STOCHASTIC MODELING OF TOPOLOGICAL GEOPHYSICAL WAVES

by

Alexander Lawson

Advisor: John Bradley Marston

Submitted to Brown University in partial fulfillment

of graduation requirements for University Honors

Department of Physics

May 2020

STOCHASTIC MODELING OF

TOPOLOGICAL GEOPHYSICAL WAVES

Abstract

by Alexander Lawson, Sc.B.
Brown University
May 2020

Thesis Advisor: John Bradley Marston

In this Senior Thesis, a derivation is presented that demonstrates how the k-space phase singularities in Poincairé inertia-gravity waves, which have been shown by Deplace, Venaille, and Marston to provided a novel, topological explanation for equatorial geophysical waves [1], translate directly to measurable signals in the Fourier-transformed, *unequal*-time cross-correlations of the relevant fields. It was shown that within the stochastically driven Linear Shallow Water Equations that for height h and velocity u, the correlation function $\tilde{C}(\omega) \equiv \int_{-\infty}^{\infty} e^{-i\omega\tau} E[h(t+\tau, k_x, k_y)u^*(t, k_x, k_y)]d\tau$, ($E$ denoting expected value), contains phase singularities analogous to those of the Poincairé waves. This result was supported by correlations of data produced by a highly idealized numerical simulation of the Linear Shallow Water Equations, and compared to data taken from a more complex global atmospheric climate model produced with the GCM program.

ACKNOWLEDGMENT

This thesis was completed under the supervision of Professor John Bradley (Brad) Marston. Without his expertise, completion of this work would not have been possible, and without his thoughtful guidance, completion of this work by myself would not have been possible. In particular, he is thanked for the gracious and considerate way he handled the advising of this thesis while the Spring 2020 semester was upended by the COVID-19 pandemic.

The Brown University Department of Physics as a whole not only provided a superb undergraduate Physics education, but also provided ample opportunities for undergraduate research, and guidance for finding the research groups and subjects that I find most fulfilling. In particular, my undergraduate advisor Professor Ian Dell'Antonio helped me to assemble a personalized curriculum and navigate research opportunities, while Professors Robert Pelcovits and James Valles guided me through the senior thesis process. Particular thanks is owed to Professor Kemp Plumb, who facilitated my first research experience in his experimental materials lab during the summer of 2018. The Brown University Undergraduate Teaching and Research Awards program is thanked for funding that research and Kemp Plumb is especially thanked for his guidance, which continued long after I has ceased contributing to that research.

Similarly Professor Aleksandar Donev and Brennan Sprinkle (Ph.D.) at the Courant Institute of Mathematical Sciences at NYU provided an opportunity for me to contribute to their theoretical and computation colloid research in the summer of 2019. The techniques they taught me translated directly into this thesis. The NYU MRSEC Research Experiences for Undergraduates program is thanked for providing funding, housing and advising during that stay in New York. The Courant Institute's SURE program is thanked for their guidance

as well.

On the whole, the greater Brown University community, administration and faculty are thanked for an amazing four years. The open curriculum, allowed me to chase after my curiosities unhindered, permitting me to radically change my course of study when I saw the need. Without this freedom, I don't know if I would have landed on this subject I enjoy so much. Finishing my last semester from home, it has already become clear what an incredible community exists at Brown, and how meaningful and enjoyable it was to be a part of it.

Of course, the greatest appreciation is owed to my parents and grandparents. Their contributions to my evolution, both personal and academic, over these past four years are so fundamental that it feels inappropriate to list them here.

A Note to the Reader

This research will continue past the time of writing, as there is plenty more left to do, specifically the overarching goal of this work: observing topological behavior in geophysical data. This work presents what these topological signals may look like, but it will soon be time to look for them.

Seeing as I will be beginning a Ph.D. program at the University of Illinois at Urbana-Champiagn this upcoming fall (2020), someone else at Brown University will likely be joining this project. This thesis is written with that person in mind. Though I will certainly be in contact, it is my intention that this thesis can be a resource for someone without much experience in this field to retrace my steps quickly. Because of that, the theoretical section of this thesis contains more detail than what would probably be of interest to a curious, but uninvolved reader.

This thesis is written to be understood by Physics students at or above a standard advanced undergraduate level. Familiarly with vector calculus and differential equations, especially regarding the use of Fourier transformations is assumed. Proficiency with linear algebra, especially linear algebra over the complex numbers is necessary. This thesis makes heavy use of Dirac's 'bra-ket' notation, so although the subject of this work is not Quantum Mechanical at all, a first semester of Quantum Mechanics is almost necessary. Loose familiarity with statistics: the concept of expectation values and probability distributions is also helpful. Some of the background section draws on the language and concepts of solid-state physics, but nothing that would be hard to understand without solid-state physics experience is essential to this thesis. Finally, though no familiarity with programming is needed to understand this work, it is certainly necessary to continue it, as the details of implementing

of these numerical techniques in Python will not be explained here. Aside from that, all other topics: general concepts from topology, fluid dynamics, and stochastic processes are presented at a sufficient level (to understand this work) in the text.

TABLE OF CONTENTS

Dedication

This thesis is dedicated to my parents and grandparents
who showed me the value of education,
and allowed for me to receive a such an excellent one.

Chapter One

Introduction and Background

1.1 An Introduction to Geophysical Fluids

Despite its outstanding importance, predicting the climate is difficult. As unprecedented concentrations of carbon dioxide are now trapping excessive heat in the atmosphere, there is an added urgency in predicting an increasingly erratic climate system. Still, much about fluids evades the grasp of science. Formulated nearly two centuries ago, the complexity of the equations governing the fluid motion are to thank for this elusive behavior.

These equations are fundamentally the natural application of conservation of mass and momentum to flowing media, with some thermodynamics added to deal with compressibility, and a possible stress tensor added to deal with viscosity. In geophysical flows, a term which will be taken to mean the fluid motions on scales similar to the Earth, mainly the atmosphere and oceans, there is the extra complication of a rotating planet. So we add centrifugal force, and more significantly the Coriolis force, which causes the spiraling storms so characteristic of our planet.

Though these equations are not too difficult to derive, they can be extremely difficult to solve, both analytically and numerically. A nonlinear velocity term introduces chaotic dynamics to the system, and turbulent flows often impede efforts to formulate analytic solutions. The famous 'Lorenz Attractor,' the archetypal model for deterministic chaos, is

in fact a model for atmospheric convection.

In an analytical context, statistical assumptions like the treatment of turbulence as a random fluctuation η, are often used to model these systems. Still there are major unanswered questions in these approaches. The onset of turbulence: why and in what regimes it occurs, lacks a precise theoretical backing. Using a stochastic variable like η also has its issues. Equations can be averaged to produce a system of statistical equations, relating statistical properties like the expected value of η to the expected value of η^2 and η^3, often called the first, second and third moments. Still, in the case of turbulence it is unclear if these moments diminish in significance with greater n and if these moment equations can be closed.

On the numerical side, geophysical flows have the challenging characteristic that their features influence each other across on a wide range of scales, both in time and in space. The specifics of numerical modeling will be addressed in Section 3, but it is fundamentally based on some sort of step-based approximation to integration, like Riemann summation:

$$\int_{t_1}^{t_2} f(t)dt \approx \sum_{n=0}^{N_t} f(t_1 + \frac{t_2 - t_1}{N_t} n\Delta t)\Delta t$$

The problem with interconnected features at many orders of magnitude is that $b - a$ must be large enough to capture them all, while the step size Δt must be small enough to not step over an important feature. Therefore the computer must be willing to do N_t operations, where N_t is very large. In one dimension this is not so bad, but for fluids N_t operations must be done on a grid of size $N_x \times N_y \times N_z$, where each N_{x_i} must be very large for the same reasons as N_t. This produces a situation where the numerical simulations can only focus on certain scales of interest are at a time. To add to this difficulty, it is very common to consider scales larger than the scales of interest as constant, while behavior on scales smaller than the scales of interest are treated as stochastic.

With such complexity and unpredictability in many of these systems, there is always a need to find consistent properties of fluids, properties that are resilient to perturbations, and

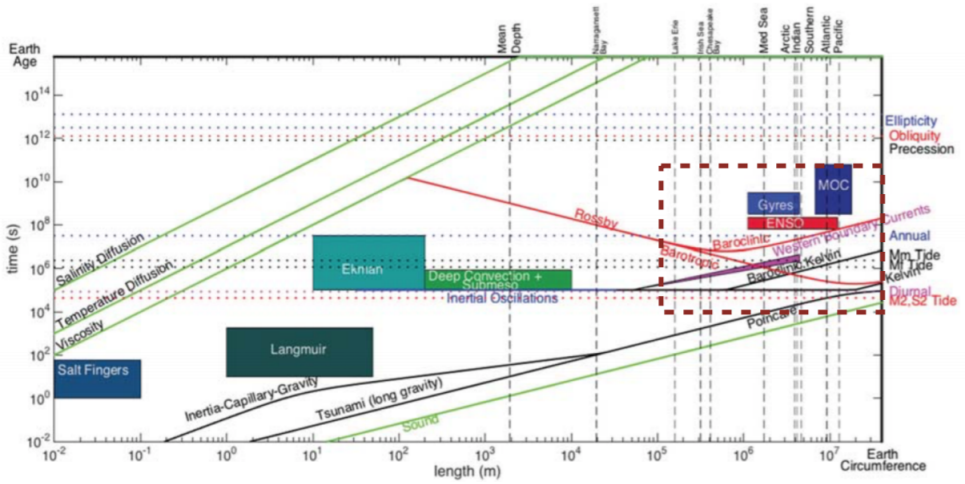

Figure 1.1 The spatial and temporal regimes are shown for important features of ocean dynamics. The dashed rectangle indicates the scales resolvable by state-of-the-art computers.[2]

properties that will persist over long timescales. Topology can help.

1.2 An Introduction to Topology

Topology is a sub-field of geometry that deals with global properties of a system rather than local ones. More precisely, it deals with the properties of functions and spaces that cannot be changed by invertible, continuous transformations.

One classic example of topological behavior is the so-called 'hairy-sphere' theorem. It states that there cannot exist a continuous, non-zero vector field tangent to a sphere. Intuitively one can think of this vector field like hair on the sphere, which must all be combed so that it stays tangent. No 'tufts,' spots where the hair is not tangent are allowed, nor are 'whorls' or 'cowlicks,' spots where the hair rotates around an a point, as these points are discontinuities in the vector field. More formally, these points, and points like them will be referred to as singularities. The 'cowlicks' that emerge when attempting to place a continuous vector field tangent to the sphere are shown on the right of Figure 1.2.

One can imagine warping the sphere to form a stick or thick disc, and this still would not

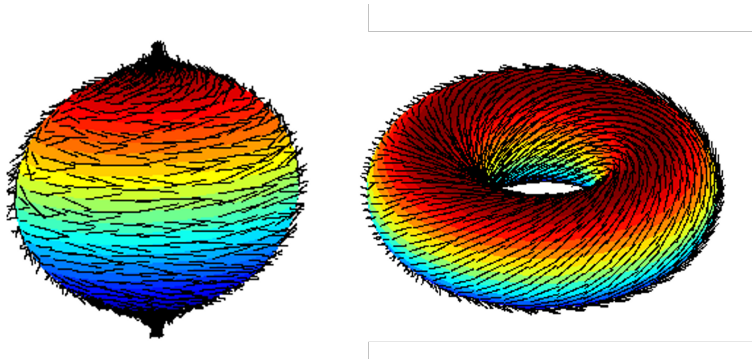

Figure 1.2 On the left, a failed attempt to place a continuous non-vanishing vector field tangent to a sphere, which results in two 'cowlicks,' points where the non-zero vector field cannot be both tangent and continuous. On the right, a continuous non-vanishing vector field is shown tangent to a torus, with no 'cowlicks'.

be possible. Yet, it is completely possible to have a continuous nonzero vector field on the plane, or tangent to a smooth warped infinite surface. It is also perfectly possible to have a vector field like this on a torus, without any of these issues, as can be seen in the left of Figure 1.2. One can imagine warping and stretching this torus, and a behaving vector-field could remain. However, one cannot warp and stretch the circle to make the torus without ripping a hole through it, a discontinuous transformation. So the properties inside a space, like the vector fields permitted is dependent on the 'general' structure of a space. Topology seeks to understand what this 'general' structure means, and how overall structure permits internal behavior.

Concepts from topology have spurred major breakthroughs in condensed matter physics since the 1970s[3][4]. In the case of a on nonzero vector field, we can predict whether or not we will see singularities based on the object the vector field lives on. In the same way, if we know something about the interior behavior of material, with some knowledge of its overall structure we can predict certain features must arrive.

One example of this is the Quantum Hall effect in two-dimensional electron gasses, and the behavior of their more advanced cousins, topological insulators. In these systems, some quantum-mechanical behaviors are robust to noise and disorder. Like in the 'hairy-sphere'

picture, they require a fundamental change in the system's topological structure to disappear. This allows for these quantum behaviors to survive the transition to observable scales.

In more recent work it has been shown that very similar properties emerge in a geophysical context[1], and can provide novel explanations for well-studied phenomena. First, a sketch of how topology plays a role in some well known condensed matter systems will be presented. This is followed by an explanation of how similar properties emerge in a geophysical system. A comparison of features between the two will be made, and following that comparison, the motivation and goals of this thesis will be presented.

1.3 The Integer Quantum Hall Effect and Topological Insulators

The integer quantum Hall effect arises in two-dimensional systems of electrons when a strong magnetic field is applied. First published in 1980 by von Klitzing et al. [5], it was observed that under high magnetic fields conductance can become quantized, and this quantization could be exploited to better measure fundamental constants. The description of this phenomenon presented here follows closely with *Modern Condensed Matter Physics* by Girvin and Yang [4].

In these high magnetic field systems, electrons move in cyclotron orbits, and an electric field applied in one direction leads to current flowing perpendicular to it, as the magnetic field causes the electrons' paths to deflect. In a classical treatment of this problem, defining resistivity ρ and conductance σ by a relation of current J to electric field E,

$$J_i = \sigma_{ij} E_j$$

$$E_i = \rho_{ij} J_j$$

one can use Drude theory to find a cross-resistivity $\rho_{xy} = \frac{-B}{nec}$, with the speed of light c, magnetic field $\mathbf{B} = B\hat{z}$, charge of an electron $-e$ and a density of particles n.

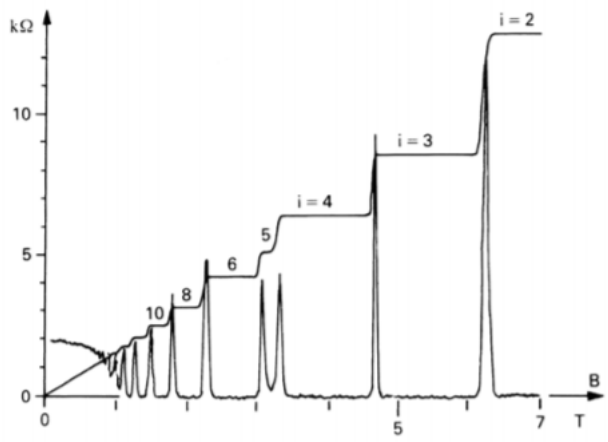

Figure 1.3 Hall resistivity ρ_{xy} shown plateauing at allowed steps of allowed values of Equation 1.1. The spikes occur in standard resistivity ρ_{xx}, when ρ_{xy} transitions from one plateau to another. Behavior like this was first seen by von Klitzing, using samples prepared by Dorda and Pepper[5]

The 'integer' Quantum Hall effect refers to how with a two-dimensional electron system and large magnetic fields, at certain values of B this resistivity changes in steps of:

$$\rho_{xy} = \frac{1}{\nu}\frac{h}{e^2} \tag{1.1}$$

with Planck's constant h and ν an integer quantum number. This resistivity can be measured from edge currents, which circle around the border of the 2D system. A thorough derivation of these features will not be presented here. Instead, the key steps and features will be outlined, and then framed in a topological-physics language. This Hall-resistivity $\rho_x y$ changes plateau, and standard resistivity $\rho_x x$ spikes when the number of electrons within an area is equal to the number of 'magnetic flux quanta' in that area ($\frac{hc}{e}$). These spikes and plateaus can be seen in figure 1.3. This can be understood in part by forcing the classical Hall-resistivity to appear like the quantum Hall-resistivity. Here $N_e = nA$ can represents the number of electrons in some area A, leading to magnetic flux $\Phi = BA$.

$$\rho_{xy} = \frac{-B}{nec} = \frac{h}{e^2}\frac{BA}{hc/e}\frac{1}{N_e} = \frac{h}{e^2}\frac{\frac{BA}{hc/e}}{N_e} = \frac{h}{e^2}\frac{\frac{\Phi}{\Phi_0}}{N_e}$$

$\Phi_0 = hc/e$, takes on units of magnetic flux, and can be shown to represent a minimum

'magnetic flux quantum.' In this context, it can be seen that the quantum resistivity arises from the classical resistivity when the number the number of electrons N_e in an area is some integer multiple of the number of magnetic flux quanta $\frac{\Phi}{\Phi_0} = \frac{BA}{hc/e}$ in that area.

To see the topological origin of this behavior, the quantum-mechanical problem of electrons in cyclotron orbits can be examined via the Hamiltonian:

$$H = \frac{1}{2m_e}(p_x^2 + (p_y - \frac{eB}{c}x)^2)$$

in the Landau Gauge $\mathbf{A}(\mathbf{r}) = -xB\hat{y}$. Here, an external electric field is not included, but will be treated shortly after. Observing in the p_y momentum basis $p_y \to \hbar k$, this becomes a harmonic oscillator problem for p_x and x:

$$h_k = \frac{p_x^2}{2m_e} + \frac{m_e\omega_c^2}{2}(x - kl^2)^2$$

where $\omega_c = \frac{eB}{m_e c}$ is the classical cyclotron frequency, and $kl^2 = k\frac{\hbar c}{eB}$ represents a displacement in x, in terms of linear and angular momentum quantum numbers. Treatment of this harmonic oscillator yields an energy independent of k or l, and wavefunctions based on the nth Hermite polynomials $H_n(x)$:

$$\epsilon_{kn} = (n + \frac{1}{2})\hbar\omega_c \tag{1.2}$$

$$\psi_{nk} = e^{iky}H_n(x/l - kl)e^{\frac{-(x-kl^2)^2}{2l^2}} \tag{1.3}$$

These energy levels are often called Landau levels, and the wavefunctions have an interesting property of grouping around a point kl^2. We should note that there is no way to impose periodic boundary conditions, as the vector potential is very different on each side of the sample. Therefore the sample's dimensions, which will be assumed to be rectangular $L_x \times L_y$, must be included in this analysis, and the system cannot be split into 'unit cells' as is typical in solid-state physics, since the vector potential varies across the sample. By integration over all $\mathbf{k}$, it can be shown that the total number of states available in each Landau level is:

$$N = \frac{L_x L_y}{2\pi l^2} = \frac{BL_x L_y}{\Phi_0},\ \Phi_0 = 2\pi l^2 B = \frac{hc}{e}$$

It is clear that the numerator $L_x L_y B$ can be identified with the total flux through the sample. Noting that $X_k = kl^2 = k\frac{\hbar c}{eB}$ represents a length scale, the denominator Φ_0 can be seen to be the magnetic-flux quantum described earlier: $\frac{hc}{e}$, totally independent of B or k.

We get the result that the number of states available in each Landau level is equal to the number of flux quanta through the system. There are gaps of size $\hbar\omega_c$, between each Landau level, and there is a very large number of degenerate states in each. At some set B, one can imagine filling a two-dimension electron gas with electrons at low temperature. As each electron is added, the energy per new electron is unchanging, assuming the Landau level is not full. Once the energy level does become full, with no capacity for new electrons, the energy per new electron jumps, beginning to fill the next Landau level. The capacity for states in each level is proportional to B, as is the distance between each level.

In Figure 1.3, we can see how this manifests in a material. It is somewhat clearer to start with large B and then show what happens as B decreases, focusing on ρ_{xx} first. We can imagine some B field large enough so that the lowest Landau level is entirely filled, with no others filled. An added electric field shifts energy in the system, preferential to one direction of k, but as there must be an equal number of states in $\pm k$, the resistivity is very high, and no current can flow. Once B decreases, some electrons must jump to the next Landau level, and are free to fill k more than $-k$ or vice versa. This resistivity remains very low until the second level is filled, when the electrons cannot prefer $\pm k$ and resistivity spikes. So when the number of electrons is equal to some integer times the number of states available in a Landau level, or the number of magnetic flux quanta in the system, there is a resistivity spike for ρ_{xx}.

For ρ_{xy}, we need to understand what happens at the edges. In a classical picture, we can imagine these electrons in cyclotron orbits, as can be seen in Figure 1.4. As the electrons cannot complete full cyclotron orbits at the edges, they bound off and 'skip' along these edges, forming currents that rotate around the material.

In the quantum context the same principle creates edge currents. With an electric field

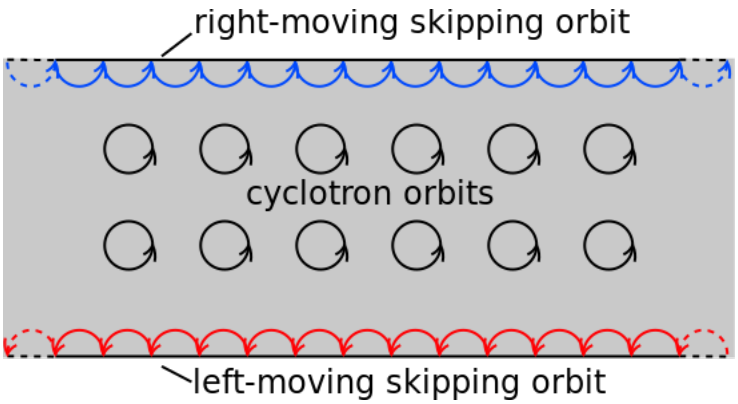

Figure 1.4 A classical picture of edge states arising in a two-dimensional electron gas confined to some region. Cyclotron orbits cannot be completed on the edges, and so the electrons bounce off of the edges, and skip along them, creating edge currents around the material. Image from [6].

E present, the energy is amended:

$$\epsilon_{kn} = (n + \frac{1}{2})\hbar\omega_c + eE(X_k - \frac{eE}{m_e\omega_c^2}) + \frac{1}{2}m_e(\frac{cE}{B})^2 \tag{1.4}$$

Here, $\frac{cE}{B}$ can be identified as the drift velocity $\bar{\boldsymbol{v}}$. With an electric field, k does have an effect on the energy, through the second term eEX_k. This can be seen in Figure 1.5, where these Landau levels are shown for a system like that in Figure 1.4. The edge states, must have large k. Similar to a particle-in-a-box scenario, zero-valued wavefunctions past the edges imply the only wavefunctions significant at the edge have large k. These edge states can move continuously between Landau levels.

It can be also found that the drift velocity is the group velocity of the waves:

$$\bar{\boldsymbol{v}} = \frac{1}{\hbar}\frac{\partial\epsilon_{kn}}{\partial k}$$

From this one can find a net current (of one side of the sample minus another) by integrating all of these group velocities over k, assuming all fit in the lowest Landau level ($n = 0$):

$$\Delta I = \frac{-e}{L_y}\int_{-\infty}^{\infty}\frac{L_y}{2\pi\hbar}\frac{\partial\epsilon_{k0}}{\partial k}n_{k0}dk = \frac{-e}{h}\int_{\mu_R}^{\mu_L}d\epsilon$$

where μ_R and μ_L are the chemical potentials on the left and right. n_{k0} is the probability

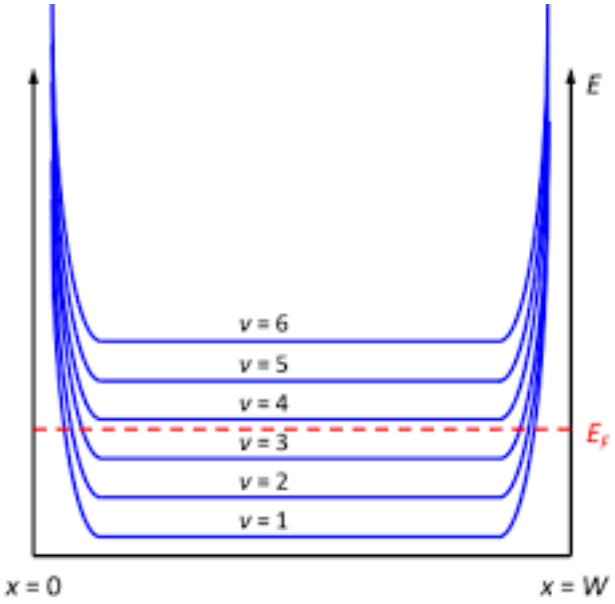

Figure 1.5 Landau levels of 'Hall Bar', a long bar with electric field perpendicular to the long-direction.[7]

that a state with wave-number k in the lowest Landau level is occupied. This gives:

$$\Delta I = \frac{e^2}{h}(V_R - V_L)$$

However, it should be noted that the electric field shifted energy (Equation 1.4), does not have terms that depend on both n and k. Therefore $\frac{\partial \epsilon_{kn}}{\partial k}$ is the same for all n. n_{kn} may be slightly different than n_{k0}, but assuming they are both below the Fermi level when $k = 0$, the edge state behavior is essentially the same, as can be seen in Figure 1.5. One can therefore do this same analysis at any n below the Fermi level. If there are ν Landau levels below the Fermi level, the current difference becomes:

$$\Delta I = \nu \frac{e^2}{h}(V_R - V_L) \implies \rho_{xy} = \frac{\frac{\Phi}{\Phi_0}}{N_e}\frac{h}{e^2} = \frac{h}{e^2} = \frac{1}{\nu}\frac{h}{e^2}$$

This current difference, coupled with precise knowledge of the speed of light, has been used to measure the fine structure constant $\alpha = \frac{e^2}{\hbar c}$ to very high degrees of precision. It is the fact that this current is robust to deformations in the solid that makes this so accurate. If a solid has impurities in the bulk, these rotating currents go around them, and the behavior does not change. In some sense, topology was already shown to be a key feature of this system.

The existence of a boundary makes internal behavior, cyclotron orbits, reflect naturally as boundary behavior: edge currents. This 'bulk-boundary correspondence' is one of the key ideas in topological physics. One can imagine how a change in the bulk (to a three dimensional system or surface of a sphere) would not permit edge currents in quite the same way. These ideas can be expressed in more precise topological language.

The Berry connection (for $\mathbf{k}$-space) is an idea that comes out of slowly evolving a wavefunction $|\psi(\mathbf{k})\rangle$ as a Hamiltonian parameterized by $H = H(\mathbf{k})$ is slowly changed through $\mathbf{k}$-space. This does not always apply to evolution through $\mathbf{k}$, it could be some other parameter, but it does in both the geophysical and condensed matter cases presented here.

$$\mathcal{A} = i \langle \psi | \nabla_{\mathbf{k}} | \psi \rangle \tag{1.5}$$

The related Berry curvature, the curl of the Berry connection is defined by:

$$\boldsymbol{\Omega} = \nabla_{\mathbf{k}} \times \mathcal{A} = i \langle \nabla_{\mathbf{k}} \psi | \times | \nabla_{\mathbf{k}} \psi \rangle \tag{1.6}$$

Berry phase around a closed loop C in $\mathbf{k}$ space is defined by

$$\varphi = i \oint_C \langle \psi | \nabla_{\mathbf{k}} | \psi \rangle \cdot d\mathbf{k} \tag{1.7}$$

The major result of analysis of these formulas is that if a state $|\psi(\mathbf{k})\rangle$ evolves slowly around some loop in $\mathbf{k}$-space, the wavefunction picks up a global phase $e^{i\varphi}$. From these, one defines the Chern number $\mathscr{C}$ to capture the cases when that phase is 2π times an integer, as is true for a loop around the Brillouin Zone (the space of all non-degenerate wave-vectors $\mathbf{k}$ for a crystal):

$$\mathscr{C} = \frac{i}{2\pi} \oint_{\text{BZ Boundary}} \langle \psi | \nabla_{\mathbf{k}} | \psi \rangle \cdot d\mathbf{k} = \frac{i}{2\pi} \int_{\text{BZ}} \langle \nabla_{\mathbf{k}} \psi | \times | \nabla_{\mathbf{k}} \psi \rangle \, d^2\mathbf{k} \tag{1.8}$$

These definitions above, provide a framework for discussion the topology of phenomena in Physics, with $\mathscr{C}$ acting as a quantifier of nontrivial topology. It should be noted that the integer nature of this Chern number implies that $|\psi\rangle$ cannot evolve adiabatically into a state with some other associated Chern number, this can be shown via Hermitian conjugation.

The quantum Hall system can also be viewed in this context. The wavefunctions of the system with no electric field, (Equation 1.3) share a strong resemblance to Bloch functions. Bloch-functions are solutions to the case of periodic potential $V(\mathbf{r})$ in the Hamiltonian, as would be the case in a crystal. They take a form $e^{i\mathbf{k}\cdot\mathbf{r}}u_{n\mathbf{k}}(\mathbf{r})$, where $u_{n\mathbf{k}}(\mathbf{r})$ is periodic in the same way as $V(\mathbf{r})$. The two-dimensional electron gas studied before does not a periodic potential, only a vector potential in the Landau gauge. The steps will not be shown here, but it is the case that periodicity and a Brillouin Zone still emerge int the two dimension electron gas, and the Landau level wavefunctions, can be treated very similarly to Bloch functions. The following steps will be somewhat less detailed than those before, as there is considerable work involved between them.

$$\psi_{nk} = e^{iky}H_n(x/l - kl)e^{\frac{-(x-kl^2)^2}{2l^2}} \rightarrow e^{i\mathbf{k}\cdot\mathbf{r}}u_{n\mathbf{k}}(\mathbf{r})$$

For the bands in the quantum Hall system one ban write down a Berry curvature of $u_{n\mathbf{k}}(\mathbf{r})$:

$$\mathbf{b}_n(\mathbf{k}) = i\left\langle \nabla_{\mathbf{k}}u_{n\mathbf{k}}\right| \times \left|\nabla_{\mathbf{k}}u_{n\mathbf{k}}\right\rangle$$

It should be noted that this $\mathbf{b}_n(\mathbf{k})$ has similar properties to $\mathbf{B}$, in that it is not time-reversal symmetric. Instead, it is invariant under a Parity-Time (PT) transformation, as will be described later. It is very important to note that if $\mathbf{b}_n(\mathbf{k})$ is both time-symmetry and parity-symmetric, for any system it must be zero. There is a very strong connection between nontrivial topology in physics, and the breaking of some symmetry: charge, time or parity.

Applying the eigenfunctions of the quantum Hall system, it can be seen that each n corresponds to a distinct Chern Number. From this, we have the idea that some $\psi_{n\mathbf{k}}(\mathbf{r})$ cannot evolve into another $\psi_{n'\mathbf{k}}(\mathbf{r})$ via a change in $\mathbf{k}$ around the Brillouin zone boundary. Though the derivation will not be shown here, it can shown that for each band the Hall conductance σ^n_{xy}, is directly related to the Chern number:

$$\sigma^n_{xy} = \frac{e^2}{A\hbar}\sum_{\mathbf{k}\text{ in first BZ}}\mathbf{b}_n(\mathbf{k}) = \frac{e^2}{h}\mathscr{C}_n$$

This gives a good explanation for the plateaus in Hall conductance. Each band adds a set amount of conductance, separated from the other bands. As wavefunctions of each band cannot change into each other via an evolution of $\mathbf{k}$, these bands are particularly stable and robust to perturbations like a impure sample.

Before turning to geophysics it is worth paying some attention to topological insulators: a step up in complexity from two-dimensional electron gasses, but a step closer to the geophysical case presented. In a topological insulator there is no magnetic field, and the bulk is nonconducting. What is notable is that spin-orbit coupling takes over the role of a magnetic field (Note the similarity between energy $E \sim \mathbf{B}\cdot\mathbf{L}$ and $E \sim \mathbf{S}\cdot\mathbf{L}$, where $\mathbf{B}, \mathbf{L}, \mathbf{S}$ are magnetic field, orbital angular momentum and electron spin respectively). Electrons therefore do have cyclotron-like orbits, though not all in the same direction, as some have positive, and some have negative spin. Focusing on the case of two-dimensions, as this has the most similarity to the relevant geophysical systems, one can think of a topological insulator as two discs, each with opposite magnetic fields $\mathbf{B} = \pm B\hat{z}$, that meet at the edges. The middle is also be stitched together, but the middle is not where interesting behavior exists. At these edges, where spin-orbit coupling is strong enough to permit edge-currants, there is a simultaneous flow of spin-up and spin-down electrons in opposite directions along the edge.

In a typical insulator, available states are usually characterized in terms of a conduction band, a regime where the electrons have enough energy to escape orbitals, and a valence band, where they are bound to atoms. A large 'band-gap' between these is characteristic of an insulator, while overlap of the bands is characteristic of a conductor. For these edge states, there can be a shift to $\pm k$ can be brought on by an electric field, and edges can conduct. This is shown in Figure 1.6, where edge states can move continuously between conduction and valence bands. Though Chern number does not exactly persist in the case of only spin-orbit coupling, a Chern 'parity' ± 1 can be associated with each connecting edge band[3]. It turns out that these properties of topological insulators show a striking similarity to geophysical systems.

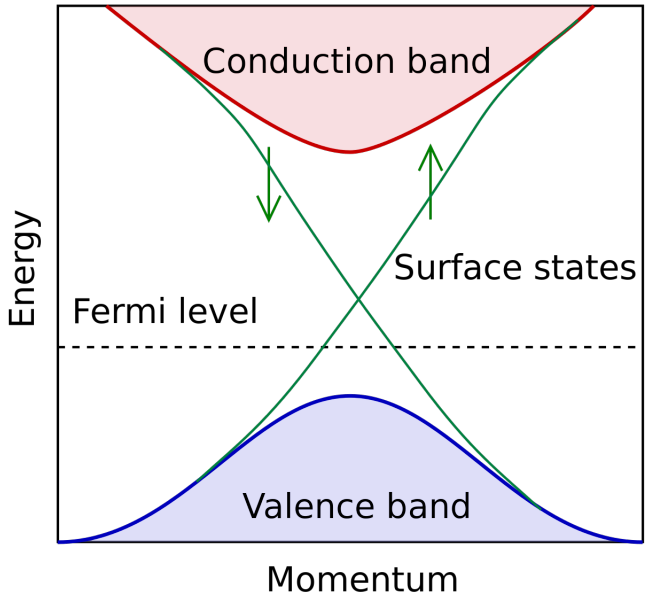

Figure 1.6 Band structure of one type of topological insulator. Though the valence and conduction band are not close to each other, edge states connect between them.

1.4 Topology in Geophysics

In *Topological Origin of Equatorial Waves*[1] written by Pierre Deplace, Antoine Venaille and J.B Marston, it was shown that this topological picture can apply in a geophysical context. At its core, the Coriolis effect is extremely to a magnetic field, so the properties presented above carry over well to a rotating planet. In the reference frame of a person standing on Earth, the Coriolis effect has opposite sign on the two hemispheres. This is the reason why hurricanes circle in different directions on each hemisphere. In this context, the Earth's surface can be though of as two discs, each with opposite Coriolis effect, stitched together at their edges (the Equator). This is very similar to the situation of a topological insulator previously described, though there are some key differences. With the idea that characteristics of the bulk necessitate behavior of boundary, this picture can be used to explain the presence of Equatorial waves that play a large role in the Earth's climate, just as it leads to edge modes in topological insulators.

First, the highly simplified equations that permitted this analysis will be derived, then

their nontrivial topological properties and implications for the Earth's climate will be described.

1.4.1 The Shallow Water Equations

The shallow water equations, more specifically the simplified linear shallow water equations are the simplest model for fluid flow where the Coriolis effect is present. To illustrate where they come from, this section will start with the general fluid momentum conservation equations, characterize the effect of a rotating planet, and make the necessary simplifications to get this highly basic model. This model will be solved, and its features will be investigated and compared to those in topological insulators. The following derivation closely follows that of Vallis (2019)[8].

First the non-rotating fluid momentum conservation equation:

$$\frac{\partial \boldsymbol{v}}{\partial t} + (\boldsymbol{v} \cdot \nabla)\boldsymbol{v} = \frac{-\nabla p}{\rho} + \boldsymbol{F}$$

Here, $\boldsymbol{v} = \boldsymbol{v}(\boldsymbol{x}, t)$ is the velocity of that fluid at a given point in space, while $\rho = \rho(\boldsymbol{x}, t)$ is the corresponding density. Similarly, the system has pressure, $p = p(\boldsymbol{x}, t)$ and sometimes a force (per unit mass) $\boldsymbol{F}$ acting on the whole body of fluid. This is often gravity in geophysical context.

There are some equations missing, as p and ρ are not always constant, and conservation of mass as well as thermodynamic equations are required to fully characterize the fluid's evolution.

It should be noted that a viscous term $\nu\nabla^2\boldsymbol{v}$ is often added, but will be omitted here. The only strange looking term is the advection term $(\boldsymbol{v} \cdot \nabla)\boldsymbol{v}$. This accounts for the change in velocity of the fluid via movement of the fluid itself. For a given direction $\hat{x}_i$, $((\boldsymbol{v} \cdot \nabla)\boldsymbol{v})_i = \sum_j v_j \frac{\partial}{\partial x_j} v_i$. This is the part that makes fluid equations extremely difficult to solve. As this term is nonlinear in $\boldsymbol{v}$, the solutions to these equations cannot be simply broken down into a sum of Fourier modes. In certain regimes, this term leads to turbulent behavior, which

eludes direct solution. Instead, turbulence is often thought of as a random fluctuation in $\boldsymbol{v}$ and incorporated into a diffusion of the whole system. A great deal of work has been put into understanding turbulence, and still there is a great deal unknown. Appropriate treatment of the advection term is, however, not within the scope of this thesis, and this term will be cautiously neglected later. In the main section of this thesis, it will be replaced by a stochastic fluctuation. The effect of a rotating planet on the momentum conservation equation is more the subject of interest for now.

In a rotating reference frame with angular frequency $\boldsymbol{\Omega}$, the Coriolis force $-2\boldsymbol{\Omega} \times \boldsymbol{v}$ and centrifugal force (both per unit mass) $-\boldsymbol{\Omega} \times (\boldsymbol{\Omega} \times \boldsymbol{r})$ are added:

$$\frac{\partial \boldsymbol{v}}{\partial t} + (\boldsymbol{v} \cdot \nabla)\boldsymbol{v} = -2\boldsymbol{\Omega} \times \boldsymbol{v} - \boldsymbol{\Omega} \times (\boldsymbol{\Omega} \times \boldsymbol{r}) + \frac{-\nabla p}{\rho} + \boldsymbol{F}$$

It is worth noting that the Coriolis force $-2\boldsymbol{\Omega} \times \boldsymbol{v}$ is qualitatively identical to the force from a magnetic field: $-q\boldsymbol{B} \times \boldsymbol{v}$. The Coriolis force also shared PT symmetry with $\mathbf{B}$, but they both lack T or P symmetry. This parity-time symmetry can be thought of directly in terms of a rotating planet. If time is reversed, the planet spins in the opposite direction, and the behavior fundamentally changes. If the planet is reflected in a mirror (a parity transformation), the rotation also reverses. Putting these together, a mirror of the Earth with time moving backwards behaves just like the normal Earth. Therefore, we have reason to believe the Berry curvature of systems with the Coriolis effect could be non-zero too.

What will follow is a series of approximations to these equations, whittling them down to something that can be solved and studied directly in a stochastic context. It is worth noting that many of these approximations eliminate behaviors that are very important in the atmosphere and oceans. With some approximations more reductive than others, they are applied with the intention of producing simple enough solvable equation. This does not necessarily imply the eliminated terms have an insignificant effect on the behavior of geophysical systems. Describing to which regimes these approximations accurately apply, and which behaviors manifest in those regimes, constitutes an entire field of study.

To begin reducing these equations to the shallow water model, the centrifugal force will first be eliminated. The centrifugal force is generally less interesting than the Coriolis force. For one, it is far smaller than gravity on Earth, and since it can also we written as the gradient of some scalar potential [1] it is often just considered a small correction to the gravitational force. We'll simply let $\boldsymbol{g}$ be gravity and ignore the centrifugal effect. For a fluid like the ocean or atmosphere, we have:

$$\frac{\partial \boldsymbol{v}}{\partial t} + (\boldsymbol{v} \cdot \nabla)\boldsymbol{v} = -2\boldsymbol{\Omega} \times \boldsymbol{v} + \frac{-\nabla p}{\rho} + \boldsymbol{g}$$

The convention is to use the coordinate system where $\hat{z}$ points outward from the center of the Earth and $\hat{x}$ and $\hat{y}$ are tangent to the Earth's surface, with $\hat{x}$ pointing east-west and $\hat{y}$ pointing north-south. In this coordinate system, we have $\boldsymbol{\Omega} = \Omega\cos(\theta)\hat{y} + \Omega\sin(\theta)\hat{z}$, where θ is the polar angle. Often the $\Omega\cos(\theta)\hat{y}$ is omitted. This is called the 'Traditional Approximation,' and can be traced back to Laplace. It stems from the fact that the atmosphere and ocean form a very thin shell around the earth. While they are at most on the order of 10km in depth, the Earth has a circumference of around $40,000$km. The typical convention is that $\boldsymbol{v} = u\hat{x} + v\hat{y} + w\hat{z}$. The Traditional Approximation is taken to imply that w is much smaller in magnitude than u or v, suggesting the reduced importance of $\Omega\cos(\theta)\hat{y}$. Incorporating this into the previous equation, we can look at this equation by components.

$$\begin{aligned}
\frac{\partial u}{\partial t} + (\boldsymbol{v} \cdot \nabla)u - 2\Omega\sin(\theta)v &= \frac{-1}{\rho}\frac{\partial p}{\partial x} \\
\frac{\partial v}{\partial t} + (\boldsymbol{v} \cdot \nabla)v + 2\Omega\sin(\theta)u &= \frac{-1}{\rho}\frac{\partial p}{\partial y} \\
\frac{\partial w}{\partial t} + (\boldsymbol{v} \cdot \nabla)w &= \frac{-1}{\rho}\frac{\partial p}{\partial z} - g
\end{aligned}$$

From here, an approximation can be made to treat flows as existing only in an area where θ is nearly constant with respect to y. This reduces the scope of these equations to only flows that exist in regions around a certain latitude. Large scale flows from north to south

[1] $-\boldsymbol{\Omega} \times (\boldsymbol{\Omega} \times \boldsymbol{r}) = \Omega^2 \boldsymbol{r}_\perp$, where $\boldsymbol{r}_\perp$, is the displacement from to the axis of rotation. So $-\boldsymbol{\Omega} \times (\boldsymbol{\Omega} \times \boldsymbol{r}) = \nabla \frac{1}{2}\Omega^2 r_\perp^2$.

are no longer modeled. Typically, this is represented by the first two terms in the Taylor series $2\Omega\sin(\theta) \approx 2\Omega\sin(\theta_0) + 2\Omega(\theta - \theta_0)\cos(\theta_0)$. f is defined to equal $2\Omega\sin(\theta)$, and these first to two terms are written as $f \approx f_0 + \beta y$. This is the so-called β-plane approximation, while $f = f_0$ is called the f-plane approximation.

The shallow water model is intended to describe the motion of a fluid at constant density ρ_0. This implies that the fluid is incompressible: $\nabla \cdot \boldsymbol{v} = 0$, and also implies that the advection terms can be somewhat simplified:

$$(\boldsymbol{v} \cdot \nabla)v_i = \nabla \cdot (\boldsymbol{v}v_i)$$

This fluid exists above some surface, which in this case will be assumed to be flat, and has a height above that surface $h(x, y, t)$. The shallowness of the water, comes from the stipulation that the waves of interest have a wavelength far greater than h.

The shallow water equations do not directly consider $\boldsymbol{v}(x, y, z, t)$. Rather, they treat the velocities averaged over the height of the fluid:

$$\boldsymbol{v}(x, y, t) = \frac{1}{h(x, y, t)} \int_0^{h(x,y,t)} \boldsymbol{v}(x, y, z, t)dz$$

Since each component is uniform along z, we can identify $w(x, y, t) = \frac{\partial h(x,y,t)}{\partial t}$. For the following analysis $\boldsymbol{u} = u\hat{x} + v\hat{y}$ will refer to the horizontal velocity vector, as the vertical velocity $w\hat{z}$ is averaged and substituted out.

The shallow water model also that the system satisfies the condition called hydrostatic balance:

$$\frac{\partial p}{\partial z} = -\rho_0 g \tag{1.9}$$

This is simply the assumption that the only contribution to pressure at a point comes from the fluid piled above it, not fluid pushing on its sides. We can use this to simplify the evolution of w:

$$\frac{\partial w}{\partial t} + (\boldsymbol{v} \cdot \nabla)w = 0$$

Integrating this equation over time produces an equation in terms of h:

$$\frac{\partial h}{\partial t} + \nabla \cdot (h\boldsymbol{u}) = 0$$

This hydrostatic simplification also allows for a straight-forward expression for pressure:

$$p(x, y, z, t) = \rho_0 g(h(x, y, t) - z)$$

From this, it can be seen that $\frac{\partial p(x,y,z,t)}{\partial z}$ is in dependant of height or z. With vertical gradients of pressure taken care of, we can define the horizontal gradient (at constant z) $\nabla_z = \hat{x}\partial_x + \hat{y}\partial_y$. From this, one constructs the standard shallow water equations:

$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla_z)\boldsymbol{u} + \boldsymbol{f} \times \boldsymbol{u} = -g\nabla_z h$$
$$\frac{\partial h}{\partial t} + \nabla_z \cdot (h\boldsymbol{u}) = 0$$

To obtain the linear shallow water equations, one can linearize u, v and h about some mean values: $u = U + u', v = V + v', h = h' + H$, where $u' \ll U, v' \ll V, h' \ll H$. From there, an term quadratic in the small perturbations can be eliminated. Adding in the f-plane approximation, this gives the linear shallow water equations:

$$\frac{\partial u'}{\partial t} - f_0 v' = -g\frac{\partial h'}{\partial x}$$
$$\frac{\partial v'}{\partial t} + f_0 u' = -g\frac{\partial h'}{\partial y}$$
$$\frac{\partial h'}{\partial t} + H(\frac{\partial u'}{\partial x} + \frac{\partial u'}{\partial x}) = 0$$

One last step is used for simplicity: non-dimensionalizing. If we allow u', v' and h' to lose dimensions, we can express these dynamics in a simpler form. Also, equally distributed drag (with drag constant θ) is added.

$$\frac{\partial h}{\partial t} = -c\frac{\partial u}{\partial x} - c\frac{\partial v}{\partial y} - \theta h \tag{1.10}$$
$$\frac{\partial u}{\partial t} = f_0 v - c\frac{\partial h}{\partial x} - \theta u$$
$$\frac{\partial v}{\partial t} = -f_0 u - c\frac{\partial h}{\partial y} - \theta v$$

where $c = \sqrt{gH}$. These are the main equations that will be investigated: the f-plane linear shallow water equations with added drag.[2]

To solve these equations, it is necessary to Fourier transform in k_x and k_y.

$$\frac{\partial}{\partial t}\begin{bmatrix} h \\ u \\ v \end{bmatrix} = L \begin{bmatrix} h \\ u \\ v \end{bmatrix}$$

$$L = \begin{bmatrix} -\theta & -ick_x & -ick_y \\ -ick_x & -\theta & f \\ -ick_y & -f & -\theta \end{bmatrix} \tag{1.11}$$

where the subscript of f_0 is dropped. From here, we can diagonalize L and Fourier transform $t \to \omega$ to obtain:

$$\begin{bmatrix} -\theta - i\sqrt{ck_x^2 + ck_y^2 + f^2} & 0 & 0 \\ 0 & -\theta & 0 \\ 0 & 0 & -\theta + i\sqrt{ck_x^2 + ck_y^2 + f^2} \end{bmatrix}, \tag{1.12}$$

with corresponding eigenvectors:

$$|\omega_+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{ck}{\sqrt{c^2k^2+f^2}} \\ \frac{k_x}{k} - \frac{ifk_y}{k\sqrt{c^2k^2+f^2}} \\ \frac{k_y}{k} + \frac{ifk_x}{k\sqrt{c^2k^2+f^2}} \end{bmatrix}, |\omega_0\rangle = \begin{bmatrix} \frac{-if}{\sqrt{c^2k^2+f^2}} \\ \frac{ck_y}{\sqrt{c^2k^2+f^2}} \\ \frac{-ck_x}{\sqrt{c^2k^2+f^2}} \end{bmatrix}$$

$$|\omega_-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{-ck}{\sqrt{c^2k^2+f^2}} \\ \frac{k_x}{k} + \frac{ifk_y}{k\sqrt{c^2k^2+f^2}} \\ \frac{k_y}{k} - \frac{ifk_x}{k\sqrt{c^2k^2+f^2}} \end{bmatrix} \tag{1.13}$$

[2]It is worth noting that qualitatively identical equations can be derived for a two dimensional plane of electrons with a constant magnetic field, using Drude theory: $m\frac{d\boldsymbol{v}}{dt} = -e\mathbf{E} - e\boldsymbol{v} \times \boldsymbol{B} - \frac{m\boldsymbol{v}}{\tau}$, where the velocity of an electron $\boldsymbol{v}$ is related to current by $\mathbf{J} = \rho\boldsymbol{v}$, where ρ is electron density. With the addition of $\mathbf{E} = \epsilon_0 \nabla \cdot \rho$, $\frac{d\rho}{dt} = -\nabla \cdot \mathbf{J}$, and linearization about some mean value, the resulting equations capture identical dynamics.

ω_+ and ω_- are known to represent Poincairé waves, or inertia-gravity waves. ω_0, represents a standard gravity wave, which would appear without f. This will often be referred to as the trivial solution.

Before considering Berry phase, connection and curvature, it is worth pointing out some properties of these eigenvectors. In $|\omega_+\rangle$, there exist phase singularities in the u and v components when $f \approx \sqrt{c^2k^2 + f^2}$, or when ck_x and ck_y are small. These singularities take a form like $\frac{k_x \pm ik_y}{k}$ This phase singularity, like a 'cowlick' in the 'hairy-sphere' theorem, is a strong sign of topological behavior.

Like in the case of a conduction and valence band, as can be seen in Figure 1.7, showing the solutions in k_x, k_y, ω space, there is a gap between nontrivial solutions $\omega_\pm = \pm\sqrt{c^2k^2 + f^2}$, where ω_0 represents the plane inbetween. It can be shown, using the formulas presented above, that $|\omega_\pm\rangle$ has a nontrivial Berry connection and curvature: $B_\pm = \pm f c^2 (f^2 + c^2k^2)^{-3/2}$. This Berry Curvature can be integrated over k to show that these represent nontrivial Chern number solutions. Each nontrivial solution $\omega_\pm$ can be associated with Chern number $\mathscr{C}_\pm = \pm sign(f)$, or a change in Chern number over hemispheres: $\{\Delta\mathscr{C}_-.\Delta\mathscr{C}_0, \Delta\mathscr{C}_+\} = \{-2, 0, 2\}$.

Taking a step back, we can imagine the Coriolis effect on each hemisphere's effect on the Equator. On each hemisphere, the atmosphere is forced to rotate in a given direction. Along the Equator, where $f = 0$, this characteristic rotation is no longer permitted, and instead the waves are trapped at the boundary, can be imagined as skipping along the equator, just like in the quantum Hall picture. Unlike the topological insulator picture, these waves do not skip in different directions, as the equator is not an edge, and is instead a plane. These skipping waves all propagate west to east. The eastward propagating waves, the Kelvin and Yanai waves, have been known to geophysics for some time though a topological description of their origin is new.

Even more, these equatorial bands connect the Poincairé bands in Figure 1.7, just like in the case of a topological insulator (Figure 1.6). These equatorial waves are well known in geophysics, only a topological explanation is new. These waves and their implications will

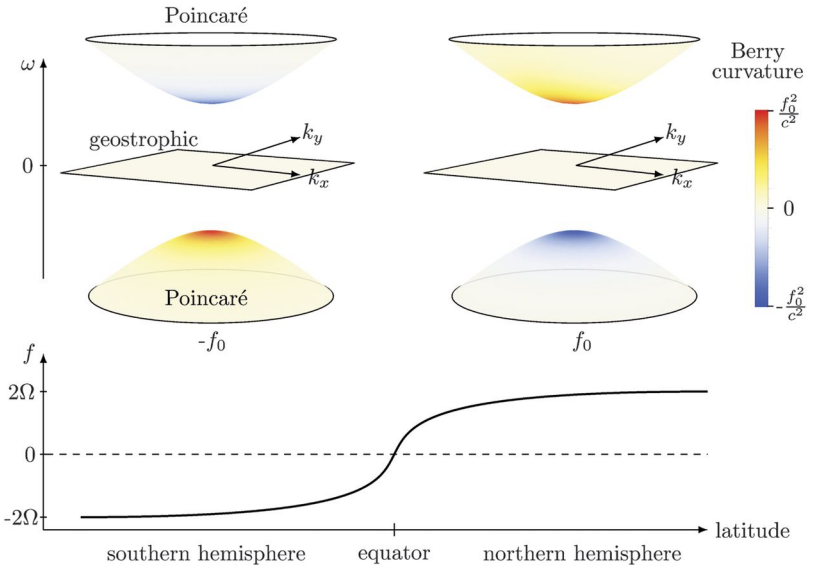

Figure 1.7 The color indicates the Berry curvature $B_n \equiv -i\nabla_p \times (\Psi_n^\dagger \nabla_p \Psi_n)$ for each wave band indexed by $n \in \{\breve{}, 0, +\}$. The Berry curvature of the Poincaré bands is $B_\pm = \pm f c^2 (f^2 + c^2 k^2)^{-3/2}$. It is concentrated around $k = 0$, with extremal value $\pm c^2/f^2$, and switches sign with f. The curvature vanishes for the geostrophic band. When integrated over the whole plane (k_x, k_y), the Berry fluxes in the three bands give integers $(\breve{}1, 0, 1)$ for $f > 0$ and $(1, 0, \breve{}1)$ for $f < 0$, consistent with the triplet of Chern numbers $\{\Delta\mathscr{C}_-.\Delta\mathscr{C}_0, \Delta\mathscr{C}_+\} = \{-2, 0, 2\}$. This shows that the set of delocalized bulk Poincaré modes cannot be continuously deformed from one hemisphere to another. Caption and Figure from [1]

be briefly discussed, before outlining the main goals of this thesis.

Climate Effects of Topologically Protected Waves

El Ninõ is one of the best known events in climatology, and is the most significant source of sea-surface temperature variability on a more-than-yearly timescale. Its episodic warming, and the coupled atmospheric Southern Oscillation climate events' heavy rains have a major impact on human activity. El Niño events are a warming events in Pacific sea surface temperature, that last an number of months, peaking in December, and are spaced by irregular intervals of 3 to 7 years. The mechanism behind ENSO is a feedback-loop, called the Bjerkes feedback, where west-ward propagating trade winds pull warming surface water with them, which is then pulled underneath the surface in the west pacific, to propagate warm water back east. The temperature gradient cause by the warm water accumulating on the western pacific increases the trade winds, which makes more warm water accumulate and so on....

In an ENSO event the west-moving trade winds are weakened, and this cycle is disrupted. When this happens, warming water on the sea surface is no longer universally pulled west, and the eastern Pacific region warms. This disruption of trade winds is usually due to a Kelvin wave, an equatorial wave moving west to east derivable using the β-plane approximation. This Kelvin wave, can also be interpreted as the necessary edge mode between positive and negative f-planes above and below the equator. The Bjerknes feedback, amplifies this Kelvin wave if it is strong enough, and an anomalous sea surface temperature persists.

1.5 Goals

The long-term goal of the work presented in this thesis is to detect these topological properties in real atmospheric data. Though Berry curvature, phase and Chern number could theoretically be observed, the necessary signature for this topological behavior is phase sin-

gularities, in the date, specifically those of the type $\frac{k_x \pm k_y}{k}$ as can be seen in the nontrivial solutions to the linear shallow water equations. From these, nontrivial berry curvature, phase etc. follow easily. Since climate data can be exceptionally noisy, some averaging is necessary to see these properties.

The central hypothesis was that topological behavior could be found in *unequal*-time field correlations, like $E[u(k_x, k_y, t)h(k_x, k_y, t + \tau)]$, where E denotes an expected value, of some kind. Seeing as the solutions to the linear shallow model are waves at some frequency ω, it hypothesized that waves could be 'followed' by Fourier transforming a cross-correlation like $E[u(k_x, k_y, t)h(k_x, k_y, t+\tau)]$ over τ, so that the topological signatures of each ω_+, ω_- solution could be observed. Before moving to true atmospheric or oceanic data, it needed to be seen whether these topological signatures truly could be extracted in a theoretical and numerical context. This thesis shows how this is possible in a theoretical context, and verifies results via numerical simulation.

This thesis models all behavior not resolved by the simplified shallow water model as constituting 'random vectors' η, that provide stochastic noise to the system. Taking averages over that noise, Fourier transformed *unequal*-time correlations of fields u, v, h can be extracted via statistical averaging. The following section, introduces the language of this stochastically driven, 'noisy' linear matrix differential equations, solves for unequal-time correlations the case of general linear operators L, then applies the formulas found to the case of the shallow water equations.

Chapter Two

Theory

2.1 Statistical Analysis of General Linear Stochastic Differential Equations

We want to understand statistical properties of linear, differential equations when we add some random fluctuations. Though I reproduced many of these derivations, the only claim of originality I'll make is an application of the resulting formulas to the shallow water equations, though this is not a huge jump from the equations themselves.

The classic example of a stochastic differential equation like this is the Langevin Equation, describing Brownian Motion[9].

$$m\frac{d\mathbf{v}}{dt} = -\lambda\mathbf{v} + \boldsymbol{\eta}(t) \tag{2.1}$$

The Langevin Equation intends to model the motion of a particle, like a bit of dust or pollen, with mass m and velocity $\mathbf{v}$ moving around in a still fluid. The fluid drags on the particle with a strength λ, proportional to the velocity of the particle relative to it. Meanwhile, as this is a tiny particle, we imagine that each atom bouncing off of it slightly pushes the particle on way or another. Rather than agonize over trying to understand the motion of every water molecule, the Langevin equation assumes that the water molecules transfer momentum to the particle randomly with a 'random force' $\boldsymbol{\eta}(t)$ as a function of

time. A variable like this is often called 'noise' or in special cases 'white noise.' A more precise definition of this noise will be given later, but for now a rough sketch will help.

A discrete analog of the random force $\boldsymbol{\eta}(t)$ would be $\boldsymbol{\eta}_{t_n}$, where $t_n = n\Delta t$. At any given t_n the force has its components chosen from a normal distribution, and at any other t_m the choice is completely independent of its previous values. These components have some standard deviation, may have some nonzero mean, and at a given time they may be correlated with each other. With Brownian motion, η_x, η_y and η_z are usually considered independent. The very rough idea of $\boldsymbol{\eta}(t)$ is to simply take the limit of $\boldsymbol{\eta}_{t_n}$ when $\Delta t \to 0$: this limit could be thought of as a function with a new, normally distributed value after every infinitesimal increment dt. This is definitely imprecise, somewhat incorrect, and a little misleading. We could just leave the issues alone until introducing the precise definition, via Wiener processes, later; however, another stipulation for the discrete case will help make the actual definition through Wiener processes feel better justified.

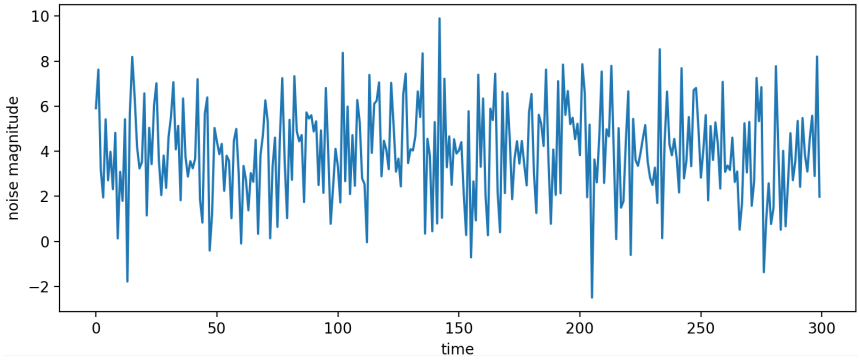

Figure 2.1 Example of a discrete noise η_{t_n} over 300 time-steps with mean = 4 and standard deviation = 2 for each time-step.

The problem with the discrete analogy is that if the standard deviation of $\boldsymbol{\eta}_{t_n}$ is independent of time-step Δt, in any finite time interval $t_f - t_i$, there will be a number of noise steps proportional to $1/\Delta t$. If we were to run this random process from time t_i to t_f over and over, and look at the mean value of all the $\boldsymbol{\eta}_{t_n}$s that contribute in that interval each time we run it, we'd find that the standard deviation of those means decreases as we increase the number of $\boldsymbol{\eta}_{t_n}$ steps in that interval. The average of all the contributions in that time fluctuates less

as more contributions are fit into that time. More simply, if we take more measurements, the error-bars go down. This is a manifestation of the Central Limit theorem, at the core of Statistics.

In this context, in the limit as Δt approaches 0, we have infinitely many contributions from $\boldsymbol{\eta}(t)$ in any finite time interval. This is a major problem: The time-averaged, not just statistical averaged, contribution of the random process has become entirely non-random over any finite time-interval. To combat this issue, before taking the $\Delta t \to 0$ limit, we need the standard deviation at each time-step to grow as Δt shrinks, that it becomes effectively infinite in the limit. Because of this strangely infinite standard deviation, a random force or noise like this is not so well-defined. To avoid these issues, the one-dimensional Langevin Equation is usually replaced with the stochastic differential equation:

$$dX(t) = -\lambda(X(t) - \mu)dt + \sigma dW(t) \tag{2.2}$$

where $X(t)$, representing velocity in Brownian motion, is called the Ornstein-Uhlenbeck process[10]. $W(t)$ is the Wiener process, and the constant σ gives it proper dimensions. The drift velocity μ is often added. Pretending derivatives are fractions and dividing though by dt, we recover something that looks like Equation 2.1, where mass and drift are ignored and we can identify the random force $\eta(t)$ as something like the time derivative of a Wiener process, though this process is not truly differentiable.

With this more exact tool in hand, statistical properties of Brownian motion can be extracted. The statistical mean value $\langle \mathbf{v}(t) \rangle$, as well as variances $\langle v_i(t)^2 \rangle - \langle v_i(t) \rangle^2$ and covariances $\langle v_i(t) v_j(t) \rangle - \langle v_i(t) \rangle \langle v_j(t) \rangle$, can be calculated for this system, as will be shown after important formulas for Wiener processes. Properties like $\langle \text{distance}(t)^2 \rangle = \frac{2k_B T}{\lambda} t$ were famously found by Einstein before the mathematical rigour was established[11], though with an approach more based in thermodynamics an diffusion. This thesis is based off of a derivation of statistical properties like these for the higher-dimensional Ornstein-Uhlenbeck

equation:

$$d\mathbf{X}(t) = -L\mathbf{X}(t)dt + \sigma d\mathbf{W}(t) \tag{2.3}$$

where L, and σ are matrices, and drift will not be considered. σ will often be referred to as the 'noise' matrix.

For the shallow water equations, velocity and height should be functions of x and y, so this really should become a coupled partial Differential Equation. Thankfully, though L contains partial derivatives with respect to position, they are all linear, so everything can be Fourier transformed into k_x and k_y to make L matrix, and wave equations at set k_x and k_y can be solved as ordinary differential equations. As these wave solutions will be complex, I'll adopt Dirac's 'bra-ket' notation [1]

[1]"Bra-Ket" notation, developed by Paul Dirac is in my opinion an extremely clever intuitive way of notating complex vectors and matrices. Dirac's notation would represent column vector of n complex scalars (typically written $\mathbf{x}$, as a so-call 'ket' vector $|x\rangle$. Over the real numbers, the magnitude squared of a vector, or its dot product with itself is usually written as $||\mathbf{x}||^2 = \mathbf{x}\cdot\mathbf{x} = \mathbf{x}^T\mathbf{x}$, where $\mathbf{x}^T$ is the transposed column vector with the same entries as $\mathbf{x} = \sum_i x_i^2$. All of these notations mean the same thing: sum up all the squared entries of $\mathbf{x}$, and their interchangeability may be very helpful (though not so elegant) for learning linear algebra over the reals. Over complex numbers, these notations are not as helpful. We still want a notion of magnitude for complex vectors, so a 'dot product' replacement should still produce a real number. So $\sum_i x_i x_i^*$, where x_i^* is the ith entry of x_i complex conjugated, would work. To notate this, typically, something like $\mathbf{x}^{T*}\mathbf{x}$ would be used, where † can stand in for $T*$. Unfortunately physicists and mathematicians seem to very much disagree on whether a raised bar $\bar{\mathbf{x}}$ or $\mathbf{x}^*$ should mean simple complex conjugation, or conjugate transposition, though † is usually unambiguous (to whom it means anything at all).

Dirac's notation replaces this $\mathbf{x}^{T*}$ with a 'bra' vector $\langle x|$. The inner product (generalized dot product) is therefore represented as a real-valued "braket" $\langle x|x\rangle$. For different vectors in the space, this "braket" $\langle x|y\rangle$ may not be real values, but it has the nice property that the scalar $\langle x|y\rangle$ is the complex conjugate $\langle y|x\rangle$, leaving out the ambiguity of a 'dot product' which is (aggressively) taught as commutative over the reals. One can take the conjugate transpose of a 'ket' to get a 'bra': $|x\rangle^\dagger = \langle x|$ and vice versa. Perhaps most elegantly, outer products ('ket-bras' rather than 'bra-kets') can represent matrices, like $|a\rangle\langle b|$. It can be seen that this can act of a vector $|x\rangle$ to produce a new vector: $|a\rangle\langle b|x\rangle = \langle b|x\rangle|a\rangle$, noting that $\langle b|x\rangle$ is a scalar that commutes with anything. Change of basis can be made far more intuitive and elegant. Suppose

The next section will be dedicated to solving:

$$|dq(t)\rangle = L\,|q(t)\rangle\,dt + \sigma\,|dW(t)\rangle \tag{2.4}$$

specifically for finding an unequal time covariance matrix $E[|q(t+\tau)\rangle\,\langle q(t)|] - E[|q(t+\tau)\rangle]E[\langle q(t)|]$, where $E[X]$ denotes a statistical average of X, to avoid confusion with inner products that may result from the notation $\langle X\rangle$. We'll assume components of the noise are linear combinations of infinitesimal steps of the components of a vector of independent 'source noises' $|W(t)\rangle$. With these conventions established, it is time to confront Wiener processes.

2.1.1 Wiener Processes and Itô Integrals

A Wiener process is characterized by these four attributes[12]:

1. $W(0) = 0$
2. $W(t)$ has independent increments: $W(t) - W(s)$ is independent of $W(t') - W(s')$ if the ranges s to t and s' to t' do not overlap.
3. $W(t) - W(s)$ is chosen from a normal distribution with mean 0 and variance $t - s$.
4. $W(t)$ is continuous, or more precisely, the functions generated by $W(t)$ are continuous.

From this, it is possible to take one stochastic process $X(t)$ and integrate it along a Wiener process $W(t)$: $Y(t) = \int_0^t X(s)dW(s)$. A very important formula, from the Itô Isometry will

there is a vector $|v\rangle = v_x\,|x\rangle + v_y\,|y\rangle$. $|x\rangle$ and $|y\rangle$ represent unit vectors in the x and y directions, where $\langle x|x\rangle = \langle y|y\rangle = 1$, and $\langle y|x\rangle = 0$. If one wishes to transform into a new basis $|x'\rangle, |y'\rangle$, that can be intuitively understood by representing the identity as a sum over basis outer products $I = |x'\rangle\,\langle x'| + |y'\rangle\,\langle y'|$, where each outer product is a projection operator onto that vector. Therefore, we can represent $|v\rangle$ in terms of $|v\rangle = (|x'\rangle\,\langle x'| + |y'\rangle\,\langle y'|)(v_x\,|x\rangle + v_y\,|y\rangle) = (v_x\,\langle x'|x\rangle + v_y\,\langle x'|y\rangle)\,|x'\rangle + (v_x\,\langle y'|x\rangle + v_y\,\langle y'|y\rangle)\,|y'\rangle$. These always generalize very well do inner products that are not necessarily dot-product like, and to 'bra' and 'ket' vectors that are not necessarily finite dimensional. There are many resources online for a more thorough explanation of Dirac's notation

be very useful for extracting properties of these systems[13]:

$$E[\int_0^{t'} X(s)dW(s) \int_0^t Y(s')dW(s')] = E[\int_0^{\min(t',t)} X(s)Y(s)ds] \tag{2.5}$$

where W is a real Wiener process,and $X(s)$ and $Y(s)$ are other stochastic processes. From this, there is the simpler case:

$$E[\int_0^{t'} f(s)dW(s) \int_0^t g(s')dW(s')] = \int_0^{\min(t',t)} f(s)g(s)ds \tag{2.6}$$

This is the formula that following derivations hinge on. Also it is worth noting that:

$$E[\int_0^{t'} f(s)dX(s) \int_0^t g(s')dY(s')] = 0 \tag{2.7}$$

where $X(t)$ and $Y(t)$ are real independent Wiener processes. From this, lets build up in complexity until we get the real and complex matrix form of these equations. Before doing so, it is worth presenting the deviation of important results from the Langevin Equation/Ornstein-Uhlenbeck Process (2.1, 2.3), as the more general derivations follow the same general steps.

2.1.2 Langevin Equation

We'll start with the one-dimensional Ornstein-Uhlenbeck Process,

$$dX(t) = (-\theta - i\lambda)X(t)dt + \sigma dW(t) \tag{2.8}$$

Here σ can be complex, and θ, λ are real. For this to be a drag problem, θ should be positive. We can define an auxiliary process

$$f(t) = X(t)e^{(i\lambda+\theta)t}$$

so that we can solve for $X(t)$ in terms of $W(t)$

$$df(t) = e^{(i\lambda+\theta)t}dX(t) + X(t)e^{(i\lambda+\theta)t}(i\lambda+\theta)dt$$

$$df(t) = e^{(i\lambda+\theta)t}((-i\lambda-\theta)X(t)dt + \sigma dW(t) + X(t)e^{(i\lambda+\theta)t}(i\lambda+\theta)dt$$

$$df(t) = e^{(i\lambda+\theta)t}\sigma dW(t)$$

$$\int_{f(0)}^{f(t)} df(s) = \int_{W(0)}^{W(t)} e^{(i\lambda+\theta)t}\sigma dW(t)$$

$$X(t)e^{(i\lambda+\theta)t} = X(0) + \int_{W(0)}^{W(t)} e^{(i\lambda+\theta)s}\sigma dW(s)$$

$$X(t) = X(0)e^{-(i\lambda+\theta)t} + \int_0^t e^{(i\lambda+\theta)(s-t)}\sigma dW(s)$$

From here, with this closed form for $X(t)$, statistics can be extracted. The mean value of $X(t)$, unsurprisingly, corresponds to the solution without any stochasticity:

$$E[X(t)] = X(0)e^{-(\theta+i\lambda)t} + \int_0^t e^{(i\lambda+\theta)(s-t)}\sigma E[dW(s)]$$

$$E[X(t)] = X(0)e^{-(\theta+i\lambda)t}$$

As we will be looking at the covariance matrix as an outer product in the matrix-version of these calculations $E[|q(t+\tau)\rangle\langle q(t)|] - E[|q(t+\tau)\rangle]E[\langle q(t)|]$, the proper one-dimensional analog would be to look at the complex covariance at unequal times:

$$Cov[X(t)^*, X(t+\tau)] = E[(X(t)^* - E[X(t)^*])(X(t+\tau) - E[X(t+\tau)])]$$

$$= E[(\int_0^t e^{(\theta-i\lambda)(s-t)}\sigma^* dW(s))(\int_0^{t+\tau} e^{(\theta+i\lambda)(s-t-\tau)}\sigma dW(s))]$$

From here, we can invoke the Itô Isometry formula given in Equation 2.6.

$$Cov[X(t)^*, X(t+\tau)] = |\sigma|^2 \int_0^{\min(t,t+\tau)} e^{\theta(2s-2t-\tau)}e^{-i\lambda\tau}ds$$

$$Cov[X(t)^*, X(t+\tau)] = |\sigma|^2 e^{-i\lambda\tau}e^{\theta(-2t-\tau)} \int_0^{\min(t,t+\tau)} e^{2\theta s}ds$$

$$Cov[X(t)^*, X(t+\tau)] = \frac{|\sigma|^2}{2\theta}e^{-i\lambda\tau}e^{\theta(-2t-\tau)}\{e^{2\theta\min(t,t+\tau)} - 1\}$$

We can then swap $\min(a,b) = \frac{(a+b)-|a-b|}{2}$ to get a result that is a little more intuitive:

$$Cov[X(t)^*, X(t+\tau)] = \frac{|\sigma|^2}{2\theta} e^{-i\lambda\tau}\{e^{-\theta|\tau|} - e^{-\theta(2t+\tau)}\} \tag{2.9}$$

There are some important features in Equation 2.9 are are worth highlighting, as they will appear in a more complicated form in the matrix version of this equation. The phase imparted by $e^{-i\lambda\tau}$ reflects the wave solutions to $\partial_t x(t) = -i\lambda x(t)$. The $e^{-\theta|\tau|}$ term, reflects the change that occurs in time τ. As drag increases, X has its magnitude decrease from what it was at some earlier time. The last term, $^{-\theta(2t+\tau)}$ is often called the memory term, as it reflects the sum of the time between $X(t)$ and $X(t+\tau)$ and their shared initial conditions. In the presented analysis, I'll mostly throw this term out, as it can be eliminated with large t while other terms cannot. It can be unnecessarily burdensome to calculations, and is justifiably unimportant when sampling numerical data over long enough times.

I'll present Equation 2.9 in one more form. Heaviside Step-Functions $\Theta(t)$ can are by $\Theta(t) = 1$ if $t > 0$ and $\Theta(t) = 0$ if $t < 0$. In all further uses, I'll add in the convention $\Theta(0) = 0$ if $t = 0$, so that we can say $f(|t|) = \Theta(t)f(t) + \Theta(-t)f(-t)$. This language makes manipulating covariances with $|\tau|$ a little cleaner.

$$Cov[X(t)^*, X(t+\tau)] = \frac{|\sigma|^2}{2\theta} e^{-i\lambda\tau}\{\Theta(\tau)e^{-\theta\tau} + \Theta(-\tau)e^{\theta\tau} - e^{-\theta(2t+\tau)}\} \tag{2.10}$$

2.1.3 Matrix Ornstein-Uhlenbeck Process

Now looking back at the General Matrix Ornstein-Uhlenbeck Process, Equation 2.4:

$$|dq(t)\rangle = L\,|q(t)\rangle\,dt + \sigma\,|dW(t)\rangle$$

which can also be written with implied Einstein-summation:

$$dq_i(t) = L_{ik}q_k(t)dt + \sigma_{ik}dW_k(t) \tag{2.11}$$

where the components $W_k(t)$ of $|W(t)\rangle$ are independent Wiener Processes,and L and σ are complex matrices. We want to find the unequal time covariance matrix as the outer product:

$$Cov[|q(t+\tau)\rangle \langle q(t)|] = E[(|q(t+\tau)\rangle - E(|q(t+\tau)\rangle)(\langle q(t)| - E(\langle q(t)|)] \tag{2.12}$$

There is some flexibility in convention for how to define the components $W_k(t)$. As they are independent, the covariance between any two components $Cov[W_i(t)W_j(s)] = (t-s)\delta_{ij}$. These components could all be real, with σ transforming the noise to be complex, but this is a somewhat restricted case. It would be better to make each component complex, with independent real and imaginary parts. Instead $|W(t)\rangle = \frac{1}{\sqrt{2}}(|W'(t)\rangle + i\,|W''(t)\rangle)$, where $|W'(t)\rangle$ and $|W''(t)\rangle$ are vectors of independent, real Wiener processes. The normalization factor ensures that the covariance of each component with its complex conjugate has the proper covariance: $E[W_k^*(t)W_k(s)] = t - s$. Much like the analysis in Section 2.1.2, to get a closed form for $|q(t)\rangle$ we'll start with an auxiliary function $|f(t)\rangle$:

$$\begin{aligned}
|f(t)\rangle &= e^{-Lt}\,|q(t)\rangle \\
|df(t)\rangle &= -Le^{-Lt}\,|q(t)\rangle\,dt + e^{-Lt}\,|dq(t)\rangle \\
|df(t)\rangle &= e^{-Lt}\,|dW(t)\rangle \\
\int_0^t |df(s)\rangle\,ds = |f(t)\rangle - |f(0)\rangle &= \int_0^t e^{-Ls}\,|dW(s)\rangle \\
|q(t)\rangle &= e^{Lt}\,|q(0)\rangle + e^{Lt}\int_0^t e^{-Ls}\,|dW(s)\rangle
\end{aligned}$$

To get statistical properties of $|q(t)\rangle$.

$$\begin{aligned}
E[|q(t)\rangle] &= E[e^{Lt}\,|q(0)\rangle] + e^{Lt}\int_0^t e^{-Ls}\sigma E[|dW(s)\rangle] \\
E[|q(t)\rangle] &= e^{Lt}\,|q(0)\rangle \\
|q(t)\rangle - E[|q(t)\rangle] &= \int_0^t e^{L(t-s)}\sigma\,|dW(s)\rangle \\
\langle q(t)| - E[\langle q(t)|] &= \int_0^t \langle dW(s)|\,\sigma^\dagger e^{L^\dagger(t-s)}
\end{aligned}$$

This leads to a covariance very similar to Equation 2.1.2, with the exception that we have non-commutative matrices to deal with, and the Itô formulas for a real scalar $W(t)$, need to be extended to the complex vector $|W(t)\rangle$.

$$Cov[|q(t+\tau)\rangle \langle q(t)|] = E[(\int_0^{t+\tau} e^{L(t+\tau-s)}\sigma \, |dW(s)\rangle)(\int_0^t \langle dW(s)| \, \sigma^\dagger e^{L^\dagger(t-s)})]$$

From the definition of $|W(t)\rangle$ given in the beginning of this section, the expected extension of the Itô formulas can be derived:

$$E[\int_0^{t+\tau} A(s) \, |dW(s)\rangle \int_0^t \langle dW(s')| \, B(s')] = \int_0^{\min(t,t+\tau)} A(s)B(s)ds \tag{2.13}$$

Here, A and B are complex matrices. Simplifying notation slightly, we'll let $C(t+\tau,t) \equiv Cov[|q(t+\tau)\rangle \langle q(t)|]$. From the previous two equations we therefore get[14]:

$$\begin{aligned} C(t+\tau,t) &= \int_0^{\min(t,t+\tau)} e^{L(t+\tau-s)}\sigma\sigma^\dagger e^{L^\dagger(t-s)}ds \\ C(t+\tau,t) &= e^{L(\tau+t)}\Big(\int_0^{\min(t,t+\tau)} e^{-Ls}\sigma\sigma^\dagger e^{-L^\dagger s}ds\Big)e^{L^\dagger t} \end{aligned} \tag{2.14}$$

This is a well known result, though the integral in parentheses is not straightforward to solve. It is called the controllability (or reachability) Gramian when $t \to \infty$, a great deal of useful formulas exist for checking its behavior, but it is not generally solvable in any satisfying way[15][14]. Even if we assumed $\sigma\sigma^\dagger = I$, so long as we don't assume that L commutes with $L^\dagger$, and we run up against an extremely annoying fact of matrices: $e^X e^Y$ often[2] $\neq e^{X+Y}$ if X and Y don't commute. In some textbooks, the solution is simply presented in terms of this integral[16]. In a mathematical context, it is often the focus to analyze characteristics of $C(t,t+\tau)$ like it's asymptotic stability, rather than to calculate the exact solution for a given L and σ. Because of that, I was unable to find a reference that duplicated the following

[2]You can find pedagogical examples where X and Y don't commute, but $e^X = e^Y = e^{X+Y}$. $\begin{bmatrix} 0 & 0 \\ 0 & 2\pi i \end{bmatrix}$ and $\begin{bmatrix} 0 & 1 \\ 0 & 2\pi i \end{bmatrix}$ don't commute but have $e^X = e^Y = I$.

results exactly, with the exception of the case where $\tau = 0$[17]. Therefore, I'll make no claim of originality in these derivations, as they are only a few steps from Equation 2.14, and probably exist in specialized contexts.

As this equation is not exactly solvable in any satisfying way, there were a few paths forward, each with significant trade-offs. I could have just left the formula as is and integrated numerically for the shallow water equations at given parameters, then looked at plots and inferred behavior. Plots would have been a significant step down in rigor from derived equations. Also I could have made major assumptions about L and σ to make this solvable: like $[L, L^\dagger] = 0$ and $[L, \sigma] = 0$. This could have lead to a situation where the only examples left are those where excessive symmetries wash out any interesting behavior. Later analysis showed this to be the case. It could also be worthwhile to apply change-of-basis matrices to diagonalize L, and change basis back at the end. Unfortunately, the integral would only surrender if $\sigma\sigma^\dagger$ also became diagonal in the eigen-basis. This would correspond to forcing the system directly along its eigenvectors, which would be far too contrived and non-physical. In the shallow water equations it would be much better to force the system along the height or velocity, as these could reasonably correspond to the influence of turbulence in a more general, nonlinear geophysical system. There is however a best bad solution for the needed generality, using a language common to Econometrics[18].

This solution makes use of Kronecker products (also called tensor products) $X \otimes Y$, and Kronecker sums (also called direct sums) $X \oplus Y = X \otimes I + I \otimes Y$. While we may have $e^X e^Y \neq e^{X+Y}$, we always have:

$$e^X \otimes e^Y = e^{X \oplus Y} \tag{2.15}$$

The idea is elevate Equation 2.14 from the space of $n \times n$ matrices to $n^2 \times n^2$ matrices via Kronecker products. We can use this property to turn the $n \times n$ integral expression into a $n^2 \times n^2$ matrix expression, then finally bring the solution back down to the space of $n \times n$ matrices. There a two problems with this. First, while we can in principle return the matrix expression for any L and σ from the higher dimensional $n^2 \times n^2$ space back to $n \times n$, this

cannot be done for all L and σ at the same time. Therefore, we'll have to plug in the L and σ of interest into the $n^2 \times n^2$ expression one at a time. The other problem is that at face value this method requires taking inverses of $n^2 \times n^2$ matrices. Wolfram Mathematica[19] is very capable of doing matrix algebra for large matrices, and was used to derive $C(t, t+\tau)$ with this $n^2 \times n^2$ method. Luckily, with some manipulation, after sending $t \to \infty$, it can be shown that Fourier transforming along τ yields a far more approachable $n \times n$ matrix expression. This is:

$$\tilde{C}(\omega) = (L - i\omega I)^{-1}\sigma\sigma^{\dagger}((L - i\omega I)^{-1})^{\dagger} \tag{2.16}$$

The following calculations make use of operations that are generally easier to illustrate than define, though proper definition's are available in the references[18][16]. First the Kronecker product:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \otimes \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{bmatrix}$$

and Kronecker sum $A \oplus B = A \otimes I + I \otimes B$.

The 'vectorization' operator *vec* acts on $n \times n$ matrices to produce column vectors with length n^2, by stacking the columns left to right into one big column. Unfortunately the notation necessary to thoroughly define this vectorization is far less clear than some concrete

examples. The vec^{-1} operator inverts the operation.

$$vec\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{12} \\ a_{22} \\ a_{32} \\ a_{13} \\ a_{23} \\ a_{33} \end{bmatrix}, vec^{-1}\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \\ a_9 \end{bmatrix} = \begin{bmatrix} a_1 & a_4 & a_7 \\ a_2 & a_5 & a_8 \\ a_3 & a_6 & a_9 \end{bmatrix}$$

so that $vec^{-1}(vec(A)) = A, vec(vec^{-1}(\mathbf{a})) = \mathbf{a}$. As these are linear operators, we have $vec(A+B) = vec(A) + vec(B)$ and $vec^{-1}(\boldsymbol{a}+\boldsymbol{b}) = vec^{-1}(\boldsymbol{a}) + vec^{-1}(\boldsymbol{b})$. In Dirac's notation, one can think of a matrix as a row vector of 'kets'. The vectorization operator turns this into a column vector of 'kets':

$$vec\begin{bmatrix} |a\rangle & |b\rangle & |c\rangle \end{bmatrix} = \begin{bmatrix} |a\rangle \\ |b\rangle \\ |c\rangle \end{bmatrix}$$

Of key importance is the vectorization triple product theorem:

$$vec(ABC) = (C^T \otimes A)vec(B) \tag{2.17}$$

where C^T denotes the transpose of C.

For now, I'll only look at the central integral in Equation 2.14, and add in the exponentials on the sides later. To 'elevate' the expression to $n^2 \times n^2$ space, we'll apply the identity

operator, in the form $vec^{-1}vec$

$$\begin{aligned}
&\int_0^{\min(t,t+\tau)} e^{-Ls}\sigma\sigma^\dagger e^{-L^\dagger s}ds \\
&= vec^{-1}vec\Big(\int_0^{\min(t,t+\tau)} e^{-Ls}\sigma\sigma^\dagger e^{-L^\dagger s}ds\Big) \\
&= vec^{-1}\Big(\int_0^{\min(t,t+\tau)} vec(e^{-Ls}\sigma\sigma^\dagger e^{-L^\dagger s})ds\Big)
\end{aligned}$$

Taking advantage of Equation 2.17, we can remove the noise matrices and prepare matrix exponentials for integration:

$$\begin{aligned}
&= vec^{-1}\Big(\int_0^{\min(t,t+\tau)} (e^{-L^* s}\otimes e^{-Ls})vec(\sigma\sigma^\dagger)ds\Big) \\
&= vec^{-1}\Big(\int_0^{\min(t,t+\tau)} (e^{-L^* s}\otimes e^{-Ls})ds(vec(\sigma\sigma^\dagger))\Big)
\end{aligned}$$

Equation 2.15 allows for integration of a single matrix exponential:

$$\begin{aligned}
=&vec^{-1}\Big(\int_0^{\min(t,t+\tau)} (e^{-(L^*\oplus L)s})ds(vec(\sigma\sigma^\dagger)\Big) \\
=&vec^{-1}\Big((L^*\oplus L)^{-1}(I\otimes I - e^{-(L^*\oplus L)(\min(t,t+\tau))})vec(\sigma\sigma^\dagger)\Big)
\end{aligned}$$

Use of Heaviside step-functions allows for separation of terms into positive and negative τ cases, which can be expanded out using linearity of the vectorization operators:

$$\begin{aligned}
=&vec^{-1}\Big((L^*\oplus L)^{-1}vec(\sigma\sigma^\dagger)\Big) \\
&- \Theta(\tau)vec^{-1}\Big((L^*\oplus L)^{-1}e^{-(L^*\oplus L)t}vec(\sigma\sigma^\dagger)\Big) \\
&- \Theta(-\tau)vec^{-1}\Big((L^*\oplus L)^{-1}e^{-(L^*\oplus L)(t+\tau)}vec(\sigma\sigma^\dagger)\Big)
\end{aligned}$$

Again, Equation 2.17 can be used to incorporate the matrix exponents to the left and right

of the integral, via the resulting equality:

$$
\begin{aligned}
& Avec^{-1}(\mathbf{b})C = vec^{-1}((C^T \otimes A)\mathbf{b}) \\
\implies C(t+\tau,t) =& e^{L(\tau+t)}vec^{-1}\big((L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big)e^{L^\dagger t} \\
& - \Theta(\tau)e^{L(\tau+t)}\big((L^* \oplus L)^{-1}e^{-(L^*\oplus L)t}vec(\sigma\sigma^\dagger)\big)e^{L^\dagger t} \\
& - \Theta(-\tau)e^{L(\tau+t)}vec^{-1}\big((L^* \oplus L)^{-1}e^{-(L^*\oplus L)(t+\tau)}vec(\sigma\sigma^\dagger)\big)e^{L^\dagger t} \\
C(t+\tau,t) =& vec^{-1}\big((e^{L^*t} \otimes e^{L(t+\tau)})(L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) \\
& - \Theta(\tau)vec^{-1}\big((e^{L^*t} \otimes e^{L(t+\tau)})(L^* \oplus L)^{-1}e^{-(L^*\oplus L)t}vec(\sigma\sigma^\dagger)\big) \\
& - \Theta(-\tau)vec^{-1}\big((e^{L^*t} \otimes e^{L(t+\tau)})(L^* \oplus L)^{-1}e^{-(L^*\oplus L)(t+\tau)}vec(\sigma\sigma^\dagger)\big)
\end{aligned}
$$

$$
\begin{aligned}
C(t+\tau,t) =& vec^{-1}\big((e^{L^*t} \otimes e^{L(t+\tau)})(L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) \\
& - \Theta(\tau)vec^{-1}\big((I \otimes e^{L\tau})(L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) \\
& - \Theta(-\tau)vec^{-1}\big((e^{-L^*\tau} \otimes I)(L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big)
\end{aligned} \tag{2.18}
$$

Equation 2.18 is the main result of this analysis. The main issue is that $(L^* \oplus L)^{-1}$ is not always representable as a sum of Kronecker products, so we can't always use a formula like Equation 2.17 to undo the vectorization. Still, Mathematica is fully capable of multiplying and inverting these matrices. It is worth noting that the major computational load will be in finding $vec^{-1}\big((L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big)$, as the everything else can be brought outside the vectorization operator. As it will show up often, we'll call it $\mathcal{C}$. Doing so, we get the somewhat simpler form:

$$
\begin{aligned}
C(t+\tau,t) =& e^{L(t+\tau)}\mathcal{C}e^{L^\dagger t} - \Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \\
\mathcal{C} \equiv& vec^{-1}\big((L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big)
\end{aligned} \tag{2.19}
$$

It is important to note that $\mathcal{C}$ commutes with L if and only if $\sigma\sigma^\dagger$ commutes with L.[3]

It worth noting that we can recover the Langevin Covariance (Equation 2.10) from this, though more thorough ways of checking its accuracy will be presented shortly. For scalars,

[3] A matrix M commutes with a matrix L with distinct eigenvalues λ_i (and eigenvectors $|i\rangle$) if and only

Kronecker products become standard products and Kronecker sums become standard sums. The vectorization and inverse vectorization operators simply become the identity, since there is only one column of one entry to stack. For $L = -\theta - i\lambda$, Equation 2.10 is recovered.

In the limit of large $t \to \infty$, the first (memory) term disappears if the hermitian part of L, $(\frac{L+L^\dagger}{2})$ is negative-definite. With some small damping $-\theta I$ in the shallow water equations, this is satisfied. Taking this limit, we'll define:

$$C(\tau) = -\Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \tag{2.20}$$

The next section is dedicated to checks that the above formula is correct in general, and deriving formulas to check the solutions for specific L, σ in Mathematica. For that reason, the derivations presented in the next section were essential to ensuring correctness, but are not essential to understanding the results of this thesis. They are included as more of a record of work completed over the past year.

---

if it can be represented as a sum of outer products $M = \sum_i m_i |i\rangle \langle i|$. We can express:

$$L = \sum_i \lambda_i |i\rangle \langle i|, \; L^* = \sum_i \lambda_i^* |i^*\rangle \langle i^*| \implies (L^* \oplus L)^{-1} = \sum_{ij} \frac{1}{\lambda_i^* + \lambda_j} |i^* j\rangle \langle i^* j|$$

with $|i^*\rangle$ implying complex conjugation of all components. Defining $|ij\rangle \equiv |i\rangle \otimes |j\rangle$,

$$\mathcal{C} = \sum_{ij} \frac{1}{\lambda_i^* + \lambda_j} vec^{-1}((|i^*\rangle \langle i^*| \otimes |j\rangle \langle j|) vec(\sigma\sigma^\dagger)) = \sum_{ij} \frac{1}{\lambda_i^* + \lambda_j} |j\rangle \langle j| \sigma\sigma^\dagger |i\rangle \langle i|$$

From here, the two directions can be proved. If $\sigma\sigma^\dagger$ commutes with L, then

$$\sigma\sigma^\dagger = \sum_k s_k |k\rangle \langle k| \implies \mathcal{C} = \sum_i \frac{s_i}{\lambda_i^* + \lambda_i} |i\rangle \langle i|$$

To prove the reverse direction suppose $\mathcal{C}$ commutes, and $\sigma\sigma^\dagger = \sum_{ij} s_{ij} |i\rangle \langle j|$,

$$\mathcal{C} = \sum_i c_i |i\rangle \langle i| = \sum_{ij,i'j'} \frac{s_{i'j'}}{\lambda_i^* + \lambda_j} |j\rangle \langle j|i'\rangle \langle j'|i\rangle \langle i| = \sum_{ij} \frac{s_{ji}}{\lambda_i^* + \lambda_j} |j\rangle \langle i|$$

Since $|i\rangle \langle j|$'s are all linearly independent this requires,

$$0I = \sum_{i \neq j} \frac{s_{ji}}{\lambda_i^* + \lambda_j} |j\rangle \langle i| \implies \frac{s_{ji}}{\lambda_i^* + \lambda_j} = 0 \; \forall i \neq j$$

All λ_i's being finite, this implies $s_{j \neq i} = 0$, making $\sigma\sigma^\dagger$ commute too.

2.1.4 Lyapunov Equations as a Check

The Lyapunov Equation is a well known result often used for analyzing the asymptotic stability for the matrix Ornstein-Uhlenbeck process [14]. The controllability Gramian, bounded when the hermitian part of L is positive definite,

$$C_\infty \equiv C(t=\infty, \tau=0) = \int_0^\infty e^{Ls}\sigma\sigma^\dagger e^{L^\dagger s} ds$$

is a solution to the continuous-time Lyapunov Equation:

$$LC_\infty + C_\infty L^\dagger + \sigma\sigma^\dagger = 0 \tag{2.21}$$

As this is a well known result, I could have just checked the agreement of Equation 2.18 with this Lyapunov Equation in the $t \to \infty$, $\tau = 0$ limit and moved on; however, behavior when $\tau \neq 0$ is the general subject of interest, so it is certainly worth checking this case too. Additionally, these formulas helped helped to correct any errors with implementation in Mathematica, as they can be applied after L and σ are plugged in. In this section, I present the derivation of a more general Lyapunov-type equation from the Matrix Ornstein-Uhlenbeck Process (Equation 2.3) that accounts for an unequal-time covariance. This is presented after the standard derivation. After each derivation, the result of the previous section is shown to agree in the appropriate regime.

Equal-Time Lyapunov Equations

To derive the standard Lyapunov Equation, we'll start with the General Matrix Ornstein-Uhlenbeck Process, but instead elect to represent the equation in Einstein-summation form. We'll also use the more informal conception of noise: considering it to be a time derivative of a Wiener process, given by $|\eta\rangle \equiv \sigma\,|\dot{W}(t)\rangle$. Though this is not well defined formally, as $W(t)$ is not differentiable, the results are known to work properly, at this is common practice. It can be derived that for a real valued white noise η, $E[\dot{W}(t)\dot{W}(s)] = \delta(t-s)$[17]. In keeping with the definition of a complex Wiener process, this naturally extends to $E[\dot{W}_i^*(t)\dot{W}_j(s)] =$

$\delta_{ij}\delta(t-s)$.

$$
\begin{aligned}
|dq(t)\rangle &= L\,|q(t)\rangle\,dt + \sigma\,|dW(t)\rangle \\
|\dot{q}(t)\rangle &= L\,|q(t)\rangle + |\eta(t)\rangle \\
\dot{q}_i(t) &= L_{ik}q_k(t) + \eta_i(t)
\end{aligned} \tag{2.22}
$$

To derive the equal-time Lyapunov Equation, we want to look at $\frac{d}{dt}(E[|q(t)\rangle\,\langle q(t)|])$, so expressed by indices we have,

$$
\begin{aligned}
\frac{d}{dt}(E[q_j^*(t)q_i(t)]) &= E[\dot{q}_j^*(t)q_i(t) + q_j^*(t)\dot{q}_i(t)] \\
\frac{d}{dt}(E[q_j^*(t)q_i(t)]) &= E[(L_{jk}^*q_k^*(t) + \eta_j^*(t))q_i(t) + q_j^*(t)(L_{ik}q_k(t) + \eta_i(t))]
\end{aligned}
$$

We'll define the covariance at equal-times $C_{ij}(t,t) = E[q_j^*(t)q_i(t)]$.

$$
\frac{d}{dt}(E[q_j^*(t)q_i(t)]) = (L_{jk}^*C_{ik}(t,t) + L_{ik}C_{kj}(t,t)) + E[\eta_j^*(t)q_i(t) + \eta_i(t)q_j^*(t)] \tag{2.23}
$$

We'll note that

$$
E[\eta_j^*(t)q_i(t)] = \int_0^t E[\eta_j^*(t)\dot{q}_i(s)]ds + E[\eta_j^*(t)q_i(0)]
$$

Since $E[\eta_j^*(t)] = 0$, and $q_i(0)$ constant, we can input Equation 2.22 to get:

$$
E[\eta_j^*(t)q_i(t)] = \int_0^t E[\eta_j^*(t)\eta_i(s)] + L_{ik}\int_0^t E[\eta_j^*(t)q_k(s)]ds
$$

Looking at the second term, since $s \leq t$ the only contribution to the second integral will be from the point $t = s$, since $\eta_j(t)$ would only contribute to the evolution of $q_k(s)$ once it hits $s = t$. Even so, since q is finite, the only contribution of the correlation function is from $dq_k(t) = \eta_i(t)\delta_{ik}dt$ is infinitesimal. Even though η is allowed to be infinite, this contribution must be finite, since for a Wiener Process, $dW(t)$ is finite. Therefore we are integrating a finite value over an infinitesimal range. So the second term is zero. We get:

$$
E[\eta_j^*(t)q_i(t)] = \sigma_{jk}^*\sigma_{ik'}\delta_{kk'}\int_0^t E[\dot{W}_k^*(t)\dot{W}_{k'}(s)] = \sigma_{jk}^*\sigma_{ik}\int_0^t \delta(t-s)ds
$$

and a very similar answer for the other component

$$E[\eta_i(t)q_j^*(t)] = \sigma_{ik'}\sigma_{jk}^*\delta_{kk'}\int_0^t E[\dot{W}_k(t)\dot{W}_{k'}^*(s)] = \sigma_{ik}\sigma_{jk}^*\int_0^t \delta(t-s)ds$$

This integral is not one that is extremely well defined, as its bounds are just where the delta function diverges. It is somewhat uncomfortable to assign this a integral a value, but if there had to be a value, 1/2 would be the appropriate choice. It would be in keeping with the conception of the Dirac delta function as a normal distribution in the limit of small standard deviation. This issue a fault of the expedient method of differentiating $W(t)$. This unsatisfying answer is given more grounding by the following derivation of the unequal-time equation. Either way, the Lyapunov equation has been known for over a century. Incorporating these terms back into Equation 2.23, we recover:

$$\frac{d}{dt}(E[|q(t)\rangle\langle q(t)|]) = C(t,t)L^\dagger + LC(t,t) + \sigma\sigma^\dagger \tag{2.24}$$

We'll call this the time-dependent Lyapunov equation. In the limit where $t \to \infty$, the $|q\rangle$ should approach a state where its variance is unchanging (assuming the system is stable). In this limit the right hand side should be eliminated, recovering the Standard Lyapunov Equation:

$$LC_\infty + C_\infty L^\dagger + \sigma\sigma^\dagger = 0 \tag{2.21}$$

Now we can look at the Equation 2.19, the fully general covariance, and see if it agrees with the standard Lyapunov and time-dependent Lyapunov Equations

$$\begin{aligned} C(t+\tau,t) =& e^{L(t+\tau)}\mathcal{C}e^{L^\dagger t} - \Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \\ \mathcal{C} \equiv& vec^{-1}\big((L^*\oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) \end{aligned} \tag{2.19}$$

For $\tau = 0$, noting the convention $\Theta(0) = 1/2$, we have:

$$C(t,t) = e^{Lt}\mathcal{C}e^{L^\dagger t} - \mathcal{C}$$

We can plug $C(t,t)$ back into each side of the time-dependent Lyapunov equation 2.24 to see if they equal. On the left, we get:

$$Le^{Lt}\mathcal{C}e^{L^\dagger t} + e^{Lt}\mathcal{C}e^{L^\dagger t}L^\dagger$$

and on the right, we get:

$$Le^{Lt}\mathcal{C}e^{L^\dagger t} + e^{Lt}\mathcal{C}e^{L^\dagger t}L^\dagger - L\mathcal{C} - \mathcal{C}L^\dagger + \sigma\sigma^\dagger$$

We can cross out the parts that clearly match, and we only then need to show $L\mathcal{C}+\mathcal{C}L^\dagger = \sigma\sigma^\dagger$:

$$\begin{aligned} L\mathcal{C} + \mathcal{C}L^\dagger &= Lvec^{-1}\big((L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) + vec^{-1}\big((L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big)L^\dagger \\ &= vec^{-1}\big((L^* \oplus L)(L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) = \sigma\sigma^\dagger \end{aligned}$$

From this, in the case where $t \to \infty$, we'll note that in the case where the system is asymptotically stable (and the standard Lyapunov equation applies), e^{Lt} and $e^{L^\dagger t}$ will go to zero. Naturally, the standard Lyapunov equation also agrees with this result. Therefore we can identify $C_\infty = -\mathcal{C}$.

Unequal Time Lyapunov Equation

Now to derive a similar equation for the case where times differ by a constant factor τ. By the same logic as in the previous section:

$$\begin{aligned} \frac{d}{dt}(E[q_j^*(t)q_i(t+\tau)]) &= E[\dot{q}_j^*(t)q_i(t+\tau) + q_j^*(t)\dot{q}_i(t+\tau)] \\ \frac{d}{dt}(E[q_j^*(t)q_i(t+\tau)]) &= E[(L_{jk}^*q_k^*(t) + \eta_j^*(t))q_i(t+\tau) + q_j^*(t)(L_{ik}q_k(t+\tau) + \eta_i(t+\tau))] \\ \frac{d}{dt}C_{ij}(t+\tau,t) &= (L_{jk}^*C_{ik}(t+\tau,t) + L_{ik}C_{kj}(t+\tau,t)) \\ &\quad + E[\eta_j^*(t)q_i(t+\tau) + \eta_i(t+\tau)q_j^*(t)] \end{aligned}$$

Taking the first random process term and applying a similar analysis to before,

$$E[\eta_j^*(t)q_i(t+\tau)] = \int_0^{t+\tau} E[\eta_j^*(t)\dot{q}_i(s)]ds + E[\eta_j^*(t)q_i(0)]$$

Again, $E[\eta_j^*(t)] = 0$, so we have:

$$E[\eta_j^*(t)q_i(t+\tau)] = \int_0^{t+\tau} E[\eta_j^*(t)(L_{ik}q_k(s) + \eta_i(s))]ds$$
$$E[\eta_j^*(t)q_i(t+\tau)] = \int_0^{t+\tau} E[\eta_j^*(t)\eta_i(s)]ds + L_{ik}\int_0^{t+\tau} E[\eta_j^*(t)q_k(s)]ds \tag{2.25}$$

For the first term, we can once again use $E[\dot{W}_i^*(t)\dot{W}_j(s)] = \delta_{ij}\delta(t-s)$.

$$E[\eta_j^*(t)\eta_i(s)] = E[\sigma_{jk}^*\dot{W}_k^*(t)\sigma_{ik'}\dot{W}_{k'}(s)ds]$$
$$E[\eta_j^*(t)\eta_i(s)] = \sigma_{jk}^*\sigma_{ik'}\delta_{kk'}\delta(t-s) = (\sigma\sigma^\dagger)_{ij}\delta(t-s)$$

Adding in the integral, we get:

$$\int_0^{t+\tau} E[\eta_j^*(t)\eta_i(s)] = (\sigma\sigma^\dagger)_{ij}\int_0^{t+\tau}\delta(t-s)ds$$

If $\tau > 0$, the delta function integral sweeps past $s = t$, covering the whole delta function and equals 1. If $\tau < 0$, the integral never makes it to $s = t$, and we get 0. These are consistent with the Heaviside step-function $\Theta(\tau)$. Finally if $\tau = 0$, it would be reasonable to presume that the Heaviside step-function remains, with $\Theta(0) = 1/2$. Knowing that this is consistent with the original Lyapunov equation, and consistent with the conception of the Dirac delta function as a very narrow Gaussian, this integral gives the Heaviside step function $\Theta(\tau)$.

$$\int_0^{t+\tau} E[\eta_j^*(t)\eta_i(s)] = (\sigma\sigma^\dagger)_{ij}\Theta(\tau) \tag{2.26}$$

Taking a second look at Eq.2.25. If we define an auxiliary matrix by $X_{ij}(t+\tau, t) = E[\eta_j^*(t)q_i(t+\tau)]$, we get a nice integral equation:

$$X(t+\tau, t) = \sigma\sigma^\dagger\Theta(\tau) + L\int_0^{t+\tau} X(s,t)ds$$

Noting that $\frac{d\Theta(\tau)}{d\tau} = \delta(\tau)$, we can differentiate with respect to τ to obtain:

$$\frac{dX(t+\tau,t)}{d\tau} = \sigma\sigma^\dagger\delta(\tau) + LX(t+\tau,t)$$

From here, there is a natural solution:

$$X(t+\tau,t) = \sigma\sigma^\dagger\Theta(\tau)e^{L\tau}$$

where the leftover $e^{L\tau}$ that should be in the delta-function term can be omitted, as that term only contributes when $\tau = 0$.

Similarly, looking at the other term in the non-equal time Lyapunov Equation,

$$E[\eta_i(t+\tau)q_j^*(t)] = (E[\eta_i^*(t+\tau)q_j(t)])^* = (E[\eta_j^*(t+\tau)q_i(t)])^\dagger$$

we can avoid more calculation noting that $E[\eta_j^*(t)q_i(t+\tau)] = \Theta(\tau)(e^{L\tau}(\sigma\sigma^\dagger))_{ij}$ is independent of t. So we can shift in time $t \to t - \tau$[4] to get

$$\begin{aligned}
E[\eta_i(t+\tau)q_j^*(t)] &= (E[\eta_j^*(t)q_i(t-\tau)])^\dagger \\
E[\eta_i(t+\tau)q_j^*(t)] &= (\Theta(-\tau)(e^{-L\tau}(\sigma\sigma^\dagger))_{ij})^\dagger \\
E[\eta_i(t+\tau)q_j^*(t)] &= \Theta(-\tau)((\sigma\sigma^\dagger)e^{-L^\dagger\tau})_{ij}
\end{aligned}$$

Putting it all together, we get

$$\begin{aligned}
\frac{d}{dt}C_{ij}(t+\tau,t) &= L_{jk}^* C_{ik}(t+\tau,t) + L_{ik}C_{kj}(t+\tau,t)) \\
&\quad + \Theta(\tau)(e^{L\tau}(\sigma\sigma^\dagger))_{ij} + \Theta(-\tau)((\sigma\sigma^\dagger)e^{-L^\dagger\tau})_{ij}
\end{aligned}$$

or in Matrix form, the Unequal-time Lyapunov Equation is:

$$\begin{aligned}
\frac{d}{dt}C(t+\tau,t) &= C(t+\tau,t)L^\dagger + LC(t+\tau,t)) \\
&\quad + \Theta(\tau)e^{L\tau}\sigma\sigma^\dagger + \Theta(-\tau)\sigma\sigma^\dagger e^{-L^\dagger\tau}
\end{aligned} \tag{2.27}$$

This certainly agrees with the equal-time Lyapunov Equation (2.24), as the last two terms combine to give $\sigma\sigma^\dagger$. We can plug the general Covariance found into both sides to see if it

[4] so long as $t > \tau$, as would be natural for a problem with an initial condition

passes this more generalized Lyapunov Equation test.

$$\begin{aligned}
C(t+\tau,t) =& e^{L(t+\tau)}\mathcal{C}e^{L^\dagger t} - \Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \\
\frac{d}{dt}C(t+\tau,t) =& Le^{L(t+\tau)}\mathcal{C}e^{L^\dagger t} + e^{L(t+\tau)}\mathcal{C}e^{L^\dagger t}L^\dagger \\
C(t+\tau,t)L^\dagger + LC(t+\tau,t) =& Le^{L(t+\tau)}\mathcal{C}e^{L^\dagger t} + e^{L(t+\tau)}\mathcal{C}e^{L^\dagger t}L^\dagger \\
& - L\Theta(\tau)e^{L\tau}\mathcal{C} - L\Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \\
& - \Theta(\tau)e^{L\tau}\mathcal{C}L^\dagger - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau}L^\dagger
\end{aligned}$$

The t-dependent terms match on both sides, the task is now to show

$$\begin{aligned}
&\Theta(\tau)e^{L\tau}\sigma\sigma^\dagger + \Theta(-\tau)\sigma\sigma^\dagger e^{-L^\dagger\tau} \\
&= L\Theta(\tau)e^{L\tau}\mathcal{C} + L\Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} + \Theta(\tau)e^{L\tau}\mathcal{C}L^\dagger + \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau}L^\dagger
\end{aligned}$$

With the intention of transforming the left-hand side into the right, can group terms by $\pm\tau$, to get:

$$\begin{aligned}
&L\Theta(\tau)e^{L\tau}\mathcal{C} + L\Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} + \Theta(\tau)e^{L\tau}\mathcal{C}L^\dagger + \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau}L^\dagger \\
&= \Theta(\tau)e^{L\tau}\big(L\mathcal{C} + \mathcal{C}L^\dagger\big) + \Theta(-\tau)\big(L\mathcal{C} + \mathcal{C}L^\dagger\big)e^{-L^\dagger\tau}
\end{aligned}$$

We already have shown previously that $L\mathcal{C} + \mathcal{C}L^\dagger = \sigma\sigma^\dagger$, so this reproduces the right hand side, and Equation 2.19 is supported as the valid solution.

2.1.5 Fourier Transformed Covariance

In the linear shallow water equations, the eigenvectors have different phase-singularities depending on their frequencies: $\omega_+, \omega_0, \omega_-$. Looking at the unequal time covariance in frequency-space, we can distinguish these modes and see their properties, while they are combined in time-space. So it is worth Fourier-transforming $C(t, t+\tau) \to \tilde{C}(\omega)$. There is a reasonable question of whether to Fourier transform along t or τ. Seeing as the main objective of this thesis was to test whether phase singularities would manifest in field-correlations at

unequal times, it is best to focus on the time-difference τ. As t is less important, with the 'memory' term only regulating similarity to an initial condition, we'll set $t \to \infty$. Taking the Covariance found in the previous section, Equation 2.19:

$$\begin{aligned} C(t+\tau,t) =& e^{L(t+\tau)}\mathcal{C}e^{L^\dagger t} - \Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \\ \mathcal{C} \equiv& vec^{-1}\big((L^*\oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) \end{aligned}$$

Taking a limit where $t \to \infty$ the memory term disappears, and can analyze $C(\tau)$:

$$\begin{aligned} C(\tau) =& -\Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \\ \mathcal{C} \equiv& vec^{-1}\big((L^*\oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) \end{aligned} \tag{2.20}$$

From this we can Fourier transform $C(\tau) \to \tilde{C}(\omega)$. As $\mathcal{C}$ has no τ dependence, we can look the τ terms

$$\begin{aligned} \int_{-\infty}^{\infty} e^{-i\omega\tau}\Theta(\tau)e^{L\tau}d\tau &= \int_0^{\infty} e^{(L-i\omega I)\tau}d\tau = -(L-i\omega I)^{-1} \\ \int_{-\infty}^{\infty} e^{-i\omega\tau}\Theta(-\tau)e^{-L^\dagger\tau}d\tau &= \int_{-\infty}^{0} e^{-(L^\dagger+i\omega I)\tau}d\tau = -(L^\dagger+i\omega I)^{-1} \end{aligned}$$

This yields the Fourier transformed covariance:

$$\tilde{C}(\omega) = (L-i\omega I)^{-1}\mathcal{C} + \mathcal{C}((L-i\omega I)^{-1})^\dagger \tag{2.28}$$

From this, there is another simplification to be made. Here we can define $\mathcal{L} = L - i\omega I$, as it will show up often. First, it is important to not that $\mathcal{L}$ is has a negative definite Hermitian part whenever L does, so if the system is stable, $\mathcal{L}$ invertible for any ω. Next, note that

$$\mathcal{L}^*\oplus\mathcal{L} = L^*\otimes I + i\omega I\otimes I + I\otimes L + I\otimes -i\omega I = L^*\oplus L$$

Because of this, we can also say that $(L^*\oplus L)^{-1} = (\mathcal{L}^*\oplus\mathcal{L})^{-1}$, which will help in simplifying $\mathcal{C}$. Therefore we have

$$\begin{aligned} \tilde{C}(\omega) &= \mathcal{L}^{-1}vec^{-1}\big((\mathcal{L}^*\oplus\mathcal{L})^{-1}vec(\sigma\sigma^\dagger)\big) + vec^{-1}\big((\mathcal{L}^*\oplus\mathcal{L})^{-1}vec(\sigma\sigma^\dagger)\big)(\mathcal{L}^{-1})^\dagger \\ \tilde{C}(\omega) &= vec^{-1}\big(((\mathcal{L}^*)^{-1}\oplus\mathcal{L}^{-1})(\mathcal{L}^*\oplus\mathcal{L})^{-1}vec(\sigma\sigma^\dagger)\big) \end{aligned}$$

Additionally, we can make use of the formula

$$(A \oplus B) = (A \otimes B)(A^{-1} \oplus B^{-1})$$
$$\implies (A \oplus B)^{-1} = (A^{-1} \oplus B^{-1})^{-1}(A^{-1} \otimes B^{-1})$$

to simplify substantially to:

$$\tilde{C}(\omega) = vec^{-1}\big(((\mathcal{L}^*)^{-1} \otimes \mathcal{L}^{-1})vec(\sigma\sigma^\dagger)\big) = \mathcal{L}^{-1}\sigma\sigma^\dagger(\mathcal{L}^{-1})^\dagger$$

Finally, with ω dependence represented, we have a result that no longer requires $n^2 \times n^2$ matrix inversion, only the inverse of a single $n \times n$ matrix.

$$\tilde{C}(\omega) = (L - i\omega I)^{-1}\sigma\sigma^\dagger((L - i\omega I)^{-1})^\dagger \tag{2.16}$$

To check this is accurate, a quick analysis of the unequal-time Lyapunov Equation (2.27) yields the Fourier transformed Lyapunov Equation equivalent:

$$0 = (L^* \oplus L)vec(\tilde{C}(\omega)) - D((L^* + Ii\omega)^{-1} \oplus (L - Ii\omega)^{-1})vec(\sigma\sigma^\dagger)$$

which Equation 2.16 satisfies. Note that the relative simplicity in this Fourier transformed covariance does not necessarily imply the ability to inverse Fourier transform the system back into a function of τ which has a equivalent simple closed form. Due to the integral of the Fourier transform, it requires reincorporating vectorization and completing $n^2 \times n^2$ matrix algebra to solve.

The condition for resonance is very clear. Seeing that

$$det(\tilde{C}(\omega)) = \frac{|det(\sigma)|^2}{|det(L - i\omega I)|^2}$$

we have a situation where the system responds strongest where $i\omega$ is an eigenvalue of the non-damped system. Incorporating damping to keep the Hermitian part of L negative definite, this will not diverge.

A Shortcut to $\tilde{C}(\omega)$

Before moving on to the shallow water equations, it should be pointed out that there is a far faster way to derive Equation 2.16. This hinges on two steps that require far more mathematical justification than will be provided here. Since the true formula has already been derived in a thorough fashion, this method will be presented as the quick way to get the right answer, glossing over subtleties.

First, it is assumed that the white noise $\eta(t) = \dot{W}(t)$, and $|q(t)\rangle$ itself are assumed to be directly Fourier transformable: $\tilde{\eta}(\omega)$, $|\tilde{q}(\omega)\rangle$. So a simple formula for the Fourier transformed $|\tilde{q}(\omega)\rangle$ can be found directly from the matrix Ornstein-Uhlenbeck process.

$$\begin{aligned} \frac{\partial}{\partial t}|q(t)\rangle &= L|q(t)\rangle + \sigma|\dot{W}(t)\rangle \\ \implies i\omega|\tilde{q}(\omega)\rangle &= L|\tilde{q}(\omega)\rangle + \sigma|\tilde{\eta}(\omega)\rangle \\ \implies |\tilde{q}(\omega)\rangle &= -(L - i\omega I)^{-1}\sigma|\tilde{\eta}(\omega)\rangle \end{aligned}$$

Second it is assumed that the average over possible paths of $q(t)$, notated by $C(\tau) = E[|q(t+\tau)\rangle\langle q(t)|]$ (for very large t) can be replaced an average over time:

$$C(\tau) = \lim_{T\to\infty}\frac{1}{2T}\int_{-T}^{T}|q(t+\tau)\rangle\langle q(t)|\,dt$$

This is in fact the way that these averages are computed for the numerical simulations. Treating this as a definition of $C(\tau)$, it can be shown in few steps that:

$$\tilde{C}(\omega) = |\tilde{q}(\omega)\rangle\langle\tilde{q}(\omega)| \tag{2.29}$$

Therefore:

$$\tilde{C}(\omega) = (L - i\omega I)^{-1}\sigma|\tilde{\eta}(\omega)\rangle\langle\tilde{\eta}(\omega)|\,\sigma^{\dagger}((L - i\omega I)^{-1})^{\dagger}$$

From Equation 2.29, $|\tilde{\eta}(\omega)\rangle\langle\tilde{\eta}(\omega)|$ must be the Fourier transform of the previously defined covariance $|\eta(t)\rangle\langle\eta(t+\tau)| = \delta(\tau)I$. The Fourier transform of $\delta(\tau)$ is 1. Therefore we have:

$$\tilde{C}(\omega) = (L - i\omega I)^{-1}\sigma\sigma^{\dagger}((L - i\omega I)^{-1})^{\dagger}$$

Rigorous justification of all of these steps is likely possible, but since this answer has already been derived, genuine proof is outside the scope of this thesis.

2.2 Covariance of Shallow Water Equations

To analyze this covariance for the shallow water equations, we can start with the damped linear shallow water model in k-space:

$$L = \begin{bmatrix} -\theta & -ick_x & -ick_y \\ -ick_x & -\theta & f \\ -ick_y & -f & -\theta \end{bmatrix} \tag{1.11}$$

As a reminder, this matrix acts on the vector height, x-velocity, y-velocity in k-space: $(h(k_x, k_y), u(k_x, k_y), v(k_x, k_y))$. For this model, the covariance $\tilde{C}(\omega)$ in frequency-space, which is simpler and easier to derive than $C(\tau)$ will be described first.

2.2.1 Frequency Space Covariance for Shallow Water L

This matrix was found three ways: via the 'vectorized' Fourier transformed covariance in Mathematica,

$$\begin{aligned} \tilde{C}(\omega) =& (L - i\omega I)^{-1}\mathcal{C} + \mathcal{C}((L - i\omega I)^{-1})^\dagger \\ \mathcal{C} \equiv& vec^{-1}\big((L^* \oplus L)^{-1} vec(\sigma\sigma^\dagger)\big) \end{aligned} \tag{2.28}$$

the simpler 'unvectorized version' of the same equation (which was realized later),

$$\tilde{C}(\omega) = (L - i\omega I)^{-1}\sigma\sigma^\dagger((L - i\omega I)^{-1})^\dagger \tag{2.16}$$

and by directly integrating the τ-dependent covariance, found using Mathematica, for a set L, σ:

$$C(\tau) = -\Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \tag{2.20}$$

$\tilde{C}(\omega)$ will be presented in the language of Equation 2.16.

From Equation 1.11, we have:

$$(L - i\omega I)^{-1} = \frac{1}{(-i\omega - \theta)((i\omega + \theta)^2 + f^2 + c^2k^2)} \tag{2.30}$$
$$* \begin{bmatrix} (i\omega + \theta)^2 + f^2 & ick_yf - ick_x(i\omega + \theta) & -ick_xf - ick_y(i\omega + \theta) \\ -ick_yf - ick_x(i\omega + \theta) & (i\omega + \theta)^2 + c^2k_y^2 & -c^2k_xk_y - f(i\omega + \theta) \\ -ick_xf - ick_y(i\omega + \theta) & -c^2k_xk_y + f(i\omega + \theta) & (i\omega + \theta)^2 + c^2k_x^2 \end{bmatrix}$$

This inverse does contain nearly divergent phase singularities in its off-diagonal elements, reflecting the phase singularities in eigenvectors of L. These are easier to illustrate when the real numbered drag constant θ is set to zero. Still, only slightly modified logic can be applied to a system $\theta \neq 0$.

$$(L + \theta I - i\omega I)^{-1} = \frac{1}{(-i\omega)(-\omega^2 + f^2 + c^2k^2)}$$
$$* \begin{bmatrix} -\omega^2 + f^2 & -ick_yf + ck_x\omega & ick_xf + ck_y\omega \\ ick_yf + ck_x\omega & -\omega^2 + c^2k_y^2 & -c^2k_xk_y - if\omega \\ -ick_xf + ck_y\omega & -c^2k_xk_y + if\omega & -\omega^2 + c^2k_x^2 \end{bmatrix}$$

In this simplified matrix, there are divergences in the prefactor at the eigenvalues of $L - \theta I$: $\omega = 0$ and $\omega = \pm i\sqrt{f^2 + c^2k^2}$. These divergences become significant maxima in the case where θ is small. In the h, u and h, v components (and their corresponding diagonal reflections) there are singularities in k_x and k_y when $f = \pm\omega$. These take a form of the type $(k_x + ik_y)/fk^2$, somewhat like the singularities in the eigenvectors of L. They are distinguished by the property that these singularities diverge at small k while in the eigenvectors

the singularities have magnitude 1, being of type $(k_x + ik_y)/k$.

$$|\omega_+\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{ck}{\sqrt{c^2k^2+f^2}} \\ \frac{k_x}{k} - \frac{ifk_y}{k\sqrt{c^2k^2+f^2}} \\ \frac{k_y}{k} + \frac{ifk_x}{k\sqrt{c^2k^2+f^2}} \end{bmatrix}, |\omega_0\rangle = \begin{bmatrix} \frac{-if}{\sqrt{c^2k^2+f^2}} \\ \frac{ck_y}{\sqrt{c^2k^2+f^2}} \\ \frac{-ck_x}{\sqrt{c^2k^2+f^2}} \end{bmatrix}$$
$$|\omega_-\rangle = \frac{1}{\sqrt{2}} \begin{bmatrix} \frac{-ck}{\sqrt{c^2k^2+f^2}} \\ \frac{k_x}{k} + \frac{ifk_y}{k\sqrt{c^2k^2+f^2}} \\ \frac{k_y}{k} - \frac{ifk_x}{k\sqrt{c^2k^2+f^2}} \end{bmatrix} \tag{1.13}$$

Just like in analysis of eigenvectors of L, these diverging ω values correspond to the smallest allowed frequencies on the Poincare Bands: $\omega = \pm f$. These have the highest Berry curvature of the bands, as can be seen in Figure 1.7.

Similar singularities k_x, k_y singularities do not appear in the u, v matrix elements. The reason for this absence, as well as the reason for the presence of singularities can be most easily understood in terms of change-of-basis matrices for the normalized eigenbasis. We have:

$$\begin{aligned} &(L + \theta I - i\omega I)^{-1} \\ &= \frac{i}{\omega - \sqrt{f^2 + c^2k^2}} |\omega_+\rangle \langle\omega_+| + \frac{i}{\omega} |\omega_0\rangle \langle\omega_0| + \frac{i}{\omega + \sqrt{f^2 + c^2k^2}} |\omega_-\rangle \langle\omega_-| \end{aligned} \tag{2.31}$$

Since the diagonalized $(L + \theta I - i\omega I)^{-1}$ contains no k_x, k_y phase singularities, the only contributions come from the outer products of eigenvectors. $|\omega_0\rangle$ is has components that are entirely real, or entirely imaginary so its outer product would have the same property, and it would not contribute any phase singularities. It is given by:

$$|\omega_0\rangle \langle\omega_0| = \begin{bmatrix} \frac{f^2}{c^2k^2+f^2} & \frac{-ifck_y}{c^2k^2+f^2} & \frac{ifck_x}{c^2k^2+f^2} \\ \frac{ifck_y}{c^2k^2+f^2} & \frac{c^2k_y^2}{c^2k^2+f^2} & \frac{-c^2k_yk_x}{c^2k^2+f^2} \\ \frac{-ifck_x}{c^2k^2+f^2} & \frac{-c^2k_xk_y}{c^2k^2+f^2} & \frac{c^2k_x^2}{c^2k^2+f^2} \end{bmatrix}$$

The h elements of $|\omega_+\rangle$ and $|\omega_-\rangle$ also lack singularities. The other two outer products are

given by:

$$|\omega_+\rangle\langle\omega_+| = \frac{1}{2(c^2k^2+f^2)} * \begin{bmatrix} c^2k^2 & ck_x\sqrt{c^2k^2+f^2}+ifck_y & ck_y\sqrt{c^2k^2+f^2}-ifck_x \\ ck_x\sqrt{c^2k^2+f^2}-ifck_y & c^2k_x^2+f^2 & c^2k_xk_y-if\sqrt{c^2k^2+f^2} \\ ck_y\sqrt{c^2k^2+f^2}+ifck_x & c^2k_xk_y+if\sqrt{c^2k^2+f^2} & c^2k_y^2+f^2 \end{bmatrix}$$

$$|\omega_-\rangle\langle\omega_-| = \frac{1}{2(c^2k^2+f^2)} * \begin{bmatrix} c^2k^2 & -ck_x\sqrt{c^2k^2+f^2}+ifck_y & -ck_y\sqrt{c^2k^2+f^2}-ifck_x \\ -ck_x\sqrt{c^2k^2+f^2}-ifck_y & c^2k_x^2+f^2 & c^2k_xk_y+if\sqrt{c^2k^2+f^2} \\ -ck_y\sqrt{c^2k^2+f^2}+ifck_x & c^2k_xk_y-if\sqrt{c^2k^2+f^2} & c^2k_y^2+f^2 \end{bmatrix}$$

It can be seen from these outer products that for both eigenvectors, for the velocity-height terms, the singularities from the velocity components persist, as they are multiplied by a real number. For the velocity-velocity components, k_x, k_y singularities cancel each other. It works out that these singularities persist when inserted into Equation 2.31. However this is not entirely necessary to show why they persist. Adjustment of ω in Equation 2.31 allows for the relative sizes of prefactors to change in magnitude. Coupled with ω_0 having no singularities, $|\omega_+\rangle\langle\omega_+|$ and $|\omega_-\rangle\langle\omega_-|$ cannot cancel with each other for arbitrary ω. With damping added back in, we have:

$$\begin{aligned}(L-i\omega I)^{-1} =& \frac{i}{(\omega-i\theta)-\sqrt{f^2+c^2k^2}}|\omega_+\rangle\langle\omega_+| \\ &+\frac{i}{\omega-i\theta}|\omega_0\rangle\langle\omega_0| + \frac{i}{(\omega-i\theta)+\sqrt{f^2+c^2k^2}}|\omega_-\rangle\langle\omega_-|\end{aligned}$$

Non-Physical Singularities

The u, v components of Equation 2.30, though they lack k_x, k_y singularities, do not lack any interesting behavior. For $\frac{-c^2k_xk_y+if\omega}{(-i\omega)(-\omega^2+f^2+c^2k^2)}$, there is a non-vanishing singularity when f is very small, and one of ck_x or ck_y is very small. Though the same reasoning can be applied when they are switched, ck_x will be considered small and while ck_y is not. In this

case, when investigating the system at $ck_y = \omega$, there is a phase singularity taking the form $(-ick_x + f)/(c^2k_x^2 + f^2)$ that would manifest near the equator $f \approx 0$.

This is, however, is somewhat misleading. By choosing $f = f_0$ rather than $f = f_0 + \beta y$ to make the shallow water Model solvable in this context, a proper treatment of the equator was sacrificed. This does not necessarily mean that analogous singularities would not show up in more advanced models. It only means that these singularities cannot be assumed to represent a significant behavior here. Whether these singularities persist in $f = \beta y$, or how they translate over is worth investigating in future work.

Returning to an analysis of the general shallow water $\tilde{C}(\omega)$, Equation 2.32, this term is written as:

$$(L - i\omega I)^{-1} = \frac{1}{(-i\omega - \theta)((i\omega + \theta)^2 + f^2 + c^2k^2)} \tag{2.30}$$
$$* \begin{bmatrix} (i\omega + \theta)^2 + f^2 & ick_yf - ick_x(i\omega + \theta) & -ick_xf - ick_y(i\omega + \theta) \\ -ick_yf - ick_x(i\omega + \theta) & (i\omega + \theta)^2 + c^2k_y^2 & -c^2k_xk_y + f(i\omega + \theta) \\ ick_xf - ick_y(i\omega + \theta) & -c^2k_xk_y - f(i\omega + \theta) & (i\omega + \theta)^2 + c^2k_x^2 \end{bmatrix}$$

In the case of any $\sigma\sigma^\dagger$ all that can be done is to separate out the prefactors of $(L - i\omega I)^{-1}$ and $((L - i\omega I)^{-1})^\dagger$. They become:

$$\frac{1}{|det|^2} \equiv \frac{1}{(\omega^2 + \theta^2)((i\omega + \theta)^2 + f^2 + c^2k^2)((i\omega - \theta)^2 + f^2 + c^2k^2)}$$
$$\frac{1}{|det|^2} = \frac{1}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$

Leaving a general formula that cannot be simplified further until σ is chosen:

$$\tilde{C}(\omega) = \frac{1}{|det|^2}\frac{1}{(-i\omega-\theta)((i\omega+\theta)^2+f^2+c^2k^2)} \tag{2.32}$$
$$* \begin{bmatrix} (i\omega+\theta)^2+f^2 & ick_yf-ick_x(i\omega+\theta) & -ick_xf-ick_y(i\omega+\theta) \\ -ick_yf-ick_x(i\omega+\theta) & (i\omega+\theta)^2+c^2k_y^2 & -c^2k_xk_y+f(i\omega+\theta) \\ ick_xf-ick_y(i\omega+\theta) & -c^2k_xk_y-f(i\omega+\theta) & (i\omega+\theta)^2+c^2k_x^2 \end{bmatrix}$$
$$* \sigma\sigma^\dagger \begin{bmatrix} (-i\omega+\theta)^2+f^2 & ick_yf+ick_x(-i\omega+\theta) & -ick_xf+ick_y(-i\omega+\theta) \\ -ick_yf+ick_x(-i\omega+\theta) & (-i\omega+\theta)^2+c^2k_y^2 & -c^2k_xk_y-f(-i\omega+\theta) \\ ick_xf+ick_y(-i\omega+\theta) & -c^2k_xk_y+f(-i\omega+\theta) & (-i\omega+\theta)^2+c^2k_x^2 \end{bmatrix}$$

Treating In the case where $\sigma\sigma^\dagger$ commutes with $L, L^\dagger$, a far cleaner formula can be derived. In this case $\sigma\sigma^\dagger$ can therefore be written as $\sigma\sigma^\dagger = s_+ |\omega_+\rangle \langle\omega_+| + s_0 |\omega_-\rangle \langle\omega_-| + s_- |\omega_-\rangle \langle\omega_-|$, leading to the straightforward formula:

$$\tilde{C}(\omega) = \frac{s_+}{(\omega-\sqrt{f^2+c^2k^2})^2+\theta^2} |\omega_+\rangle \langle\omega_+| \tag{2.33}$$
$$+ \frac{s_0}{\omega^2+\theta^2} |\omega_0\rangle \langle\omega_0| + \frac{s_-}{(\omega+\sqrt{f^2+c^2k^2})^2+\theta^2} |\omega_-\rangle \langle\omega_-|$$

It is worth noting that s_+, s_0, s_- must all be positive as $\sigma\sigma^\dagger$ is hermitian. In this case, phase singularities can be observed in a very straightforward way. These will take the form given in the above inner products, and can have their relative magnitude controlled by the distance of ω to either $\omega_\pm$. This formula would apply to any noise matrix of the form $\sigma = (\sigma_{++} |\omega_+\rangle \langle\omega_+| + \sigma_{00} |\omega_-\rangle \langle\omega_-| + \sigma_{--} |\omega_-\rangle \langle\omega_-|)U$, where U is some unitary matrix.

Therefore, we can conclude that phase singularities should be clearly visible in $\tilde{C}(\omega)$, for at least a simple class of σ. It is expected, though not proven here that phase singularities should be present in $\tilde{C}(\omega)$ for any σ where $\langle\omega_\pm| \sigma\sigma^\dagger |\omega_\pm\rangle \neq 0$. With this established, the somewhat more complicated time-dependent $C(\tau)$ will be presented for the linear shallow water model.

2.2.2 Time-dependent Covariance for Shallow Water L

This section, like the last, will probe the τ-dependent covariance given by Equation 2.20 as far as possible before settling on a set noise matrix σ:

$$C(\tau) = -\Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau}$$
$$\mathcal{C} \equiv vec^{-1}\big((L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big)$$

For most σ, $\mathcal{C}$ is unfortunately somewhat of a black-box, and Mathematica is necessary to obtain a clear answer. However, like in the previous section important characteristics of $\mathcal{C}$ and $e^{Lt}, e^{-L^\dagger t}$ can be addressed in the special case where where L commutes with $\sigma\sigma^\dagger$. It has already been shown that this happens if and only if $\mathcal{C}$ commutes with L.

$$e^{Lt} = e^{-\theta t}(e^{-i\sqrt{c^2k^2+f^2}t}\left|\omega_+\right\rangle\left\langle\omega_+\right| + \left|\omega_0\right\rangle\left\langle\omega_0\right| + e^{i\sqrt{c^2k^2+f^2}t}\left|\omega_-\right\rangle\left\langle\omega_-\right|)$$

In this ideal scenario, drag terms with Heaviside step-functions can be combined to form:

$$C(\tau) =(-e^{-\theta|\tau|})(e^{-i\sqrt{c^2k^2+f^2}\tau}\left|\omega_+\right\rangle\left\langle\omega_+\right| + \left|\omega_0\right\rangle\left\langle\omega_0\right| + e^{i\sqrt{c^2k^2+f^2}\tau}\left|\omega_-\right\rangle\left\langle\omega_-\right|)\mathcal{C}$$

Just as in the Fourier-transformed case, $\mathcal{C}$ can be diagonalized in the eigenbasis, with entries $\kappa_+, \kappa_0, \kappa_-$ to avoid confusion with the wave-speed c.

$$C(\tau) =(-e^{-\theta|\tau|})(\kappa_+e^{-i\sqrt{c^2k^2+f^2}\tau}\left|\omega_+\right\rangle\left\langle\omega_+\right| + \kappa_0\left|\omega_0\right\rangle\left\langle\omega_0\right| + \kappa_-e^{i\sqrt{c^2k^2+f^2}\tau}\left|\omega_-\right\rangle\left\langle\omega_-\right|)$$

The constants κ_i can be related back to the entries of a diagonal $\sigma\sigma^\dagger$ by $\kappa_i = \frac{s_i}{-2\theta}$. It should be emphasized that in this specific commuting case, $\mathcal{C}$'s diagonal entries have no dependence on the oscillatory part of L, only the drag part. For a better description of why, see footnote 3 on page 39. As shown in the previous section, every s_i is real and positive.

Referring, back to the outer products written in the previous section, we can look at just the h, u component of $\left|\omega_i\right\rangle\left\langle\omega_i\right|$'s, as h, v and their transpose counterparts share the same

qualitative behavior.

$$\begin{aligned} C(\tau)_{hu} = \frac{e^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}(s_+e^{-i\sqrt{c^2k^2+f^2}\tau}(ck_x\sqrt{c^2k^2+f^2} - ifck_y) \\ + 2s_0(ifck_y) \\ + s_-e^{i\sqrt{c^2k^2+f^2}\tau}(-ck_x\sqrt{c^2k^2+f^2} - ifck_y)) \end{aligned} \tag{2.34}$$

$$\begin{aligned} C(\tau)_{hu} = \frac{e^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}(ck_x\sqrt{c^2k^2+f^2}(s_+e^{-i\sqrt{c^2k^2+f^2}\tau} - s_-e^{i\sqrt{c^2k^2+f^2}\tau}) \\ + ifck_y(2s_0 - s_+e^{-i\sqrt{c^2k^2+f^2}\tau} - s_-e^{i\sqrt{c^2k^2+f^2}\tau})) \end{aligned}$$

As will be further explored in an analysis of $\sigma = I$, phase singularities do not persist in this time-dependent when $s_+ = s_- \equiv s_\pm$. In these cases, the oscillatory terms both become imaginary:

$$C(\tau)_{hu} = \frac{2ie^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\big(s_\pm(ck_x\sqrt{c^2k^2+f^2}\sin(\sqrt{c^2k^2+f^2}\tau) - ck_yf\cos(\sqrt{c^2k^2+f^2}\tau)) + s_0ck_yf)\big)$$

This is a major issue for observation of phase singularities it τ-space, and it can be applied to every term in this covariance:

> For the easily solvable case where $\sigma\sigma^\dagger$ diagonalizes with L, unless one topological mode is consciously driven more than the other, no phase singularities appear in $C(\tau)$.

Still, this is by no means an uninteresting function. At small $k_x = k\cos(\phi)$ and $k_y = k\sin(\phi)$, the evolution of $C(\tau)_hu$ when k goes around the origin of the k_x k_y plane ($\phi \to \phi \pm 2\pi$) is identical to an evolution of $C(\tau)_hu$ as $t \to \pm\frac{2\pi}{\sqrt{c^2k^2+f^2}}$.

With that in mind, it is now time to chose and investigate an appropriate σ.

Noise Matrix $\sigma = I$

The simplest choice : $\sigma = I$ has the previously addressed issue of no singularities in $C(\tau)$:

$$C(\tau)_{hu} = \frac{2ie^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\big((ck_x\sqrt{c^2k^2+f^2}\sin(\sqrt{c^2k^2+f^2}\tau) + ck_yf(1-\cos(\sqrt{c^2k^2+f^2}\tau)))\big)$$

The other terms share this feature:

$$C(\tau)_{hh} = \frac{2e^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(f^2+\cos(\sqrt{c^2k^2+f^2}\tau)c^2k^2\right)$$
$$C(\tau)_{hu} = \frac{2ie^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(((\cos(\sqrt{c^2k^2+f^2}\tau)-1)ck_yf-(\sin(\sqrt{c^2k^2+f^2}\tau))ck_x\sqrt{c^2k^2+f^2})\right)$$
$$C(\tau)_{hv} = \frac{-2ie^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(((\cos(\sqrt{c^2k^2+f^2}\tau)-1)ck_xf+(\sin(\sqrt{c^2k^2+f^2}\tau))ck_y\sqrt{c^2k^2+f^2})\right)$$
$$C(\tau)_{uh} = \frac{-2ie^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(((\cos(\sqrt{c^2k^2+f^2}\tau)-1)ck_yf+(\sin(\sqrt{c^2k^2+f^2}\tau))ck_x\sqrt{c^2k^2+f^2})\right)$$
$$C(\tau)_{uu} = \frac{2e^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(\cos(\sqrt{c^2k^2+f^2}\tau)(f^2+c^2k_x^2)+c^2k_y^2\right)$$
$$C(\tau)_{uv} = \frac{2e^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(c^2k_xk_y(\cos(\sqrt{c^2k^2+f^2}\tau)-1)-\sqrt{c^2k^2+f^2}f\sin(\sqrt{c^2k^2+f^2}\tau)\right)$$
$$C(\tau)_{vh} = \frac{2ie^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(fck_x(\cos(\sqrt{c^2k^2+f^2}\tau)-1)+\sqrt{c^2k^2+f^2}ck_y\sin(\sqrt{c^2k^2+f^2}\tau)\right)$$
$$C(\tau)_{vu} = \frac{2e^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left(c^2k_xk_y(\cos(\sqrt{c^2k^2+f^2}\tau)-1)+f\sqrt{c^2k^2+f^2}\sin(\sqrt{c^2k^2+f^2}\tau)\right)$$
$$C(\tau)_{vv} = \frac{2e^{-\theta|\tau|}}{4\theta(c^2k^2+f^2)}\left((f^2+c^2k_y^2)\cos(\sqrt{c^2k^2+f^2}\tau)+c^2k_x^2\right)$$

It was shown in Mathematica that these satisfy the time-dependent Lyapunov Equation for τ (Equation 2.27), and the standard Lyapunov equation when $\tau = 0$, where the $C(0)$ becomes $\mathcal{C} = \frac{-1}{2\theta}$. There is little else to address about this simple case, other than the interesting property that while its diagonal element are invariant under $\tau \to -\tau$, the off-diagonal elements require time-reversal of the Coriolis term in L. $C(\tau, f) = C(-\tau, -f)^\dagger$.

For the Fourier transformed covariance, the formula:

$$\begin{aligned}\tilde{C}(\omega) =& \frac{1}{(\omega-\sqrt{f^2+c^2k^2})^2+\theta^2}\left|\omega_+\right\rangle\left\langle\omega_+\right| \\ &+ \frac{1}{\omega^2+\theta^2}\left|\omega_0\right\rangle\left\langle\omega_0\right| + \frac{1}{(\omega+\sqrt{f^2+c^2k^2})^2+\theta^2}\left|\omega_-\right\rangle\left\langle\omega_-\right|\end{aligned} \tag{2.33}$$

was used to obtain the covariance:

$$\tilde{C}(\omega)_{hh} = \frac{(f^2 + c^2k^2 + \theta^2 + \omega^2)(\theta^2 + \omega^2) + (f^2(f^2 + c^2k^2 + \theta^2 - 3\omega^2)))}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{hu} = \frac{-2ck_x\omega(\theta^2 + \omega^2) + ick_yf(c^2k^2 + f^2 + \theta^2 - 3\omega^2)}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{hv} = \frac{-2ck_y\omega(\theta^2 + \omega^2) - ick_xf(f^2 + ck^2 + \theta^2 - 3\omega^2)}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{uh} = \frac{-2ck_x\omega(\theta^2 + \omega^2) - ick_yf(c^2k^2 + f^2 + \theta^2 - 3\omega^2)}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{uu} = \frac{(f^2 + c^2k^2 + \theta^2 + \omega^2)(\theta^2 + \omega^2) + (c^2k_y^2(f^2 + c^2k^2 + \theta^2 - 3\omega^2)))}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{uv} = \frac{-2if\omega(\theta^2 + \omega^2) - (c^2k_xk_y(f^2 + c^2k^2 + \theta^2 - 3\omega^2)))}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{vh} = \frac{-2ck_y\omega(\theta^2 + \omega^2) + ick_xf(f^2 + ck^2 + \theta^2 - 3\omega^2)}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{vu} = \frac{2if\omega(\theta^2 + \omega^2) - (c^2k_xk_y(f^2 + c^2k^2 + \theta^2 - 3\omega^2)))}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$
$$\tilde{C}(\omega)_{vv} = \frac{(f^2 + c^2k^2 + \theta^2 + \omega^2)(\theta^2 + \omega^2) + (c^2k_x^2(f^2 + c^2k^2 + \theta^2 - 3\omega^2)))}{(\omega^2 + \theta^2)((c^2k^2 + f^2 - \omega^2)^2 + 2\theta^2(c^2k^2 + f^2 + \omega^2) + \theta^4)}$$

which is confirmed by Equation 2.32. Treating only leading order terms in θ, we can inspect the singularities of $\tilde{C}(\omega)_{hu}$, which is representative of the other velocity-height correlation singularities.

$$\tilde{C}(\omega)_{hu,\theta=0} = \frac{-2ck_x\omega^3 + ick_yf(c^2k^2 + f^2 - 3\omega^2)}{(\omega^2)(c^2k^2 + f^2 - \omega^2)^2}$$

We'll treat the system in the case of $\omega = \pm f$, where ck_x and ck_y are very small compared to them. This simplifies to:

$$\tilde{C}(\omega)_{hu,\theta=0} = \frac{-2f(\pm ck_x + ick_y)}{(c^2k^2)^2} + \frac{ick_y}{f^2c^2k^2}$$

The two terms both diverge as $k \to 0$, but the while the first term diverges scales as $1/c^3k^3$, the second only scales as $1/ck$. So the signal from the first should be much stronger at $f = \omega$. More significantly, this term does contain a phase singularity: $f(\pm ck_x + ick_y)$. The form of the first term is nearly identical to that of the $\omega_\pm$ eigenvectors, the only difference being more dramatic scaling. A deeper analysis of phase singularities is given for the next choice of σ.

Seeing that $[\sigma\sigma^\dagger, L] = 0$ produces no phase singularities unless certain topological modes are driven more than others, the non-commuting case of $\sigma\sigma^\dagger = |h\rangle\langle h|$ will be investigated next.

Noise Matrix $\sigma = |h\rangle\langle h|$

$\sigma = |h\rangle\langle h|$[5] is a decent choice for noise as it does not preferentially treat either topological eigenvector: $\langle\omega_+|h\rangle\langle h|\omega_+\rangle = \langle\omega_-|h\rangle\langle h|\omega_-\rangle$, and it can justifiably represent some physical process not incorporated into L. This could very roughly model effect of a the system no longer satisfying hydrostatic balance (Equation 1.9), or a reincorporation of vertical advection.

For some time during this process I was under the false impression that $\sigma = |h\rangle\langle h|$ gave rise to $C(\tau)$ that could had interesting phase behavior, of a form similar to that of Equation 2.34

[5]This analysis also applies to $\sigma = |h\rangle\langle h| U$ where U is any unitary matrix.

Using Equation 2.32, $\sigma = |h\rangle \langle h|$ yields:

$$\tilde{C}(\omega)_{hh} = \frac{(f^2-\omega^2)^2 + 2\theta^2(f^2+\omega^2) + \theta^4}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{hu} = \frac{(f^2-\omega^2)(\omega c k_x + i f c k_y) + \theta(-2f\omega c k_y + i c k_x(f^2+\omega^2)) + \theta^2(-c k_x \omega + i f c k_y) + \theta^3(i c k_x)}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{hv} = \frac{(f^2-\omega^2)(\omega c k_y - i f c k_x) + \theta(2f\omega c k_x + i c k_y(f^2+\omega^2)) + \theta^2(-\omega c k_y - i f c k_x) + \theta^3(i c k_y))}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{uh} = \frac{(f^2-\omega^2)(\omega c k_x - i f c k_y) + \theta(-2f\omega c k_y - i c k_x(f^2+\omega^2)) + \theta^2(-\omega c k_x - i f c k_y) + \theta^3(i c k_x))}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{uu} = \frac{f^2c^2k_y^2 + \omega^2c^2k_x^2 + 2\theta f c^2 k_x k_y + \theta^2 c^2 k_x^2}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{uv} = \frac{-(f^2-\omega^2)c^2k_xk_y - i\omega f c^2k^2 + \theta f^2(c^2k_x^2 - c^2k_y^2) + \theta^2 c^2 k_x k_y}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{vh} = \frac{(f^2-\omega^2)(\omega c k_y + i f c k_x) + \theta(2\omega f c k_x - i c k_y(f^2+\omega^2)) + \theta^2(-\omega c k_y + i f c k_x) - \theta^3(i c k_y)}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{vu} = \frac{-(f^2-\omega^2)c^2k_xk_y + i\omega f c^2k^2 + \theta f(c^2k_y^2 - c^2k_x^2) + \theta^2(c^2k_xk_y)}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

$$\tilde{C}(\omega)_{vv} = \frac{f^2c^2k_x^2 + \omega^2c^2k_y^2 - 2\theta f c^2 k_x k_y + \theta^2 c^2 k_y^2}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)}$$

Inspecting just one velocity-height correlation, we want to gain an understanding of the behavior at the special point $\omega = \pm f$, $k_x, k_y = 0$.

In truth, this analysis deserves a more precise mathematical approach than what I am qualified for. Therefore, the observations presented here are made with weaker mathematical grounding than those preceding them. Still, there is no doubt that interesting singular behavior exists in these functions. As in the previous section, $C(\omega)_{hu}$ will do a sufficient job of representing the behavior of the other velocity-height correlations.

$$\tilde{C}(\omega)_{hu} = \frac{(f^2-\omega^2)(\omega c k_x + i f c k_y) + \theta(-2f\omega c k_y + i c k_x(f^2+\omega^2)) + \theta^2(-c k_x \omega + i f c k_y) + \theta^3(i c k_x)}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)} \tag{2.35}$$

Just as the was done for the $\sigma = I$ case, the first means of taking this limit will be taken by setting $\omega = \pm$, then observing the behavior as $ck \to 0$. Doing this, the first order θ^0 term in

the numerator vanishes:

$$\tilde{C}(\omega)_{hu} = \frac{2\theta f^2(\mp ck_y + ick_x) + \theta^2 f(\mp ck_x + ick_y) + \theta^3(ick_x)}{(f^2+\theta^2)((c^2k^2+\theta^2)^2)}$$

This unusual behavior will be considered, and it is a consequence of taking the limit this way. We can show that under a certain set of assumptions, phase singularities are suppressed in all terms this way. We'll suppose that $ck \ll \theta$, and $\theta \ll f$. In a damped harmonic oscillator, this corresponds the the under-damped regime. Separating terms, in the numerator, and noting that the largest term in the denominator is $f^2\theta^4$

$$\tilde{C}(\omega)_{hu} \approx \frac{2\theta f^2(\mp ck_y + ick_x)}{f^2\theta^4} + \frac{\theta^2 f(\mp ck_x + ick_y)}{f^2\theta^4} + \frac{\theta^3(ick_x)}{f^2\theta^4}$$

It can be seen that even the highest order term becomes proportional to ck/θ^3, the signal is suppressed in this limit. θ must be nonzero for these derivations to be valid and possible, as it keeps the system from diverging in magnitude. So, this shows the $k_x, k_y = 0$ point does not divergence in any valid case.

Though it gives no indication of what happens at the origin of the k-plane, in the limit where $ck \ll f$, but $\theta \ll ck$–the under-damped regime–the highest order term in the denominator becomes:

$$\tilde{C}(\omega)_{hu} = \frac{2\theta f^2(\mp ck_y + ick_x) + \theta^2 f(\mp ck_x + ick_y) + \theta^3(ick_x)}{f^2c^4k^4}$$
$$\tilde{C}(\omega)_{hu} = \frac{2\theta(\mp ck_y + ick_x)}{c^4k^4} + \mathcal{O}(\theta^2) + ...$$

This does not suggest any divergences either, but in the under-damped case, there is a clear phase behavior that is complex conjugated when swapping ω_+ and ω_-. It is also worth noting that a change of $f \to -f$ does not alter this signal. For good measure, it is worth showing the formula in near the critically damped regime and ($\theta = ck$):

$$\tilde{C}(\omega)_{hu} = \frac{(\mp ck_y + ick_x)}{c^3k^3} + \mathcal{O}(1/ck) + ...$$

Returning to the general case, it should also be noted that the θ^0 term can make a significant contribution if a limit is taken differently. Focusing just on this term, we can

chose observe the system at $\omega = \pm\sqrt{c^2k^2/2 + f^2}$, then take $ck \to 0$. In this case, the equation becomes:

$$\tilde{C}(\omega)_{hu} = \frac{(c^2k^2/2)(\pm\sqrt{c^2k^2/2 + f^2}ck_x + ifck_y)}{(c^2k^2/2 + f^2 + \theta^2)((c^2k^2/2)^2 + 2\theta^2(c^2k^2(3/2) + 2f^2) + \theta^4)} + \mathcal{O}(\theta) + ...$$

Assuming the under-damped condition $\theta \ll ck \ll f$, this becomes:

$$\tilde{C}(\omega)_{hu} = \frac{(\sqrt{c^2k^2/2 + f^2}ck_x + ifck_y)}{f^2(c^2k^2/2)} + \mathcal{O}(\theta) + ...$$

From here a clear diverging phase singularity can be seen, indicating that the value at the origin when $\omega \to f$ is dependent on how that limit is taken. That is in fact a reasonable definition of a singularity: its value depends on how it is approached. This is not only a phase-singularity, but also a point of undefined magnitude. It seems that the limit could be zero, or could be infinite. To reiterate, a proper analysis of this function is of this function and its topological structure is needed. Without probing any further, the time-dependent covariance $C(\tau)$ will be analyzed.

Making use of Equation 2.20,

$$\begin{aligned} C(\tau) = &- \Theta(\tau)e^{L\tau}\mathcal{C} - \Theta(-\tau)\mathcal{C}e^{-L^\dagger\tau} \\ &\mathcal{C} \equiv vec^{-1}\big((L^* \oplus L)^{-1}vec(\sigma\sigma^\dagger)\big) \end{aligned} \tag{2.20}$$

We'll first separate out the drag term by defining the anti-Hermitian part of L $\mathbb{A}(L) = (L - L^\dagger)/2 \to \mathbb{A}(L^\dagger) = -\mathbb{A}(L)$, and the Hermitian part of L, $\mathbb{H}(L) = (L + L^\dagger)/2 \to \mathbb{H}(L^\dagger) = \mathbb{H}(L)$. Holding on to the notation $\mathbb{A}(L)$, we can quickly identify $\mathbb{H}(L) = -\theta I$

$$C(\tau) = - \Theta(\tau)e^{-\theta\tau}e^{\mathbb{A}(L)\tau}\mathcal{C} - \Theta(-\tau)e^{\theta\tau}\mathcal{C}e^{+\mathbb{A}(L)\tau}$$

$\mathcal{C}$ was computed in Mathematica and checked with the standard Lyapunov Equation.

$$
\begin{aligned}
\alpha &\equiv -4\theta((f^2+c^2k^2)^2+5\theta^2(f^2+c^2k^2)+\theta^4) \\
\mathcal{C}_{hh} &= \frac{(2f^4+c^4k^4)+\theta^2(10f^2+6c^2k^2)+8\theta^4}{\alpha} \\
\mathcal{C}_{hu} &= \frac{ic(-fk_y(c^2k^2-2f^2)+k_x\theta(4f^2+c^2k^2)+2fk_y\theta^2+4k_x\theta^3)}{\alpha} \\
\mathcal{C}_{hv} &= \frac{ic(fk_x(c^2k^2-2f^2)+k_y\theta(4f^2+c^2k^2)-2fk_x\theta^2+4k_y\theta^3)}{\alpha} \\
\mathcal{C}_{uu} &= \frac{c^2(k_x^2(f^2+c^2k^2)+3k_y^2f^2+6fk_xk_y\theta+4k_x^2\theta^2))}{\alpha} \\
\mathcal{C}_{uv} &= \frac{c^2(k_xk_y(c^2k^2-2f^2)-3f(k_x^2-k_y^2)\theta+4k_xk_y\theta^2))}{\alpha} \\
\mathcal{C}_{vv} &= \frac{c^2(k_y^2(f^2+c^2k^2)+3k_x^2f^2+6fk_xk_y\theta+4k_y^2\theta^2)}{\alpha} \\
\mathcal{C}_{uh} &= -\mathcal{C}_{hu},\ \mathcal{C}_{vh} = -\mathcal{C}_{hv},\ \mathcal{C}_{vu} = -\mathcal{C}_{uv}
\end{aligned}
$$

The Unitary exponential $e^{\mathbb{A}\tau}$ was derived with change-of-basis matrices. To avoid clutter one can define $r = \sqrt{f^2+c^2k^2}$

$$
e^{\mathbb{A}(L)t} = \begin{bmatrix}
\frac{f^2+c^2k^2\cos(rt)}{r} & \frac{-ifck_y(1-\cos(rt))-ick_xr\sin(rt)}{r} & \frac{ifck_x(1-\cos(rt))-ick_yr\sin(rt))}{r} \\
\frac{ifck_y(1-\cos(rt))-ick_xr\sin(rt))}{r} & \frac{c^2k_y^2+(f^2+c^2k_x^2)\cos(rt)}{r} & \frac{-c^2k_xk_y(1-\cos(rt))-fr\sin(rt)}{r} \\
\frac{-ifck_x(1-\cos(rt))-ick_yr\sin(rt)}{r} & \frac{-c^2k_xk_y(1-\cos(rt))+fr\sin(rt)}{r} & \frac{c^2k_x^2+(f^2+c^2k_y^2)\cos(rt)}{r}
\end{bmatrix}
$$

The resulting expression, when these terms are multiplied through is far too large to fit in this document. However, its components are either all real or all imaginary. Given for only positive τ, we have

$$
\begin{aligned}
C(|\tau|)_{hu} = e^{-\theta\tau}(&i(2cf^2(-f^3k_y+2c^2fk_yk^2-2f^2k_x\theta+c^2k_xk^2\theta-fk_y\theta^2-2k_x\theta^3) \\
&-c^3k^2(5f^3k_y-c^2fk_yk^2+7f^2k_x\theta+c^2k_xk^2\theta+2fk_y\theta^2+4k_x\theta^3)\cos(\sqrt{c^2k^2+f^2}t) \\
&+c^3k^2\sqrt{f^2+c^2k^2}(f^2k_x+c^2k_xk^2+3fk_y\theta+4k_x\theta^2)\sin(\sqrt{c^2k^2+f^2}t))) \\
&/(4\theta(f^2+c^2k^2)(f^4+c^4k^4+5c^2k^2\theta^2+4\theta^4+f^2(2c^2k^2+5\theta^2)))
\end{aligned}
$$

This is far more complicated but not much more interesting than the case of $\sigma = I$.

Due to the physical justification for $\sigma = |h\rangle \langle h|$, and a miscalculation that suggested phase-singular behavior may exist in this case, it was used for the numerical simulations described in the next section. $C(\tau)$ was not directly calculated an compared to this result. Instead, statistics from simulations were compared to $\tilde{C}(\omega)$, given in

Chapter Three

Numerical Simulation

Two numerical simulations were used to test these covariance functions. These were intended to serve as a intermediate step between the theoretical formulas for these unequal time field correlations and real climate data. First, the GCM climate modeling program[1], a program typically used for direct statistical simulation was adapted to numerically simulate the evolution of stochastically driven non-linear flows on a model Earth, with a different Coriolis effect at the poles and Equator. As the data produced did not yield clear results, a very simple numerical simulation was created in Python to simulate stochastically driven linear shallow water waves for a constant Coriolis parameter. This basic model produced data that agreed well with the results of the previous section. Since this model is far simpler, it will be presented first, though it was primarily created to verify to diagnose issues with the strange data produced with GCM data. Before that, a description of how climate data was analyzed, as applies to both simulated and real data will be presented.

3.1 General Method for Analyzing Climate Data

In the theory section, we defined the Unequal time Covariance by $C(t, t+\tau) = E[|q(t+\tau)\rangle \langle q(t)|]$ where E denotes an expected value of the outer product, taken over all the possible paths

[1]The application "GCM" is available for OS X 10.9 and higher on the Apple Mac App Store at URL http://appstore.com/mac/gcm

of $|q(t)\rangle$:

$$E[|q(t+\tau)\rangle\langle q(t)|] = \sum_{\text{all possible } |q\rangle} |q(t+\tau)\rangle\langle q(t)| \cdot (\text{probability of path } |q\rangle)$$

With the real climate, one cannot know all the possible ways the system could have evolved. Instead the only available information is the one way the system did evolve, so to find $C(t, t+\tau)$ a different approach is necessary. As a replacement for this statistical average, a time-average is used. This only works if initial conditions are not a factor, so that all time t can be treated equally, and τ is far more important. This time-average takes the form:

$$C(\tau) = \frac{1}{2T}\int_{-T}^{T} |q(t+\frac{\tau}{2})\rangle\langle q(t-\frac{\tau}{2})|\, dt$$

where $2T$ is the duration of the evolution of $|q(t)\rangle$. Precise mathematical reasoning for when and why these two conceptions of an average value are the same for large T will not be provided here. For a function of variable $|q(t)\rangle$, as is shown in the appendix of Skitka et al.[20] (though adapted to outer products), by Fourier transforming these vectors over the frequency domain

$$|q(\omega)\rangle = \int_{-\pi/T}^{\pi/T} e^{-i\omega t}\,|q(t)\rangle\, d\omega$$

one can relate:

$$C(\tau) = \frac{1}{4\pi}\int_{-T}^{T} e^{i\tau\omega}\,|q(\omega)\rangle\langle q(\omega)^*|\, d\omega$$

$$\tilde{C}(\omega) = |q(\omega)\rangle\langle q(\omega)| = |q(\omega)\rangle\langle q(-\omega)^*|$$

This generalizes very well to the case where the evolving vector is a function of many variables, like in $|q(x, y, t)\rangle$.

$$\tilde{C}(\omega) = |q(k_x, k_y, \omega)\rangle\langle q(k_x, k_y, \omega)| = |q(k_x, k_y, \omega)\rangle\langle q(-k_x, -k_y, -\omega)^*| \tag{3.1}$$

From here, there is a simple mechanism to find $\tilde{C}(\omega)$: Fourier transform the data, and take its outer product with itself. The only issue is that the dimensions of T, L_x and L_y may not be large enough for this time/space-averaged $C(\omega)$ to approach the statistical-averaged $\tilde{C}(\omega)$.

To combat this, $|q(k_x, k_y, \omega)\rangle \langle q(k_x, k_y, \omega)|$ can be averaged over some window $k_y \pm \Delta k_y$, $k_x \pm \Delta k_x$, to smooth out noise. This smoothing is done via a Gaussian average integral, represented here in terms of one variable.

$$C(\omega)_{smooth} = \frac{1}{\sqrt{2\pi\sigma^2}} \int_{-\pi/T}^{\pi/T} \exp(\frac{(\omega - \omega')^2}{2\sigma^2}) C(\omega') d\omega'$$

This operation was applied to each dimension of the data independently. From there, the complex data was plotted on the k_x, k_y plane for a given f and ω.

3.2 Simple Numerical Shallow Water Simulation

The simpler numerical simulation used the non-dimensional linear shallow water model (Equation 1.10), with added noise of the form $\eta(t) = |h\rangle \langle h|\dot{W}(t)\rangle$, and damping θ:

$$\begin{aligned} \frac{\partial h}{\partial t} &= -c\frac{\partial u}{\partial x} - c\frac{\partial v}{\partial y} + \eta(t, x, y) - \theta h \\ \frac{\partial u}{\partial t} &= f_0 v - c\frac{\partial h}{\partial x} - \theta u \\ \frac{\partial v}{\partial t} &= -f_0 u - c\frac{\partial h}{\partial y} - \theta v \end{aligned} \tag{3.2}$$

c was set to 1, and f_0 was set to ± 1. A numerical simulation was performed on a 400×400 grid. $\eta(t)$, was set to be a real-valued discrete white noise of the type described in the begging of Section 2.1. As this thesis is meant to assume no experience with numerical techniques, a description of this simulation and an introduction to elementary integration techniques for partial differential equations will be given. For the experienced reader, an Euler integration scheme was used with finite-difference spatial derivatives, periodic boundary conditions, and grid resolution was set at $\Delta x = \Delta y = 1, \Delta t = 0.05$. All fields were initialized to 0. An important discussion of $\eta(t, x, y)$, follows that introduction.

Basic Simulation of a Partial Differential Equation

Numerical simulation is process of discretizing continuous equations. To do this, at any given t, the continuous function $h(x, y, t)$ is replaced with a two-dimensional grid of values

$h_{x_i,y_j,t}$. The same pertains to u and v. The indices i and j span the dimensions of the grid, in this case 0 to 400, making $x_i = i\Delta x$. The set step size Δx, and Δy were both set to 1. To compute a spatial derivative in at each point in time, a finite-difference approximation is used:

$$\frac{\partial h(x,y,t)}{\partial x} \approx \frac{h_{x_{i+1},y_j,t_i} - h_{x_{i-1},y_j,t_i}}{2\Delta x}$$

Periodic boundary conditions were implemented for both x and y, so that:

$$\frac{\partial h(x_0,y,t)}{\partial x} \approx \frac{h_{x_1,y_j,t_i} - h_{x_{399},y_j,t_i}}{2\Delta x}$$

The time-derivative has to be must be treated differently. Unlike it space, where the field values at all points were known, and the derivative needed to be found, in this case we know the derivative (the right-hand-side of Equation 3.2), and the field value must be found. This is done with by rearranging a finite difference formula, giving the 'forward-Euler' method:

$$h(t_{k+1},x_i,y_j) = h(t_k,x_i,y_j) + \frac{\partial h(t_k,x_i,y_j)}{\partial t}\Delta t$$

Here, $\frac{\partial h(t_k,x_i,y_j)}{\partial t}$ is the right-hand-side of the relevant equations. These approximations to true integration and differentiation are improved by smaller steps in time and space, but those require more time for a computer to run. What is crucial is that the behaviors of interest happen on a time and spatial scale greater than those step-sizes. Otherwise, these techniques no longer properly approximate derivatives, yielding nonphysical behavior. Proper analysis of the equations can yield appropriate step-sizes, but often the nonphysical behavior is fairly obvious, and trial and error works well. There is a wealth of more advanced techniques for numerically 'solving' differential equations, though they are largely based on the same principles. In this instance the simplest method was sufficient.

Random Noise

Due to the finite grid resolution Δx and Δy, a finite random noise vectors $\eta(x_i,y_j,t_k)$ that are totally independent for different x_i, y_j would disrupt the finite difference method, as

the derivative approximations would no longer be valid. Therefore $\eta(x_i, y_j, t_k)$ should be self-similar on a spatial scale $\sim \Delta x, \Delta y$, but random on a spatial scale any greater. This was done by generating random noise vectors in k-space $\eta(k_{x,i}, k_{y,j}, t_k)$, just like those in the theoretical framework, only within a set window $|k_x| < \frac{1}{\Delta x}$, then using the discrete Fourier transform to bring these back into x and y. After computing generating a random real number $\eta(k_{x,i}, k_{y,j}, t_k)$ from a normal distribution with standard deviation 1.4, then the grid was multiplied by a by separate normal distributions centered at $k_x = 0$ and $k_y = 0$ with standard deviation equal to one sixth the grid size (67) producing a square window for k_x and k_y, that artificially strengthens the signal around $k = 0$.

Hue-Saturation Plotting for Complex Numbers in 2D

The typical way of graphing a real function of two variables $f(x, y)$ is either with a surface plot or with a colormap. Between two bounds, the value of a function is linearly mapped to a color, and those colors become an image. For positive data, white = 0 → gray → black = max is decently intuitive. If a function can be negative, it is nice to have a symmetry between positive and negative values, blue = min → white = 0 → red = max is a nice choice. An example of a blue–white–red ('bwr') colormap as generated by this numerical simulation in is given in Figure 3.4.

The 'bwr' colormap allows for the strength of a signal (its magnitude) to be mapped to the strength of a color (its saturation), while the quality of a signal (its sign) is associated with a specific hue (red or blue). It could be taken for granted that this colormap system provides a visually intuitive representation of how decreased magnitude makes sign becomes less important, with very light blue being pretty similar to very light pink

Complex functions are not afforded this luxury. Since complex phase is so important to this data, a visually clear representation of complex numbers is necessary to extract the optimal amount of information from potentially noisy data. two-dimensional vector plots are common, as the real and imaginary parts can be mapped to their components. The issue is

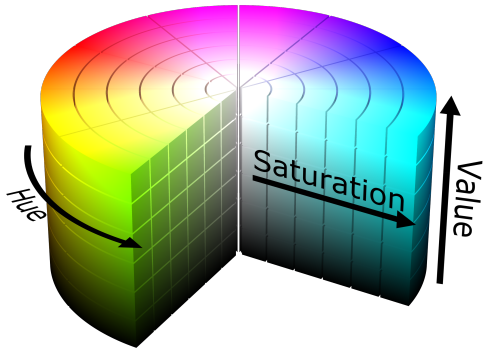

Figure 3.1 The Hue-Saturation-Value Color-Space

that for a 400×400 grid of data, not all of these vectors can be presented: either certain points must be singled out, or the data must be averaged, which could misconstrue the noise inherent to it. Seeing as color is represented by three three channels (RGB) there is more information to be carried in color than just a single dimension. The simplest answer, mapping real and imaginary parts directly to two of these channels yields visually perplexing results. Instead, the Hue-Saturation-Value color-space, represents color on a cylinder, with the azimuthal angle representing the Hue, or quality of the color, height representing darkness. Just like how the complex plane does not have a defined phase at the origin, the HSV cylinder does not have a well defined hue at the center, where it is white. This can be seen in Figure 3.1. The 2D complex number plots make use of a the top of this cylinder, where the color is at full value. For a complex number $re^{i\theta}$ we have $r \to$ saturation, $\theta \to$ hue. The more typical vector plot is compared with the Hue-Saturation (HS) plot of $f(x, y) = x + iy$ in Figures 3.2 and 3.3. As a general convention, blue will always mean positive real numbers, implying yellow-green is negative. and moving with θ counterclockwise around the complex plane, the sequence of colors colors is red-green-blue like RGB data. A rotation of the HS complex color-wheel corresponds to to multiplication by a constant phase factor, while reversing its direction corresponds to complex conjugation. Plotting in color-spaces like this has been used in complex analysis for some time, but it unfortunately does not seem to be an established convention in physics[21].

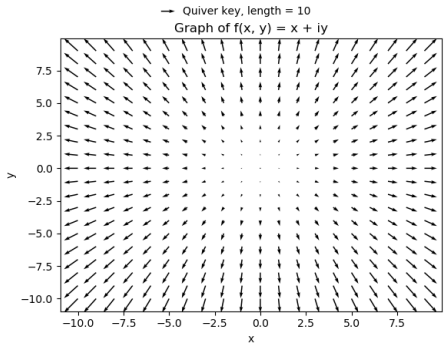

Figure 3.2 Vector plot of $f(x, y) = x + iy$. The x-direction of the vector is mapped to the real part of the function, while the y-direction is mapped to the Imaginary part.

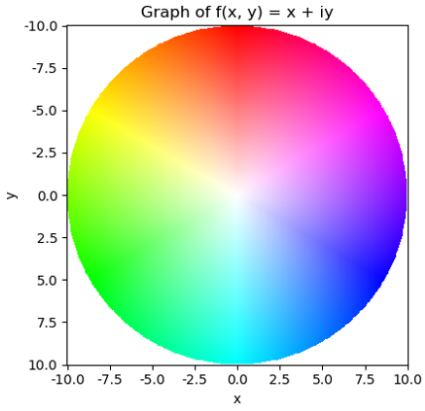

Figure 3.3 Hue-Saturation plot of $f(x, y) = x + iy$. With $re^{i\theta} = x + iy$ The hue, or color quality is mapped to the of the complex number θ, while the saturation: the strength of the color is mapped to the magnitude r of the complex number. $r_{\max}$, where the color is fully saturated, is set to 10, and higher magnitudes are not plotted.

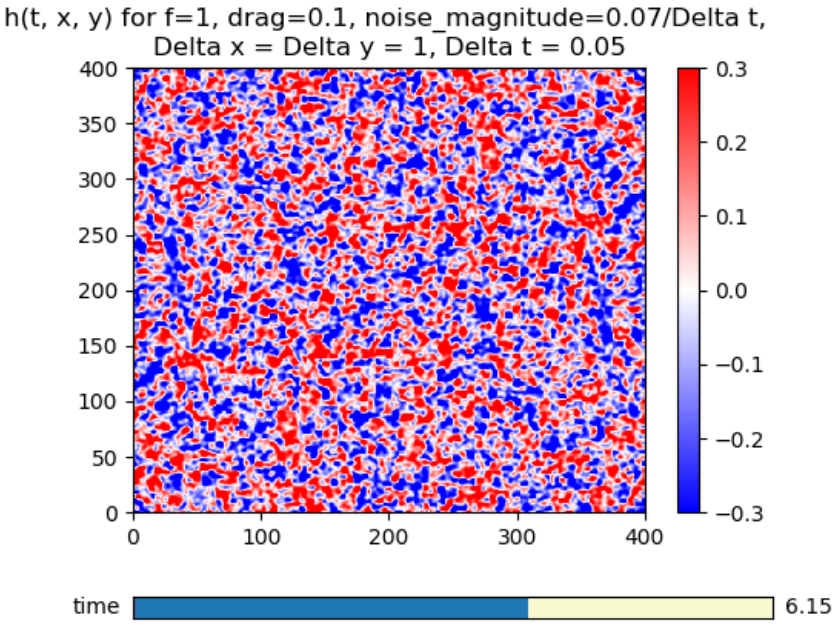

Figure 3.4 The height field in a direct numerical simulation of the linear shallow water equations, a real valued function shown using a 'bwr' colormap

3.2.1 Results

By integrating the linear shallow water equations driven by noise in h, clear phase singularities could be found in the covariance. The following analysis focuses exclusively on the $\tilde{C}(\omega)_{hu}$ covariance component, as all velocity-height correlations take a similar form. The theory predicted:

$$\tilde{C}(\omega)_{hu} = \frac{(f^2-\omega^2)(\omega c k_x + i f c k_y) + \theta(-2f\omega c k_y + i c k_x (f^2+\omega^2)) + \theta^2(-c k_x \omega + i f c k_y) + \theta^3(i c k_x)}{(\omega^2+\theta^2)((c^2k^2+f^2-\omega^2)^2 + 2\theta^2(c^2k^2+f^2+\omega^2)+\theta^4)} \tag{2.35}$$

According to the derived linear shallow water covariance (2.35), we expect to see at $\omega = \pm 1$, and $f = \pm 1$, the four plots in Figure 2.35. A few features are worth noting. First, the plots at diagonals from each other have the direction of phase-rotation and are almost identical to each other, reflecting the system's symmetry under time, and Coriolis parameter reversal: $f \to -f, \omega \to -\omega$. They do become truly identical when damping $\theta \to 0$. It is also worth noting that there is no true phase singularity in the center, though this is permitted by very

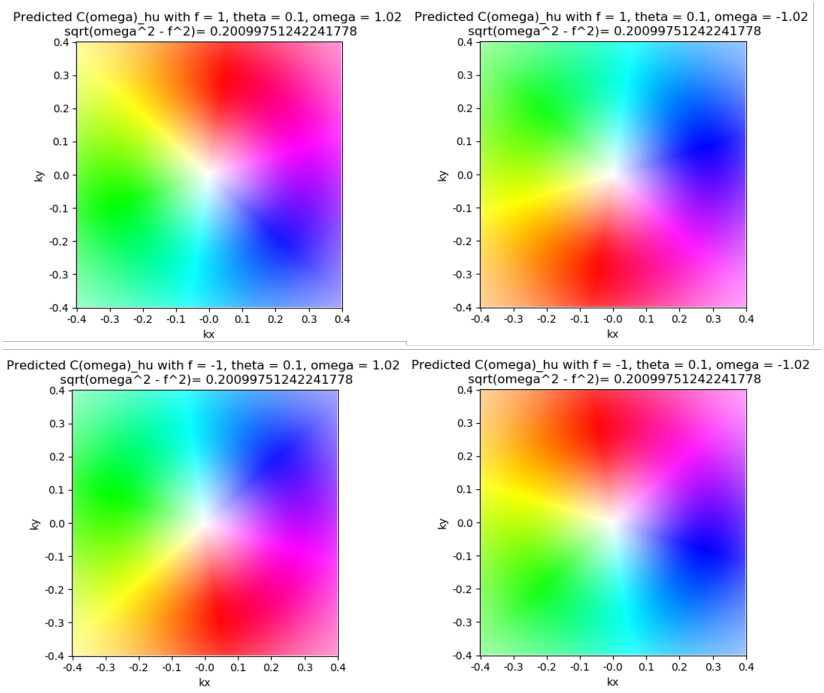

Figure 3.5 $\tilde{C}(\omega)_{hu}$ predicted by the Equation 2.35 derived in the previous section. The left has omega slightly larger than $|f|$, while the right has ω slightly smaller than $-|f|$. The sign on f is divided top to bottom.

small θ and ω very close to f. These match the conditions in the numerical simulation, which unfortunately required a large damping parameter $\theta = 0.1$ to not have numerical divergences and errors. However, $\tilde{C}(\omega)_{hu}$ allows for other interesting behaviors, like those in Figure 3.6. Seeing that c is set to 1, the condition for resonance is given by $k^2 = \omega^2 - f^2$. Therefore, above each image this the radius of this resonant circle, is written so that it may be compared to the region of high saturation in the figure. Though the signal is not as clear due to the high damping in Figure 3.5, it is very apparent in the low damping case shown in 3.6. In Figure 3.7, it can be seen that just as in Figure 3.5, $\omega \to -\omega$ corresponds to a rotation about the k_y axis, with some minor corrections. I believe that these corrections are due to damping. It should be noted that while the linear shallow water equations are rotationally symmetric with respect to x and y, these correlations do not necessarily have to be, a spatial dimension has been singled out by choosing x-velocity $\tilde{C}_{hu}$. While these results all strongly suggest that phase singularities can be detected from correlations in systems like

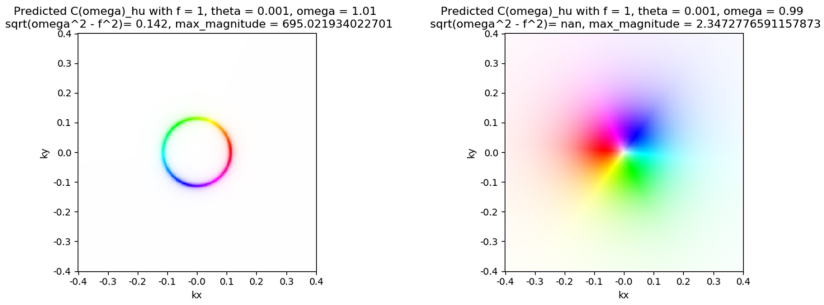

Figure 3.6 A comparison of the covariance signal for $\omega = f + \pm 0.01$, all else equal. Note the striking change in maximum magnitude in the titles. The only signal that does make it from through to the gap between Poincairé bands can be thought of as the bands' signal 'bleeding through' due to damping-related smoothing. The signal change in sign is due to the term that suppresses signals at $\omega = f$ in Equation 3.5 (in some limits): $f^2 - \omega^2$.

this one, there is some disagreement among them, and I have little doubt that these are due to fixable errors in my implementations. Most obviously, the values of ω or k have to be wrong in 3.7, as they clearly form resonance circles with complex phases that match those of the Poincairé bands. It cannot be that $|\omega| < |f|$ in these instances.

I am quite confident that this is due to a mistake in my use of discrete Fourier transformations in my code. Additionally, the negative ω covariances in the numerical data seem to be related to their theoretical counterparts via complex conjugation. (Note that the color-wheel rotates in the opposite direction). This is not the whole story either, as the real-valued red and cyan sections that should stay in place do not. I suspect that this is an error in implementation or derivation. There were many steps where rearranging axes and taking complex conjugates would change the phase of the answer. Though these minor details need to be worked out, the direct numerical simulation data certainly supports the asserting that these phase singularities can be observed from $\tilde{C}(\omega)$.

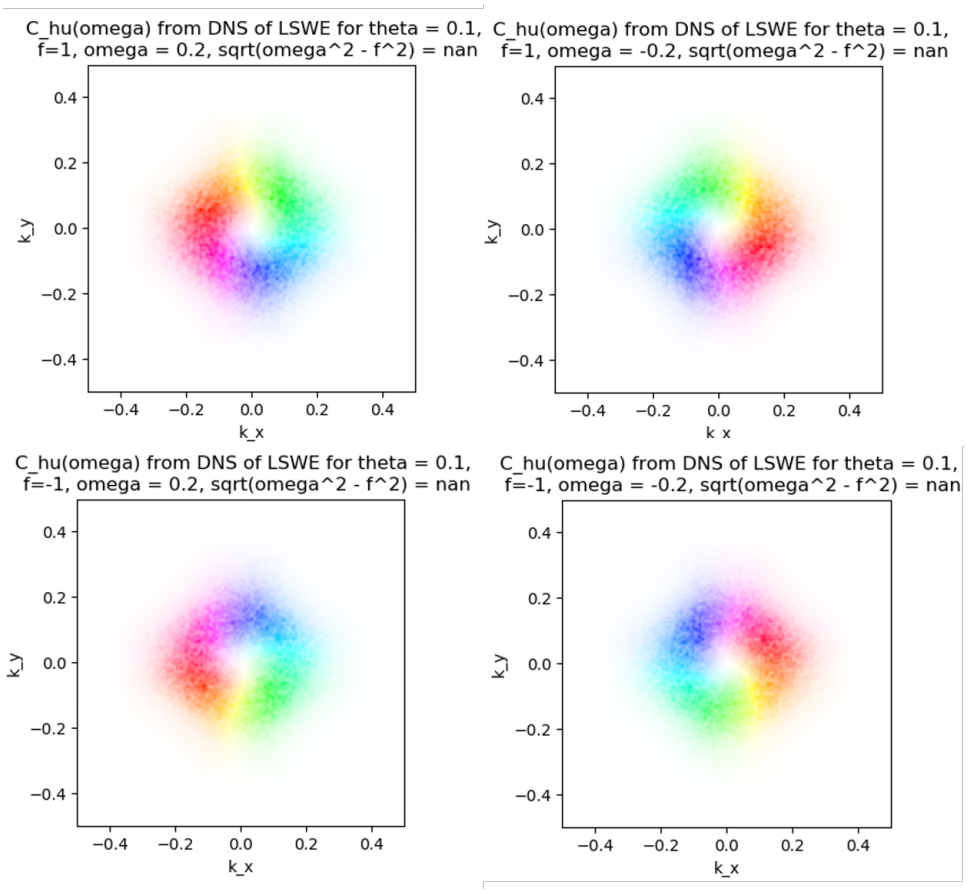

Figure 3.7 $\tilde{C}(\omega)_{hu}$ was found by time-integration methods for a direct numerical simulation (DNS) of the damped linear shallow water equations (LSWE). These should be compared with the theoretical predictions by the Equation 2.35, shown in Figure 3.5. The value of omega shown is without a doubt incorrect, and probably the result of not properly non-dimensionalizing. Left has omega slightly larger than $|f|$, while the right has ω slightly smaller than $-|f|$. The sign on f is divided top to bottom.

3.3 Planetary Atmosphere Simulation with GCM

3.3.1 The GCM Program

GCM provided data that was much more complicated, and a far better model of Earth's climate than the linear shallow water simulation. For one, the equations it solves are non-linear, but they are also solved on a grid that mimics the Earth, with a changing Coriolis parameter. Similarly the unequal-time covariance for data it produced was not clear enough for the subtleties of these equations to truly manifest. Therefore, only a cursory description will be provided, but a far better description can be found in *Planetary Atmospheres as Nonequilibrium Condensed Matter*[22].

For this project, the GCM was used to simulate a two-layer nonlinear atmospheric model, where a warm upper layer sits above a cooler lower layer. The interface between these layers may move up and down, allowing for temperature to change, much like h in the shallow water system. The two-layer model is more complicated, but it does contain the shallow water model, as temperature differences drive the fluid to motion, which is deflected by the Coriolis force. This two layer model is given by equations that can more precisely understood in [20]:

$$\begin{aligned}
\dot{\bar{q}} &= J[\bar{q},\bar{\psi}] + J[\hat{q},\hat{\psi}] - F[\hat{q},\hat{\chi}] - J[\hat{\delta},\hat{\chi}] - F[\hat{\delta},\hat{\psi}] \\
\dot{\hat{q}} &= J[\hat{q},\bar{\psi}] + J[\bar{q},\hat{\psi}] - F[\bar{q},\hat{\chi}] \\
\dot{\hat{\delta}} &= J[\bar{q},\hat{\chi}] + F[\hat{q},\bar{\psi}] + F[\bar{q},\hat{\psi}] - \nabla^2(\hat{K} + C_p B\bar{\theta}) \\
\dot{\bar{\theta}} &= J[\bar{\theta},\bar{\psi}] + J[\hat{\theta},\hat{\psi}] - F[\hat{\theta},\hat{\chi}] \\
\dot{\hat{\theta}} &= J[\hat{\theta},\bar{\psi}] + J[\bar{\theta},\hat{\psi}] - F[\bar{\theta},\hat{\chi}] + \bar{\theta}\hat{\delta}
\end{aligned}$$

Here each variable A is split into a baroclinic and barotropic parts $A = \hat{A} + \bar{A}$, where δ is the velocity divergence, q is a vorticity, θ is a (potential) temperature. K is an energy density with a more complex form than is written here. The Jacobian $J[A,B] \equiv \hat{r} \cdot (\nabla A \times \nabla B)$, while flux-divergence $F[A,B] \equiv \nabla \cdot (A \nabla B)$. These equations were modified to include a

stochastic driving force along barotropic potential temperature, damping and friction with in the bottom layer caused by contact with the planet for their use in the simulation.

Adding to its complication GCM models a full Earth, and the Coriolis parameter $f = 2\Omega \sin(\theta)$ it uses is not approximated. This was a major challenge, since the derivations

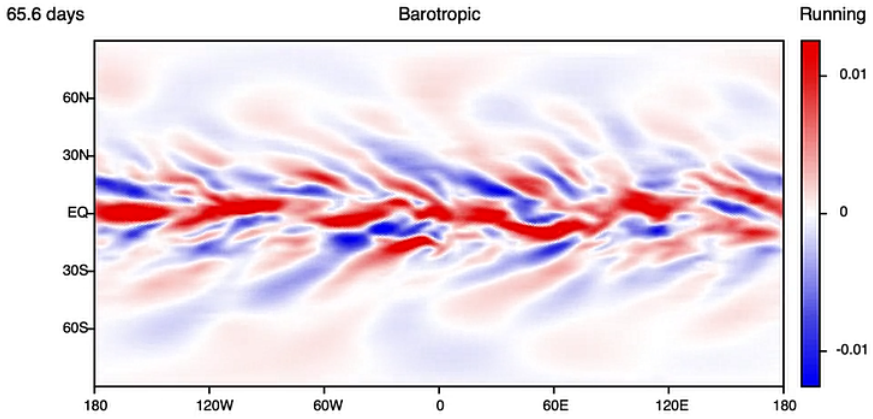

Figure 3.8 A snapshot of the (Barotropic) Temperature data produced by GCM.

that make up this thesis required that L has a constant Coriolis parameter. Therefore, the primary technique used was to slice the data into horizontal strips, where the Coriolis parameter can be approximated as constant, and look for phase singularities in each slice. The other techniques were identical to those outlined in the previous section. The issue with slicing the data vertically is is that as height of the strip L_y decreases, the resolution of the image also decreases. In truth, there was a decent fix for this issue: to set a Gaussian Window around each latitude y, and perform a Fourier transform for each y, then average them, again by convolution with a Gaussian. This was attempted, and certain can be attempted in the but an implementation that was computationally realistic for my PC was not produced.

Other complications, like strangely strong signals at constant k_x added a great deal of uncertainty in the analysis of this data. There was a strong possibility that unusual artifacts from compression, related to how the data went from GCM to Python, were the source of these structures. The simple, numerical simulation used in the previous section was originally created to make sure the fault was not in my implementation.

3.3.2 Analysis of GCM Data

A good analysis of GCM data was mainly hindered by binning along $f_0 \pm \Delta f$ bands. Clearest results where seen when the data was split into six bands, and the bands $0°$ to $30°$ latitude and $-30°$ to $0°$ has the clearest results. In this range, the small k_x signal did not dominate as much as other ranges, but it suffers from low k_y resolution. A few features of this data

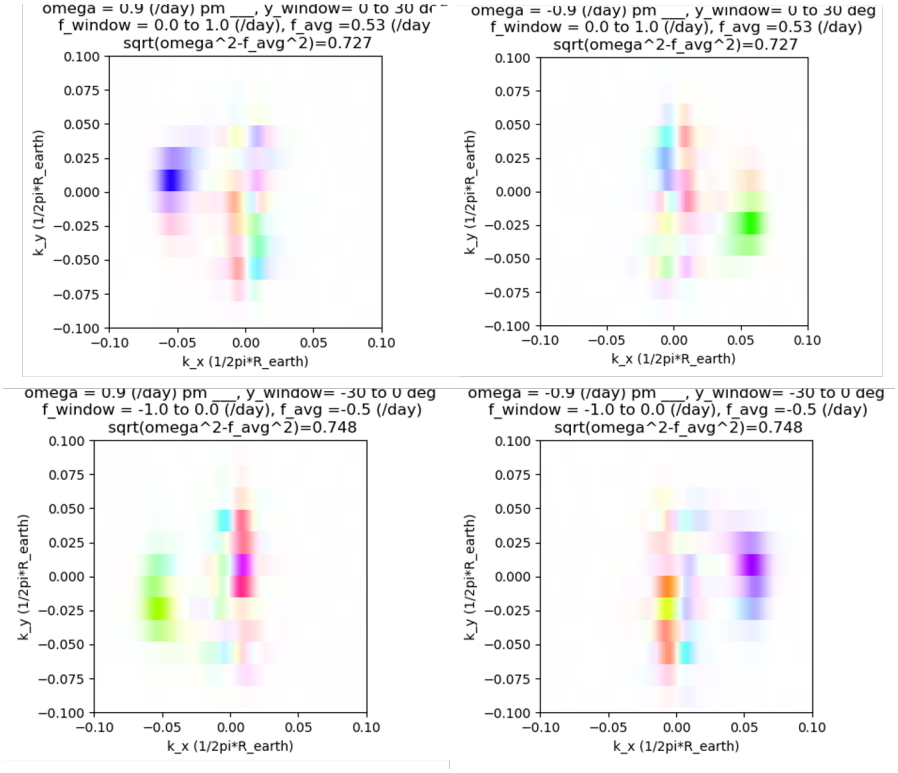

Figure 3.9 GCM h, u Fourier transformed covariance at $\omega = \pm 0.9$, and average $f = \pm 0.5$.

stand out, first, singularities near the center are not quite resolvable. However, a part of the resonant circle is resolvable, at $\pm k_x$. I believe this k_x preference is the signal of equatorially trapped waves, which propagate east. This k_x skewness can be seen for either $\pm f$, indicating a strong signal for equatorial waves. Since $sign(\omega) = -sign(k_x \text{ preference})$, this signal indicated a negative phase velocity, which would seem to indicate east-to-est propagation, but the group velocity may have opposite sign. There is also a slight preference for k_y depending on whether $sign(\omega) = sign(f)$. One possible explanation for this is that when f

is significantly less that ω (as could not be resolvable with this course grid), $\omega c k_x + i f c k_y$ does not represent a simple singularity, and is instead skewed real or imaginary. However, this behavior does not match up properly with Equation 2.35. We can also note that the strong signals are real-valued along the resonant circle. I am unsure how to explain this. In all, more precise data, processing with a finer grid size is necessary for clear interpretation of GCM data.

Chapter Four

Conclusions

4.1 Summary of Results

In this thesis, it was shown that the phase singularities in k-space Poincairé Inertia-Gravity waves, which have been shown by Deplace et al. [1] provided a novel, 'topological' explanation for equatorial geophysical waves, translate directly to measurable signals in the Fourier-transformed, *unequal*-time cross-correlations of relevant fields. It was shown that within the idealized linear stochastic shallow water model that for height h and velocity u, the correlation function $\tilde{C}(\omega) \equiv \int_{-\infty}^{\infty} e^{-i\omega\tau} h(t+\tau, k_x, k_y) u^*(t, k_x, k_y)] d\tau$, contains phase singularities analogous to those of the Poincairé waves. This result was supported by correlations of data produced by a highly idealized numerical simulation of the linear shallow water equations, and very loosely suggested by data taken from a nonlinear simulation of a global atmospheric model produced with the GCM program.

4.2 Implications and Future Work

This work was completed with the overarching goal of observing these topological features in statistical correlations of genuine climate data, which could provide a new technique for predicting the evolution of the Earth's climate. The obvious next step is to begin downloading

climate data, averaging cross-correlations in frequency-space, and looking for singularities.

This seems on the horizon, but a few issues should be addressed first.

4.2.1 Refining

As is shown in Section 3.2.1, there errors and inconsistencies in this work that I need to address. There is certainly evidence that these errors do not topple the fundamental implications of this work, but they need to be addressed before one can consider these well substantiated results.

4.2.2 Another Try with GCM Data

It is still very possible to get substantial results from the GCM data, but a more subtle technique for treating the cross-correlation as function of y, than the simple slice-technique I used, should be implemented. One was already suggested, but some more time needs to be put into it. GCM can be a very helpful intermediate step between simple linear shallow water simulations and the real climate data.

4.2.3 Precise Examination of Complex Functions in $\tilde{C}(\omega, k_x, k_y)$

As an advanced undergraduate, though I will graduate and begin my graduate studies very soon, I do not yret have the mathematical background to treat these topological concepts with a necessary degree of rigor. I have pointing out that the functions derived have interesting properties when taking limits, and have phase singularities much like those of the topological Poincair/'e waves, but I do not yet have the tools to properly analyze them.

Special consideration should be payed to the zero-drag limit of Equation 2.35, as it is in some sense, *the* result of this thesis:

$$\tilde{C}(\omega, \theta = 0)_{hu} \frac{(f^2 - \omega^2)(\omega c k_x + i f c k_y)}{\omega^2 (c^2 k^2 + f^2 - \omega^2)^2}$$

I know it has a nontrivial phase, and I know it has strange behavior in the limit of small ck when $|\omega|$ is very close to $|f|$. I do not know much more.

4.2.4 Application to Climate Data

With those other steps out of the way, there will be a decently well developed apparatus for detecting these topological waves. It would be nice to truly find them.

REFERENCES

[1] Pierre Delplace, J. B. Marston, and Antoine Venaille. "Topological Origin of Equatorial Waves". In: *Science* 358.6366 (Oct. 2017), pp. 1075–1077. ISSN: 1095-9203. DOI: 10.1126/science.aan8819. URL: http://dx.doi.org/10.1126/science.aan8819.

[2] Dale B. Haidvogel et al. "Numerical modelling in a multiscale ocean". In: *Journal of Marine Research* 75.6 (Nov. 1, 2017), pp. 683–725. ISSN: 0022-2402. DOI: doi:10.1357/002224017823523964. URL: https://www.ingentaconnect.com/content/jmr/jmr/2017/00000075/00000006/art00002.

[3] D. Vanderbilt. *Berry Phases in Electronic Structure Theory: Electric Polarization, Orbital Magnetization and Topological Insulators*. Cambridge University Press, 2018. ISBN: 9781107157651. URL: https://books.google.com/books?id=vRNwDwAAQBAJ.

[4] S.M. Girvin and K. Yang. *Modern Condensed Matter Physics*. Cambridge University Press, 2019. ISBN: 9781107137394. URL: https://books.google.com/books?id=ymYNuQEACAAJ.

[5] K. v. Klitzing, G. Dorda, and M. Pepper. "New Method for High-Accuracy Determination of the Fine-Structure Constant Based on Quantized Hall Resistance". In: *Phys. Rev. Lett.* 45 (6 Aug. 1980), pp. 494–497. DOI: 10.1103/PhysRevLett.45.494. URL: https://link.aps.org/doi/10.1103/PhysRevLett.45.494.

[6] Delft Institute of Technology. *EdX Course: Topology in Condensed Matter: Tying Quantum Knots*. Available at https://topocondmat.org/w3_pump_QHE/QHEedgestates.html (2020/05/11).

[7] David K Ferry. "The quantum Hall effect". In: *Transport in Semiconductor Mesoscopic Devices*. 2053-2563. IOP Publishing, 2015, 6-1 to 6–26. ISBN: 978-0-7503-1103-8. DOI: 10.1088/978-0-7503-1103-8ch6. URL: http://dx.doi.org/10.1088/978-0-7503-1103-8ch6.

[8] Geoffrey K. Vallis. *Essentials of Atmospheric and Oceanic Dynamics*. Cambridge University Press, 2019. DOI: 10.1017/9781107588431.

[9] Anthony Gythiel and I. Langevin. "Paul Langevin's 1908 paper "On the Theory of Brownian Motion" ("Sur la theorie du mouvement brownien," C. R. Acad. Sci. (Paris) 146". In: *American Journal of Physics* 65 (Nov. 1997). DOI: 10.1119/1.18725.

[10] G. E. Uhlenbeck and L. S. Ornstein. "On the Theory of the Brownian Motion". In: *Phys. Rev.* 36 (5 Sept. 1930), pp. 823–841. DOI: 10.1103/PhysRev.36.823. URL: https://link.aps.org/doi/10.1103/PhysRev.36.823.

[11] Albert Einstein et al. "On the motion of small particles suspended in liquids at rest required by the molecular-kinetic theory of heat". In: *Annalen der Physik* 17 (1905), pp. 549–560.

[12] Sidney I. Resnick. *Adventures in Stochastic Processes*. CHE: Birkhauser Verlag, 1992. ISBN: 0817635912.

[13] B. Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Universitext. Springer Berlin Heidelberg, 2013. ISBN: 9783662028476. URL: https://books.google.com/books?id=%5C_6%5C_rCAAAQBAJ.

[14] G.A. Pavliotis. *Stochastic Processes and Applications: Diffusion Processes, the Fokker-Planck and Langevin Equations*. Texts in Applied Mathematics. Springer New York, 2014, pp. 70–72. ISBN: 9781493913237. URL: https://books.google.com/books?id=8213BQAAQBAJ.

[15] Eric Feron Stephen Boyd Laurent El Ghaoui and Venkataramanan Balakrishnan. "6. Analysis of LDIs: Input/Output Properties". In: *Linear Matrix Inequalities in System and Control Theory*. 1994, p. 78. DOI: 10.1137/1.9781611970777.ch6. eprint: https://epubs.siam.org/doi/pdf/10.1137/1.9781611970777.ch6. URL: https://epubs.siam.org/doi/abs/10.1137/1.9781611970777.ch6.

[16] A.J. Laub. *Matrix Analysis for Scientists and Engineers*. Other Titles in Applied Mathematics. Society for Industrial and Applied Mathematics, 2005. ISBN: 9780898715767. URL: https://books.google.com/books?id=0kdukXSX-9AC.

[17] P. Vatiwutipong and N. Phewchean. "Alternative way to derive the distribution of the multivariate Ornstein–Uhlenbeck process". In: *Advances in Difference Equations* (July 2019), pp. 276–. DOI: 10.1186/s13662-019-2214-1. URL: %60https://doi.org/10.1186/s13662-019-2214-1'.

[18] P.J. Dhrymes. *Mathematics for Econometrics*. Springer New York, 2013. ISBN: 9781461481447. URL: https://books.google.com/books?id=EnO3nAEACAAJ.

[19] Wolfram Research Inc. *Mathematica, Version 12.1*. Champaign, IL, 2020. URL: https://www.wolfram.com/mathematica.

[20] Joseph Skitka, J. B. Marston, and Baylor Fox-Kemper. "Reduced-Order Quasilinear Model of Ocean Boundary-Layer Turbulence". In: *Journal of Physical Oceanography* 50.3 (Mar. 2020), pp. 537–558. ISSN: 1520-0485. DOI: 10.1175/jpo-d-19-0149.1. URL: http://dx.doi.org/10.1175/JPO-D-19-0149.1.

[21] E. Wegert. *Visual Complex Functions: An Introduction with Phase Portraits*. SpringerLink : Bücher. Springer Basel, 2012. ISBN: 9783034801805. URL: https://books.google.com/books?id=zRppM7WO0vIC.

[22] J.B. Marston. "Planetary Atmospheres as Nonequilibrium Condensed Matter". In: *Annual Review of Condensed Matter Physics* 3.1 (2012), pp. 285–310. DOI: 10.1146/annurev-conmatphys-020911-125114. eprint: https://doi.org/10.1146/annurev-conmatphys-020911-125114. URL: https://doi.org/10.1146/annurev-conmatphys-020911-125114.

APPENDIX

Appendix A

Python Code

A.1 Processing MP4 files into Numpy Data via "Reverse Colormap"

The GCM data began in the form of mp4 videos, where the value of each component was mapped to a 'rwb' colormap. This code, reversed that colormap, and saved numpy arrays to be processed.

```
import cv2 # Used for image processing NB: use pip install open_cv (NOT pip install cv2)
import numpy as np # standard
from tqdm import tqdm # Easy progress_bar
import scipy.interpolate # Used to interpolate points not necessarily given directly by colormap
import matplotlib.pyplot as plt # Standard
from numba import jit #Just-in-time compiling speeds up simple functions
def gather_4d_array(mp4name, tmin, tmax, xmin, xmax, ymin, ymax,
    gridxmin, gridxmax, gridymin, gridymax):
    '''gather_4d_array is used to convert an mp4 colormap video of some value evolving in time
    into a numpy array of that value with dimensions for time, x and y.

    It uses the python package cv2, which can be installed with pip install open_cv (NOT pip install cv2)

```

```
    (t, x, y)min or max is the actual value of the min and max for those dimesnions, which should be
    read off of the graph.

    grid(x or y)(min or max) are the pixel values that contain meaningful color contain data (not titles etc...)

    Also HARD CODED in here is the method reversng the colormap
    '''
    cap = cv2.VideoCapture(mp4name)
    frameCount = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    frameWidth = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    frameHeight = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

    frame_1 = cv2.imread('frame1.png')

    # get colormap data (hard coded)
    colormapim = frame_1[52:428, 915:916]
    colormapimheight, colormapimwidth = tuple(np.shape(colormapim)[0:2])
    colormapim = np.reshape(colormapim, (colormapimheight, 3))
    colormapvalues = np.linspace(0.4, -0.4, num = colormapimheight)
    usedframeCount = int(frameCount)
    time_array = np.linspace(tmin, tmax, num = usedframeCount)
    used_x_array = np.linspace(xmin, xmax, num = gridxmax - gridxmin)
    used_y_array = np.linspace(ymin, ymax, num = gridymax - gridymin)

    cap_array = np.zeros((int(usedframeCount), gridymax - gridymin, gridxmax - gridxmin, 3))
    for i in tqdm(range(int(usedframeCount))):
        frame = cap.read()[1][gridymin: gridymax,gridxmin:gridxmax, :]
        cap_array[i,:,:,:] = frame

```

```
43      transposed_cap_array = cap_array
44      spline = scipy.interpolate.griddata(colormapim, colormapvalues,
    transposed_cap_array, method='nearest')
45
46      return time_array, used_x_array, used_y_array, spline
47
48  @jit
49  def inverse_bwr(rgb_arr):
50      r, g, b = rgb_arr[0], rgb_arr[1], rgb_arr[2]
51      if r <= b: #val <= 0.5
52          return ((g/2) + (r/2))/2
53      elif r > b: # val > 0.5
54          return (((2 - g)/2) + ((2 - b)/2))/2
55
56  def inverse_bwr_big_arr(y_x_rgb_arr):
57      r, g, b = y_x_rgb_arr[:, :, 0]/255, y_x_rgb_arr[:, :, 1]/255,
    y_x_rgb_arr[:, :, 2]/255
58      # return np.where(r <= b, ((g/2) + (r/2))/2, (((2 - g)/2) + ((2 - b)
    /2))/2)
59      return np.where(r <= b, ((g/2) + (r/2))/2, (((2 - g)/2) + ((2 - b)/2))
    /2)
60
61  def inverse_bwr_arr(rgb_arr):
62      val_arr = []
63      for rgb_val in rgb_arr:
64          val_arr.append(inverse_bwr(rgb_val))
65      return val_arr
66
67
68  def gather_4d_array_bwr(mp4name, tmin, tmax, xmin, xmax, ymin, ymax,
69      gridxmin, gridxmax, gridymin, gridymax, scale_min_max = 1):
```

```
    '''gather_4d_array is used to convert an mp4 colormap video of some value evolving in time
    into a numpy array of that value with dimensions for time, x and y.

    It uses the python package cv2, which can be installed with pip install open_cv (NOT pip install cv2)

    (t, x, y)min or max is the actual value of the min and max for those dimesnions, which should be
    read off of the graph.

    grid(x or y)(min or max) are the pixel values that contain meaningful color contain data (not titles etc...)

    Also HARD CODED in here is the method reversng the colormap
    '''
    cap = cv2.VideoCapture(mp4name)
    frameCount = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    frameWidth = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
    frameHeight = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))

    frame_1 = cv2.imread('frame1.png')


    usedframeCount = int(frameCount)
    time_array = np.linspace(tmin, tmax, num = usedframeCount)
    used_x_array = np.linspace(xmin, xmax, num = gridxmax - gridxmin)
    used_y_array = np.linspace(ymin, ymax, num = gridymax - gridymin)

    val_array = np.zeros((int(usedframeCount), gridymax - gridymin, gridxmax - gridxmin))
    for i in tqdm(range(int(usedframeCount))):
```

```
        frame = cap.read()[1][gridymin: gridymax,gridxmin:gridxmax, :]
        val_array[i,:,:] = inverse_bwr_big_arr(frame)
    scaled_val_array = scale_min_max* (val_array - np.full(np.shape(
    val_array), 0.5))
    print('np.shape(scaled_val_array) is %s'%list(np.shape(
    scaled_val_array)))
    return time_array, used_x_array, used_y_array,scaled_val_array

t_arr, x_arr, y_arr, val_arr = gather_4d_array_bwr('RedBlue_Data/
    MeridionalVelocity.mp4', 0, 60, -180, 180, -90, 90,
    115, 885, 50, 425, scale_min_max = .0124)
np.save("RedBlue_Data/MeridonalVel", val_arr)

t_arr, x_arr, y_arr, val_arr = gather_4d_array_bwr('RedBlue_Data/
    ZonalVelocity.mp4', 0, 60, -180, 180, -90, 90,
    115, 885, 50, 425, scale_min_max = .0124)
np.save("RedBlue_Data/ZonalVel", val_arr)

t_arr, x_arr, y_arr, val_arr = gather_4d_array_bwr('RedBlue_Data/
    BarotropicTemperature.mp4', 0, 60, -180, 180, -90, 90,
    115, 885, 50, 425, scale_min_max = 0.4)
np.save("RedBlue_Data/BarotropicTemp", val_arr)
```

A.2 Simple Linear Shallow Water Simulation

```
import numpy as np # Standard
import matplotlib.pyplot as plt #Standard
from tqdm import tqdm #Great progress bar
import copy # Standard
from matplotlib.widgets import Slider #HElps with graphing and watchign
    evolution
```

```
6
7  #Simple array manipulation schemes
8  def reorder_axis(arr, ax):
9      pos_part, neg_part = np.array_split(arr, 2, axis =ax)
10     rejoined_arr = np.concatenate((neg_part, pos_part), axis = ax)
11     return rejoined_arr
12
13 def reorder_axes(arr, axes_arr):
14     if len(axes_arr)==0:
15         return arr
16     else:
17         return reorder_axes(reorder_axis(arr, axes_arr[0]), axes_arr[1:])
18
19 def find_nearest_index(array, value):
20     array = np.asarray(array)
21     idx = (np.abs(array - value)).argmin()
22     return idx
23 #USEFUL FUNCTIONS FOR AVERAGING AND MAKING REASONABLE NOISE (NOT ALL USED)
24 def gaussian(x, y, sigma, mu1 = 0, mu2 = 0):
25   return (1/ (sigma* np.sqrt(2*np.pi))) * np.exp(-0.5 * (np.sqrt((x - mu1)
       **2 + (y - mu2)**2)/ sigma)**2)
26
27 def sigmoid(x):
28   return 1/(1+np.exp(-x))
29
30 def two_sided_sigmoid(size, half_drop, a1, a2):
31   x = np.arange(size)
32   return sigmoid((x - 10*half_drop - a1)/half_drop) - sigmoid((x+10*
       half_drop - a2)/half_drop)
33
34 #SIMPLE FINITE DIFFERENCE DERIVATIVES
35 def drv_x(arr):
```

```
  return (np.roll(arr, 1, axis = 1) - np.roll(arr, -1, axis = 1) )/(2*
   x_step)

def drv_y(arr):
  derivatived = (np.roll(arr, 1, axis = 0) -np.roll(arr, -1, axis = 0))
   /(2*y_step)
  return derivatived

#SETTING UP INITIAL CONDITIONS AND GRID PARAMETERS
x_step = 1
y_step = 1

grid_size = 400
xx, yy   = np.meshgrid(np.arange(grid_size),np.arange(grid_size))
gaussian_for_noise = gaussian(xx, yy, grid_size/10, grid_size/2, grid_size
   /2)
h_arr = np.zeros((grid_size, grid_size))
u_arr = np.zeros((grid_size, grid_size))
v_arr = np.zeros((grid_size, grid_size))
f = 1

def random_force(noise_scale):
  #RANDOM FORCE FOR AN ENTIRE GRID. CUTS OFF HGH FREQUENCIES OF NOISE THT
   COULD DISRUPT FINITE DIFFERENCE
  unfourier_noise = np.random.normal(scale = noise_scale, size = (
   grid_size, grid_size))
  unfourier_noise_shifted = reorder_axes(gaussian_for_noise*
   unfourier_noise, (0, 1))
  unfourier_noise_shifted = unfourier_noise_shifted
  ftn = np.real(np.fft.fftn(unfourier_noise_shifted))
  smoothed_noise = 0.5*ftn + (1/8)*(np.roll(ftn, 1, axis=0) + np.roll(ftn,
    -1, axis=0) + np.roll(ftn, 1, axis=1) + np.roll(ftn, -1, axis=1))
```

```
  return smoothed_noise


def update_arrs(h_arr, u_arr, v_arr, time_step, f):
  #EULER METHOD TIME STEPPING GIVEN BY LINEAR SW EQUATIONS WITH DAMPING
   AND A RANDOM FORCE ON h
  noise_scale = 1.4
  drag = 0.1
  h_arr_change = +random_force(1.4* (0.05/time_step)) - drag*h_arr - drv_x
   (u_arr) - drv_y(v_arr)
  u_arr_change = -drag*u_arr  -drv_x(h_arr) + f*v_arr#+  np.random.normal(
   scale = noise_scale,size = (200, 200))
  v_arr_change = -drag*v_arr  -drv_y(h_arr) - f*u_arr#+  np.random.normal(
   scale = noise_scale,size = (200, 200))
  h_arr_new = time_step*h_arr_change + h_arr
  u_arr_new = u_arr + time_step*u_arr_change
  v_arr_new = v_arr + time_step*v_arr_change

  return h_arr_new, u_arr_new, v_arr_new

#Setup time
num_timesteps = 200
time_step = 0.05
time_arr = np.arange(num_timesteps)*time_step

#SIMPLE SHEME TO EULER STEP THROUGH AND GATHER DATA
h_total_array = [h_arr]
u_total_array = [u_arr]
v_total_array = [v_arr]
for i in tqdm(range(num_timesteps)):
  h_arr, u_arr, v_arr = update_arrs(h_arr, u_arr, v_arr, time_step, f)
  h_total_array.append(copy.copy(h_arr))
```

```
  u_total_array.append(copy.copy(u_arr))
  v_total_array.append(copy.copy(v_arr))

h_total_nparr = np.array(h_total_array)
u_total_nparr = np.array(u_total_array)
v_total_nparr = np.array(v_total_array)

#GROUP DATA TOGETHER FOR EASY SAVING/HANDLING
huv_nparr = np.stack((h_total_nparr, u_total_nparr, v_total_nparr), axis =
    3)

#DYNAMIC PLOTTING OF DATA
fig, ax = plt.subplots()
plt.subplots_adjust(left=0.25, bottom=0.25)

pcolormesh = plt.pcolormesh(h_arr)
fig.colorbar(pcolormesh)
plt.set_cmap('bwr')
pcolormesh.set_clim(vmin=-0.3, vmax=0.3)
ax.margins(x=0)
ax.set_title('h(t, x, y) for f=%s, drag=%s, noise_magnitude=%s/Delta t, \
   nDelta x = Delta y = 1, Delta t = %s'%(f, 0.1, np.round(1.4*0.05, 2),
   time_step))
axcolor = 'lightgoldenrodyellow'
axtime = plt.axes([0.25, 0.1, 0.65, 0.03], facecolor=axcolor)
time_index = 0

stime= Slider(axtime, 'time', time_arr[0], time_arr[-1],
 valinit=time_arr[time_index], valstep=time_arr[1]- time_arr[0])


def update(val):
```

```
    time = stime.val
    time_index = find_nearest_index(time_arr, time)
    new_array = np.array(h_total_array[time_index])
    pcolormesh.set_array(new_array.ravel())
    fig.canvas.draw_idle()


stime.on_changed(update)
plt.show()
```

A.3 Plotting Theoretical Solutions

This code was used to generate hue-saturation plots of the theoretical solution.

```
import cv2 #NB: use pip install open_cv (NOT pip install cv2)
import numpy as np # Standard
from tqdm import tqdm # Excellent easy automated progress bar
import matplotlib.pyplot as plt # Standard
from skimage.color import hsv2rgb # Converts Hue-Saturation-Value color
    data to RGB
import copy # Standard

c =25
#COVARIANCE IN hu CALCULATED IN THEORY
def C(f, kx, ky, th, omega):
  det = ((omega**2 + th**2)*((c**2*kx**2 + c**2*ky**2+ f**2 - omega**2)**2
     + 2*th**2*(c**2*kx**2 + c**2*ky**2+ f**2 + omega**2)+ th**4))
  real_part = ((f**2 - omega**2)*(omega*c*kx) + th*(-2*f*omega*c*ky) + th
    **2*(-c*kx*omega))/det
  imag_part = ((f**2 - omega**2)*(f*c*ky) + th*kx*c*(f**2 + omega**2) + th
    **2*(f *c*ky) + -th**3*(c*kx))/det
  return real_part, imag_part
```

```
15
16 def Simple(f, kx, ky, th, omega):
17   return kx, ky
18
19 #GRAPHING COMPLEX NUMBER AS HUE-SATURATION PLOT
20 x_num, y_num = 400, 400
21 base_shape_array, dummy = np.meshgrid(np.ones(x_num), np.ones(y_num))
22 base_x_arr = np.arange(-1/2, 1/2, 1/400)
23 x_arr, y_arr = np.meshgrid(base_x_arr, -base_x_arr)
24 f = 0
25 theta = 0.1
26 omega = .1
27 real, imag = C(f, x_arr, y_arr, theta, omega)
28 r_arr = np.sqrt(real**2 + imag**2)
29 r_max = np.amax(r_arr)
30 clipped_r_arr = np.where(r_arr < r_max, r_arr, r_max)
31 theta_arr = np.arctan2(imag, real)
32 hue_arr = (theta_arr + np.pi)/(2*np.pi)
33 sat_arr = clipped_r_arr / r_max
34 val_arr = base_shape_array
35 hsv_arr = np.dstack([hue_arr, sat_arr, val_arr])
36 rgb_arr = hsv2rgb(hsv_arr)
37
38 plt.title('\nPredicted C(omega)_hu with f = %s, theta = %s, omega = %s \n
     sqrt(omega^2 - f^2)/c= %s, max_magnitude = %s'%(f, theta, omega, np.
     round(np.sqrt(omega**2 - f**2)/c, 3), r_max))
39 plt.xlabel('kx')
40 plt.ylabel('ky')
41
42
43 plt.imshow(rgb_arr, interpolation='nearest')
44
```

```
#DEALING WITH ANNOYING MATPLOTLIB TICKMARK HANDLING
new_labels = np.arange(-1/2, 1/2, 1/400)
locs, labels = plt.xticks()
num_labels = len(labels)
new_labels = np.round(np.arange(-1/2, 1/2, 1/num_labels), 4)
labels = new_labels
plt.xticks(locs[1:], labels[1:])
locs, labels = plt.yticks()
labels = -new_labels
plt.yticks(locs[1:], labels[1:])

plt.show()
```

A.4 Data Analysis Code

Used for analyzing GCM and simple linear shallow water data. Shown for GCM only, but easily adapted to linear shallow water.

```
import cv2 #NB: use pip install open_cv (NOT pip install cv2)
import numpy as np # Standard
from tqdm import tqdm # Excellent easy automated progress bar
import matplotlib.pyplot as plt # Standard
from skimage.color import hsv2rgb # Converts Hue-Saturation-Value color
    data to RGB
import copy # Standard

#SOME HELPER FUNCTIONS FOR ARRAY MANIPULATION
def brickify(arr, subgrid_shape_list, ax_depth =0):
    arr = np.array(arr, dtype = 'complex64')
    base_dimensions = list(np.shape(arr))
    base_dimensions_len = len(base_dimensions) - 1
```

```
13      cur_num_bricks = None
14      if base_dimensions[base_dimensions_len - ax_depth] %
    subgrid_shape_list[-1] == 0:
15          cur_num_bricks = int(base_dimensions[base_dimensions_len -
    ax_depth]/ subgrid_shape_list[-1])
16      else:
17          return "Error: subgrid shape does not match well with the data
    shape: %s does not divide %s"%(subgrid_shape_list[-1], base_dimensions[
    base_dimensions_len - ax_depth])
18      if ax_depth == base_dimensions_len/ 2:
19          return np.array(np.split(arr, cur_num_bricks, axis =
    base_dimensions_len - ax_depth))
20      else:
21          split = np.array(np.split(arr, cur_num_bricks, axis =
    base_dimensions_len - ax_depth))
22          return brickify(split, subgrid_shape_list[:-1], ax_depth =
    ax_depth + 1)
23
24  def reorder_axis(arr, ax):
25      pos_part, neg_part = np.array_split(arr, 2, axis =ax)
26      rejoined_arr = np.concatenate((neg_part, pos_part), axis = ax)
27      return rejoined_arr
28
29  def find_nearest_index(array, value):
30      array = np.asarray(array)
31      idx = (np.abs(array - value)).argmin()
32      return idx
33
34  def reorder_axes(arr, axes_arr):
35      if len(axes_arr)==0:
36          return arr
37      else:
```

```
38          return reorder_axes(reorder_axis(arr, axes_arr[0]), axes_arr[1:])
39
40  def gaussian(x, sigma, mu = 0):
41      return (1/ (sigma* np.sqrt(2*np.pi))) * np.exp(-0.5 * ((x - mu)/ sigma
    )**2)
42
43  def gaussian_average_over_axis_forloop(nd_arr, window_res, axis):
44      #THE WAY OF DOING GAUSSIAN AVERAGING THAT MY COMPUTER COULD HANDLE.
45      if window_res == 1:
46          return nd_arr
47      else:
48          #build gaussian frame
49          half_size = int(window_res // 1.5)
50          std_dev = 0.25* half_size # Automaticaal do gaussian out to 4
    sigma
51          frame = np.arange(-half_size, half_size + 1)
52          gaussian_frame = np.complex64(gaussian(frame, std_dev))
53          frame_length  = np.shape(frame)[0]
54
55          #transpose array so that last axis is averaging axis
56          original_shape = list(np.shape(nd_arr))
57          num_dims = len(original_shape)
58          pre_transpose_permute = [i for i in range(num_dims)]
59          transpose_permute = [i for i in range(num_dims)]
60          transpose_permute[axis] = pre_transpose_permute[num_dims - 1]
61          transpose_permute[num_dims - 1]= pre_transpose_permute[axis]
62          last_ax_arr = np.transpose(nd_arr, transpose_permute)
63          last_ax_arr_shape = list(np.shape(last_ax_arr))
64          last_ax_length = last_ax_arr_shape[-1]
65
66          # AVERAGE DATA USING GAUSSIAN FILTER
67          avged_transposed_data = np.zeros(np.shape(last_ax_arr))
```

```
68          for i in tqdm(range(len(gaussian_frame))):
69              weight = gaussian_frame[i]
70              shift = frame[i]
71              rolled_arr = weight * np.roll(last_ax_arr, shift,axis =
    num_dims-1)
72              avged_transposed_data = avged_transposed_data + rolled_arr
73
74          avged_data = np.transpose(avged_transposed_data, transpose_permute
    )
75          return avged_data
76
77  def gaussian_average_over_axes(nd_arr, window_res_arr, axes_arr):
78      # SAME AS GUASSIAN AVERAGE OVER AXIS, JUST FOR MULTIPLE AXES AT ONCE
79      if len(axes_arr) == 1:
80          return gaussian_average_over_axis_forloop(nd_arr, window_res_arr
    [0], axes_arr[0])
81      else:
82          return gaussian_average_over_axes(
    gaussian_average_over_axis_forloop(nd_arr, window_res_arr[0], axes_arr
    [0]),
83              window_res_arr[1:], axes_arr[1:])
84
85  class d2_colormesh(object):
86      #CLASS FOR HUE-SATURATION PLOTS IN 2D-- CALLED d2_colormesh AFTER
    COLORMESH IN MATPLOTLIB
87      def __init__(self, axis, theta_arr, magnitude_arr, x_min, x_max, y_min
    , y_max, min_magnitude = None, max_magnitude = None):
88          self.theta_arr = theta_arr
89          self.magnitude_arr = magnitude_arr
90          self.x_min = x_min
91          self.x_max = x_max
92          self.y_min = y_min
```

```
93          self.y_max = y_max
94          self.axis = axis
95          self.register_min_max_magnitude()
96          self.make_rgb_array()
97
98      def register_min_max_magnitude(self):
99          self.min_magnitude = np.amin(self.magnitude_arr)
100         self.max_magnitude = np.amax(self.magnitude_arr)
101
102     def make_rgb_array(self):
103         hue_arr = (np.full(np.shape(self.theta_arr), np.pi) + self.
    theta_arr) * (1/(2*np.pi))
104         hue_arr = np.full(np.shape(hue_arr), 0)
105         new_mag_arr = np.where(self.magnitude_arr < self.max_magnitude,
    self.magnitude_arr, self.max_magnitude)
106         sat_arr = np.clip((new_mag_arr - np.full(np.shape(new_mag_arr),
    self.min_magnitude)), a_min = 0, a_max = None) * (1/(self.max_magnitude
     - self.min_magnitude))
107         val_arr = np.array(np.full(np.shape(sat_arr), 1))
108         hsv_arr = np.dstack([hue_arr, sat_arr, val_arr])
109         self.rgb_arr = hsv2rgb(hsv_arr)
110
111     def make_img(self):
112         self.aximg_obj = self.axis.imshow(self.rgb_arr, interpolation='
    nearest', extent = [self.x_min, self.x_max, self.y_min, self.y_max])
113
114     def register_new_theta_mag(self, theta_new, mag_new):
115         self.theta_arr = theta_new
116         self.magnitude_arr = mag_new
117         self.register_min_max_magnitude()
118         self.make_rgb_array()
119
```

```
120     def update_data(self):
121         self.aximg_obj.set_data(self.rgb_arr)
122 
123 def process_data():
124     #TAKES NUMPY ARRAYS OF BASE DATA AND PROCESSES IT TO FIDN CROSS-
    COVARIANCE
125 
126     # DECIDE WHAT PART OF DATA TO TAKE
127     start_time_index = 0
128     end_time_index = 800
129     crop_x_start = 6
130     crop_x_end = -6
131     crop_y_start = 1
132     crop_y_end = -2
133 
134     #LOADING NUMPY ARRAYS OF DATA
135     temp_arr = np.complex64(np.load("RedBlue_Data/BarotropicTemp.npy")[
136         start_time_index:end_time_index, crop_y_start:crop_y_end,
    crop_x_start:crop_x_end] )
137     u_arr = np.complex64(np.load("RedBlue_Data/MeridonalVel.npy")[
138         start_time_index:end_time_index, crop_y_start:crop_y_end,
    crop_x_start:crop_x_end] )
139     v_arr = np.complex64(np.load("RedBlue_Data/ZonalVel.npy")[
140         start_time_index:end_time_index, crop_y_start:crop_y_end,
    crop_x_start:crop_x_end] )
141     num_time_steps = end_time_index - start_time_index
142 
143     # SETTING UP SYSTEM SPECIFIC PARAMETERS
144     total_time = 80 #days
145     total_steps_to_sample_from = 800
146     selected_total_time_interval = total_time * (num_time_steps/
    total_steps_to_sample_from)
```

```
147     time_step = selected_total_time_interval / num_time_steps
148 
149     total_y_num = np.shape(temp_arr)[1]
150     y_steps = total_y_num / 6
151     y_step = 180/ total_y_num
152     x_range = 360
153     kx_scale = 1/360
154     ky_scale = 1/(y_steps*180/total_y_num)
155     total_omega_num = num_time_steps
156     desired_omega_num = 1
157 
158     def fourier_and_all(val_arr):
159         # SLICES DATA INTO STRIPS AND FOURIER TRANSFORMS IT
160         val_omega_x_y = np.fft.fft(val_arr, axis =0)
161         x_len = np.shape(val_omega_x_y)[2]
162         bricked_val_omega_x_y =  brickify(val_omega_x_y, [
    desired_omega_num, y_steps, x_len])
163         bricked_val_omega_x_y_small_omega_kx_ky = np.fft.fftn(
    bricked_val_omega_x_y, axes = (4,5))
164         x_step = 360/bricked_val_omega_x_y.shape[5]
165         kx_arr, ky_arr = np.fft.fftfreq(bricked_val_omega_x_y.shape[5], d=
    x_step), np.fft.fftfreq(bricked_val_omega_x_y.shape[4], d=y_step)
166 
167         y_arr = np.linspace(-90, 90, int(total_y_num/y_steps))
168 
169         kx_arr = reorder_axis(kx_arr, 0)* (kx_scale)
170         ky_arr = reorder_axis(ky_arr, 0)* (ky_scale)
171         return bricked_val_omega_x_y_small_omega_kx_ky, kx_arr, ky_arr,
    y_arr
172 
173     temp_processed, kx_arr, ky_arr, y_arr = fourier_and_all(temp_arr)
174     u_processed, kx_arr, ky_arr, y_arr = fourier_and_all(u_arr)
```

```
175     v_processed, kx_arr, ky_arr, y_arr = fourier_and_all(v_arr)
176 
177     def manual_cov(arr1, arr2, mean_axes):
178         # COVARIANCE TAKEN IN OMEGA SPACE IN SIMPLE WAY DESCRIBED BY
    SKITKA ET AL.
179         prereorder = np.mean(np.multiply(arr1, np.conj(arr2)), axis =
    mean_axes)
180         return reorder_axes(prereorder, (0, 2, 3))
181 
182     omega_arr = np.fft.fftfreq(int(total_omega_num/desired_omega_num))
183     omega_arr = reorder_axis(omega_arr, 0)
184 
185 
186     temptemp_along_x_bricks_omega_y_kx_ky = manual_cov(temp_processed,
    temp_processed, (2, 3))
187     tempu_along_x_bricks_omega_y_kx_ky = manual_cov(temp_processed,
    u_processed, (2, 3))
188     tempv_along_x_bricks_omega_y_kx_ky = manual_cov(temp_processed,
    v_processed, (2, 3))
189     utemp_along_x_bricks_omega_y_kx_ky = manual_cov(u_processed,
    temp_processed, (2, 3))
190     uu_along_x_bricks_omega_y_kx_ky = manual_cov(u_processed, u_processed,
     (2, 3))
191     uv_along_x_bricks_omega_y_kx_ky = manual_cov(u_processed, v_processed,
     (2, 3))
192     vtemp_along_x_bricks_omega_y_kx_ky = manual_cov(v_processed,
    temp_processed, (2, 3))
193     vu_along_x_bricks_omega_y_kx_ky = manual_cov(v_processed, u_processed,
     (2, 3))
194     vv_along_x_bricks_omega_y_kx_ky = manual_cov(v_processed, v_processed,
     (2, 3))
195 
```

```
196     return (temptemp_along_x_bricks_omega_y_kx_ky,
197     tempu_along_x_bricks_omega_y_kx_ky,
198     tempv_along_x_bricks_omega_y_kx_ky,
199     utemp_along_x_bricks_omega_y_kx_ky,
200     uu_along_x_bricks_omega_y_kx_ky,
201     uv_along_x_bricks_omega_y_kx_ky,
202     vtemp_along_x_bricks_omega_y_kx_ky,
203     vu_along_x_bricks_omega_y_kx_ky,
204     vv_along_x_bricks_omega_y_kx_ky, omega_arr, kx_arr, ky_arr, y_arr)
205
206 def plot_at_omega_y(val_arr, axis, omega_desired, y_desired, omega_arr,
    y_arr):
207     # GENERAL PLOTTING FOR DATA
208
209     # FIND AN OMEGA AND Y VALUE CLOSE TO WHAT IS WANTED
210     omega_index = find_nearest_index(omega_arr, omega_desired)
211     y_index = find_nearest_index(y_arr, y_desired)
212
213     #REGISTERING HUE SATURATION PLOT
214     norm = np.absolute(np.array(val_arr[omega_index, y_index,:,:]))
215     imag_part = np.float32(np.imag(np.array(val_arr[omega_index,y_index
    ,:,:]))/(norm[:, :] + np.full(np.shape(norm[:, :]), 0.000001)))
216     real_part = np.float32(np.real(np.array(val_arr[omega_index,y_index
    ,:,:]))/(norm[:, :] + np.full(np.shape(norm[:, :]), 0.000001)))
217     theta_arr = np.arctan2(imag_part, real_part)
218     d2_colorplot_obj = d2_colormesh(axis, theta_arr, norm, kx_arr[0],
    kx_arr[-1], ky_arr[0], ky_arr[-1])
219     d2_colorplot_obj.make_img()
220
221     #STUFF FOR TITLE TO SET AUTOMATICALLY WITH IMPORTANT INFO
222     round_level = 3
223     omegaval = np.round(omega_arr[omega_index], round_level)
```

```
224     yval = y_arr[y_index]
225     ymin = yval - 15
226     ymax = yval + 15
227     fmin = np.round(2*np.sin(2*np.pi*ymin/360), round_level)
228     fmax = np.round(2*np.sin(2*np.pi*ymax/360), round_level)
229     favg = np.round(np.average(2*np.sin(2*np.pi*np.arange(ymin, ymax+1,
        (1/29))/360)), 2)
230     ck_avg = np.round(np.sqrt(omegaval**2 - favg**2), round_level)
231     axis.set_title("\n omega = %s (/day) pm ___, y_window= %s to %s deg \n
        f_window = %s to %s (/day),"%(omegaval, ymin, ymax, fmin, fmax)
232         +" f_avg =%s (/day)\n sqrt(omega^2-f_avg^2)=%s"%(favg, ck_avg))
233
234 # SETUP ARRAYS AND LOAD DATA
235 omega_arr = np.arange(-4, 4, 8/800)
236 kx_arr = np.arange(-0.5, 0.5 + 1/(758-1), 1/(758-1))
237 ky_arr = np.arange(-0.5, 0.5 + 1/(62-1), 1/(62-1))
238 y_arr = np.array([-75, -45, -15, 15, 45, 75]) #pm 15 degrees
239 averaged_data = np.load("RedBlue_Data/process_800time_no_omega_blocks/
        averaged_tempu_6.npy")
240
241 # PLOT GCM DATA
242 print("data shape is %s"%list(np.shape(averaged_data)))
243 fig, ax = plt.subplots()
244 plt.subplots_adjust(left=0.25, bottom=0.25)
245 ax.set_xlabel("k_x (1/2pi*R_earth)")
246 ax.set_ylabel("k_y (1/2pi*R_earth)")
247
248 plot_at_omega_y(averaged_data, ax, -0.9, -0.5, omega_arr, y_arr)
249 ax.set_xlim([-0.1, 0.1])
250 ax.set_ylim([-0.1, 0.1])
251 plt.show()
```

Appendix B

Mathematica Code

This is the general code used to calculate relevant matrices in Mathematica. Simple matrix multiplication is not included. Only code for complex matrix operations like vectorization is included, as resulting matrices were very large, and required a great deal of manual simplification.

```
Id = {{1, 0, 0}, {0, 1, 0}, {0, 0, 1}};
InverseVectorize[a_List] :=
  Transpose[
   ArrayReshape[a, Join[Sqrt[Dimensions[a]], Sqrt[Dimensions[a]]]]];
Vectorize[a_List?MatrixQ] :=
 ArrayReshape[Transpose[a], Dimensions[a][[1]]*Dimensions[a][[2]]] /;



  Equal @@ Dimensions[a]
KroneckerSum[a_List?MatrixQ, b_List?MatrixQ] :=
 KroneckerProduct[a, IdentityMatrix[Length[b]]] +
   KroneckerProduct[IdentityMatrix[Length[a]], b] /;
  Equal @@ Dimensions[a] && Equal @@ Dimensions[b]

L = {{-\[Theta], -I c x, -I c y }, {-I c x, -\[Theta], -f}, {-I c y,
    f, -\[Theta]}};
Lstar = {{-\[Theta], I c x, I c y }, {I c x, -\[Theta], -f}, {I c y,
    f, -\[Theta]}};

mathcalC[ sigsigdag_] :=
```

```
  InverseVectorize[
   Inverse[KroneckerSum[Lstar, L]].Vectorize[sigsigdag]];

Simplify[mathcalC[{{1, 0, 0}, {0, 0, 0}, {0, 0, 0}}]]
```