UNIVERSITY OF CALIFORNIA

SANTA CRUZ

DATA DRIVEN MODELING OF GEOPHYSICAL FLOWS WITH PARTIAL STATES

A thesis submitted in partial satisfaction

of the requirements for the degree of

MASTER OF SCIENCE

in

SCIENTIFIC COMPUTING & APPLIED MATHEMATICS

by

Hayley Coyle

September 2024

The Thesis of Hayley Coyle
is approved:

Dr. Ashesh Chattopadhyay, Chair

Dr. Marcella Gomez

Dr. Dongwook Lee

Peter Biehl

Vice Provost and Dean of Graduate Studies

Contents

List of Figures

Abstract

Data Driven Modeling of Geophysical Flows with Partial States

by Hayley Coyle

In recent years, machine learning models have offered an efficient approach to studying geophysical fluid dynamics, particularly in scenarios where data availability is often limited. This thesis project presents a study on the application of a Fourier neural operator (FNO) to the quasi-geostrophic (QG) system, an important system in geophysical fluid dynamics used to simulate large scale atmospheric flows. The primary objective of this research is to evaluate the performance of an FNO-based data-driven autoregressive model in predicting the evolution of streamfunctions ψ_1 and ψ_2 under various integration schemes, such as first-order Euler, fourth-order Runge-Kutta, as well as a simpler predictive approach where the FNO directly computes the next state in a sequence from the current state without intermediate calculations or corrections.

The key question driving this study is the exclusion of the moisture channel from the training data, exploring whether or not we can effectively train the model on only partial states of data and still be able to get accurate assessments of large scale atmospheric flows. Our experiments demonstrate that while the FNO-based approach shows some promise in capturing the underlying dynamics of the QG system, excluding the moisture channel leads to challenges in achieving stable and accurate predictions. Our results demonstrate sensitivity of FNOs to missing state information, with evaluation metrics such as spectral analysis, Anomaly Correlation Coefficient (ACC), and Root Mean Square Error (RMSE) metrics showing us the impact of the moisture exclusion on the accuracy of the predictions.

Acknowledgments

I'd like to acknowledge professor Ashesh Chattopadhyay in providing me with invaluable opportunities to work in his research group and challenge myself. I could not have done any of this without his support and advice. I owe much of my knowledge of these models and their interpretation to him, and I thank him for taking the time to help me understand and work with my results.

I would also like to sincerely thank Leonard Lupin for his time and effort in helping me with coding, debugging, and providing support, collaboration, and ideas when needed.

An additional thank you to Professors Marcella Gomez and Dongwook Lee as committee members and educators while I engaged in research.

Lastly, I'd like to thank Matthew Millette for his unwavering encouragement and support.

Chapter 1

Introduction

Climate change and its impact on the global weather system and human population is a serious concern within the scientific community. Current research has already identified human influences on the frequency and intensity of these extreme events, so there is great importance on our understanding how these events will evolve in a warmer climate. Determining the expected frequency with which a particular extreme weather event may occur is crucial for understanding and managing the associated risks posed to humanity.

In recent years, predictive deep-learning models have shown great promise in weather forecasting by leveraging vast amounts of data to anticipate atmospheric phenomena. However, these models face significant challenges, particularly when it comes to long-term instabilities and data availability. The frequent scarcity of complete weather data contributes to the challenge of accurate weather prediction [5], and thus necessitates predictions based on only partial data states.

This issue is only amplified by spectral bias, which is a tendency for neural networks to prioritize low-frequency components over high-frequency ones, and has been identified as a universal causal mechanism for instabilities in turbulent flow within deep learning architectures [2]. These models, by their

nature, have a tendency to encounter spectral bias, causing the model to focus primarily on learning and predicting large-scale dynamics while neglecting the finer, small-scale details that are crucial for accurate long-term predictions. This results in an amplification of errors as the prediction extends over time and leads to significant deviations from the actual atmospheric state. The issue of spectral bias is not limited to atmospheric models but is also prevalent in various engineering applications involving turbulent flow [12].

This project aims to address long-term instabilities and the associated challenge of missing data by experimenting with a Fourier neural operator (FNO) data-driven machine learning network on a QG system. We have implemented integration schemes to the architecture in order to mitigate this error growth, and have begun to see some marginally successful results with data-driven methods.

1.1 Conventional vs. Data-driven methods

The field of weather prediction has traditionally relied on numerical weather prediction (NWP) models, which are based on solving systems of PDEs that describe atmospheric dynamics. These models are successful but they come with significant computational costs [10]. In contrast, data-driven methods, such as FNOs, offer a cost effective alternative that are generally faster and less resource intensive while still providing accurate results.

One of the primary advantages of data-driven models over conventional ones is computational efficiency. FNOs bypass the need to solve these equations directly by learning the underlying dynamics directly from the data. The reduced computational demand of FNOs translates directly into lower infrastructure costs and can also offer scalability and flexibility that traditional methods struggle to match. Scaling up an NWP model to higher resolutions or

longer time frames typically involves a substantial increase in computational costs and complexity. Each increase in resolution requires more grid points and and thus more computational power. However, because FNOs learn at a logarithmic rate directly from data, they can be trained much more efficiently on a variety of resolutions and and time scales without the same exponential increase in computational demands. The issue then becomes about dialing in the model with the best parameters in order to get the most out of our data-driven framework.

The FNO is unique in this context because, unlike a physics informed neural network (PINN), the FNO does not explicitly incorporate the governing physical laws into a loss function during training. A PINN is trained not only to fit data but also to satisfy the physical laws that govern the system being modeled. Instead, the FNO model learns the dynamics that are governed by physical laws as encoded in the data. This is done implicitly, rather than through explicit physical constraints in the loss function.

Chapter 2

Model and Data

2.1 Quasi-Geostrophic model

For this study we have chosen to work with the Quasi-Geostrophic (QG) system, an idealized two-channel system that focuses on large-scale geostrophic flows by retaining the balance between the Coriolis force and pressure gradients, and is a good choice for mid-latitude regions where baroclinic instability dominates [9] [11]. The data we used to train, test, and compare our FNO to was generated by this model.

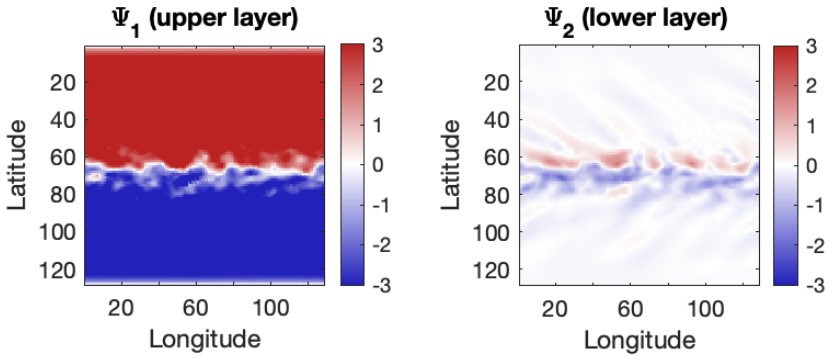

Figure 2.1: An example of the upper (ψ_1) and lower (ψ_2) layers of the QG system, discussed in more detail in section 2.1.1.

The two layers of the QG system as seen in figure 2.1 have a characteristic distance y between ψ_1 and ψ_2, typically given by the Rossby deformation radius, L_d, which is defined as $\frac{NH}{f_0}$. Here, N is the buoyancy frequency, H is the depth of the layer, and f_0 is the Coriolis parameter. This distance represents the scale at which rotational effects become significant in the dynamics of the flow between the two layers.

2.1.1 Potential Vorticity

The potential vorticity (q_k), for both the upper ($k = 1$) and lower ($k = 2$) layers are defined by:

$$q_k = \nabla^2\psi_k + (-1)^k(\psi_1 - \psi_2) + \beta y, \tag{2.1}$$

where ψ_k represents each layer's stream function, and β is an approximation for the gradient of the Coriolis parameter with respect to the latitude, which is denoted as y.

The governing equations for the model, which describe the conservation of potential vorticity in each of the two layers, is given by:

$$\frac{\partial q_k}{\partial t} + J(\psi_k, q_k) = -\frac{1}{\tau_d}(-1)^k(\psi_1 - \psi_2 - \psi_R) - \frac{1}{\tau_f}\delta_{k2}\nabla^2\psi_k - \nu\nabla^4 q_k + (-1)^k LP, \tag{2.2}$$

respectively. Here, τ_d is the relaxation time scale, τ_f is the frictional time scale, and A and σ are constants. This equation (2.2) is used separately to define each layer, equal to the partial derivative of each layer with respect to time,

added to the the Jacobian operator:

$$J(\psi_k, q_k) = \psi_{kx} q_{ky} - \psi_{ky} q_{kx}, \tag{2.3}$$

where ψ_{kx} and ψ_{ky} denote the partial derivatives of ψ_k with respect to the x and y spatial coordinates, respectively. Similarly, q_{kx} and q_{ky} represent the partial derivatives of q_k with respect to x and y.

The term $\psi_{kx} q_{ky}$ here describes the interaction between the changes in the streamfunction in the x-direction and the changes in the potential vorticity in the y-direction, where the second term $\psi_{ky} q_{kx}$ describes the interaction between the changes in the streamfunction in the y-direction and the changes in the potential vorticity in the x-direction. The result is that the Jacobian $J(\psi_k, q_k)$ represents the advection of potential vorticity by the flow described by the streamfunction ψ_k.

In equation 2.2, the first term $-\frac{1}{\tau_d}(-1)^k(\psi_1 - \psi_2 - \psi_R)$, accounts for solar radiation with τ_d by relaxing the temperature $\tau = \psi_1 - \psi_2$ towards ψ_R, which is the equilibrium profile given by:

$$\psi_R = -\sigma A \tanh\left(\frac{y}{\sigma}\right) \tag{2.4}$$

This equilibrium profile (1.4) is asymmetric around $y = 0$.

In equation 2.2, the second term $-\frac{1}{\tau_f}\delta_{k2}\nabla^2\psi_k$ models surface friction in the lower layer, with τ_f being the frictional time scale, and δ_{k2} represents the Kronecker delta, which is 1 when $k = 2$ (the lower layer) and 0 otherwise.

In equation 2.2, the third term $-\nu\nabla^4 q_k$ introduces a hyperviscosity effect by acting to dampen the small scale features of the potential vorticity field q_k through diffusion.

Finally, the last term $(-1)^k LP$ represents latent heating due to precipi-

tation, where L is a constant and P is the precipitation rate, influenced by moisture.

2.1.2 Moisture Conservation of Mass

In this model, the moisture m is assumed to be advected within $k = 2$ only. The equation governing moisture conservation is given by:

$$\frac{\partial m}{\partial t} + J(\psi_2, m) = E - P - \nabla \cdot \mathbf{u}_2, \tag{2.5}$$

where E is evaporation, P is precipitation, and the last term $\nabla \cdot \mathbf{u}_2$ is the divergence of the ageostrophic wind velocity field of the flow in the lower layer, measuring the rate at which the flow is expanding or compressing at any point in the lower layer. This is essential for maintaining the consistency of the quasi-geostrophic framework.

Evaporation (2.6) and precipitation (2.8) are defined as follows:

$$E = \begin{cases} \hat{E}|\mathbf{u}_2|(m_s - m), & \text{if } m < m_s, \\ 0, & \text{if } m \geq m_s, \end{cases} \tag{2.6}$$

where $\hat{E}$ is a constant, and m_s represents the saturation of moisture, which is determined through the linearized Clausius-Clapeyron relation (2.7) which simply states that that the saturation moisture content is proportional to the temperature difference between the two layers, represented by the difference in streamfunctions ψ_1 and ψ_2:

$$m_s \equiv CT = C(\psi_1 - \psi_2), \tag{2.7}$$

where C is a constant. In equation 2.5, precipitation is defined as:

$$P = \begin{cases} 0, & \text{if } m \leq m_s, \\ \frac{m-m_s}{\tau_p}, & \text{if } m > m_s, \end{cases} \tag{2.8}$$

where τ_p is the timescale for precipitation. Thus, precipitation occurs wherever the moisture exceeds m_s, indicating a transfer of mass between the layers, while evaporation dominates in regions where moisture is below m_s.

2.1.3 Numerical simulation of the QG system

For the high-resolution numerical simulation of the QG system, considered the "truth", we have used a Fourier-Fourier pseudospectral method. The spatial discretization is spectral in both x and y, where we have retained 128 Fourier modes. The length and width of the domain are equal to 46 and 68, respectively. Sponge layers are applied to the northern and southern boundaries. Note that the domain is wide enough for the sponges to not affect the dynamics. Here $5\tau_{adv} \approx 1$ Earth day $\approx 200\Delta t$, where $\Delta t = 0.025$ is the time step of the leapfrog time integrator used in the numerical scheme, and τ_{adv} is the advection time scale of the system.

2.2 Exclusion of Moisture Channel

From the QG system, we generated three channels of data: ψ_1, ψ_2, and moisture, where ψ_1 is the stream function for the upper layer, and ψ_2 is the stream function for the lower layer. However, since the primary objective of this experiment is to investigate whether the model can effectively learn and predict the behavior of a system that includes moisture dynamics, even when a critical piece of data (moisture itself) is missing. Thus, we have challenged our model by only providing it with partial states, ψ_1 and ψ_2, which implicitly contain

latent information about moisture, despite the exclusion of the generated moisture channel generated by the QG system. Any information about moisture or precipitation is still inherent within the streamfunctions. If the model can still predict the system's evolution accurately, it would indicate that the model has learned to compensate for the missing data by learning the finer details of the underlying dynamics. This approach could be significant in real-world applications, where data availability is frequently limited or incomplete.

Chapter 3

Fourier Neural Operator

A Fourier neural operator (FNO) is a neural network that's been specifically designed to learn mappings between function spaces. Traditional neural networks operate directly in the spatial or temporal domains but FNOs perform computations in the Fourier domain and by doing so allow us to better capture long range dependencies and complex patterns in data.

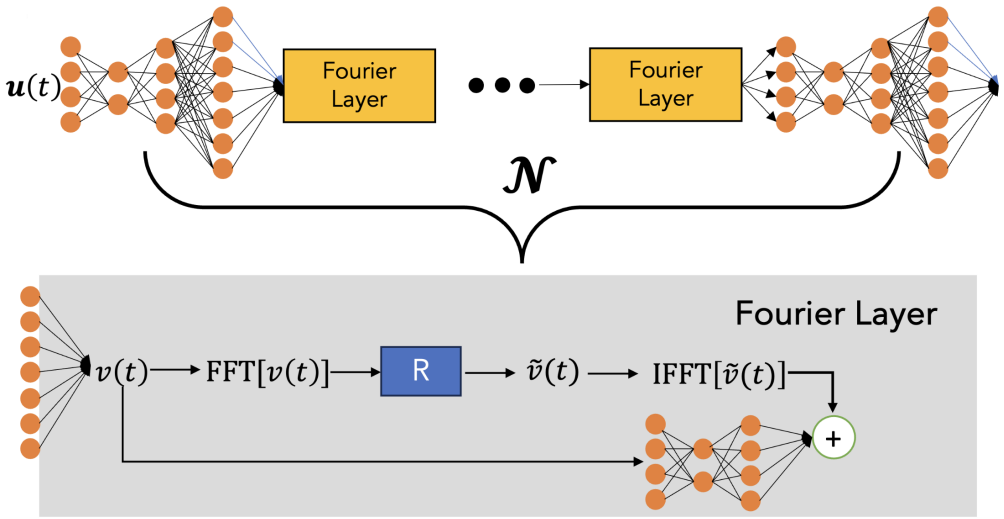

Figure 3.1: The architecture of our model containing $\mathcal{N}$, the FNO, where $u(t)$ is the state of the system that is predicted. This model is further detailed by Chattopadhyay et al [1]. See section 3.3 for more details.

3.1 Spectral Convolutional Layer

The FNO uses spectral convolution layers that perform the fast Fourier transform (FFT) to reduce the complexity of the convolution operation to $\mathcal{O}(n \log n)$ where n is the number of grid points in the spatial domain. Then, an inverse fast Fourier transform (IFFT) is used to reconstruct the spatial features in the original domain after the filtering operation.

Our model applies four layers of integral operators and activation functions, where each Fourier layer retains modes up to 64 using the linear transform R (fig. 3.1). After these operations, the output is projected back to the target dimension by a neural network to produce the final output $u(t)$.

We use two sets of weights to handle the different frequency components of the input data. The first set of weights is responsible for processing the positive frequencies which correspond to the Fourier modes with non negative indices. These frequencies capture the forward propagating waves or the regular, positive oscillations in the data. The second set of weights handles the negative frequencies, which correspond to the Fourier modes with negative indices and are crucial for capturing the backward-propagating waves or inverse oscillations in the data. This is how the model learns and represents the complete range of frequency components present in the input.

Fourier transform

In our FNO, FFT is used to transform spatial data into the frequency domain where we perform convolution operations. Given a 2D input $u(t)$, which in the context of this experiment is a two-channel data input of ψ_1 and ψ_2, the FFT computes the frequency domain representation $\hat{u}(k_x, k_y)$ as:

$$\hat{u}(k_x, k_y) = \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} u(t)\, e^{-2\pi i \left(\frac{k_x x}{N} + \frac{k_y y}{M}\right)} \tag{3.1}$$

where $\hat{u}(k_x, k_y)$ are the Fourier coefficients corresponding to the wave numbers k_x and k_y.

Once in the frequency domain, the spectral convolution is performed by multiplying the Fourier coefficients with learned weights in the frequency domain:

$$\hat{v}(k_x, k_y) = \hat{u}(k_x, k_y) \cdot R(k_x, k_y) \tag{3.2}$$

where $R(k_x, k_y)$ represents the learned filter in the frequency domain. Then we apply the IFFT to bring the transformed data back to the spatial domain:

$$u(t) = \frac{1}{NM} \sum_{k_x=0}^{N-1} \sum_{k_y=0}^{M-1} \hat{v}(k_x, k_y)\, e^{2\pi i \left(\frac{k_x x}{N} + \frac{k_y y}{M}\right)} \tag{3.3}$$

This process allows the FNO to efficiently capture and manipulate patterns in the data across different scales, leveraging the FFT and IFFT to handle convolution operations in the frequency domain.

3.2 Multi-Layer Perceptron

Our FNO uses Multi-Layer Perceptrons (MLP) to complement the spectral convolutional layers, which are used to project the data to higher dimensional spaces or reduce it back to the original dimensions through a series of linear transformations followed by non-linear activation functions. Given an input vector $\mathbf{x} \in \mathbb{R}^d$ the MLP applies a linear transformation using a weight matrix $\mathbf{W} \in \mathbb{R}^{d' \times d}$ and a bias vector $\mathbf{b} \in \mathbb{R}^{d'}$ to obtain a transformed vector in a higher-dimensional space:

$$\mathbf{h} = \sigma(\mathbf{W}\mathbf{x} + \mathbf{b}) \tag{3.4}$$

where σ is a non-linear activation function, and $\mathbf{h} \in \mathbb{R}^{d'}$ represents the data in the new dimensional space. In our model, we use the Gaussian error linear unit (GELU) as our activation function. GELU combines the properties of ReLU and Gaussian noise, defined as $\text{GELU}(x) = x \cdot \Phi(x)$, where $\Phi(x)$ is the cumulative distribution function of the standard normal distribution.

To reduce the dimensionality back to the original or target dimensions, another linear transformation is applied, typically using another weight matrix $\mathbf{W}' \in \mathbb{R}^{d'' \times d'}$ and bias vector $\mathbf{b}' \in \mathbb{R}^{d''}$, followed by an activation function:

$$\mathbf{y} = \sigma(\mathbf{W}'\mathbf{h} + \mathbf{b}') \tag{3.5}$$

Here, $\mathbf{y} \in \mathbb{R}^{d''}$ is the output in the reduced dimensional space.

3.3 Model Architecture

3.3.1 Stepping Schemes for Prediction

The purpose of integrating the stepping schemes with the FNO is to help with error mitigation and provide a structured way to advance the state of the system through time. This ultimately helps with making sure the predictions remain coherent and accurate over multiple time steps, especially for nonlinear systems like the QG system (and many other systems that involve nonlinear dynamics). We have opted to utilize first-order Euler and fourth-order Runge-Kutta schemes that can handle these nonlinearities when advancing the system state and making sure the predictions remain physically meaningful. Additional information regarding the benefits to using these schemes are explained later in this chapter.

These stepping schemes are applied after the FNO generates a prediction. The FNO itself processes input data to produce an output that represents the

state of the system at the next time step. The stepping schemes then take this output and adjust it according to the chosen method (1st order Euler, 4th order Runge-Kutta, or directly) to mitigate errors and attempt to improve the accuracy of the prediction.

As for the choice of schemes, we opted to use differing stepping schemes because they offer varying levels of accuracy and stability that can be useful when attempting to dial in a model's parameters or analyze its success. Generally the RK4 method is known for its higher accuracy compared to the simpler Euler method, which can be less accurate but computationally cheaper.

Direct-step prediction method

Our first stepping scheme is a straightforward one-step prediction method where the FNO is applied directly to the current state to predict the next state without considering intermediate steps or the rate of change.

Similar to Chattopadhyay et al. [1], we define the governing partial differential equation for the reduced atmospherical system using initial condition $X(t)$. The evolution of the state, $\mathbf{X}$, is given by:

$$\frac{d\mathbf{X}}{dt} = \mathbf{F}(\mathbf{X}(t)). \tag{3.6}$$

In order to integrate our system from the initial condition, we define equation 3.6 in its discrete form:

$$\mathbf{X}(t+\Delta t) = \underbrace{\mathbf{X}(t) + \int_{t}^{t+\Delta t} \mathbf{F}(\mathbf{X}(t))\, dt}_{\mathcal{N}[o,\theta]}. \tag{3.7}$$

Here, $\mathcal{N}$ is an FNO that parameterizes $\mathbf{F}$ with four Fourier layers, similar to Li et al. [8], each layer retaining 64 Fourier modes. $\boldsymbol{\theta}$ represents approximately 80×10^6 trainable parameters for the FNO.

Fourth-order Runge-Kutta (RK4) integration scheme

We chose RK4 as a stepping scheme for prediction to dampen error propagation and because it provides a balanced trade off between computational efficiency and accuracy, allowing for stable integration over time. It can also handle non linearities which makes it espsecially well-suited for simulating geophysical fluid dynamics.

Instead of a direct prediction $\mathbf{X}(t+\Delta t)$, as done in Eq. (3.7), we represent $\mathbf{F}(\mathbf{X}(t))$ in Eq. (3.6) with an FNO, $\mathcal{N}[\mathbf{o},\theta]$, with trainable parameters, θ:

$$\mathbf{X}(t+\Delta t) = \mathbf{X}(t) + \underbrace{\int_t^{t+\Delta t} \mathbf{F}(\mathbf{X}(t))\, dt}_{\mathcal{H}[\mathcal{N}[o,\theta]]} \tag{3.8}$$

The layer $\mathbf{H}[\mathbf{o}]$ performs the integration between t and $t+\Delta t$ via the RK4 scheme following Krishnapriyan *et al.* [7] as shown in Eq. (3.3). The operations in the $\mathbf{H}$ are given by:

$$\begin{aligned}
i_1 &= \mathcal{N}\left[\mathbf{X}(t),\theta\right], && \text{(3.7a)}\\
i_2 &= \mathcal{N}\left[\mathbf{X}(t)+\frac{1}{2}i_1,\theta\right], && \text{(3.7b)}\\
i_3 &= \mathcal{N}\left[\mathbf{X}(t)+\frac{1}{2}i_2,\theta\right], && \text{(3.7c)}\\
i_4 &= \mathcal{N}\left[\mathbf{X}(t)+i_3,\theta\right], && \text{(3.7d)}\\
z &= \mathbf{X}(t)+\frac{1}{6}\left(i_1+2i_2+2i_3+i_4\right). && \text{(3.7e)}
\end{aligned} \tag{3.9}$$

We then obtain the predicted state with $z = \mathbf{H}\left[\mathcal{N}\left[\mathbf{X}(t),\theta\right]\right]$.

First-order Euler integration scheme

We chose 1st order Euler step as a stepping scheme for prediction because it offers simplicity and speed. It approximates the solution at the next time step by adding the product of the current derivative and the step size to the current

value.

Given an ODE of the form

$$\frac{d\mathbf{X}}{dt} = \mathbf{F}(\mathbf{X}(t)), \tag{3.10}$$

with an initial condition $\mathbf{X}(t_0) = \mathbf{X}_0$, the Euler method updates the solution as follows:

$$\mathbf{X}_{n+1} = \mathbf{X}_n + h \cdot \mathbf{F}(\mathbf{X}_n, t_n), \tag{3.11}$$

where h is the step size, $\mathbf{X}_n$ is the current state, and $\mathbf{F}(\mathbf{X}_n, t_n)$ is the derivative of $\mathbf{X}$ at time t_n. Our stepping scheme computes the next state using the FNO to estimate the derivative, which is then added to the current input batch, thus advancing the solution by one time step at a time.

3.3.2 MSE Loss Function

The goal of training an FNO is to minimize a loss function thereby improving the model's predictions. The loss function in this context is based on the mean squared error (MSE) between the predicted output and the target. We define this as:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{y}_i - y_i\right)^2, \tag{3.12}$$

where $\hat{y}_i$ is the predicted value, y_i is the true value, and N is the number of samples. The MSE measures the average squared difference between the estimated values and the actual values, providing a way to quantify the accuracy of the predictions.

We apply the loss function at each time step during training to the predicted output $\mathcal{N}(u(t))$ generated by the FNO. Specifically, after every step where

the FNO produces an intermediate prediction $u'(t + \Delta t)$ or a final prediction $u(t + \Delta t)$, the loss function evaluates the difference between these predicted states and the true state $u^{\text{true}}(t + \Delta t)$. This difference is then used to compute the gradient that informs the adjustment of the model parameters through backpropagation.

3.3.3 Spectral Loss Function

The purpose of utilizing a spectral loss function is to measure the prediction error while accounting for spectral bias [1]. Spectral bias refers to the tendency of neural networks to underrepresent high-frequency components, which can lead to inaccurate predictions in models dealing with turbulent flow or other complex systems [1].

The spectral loss function is defined as:

$$L(\theta) = \frac{1}{T} \sum_{t=0}^{T} \left\| u^p(t + \Delta t) - u^{\text{true}}(t + \Delta t) \right\|_2 + \lambda \left\| \hat{u}^p(t + \Delta t) - \hat{u}^{\text{true}}(t + \Delta t) \right\|_2 \tag{3.13}$$

In this equation, $u^p(t+\Delta t)$ denotes the predicted state at time $t+\Delta t$, while $u^{\text{true}}(t + \Delta t)$ represents the true state at the same time. The terms $\hat{u}^p(t + \Delta t)$ and $\hat{u}^{\text{true}}(t + \Delta t)$ refer to the Fourier-transformed versions of the predicted and true states, respectively. The parameter λ is a regularization factor that balances the contributions of the prediction error in the physical domain and the spectral domain.

This loss function helps in mitigating the spectral bias by incorporating the error in both the physical and spectral domains. The first term of the loss function measures the difference between the predicted and true states in the physical domain using the L_2 norm. The second term, weighted by the regularization parameter λ (see section 3.4), measures the difference in the spectral domain, where the Fourier transforms of the predicted and true states

are compared. By including both terms, the loss function encourages the model to accurately predict both the low-frequency and high-frequency components, which we hope that by incorporating into our experiment, will lead to more stable and accurate predictions and result in a noticeable decrease of spectral bias. [1].

3.4 Training Parameters

Detailed below are the values of selected, key parameters for both the QG system and the FNO that were utilized in the training of our network.

Initial Wavenumbers: We use parameters k_{init} and $k_{\text{init},y}$ to specify the initial wavenumbers in the x and y directions, respectively. In the context of the QG system, these define the scale of the smallest waves that the spectral loss function considers in the Fourier-transformed space.

Fourier Modes: The parameter for the Fourier modes, set to 64 for both the x and y directions, controls the resolution of the spectral convolution operation by limiting the number of frequencies considered in each direction.

Spinup Time: The spinup time is set to 10,000 iterations or time steps, defining the initial period required for the QG system to reach a quasi-equilibrium state before the actual data collection or analysis begins. This ensures that the initial transients are not included in the final analysis.

Regularization Parameter λ: Our regularization parameter λ is set to 0.2 in this model and is used in the loss function to balance the mean squared error (MSE) loss and the spectral loss. A smaller value of λ_{reg} places more

emphasis on the MSE loss, which directly measures the error in predicting the physical state, while a larger value emphasizes the spectral loss, focusing on the accuracy of the Fourier-transformed components.

Learning Rate: The learning rate, set to 0.001, controls the step size at each iteration while moving towards a minimum of the loss function. A smaller learning rate allows for finer adjustments to the model parameters but requires more iterations while a larger learning rate speeds up convergence but risks overshooting the minimum.

Epochs: We have set the number of epochs to 40, specifying the number of times the entire training dataset passes through the neural network during training. More epochs can improve model performance by allowing the network more opportunities to learn, but excessive epochs can lead to overfitting.

Chapter 4

Results

Our experiment did not achieve long-term stability, but does reflect some interesting findings about short-term accuracy.

To discuss these findings, we use standard metrics such as root mean square error (RMSE) and anomaly correlation coefficient (ACC) to evaluate the FNOs predictions against the corresponding "truth" time series produced by the QG system. During these evaluations, we use a snapshot from the state vector that serves as the initial condition for the model.

Additionally, we perform analysis on amplitude spectrum plots to gain insights into spectral bias, the decay of amplitude over time, and how well our model retains accuracy across various spectral modes (particularly in predicting small-scale features).

The truth data in this context refers to data generated using the dry QG system. Each timestep in the simulation represents a specific point in the model's evolution, and for the purposes of these analyses, every five timesteps are considered to represent one "day" in the simulated atmospheric system.

4.1 Benchmarks and metrics

4.1.1 Stable Model Benchmark

In addition to the truth, we use a stable, dry-QG FNO model as a benchmark against which to compare our model. The stable model is a direct step model trained with the spectral loss function and specifically trained on all three states of data: the two streamfunction variables, ψ_1 and ψ_2, as well as the third moisture state. Direct access to the moisture data allows the model to access and utilize more than just the latent information about moisture in the system through our two streamfunctions.

This benchmark model was able to achieve numerical stability over 250 days for the particular set of parameters it was trained on. One difference in the stable model's design is its use of a modified regularization parameter λ (see section 3.4). Unlike our experimental model where $\lambda = 0.2$, the stable model's λ was adjusted by dividing it by a factor of $\frac{1}{110}$, resulting in a modified λ value. This adjustment was made after a careful comparison of the average spectral loss to the grid loss, as we had determined that the spectral loss was approximately 110 times greater than the associated grid loss. This is a small adjustment but it is worth noting, as it directly affects the training process.

4.1.2 Root Mean Square Error (RMSE) Metric

The RMSE provides a sense of the total error accumulated through the prediction. The RMSE is calculated simply as the root mean square difference between the truth and the prediction of each variable at each time step as

$$\mathrm{RMSE}_{p,i} = \sqrt{\sum_{j=1}^{N_y}\sum_{k=1}^{N_x}\left(p_{\mathrm{pred},(i,j,k)} - p_{(i,j,k)}\right)^2}. \tag{4.1}$$

where $p_{\mathrm{pred},(i,j,k)}$ is the predicted value and $p_{(i,j,k)}$ is the truth value.

4.1.3 Anomaly Correlation Coefficient (ACC) Metric

The ACC metric is used to assess the similarity between the temporary or short-lived features and variations predicted by the FNO against the "truth" patterns obtained from the QG system. This helps us in identifying extreme events, which are an important part of the climate systems variability. Here "extreme" refers to anomalies with magnitudes farther than average from the temporal mean.

We first establish the temporal mean for any given variable p by

$$p_c(j,k) = \frac{1}{N_t} \sum_{i=1}^{N_t} p(i,j,k), \tag{4.2}$$

where N_t denotes the total number of time steps within the time series, and (j,k) represent the (y,x) spatial indices. Therefore, for each variable p at a specific time step i (with $i = 0$ indicating the initial snapshot used for prediction), the ACC is determined as the correlation,

$$\mathrm{ACC}_{p,i} = \frac{\sum_{j=1}^{N_y} \sum_{k=1}^{N_x} \left(a_{p,i,j,k} \cdot b_{p,i,j,k}\right)}{\sqrt{\sum_{j=1}^{N_y} \sum_{k=1}^{N_x} \left(a_{p,i,j,k} \cdot a_{p,i,j,k}\right) \sum_{j=1}^{N_y} \sum_{k=1}^{N_x} \left(b_{p,i,j,k} \cdot b_{p,i,j,k}\right)}}, \tag{4.3}$$

between the forecasted anomalies

$$a_{p,i} = p_{\mathrm{pred},i} - p_c - \frac{1}{N_y N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{N_x} \left(p_{\mathrm{pred},i,j,k} - p_c(j,k)\right), \tag{4.4}$$

and the truth anomalies,

$$b_{p,i} = p_i - p_c - \frac{1}{N_y N_x} \sum_{j=1}^{N_y} \sum_{k=1}^{N_x} \left(p(i,j,k) - p_c(j,k)\right), \tag{4.5}$$

where N_x and N_y correspond to the number of grid points in each the x and y directions.

4.2 Models trained with MSE loss function

Below we detail the results for three variations of our model trained with the MSE loss function as defined in section 3.3.2: Direct Step with MSE loss, Euler with MSE loss, and RK4 with MSE loss.

4.2.1 RMSE Results

For the upper layer in the QG system (ψ_1), our FNO displays, on average, a higher total error accumulated than the stable model. However, for the lower layer ψ_2, we see a lower accumulation of error between the prediction and truth values. This is a particularly interesting find because the lower layer in the QG system has the only moisture advection seen in between both layers in the system. There could be many reasons for this result but we suspect it may be because of the integration system that is coupled with the FNO, mitigating error as it is supposed to.

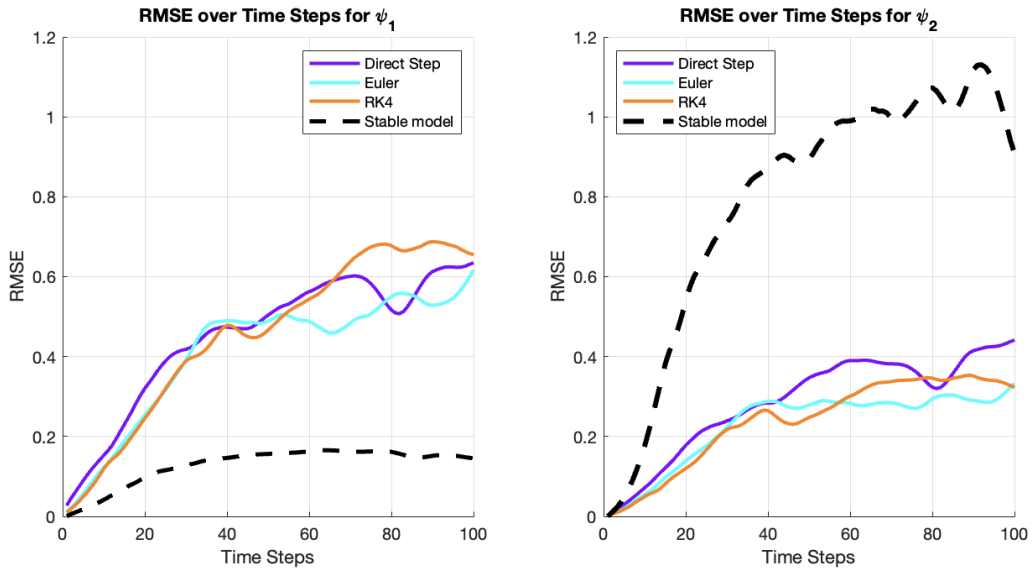

Figure 4.1: The RMSE comparison between the upper (ψ_1) and lower (ψ_2) layers of the QG system for each of the three stepping schemes: RK4, Euler, and the direct-step method wherein no integration is applied.

4.2.2 ACC Results

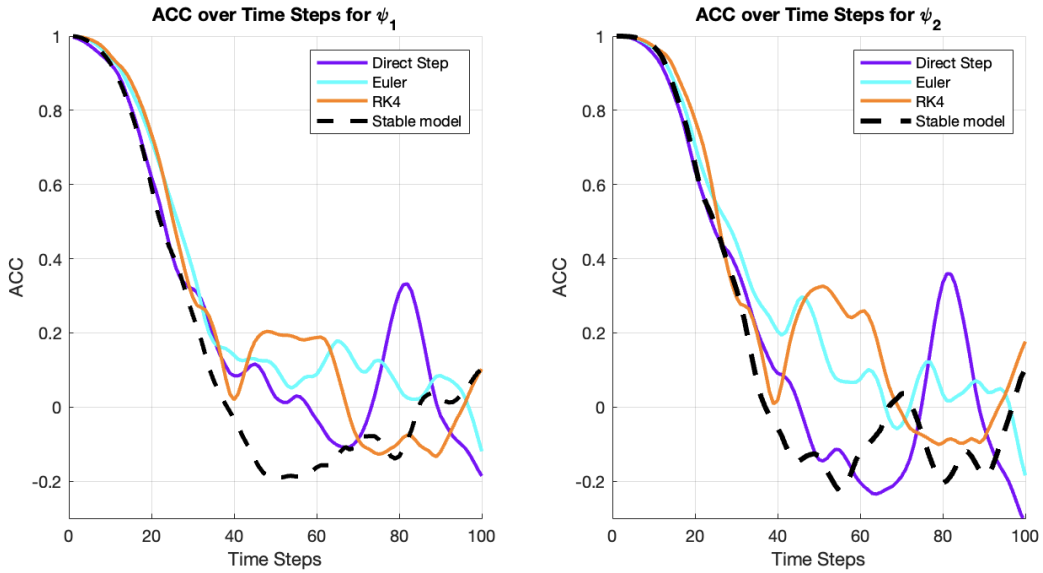

Figure 4.2: The ACC comparison between the upper (ψ_1) and lower (ψ_2) layers of the QG system for each of the three stepping schemes: RK4, Euler, and the direct-step method wherein no integration is applied. In all three cases, long-term instabilities are seen but short-term accuracy is greater than a stable model trained with full states of data.

The ACC results indicate that all three experimental models—Direct Step, Euler, and RK4—are unstable in predicting the evolution of ψ_1 and ψ_2 over time. In particular, the ACC for all models drops below 0.6 relatively early in the time steps, which is a critical threshold indicating that the models no longer provide meaningful predictions. All models fail to maintain an ACC above 0.6 past the 30th time step.

4.2.3 QG system Flows

We use a comparison of our 'truth' data to compare each of our FNO models trained on each of our three stepping schemes: RK4, Euler, and Direct. The stable model has access to additional moisture data, and noticeably shows more accurate and stable predictions across all time steps, as evidenced by the more coherent and consistent structures over time.

In contrast, the three FNO-trained models demonstrate increasing instability, with noticeable differences in the flow patterns and the degradation of predictive accuracy as time progresses. This shows a lack of accuracy over extended time periods without the additional moisture data.

Below we see figures for ψ_1 and ψ_2 of the QG system over time, and for each of the models. As we see in figure 4.3, the models fall under the threshold of meaningful prediction at around 30 time steps, which is around 7 days. This decline in accuracy is significant because it highlights the limitations of the FNO-based autoregressive models when they are not provided with moisture information. The exclusion of moisture seems to cause a quicker degradation in the model's predictive capabilities, particularly as we move beyond the short-term predictions. The visual differences between the prediction and the truth also become increasingly apparent as time progresses with each of the models deviating more significantly from the true states.

In contrast, figure 4.4 shows the behavior of ψ_2 where the patterns are similar but the model degradation appears more rapid possibly suggesting that ψ_2 may be even more sensitive to the exclusion of moisture data or to the particular small-scale dynamics being modeled.

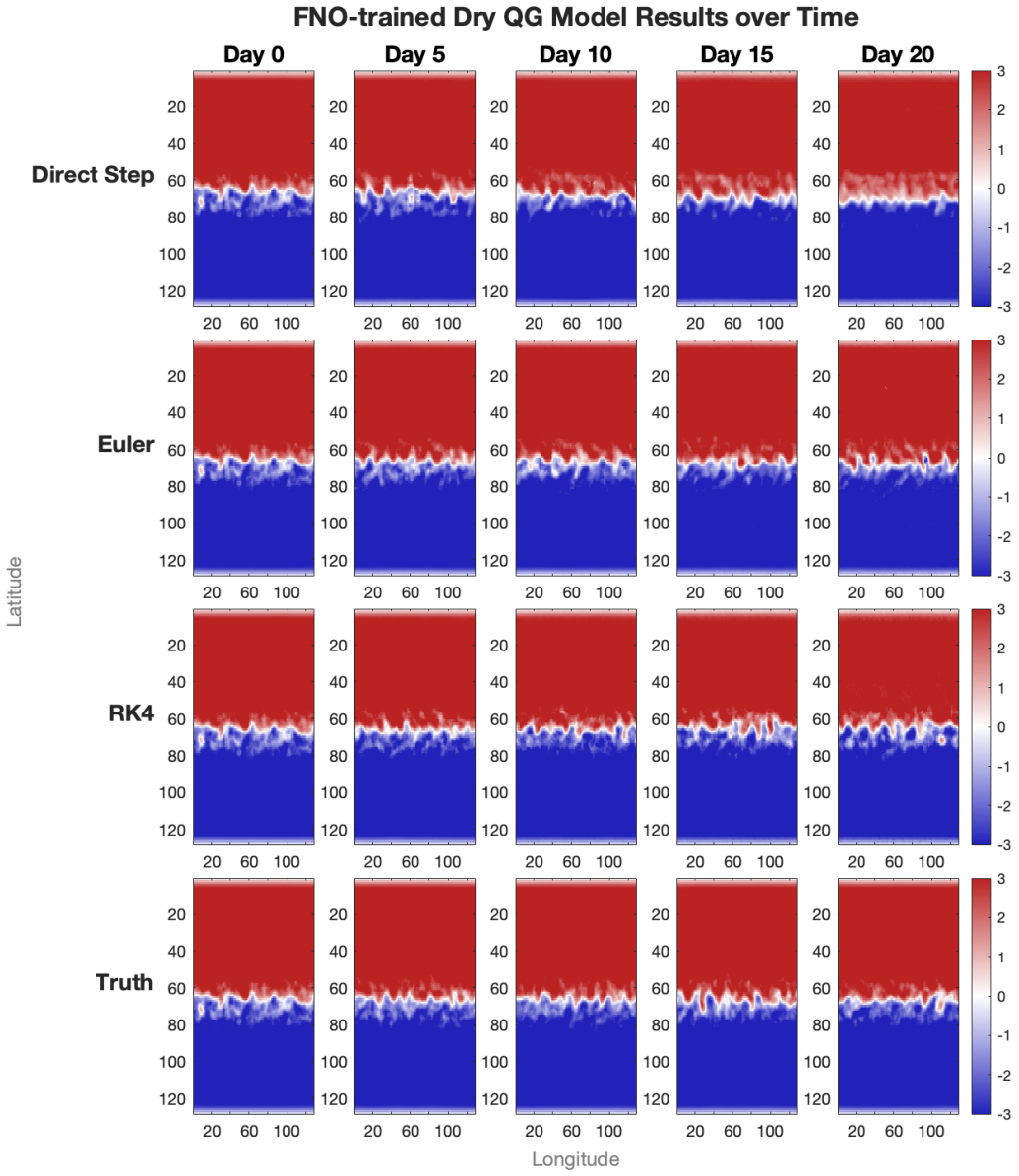

Figure 4.3: Flows for ψ_1 of the QG system as predicted with various FNOs trained on MSE loss. Our FNO is trained on three different stepping schemes, resulting in three models, compared to a stable model benchmark. Each row represents a different model, while each column corresponds to a different day of prediction, where one day is 5 time steps through the FNO.

4.3 Models trained with Spectral Loss

Below we detail the results for three variations of our model trained with the spectral loss function as defined in section 3.3.3: Direct Step with MSE loss, Euler with MSE loss, and RK4 with MSE loss.

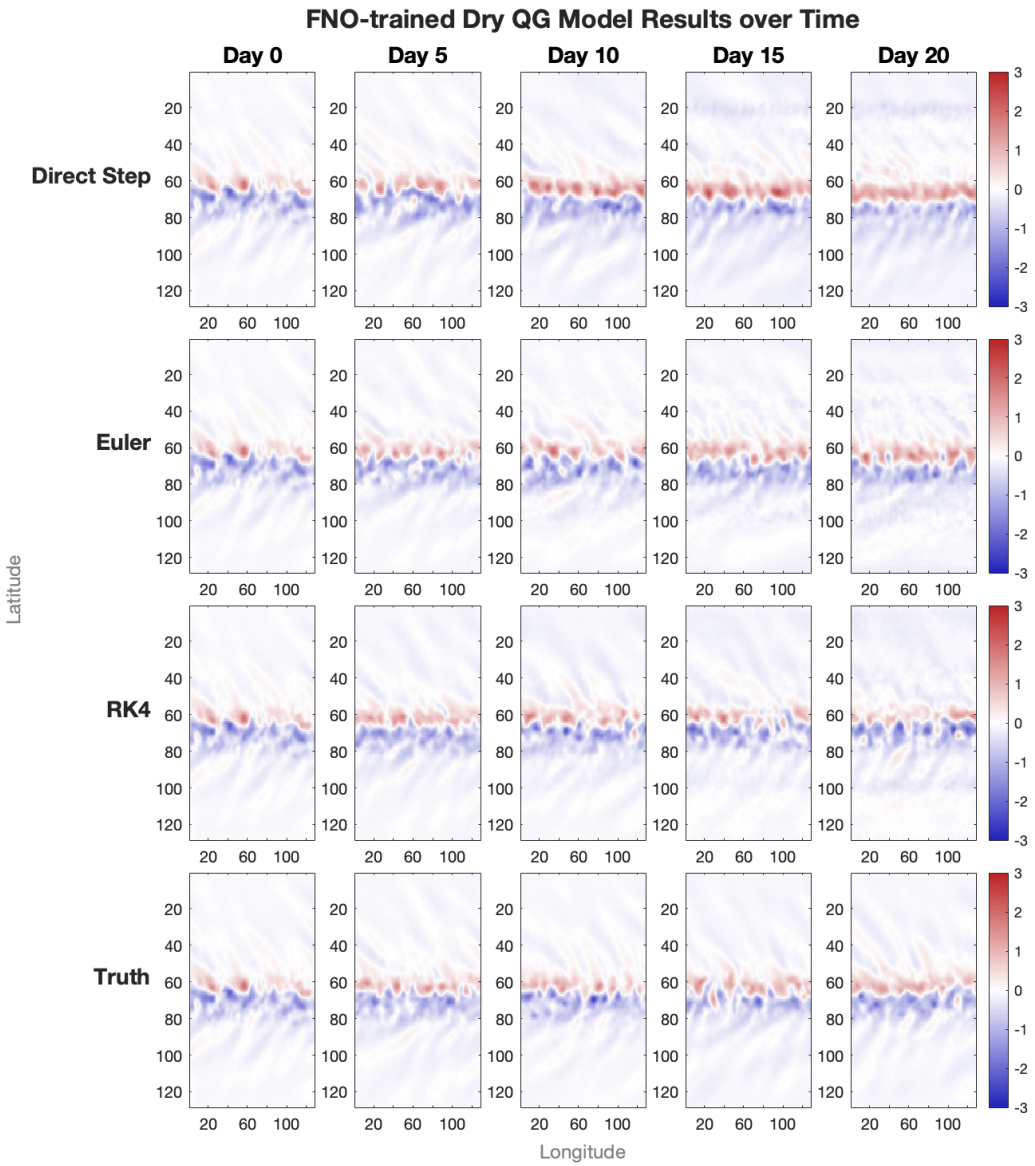

Figure 4.4: Flows for ψ_2 of the QG system as predicted with various FNOs trained on MSE loss. Our FNO is trained on three different stepping schemes, resulting in three models, compared to a stable model benchmark. Each row represents a different model, while each column corresponds to a different day of prediction, where one day is 5 time steps through the FNO.

4.3.1 RMSE Results

Our spectral loss models failed to achieve stability and, in fact, exhibited rapid divergence as is seen in the RMSE plot (figure 4.5). Despite testing various regularization parameters, this scenario represented the best outcome, or the slowest divergence, for models employing spectral loss on partial states. In-

terestingly, the direct step model showed some signs of stabilization, although the error remains significantly high. This observation is noteworthy, especially considering that our stable benchmark model also utilizes a direct step approach with spectral loss. Among the six models we trained, this was the one I was most optimistic about. Although further training is certainly needed, the current results provide intriguing insights.

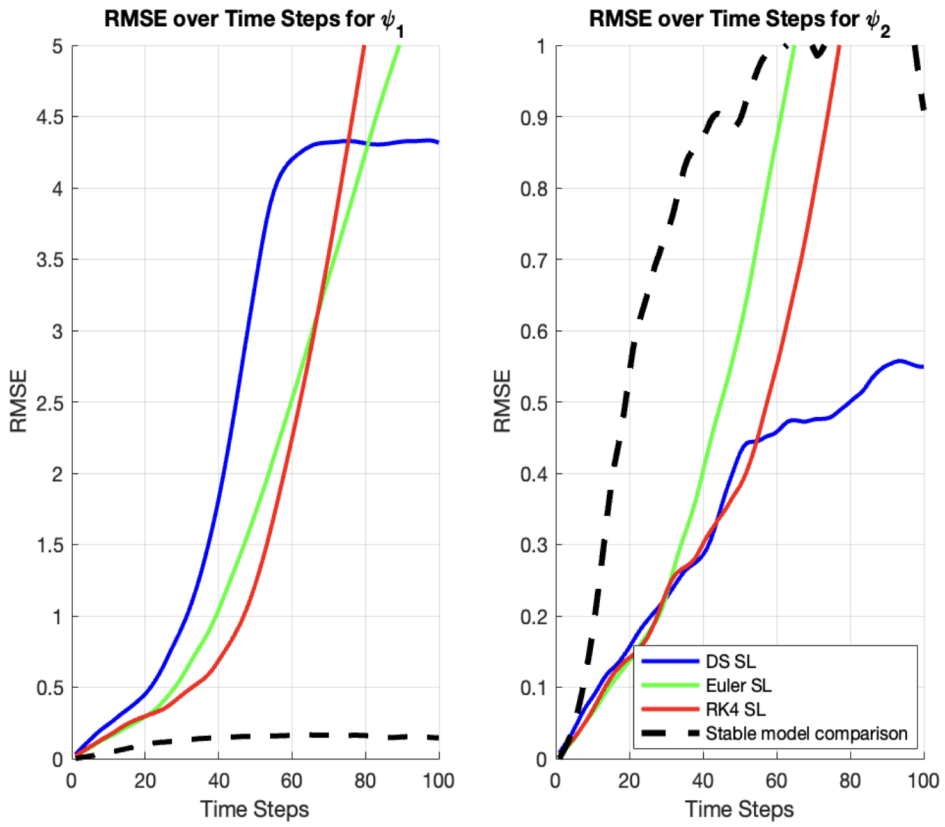

Figure 4.5: The RMSE comparison between the upper (ψ_1) and lower (ψ_2) layers of the QG system for each of the three stepping schemes: RK4, Euler, and the direct-step method wherein no integration is applied.

4.3.2 ACC Results

The ACC plot (figure 4.6) suggest some short-term accuracy despite the lack of stability. This is possibly due to the hard constraint of the integration scheme forcing the model towards lower error rates, as discussed in Chapter 3.

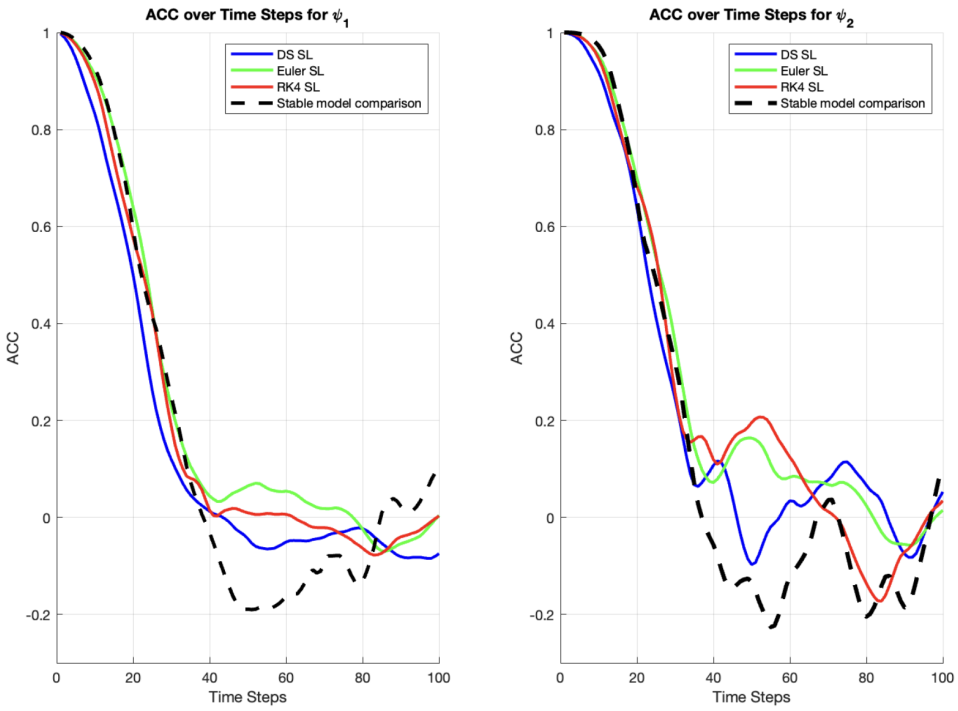

Figure 4.6: The ACC comparison between the upper (ψ_1) and lower (ψ_2) layers of the QG system for each of the three stepping schemes: RK4, Euler, and the direct-step method wherein no integration is applied. In all three cases, long-term instabilities are seen but short-term accuracy is greater than a stable model trained with full states of data.

4.3.3 QG system Flows

As described in section 4.2.3, we use a comparison of our 'truth' data to compare each of our FNO models trained on each of our three stepping schemes: RK4, Euler, and Direct. With the models that were trained with the spectral loss scheme, the three FNO-trained models quickly lost stability, with drastic differences in the flow as soon as day 10, especially for ψ_2.

This quick degradation in the model's predictive capabilities despite a loss scheme explicitly meant to mitigate error is counterintuitive and suggests that additional experimentation is needed. The visual differences between the prediction and the truth become immediately apparent as time progresses with each of the models deviating more significantly from the true states.

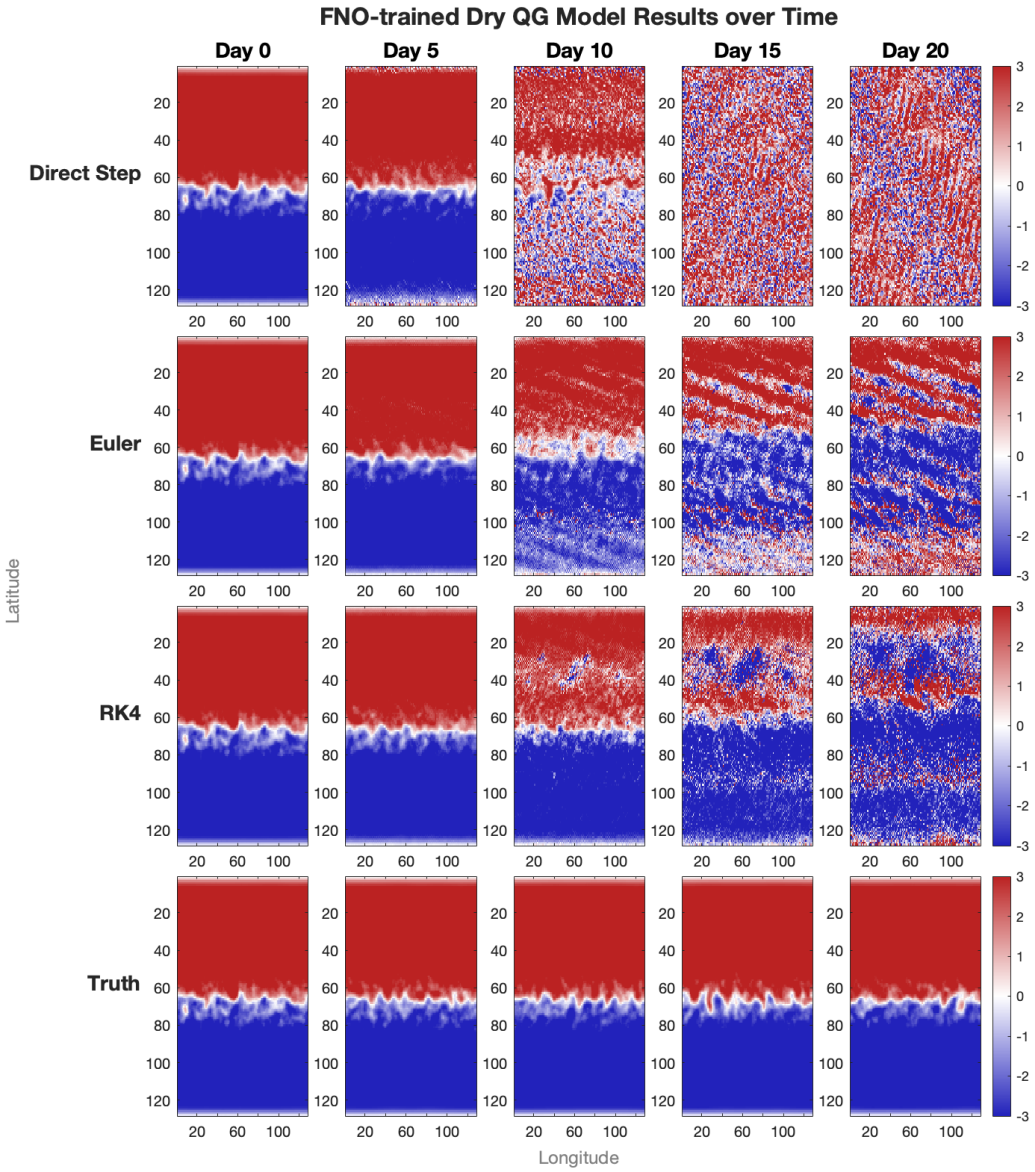

Figure 4.7: Flows for ψ_1 of the QG system as predicted with various FNOs trained on spectral loss. These flows are shown at different days in the experiment, compared against the 'truth' data. Our FNO is trained on three different stepping schemes, resulting in three models, compared to a stable model benchmark. Each row represents a different model, while each column corresponds to a different day of prediction, where one day is 5 time steps through the FNO.

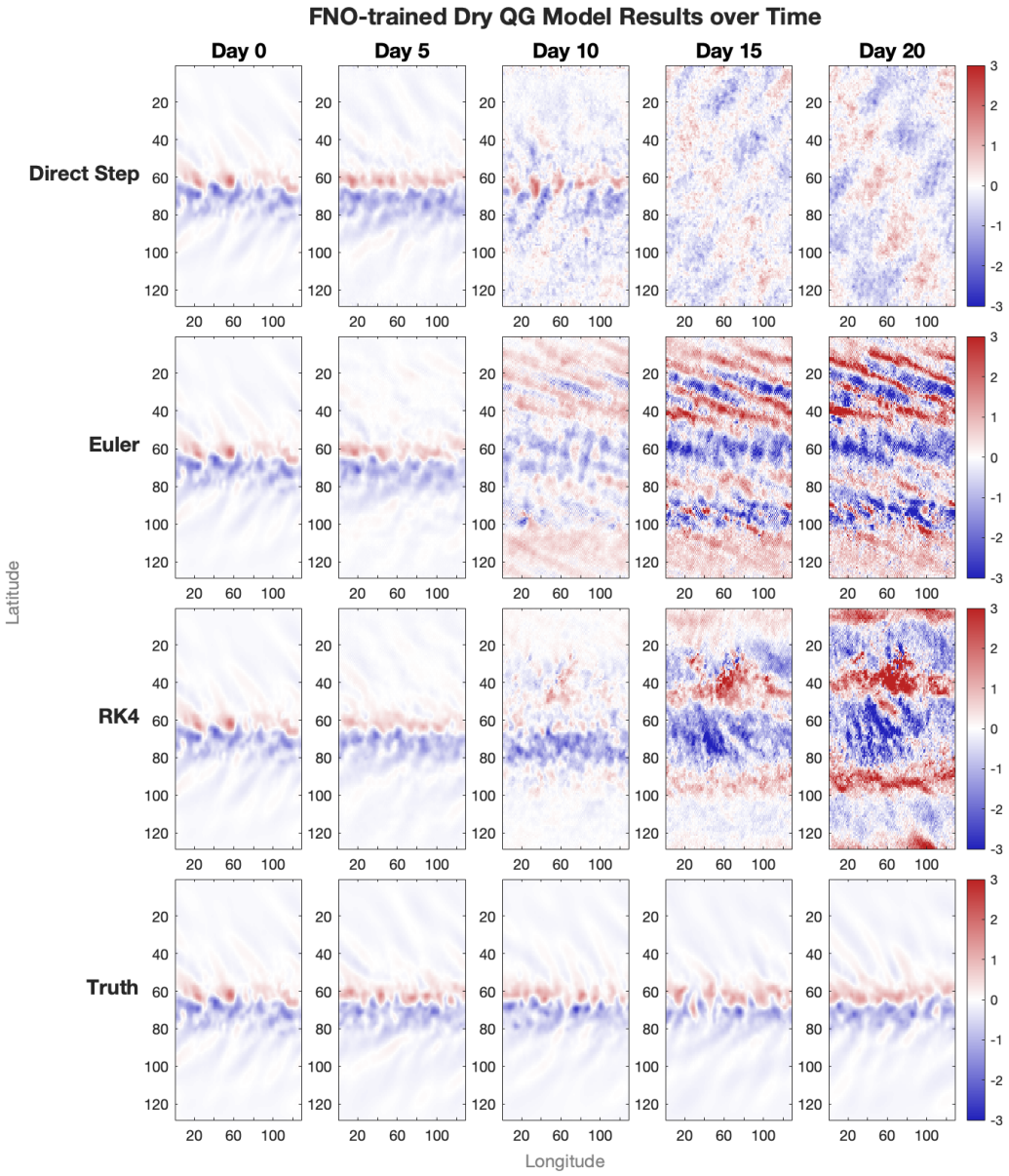

Figure 4.8: Flows for ψ_2 of the QG system as predicted with various FNOs trained on spectral loss. These flows are shown at different days in the experiment, compared against the 'truth' data. Our FNO is trained on three different stepping schemes, resulting in three models, compared to a stable model benchmark. Each row represents a different model, while each column corresponds to a different day of prediction, where one day is 5 time steps through the FNO.

Chapter 5

Discussion

In this project, we report the development of an FNO-based autoregressive model for predicting Ψ_1 and Ψ_2 of two-layer QG moist system without explicit access to moisture information. We showed that a hard-constrained higher-order integrator inside the FNO architecture results in better prediction accuracy and lower error growth. While this is a prototypical system for weather and climate, we hope to extend this work to systems that are close to fully coupled ocean-atmosphere. Here, we discuss some of the challenges, ongoing work, and future directions we have to investigate.

5.1 Impact of Moisture Exclusion

While the model showed comparable short-term performance in predicting ψ_1 and ψ_2 in the absence of moisture information, we could not realize a stable model despite the inclusion of the integration scheme. This is expected since the absence of small-scale physics typically available to the system via the moisture data would eventually lead to a mismatch in the energy budget of the system. However, since most systems that are operational would only have a few observable states, further work into understanding how these small-scale

processes can be parameterized in data-driven models need to be investigated.

5.2 Additional Experiments with Spectral Loss

One of the ways that previous work [2] had incorporated small-scale information was by correcting the spectrum of the dynamics in the small-scales with a spectral regularizer. However, that model still had access to all the states. In our experiments, preliminary results show that the spectral regularizer does not improve our performance and in fact leads to quick numerical blow up. While we conducted searches in the scaling parameter that weighs the regularizer, further work in this direction may improve the stability of the model.

5.3 Considerations on λ

The choice of the regularization parameter λ plays a very important role in balancing the different components of the loss function particularly when incorporating spectral regularization. In this study we set λ to 0.2 as it provided the closest match to stabilize the model during training. However, it is important to note that this value was chosen based on limited experimentation due to the constraints of training on partial data.

While $\lambda = 0.2$ offered a preliminary solution it is evident that further tuning could potentially improve model performance and stability especially as the spectral regularization is highly sensitive to this parameter. More extensive experimentation with different λ values might reveal a better balance that mitigates the issues observed, such as quick numerical blow ups. However the need for a more optimal regularization parameter becomes even more pronounced given that the FNO was trained without explicit moisture information, which

likely exacerbated the challenges in achieving stable, long term predictions. Future work will need to involve more comprehensive searches in the parameter space, considering not only λ but also other aspects of the loss function, to fully understand its impact on model stability and performance.

Bibliography

[1] Chattopadhyay, A., Gray, M., Wu, T., Lowe, A. B., & He, R. (2023). *OceanNet: A principled neural operator-based digital twin for regional oceans*. arXiv preprint arXiv:2310.00813.

[2] Chattopadhyay, A., & Hassanzadeh, P. (2023). *Long-term instabilities of deep learning-based digital twins of the climate system: The cause and a solution*. arXiv preprint arXiv:2304.07029.

[3] Chattopadhyay, A., Nabizadeh, E., Bach, E., Hassanzadeh, P. (2023). *Deep learning-enhanced ensemble-based data assimilation for high-dimensional nonlinear dynamical systems*. Journal of Computational Physics, 477, 111918.

[4] Gavriel, N., Chattopadhyay, A. K., Shaw, T., and Hassanzadeh, P. *Can AI-based climate models learn rare, extreme weather events?*, vol. 2023.

[5] Georgeson, L., Maslin, M., Poessinouw, M. (2017). *Global disparity in the supply of commercial weather and climate information services*. Science advances, 3(5), e1602632.

[6] Kovachki, N., Lanthaler, S., Mishra, S. (2021). *On universal approximation and error bounds for Fourier neural operators*. Journal of Machine Learning Research, 22(290), 1-76.

[7] Krishnapriyan, A.S., Queiruga, A.F., Erichson, N.B., Mahoney, M.W.: *Learning continuous models for continuous physics.* arXiv preprint arXiv:2202.08494 (2022)

[8] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., & Anandkumar, A. (2020). *Fourier neural operator for parametric partial differential equations.* arXiv preprint arXiv:2010.08895.

[9] Lutsko, N. J., & Hell, M. C. (2021). *Moisture and the Persistence of Annular Modes. Journal of the Atmospheric Sciences*, 78(12), 3951-3964. https://doi.org/10.1175/JAS-D-21-0055.1

[10] Nguyen, T., Shah, R., Bansal, H., Arcomano, T., Madireddy, S., Maulik, R., ... Grover, A. (2023). *Scaling transformer neural networks for skillful and reliable medium-range weather forecasting.* arXiv preprint arXiv:2312.03876.

[11] Pedlosky, J. (2019). *The effect of beta on the downstream development of unstable, chaotic baroclinic waves.* Journal of Physical Oceanography, 49(9), 2337-2343.

[12] Stachenfeld, K., Fielding, D.B., Kochkov, D., Cranmer, M., Pfaff, T., Godwin, J., Cui, C., Ho, S., Battaglia, P., Sanchez-Gonzalez, A. (2022). *Learned simulators for turbulence.* In: International Conference on Learning Representations.