resolve an optimal range of length scales, spectral methods are applied to solve the governing equations and a third-order Adams-Bashforth method is used for the temporal discretization. Several simulations of both models are computed by varying the strength of dissipation. The simulations either tend to a laminar steady state, or a turbulent flow with dynamics occurring at a wide range of length and time scales. The laminar results show similar behaviours in both models, thus QG and SW tend to agree when describing slow, large-scale flows. The turbulent simulations begin to differ as QG breaks down when faster and smaller scale motions occur. Essential differences in the underlying assumptions between the QG and SW models are highlighted using the results from the numerical simulations.

Acknowledgements

First of all, I would like to thank my supervisor, Dr. Francis J. Poulin, for accepting me as a graduate student and providing me with an opportunity to learn how to be an academic writer and researcher. I am grateful for the helpful comments and suggestions that he has provided in writing this thesis. Also, I would like to thank Helen Warren for always being so helpful with any administrative concerns I have had. Finally, I would like to thank Dr. Mike Waite and Dr. Nathalie Lanson for serving on the reading committee for this thesis.

This work was made possible by the facilities of the Shared Hierarchical Academic Research Computing Network (SHARCNET) and Compute Canada.

Table of Contents

List of Tables vii

List of Figures ix

List of Algorithms x

1 Introduction 1
- 1.1 Thesis Outline . . . 3

2 Numerical Methods 4
- 2.1 Model Setup . . . 4
- 2.2 Numerical Method for SW Model . . . 6
  - 2.2.1 Physical Model . . . 6
  - 2.2.2 Spatial Discretization . . . 8
  - 2.2.3 FFTW . . . 10
  - 2.2.4 Time-Stepping Method . . . 13
  - 2.2.5 Parallel Implementation . . . 15
  - 2.2.6 Filtering . . . 16
  - 2.2.7 Pseudocode . . . 18
- 2.3 Numerical Method for QG Model . . . 18
  - 2.3.1 Physical Model . . . 18
  - 2.3.2 Spatial Discretization . . . 21
  - 2.3.3 Chebyshev Polynomial Interpolation . . . 22
  - 2.3.4 Time-Stepping Method . . . 23
  - 2.3.5 Inverse Problem . . . 25
  - 2.3.6 Parallel Implementation . . . 26
  - 2.3.7 Filtering . . . 27
  - 2.3.8 Pseudocode . . . 28

3 Numerical Results 31
- 3.1 Computational Parameters . . . 31
- 3.2 Simulations of Rotating Shallow Water Model . . . 33
- 3.3 Comparison of QG and SW Models . . . 43
  - 3.3.1 Barotropic Gyre Flow . . . 43
  - 3.3.2 Kinetic Energy Spectra . . . 48

3.3.3 Horizontal Divergence 50
3.3.4 Surface Gravity Waves 50

4 Conclusion **54**
4.1 Summary 54
4.2 Future Work 55

References **57**

List of Tables

2.1 Discrete Cosine Transforms implemented in the FFTW library 10
2.2 Discrete Sine Transforms implemented in the FFTW library 10
2.3 Inverse DCT and DST 11
2.4 Parallel efficiency of the SW solver of a test simulation on a 2048×2048 grid 17
2.5 Parallel efficiency of the QG solver of a test simulation on a 512×512 grid 27
2.6 Parallel efficiency of the QG solver of a test simulation on a 256×2048 grid 27

3.1 Physical parameters used in numerical simulations. 31
3.2 Simulation cases 32
3.3 Maximum values of mean meridional velocity, v, and relative vorticity, ζ, computed from the SW and QG simulations within a distance of $2\delta_M$ from the western boundary. 47

List of Figures

2.1 Values of $\gamma\Delta t$ on the complex plane such that the third-order Adams-Bashforth method is stable (solid contour). $|\gamma\Delta t| = 0.5$ (dashed curve) . 14
2.2 Computational grid of SW problem. Solutions are calculated on $*$. The numerical domain of dependence is shown by the solid lines with slope $\pm\Delta t/\Delta x$. The physical domain of dependence is shown by the dashed lines with slope $\pm 1/c$. 15
2.3 Schematic of the computational domain divided among four processors, P1 to P4, in the a) zonal direction, b) meridional direction 16
2.4 An example of aliasing on 8 grid points. The dashed sine wave has frequency 14π but is misinterpreted as a wave with frequency -2π. 17
2.5 A Chebyshev grid with $N = 7$ is formed by projecting points that are equidistant on a circle down onto the x-axis. $\{x_i\}$ is defined in descending order with $x_0 = 1$ and $x_N = -1$ 22

3.1 $y = 0$ cross-section of the normalized, time-averaged meridional velocity at the western boundary of SW simulation. 33
3.2 free-surface plot of SW simulation for $\delta_M = 200\,\text{km}$ at day 200. 36
3.3 free-surface plot of SW simulation for $\delta_M = 100\,\text{km}$ at day 800. 36
3.4 free-surface plot of SW simulation for $\delta_M = 50\,\text{km}$ at days 25, 50, 75 and 100. 37
3.5 free-surface plot of SW simulation for $\delta_M = 50\,\text{km}$ at day 5000. 38
3.6 free-surface plot of SW simulation for $\delta_M = 25\,\text{km}$ at days 25, 50, 75 and 100. 39
3.7 free-surface plot of SW simulation for $\delta_M = 12.5\,\text{km}$ at days 25, 50, 75 and 100. 40
3.8 Power spectrum of η, u and v for $\delta_M = 50\,\text{km}$ at day 500. The wavelengths to the left of the dashed line are the filtered modes. δ_M is indicated by the dash-dot line. 41
3.9 Power spectrum of η, u and v for $\delta_M = 25\,\text{km}$ at day 500. The wavelengths to the left of the dashed line are the filtered modes. δ_M is indicated by the dash-dot line. 41
3.10 Power spectrum of η, u and v for $\delta_M = 12.5\,\text{km}$ at day 500. The wavelengths to the left of the dashed line are the filtered modes. δ_M is indicated by the dash-dot line. 42

3.11 Power spectrum of the mean kinetic energy. Slopes representing k^{-3} and k^{-6} are indicated by the dashed lines 42
3.12 Stream-function (left) and relative vorticity (right) plots of QG simulation for $\delta_M = 200, 100$ and 50 km at equilibrium. 44
3.13 Stream-function (left) and relative vorticity (right) plots of QG simulation for $\delta_M = 25$ km at days 25, 50 and 75. 45
3.14 Stream-function (left) and relative vorticity (right) plots of QG simulation for $\delta_M = 12.5$ km at days 25, 50 and 75. 46
3.15 $y = 0$ cross-section of the normalized, time-averaged meridional velocity near the western boundary of QG simulation. 47
3.16 Extreme values of relative vorticity in the QG and SW simulations for case 5 48
3.17 Kinetic energy spectra of time-averaged mean states for QG (solid) and SW (dashed) simulations. 49
3.18 Maximum divergence of SW simulations over 500 days 51
3.19 Close-up of a) relative vorticity and b) $\log|\frac{\partial h}{\partial t}|$ for $\delta_M = 12.5$ km simulation on day 239. 52
3.20 Close-up of a) relative vorticity and b) $\log|\frac{\partial h}{\partial t}|$ for $\delta_M = 12.5$ km simulation on day 301. 53

List of Algorithms

2.1 `function DCT_dx`: Compute first derivative with DCT 12
2.2 `function DST_dx`: Compute first derivative with DST 12
2.3 `function DCT_dx`^{2}: Compute second derivative with DCT 12
2.4 `function DST_dx`^{2}: Compute second derivative with DST 12
2.5 Numerical method to solve Rotating Shallow Water equations 19
2.6 Filtering procedure for SW solver 19
2.7 Numerical method to solve Quasi-Geostrophy equations 29
2.8 `function` F_{QG} 29
2.9 Filtering procedure for QG solver 30
2.10 QG Inverse Problem 30

Chapter 1

Introduction

The dynamics of the world's oceans occur at a vast range of length scales and are influenced by continental boundaries, Earth's rotation, atmospheric winds and many other factors. At the synoptic length scale, $O(10^3\,\mathrm{km})$, energy is entered into the ocean by external forcing, such as atmospheric winds, and the motions are essentially two-dimensional. This energy cascades across different length scales until, at the microscales, the energy is dissipated by molecular viscosity and the motions are fully three-dimensional. Models exists that describe the dynamics of water at the synoptic scales and microscales, but to truly understand the complicated dynamics of the ocean, the intermediate length scales must be thoroughly studied.

Modern understanding of large scale, wind-driven gyre flow originated with the work of Sverdrup in 1947 [37]. In his paper, Sverdrup found that the meridional transport in most of the ocean is directly proportional to the vorticity induced by the wind stress. This relationship is now referred to as the *Sverdrup Relation*. Sverdrup's theory is only valid in the open ocean where frictional and nonlinear effects are negligible. Thus, there is a region in the ocean where the Sverdrup Relation breaks down and a boundary layer solution exists where the neglected effects must be considered. Soon after, it was discovered by Stommel that the vorticity added by the atmospheric winds is necessarily dissipated along the western boundary current (WBC) and the width of this current is directly related to the strength of the dissipation [36]. Shortly after Stommel's work, Munk studied wind-driven flows by modelling the dissipation using a lateral friction term and found that the WBC width can, again, be parameterized by the strength of the dissipation [23]. These theories have been useful in understanding the qualitative behaviour of large-scale ocean dynamics but they are insufficient as real oceans are turbulent in nature and smaller scale dynamics must be considered.

To understand why a western intensification occurs, we first introduce *potential vorticity* [24],

$$q = \frac{f + \zeta}{h}, \tag{1.1}$$

where f is the ambient or planetary vorticity due to Earth's rotation, ζ is the relative

vorticity and h is the fluid depth. The potential vorticity can be interpreted as a ratio between the circulation and the volume of a fluid column. An important feature of the potential vorticity is that it is a conserved quantity following the flow [24]. Now for the sake of argument, suppose we look at a subtropical gyre on the northern hemisphere where the curl of the wind stress is negative. From Sverdrup's theory, the meridional transport is southward for the majority of the ocean and conservation of mass requires that a northward transport occurs within the boundary layer. To arrive at a contradiction suppose that the boundary layer occurs along the eastern wall [7]. Within the boundary layer, any changes in the zonal velocity is small compared to the steep increase of the meridional velocity. Thus the vorticity can be approximated by

$$\zeta = \frac{\partial v}{\partial x} - \frac{\partial u}{\partial y} \approx \frac{\partial v}{\partial x}. \tag{1.2}$$

Based on the supposition that the northward velocity occurs on the eastern boundary, there is a region near the eastern boundary such that the zonal gradient of v is positive. Therefore, we have $\zeta > 0$. Since the flow where Sverdrup's theory is valid is slow, we have $\zeta \approx 0$, but relative vorticity becomes important in the boundary layer so as f increases northward, conservation of potential vorticity requires ζ to decrease to a negative value within the boundary layer, which is a contradiction. If we assumed that the boundary layer occurs along the western wall, no contradiction arises. Thus the boundary layer must occur along the western boundary and the eastern boundary satisfies Sverdrup's solution. It is interesting to note that WBCs occur in both northern and southern hemispheres since f is negative south of the equator.

Numerical computation is a pivotal tool in studying fluid dynamics. However, even with powerful modern computers, it is impossible to resolve all length and time scales that exist in a real ocean. This thesis presents a high-resolution numerical model that takes advantage of modern, high-performance computer clusters in an attempt to resolve a greater range of length and time scales than what has been previously considered. The primary model for studying geophysical fluid dynamics is the Quasi-Geostrophic (QG) model due to its simplicity and accuracy in large-scale flows. Although a lot has been learned through this model, it is inaccurate when dynamics enter the submesoscale regime, $O(10\,\text{km})$. To achieve more physically relevant results at smaller length scales, the Rotating Shallow Water (SW) model is used as it can resolve a greater range of horizontal length scales. This work focusses on barotropic flows where it is assumed that the motions are uniform in depth.

The SW model is an improvement over the QG model to study the mechanisms that create a western intensification in wind-driven gyres as it can describe motions at the synoptic scale, mesoscale and submesoscale. Submesoscale dynamics in a real ocean tend to be more three-dimensional, but the velocity field in the SW model is two-dimensional and therefore cannot accurately describe many aspects of submesoscale motions. However, the SW model allows for a free-surface and assumes three-dimensional incompressibility which should yield some insight into the generation of the submesoscale dynamics that QG cannot. Furthermore, the free-surface in the SW model allows for the generation and propagation of gravity waves that are absent in the QG model.

Several numerical models exist for studying wind-driven, barotropic gyre flow. LaCasce [19] studied turbulent oceans by solving the QG equation with a numerical model that uses finite differences to compute spatial derivatives and Fourier sine transforms to invert the vorticity to obtain the stream-function. Fox-Kemper [11] uses a Chebyshev discretization in both zonal and meridional directions to study large-scale, low Reynolds number, wind-driven gyre flow with variable viscosity. Both of these models are in the context of QG and thus have its limitations in resolving small scale dynamics. The major advantage that the numerical model in this thesis presents is the ability to resolve more accurate dynamics that occur at synoptic length scales down to the submesoscales. This is achieved by implementing a high-performance algorithm and also solving the SW equations.

Numerical studies of wind-driven flow in a shallow water context includes work done by Primeau and Newman [28] which studied double-gyre flow in a reduced gravity SW model. However, the focus of this study was to find steady state solutions at low Reynolds numbers whereas this thesis is more interested in turbulent dynamics which occur at high Reynolds numbers. Crowley [6] developed a finite differencing scheme for wind-driven barotropic gyre flow however it does not resolve wide range of motions. Yuan and Hamilton [40] used a spectral method to solve the SW model on a f-plane. Their numerical model is different as they studied turbulent flow on a doubly-periodic domain with random external forcing. We look at a bounded domain problem with a forcing that simulates a single-gyre flow at a higher grid resolution.

1.1 Thesis Outline

The goal of this thesis is to discuss high-resolution numerical methods for both wind-driven ocean models and then compare the numerical results. Several features of ocean dynamics that exist in the SW model but are absent in the QG model are highlighted. Chapter 2 provides a detailed description of the numerical methods for the barotropic, wind-driven ocean models. Section 2.1 sets up the coordinate system on a rotating reference frame to incorporate the Earth's rotation and its curvature then describes an ideal ocean basin, boundary conditions and wind forcing that are appropriate for a single-gyre flow. Section 2.2 discusses a Fourier spectral method to solve the SW equations. Section 2.3 discusses a Chebyshev-Fourier spectral method to solve the QG equations. Chapter 3 presents the numerical simulations of both models. Section 3.1 presents physical parameters that are used in the simulations that are typically observed in a midlatitude, subtropical gyre in the northern hemisphere. Section 3.2 shows simulations of the SW model that show dynamics at a wide range of length and time scales then comparisons between the SW and QG simulations are discussed in section 3.3. Chapter 4 summarizes the discussion and presents future work.

Chapter 2

Numerical Methods

In this chapter, we discuss numerical methods to solve the barotropic, wind-driven SW and QG models. We begin by setting up an idealized ocean basin and prescribing boundary conditions appropriate for a single-gyre simulation. Section 2.2 discusses a Fourier spectral method to solve the SW equations. Section 2.3 discusses a Chebyshev-Fourier spectral method to solve the QG equations. For both methods, a third-order temporal discretization is discussed as well as a parallel implementation to improve runtime of the simulation. Also, spectral filtering procedures are presented to overcome the numerical instability that occurs in the method. Pseudocode is presented to summarize the methods discussed in each section.

2.1 Model Setup

The models are situated on a rotating reference frame to incorporate the Earth's spin around its axis. The effects of Earth's rotation is denoted by the Coriolis parameter,

$$f = 2\Omega_E \sin(\theta), \tag{2.1}$$

where Ω_E is Earth's rotation rate and θ is the latitude measured from the equator. The β-plane approximation is used where the Coriolis parameter is approximated linearly by,

$$f = f_0 + \beta y, \tag{2.2}$$

where

$$f_0 = 2\Omega_E \sin(\theta_0) \tag{2.3}$$

and

$$\beta = 2\frac{\Omega_E}{R_E}\cos(\theta_0). \tag{2.4}$$

R_E is the mean radius of the Earth and θ_0 is a reference latitude. The models describe motions on a cartesian coordinate system where the x and y coordinates represent the zonal (east) and meridional (north) directions, respectively. The velocity components

in those directions will be denoted u and v, with $\vec{u} = (u, v)$. As the velocity field in both models are two-dimensional, the gradient operator, $\vec{\nabla}$, will only be applied to the horizontal components,

$$\vec{\nabla} = \hat{i}\frac{\partial}{\partial x} + \hat{j}\frac{\partial}{\partial y}. \tag{2.5}$$

We idealize the ocean by assuming our domain is a rectangular box described by

$$\Omega = \left[-\frac{L_x}{2}, \frac{L_x}{2}\right] \times \left[-\frac{L_y}{2}, \frac{L_y}{2}\right], \tag{2.6}$$

where L_x and L_y represent the basin lengths in the zonal and meridional directions, respectively. To ensure that there is no flow leaving or entering the system, we impose a no normal flow condition on the entire boundary,

$$\vec{u} \cdot \hat{n} = 0 \text{ on } \Gamma, \tag{2.7}$$

where $\hat{n}$ is an outward pointing unit vector normal to the boundary, Γ. On the eastern and western boundaries, we simulate continental coastlines by assuming that there is no tangential flow (or no slip) on the boundaries,

$$\vec{u} \cdot \hat{t} = 0 \text{ on } \Gamma_W \cup \Gamma_E, \tag{2.8}$$

where $\hat{t}$ is a unit vector tangential to the boundary. For large-scale flows, it is not entirely clear as to which boundary condition should be used to emulate coastlines. The no slip condition is chosen in our model to get a strong shear in the WBC. A real ocean is not entirely encompassed by solid boundaries, and so we impose no stress (sometimes called free slip) condition on the north and the south by simulating "fluid" boundaries,

$$\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} = 0 \text{ on } \Gamma_N \cup \Gamma_S. \tag{2.9}$$

The last boundary condition also prevents a shear instability from occurring along the north and south boundaries which is typically not observed in a real ocean [11].

The wind forcing of choice is appropriate for a subtropical, single-gyre simulation,

$$\vec{\tau} = \frac{\tau_0}{\rho_0 H} \sin\left(\frac{\pi y}{L_y}\right) \hat{i}, \tag{2.10}$$

where τ_0 is the forcing strength measured in $\mathrm{N/m^2}$, ρ_0 is the fluid density and H is the mean fluid depth. There is no zonal variation and thus the curl depends on the meridional derivative,

$$\hat{k} \cdot \vec{\nabla} \times \vec{\tau} = -\frac{\pi \tau_0}{\rho_0 H L_y} \cos\left(\frac{\pi y}{L_y}\right). \tag{2.11}$$

This is negative throughout Ω and thus this form of wind forcing inputs negative vorticity locally throughout the basin. Integrating the above equation across the entire basin gives,

$$\iint_\Omega \hat{k} \cdot \vec{\nabla} \times \vec{\tau}\, dA = -\frac{2L_x \tau_0}{\rho_0 H} < 0, \tag{2.12}$$

implying that the single-gyre wind forcing inputs net negative vorticity into the system. It is shown in [24] that the vorticity entering the system due to the wind stress is dissipated within the WBC. Thus, a mechanism is required to simulate the dissipation. We use lateral viscosity in our models that can be interpreted as sub-grid scale viscosity that parameterizes the effects of small-scale motions that are not resolved in our numerical model. The lateral dissipation is preferable in our models since the higher-order derivatives will allow us to impose all of the boundary conditions, (2.7)-(2.9). The details of the dissipation will be discussed in later sections.

When comparing with a double-gyre simulation,

$$\vec{\tau} = -\frac{\tau_0}{\rho_0 H}\cos\left(\frac{2\pi y}{L_y}\right)\hat{i}, \tag{2.13}$$

the problem becomes more simple. Integrating the new wind forcing above shows that the global vorticity input is zero. In a double-gyre simulation, a clockwise flow (negative vorticity) in the northern gyre is balanced by a counter-clockwise flow (positive vorticity) in the southern gyre. Thus a vorticity balance is achieved by the cyclone-anticlone pair and dissipation need not be considered [28].

2.2 Numerical Method for SW Model

2.2.1 Physical Model

As the name suggests, the SW model describes motion of a shallow layer of fluid with aspect ratio, $H/L \ll 1$, where L is a horizontal length scale and H is the fluid layer depth. In our model, we use $H = 500\,\text{m}$ thus SW is valid when dynamics have horizontal length scales greater than 1 km. The small aspect ratio allows the assumption that the fluid is homogeneous and the vertical variation in the horizontal velocities can be neglected. We can also assume that the fluid is in hydrostatic balance,

$$\frac{\partial p}{\partial z} = -g\rho, \tag{2.14}$$

which simplifies to

$$p = p_0 - gh\rho_0 \tag{2.15}$$

since density is assumed constant. The important aspect of pressure is its gradient which can be written in terms of the fluid depth, $h(x, y, t)$,

$$\frac{1}{\rho_0}\vec{\nabla}p = -g\vec{\nabla}h \tag{2.16}$$

Thus for the SW model, the momentum equation is,

$$\frac{\partial \vec{u}}{\partial t} + \vec{u}\cdot\vec{\nabla}\vec{u} + f\hat{k}\times\vec{u} = -g\vec{\nabla}h + \vec{D} + \vec{\tau}. \tag{2.17}$$

where g is the acceleration due to gravity and $\vec{D}$ is a dissipation function. By depth-integrating the three-dimensional continuity equation,

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0, \tag{2.18}$$

and applying the kinematic boundary condition at the surface and no-normal flow at the flat bottom, we arrive at an evolution equation for the fluid depth,

$$\frac{\partial h}{\partial t} + \vec{\nabla} \cdot (h\vec{u}) = 0. \tag{2.19}$$

The wind forcing, used for the SW model is,

$$\vec{\tau} = \frac{\tau_0}{\rho_0 h} \sin\left(\frac{\pi y}{L_y}\right) \hat{i}. \tag{2.20}$$

This differs from (2.10) where this forcing is normalized by h, which varies in time and space, rather than the H. It can be shown that this form induces a net negative vorticity into the system as well. A lateral viscosity term is included in the model to dissipate the vorticity in the system. A common choice for lateral viscosity is,

$$\vec{D} = \nu \nabla^2 \vec{u}, \tag{2.21}$$

where ν is the kinematic viscosity coefficient. However, for the SW model it can be shown that this form of dissipation can cause the kinetic energy to increase locally which is contrary to what one would expect to happen physically. This is shown in [14] and a suggestion for dissipation is given such that the model is energetically consistent. For our model we choose a dissipation that is derived in [30] which uses first principles and chooses the stress tensor appropriately. In component form it is,

$$\vec{D} = \nu \begin{bmatrix} \nabla^2 u + (u_x - v_y)\frac{h_x}{h} + (u_y + v_x)\frac{h_y}{h} \\ \nabla^2 v + (u_y + v_x)\frac{h_x}{h} - (u_x + v_y)\frac{h_y}{h} \end{bmatrix}. \tag{2.22}$$

The boundary conditions (2.7), (2.8) and (2.9) simplifies to

$$u = 0, \tag{2.23}$$

$$v = 0 \text{ on } \Gamma_W \cup \Gamma_E, \tag{2.24}$$

and

$$\frac{\partial u}{\partial y} = 0, \tag{2.25}$$

$$v = 0 \text{ on } \Gamma_N \cup \Gamma_S, \tag{2.26}$$

on the rectangular basin. To conserve mass in the system, a no-flux boundary condition is imposed on the fluid depth,

$$\frac{\partial h}{\partial x} = 0 \text{ on } \Gamma_W \cup \Gamma_E, \tag{2.27}$$

$$\frac{\partial h}{\partial y} = 0 \text{ on } \Gamma_N \cup \Gamma_S. \tag{2.28}$$

The evolution of the fluid depth implies that there is a free-surface, $\eta = h - H$. This free-surface introduces *surface gravity waves* which have speeds that are $O(\sqrt{gH})$, which can be very fast for a deep ocean. The variation of the Coriolis parameter with latitude allows for planetary waves to exist, also known as Rossby waves [24]. These waves exist in the QG model as well. Rossby waves have speeds that are $O(gH\beta/f_0^2)$, which can be smaller than the speed of gravity waves by several orders of magnitude. Simultaneously including these two waves, one fast one slow, makes the SW equations very stiff. This fact will be carefully considered when developing the numerical method.

2.2.2 Spatial Discretization

For the analysis of the methods presented in this chapter, we take a Method of Lines (MOL) approach that is discussed in [21] and [38]. The idea is to first consider a spatial discretization of the partial differential equations (PDE) which results in a system of ordinary differential equations (ODE) that depend on time. We then apply a temporal discretization scheme to the ODE system which evolves the original PDE over time.

Spectral methods are used for the spatial discretization. It is assumed that each field is of the form

$$\sum_{m,n} \hat{a}_{mn} \Psi_m(x) \Phi_n(y), \tag{2.29}$$

for globally interpolating orthogonal basis functions $\Psi_m(x)$ and $\Phi_n(y)$. The spatial derivatives are approximated in terms of these basis functions. Spectral methods are advantageous as numerical accuracy can be improved by increasing the number of basis functions used in the approximation. This is known as *spectral accuracy* [38]. The choice of basis functions depend on the particular problem to be solved.

We begin by defining a computational domain by discretizing Ω on a uniform $N_x \times N_y$ grid as follows,

$$(x_i, y_j) = \left(-\frac{L_x}{2} + i\Delta x, -\frac{L_y}{2} + j\Delta y \right), \tag{2.30}$$

for $i = 0, \ldots, N_x - 1,\ j = 0, \ldots, N_y - 1$, and where $\Delta x = L_x/N_x$ and $\Delta y = L_y/N_y$. To capture motions at the optimal range of length scales, we use a Fourier basis for our spectral method. Suppose $[U_0, \ldots, U_{N-1}]$ is a discretized field in one dimension, then we represent the field by a linear combination of Fourier modes by

$$U_j = \sum_n \hat{U}_n e^{ik_n j}, \tag{2.31}$$

where $j = 0, \ldots, N-1$ and $k_n = \frac{2\pi n}{L}$ are wave numbers. The complex Fourier coefficients, $\hat{U}_n$, are obtained by the Discrete Fourier Transform (DFT) defined by

$$\hat{U}_n = \frac{1}{N} \sum_j U_j e^{-ik_n j}, \tag{2.32}$$

where $n = -\frac{N}{2}+1, \ldots, \frac{N}{2}$ [17].

Using a Fourier basis in our spectral method has its advantages. There are Fast Fourier Transform (FFT) algorithms available that are $O(N\log(N))$ as $N \to \infty$ for one-dimensional transforms, as opposed to a brute force calculation which is $O(N^2)$ [3]. Also, computing derivatives become an algebraic operation by scaling Fourier modes by wave numbers. The first-order derivatives can be computed by

$$U_j' = \sum_n ik_n \hat{U}_n e^{ik_n j}, \tag{2.33}$$

and the second-order derivatives by

$$U_j'' = \sum_n -k_n^2 \hat{U}_n e^{ik_n j}, \tag{2.34}$$

for each $j = 0, \ldots, N-1$.

The DFT assumes that the array is periodic but our simulations impose boundary conditions that are non-periodic. We take advantage of the fact that each field prescribes either a zero-Dirichlet condition (no normal flow and no slip) or a zero-Neumann condition (free slip and no flux) in each direction. For the zero-Dirichlet condition, only sine modes are used to ensure that the boundary value is always zero and only cosine modes are used to impose the zero-Neumann condition. Thus each field can be written as,

$$u(x,y) = \sum_{m,n} \hat{u}_{mn} \sin(k_m x)\cos(l_n y), \tag{2.35}$$

$$v(x,y) = \sum_{m,n} \hat{v}_{mn} \sin(k_m x)\sin(l_n y), \tag{2.36}$$

$$h(x,y) = \sum_{m,n} \hat{h}_{mn} \cos(k_m x)\cos(l_n y), \tag{2.37}$$

where $k_n = \frac{2\pi n}{L_x}$ and $l_m = \frac{2\pi m}{L_y}$ are wave numbers in the x and y directions, respectively. Since $\frac{d}{dx}\cos(x) = -\sin(x)$ and $\frac{d}{dx}\sin(x) = \cos(x)$, one must be careful when computing derivatives in Fourier space and be certain that the correct inverse transforms are used. When computing the derivative of an even transform, the spectral coefficients must be multiplied by -1. This is not done for derivatives of odd transforms. Also, the derivative of an odd function is even, and vice versa, so when computing the derivative, the appropriate inverse transform must be used. The coefficients in (2.35)-(2.37) are real so using FFT algorithms designed specifically for Discrete Cosine Transforms (DCTs) and Discrete Sine Transforms (DSTs) will require half the memory and computation time compared to full DFT algorithms [38] .

In Fourier theory, products in physical space are equivalent to convolutions in Fourier space. However, when computing these, term-by-term products are less expensive than convolutions. Thus due to the nonlinear terms in the SW equations, upon computing the Fourier coefficients and the appropriate derivatives, it is necessary to transform back to physical space to compute the SW evolution equations, (2.17) and (2.19). Although this requires more DFTs to be performed, it is more efficient overall.

Type	Boundary Conditions	Formula
DCT-I	even at $j=0$, even at $j=N-1$	$\hat{U}_n = U_0+(-1)^n U_{N-1}+2\sum_{j=1}^{N-2} U_j \cos\left(\frac{\pi jn}{N-1}\right)$
DCT-II	even at $j=-\frac{1}{2}$, even at $j=N-\frac{1}{2}$	$\hat{U}_n = 2\sum_{j=0}^{N-1} U_j \cos\left(\frac{\pi(j+\frac{1}{2})n}{N}\right)$
DCT-III	even at $j=0$, odd at $j=N$	$\hat{U}_n = U_0 + 2\sum_{j=1}^{N-1} U_j \cos\left(\frac{\pi j(n+\frac{1}{2})}{N}\right)$
DCT-IV	even at $j=-\frac{1}{2}$, odd at $j=N-\frac{1}{2}$	$\hat{U}_n = 2\sum_{j=0}^{N-1} U_j \cos\left(\frac{\pi(j+\frac{1}{2})(n+\frac{1}{2})}{N}\right)$

Table 2.1: Discrete Cosine Transforms implemented in the FFTW library

Type	Boundary Conditions	Formula
DST-I	odd at $j=-1$, odd at $j=N$	$\hat{U}_n = 2\sum_{j=0}^{N-1} U_j \sin\left(\frac{\pi(j+1)(k+1)}{N+1}\right)$
DST-II	odd at $j=-\frac{1}{2}$, odd at $j=N-\frac{1}{2}$	$\hat{U}_n = 2\sum_{j=0}^{N-1} U_j \sin\left(\frac{\pi(j+\frac{1}{2})(k+1)}{N}\right)$
DST-III	odd at $j=-1$, even at $j=N-1$	$\hat{U}_n = (-1)^n U_{N-1}+2\sum_{j=0}^{N-2} U_n \sin\left(\frac{\pi(j+1)(n+\frac{1}{2})}{N}\right)$
DST-IV	odd at $j=-\frac{1}{2}$, even at $j=N-\frac{1}{2}$	$\hat{U}_n = 2\sum_{j=0}^{N-1} U_n \sin\left(\frac{\pi(j+\frac{1}{2})(n+\frac{1}{2})}{N}\right)$

Table 2.2: Discrete Sine Transforms implemented in the FFTW library

2.2.3 FFTW

The FFTW (Fastest Fourier Transform in the West [13]) library has several efficient implementations for DFT including real DCTs and DSTs [12] and so it is the library of choice in our algorithm. Tables 2.1 and 2.2 list the available DCT and DST implementations in the FFTW library for a one-dimensional array, $[U_0, \ldots, U_{N-1}]$. The idea behind the DCT and DST algorithms is to periodically extend the array and then perform a DFT that ignores any redundant operations due to the larger array [32, 33]. The Boundary Conditions column determines the parity of the extension and with respect to which point the array is to be extended. $j = 0$ or $j = N-1$ indicate that the array is to be extended with respect to an endpoint of the array. Extensions about $j = -1$ or $j = N$ occur with respect to a point that is one grid-scale outside of the physical array. Extensions with respect to $j = -\frac{1}{2}$ or $j = N - \frac{1}{2}$ occur similarly as the previous except that the point of extension occurs a half grid-scale outside of the array.

Type-I and Type-II transforms have the same parity on each boundary and so the array is copied once to achieve a periodic extension. Type-III and Type-IV transforms have opposite parity on the boundaries so the physical data needs to be copied four times over to produce a periodic extension. Type-I and Type-III transforms are performed on a regular grid where the physical boundary points are a full grid scale length away from the adjacent array points. However, for Type-II and Type-IV transforms, the Fourier basis is on a staggered grid where the physical grid points are shifted by half of a grid scale and the boundary point is a half grid-scale away from an adjacent point on the array. The choice of which DCT/DST type to use in our numerical method depends on the boundary conditions and Fourier basis we choose. Note that the boundary conditions listed on table 2.1 and 2.2 do not impose any restrictions on our physical data.

Forward	Inverse
DCT-I	DCT-I
DCT-II	DCT-III
DCT-III	DCT-II
DCT-IV	DCT-IV

Forward	Inverse
DST-I	DST-I
DST-II	DST-III
DST-III	DST-II
DST-IV	DST-IV

Table 2.3: Inverse DCT and DST

All of these transforms are invertible. The inverse of Type-I and Type-IV transforms are the transform themselves, the inverse of Type-II is Type-III and vice versa. This is summarized in table 2.3. However, the transforms computed by FFTW are unnormalized, so computing a transform followed by its inverse yields the original array scaled by the "logical" array size. For DCT-I the scaling factor is $2(N-1)$, for DST-I it is $2(N+1)$ and for all else it is $2N$. The transforms implemented in FFTW are most efficient when the "logical" array length is a product of small prime factors which is why Type-I transforms may not always be the best choice [13, 16]. In our algorithm, the required boundary conditions leads us to use either Type-I or Type-II transforms. Thus to use the more efficient implementation available, we use Type-II transforms and compute the equations on a staggered grid, $(x_{i+1/2}, y_{j+1/2})$. On the staggered grid, the boundary conditions of our physical problem are imposed implicitly.

Subroutines to compute the first and second derivatives using either sine or cosine transforms are shown in algorithms 2.1-2.4. The wavenumbers, $k_n = \frac{\pi n}{L}$, are defined on the "logical" array which is twice as long as the physical length, L. It is important to note that the FFTW transforms only stores relevant Fourier modes. Thus the real transform for an array of length N results in an array of length N but the spectral coefficients for the cosine transforms correspond to wavenumbers $\{k_n\}_0^{N-1}$ whereas for the sine transform, the spectral coefficients correspond to wavenumbers $\{k_n\}_1^N$. When computing first derivatives (or any odd-order derivative for that matter), the shift in spectral coefficients needs to be considered. In algorithm 2.1, the spectral coefficients, $\hat{U}_n$, correspond to an even transform but $\hat{U}'_n$ correspond to an odd transform. When computing $\hat{U}'_n$, the even transform $\hat{U}_n$ and wavenumbers k_n are shifted accordingly while the last coefficient, $\hat{U}'_{N-1}$ is set to zero. Algorithm 2.2 shows a similar shift in computing the first derivative using a sine transform. Note that there is a negative sign when scaling by the wavenumbers in algorithm 2.1 but not in algorithm 2.2. This is due to the fact that $\frac{d}{dx}\cos(x) = -\sin(x)$ and $\frac{d}{dx}\sin(x) = \cos(x)$. A negative sign arises when computing the derivative of a cosine and so it must be included when computing the derivative. Computing second-order derivatives (or any even-order derivatives) does not require any coefficient shifting since the parity of an even-numbered derivative is the same as the original function. The subroutines to compute second-order derivatives are shown in algorithms 2.3 and 2.4.

Algorithm 2.1 `function DCT_dx`: Compute first derivative with DCT

Input: $\{U_j\}$
Output: $\{U'_j\}$
  $\{\hat{U}_n\} \leftarrow$ DCT-II($\{U_j\}$)
  for $n = 0 \rightarrow N-2$ **do**
    $\hat{U}'_n \leftarrow -k_{n+1}\hat{U}_{n+1}$
  end for
  $\hat{U}'_{N-1} \leftarrow 0$
  $\{U'_j\} \leftarrow$ DST-III($\{\hat{U}'_n\}$)
  return $\{U'_j\}$

Algorithm 2.2 `function DST_dx`: Compute first derivative with DST

Input: $\{U_j\}$
Output: $\{U'_j\}$
  $\{\hat{U}_n\} \leftarrow$ DST-II($\{U_j\}$)
  for $n = 1 \rightarrow N-1$ **do**
    $\hat{U}'_n \leftarrow k_n\hat{U}_{n-1}$
  end for
  $\hat{U}'_0 \leftarrow 0$
  $\{U'_j\} \leftarrow$ DCT-III($\{\hat{U}'_n\}$)
  return $\{U'_j\}$

Algorithm 2.3 `function DCT_dx`^{2}: Compute second derivative with DCT

Input: $\{U_j\}$
Output: $\{U''_j\}$
  $\{\hat{U}_n\} \leftarrow$ DCT-II($\{U_j\}$)
  for $n = 0 \rightarrow N-1$ **do**
    $\hat{U}''_n \leftarrow -k_n^2\hat{U}_n$
  end for
  $\{U''_j\} \leftarrow$ DCT-III($\{\hat{U}''_n\}$)
  return $\{U''_j\}$

Algorithm 2.4 `function DST_dx`^{2}: Compute second derivative with DST

Input: $\{U_j\}$
Output: $\{U''_j\}$
  $\{\hat{U}_n\} \leftarrow$ DST-II($\{U_j\}$)
  for $n = 0 \rightarrow N-1$ **do**
    $\hat{U}''_n \leftarrow -k_{n+1}^2\hat{U}_n$
  end for
  $\{U''_j\} \leftarrow$ DST-III($\{\hat{U}''_n\}$)
  return $\{U''_j\}$

2.2.4 Time-Stepping Method

The time-stepping scheme of choice is the third-order Adams-Bashforth method (AB3),

$$U^{n+1} = U^n + \Delta t \left(\frac{23}{12} F^n - \frac{4}{3} F^{n-1} + \frac{5}{12} F^{n-2} \right), \tag{2.38}$$

where U^n is the numerical approximation of the dependent variable and F^n represents the function evaluation of the evolution equation at time $n\Delta t$ [31]. Linear multi-step methods such as AB3 are advantageous since only a single function evaluation is required at each new step, however, this comes at a cost of more memory. When compared with a single-step, multi-stage, fourth-order Runge-Kutta method (RK4) it is found that AB3 provides an overall faster runtime. Although RK4 has a larger stability regime than AB3 which allows for a larger time-step, it requires four function evaluations [21]. Numerical experiments have shown that simulations using AB3 is about twice as fast as RK4 overall.

To determine an appropriate time-step size for the algorithm, first consider Burger's equation

$$\frac{\partial U}{\partial t} + U \frac{\partial U}{\partial x} = \nu \frac{\partial^2 U}{\partial x^2}, \tag{2.39}$$

where ν is a diffusion coefficient. Burger's equation contains a nonlinear term that is similar to the nonlinear advection terms in (2.17) and (2.19). This equation contains two time scales that are important in the SW model. One is due to advection and another is due to diffusion. The nonlinear advection depends on the current state of U and the fastest advection (and thus the shortest time scale) occurs where U is maximal. With this in mind, now consider the more simple linear, advection-diffusion equation,

$$\frac{\partial U}{\partial t} + c \frac{\partial U}{\partial x} = \nu \frac{\partial^2 U}{\partial x^2}, \tag{2.40}$$

where c is some advection speed which can be $\max\{U\}$ or another speed that occurs in the SW (ie. surface gravity wave speed). This equation will be used to determine our time-step size. We apply the AB3 method and substitute in a single Fourier mode with wave number k and amplification factor α,

$$U^{n+j} = \alpha^j e^{ikx}, \tag{2.41}$$

to obtain the *stability polynomial*

$$\alpha^3 = \alpha^2 + \gamma \Delta t \left(\frac{23}{12} \alpha^2 - \frac{4}{3} \alpha - \frac{5}{12} \right), \tag{2.42}$$

where,

$$\gamma = -\nu k^2 - cki. \tag{2.43}$$

To ensure stability, we must choose Δt such that each solution of (2.42) satisfy $|\alpha| \leq 1$. The region within the solid curve in figure 2.1 shows values of $\gamma \Delta t$ on the complex plane

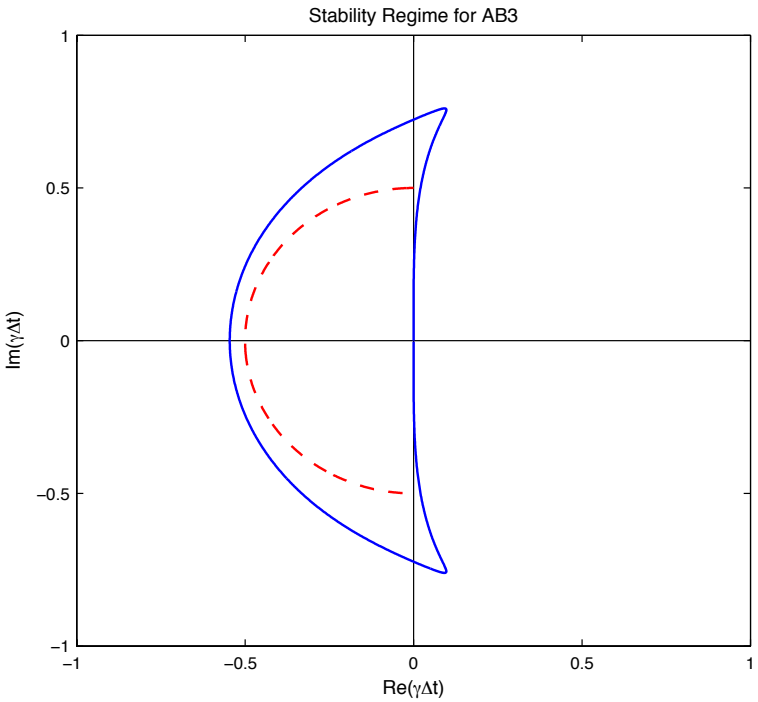


Figure 2.1: Values of $\gamma\Delta t$ on the complex plane such that the third-order Adams-Bashforth method is stable (solid contour). $|\gamma\Delta t| = 0.5$ (dashed curve)

for which AB3 is stable. To ensure $\gamma\Delta t$ lies within the region of stability, we choose Δt such that $|\gamma\Delta t| = 0.5$ or

$$\Delta t = \frac{1}{2|\gamma|} = \frac{1}{2\sqrt{(ck)^2 + (\nu k^2)^2}}. \tag{2.44}$$

This is indicated by the dashed curve in figure 2.1. Since $Re(\gamma) < 0$, $\gamma\Delta t$ will always lie within the stability regime for this choice of Δt. This analysis was done for a single Fourier mode with wavenumber k so for the algorithm to be stable for all modes, we minimize Δt using the largest wavenumber in the discretized domain. For the inviscid case, $\nu = 0$, we have $\Delta t = O(\Delta x)$ as $\Delta x \to 0$ and for the limiting case $c = 0$, which is just the diffusion equation, we have $\Delta t = O(\Delta x^2)$ as $\Delta x \to 0$. For the latter case, where the diffusion time scale is much shorter than the advective time scale, the time-step size can be rather small. But, using typical physical parameters for the SW model, the advective time scale due to the speed of the surface gravity waves dominate all other time scales in the model. Thus in our simulations, when computing the time-step with (2.44), our time-step size is $O(\Delta x)$. For a parameter set that where $O(\Delta x)$, such as a reduced gravity model, a semi-implicit scheme can be employed to allow larger Δt.

For our choice of Δt, it is necessary for explicit methods to satisfy the Courant-Friedrichs-Lewy (CFL) condition [5] for stability. The CFL condition requires that the time-stepping method can be convergent only if its numerical domain of dependence contains the physical domain of dependence of the PDE to be solved. To allow larger

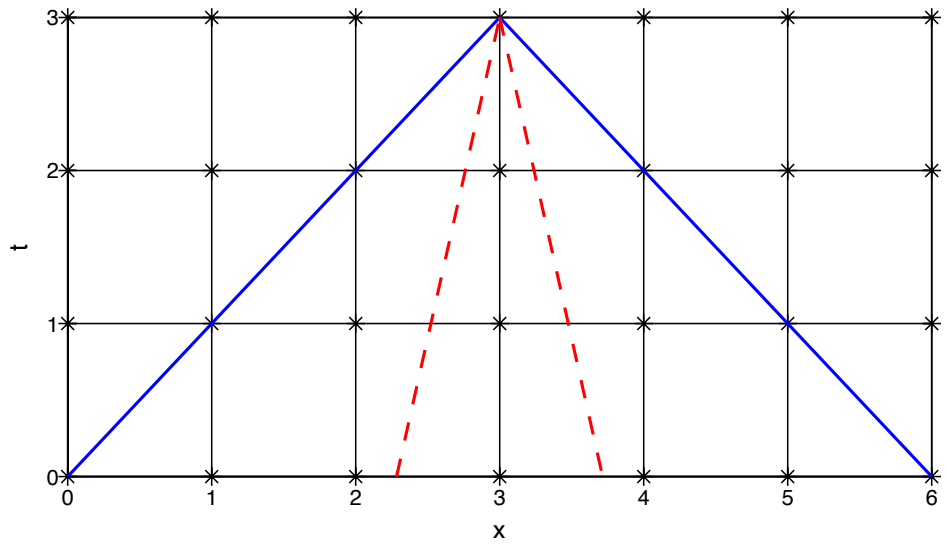


Figure 2.2: Computational grid of SW problem. Solutions are calculated on $*$. The numerical domain of dependence is shown by the solid lines with slope $\pm\Delta t/\Delta x$. The physical domain of dependence is shown by the dashed lines with slope $\pm 1/c$.

time-step sizes that does not necessarily satisfy the CFL condition, the alternative would be to use an implicit method. However, larger time-steps do not resolve the fast gravity waves in the SW model. Since we want include these motions, we use an explicit scheme with small Δt. Figure 2.2 shows the x-t plane to solve (2.40) and computing Δt with (2.44). The numerical domain of dependence is shown by the solid line and the physical domain of dependence is shown by the dashed line. It can be shown that the slopes of these lines satisfy

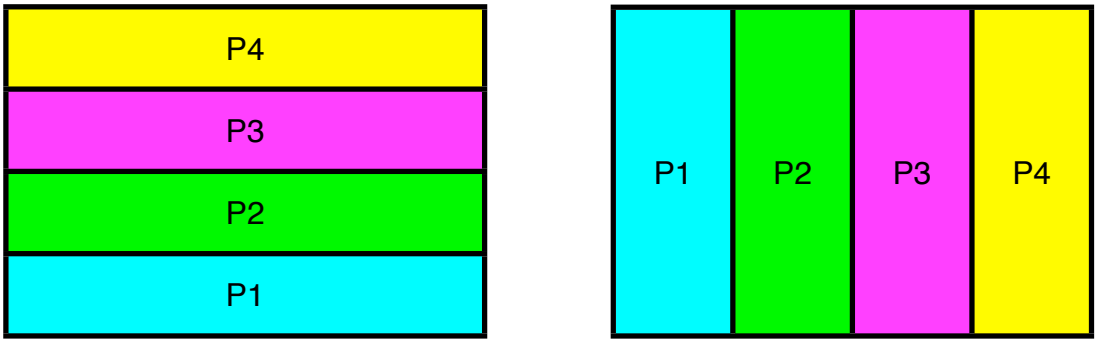
$$\frac{\Delta t}{\Delta x} < \frac{1}{c}, \tag{2.45}$$

implying that the numerical domain of dependence will always contain the physical domain of dependence and thus the CFL condition is always satisfied with this choice of the time-step size.

The AB3 scheme is a three-step method which requires another method to compute the first two steps. We avoid using low-order methods such as Forward Euler (which is first-order) or the second-order Adams-Bashforth method. Instead, we use RK4 to take the first two steps since it is fourth-order accurate. Although, RK4 requires more function evaluations, only two steps are taken so the longer runtime can be neglected.

2.2.5 Parallel Implementation

The choice of time-step described in section 2.2.4 can be rather small and will result in long runtimes. To improve the overall runtimes of the simulations, the program is written with the Message Passing Interface (MPI) communication protocol and run in parallel on a computer cluster. MPI allows the program to run different computations



(a) (b)

Figure 2.3: Schematic of the computational domain divided among four processors, P1 to P4, in the a) zonal direction, b) meridional direction

simultaneously on multiple processes and thus give an overall improvement in runtime. In this simulation, the computational domain is divided into n_p zonal strips, where n_p is the number of processes used in the simulation. Each process computes the SW equations in their respective zonal strips. Computing derivatives in the x-direction is trivial, but to compute derivatives in the y-direction is more intricate. Several `MPI_SendRecv` calls are required to send data from one process to another to re-divide the computational domain into meridional strips. Only then can the y-derivatives be computed. Figure 2.3 shows a schematic of the computational domain divided amongst four processors labelled P1 to P4 in both the zonal and meridional directions.

An important metric to determine the scalability of a parallel program is the *parallel efficiency*,

$$PE = \frac{T_{seq}}{n_p T(n_p)}, \tag{2.46}$$

where T_{seq} is the runtime of the sequential (not parallel) version of the program and $T(n_p)$ is the runtime of the parallel program on n_p processors. A well scaled code will have $PE \approx 1$ for any n_p. Table 2.4 lists the parallel efficiencies of a short test simulation on a 2048×2048 grid. When n_p is 2 or 4, we have $PE \approx 1$ but as n_p increases, the parallel efficiency decreases. This is due to the relatively fewer number of grid points on each processor. For this reason, 3D simulations have better scalability results running on higher number of processors than 2D simulations. Although, parallel efficiency is lower for larger n_p, a scalable version of the program has been effective in reducing the runtime significantly. Note that n_p must be chosen such that it divides both N_x and N_y.

2.2.6 Filtering

The nonlinear terms in the SW equations are responsible for energy being transferred between different length scales. This may cause problems in the simulation for there is a limit to the smallest range of motion that can be resolved on the computational

n_p	Parallel Efficiency
2	1.073
4	0.916
8	0.766
16	0.663
32	0.624
64	0.528
128	0.518

Table 2.4: Parallel efficiency of the SW solver of a test simulation on a 2048×2048 grid

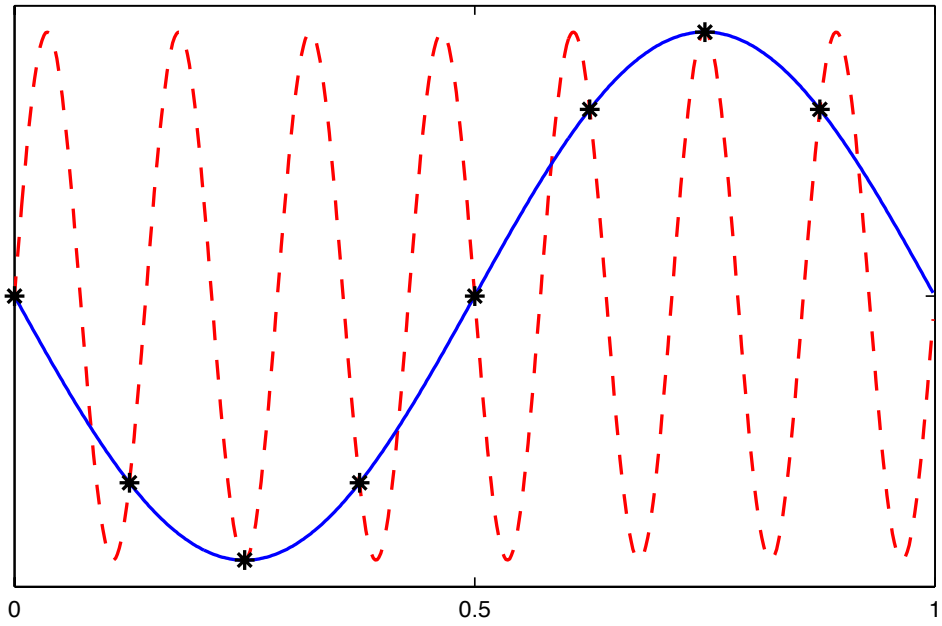


Figure 2.4: An example of aliasing on 8 grid points. The dashed sine wave has frequency 14π but is misinterpreted as a wave with frequency -2π.

grid. Although the lateral viscosity term aids in reducing the energy at small scales, it is insufficient for numerical stability for most cases. As the unresolved motions become important, the simulations may give results that are unphysical or lead to numerical instabilities. Another issue that arises due to the energy transfers between length scales is *aliasing* [38]. Since the range of wavenumbers on a computational grid is finite, the modes with frequencies that are too high for the grid are misinterpreted as a mode with lower frequency. Figure 2.4 shows an illustration of aliasing on 8 grid points. The dashed sine wave has frequency 14π which can not be properly resolved on an 8-point grid. Instead, it is interpreted as a sine wave with frequency -2π.

To work around these issues, we filter or *dealias* the high frequency modes. The filter of choice is a truncation method using the "two-thirds rule" [2]. In two-dimensions, the wavenumber space is truncated to an ellipse where the semi-major and semi-minor axes are two-thirds of the highest wavenumber in each direction. The filter follows the

formula

$$\hat{U}_{mn} = \begin{cases} 0 & \text{if } \left(\frac{k_m}{\frac{2}{3}k_{max}}\right)^2 + \left(\frac{l_n}{\frac{2}{3}l_{max}}\right)^2 \geq 1, \\ \hat{U}_{mn} & \text{otherwise.} \end{cases} \tag{2.47}$$

where k_{max} and l_{max} are the largest positive wavenumbers in the computational domain in the zonal and meridional directions, respectively. The filter is applied to each field, u, v, and h, after each time-step.

Truncation effectively limits the range of wavelengths that are resolved in the numerical model but the difference rather minor. Without truncation, the smallest wavelength that can be resolved is $\lambda_{min} = 2\Delta x$ but with truncation, wavelengths as small as $4\Delta x$ are resolved.

2.2.7 Pseudocode

The final algorithm is described in pseudocode in algorithm 2.5. To ensure stability as the velocity fields evolve, Δt_{new} is calculated using (2.44) with c as the max speed of the current state. If $\Delta t < \Delta t_{new}$, then a new time-step is chosen based on Δt_{new} and the simulation proceeds. In the code, the discrete Fourier transforms are the most computationally expensive subroutines so in practice we minimize the number of transforms required, which is not shown in the pseudocode. When computing derivatives, the forward transforms are taken once to compute both first-order and second-order derivatives. Algorithm 2.6 describes the filtering procedure.

2.3 Numerical Method for QG Model

2.3.1 Physical Model

The barotropic QG model is the predominant model used to study large-scale ocean circulation theory. It describes a two-dimensional flow of a one-layer, homogeneous, incompressible fluid in the rigid lid limit [24]. The QG model can be derived directly from the SW model by first introducing characteristic scales L, U, T and N_0 to describe typical magnitudes of length, velocity, time and free-surface elevation. Then define non-dimensional variables, denoted with primes, by

$$(x, y) = L(x', y'), \tag{2.48}$$
$$(u, v) = U(u', v'), \tag{2.49}$$
$$t = Tt', \tag{2.50}$$
$$\eta = N_0\eta'. \tag{2.51}$$

Algorithm 2.5 Numerical method to solve Rotating Shallow Water equations

Compute Δt using (2.44) with $c = \sqrt{gH}$
$\{u_{ij}\} \leftarrow u_0, \{v_{ij}\} \leftarrow v_0, \{h_{ij}\} \leftarrow h_0$
$t \leftarrow 0$
$FirstStep \leftarrow$ **true**, $SecondStep \leftarrow$ **true**
while $t \leq t_{end}$ **do**
  Compute terms in (2.17) and (2.19)
  if $FirstStep =$ **true then**
    RK4 time-step
    $FirstStep \leftarrow$ **false**
  else if $SecondStep =$ **true then**
    RK4 time-step
    $SecondStep \leftarrow$ **false**
  else
    AB3 time-step
  end if
  $t \leftarrow t + \Delta t$
  Filter($\{u_{ij}\}, \{v_{ij}\}, \{h_{ij}\}$)
  Compute Δt_{new} using (2.44) with $c = \max |(u, v)|$
  if $\Delta t_{new} < \Delta t$ **then**
    $\Delta t \leftarrow \frac{3}{4}\Delta t_{new}$
    $FirstStep \leftarrow$ **true**, $SecondStep \leftarrow$ **true**
  end if
end while

Algorithm 2.6 Filtering procedure for SW solver

Input: $\{U_{ij}\}$
Output: $\{U_{ij}\}$
compute appropriate forward Fourier transforms
$\{\hat{U}_{mj}\} \leftarrow DFT_x(\{U_{ij}\})$
$\{\tilde{U}_{mn}\} \leftarrow DFT_y(\{\hat{U}_{mj}\})$
Truncate high-frequency wave modes
for $m = 0, \ldots, N_x - 1$ ***do***
  for $n = 0, \ldots, N_y - 1$ ***do***
    if $\left(\frac{k_m}{2/3 k_{max}}\right)^2 + \left(\frac{l_n}{2/3 l_{max}}\right)^2 \geq 1$ ***then***
      $\tilde{U}_{mn} \leftarrow 0$
    end if
  end for
end for
Transforms to physical space using appropriate inverse transform
$\{\hat{U}_{mj}\} \leftarrow DFT_y(\{\tilde{U}_{mn}\})$
$\{U_{ij}\} \leftarrow DFT_x(\{\hat{U}_{mj}\})$
return $\{U_{ij}\}$

To ensure that u' and v' are non-zero, we choose the free-surface scaling, N_0, such that the pressure gradient is large enough to balance the Coriolis acceleration,

$$N_0 = \frac{ULf_0}{gH} \tag{2.52}$$

Substituting (2.48)-(2.51) into the SW model introduces nondimensional parameters,

$$\epsilon = \frac{U}{f_0 L}, \tag{2.53}$$

$$\epsilon_T = \frac{1}{f_0 T}, \tag{2.54}$$

called the *Rossby number* and the *temporal Rossby number*, respectively. If we enforce that both ϵ and ϵ_T are equal and small, implying that the advective time scale, U/L, is small compared to the Coriolis time scale, $1/f_0$, then the dependent variables can be written as a perturbation expansion about ϵ,

$$u' = u_0 + \epsilon u_1 + \epsilon^2 u_2 + \cdots, \tag{2.55}$$

$$v' = v_0 + \epsilon v_1 + \epsilon^2 v_2 + \cdots, \tag{2.56}$$

$$\eta' = \eta_0 + \epsilon \eta_1 + \epsilon^2 \eta_2 + \cdots. \tag{2.57}$$

The $O(1)$ terms gives,

$$u_0 = -\frac{\partial \eta_0}{\partial y}, \tag{2.58}$$

$$v_0 = \frac{\partial \eta_0}{\partial x}, \tag{2.59}$$

and

$$\frac{\partial u_0}{\partial x} + \frac{\partial v_0}{\partial y} = 0. \tag{2.60}$$

This means that the $O(1)$ terms are in geostrophic balance and are incompressible in two-dimensions. The $O(\epsilon)$ equations can be combined to arrive at an $O(1)$ vorticity equation where

$$\zeta_0 = \frac{\partial v_0}{\partial x} - \frac{\partial u_0}{\partial y}. \tag{2.61}$$

Upon dropping the subscripts and replacing the order-one free-surface, η_0, with a stream-function, ψ, the QG equations in dimensional form are

$$\frac{\partial \zeta}{\partial t} + \frac{\partial \psi}{\partial x}\frac{\partial \zeta}{\partial y} - \frac{\partial \psi}{\partial y}\frac{\partial \zeta}{\partial x} + \beta \frac{\partial \psi}{\partial x} = \nu \nabla^2 \zeta + \hat{k} \cdot \vec{\nabla} \times \vec{\tau}, \tag{2.62}$$

$$\zeta = \nabla^2 \psi. \tag{2.63}$$

The second and third terms on the left-hand side is the nonlinear advection written in terms of the stream-function. The last term on the left-hand is the ambient vorticity

which incorporates the Earth's rotation into the model. In this model, a simple Laplacian term with the kinematic viscosity coefficient, ν, is used for lateral viscosity. The dissipation term (2.22) reduces to a Laplacian term in the QG limit. This form models the sub-grid motions that are not resolved in our numerical model [25]. The wind forcing used is (2.10). The QG model describe $O(1)$ motions in the perturbation expansion and thus the error in the model is $O(\epsilon)$. This implies that the QG model is valid when the advective time scale is small compared to the Coriolis time scale. As faster dynamics occur at smaller length scales, then QG is no longer a reasonable approximation. For a more detailed discussion on the QG model, the reader is directed to [24] or [25].

The boundary conditions (2.7), (2.8) and (2.9) in terms of the stream-function are

$$\psi = 0 \text{ on } \Gamma, \tag{2.64}$$

for no normal flow,

$$\frac{\partial \psi}{\partial x} = 0 \text{ on } \Gamma_E \cup \Gamma_W, \tag{2.65}$$

for no slip, and

$$\frac{\partial^2 \psi}{\partial y^2} = 0 \text{ on } \Gamma_N \cup \Gamma_S, \tag{2.66}$$

for free slip on the north and south boundaries.

2.3.2 Spatial Discretization

At first, one may try to use a Fourier basis in the spectral method as in section 2.2 but boundary conditions (2.64) and (2.65) make it problematic to use a Fourier decomposition in the x-direction since a zero-dirichlet condition and a zero-neumann condition needs to be satisfied. This inconsistent with the boundary conditions of a sine or cosine spectral decomposition. So instead, we use a Fourier basis in the y-direction only where both boundary conditions to the north and south can be satisfied using sine modes. In the x-direction, we use a Chebyshev polynomial basis which will allow us to impose both (2.64) and (2.65). Thus our stream-function will be of the form,

$$\psi(x, y) = \sum_{m,n} \hat{a}_{mn} T_m(x) \sin(l_n y), \tag{2.67}$$

where $T_m(x)$ are Chebyshev polynomials. The algorithm developed in this section is similar to the numerical method used in Chapter 4 of [35] where simulation of the QG equations were used to study WBC separations.

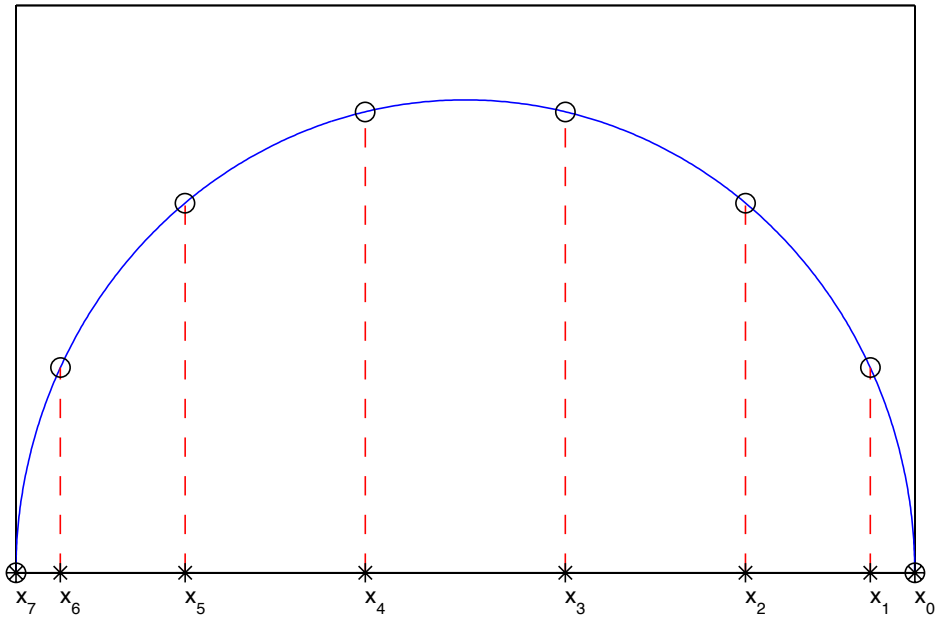


Figure 2.5: A Chebyshev grid with $N = 7$ is formed by projecting points that are equidistant on a circle down onto the x-axis. $\{x_i\}$ is defined in descending order with $x_0 = 1$ and $x_N = -1$

2.3.3 Chebyshev Polynomial Interpolation

In this section, we discuss Chebyshev polynomials on the interval $[-1, 1]$. The ideas discussed can easily be extended to the interval $\left[-\frac{L_x}{2}, \frac{L_x}{2}\right]$ by a change of variable. We first define a non-uniform grid on $[-1, 1]$ by

$$x_i = \cos\left(\frac{\pi i}{N}\right), \tag{2.68}$$

for $i = 0, \ldots, N$, called *Chebyshev collocation points*. Note that this defines the grid in descending order but does not affect the derivation of the numerical method. The grid points are calculated on equidistant points of a unit semi-circle then projected down to the x-axis. Figure 2.5 illustrates a Chebyshev grid for $N = 7$. One important feature of (2.68) is the crowding of grid points near the boundaries. A polynomial interpolation using a uniform grid may result in highly oscillatory behaviour near the boundaries. Furthermore, when interpolating a smooth function on a uniform grid, the convergence rate becomes worse as $N \to \infty$. This is called the *Runge phenomenon*. Having a higher concentration of grid points near the boundaries eliminates the oscillations and polynomial interpolations converge [38]. In practice, we use $N = N_x - 1$ to be consistent with the previous section so we have N_x to mean the exact number of grid points. For ease of notation, the Chebyshev interpolation will continue to be discussed is this section in terms of N.

The Chebyshev polynomials are defined in [10] as

$$T_m(x) = \cos(m\theta), \quad \theta = \arccos(x). \tag{2.69}$$

The first case, $m = 0$, gives

$$T_0(x) = 1, \tag{2.70}$$

and $m = 1$ gives

$$T_1(x) = x. \tag{2.71}$$

Using the trigonometric identity, $\cos(\theta)\cos(m\theta) = \frac{1}{2}[\cos((m+1)\theta) + \cos((m-1)\theta)$, the remaining Chebyshev polynomials can be found recursively by

$$T_m(x) = 2xT_{m-1}(x) - T_{m-2}(x), \tag{2.72}$$

for $m = 3, \ldots, N$. Although we assume that the stream-function can be decomposed in terms of $T_m(x)$ in the zonal direction, the spectral coefficients are not computed explicitly. Thus we do not use a transform method to compute the derivatives as in section 2.2. Instead, we compute the derivatives in physical space using a Chebyshev differentiation matrix, D_x, defined as

$$[D_x]_{00} = \frac{2N^2+1}{6}, \quad [D_x]_{NN} = -\frac{2N^2+1}{6} \tag{2.73}$$

$$[D_x]_{jj} = \frac{-x_j}{2(1-x_j^2)}, \qquad j = 1, \ldots, N-1 \tag{2.74}$$

$$[D_x]_{ij} = \frac{c_i}{c_j}\frac{(-1)^{i+j}}{(x_i - x_j)}, \qquad i, j = 0, \ldots, N, i \neq j \tag{2.75}$$

where

$$c_i = \begin{cases} 2, & \text{if } i = 0 \text{ or } N, \\ 1, & \text{otherwise.} \end{cases} \tag{2.76}$$

D_x is a dense matrix since Chebyshev polynomials are globally interpolating basis functions as opposed to other finite differencing schemes which interpolate locally resulting in sparse differentiation matrices. The use of a differentiation matrix will be costly in terms of computational resources but it is necessary to solve the inverse problem that will be discussed in section 2.3.5.

2.3.4 Time-Stepping Method

Consider a linear system of ODEs,

$$\vec{U}'(t) = A\vec{U}, \tag{2.77}$$

where A is a constant matrix operator. When solving the above system with an explicit time-stepping scheme, it is shown in [21] that we require that $\Delta t|\lambda|$ lies within the stability region of the time-stepping scheme on the complex plane for each eigenvalue, λ, of A. This implies that

$$\Delta t = O(\lambda_{max}^{-1}), \tag{2.78}$$

where λ_{max} is the eigenvalue of A with the largest magnitude. When calculating the terms in (2.62), we make use of the first and second-order Chebyshev differentiation matrix,

D_x and D_x^2. It is shown in [2] that the modulus of the largest eigenvalue of D_x is $O(N^2)$, and D_x^2 is $O(N^4)$. Thus the advective terms in (2.62) requires $\Delta t = O(N^{-2})$ and the dissipation term requires $\Delta t = O(N^{-4})$, as $N \to \infty$, if solving with an explicit scheme. The latter is a severe restriction on Δt and not a suitable choice in our algorithm. To allow larger Δt, we solve the dissipation in (2.62) implicitly. The nonlinear advection terms are to be computed explicitly as an implicit method would require a nonlinear, iterative solver [31]. Computing the advection terms in (2.62) with an explicit method, it is necessary to have $\Delta t = O(N^{-2})$ due to the CFL condition. Near the boundaries, the grid points are clustered with distribution $O(N^{-2})$ and as a result, the CFL condition requires $\Delta t = O(N^{-2})$.

The time-stepping scheme used is a combination of the explicit AB3 method and the implicit Backward Euler method (BE). The implicit/explicit time-stepping scheme for the vorticity is

$$\zeta^{n+1} = \zeta^n + \Delta t \left(\frac{23}{12} F_{QG}^n - \frac{4}{3} F_{QG}^{n-1} + \frac{5}{12} F_{QG}^{n-2} \right) + \Delta t \nu \nabla^2 \zeta^{n+1} \tag{2.79}$$

where

$$F_{QG}^n = -\frac{\partial \psi^n}{\partial x} \frac{\partial \zeta^n}{\partial y} + \frac{\partial \psi^n}{\partial y} \frac{\partial \zeta^n}{\partial x} - \beta \frac{\partial \psi^n}{\partial x} + \hat{k} \cdot \vec{\nabla} \times \vec{\tau}. \tag{2.80}$$

At each time-step, the explicit terms are solved first then the implicit terms are computed. The method to solve the implicit terms as well as the inverse problem (2.63) is described in section 2.3.5. AB3 is chosen in favor of a multi-stage method such as RK4 since it requires only a single function evaluation at each step. A multi-stage method is problematic since there is no evolution equation for the stream-function, ψ. The inverse problem would be solved after each stage to find ψ before computing the next stage. The inverse problem is computationally expensive so this is not a viable choice. We must take steps that require only a single evaluation of F_{QG} before solving the inverse problem.

We require two initial time-steps before executing the AB3 scheme but we must be careful on what method to use. Our first step is taken with the Forward Euler method (FE)

$$\zeta^{n+1} = \zeta^n + \Delta t F_{QG}^n, \tag{2.81}$$

then we take a step with the 2nd-order Adams-Bashforth method (AB2),

$$\zeta^{n+1} = \zeta^n + \Delta t \left(\frac{3}{2} F_{QG}^n - \frac{1}{2} F_{QG}^{n-1} \right), \tag{2.82}$$

using the function evaluation from the FE step. We then proceed with using AB3 for the remainder of the simulation. Although lower-order methods are used as the start-up procedure, it is presumed that does not affect the overall accuracy of the simulation as the start-up procedure only occurs a few times during the computation.

The step size, Δt, is chosen such that

$$\Delta t < \frac{0.5}{\max\limits_{i,j}\{|\frac{u_{ij}}{\Delta x_i}\}| + \max\limits_{i,j}\{\frac{v_{ij}}{\Delta y}\}}, \tag{2.83}$$

where $\Delta x_i = x_i - x_{i-1}$, to ensure that the CFL condition is satisfied in both x and y-directions. The factor of 0.5 was found empirically. It is difficult to determine if this choice of Δt lies within the stability regime of AB3 due to the nonlinearity of the problem and the different discretizations in each direction of the computational domain. However, simulations of the QG equations with this choice of Δt has yet to experience a numerical instability.

2.3.5 Inverse Problem

What remains to be solved are the implicit terms in (2.79) and the inverse problem (2.63). These two equations can be written in matrix form as

$$\underbrace{\begin{bmatrix} 0 & I - \Delta t\nu\nabla^2 \\ \nabla^2 & I \end{bmatrix}}_{A} \begin{bmatrix} \psi^{n+1} \\ \zeta^{n+1} \end{bmatrix} = \underbrace{\begin{bmatrix} \zeta^n + \Delta t \left(\frac{23}{12}F_{QG}^n - \frac{4}{3}F_{QG}^{n-1} + \frac{5}{12}F_{QG}^{n-2}\right) \\ 0 \end{bmatrix}}_{B}, \tag{2.84}$$

where I denotes an identity operator. Equation (2.84) will become a system of linear equations upon discretizing ψ^{n+1} and ζ^{n+1}. Solving linear systems can be computationally expensive so we take caution when discretizing the variables. Consider discretizing ψ^{n+1} and ζ^{n+1} as $(N_xN_y)\times 1$ vectors and A would be discretized as a $(2N_xN_y)\times(2N_xN_y)$ matrix. Note that when discussing matrices, we use the *row* $\times$ *column* coordinates and not the cartesian grid coordinates and thus differentiation matrix operators will be applied to the left of the field to compute x-derivatives. The computational complexity of solving this linear system is $O((N_xN_y)^a)$ for some $a > 2$. Direct methods for solving linear systems gives $a = 3$ but fast algorithms exist where the current lowest known value is $a = 2.376$ [4, 39]. This can be very expensive for high-resolution simulations. In an attempt to improve the computational complexity, we take advantage of the Fourier basis in the y-direction and use FFTs. We discretize ψ^{n+1} and ζ^{n+1} as $N_x \times N_y$ matrices, then we take the Fourier transform of (2.84) in the y-direction to get

$$\underbrace{\begin{bmatrix} 0_{N_x} & I_{N_x} - \Delta t\nu\tilde{\nabla}^2 \\ \tilde{\nabla}^2 & I_{N_x} \end{bmatrix}}_{\tilde{A}_l} \begin{bmatrix} \tilde{\psi}_l^{n+1} \\ \tilde{\zeta}_l^{n+1} \end{bmatrix} = \tilde{B}_l, \tag{2.85}$$

where 0_{N_x} and I_{N_x} are $N_x \times N_x$ zero and identity matrices, respectively. The Laplacian operator becomes $\tilde{\nabla}^2 = D_x^2 - l^2 I_{N_x}$ where l denotes meridional wave numbers and tildas indicate DSTs in the y-direction. $\tilde{A}_l$ is now a $2N_x \times 2N_x$ matrix. There are N_y linear systems to be solved for each l in the domain and two more DSTs to be executed (one forward, one backward). Therefore, the computational complexity of the new discretization, is $O(N_x^a N_y + N_x N_y \log(N_y))$. Since $a > 2$, this is an improvement from what we had before but it is still expensive for high-resolution simulations.

Before computing the inverse, $\tilde{A}_l$ and $\tilde{B}_l$ must be modified such that the resulting ψ^{n+1} satisfies the no normal flow and no slip conditions on the east and west boundaries.

The first and last row of each submatrix of $\tilde{A}_l$ in (2.85), along with the corresponding rows in $\tilde{B}$, are replaced in the following manner,

$$\begin{bmatrix} [I_{N_x}]_{1,1:N_x} & 0 & \cdots & 0 \\ \vdots & & & \\ [I_{N_x}]_{N_x,1:N_x} & 0 & \cdots & 0 \\ [D_x]_{1,1:N_x} & 0 & \cdots & 0 \\ \vdots & & & \\ [D_x]_{N_x,1:N_x} & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} \tilde{\psi}^{n+1} \\ \tilde{\zeta}^{n+1} \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \tag{2.86}$$

Recall that the boundary conditions on the north and south boundaries are already satisfied by the use of the sine spectral basis.

The solutions to the linear systems are computed using the linear solvers available in the library LAPACK (Linear Algebra PACKage [1]). In an attempt to use the most efficient implementation available in LAPACK, the matrix $\tilde{A}$ and $\tilde{B}$ can be rearranged by swapping the top half with the bottom half and so $\tilde{A}$ becomes a banded matrix with upper bandwidth $b_U = N_x$ and lower bandwidth $b_L = N_x$. LAPACK has a linear solver for general banded systems, `dgbsv`, but numerical tests have shown that this method has a longer runtime than using a general solver, `dgesv`. This is due to the relatively large upper and lower bandwidths. The banded solver reshapes the matrix to a $(2b_L + b_U + 1) \times 2N_x$ matrix which is larger than the original matrix $\tilde{A}$.

2.3.6 Parallel Implementation

The computational cost of solving the inverse problem dominates the total runtime of the simulation. As with the SW solver, the program is written using MPI to run the simulation on n_p processors. First, consider the explicit time-stepping. The computational domain is divided into zonal strips to solve the equations. To compute the zonal derivatives, the differentiation matrix is multiplied on to the left of the discretized variables. To compute meridional derivatives, the domain is divided into meridional strips as in the previous section for the SW solver. For the inverse problem, we have N_y linear systems to solve that are independent of each other thus the linear systems are divided evenly among the processors and solved simultaneously.

Table 2.5 lists the parallel efficiency of a test simulation on a 512×512 grid. The values show a much better scalability result than the SW solver. This is due to the even distribution of the computationally expensive inverse problem amongst the processors. But in practice, we choose $N_x < N_y$ for high-resolution cases to reduce the overall runtime of the simulation. This effectively decreases the relative size of the inverse problem. Since a Chebyshev collocation is used in the zonal direction, a smaller N_x does not effect the resolution of the motions as much of the dynamics occur near the WBC and the interior is relatively slow. Table 2.6 shows the parallel efficiency of 256×2048 simulation. With the relatively smaller N_x, some parallel efficiency is lost but still shows better scalability for large n_p than the SW solver.

n_p	Parallel Efficiency
2	0.991
4	1.032
8	1.002
16	0.998
32	0.944
64	0.979
128	0.903

Table 2.5: Parallel efficiency of the QG solver of a test simulation on a 512×512 grid

n_p	Parallel Efficiency
2	1.014
4	0.971
8	0.909
16	0.902
32	0.864
64	0.867
128	0.748

Table 2.6: Parallel efficiency of the QG solver of a test simulation on a 256×2048 grid

2.3.7 Filtering

The nonlinear advection terms in (2.62) transfers energy to different length scales. To avoid numerical instability due to under-resolved motions, a spectral filtering procedure is required. The spectral method uses a Fourier basis the y-direction and a Chebyshev polynomial basis in the x, each requiring a different filter for numerical stability. The sine modes in the y-direction are filtered using the two-thirds truncation rule,

$$\tilde{U}_n = \begin{cases} 0 & \text{if } l_n \geq \frac{2}{3} l_{max}, \\ \tilde{U}_n & \text{otherwise.} \end{cases} \tag{2.87}$$

In the x-direction, we must filter the high-degree Chebyshev polynomials to avoid unresolved, sub-grid dynamics. A truncation filter was found to cause numerical instabilities in the code so instead we use a damping approach. Consider a fourth-order, hyperviscosity equation in one dimension,

$$\frac{\partial U}{\partial t} = \mu \frac{\partial^4 U}{\partial x^4}. \tag{2.88}$$

If we assume a wave-like solution of the form $U(x,t) = \exp(i(kx - \omega t))$, we get a dispersion relation

$$\omega = \mu k^4 i, \tag{2.89}$$

and the wave-like solution becomes

$$U(x,t) = e^{\mu k^4 t} e^{ikx}. \tag{2.90}$$

Here, the solution decays if $\mu < 0$ and different wave modes decay at different rates. Higher values of k dampens the solution very strongly whereas lower values experience a weaker damping. We take a similar approach to our Chebyshev damping filter but we modify μ and k such that only high-degree Chebyshev polynomials are dampened and lower-degree polynomials remain unchanged.

Before proceeding, we need a method to obtain the spectral coefficients for the Chebyshev decomposition. Recall from equation (2.69) that the Chebyshev polynomials can be written in terms of cosine functions. In series form, the Chebyshev decomposition can be written as

$$\sum_m \hat{U}_m \cos(m\theta), \tag{2.91}$$

which is a Fourier cosine series in θ. Thus filtering a Fourier series in θ is equivalent to filtering a Chebyshev series in x and the spectral coefficients can be found by taking an even transform using DCT-I. Upon computing the coefficients, the filter follows the formula

$$\hat{U}_m = \begin{cases} e^{\mu\left(\frac{m-m_c}{N-m_c}\right)^4}\hat{U}_m & \text{if } m > m_c, \\ \hat{U}_m & \text{otherwise.} \end{cases} \tag{2.92}$$

Polynomials with degree greater than the cutoff value m_c are dampened in a manner that is similar to the hyper-viscosity equation. This "exponential cutoff filter" is commonly used when studying physical problems with spectral methods [8, 15]. The value of the damping coefficient is chosen such that $\mu \approx \ln(\varepsilon)$ so the highest frequency is driven down to machine precision, ε. In our simulations, we choose $m_c = \frac{2}{3}N_x$ to correspond with the "two-third" truncation filter applied in the meridional direction.

Both filters are applied to matrix B in (2.84) after computing the explicit terms but before solving the inverse problem. This ensures that any aliasing effects are removed before we compute ψ and q.

2.3.8 Pseudocode

Pseudocode is presented to summarize the numerical procedure to solve the QG equations. Algorithm 2.7 provides an overview of the QG solver. To ensure that the CFL condition is satisfied, Δt_{new} is computed prior to each time-step and compared with the current Δt. If the CFL condition is violated, then a new Δt is computed and the AB3 time-stepping restarts by taking a FE step and a AB2 step with the new Δt. The function evaluation of F_{QG} is shown in detail in algorithm 2.8. The pseudocode for the filtering procedure is shown in algorithm 2.9 and for the inverse problem is shown in algorithm 2.10.

Algorithm 2.7 Numerical method to solve Quasi-Geostrophy equations

$\Delta t \leftarrow \Delta t_{init}$
$\{\psi_{ij}\} \leftarrow 0, \{\zeta_{ij}\} \leftarrow 0$
$t \leftarrow 0$
$FirstStep \leftarrow$ **true**, $SecondStep \leftarrow$ **true**
while $t \leq t_{end}$ **do**
  Compute F_{QG}
  if $FirstStep =$ **true then**
    $B \leftarrow$ FE time-step
    $FirstStep \leftarrow$ **false**
  else if $SecondStep =$ **true then**
    $B \leftarrow$ AB2 time-step
    $SecondStep \leftarrow$ **false**
  else
    $B \leftarrow$ AB3 time-step
  end if
  $t \leftarrow t + \Delta t$
  Filter(B)
  Solve inverse problem
  Compute Δt_{new} using (2.83)
  if $\Delta t_{new} < \Delta t$ **then**
    $\Delta t \leftarrow \frac{3}{4}\Delta t_{new}$
    $FirstStep \leftarrow$ **true**, $SecondStep \leftarrow$ **true**
  end if
end while

Algorithm 2.8 `function` F_{QG}

Input: $\{\psi_{ij}\}, \{\zeta_{ij}\}$
Output: F_{QG}
  Compute y-derivatives using algorithm 2.2
  Compute x-derivatives using differentiation matrix D_x
  Compute F_{QG} following (2.62)
  return F_{QG}

Algorithm 2.9 Filtering procedure for QG solver

Input: U_{ij}
Output: U_{ij}
compute odd Fourier transforms in y
$\tilde{U}_{in} \leftarrow DST_y\text{-}II(U_{ij})$
truncate high-frequency wave modes
for all $n = 0, \ldots, N_y - 1$ ***do***
if $l_n \geq 2/3 l_{max}$ ***then***
$\tilde{U}_{in} \leftarrow 0$
end if
end for
transform to physical space
$U_{ij} \leftarrow DST_y\text{-}III(\tilde{U}_{in})$
compute even Fourier transforms in x
$\hat{U}_{mj} \leftarrow DCT_x\text{-}I(U_{ij})$
dampen high-degree polynomials
for all $m = 0, \ldots, N_x - 1$ ***do***
if $m \geq m_c$ ***then***
$\hat{U}_{in} \leftarrow 0$
end if
end for
transform to physical space
$U_{ij} \leftarrow DCT_x\text{-}I(\hat{U}_{in})$
return U_{ij}

Algorithm 2.10 QG Inverse Problem

Input: B
Output: ψ_{ij}, ζ_{ij}
$\tilde{B} \leftarrow DST_y(B)$
$\tilde{A} \leftarrow \begin{bmatrix} 0 & I - \Delta t \nu (D_x^2 - l_n^2 I) \\ D_x^2 - l_n^2 I & I \end{bmatrix}$
modify $\tilde{A}$ and $\tilde{B}$ to impose zonal boundary conditions
solve $\tilde{A} \begin{bmatrix} \tilde{\psi}_{ij} \\ \tilde{\zeta}_{ij} \end{bmatrix} = \tilde{B}$
$\psi_{ij} \leftarrow IDST_y(\tilde{\psi}_{ij})$
$\zeta_{ij} \leftarrow IDST_y(\tilde{\zeta}_{ij})$
return ψ_{ij}, ζ_{ij}

Chapter 3

Numerical Results

In this chapter, numerical simulations of the Rotating Shallow Water equations and the Quasi-Geostrophic equations using the methods discussed in chapter 2 are presented. The parameters used in all simulations are discussed in section 3.1. Results from the SW solver are presented in section 3.2. In section 3.3 we compare some results between the SW simulations and QG simulations.

3.1 Computational Parameters

The simulations take place on a square ocean basin with side lengths of 4000 km. The choice of Coriolis parameters, f_0 and β, are appropriate for a mid-latitude ocean centered at 45 degrees north. The mean fluid depth, H, is set to be 500 m where it is estimated by [29] that the effects of the atmospheric winds reaches down to this depth. Since this is a one-layer model, we assume the ocean beneath this depth is static. In the simulations, a wind stress parameter of $\tau_0 = 0.2\,\mathrm{N/m^2}$ is used which is higher than the typically observed value of $0.1\,\mathrm{N/m^2}$. The higher value inputs more energy into the system and allows us to observe gyre flow at an earlier time however realistic velocity fields are observed. All of the physical parameters are summarized in table 3.1.

L_x	4×10^3 km
L_y	4×10^3 km
g	9.81 m/s^2
H	5×10^2 m
f_0	1×10^{-4} s^{-1}
β	2×10^{-11} (ms)$^{-1}$
τ_0	0.2 N/m^2
ρ_0	1000 kg/m^3

Table 3.1: Physical parameters used in numerical simulations.

Case	δ_M (km)	ν (m^2/s)	$N_x \times N_y$ (SW)	$N_x \times N_y$ (QG)	n_p
Case 1	200	1.6×10^5	128×128	128×128	8
Case 2	100	2.0×10^4	256×256	128×256	16
Case 3	50	2.5×10^3	512×512	128×512	32
Case 4	25	312.5	1024×1024	256×1024	64
Case 5	12.5	39.0625	2048×2048	512×2048	128

Table 3.2: Simulation cases

To test the range of length scales that the simulations can resolve, we vary the kinematic viscosity coefficient, ν, as it parameterizes the width of the WBC [25]. By balancing the ambient vorticity with the dissipation term in the vorticity equation of either model, a relationship between ν and the boundary layer width, δ_M, is found by

$$\delta_M = \left(\frac{\nu}{\beta}\right)^{\frac{1}{3}}. \tag{3.1}$$

This particular boundary layer is called the *Munk boundary layer* [23]. It is important to note that the width of the WBC depends directly on the strength of the dissipation term. The vorticity dissipation in Stommel's model is a bottom drag term that simulates the frictional effects due to the bottom Ekman layer, $-r\zeta$. The WBC in this situation would be parameterized by the drag coefficient described by,

$$\delta_S = \frac{r}{\beta}, \tag{3.2}$$

known as the *Stommel boundary layer* [36].

The first simulation is with a Munk layer of 200 km and for each subsequent case, the Munk layer was decreased by a factor of two, (thus ν was decreased by a factor of 8) with the final case having a Munk layer of 12.5 km ($\nu = 39.0625\,m^2/s$). As δ_M gets smaller, the shear flow is stronger along the western boundary and thus more susceptible to instabilities. For each case, the grid resolution is chosen such that there are sufficient grid points within the Munk layer to ensure that the dynamics are well resolved. In the SW simulations, a square grid was chosen such that there are 7 grid points within the Munk layer. For the QG simulations, there are relatively fewer zonal grid points since the crowding of the Chebyshev grid points near the boundaries ensures a well resolved boundary layer. Table 3.2 lists the five cases used in our simulations along with the kinematic viscosity coefficient, the grid resolution for the SW and QG simulations and number of processors used in the computation, n_p.

The numerical simulations are initially at rest and atmospheric winds begin to generate gyre flow that has qualitatively similar features to that observed in real oceans. The net vorticity of the wind forcing is negative so we expect a clockwise flow at the large scales which is indeed what is observed in our simulations. With the variation of the Coriolis parameter with latitude, a western intensification in our gyre flow occurs [36]. Throughout most of the interior, the flow is southward then westward and near the

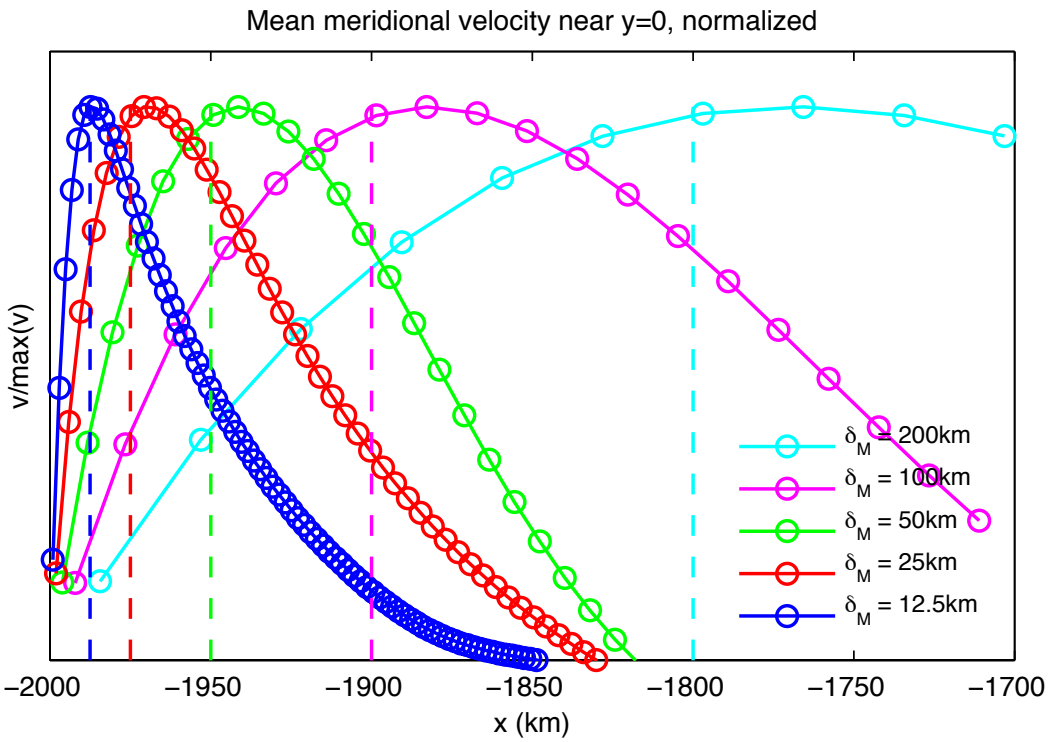


Figure 3.1: $y = 0$ cross-section of the normalized, time-averaged meridional velocity at the western boundary of SW simulation.

western boundary, the northward return flow occurs in the WBC and is much more rapid. As the WBC becomes more narrow, the conservation of mass dictates that the velocities increase in the northward transport. When δ_M is small enough, the strong shear flow generates a turbulent field with a wide range of length and time scales.

3.2 Simulations of Rotating Shallow Water Model

Figure 3.1 shows a zonal slice of the boundary region of the time-averaged meridional velocity for each case near $y = 0$. The mean states are computed starting from day 50 so the initial spin-up time of the gyre flow is ignored. The circles represent the grid points and the vertical dashed lines indicate the δ_M for each case. The velocity was normalized so that the maximum for each case is 1. This figure shows that there is a good agreement between the theoretical Munk layer δ_M and the computed boundary layer.

Figure 3.2 shows a snapshot of the free-surface height, $\eta = h - H$, for case 1 and the arrows indicate the velocity field. We observe a slow southward and westward flow in the interior with maximum speed of approximately 0.03 m/s and a western intensification with a faster northward velocity with maximum speed approximately 0.14 m/s. The dissipation balances the vorticity that is inputted by the wind stress and thus settles towards an equilibrium after 80 days. With the smaller value of ν in case 2 (Figure

3.3), we see a stronger western intensification and an equilibrium is reached at a much later time (approximately 750 days). Case 3 initially generates very large waves near the eastern boundary that then propagate westward. The velocities in the interior have a maximum value of approximately 0.03 m/s as in case 1. We observe a strong western intensification with a steady northward return flow where the maximum velocities are about 0.6 m/s. Figure 3.4 shows the large Rossby waves [24] generated at the early stages of the simulation on days 25, 50, 75 and 100. The waves that are generated tend to slow down after long time and settle towards an equilibrium solution as shown in figure 3.5.

In case 4, the narrow Munk layer and the faster northward velocity creates a strong shear flow along the boundary and generates eddies along the western boundary. Figure 3.6 shows the dynamics of case 4 at the early stages of the simulation. A western intensification is observed as in case 3, but the simulation quickly deviates from this behaviour and generates mesoscale eddies. The dynamics of the eddies are contained within a region near the north-west corner of the basin. The last case in figure 3.7 begins in a similar manner as case 4 where a western intensification is observed in the early stages then eddies are generated. The vortical dynamics occur in a larger region than that observed in case 4 as there are eddies that are travelling eastward along the northern boundary. In the final two cases, the magnitude of velocities are $O(10^{-1}\,\mathrm{m/s})$ in the interior and $O(1\,\mathrm{m/s})$ in the WBC.

To further analyze the numerical results, we examine the power spectrum of the data. The power spectrum can computed by,

$$\int_{\mathcal{C}_{|\vec{k}|}} \left|\hat{U}(\vec{k})\right|^2 ds \tag{3.3}$$

where $\vec{k} = (k, l)$ is a wave number vector, $\hat{U}$ is the Fourier transform of the field and $\mathcal{C}_{|\vec{k}|}$ is a circle with radius $|\vec{k}|$. The power spectrum provides information on how much energy exists in the fields for a given wavenumber vector, $\vec{k}$. We first compute the spectra for each field, η, u and v, and examine the range of length scales that are present in the simulations. Although the fields are nonperiodic, we compute the Fourier transform using odd or even transforms as discussed in 2.2.2. Figures 3.8, 3.9 and 3.10 show the power spectrum of η, u and v plotted against the wavenumbers of the Fourier modes for Munk layer cases $\delta_M = 50$, 25 and 12.5 km at day 500. The vertical dashed line indicates the cutoff from the truncation filter so any information to the left of the line is deemed unphysical and thus should be ignored. The vertical dash-dot line denotes the theoretical Munk layer length, δ_M. Case 3 in figure 3.8 can resolve dynamics in the mesoscale regime well but dynamics towards the submesoscale regime are decimated due to the spectral filtering. Case 4 (figure 3.9) on a 1024×1024 grid does a better job of resolving submesoscale dynamics. The power spectrum shows wavelengths as small as 12 km exist in the physical spectrum howevery the dynamics are damped by lateral viscosity. From figure 3.10, a resolution of 2048×2048 can resolve wavelengths as low as 6 km even with the spectral filtering. Thus with the appropriate grid resolution, submesoscale dynamics can begin to be resolved.

However, in our simulations there is very little motion in the submesoscale range as seen by the low energy in the power spectrum.

Finally, we examine the distribution of kinetic energy across different length scales. The kinetic energy spectrum is calculated by,

$$\int_{\mathcal{C}_{|\vec{k}|}} \left|\hat{u}(\vec{k})\right|^2 + \left|\hat{v}(\vec{k})\right|^2 ds. \tag{3.4}$$

Figure 3.11 shows that power spectrum of kinetic energy computed with the time-average velocity field. The spectrum is plotted against relevant wavenumbers that are unaffected by the spectral filtering. Slopes representing the lines k^{-3} and k^{-6} have been added for comparison. The -3 slope significant as it is predicted by the Kraichnan-Leith-Batchelor (KLB) theory [20]. According to KLB theory, there are two inertial ranges in a two-dimensional turbulent fluid where the effects of viscosity and external forces are negligible. One inertial range is energy and the other is enstrophy. The energy and the enstrophy are injected by external forcing in some intermediate scales between energy and enstrophy inertial ranges. The injected energy is then transferred to larger scales through the energy inertial range by a $-5/3$ scaling, while the enstrophy is transferred to smaller scales through the enstrophy inertial range, by -3 scaling, until it is eventually dissipated by molecular viscosity.

For the laminar flows (Case 1, 2 and 3), there is a local maximum at the intermediate wavenumbers that tends to shift to the right with decreasing viscosity. The kinetic energy tend to be concentrated where the local maximum occurs. For the turbulent cases (Case 4 and 5), the slope is more shallow compared to the laminar cases at small wavenumbers and the local maximum shifts towards larger wavenumbers while increasing in magnitude. Each case drops off at a -6 slope at large wavenumbers which is the dissipation range due to lateral viscosity. It can be shown that a -3 cascade slope arises when computing the power spectrum of the kinetic energy perturbation, where the perturbation is computed by subtracting the mean kinetic energy from the total kinetic energy at a given time. A similar study has been done in [27], where simulations have been compute but with weaker wind forcing. It was shown that the -3 cascade is more apparent when computing the kinetic energy perturbation spectrum. Subtracting the mean state from the total field effectively removes WBC from the data. It was found that the steep WBC produces a shallow slope at lower wavenumbers which altered the spectrum of the mean total kinetic energy. Also, since the vorticity dissipation occurs in the WBC, the steep -6 slope does not appear in the dissipation range of the perturbation field and a -3 slope is observed instead.

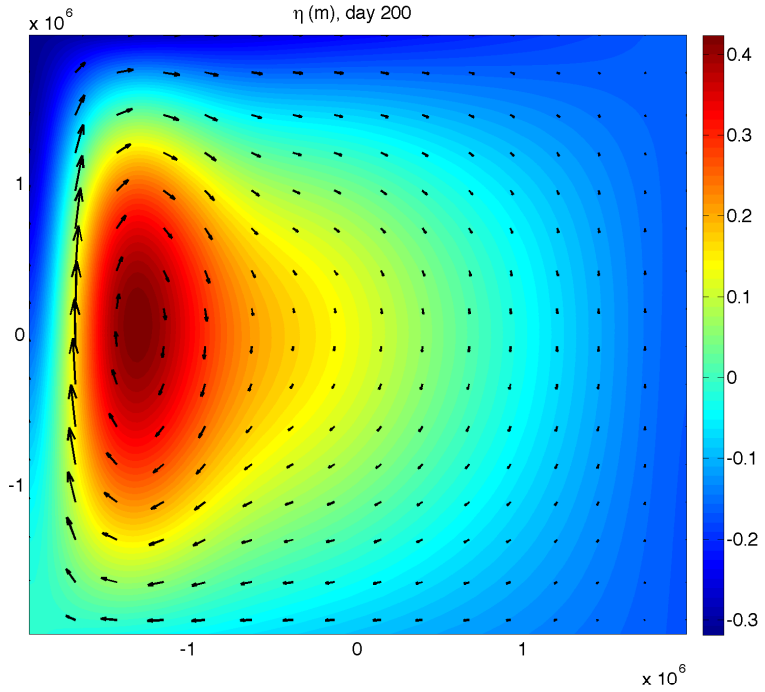


Figure 3.2: free-surface plot of SW simulation for $\delta_M = 200\,\text{km}$ at day 200.

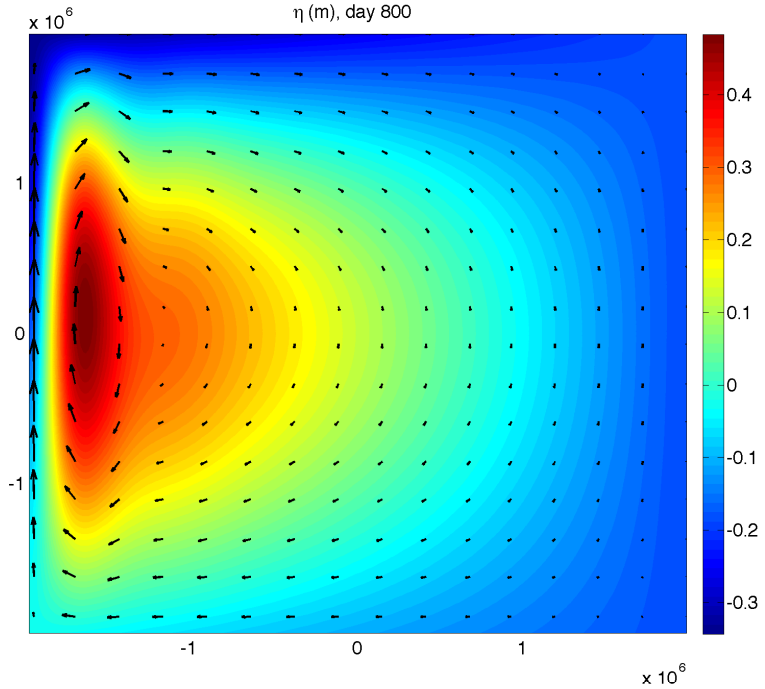


Figure 3.3: free-surface plot of SW simulation for $\delta_M = 100\,\text{km}$ at day 800.

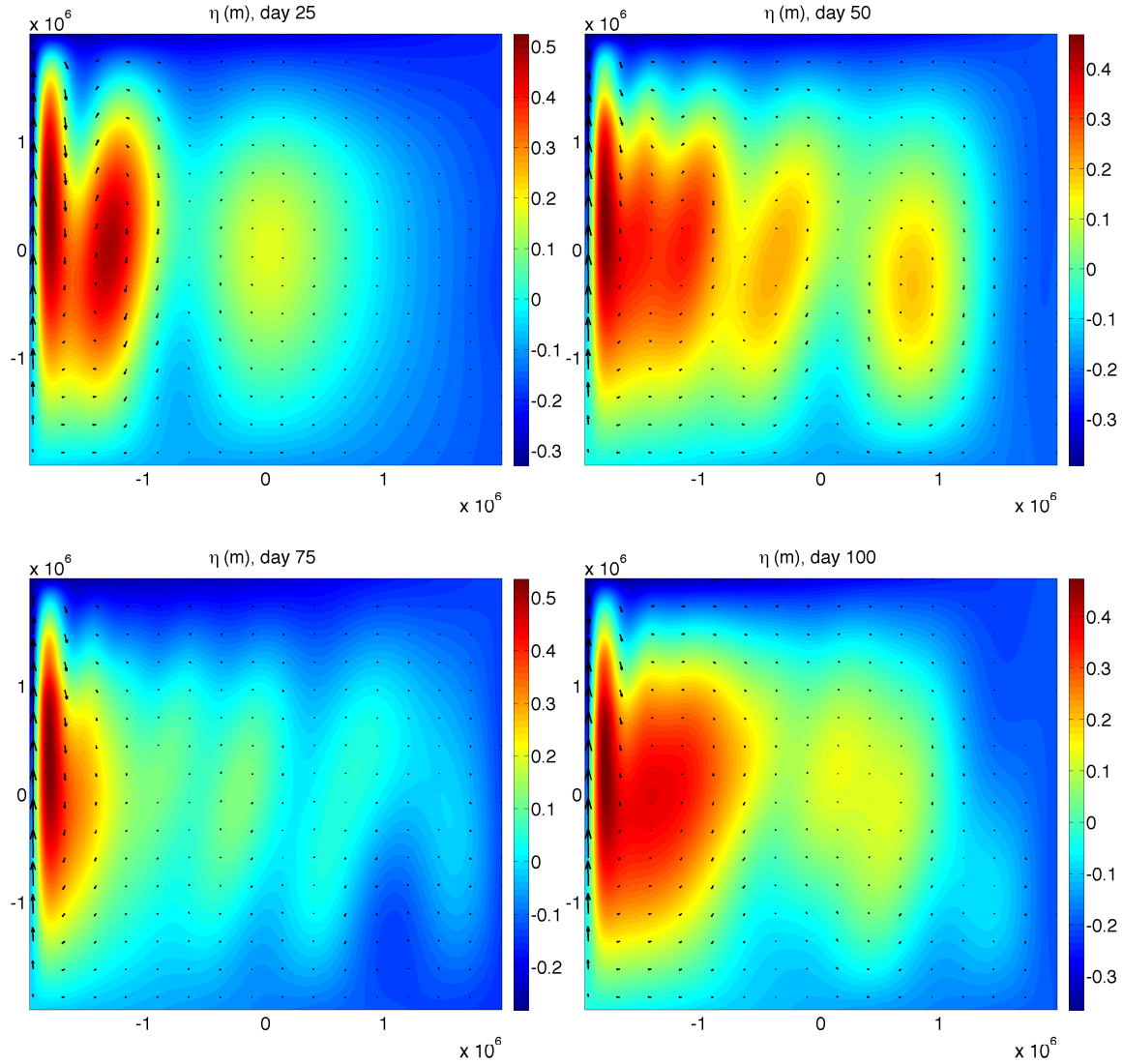


Figure 3.4: free-surface plot of SW simulation for $\delta_M = 50\,\text{km}$ at days 25, 50, 75 and 100.

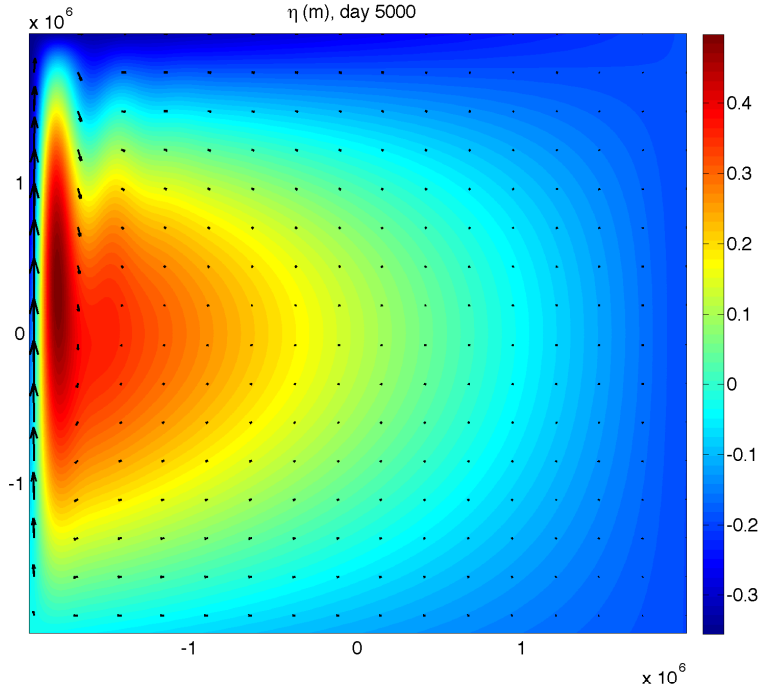


Figure 3.5: free-surface plot of SW simulation for $\delta_M = 50\,\mathrm{km}$ at day 5000.

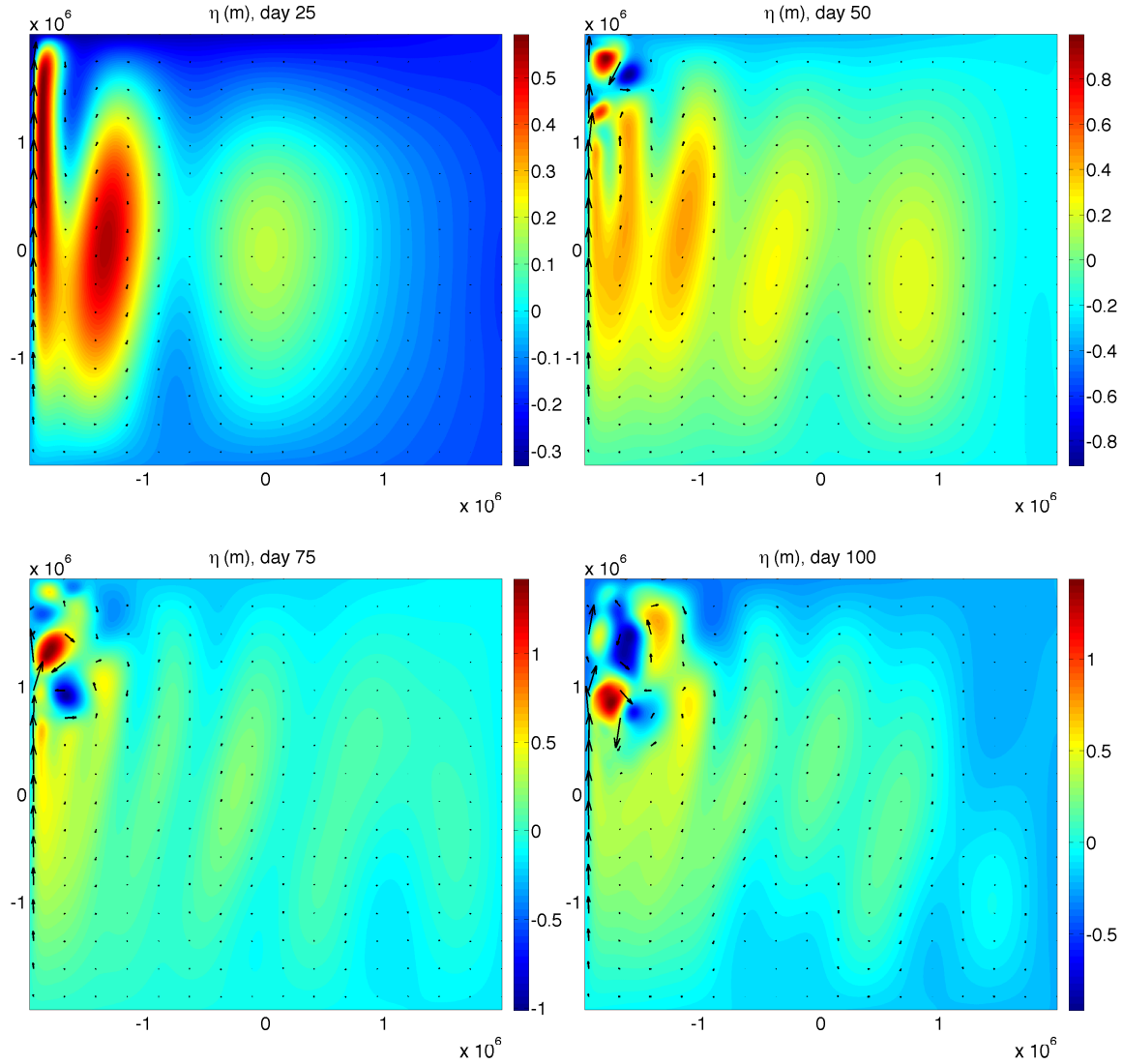


Figure 3.6: free-surface plot of SW simulation for $\delta_M = 25\,\text{km}$ at days 25, 50, 75 and 100.

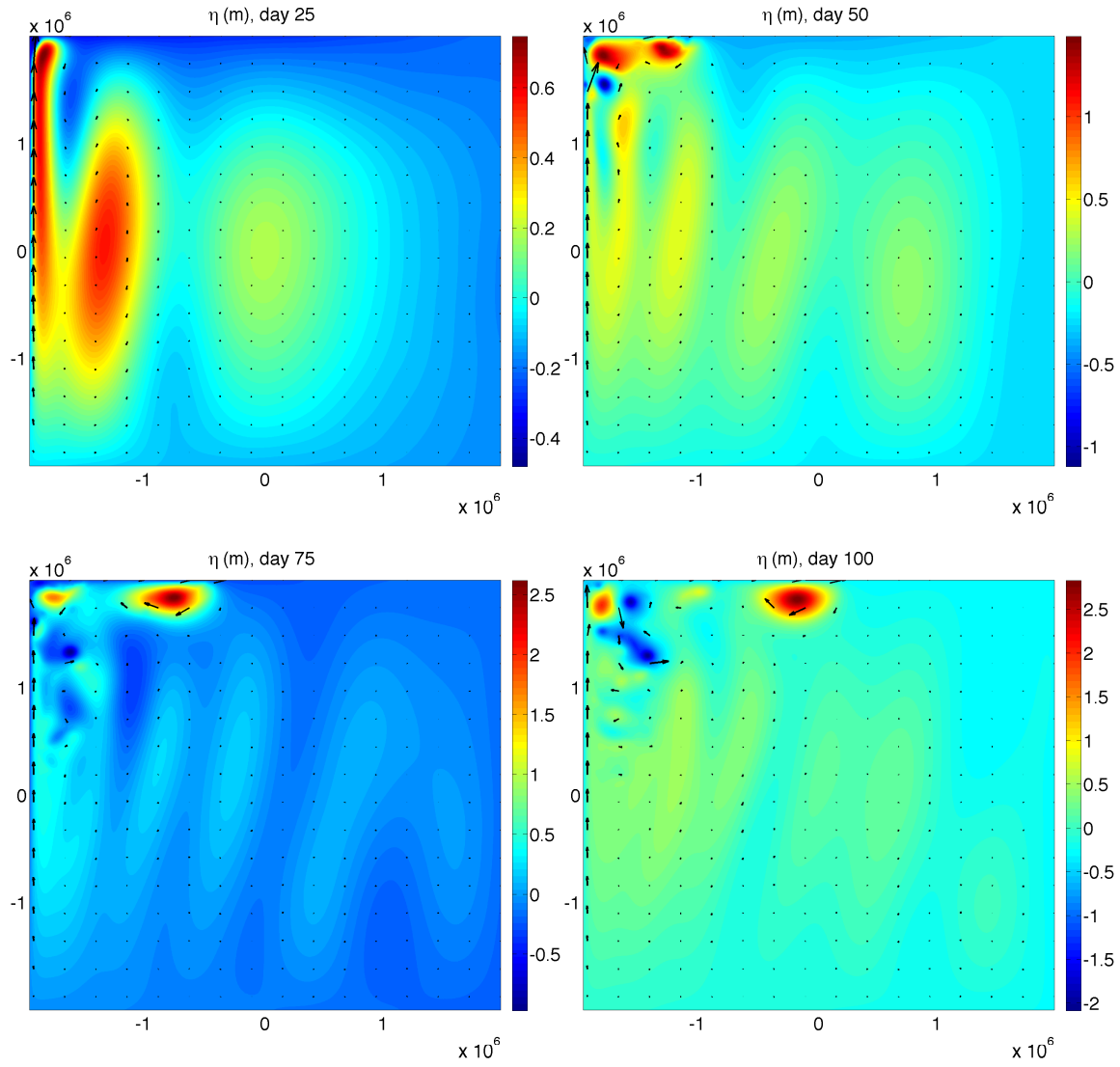


Figure 3.7: free-surface plot of SW simulation for $\delta_M = 12.5\,\text{km}$ at days 25, 50, 75 and 100.

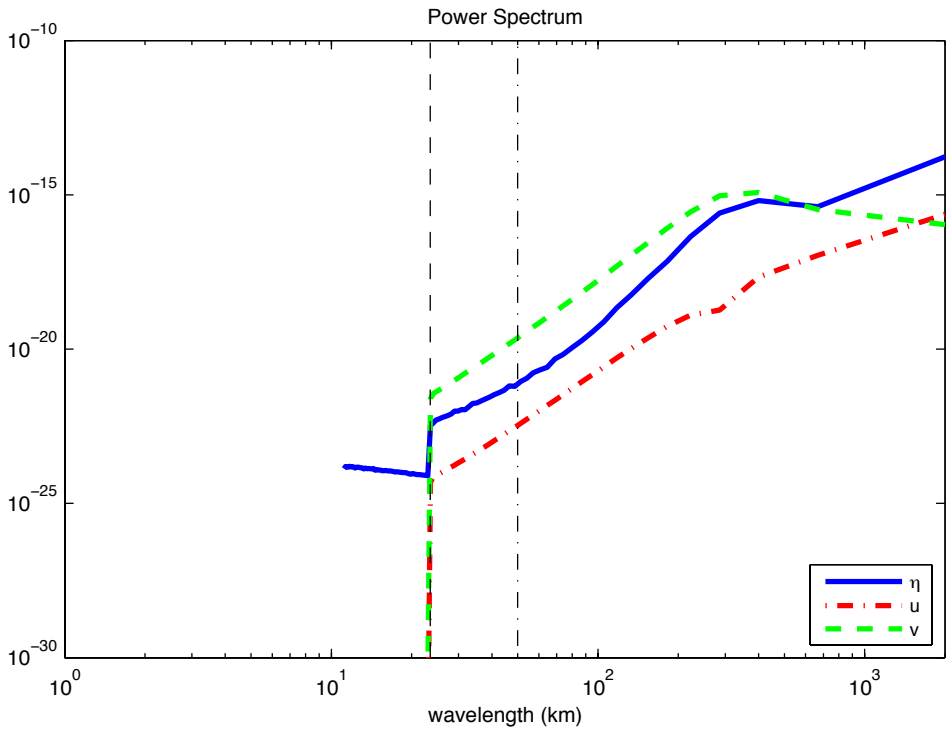


Figure 3.8: Power spectrum of η, u and v for $\delta_M = 50\,\text{km}$ at day 500. The wavelengths to the left of the dashed line are the filtered modes. δ_M is indicated by the dash-dot line.

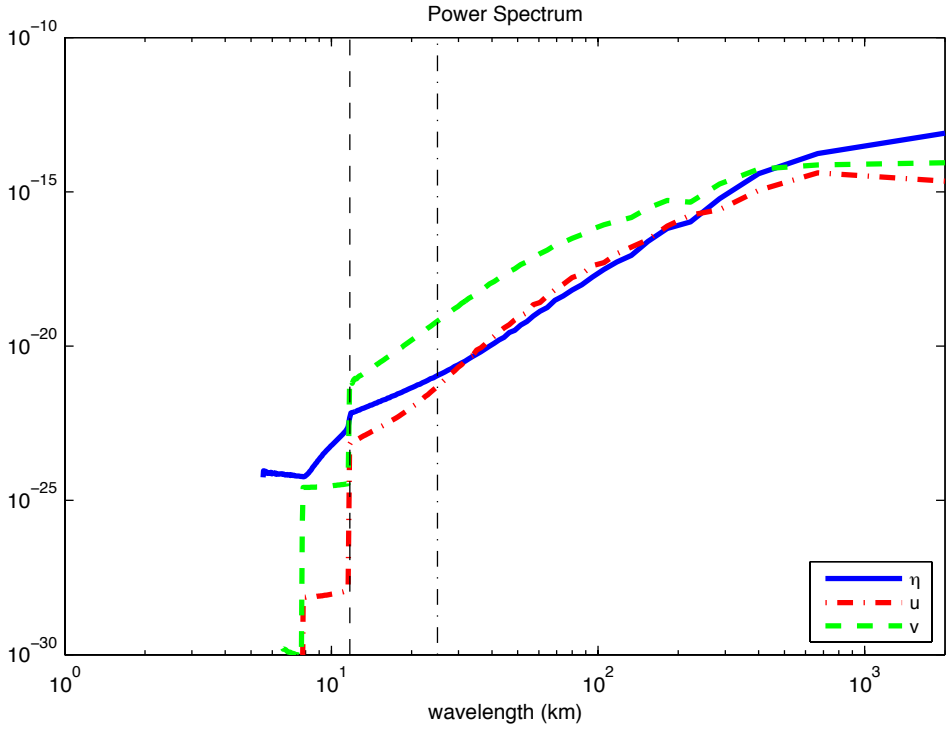


Figure 3.9: Power spectrum of η, u and v for $\delta_M = 25\,\text{km}$ at day 500. The wavelengths to the left of the dashed line are the filtered modes. δ_M is indicated by the dash-dot line.

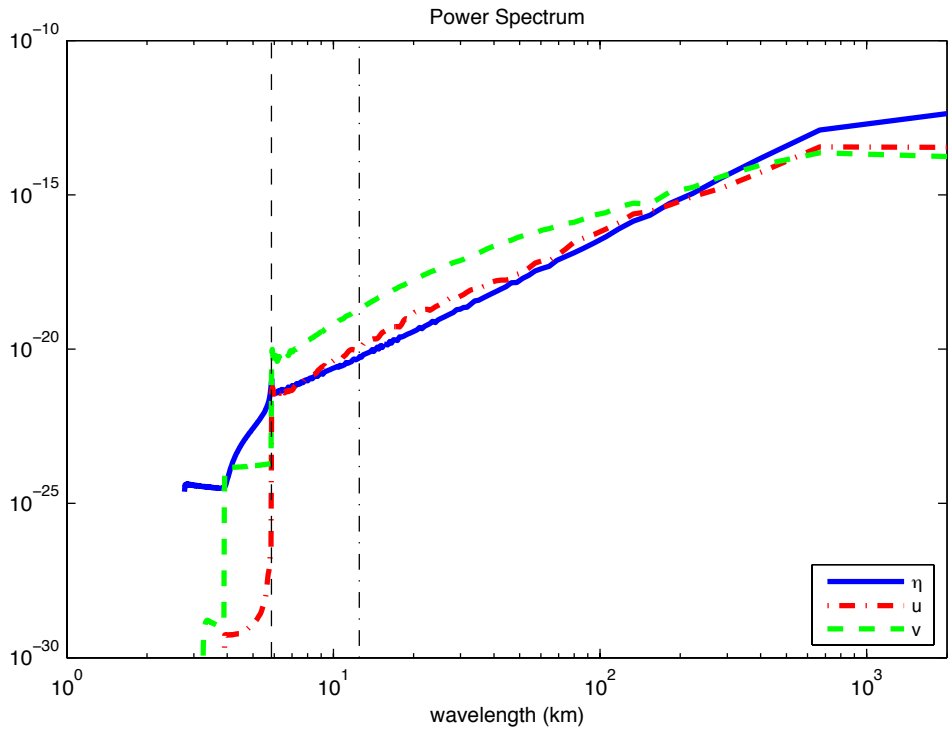


Figure 3.10: Power spectrum of η, u and v for $\delta_M = 12.5\,\text{km}$ at day 500. The wavelengths to the left of the dashed line are the filtered modes. δ_M is indicated by the dash-dot line.

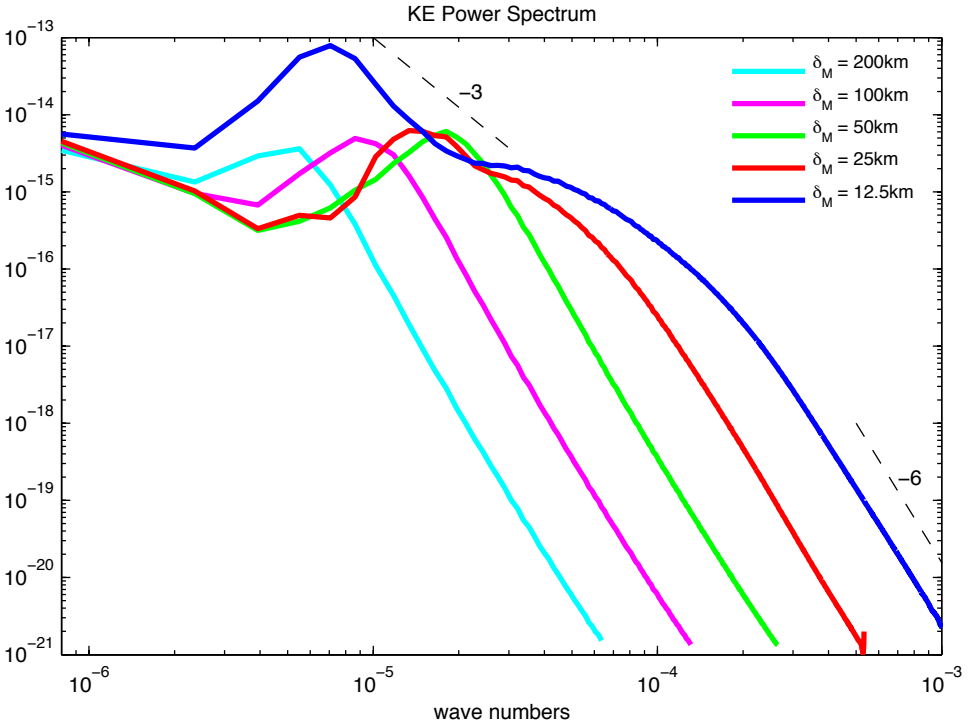


Figure 3.11: Power spectrum of the mean kinetic energy. Slopes representing k^{-3} and k^{-6} are indicated by the dashed lines

3.3 Comparison of QG and SW Models

In this section, several key differences between the QG model and the SW model are discussed that are evident in the numerical simulations. First we compare the results of the gyre flow simulations. Then, the total kinetic energy spectra of the simulations are computed to better compare the simulations quantitatively. We then examine the horizontal divergence of the SW simulations and see for which cases it becomes significant in the gyre flow since QG assumes that the flow is horizontally non-divergent to leading order. Finally, we look at surface gravity waves in the SW model as these waves are neglected in the QG model.

3.3.1 Barotropic Gyre Flow

Simulations of the QG model are shown for each case. Figure 3.12 shows the stream-function, ψ, and vorticity, ζ of cases 1, 2 and 3 at equilibrium. The flow in all of these cases are laminar and have similar qualitative features as the SW simulations. As δ_M decreases, a stronger western intensification is observed in the streamlines. In the vorticity plots, there is a region of relatively strong, positive vorticity within the WBC. The width of this region is determined directly by δ_M. Figure 3.13 shows the early stages of the QG simulation for case 4 at days 25, 50 and 75. The flow is initially laminar but the strong shear flow is unstable and vortex shedding occurs off the western boundary as seen in day 50. This instability produces eddies with some similar behaviour as case 4 in the SW simulation. The vortices tend to be confined within a region in the north-west corner of the basin. Figure 3.14 shows the final case, $\delta_M = 12.5\,\text{km}$ at days 25, 50 and 75. The flow in this case is much more turbulent as vortices are generated in greater region than case 4.

Figure 3.15 shows a cross-section of the mean meridional velocity near the western boundary. Although there are fewer zonal grid points, Chebyshev collocation ensures that there are at least 9 grid points within the Munk boundary layer. This is to be compared to the boundary layer profiles in figure 3.1. The profiles appear to be similar but there are slight quantitative differences between SW and QG within the WBC. Table 3.3 lists the maximum values of the mean meridional velocity and the mean relative vorticity within a region of width $2\delta_M$ from the western boundary. In every case, with the exception of case 1, the mean meridional velocity is slightly higher in the QG simulations. Thus QG tends to have a slightly stronger meridional transport within the WBC than SW with the difference being higher for cases 4 and 5. Also, QG tends show higher relative vorticity values than SW within the WBC. It has been shown in [26] that the QG model overestimates the growth rate of the fields compared to the SW model. Figure 3.16 shows the extreme values of relative vorticity in both QG and SW simulations for case 5. The fluctuations in the QG simulation is much larger than that of the SW simulations. Thus one shortcoming of the QG model is that it provides relative vorticity values that are too large.

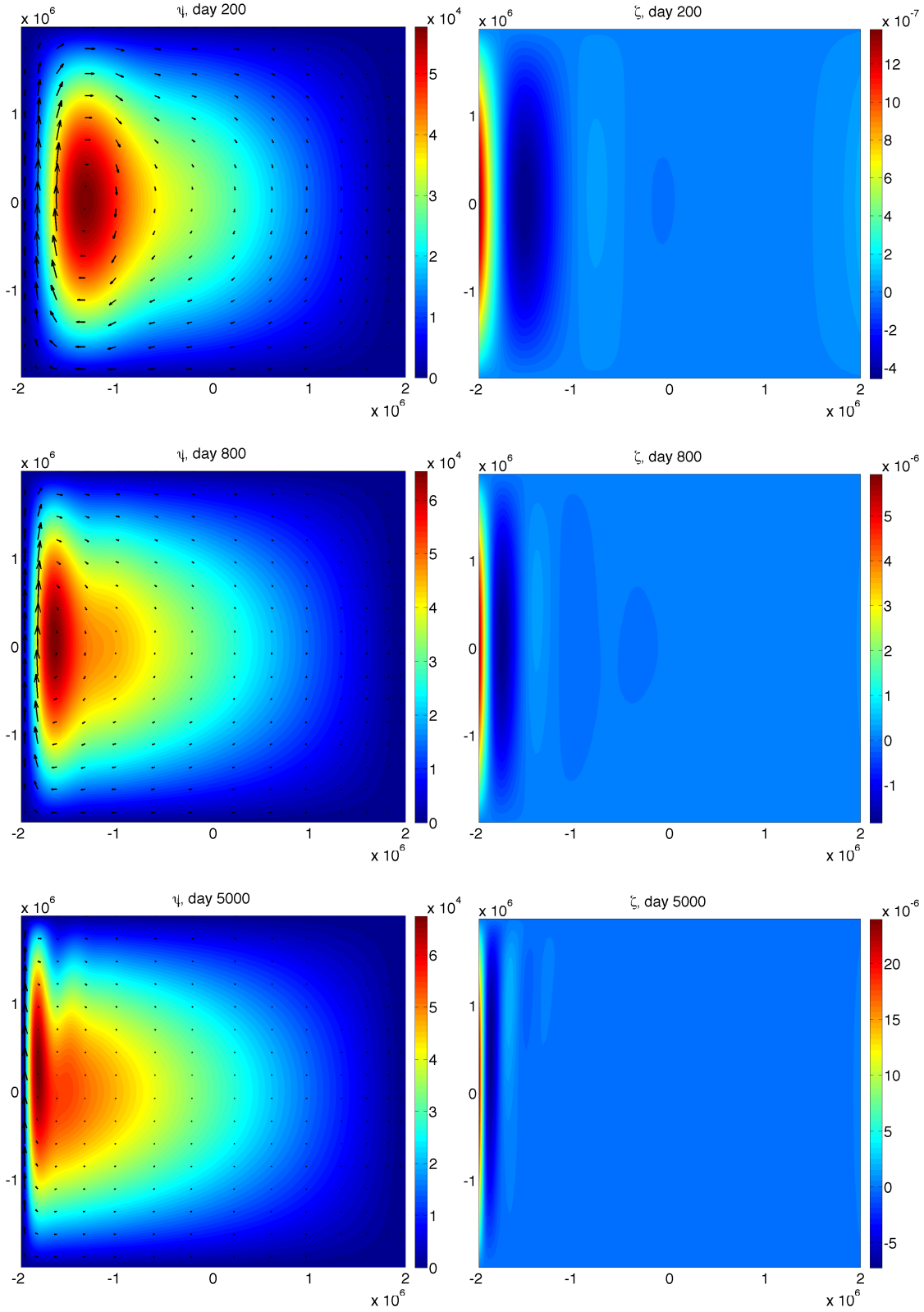


Figure 3.12: Stream-function (left) and relative vorticity (right) plots of QG simulation for $\delta_M = 200, 100$ and $50\,\text{km}$ at equilibrium.

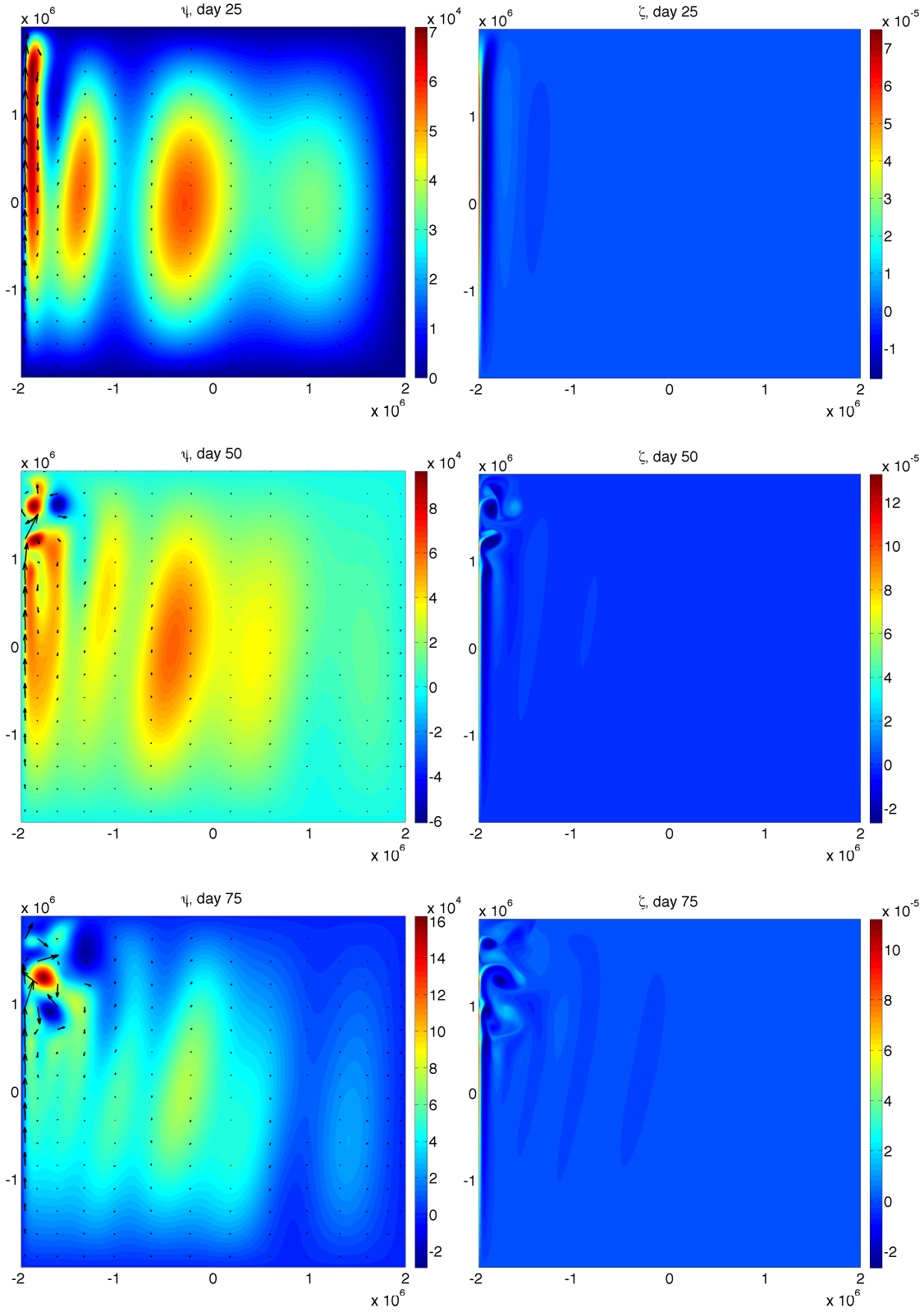


Figure 3.13: Stream-function (left) and relative vorticity (right) plots of QG simulation for $\delta_M = 25\,\text{km}$ at days 25, 50 and 75.

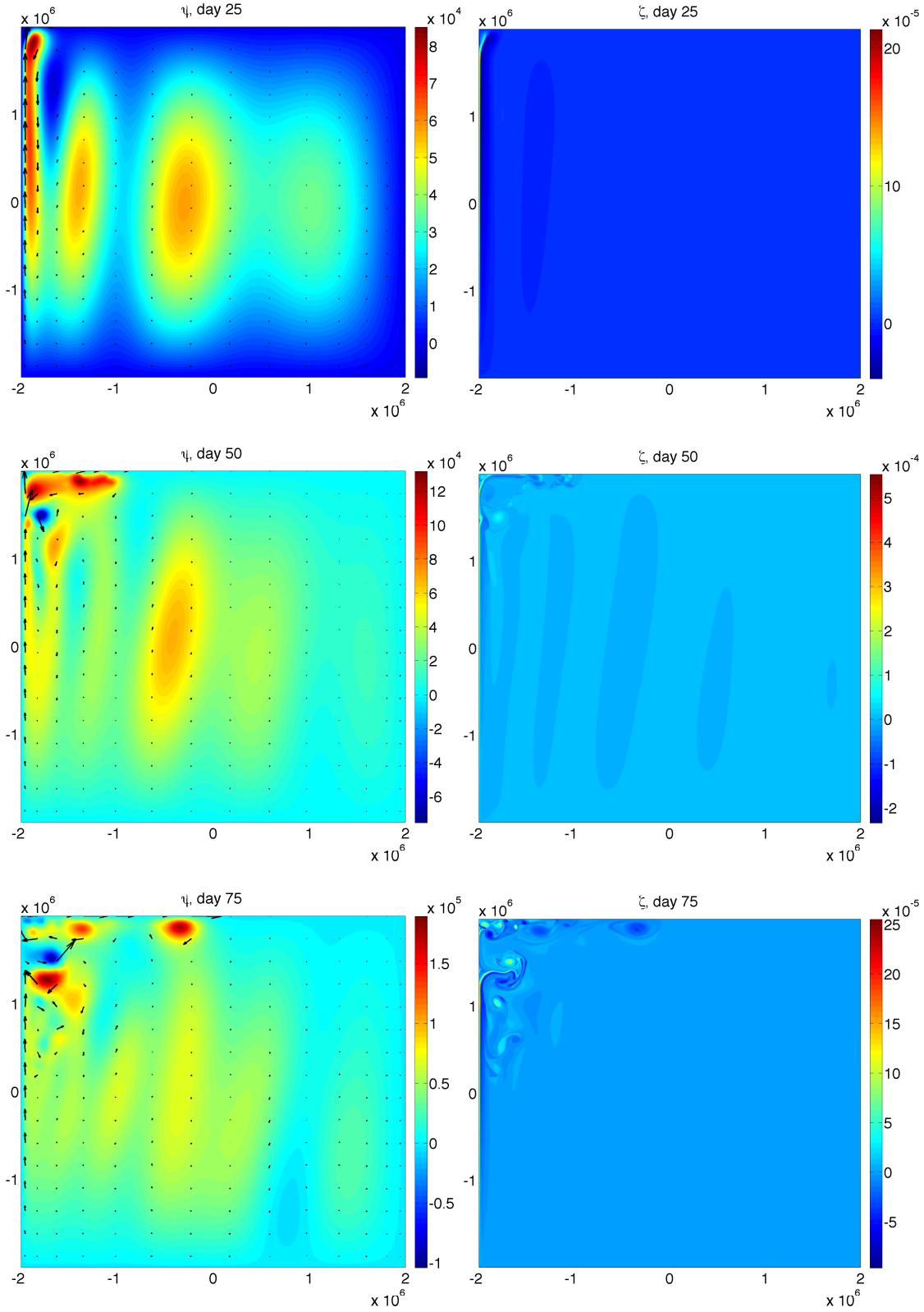


Figure 3.14: Stream-function (left) and relative vorticity (right) plots of QG simulation for $\delta_M = 12.5\,\text{km}$ at days 25, 50 and 75.

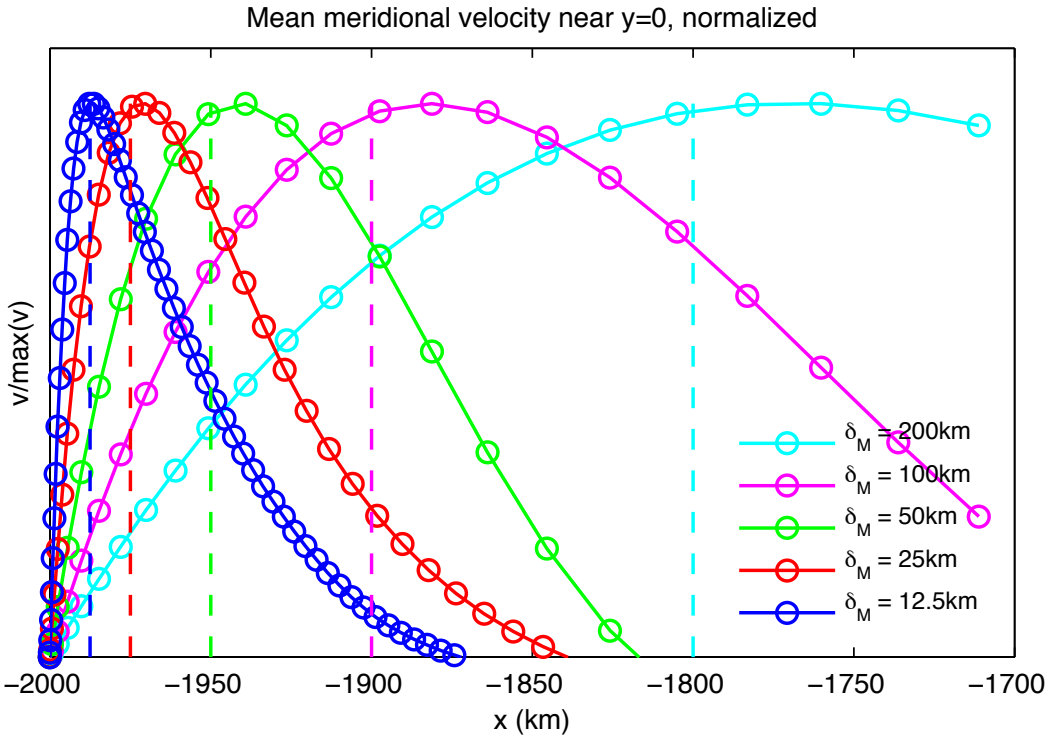


Figure 3.15: $y = 0$ cross-section of the normalized, time-averaged meridional velocity near the western boundary of QG simulation.

	SW		QG	
Case	$\max \bar{v}\ (m/s)$	$\max \bar{\zeta}\ (1/s)$	$\max \bar{v}\ (m/s)$	$\max \bar{\zeta}\ (1/s)$
Case 1	0.14366	1.2835×10^{-6}	0.14358	1.3819×10^{-6}
Case 2	0.31638	5.5218×10^{-6}	0.31650	5.9296×10^{-6}
Case 3	0.64273	2.2333×10^{-5}	0.64289	2.3974×10^{-5}
Case 4	0.91474	7.3169×10^{-5}	0.92611	7.9821×10^{-5}
Case 5	1.3074	2.5236×10^{-4}	1.3704	2.8369×10^{-4}

Table 3.3: Maximum values of mean meridional velocity, v, and relative vorticity, ζ, computed from the SW and QG simulations within a distance of $2\delta_M$ from the western boundary.

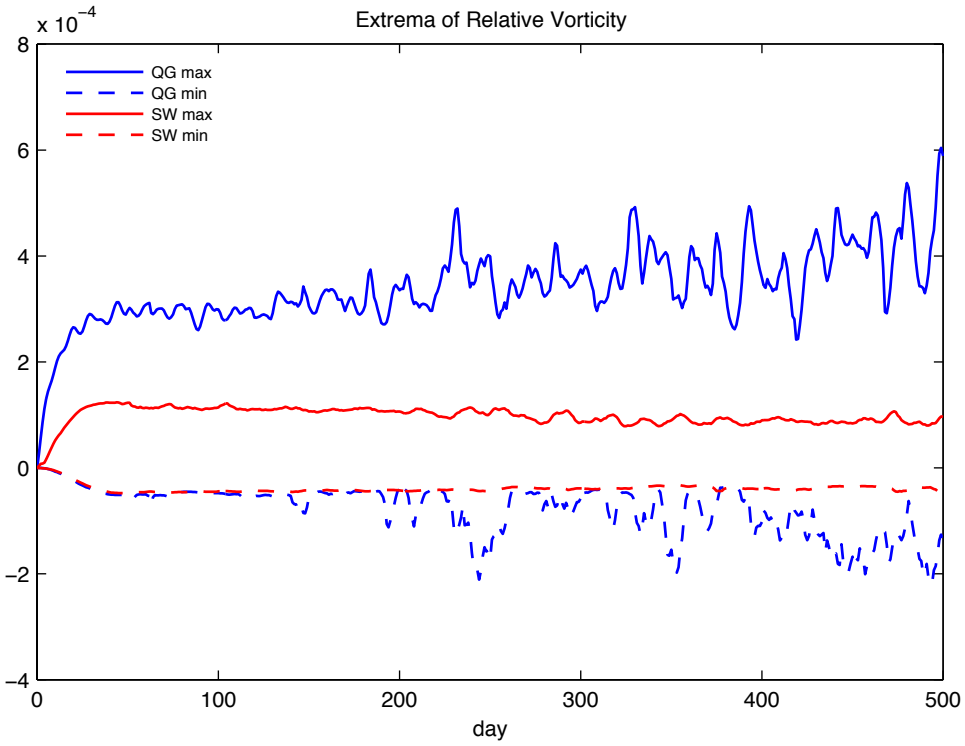


Figure 3.16: Extreme values of relative vorticity in the QG and SW simulations for case 5

3.3.2 Kinetic Energy Spectra

The power spectrum of the mean kinetic energy is computed for each simulation of the QG model and compared to the spectra computed for the SW model. When computing the spectra for the QG data, the Fourier modes cannot be calculated directly due to the Chebyshev grid in the zonal direction. The data is first interpolated on the same uniform grid as the SW simulations, then the same odd/even periodic extensions are used to compute the DFT as discussed in the section 3.2.

Figure 3.17 shows the power spectra of the mean total kinetic energy for each of the five cases. The QG spectrum is indicated by the solid line and the SW spectrum is indicated by the dashed line. The oscillations that occur at high wavenumbers can be attributed to the error due to interpolation of the QG results. The spectra for the laminar cases, $\delta_M = 200, 100$ and $50\,\text{km}$, are virtually identical throughout the spectra preceding the noticeable error. Thus for large scale gyre flow, there is a lot of agreement between QG and SW. For case 4 and 5, there is good agreement between QG and SW at large scales and at the dissipation scales, however, there is an intermediate range where the spectra differ. Although the differences appear to be rather minor, it is these two cases where the differences in the underlying assumptions in the QG and SW model become apparent.

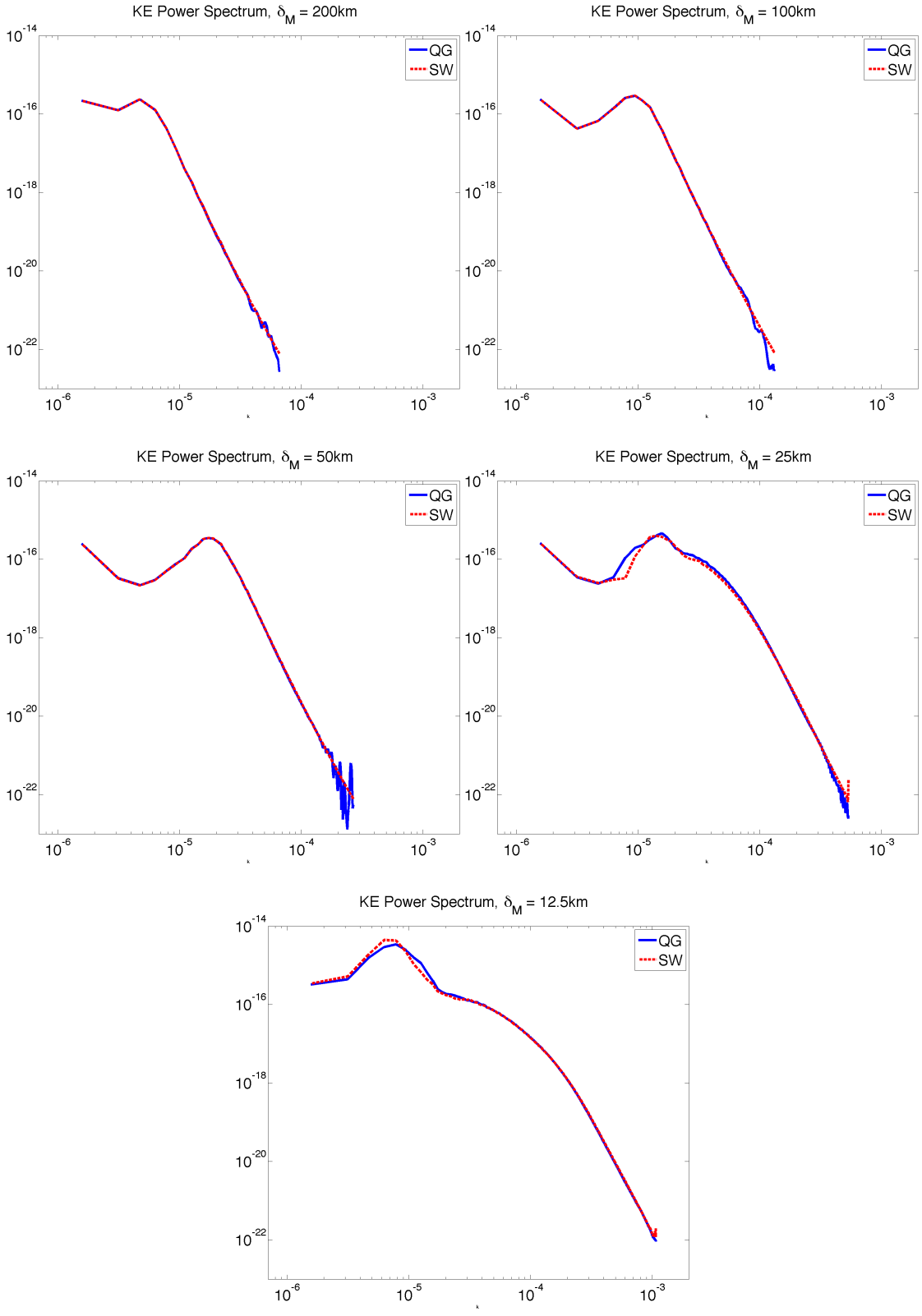


Figure 3.17: Kinetic energy spectra of time-averaged mean states for QG (solid) and SW (dashed) simulations.

3.3.3 Horizontal Divergence

The QG model assumes that the fluid is incompressible in two-dimensions and thus horizontally non-divergent, whereas the SW model assumes three-dimensional incompressibility with the free-surface allowing a non-zero horizontal divergence. Examining the horizontal divergence has implications in the SW model as it is directly related to the vertical velocity at the surface. By depth-integrating the three-dimensional incompressibility equation, (2.18), and applying the assumptions that there is no depth variation in the horizontal velocities and that there is no-normal flow at the bottom of the layer, we get the relation

$$w = -h\vec{\nabla} \cdot \vec{u} \tag{3.5}$$

at the free-surface, $z = \eta$.

Figure 3.18 shows a time series of the maximum magnitude of the horizontal divergence in the SW simulations. For the first three cases, where the flow is laminar, the magnitude of the horizontal divergence is less than 10^{-8} and thus the vertical velocity is less than 5×10^{-6} m/s. The vertical velocity at this magnitude can easily be neglected and thus the laminar SW simulations is very close to the laminar QG simulations. In case 4 and 5 where instabilities occur, the magnitudes of the horizontal divergence increases greatly and the vertical velocity can be as high as 3×10^{-4} m/s for the most turbulent case. In these simulations, it is observed that the region with the highest horizontal divergence occurs within the turbulent WBC and thus it is where the vertical velocity becomes significant. Thus when gyre flow contains dynamics close to the submesoscale regime, the horizontal divergence becomes more significant and thus QG can not properly describe these motions.

3.3.4 Surface Gravity Waves

The free-surface in the SW model allows for a class of waves called *inertia-gravity waves*. These waves are driven by gravity and, without rotation, travel at speeds

$$c_g = \sqrt{gH}. \tag{3.6}$$

The dispersion relation of gravity waves in a non-rotating reference frame is,

$$\omega = \pm c_g |\vec{k}|, \tag{3.7}$$

but with rotation, the dispersion relation becomes

$$\omega = \pm\sqrt{f^2 + c_g^2 |\vec{k}|}, \tag{3.8}$$

which differs slightly compared to the non-rotating case. The scaling done in the QG model eliminates the gravity waves in the simulations. Surface gravity waves play a role in transferring energy across different length scales as it carries kinetic energy due to the fluid motion and potential energy due to the free-surface displacements [18]. To examine

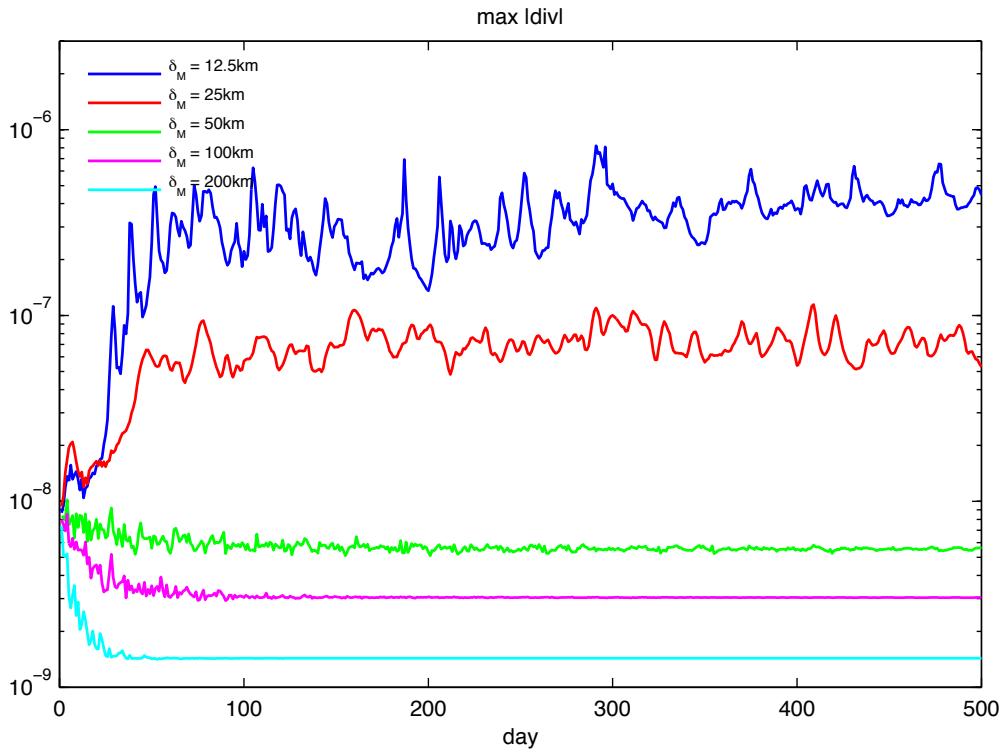


Figure 3.18: Maximum divergence of SW simulations over 500 days

the gravity waves in the SW simulation, the equations can be combined to obtain a wave equation for the time-derivative of the fluid depth,

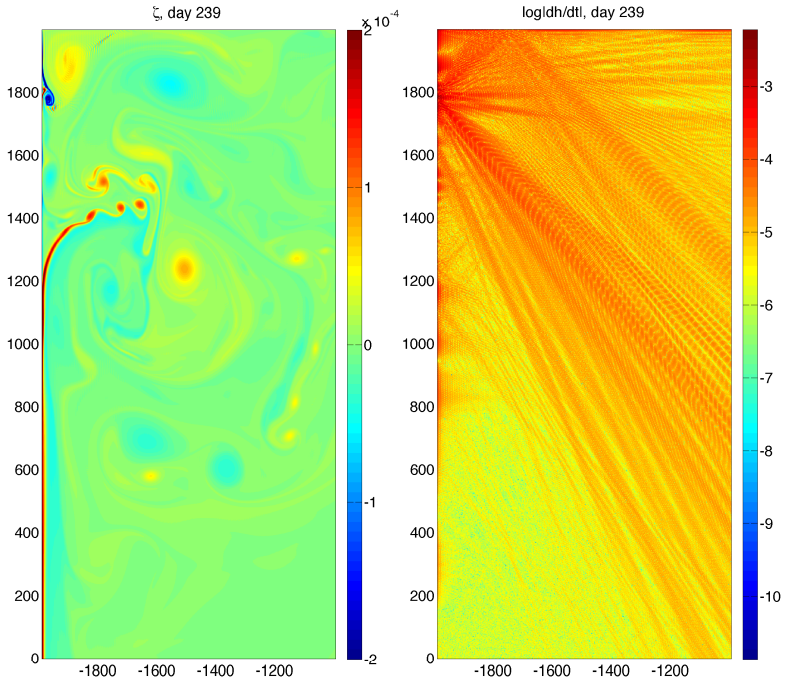
$$\left[\frac{\partial^2}{\partial t^2} + f^2 - c_g\nabla^2\right]\frac{\partial h}{\partial t} = \text{nonlinear terms.} \tag{3.9}$$

The linear version of the above equation is derived in [22] and the nonlinear form follows the same method. The left-hand side of (3.9) is the gravity wave equation in a rotating reference frame. Thus the term, $\frac{\partial h}{\partial t}$, satisfies the nonlinear gravity wave equation and provides us with a means to examine the surface gravity waves. The temporal derivative can be computed by the continuity equation of the SW model,

$$\frac{\partial h}{\partial t} = -\vec{u}\cdot\vec{\nabla}h - h\vec{\nabla}\cdot\vec{u}, \tag{3.10}$$

providing a means to find gravity waves in our simulations. Note that the horizontal divergence appears in the above equation and so we should expect that gravity waves become significant as the horizontal divergence does, namely in the turbulent simulations as discussed in the section 3.3.3. For the case where there is very little depth variation, the horizontal divergence is proportional to $\frac{\partial h}{\partial t}$.

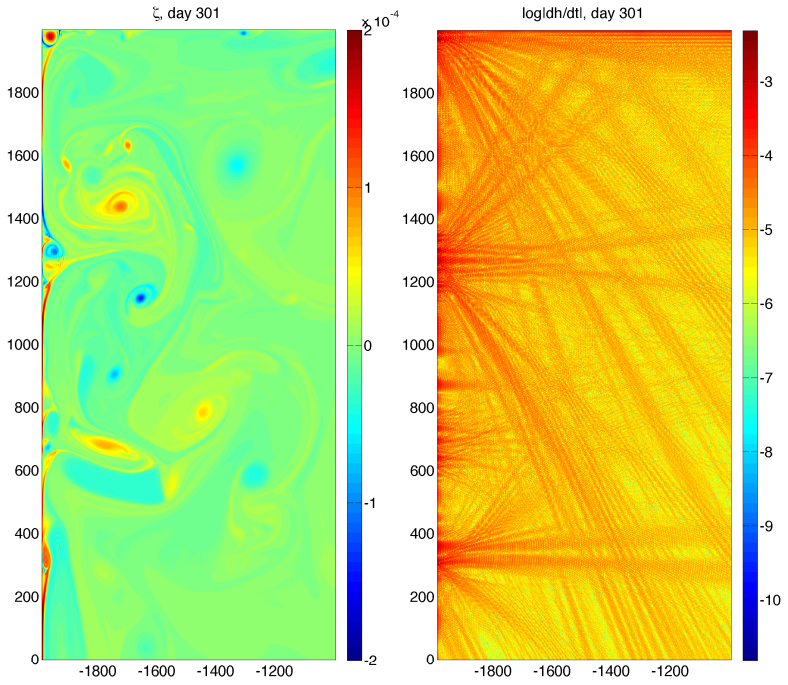
In our simulations, gravity waves appear as highly oscillatory beams in logarithmic plots of $\frac{\partial h}{\partial t}$. The beams radiate from the basin walls and are triggered by various mechanisms for generating gravity waves. However, even at our highest resolution, these



(a) (b)

Figure 3.19: Close-up of a) relative vorticity and b) $\log|\frac{\partial h}{\partial t}|$ for $\delta_M = 12.5\,\text{km}$ simulation on day 239.

gravity waves tend to oscillate at the grid-scales and thus are not properly resolved. Figure 3.19 shows a close-up of the relative vorticity and the logarithm of $\frac{\partial h}{\partial t}$ for case 5 on day 239. In figure 3.19a, a region of strong negative vorticity occurs along the western wall near 1800 km north. At the corresponding location, in figure 3.19b, beams of gravity waves are emitted symmetrically towards the north and south. This result was previously studied in [9] where gravity waves radiated regions of negative absolute vorticity. Figure 3.20 shows plots for the same simulation on day 301. Gravity waves are generated by vortex-wall interactions along the western boundary, most notably near 300 km and, by a vortex dipole occurring near 1300 km north. This is similar to [34] where gravity wave generation by large-scale vortex dipoles is studied in the context of a continuously stratified fluid on a f-plane.



(a) (b)

Figure 3.20: Close-up of a) relative vorticity and b) $\log|\frac{\partial h}{\partial t}|$ for $\delta_M = 12.5\,\text{km}$ simulation on day 301.

Chapter 4

Conclusion

4.1 Summary

The dynamics of the world's oceans occur at a wide range of length and time scales. External forcing, such as atmospheric winds, adds energy at large length scales which cascades down to smaller length scales due to the nonlinear nature of fluids. The predominant model used to study large-scale, barotropic ocean dynamics has been the Quasi-Geostrophic model. However, our simplifying assumptions restricts the motions to two dimensions and thus it is unable to properly describe submesoscale dynamics as QG breaks down when the Rossby number is $O(1)$. Thus when developing a numerical method to study submesoscale dynamics, the QG model is not an appropriate choice. Thus, we solve the Rotating Shallow Water model as it can resolve motions at a wider range of length scales than the Quasi-Geostrophic model.

Numerical methods for both the SW model and QG model that are capable of high-resolution simulations are discussed. To compute dynamics at an optimal range of motions, spectral methods are used for the spatial discretization where the SW model uses a Fourier spectral basis and the QG model uses a Fourier-Chebyshev spectral basis. Both models have an impeding feature that leads to long runtimes in computation. The SW model is stiff due to slow Rossby waves and fast gravity waves and thus requires small time steps for accuracy and stability. The QG model requires the solution of an inverse problem which is computationally expensive. To overcome the long runtimes, the methods have been implemented using the MPI protocol and executed on parallel computer clusters. The nonlinear nature of the models and the finite number of spectral basis functions require a spectral filtering procedure for each method.

Simulations for both SW and QG models have been computed using physical parameters that are typically observed in a midlatitude, subtropical ocean in the northern hemisphere. The wind stress is chosen such that it induces a single-gyre flow. By varying the width of the Munk boundary layer, δ_M, a wide range of dynamics on various length and time scales are observed. The simulations where $\delta_M = 100, 200$ and $50\,\text{km}$ result in a laminar flow which eventually reaches an equilibrium. It is observed in these cases that

the SW and QG simulations have qualitatively similar behaviour. A western intensification is observed in both the free-surface of the SW simulations and the streamlines of the QG simulations and this intensification becomes stronger as the Munk layer narrows. The simulations where $\delta_M = 25$ and 12.5 km produce a strong shear which becomes unstable resulting in a more turbulent flow. The quantitative and qualitative behaviour between SW and QG begin to differ in these cases as was apparent in the kinetic energy spectra.

For the turbulent cases, the differences between the underlying assumptions in the barotropic ocean models are evident in the simulations. It is for these less viscous cases that the horizontal divergence and the surface gravity waves, which are absent in the QG model, become substantial in oceanic gyre flow. The horizontal divergence is small in magnitude and can be neglected in the laminar simulations. However in the turbulent cases, the magnitude increases by an order of magnitude. Although still small, it is apparent that the horizontal divergence is more significant in turbulent flows. The horizontal divergence has vital implications as it is directly related to the vertical velocity at the surface of the ocean and thus three-dimensional motions must be considered. Furthermore, SW simulations have shown the existence of gravity waves in turbulent dynamics that are completely absent in QG simulations. It is observed in the $\delta_M = 12.5$ km case that vortex interactions with the boundaries generate beams of gravity waves that propagate across the ocean basin. The importance of gravity waves in large-scale flows is to be studied in future work.

4.2 Future Work

Future work includes extending the current SW model to incorporate more physically relevant features of a real ocean. An ocean is a continuously stratified medium, whereas the models presented in this thesis assume a homogeneous fluid. Thus, a natural extension to the numerical method presented is to implement a multi-layer SW model with varying fluid densities to observe the effects of stratification in wind-driven gyre flow in a discretely stratified context. A two-layer SW model will effectively double the number of equations to compute and thus double the computation time. A convenient feature of the SW model is that it can easily incorporate the effects of bottom topography without modifying the numerical methods. By adding terms to the continuity equation, the consequences of an uneven ocean floor can be studied. For a multi-layer method, the fluid-depth evolution equation corresponding to the bottom-most layer would be modified.

Further work in improving the QG solver is considered. Currently, the initial steps prior to the AB3 time-stepping uses FE and AB2 methods which are first-order and second-order accurate, respectively. An improvement to the method would be to consider higher-order startup schemes to remove the impact of the low-order methods to the rest of the simulation. Furthermore, the time-step size changes throughout the simulation to satisfy the CFL condition, albeit not often, and thus startup methods are employed again to reset the AB3 time-stepping. Thus higher-order startup methods should be explored to maintain the third-order accuracy throughout the simulations. Another improvement

to consider is the number of inverse problems to compute. The spectral filtering in the meridional direction completely removes Fourier modes at high wavenumbers. Thus, there is a redundancy when solving for the stream-function as inversions associated with high wavenumbers need not be solved. The current implementation should be revised to provide a more efficient simulation.

References

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[2] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods: Fundamentals in Single Domains*. Springer-Verlag, 2006.

[3] J. W. Cooley and J. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, 19(90):297–301, 1965.

[4] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9:251–180, 1990.

[5] R. Courant, K. Friedrichs, and H. Lewy. On the partial difference equations of mathematical physics. *IBM Journal*, 11:215–234, 1967.

[6] W. P. Crowley. A numerical model for viscous, free-surface, barotropic wind driven ocean circulations. *Journal of Computational Physics*, 5:139–168, 1970.

[7] B. Cushman-Roisin and J.-M. Beckers. *Introduction to Geophysical Fluid Dynamics*. Academic Press, 2011.

[8] W.-S. Don and D. Gottlieb. Spectral simulation of unsteady compressible flow past a circular cylinder. *Computer Methods in Applied Mechanics and Engineering*, 80:39–58, 1990.

[9] R. Ford. Gravity wave radiation from vortex trains in rotating shallow water. *Journal of Fluid Mechanics*, 281:81–118, 1994.

[10] L. Fox and I. B. Parker. *Chebyshev Polynomials in Numerical Analysis*. Oxford University Press, 1968.

[11] B. Fox-Kemper and J. Pedlosky. Wind-driven barotropic gyre I: Circulation control by eddy vorticity fluxes to an enhanced removal region. *Journal of Marine Research*, 62:169–193, 2004.

[12] M. Frigo and S. G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. Special issue on "Program Generation, Optimization, and Platform Adaptation".

[13] M. Frigo and S. G. Johnson. FFTW Home Page. http://www.fftw.org, 2009.

[14] P. R. Gent. The energetically consistent shallow-water equations. *Journal of the Atmospheric Sciences*, 50(9):1323–1325, 1993.

[15] P. Godon and G. Shaviv. A two-dimensional time dependent chebyshev method of collocation for the study of astrophysical flows. *Computer Methods in Applied Mechanics and Engineering*, 110:171–194, 1993.

[16] S. G. Johnson and M. Frigo. A modified split-radix FFT with fewer arithmetic operations. *IEEE Transactions on Signal Processing*, 55(1):111–119, 2007.

[17] D. W. Kammler. *A First Course in Fourier Analysis*. Cambridge University Press, 2nd edition, 2007.

[18] P. K. Kundu and I. M. Cohen. *Fluid Mechanics*. Academic Press, 4th edition, 2008.

[19] J. H. LaCasce. On turbulence and normal modes in a basin. *Journal of Marine Research*, 60:431–460, 2002.

[20] C. E. Leith. Diffusion approximation for two-dimensional turbulence. *Physics of Fluids*, 11:671–673, 1968.

[21] R. J. LeVeque. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*. SIAM, 2007.

[22] J. C. McWilliams. *Fundamentals of Geophysical Fluid Dynamics*. Cambridge University Press, 2006.

[23] W. H. Munk. On the wind-driven ocean circulation. *Journal of Meteorology*, 7(2):19–93, 1950.

[24] J. Pedlosky. *Geophysical Fluid Dynamics*. Springer-Verlag, 2nd edition, 1987.

[25] J. Pedlosky. *Ocean Circulation Theory*. Springer-Verlag, 1996.

[26] F. J. Poulin and G. R. Flierl. The nonlinear evolution of barotropically unstable jets. *Journal of Physical Oceanography*, 33:2173–2192, 2003.

[27] F. J. Poulin, W. Ko, B. Fox-Kemper, and N. K.-R. Kevlahan. Spectral and ageostrophic characteristics of turbulent wind-driven gyre flow. *Journal of Fluid Mechanics*, Submitted.

[28] F. W. Primeau and D. Newman. Bifurcation structure of a wind-driven shallow water model with layer-outcropping. *Ocean Modelling*, 16:250–263, 2007.

[29] P. B. Rhines and W. R. Holland. A theoretical discussion of eddy-driven mean flows. *Dynamics of Atmospheres and Oceans*, 3:289–325, 1979.

[30] C. Schar and R. B. Smith. Shallow-water flow past isolated topography. part 1: Vorticity production and wake formation. *Journal of the Atmospheric Sciences*, 50(10):1373–1412, 1993.

[31] G. Sewell. *The Numerical Solution of Ordinary and Partial Differential Equations*. John Wiley & Sons, 2nd edition, 2005.

[32] X. Shao and S. G. Johnson. Type-II/III DCT/DST algorithms with reduced number of arithmetic operations. *Signal Processing*, 2008.

[33] X. Shao and S. G. Johnson. Type-IV DCT, DST, and MDCT algorithms with reduced numbers of arithmetic operations. *Signal Processing*, 88(6):1313–1326, 2008.

[34] C. Snyder, D. J. Muraki, R. Plougonven, and F. Zhang. Inertia-gravity waves generated within a dipole vertex. *Journal of the Atmospheric Sciences*, 64:4417–4441, 2007.

[35] D. Steinmoeller. Flow separation on the β-plane. Master's thesis, University of Waterloo, 2009.

[36] H. Stommel. The westward intensification of wind-driven ocean currents. *Transactions American Geophysical Union*, 29(2):202–206, 1948.

[37] H. U. Sverdrup. Wind-driven currents in a baroclinic ocean: with applications to the equatorial currents of the eastern Pacific. *Proceedings of the National Academy of Sciences*, 33(11):318–326, 1947.

[38] L. N. Trefethen. *Spectral Methods in MATLAB*. SIAM, 2000.

[39] L. N. Trefethen and D. Bau. *Numerical Linear Algebra*. SIAM, 1997.

[40] L. Yuan and K. Hamilton. Equilibrium dynamics in a forced-dissipative f-plane shallow-water system. *Journal of Fluid Mechanics*, 280:369–394, 1994.