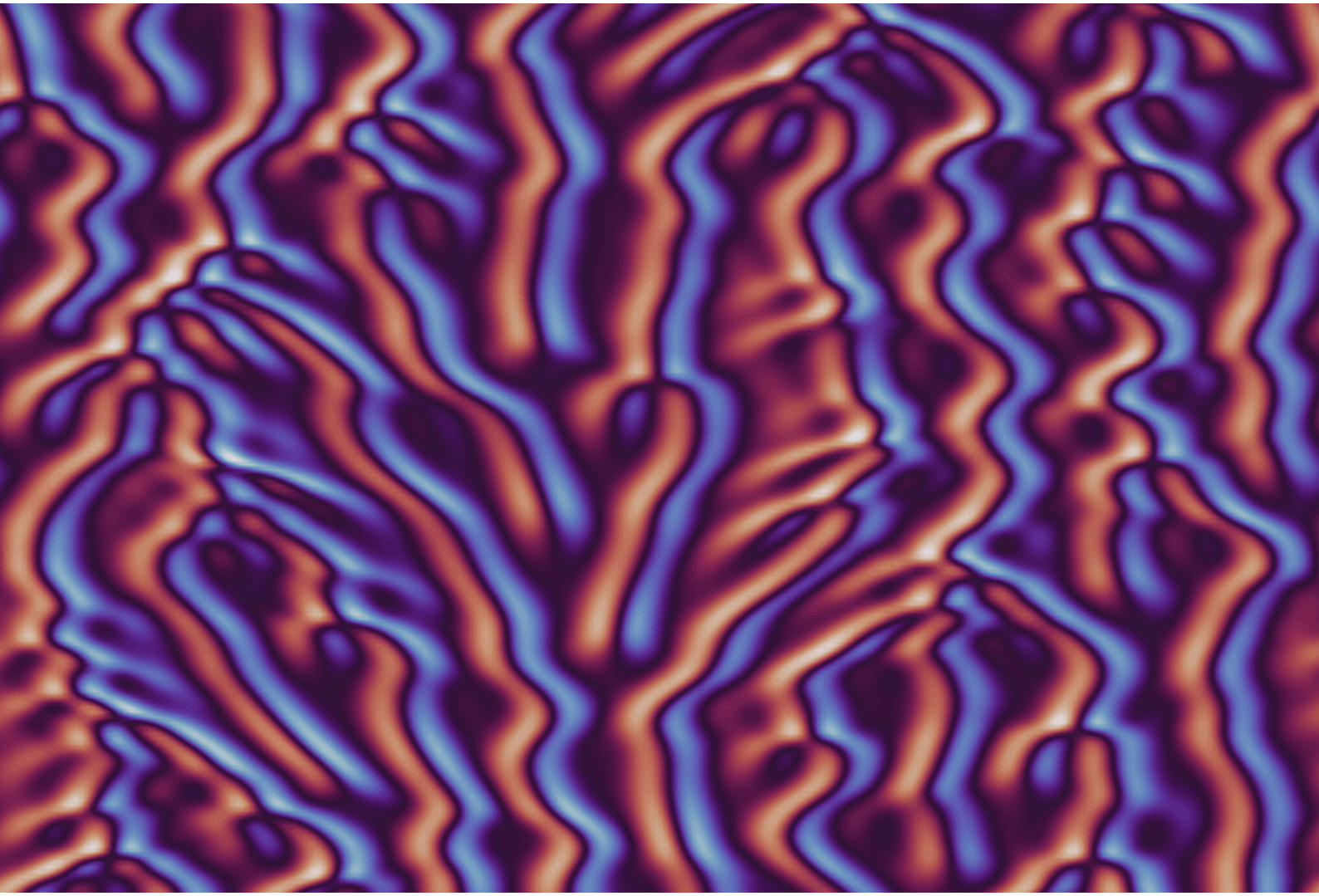
Can Fourier Neural Operators Replicate the Intrinsic Predictability of Spatiotemporal Chaos?

for the Kuramoto-Sivashinsky System



TU Delft
Delft University of Technology

K. R. Schuurman

Can Fourier Neural Operators Replicate the Intrinsic Predictability of Spatiotemporal Chaos?

for the Kuramoto-Sivashinsky System

by

K. R. Schuurman

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday May 12, 2023 at 15:45 AM.

Student number:	4557859	
Project duration:	September 1, 2022 – May 12, 2023	
Thesis committee:	Dr. S. Basu,	TU Delft, supervisor
	Prof. dr. P. A. Siebesma,	TU Delft
	Dr. M. A. Schleiss,	TU Delft
	Dr. R. Taormina,	TU Delft

TU Delft

Preface

Modelling the atmosphere was the reason I picked Geosciences & Remote Sensing. I was fascinated by how we can model something so untangibly complex as the fluid flow in our atmosphere. From a mathematical standpoint, the computational models used in weather forecasting are fascinating. In the last few years, the deep learning revolution brought a new type of model that can forecast the weather through training on the giant vault of weather datasets there are. The intersection between a bleeding edge type deep learning model combined with complex weather systems led me to this research. The journey taught me exciting topics in chaos theory, functional analysis, and predictability. The common knowledge that we should take a two-week forecast with a grain of salt became quantifiable for me. I hope to help the reader develop a similar understanding of weather predictability.

I would like to, first and foremost, thank my supervisor Dr. S. Basu for guiding me through my thesis. Basu knew how to excite me about the current state of weather forecasting. He involved me in the atmospheric sciences community, advised me on my career, and helped me form my ambitions. He always managed to steer me in new directions when I tunnel-visioned within my research. He is one of the most passionate researchers that I have met and could not have wished for a better supervisor. I am also very grateful for the other thesis committee members, whose constructive criticism helped me while their excitement showed me that my work was not for nothing. I want to thank my partner for supporting me throughout the year with the many ups and downs while letting me practice my presentations in her presence. Thanks should also go to my parents, who helped comb through my report for grammar mistakes. I'd like to acknowledge the assistance of chatGPT in rewriting some smaller parts of the report when I lost the patience to do it properly.

K. R. Schuurman
Delft, May 2023

Abstract

The use of deep learning in global weather forecasting has shown significant promise in improving both forecasting accuracy and speed. Traditional numerical weather prediction models have gradually improved forecasting skills but at the cost of increased computational complexity. In contrast, new deep learning models, trained directly on reanalysis data, have demonstrated significant gains in forecasting accuracy, achieving competitive levels of performance.

However, the potential of deep learning in predicting spatiotemporal chaotic systems, such as weather patterns, remains unexplored. To address this gap, we investigate the efficacy of a data-driven Fourier neural operator Markovian forecaster to replicate the intrinsic predictability and the characteristic Lyapunov spectrum of the Kuramoto-Sivashinsky system. FNO reproduces intrinsic predictability and precisely estimates the characteristic Lyapunov spectrum, even with a small dataset. They cannot represent one of the invariant symmetries, a zero characteristic Lyapunov exponent in the spectrum.

Our findings suggest that deep learning can not only enhance the speed and accuracy of traditional numerical forecasting models but also replicate the weather's chaotic nature. This has significant implications for generating large ensembles and improving overall probabilistic forecasts.

The research is limited to a deterministic system defined on a single process time scale surrogated by FNO. The future merits a similar study to investigate if FNO models can perform similarly on larger, stochastic, coupled, and/or multiple time-scale spatiotemporal chaotic systems.

Contents

1

Introduction

In 1922 Lewis Fry Richardson was the first to idealize modelling a global weather system using numerical methods [1]. At the time, Richardson lacked a way to compute the numerical simulation efficiently. That did not stop him from theorizing how it would work. Richardson imagined a theatre in the form of a globe filled with computers [people who compute]. Each computer would calculate the physical equations related to their box in the theatre and communicate them to the adjacent computers. At the centre of the orchestra of computers would be a conductor guiding the entire process and ensuring no computer slacked behind.

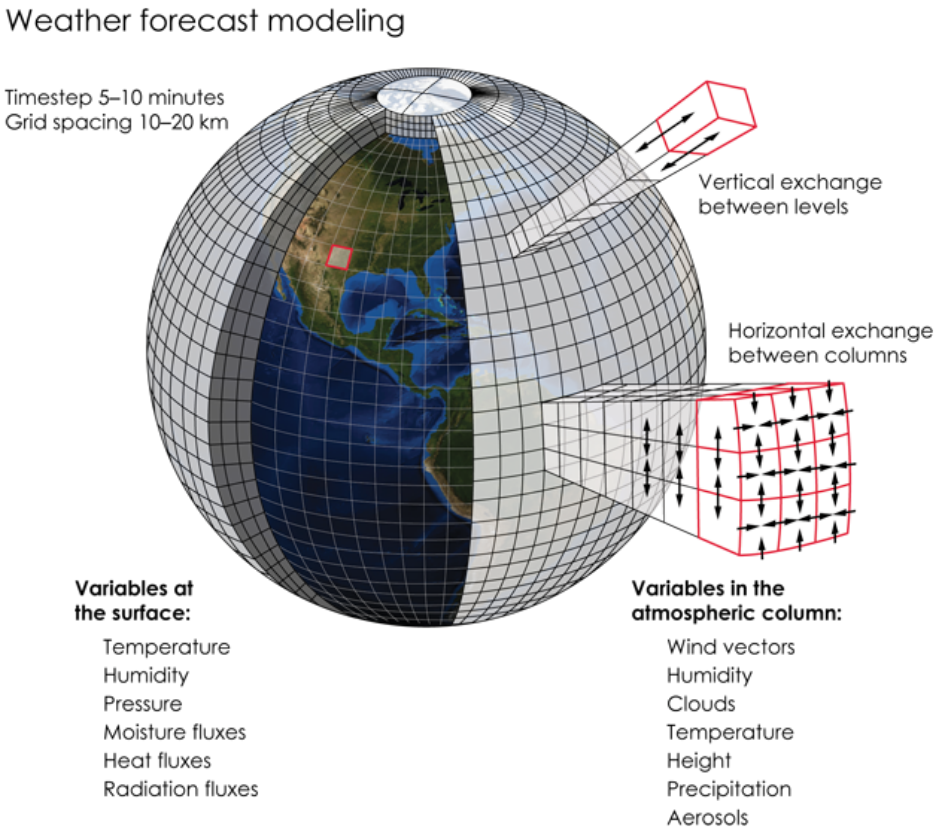


Figure 1.1. Graphical explanation of a weather model. Credit: Cantner K., AGI

The idea of the process being divided into smaller boxes is a reality that has come true. Numerical Weather Prediction (NWP) appeared with the arrival of modern computers. NWP divides the

atmosphere into smaller grid boxes, each connected with adjacent grid boxes. In figure 1.1, a graphical representation of these grid boxes spanning the globe is given. Within each box, a set of equations specify the evolution of a group of variables, including temperature, precipitation, humidity, wind vectors and aerosols. The boxes communicate to each other in the form of exchanging fluxes. As an example, a persistent northerly wind will convect aerosols from a northern grid box to a southern grid box. Subgrid scale processes are parametrised, such as the small-scale deviations in humidity across distances within the grid or drag induced by the topography. The parametrization adds the subgrid processes to the bulk forces weighing on the grid box. After the communication between the grid boxes, a consistent time-stepping is applied to propagate the system further in time from an initial weather state.

These painstaking processes have never been hand calculated by a group of human computers locked in a theatre. Still, Richardson did get to experience the first functional weather forecast by a modern computer ENIAC in 1950. He responded that the results were an "enormous scientific advance". It took ENIAC 24 hours to produce a 24-hour forecast [2].

The first NWPs had coarse resolutions and limited value because they could only forecast the weather a few days in advance. After these few days, the forecasts would deviate too far from reality. What made it more difficult was that it was impossible to observe the global weather over the entire earth, meaning the starting point of each forecast was very uncertain. These initial condition uncertainties have a significant impact on the predictability of an NWP model.

Nowadays, state-of-the-art NWP uses grid boxes in order of 0.1° latitude longitude with many more vertical levels than before. The governing equations are essentially the same, but the numerical methods for solving them have evolved. Many more new processes and effects on subgrid scales were also added to the parametrizations. Simultaneously, by assimilation of satellite observations and local observations, it is possible to get a current state of the weather more accurately. These combined developments have increased the predictability score, which makes it possible to forecast around 10 days ahead of time using deterministic forecasts [3].

On the flip side of all the improvements, the needed computing power increases by a power law. With every decrease in grid size by a power of 10, the amount of computing necessary is raised by an order of 1000. Currently, the global NWP of the European Center for Medium-Range Weather Forecasts (ECMWF) runs on a supercomputer with over one-million cores and a total output of 30 petaflops [E1]. The necessary growth in computing power and associated power consumption is plotted against the average grid size in figure 1.2. Even though computer transistor count has evolved exponentially (according to Moore's Law), higher resolution forecasts start to cost more and more energy. The extensive material and energy cost makes global NWP a task for big institutes.

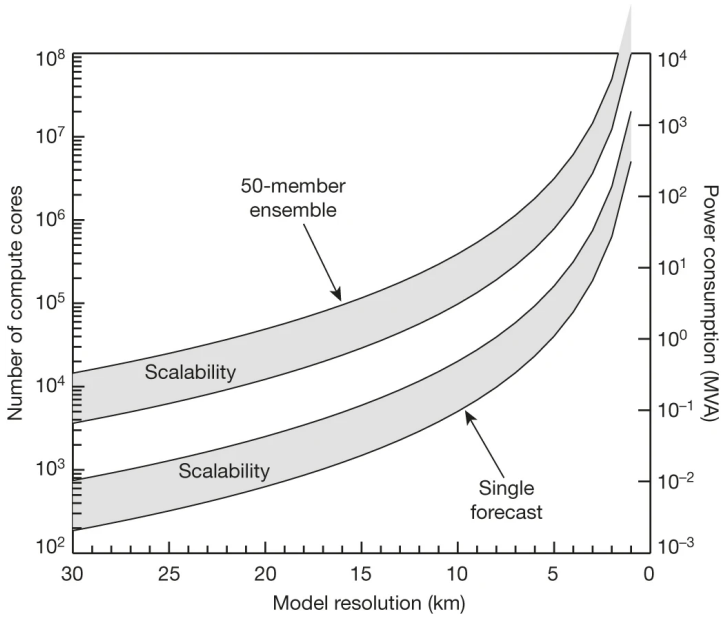


Figure 1.2. Schematic evaluation of the required compute cores and energy for global weather forecasting models. Bauer et al. 2015 [3]

A century later, a new revolution in Deep Learning (DL) emerged, opening up many exciting possibilities. DL has found its way into challenging or unappealing domains for humans, such as image classification, translation, essay writing, and various other automation tasks prone to human errors.

Weather forecasting has conformed to the trend of the DL revolution. Currently, it is possible to run global weather forecasts with DL. Even though the dislike of many atmospheric physicists and meteorologists, DL forecasts of the global weather display a lot of potential. DL forecasting models train solely on historical data without the constraints of fundamental laws of physics. In figure 1.3, a DL model named Pangu-Weather is used to predict the track of hurricane Michael in 2018. The model is presented against the best deterministic forecasting model available in 2022, the High-Resolution Forecast (HRES). Where HRES deviates from the ground truth earlier, Pangu-Weather predicts the track almost perfectly until the storm crosses the Atlantic Ocean. At the time, this HRES forecast was used by ECMWF to boast its accuracy during hurricane Michael (P. Siebesma, personal communication, April 27, 2023).

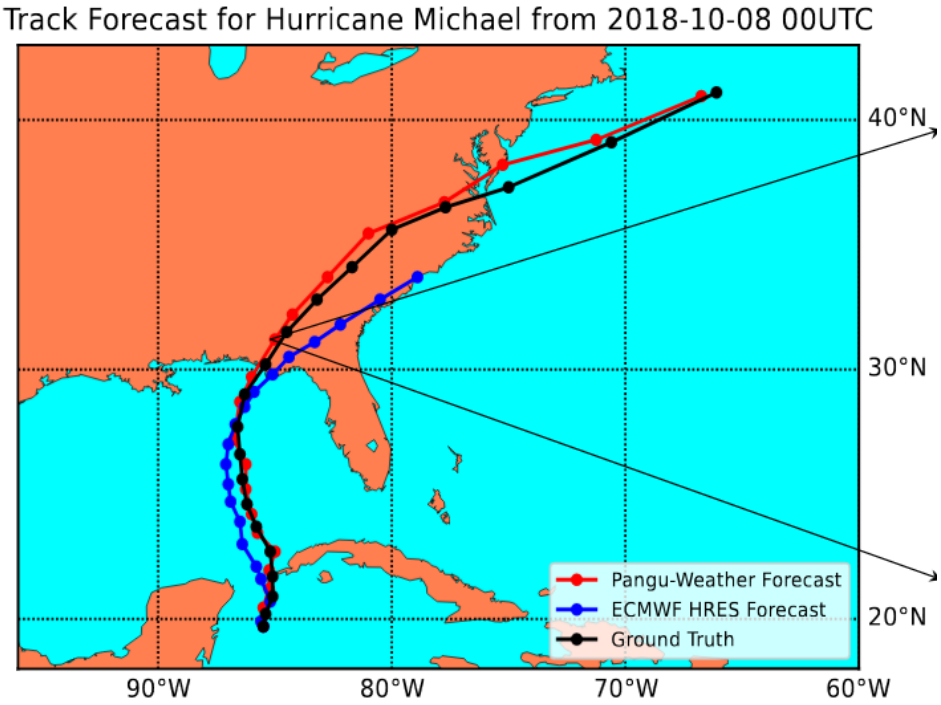


Figure 1.3. Pangu-Weather performance on the Hurricane Michael tracking compared to HRES deterministic ECMWF forecast. Adapted from Bi et al. 2022 [4]

The first DL model to compare itself to HRES was by Weyn et al. 2021 [5]. The main advantage of using a DL model is the computing power required for a forecast. DL models are accelerated on GPU hardware which efficiently parallelises the computation. Secondly, DL can take larger timesteps for forecasting than NWP without instabilities, reducing the computing power significantly. In table 1.1 the largest DL-based global weather prediction networks are summarized as of April 2023. Compared to an NWP, a single 10-day forecast Integrated Forecasting System (IFS) running on thousands of CPU nodes, the computational time cost is estimated to be on order 10^5 lower [6]. The performance gains in computer power would hint that running weather forecasts will be possible on a future laptops instead of waiting for forecasts published by weather institutes.

However, there are many questions left unanswered about the DL forecasting methods. The NWP models are chaotic, and the problems of predictability and limit of predictability are not studied yet. The unpredictable behaviour in weather prediction is a form of chaos. In 1961 Edward Lorenz was the first to document chaos when a seemingly predictable deterministic system showed erratic behaviour. Lorenz worked with a simplified weather model simulating convection in the atmosphere [11]. Lorenz ran the model numerically with certain initial conditions, similar to how a weather forecast is made from an initial weather state. Running a numerical model in those days involved punching holes in a

Model	Published in	Type	Resolution	Efficiency	Hardware
Weyn et al. [5]	2021	CNN	1.4°	320-member forecast in 3 1/2 minutes	NVIDIA Tesla V100
FourCastNet [6]	2022	U-FNO	0.25°	24-hour 100-member forecast in 7 seconds	4 NVIDIA A100
Keisler et al. [7]	2022	GNN	1.4°	5-day forecast in 0.8 seconds	NVIDIA A100
Pangu-Weather [4]	2022	Transformer	0.25°	24-hour forecast in 1.4 seconds	4 NVIDIA Tesla V100
GraphCast [8]	2022	GNN	0.25°	10-day forecast in under 60 seconds	Cloud TPU v4

Table 1.1. Efficiency of the current DL global weather forecasting models. The abbreviations are CNN; Convolutional Neural Network, AFNO; Adaptive Fourier Neural Operator [9, 10], GNN; Graph Neural Network.

card to program the computer. The results would come out in the form of printouts. At some point in his research, Lorenz wanted to revisit a simulation he had done previously. To save time, Lorenz used a printout from his previous simulation as the initial condition for his second simulation. The second time he ran the simulation, the results came out entirely different.

The culprit in Lorenz's work was the printout paper he used as starting point for his second simulation. During printing, the computer rounded off the last three decimals of the internally stored 6-decimal variables. While the effect of the round-off would be considered to be a minimal deviation because of the small relative error, in Lorenz's simulation, the slight deviation in initial conditions caused a wildly different simulation outcome [11].

Lorenz studied this case of chaos and popularized the famous concept: A sensitive dependence on initial conditions. The fact that Lorenz discovered chaos in a simplified weather model was no coincidence because research has shown that the weather is a chaotic system. However, the weather is not the only system that contains chaos. In Lorenz's own words

> "Two states differing by imperceptible amounts may eventually evolve into two considerably different states ... If, then, there is any error whatever in observing the present state-and in any real system such errors seem inevitable-an acceptable prediction of an instantaneous state in the distant future may well be impossible....In view of the inevitable inaccuracy and incompleteness of weather observations, precise very-long-range forecasting would seem to be nonexistent."

The Lorenz butterfly wings shown in figure 1.4 are a product of Lorenz's model. The butterfly wings would later become famous in popular culture under the quote and many different variants: "Predictability: does the flap of a butterfly's wings in Brazil set off a tornado in Texas?". The sentence implies that a tiny weather disturbance in Brazil can set off a tornado in due time. In practice, this is a gross generalization of sensitivity to initial conditions.

The importance of chaos to our research is related to the predictability of our weather. From previous NWP simulations, it is known that some weather states have higher predictability than others. Given certain sea-surface temperature anomalies, the weather can be entirely predictable in the tropics [12]. While in some other cases, such as with the forecast of the severe storm 'Lothar' in 1999, the predictability was limited to 4 days [13]. Figure 1.4 shows changing predictability in the case of the Lorenz system. Three different initial conditions are picked. A group of equally perturbed points around these initial conditions are propagated in time. Situation (a) depicts a highly predictable state where the ensembles stay together, (b) depicts a less predictable state and (c) depicts a highly unpredictable state in which the spread explodes quickly.

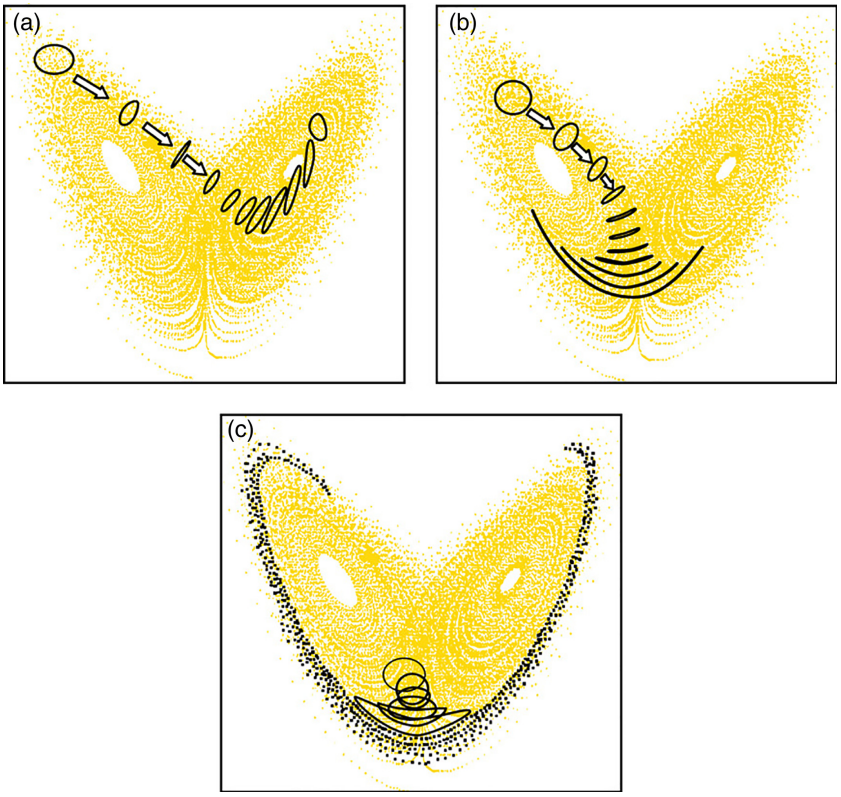


Figure 1.4. Predictability of the Lorenz system by Palmer et al. 2019 [14]

The group of perturbed points is named a group of ensembles. Equivalently, ensembles are used for NWP to quantify predictability in weather. Meteorologists use ensembles to specify in which timescale weather is predictable given an initial weather state. Supercomputers run large ensembles at large weather prediction institutes, such as ECMWF, NOAA/NCEP and KNMI, to forecast where different ensemble members could evolve to. In ensemble forecasting, a large group of models is initialized with slightly different initial conditions from the current weather state, together with differences in boundary conditions, parametrizations and numerical schemes. Different parametrization schemes accentuate different processes in the atmosphere, and specific numerical schemes contain biases when used in forecasts. The combined ensembles give a better forecast by spreading out the uncertainties in the initial conditions and the model.

Forecasting with an ensemble shows how the system will evolve statistically. It can be seen as a probability distribution of the range of future states. Figure 1.5 shows a schematic depiction of an ensemble precipitation forecast in the UK. Each ensemble member follows a perturbed initial condition path towards the future state. Together the ensembles give an approximate precipitation probability at the forecast time. People that use this probabilistic forecast can decide if they want to risk painting their house on that specific day based on the probability of precipitation.

Ensembles aim to account for three sets of uncertainty: model uncertainty, uncertainty in boundary conditions and uncertainty in initial conditions. The first is an uncertainty that is aimed to be minimized by constant model improvements. The second relates to uncertainties in, for example, sea-surface temperature. The last uncertainty relates to the uncertainty in observations of the atmosphere. With a perfect model assumption, the first two uncertainties are negligible. The perfect model would be able to simulate everything without error leaving only the intrinsic predictability of the system to quantify.

For a perfect model, the spread is a predictability metric: a large spread in a short timeframe indicates very low predictability in the weather, while conversely, a small spread indicates high predictability.

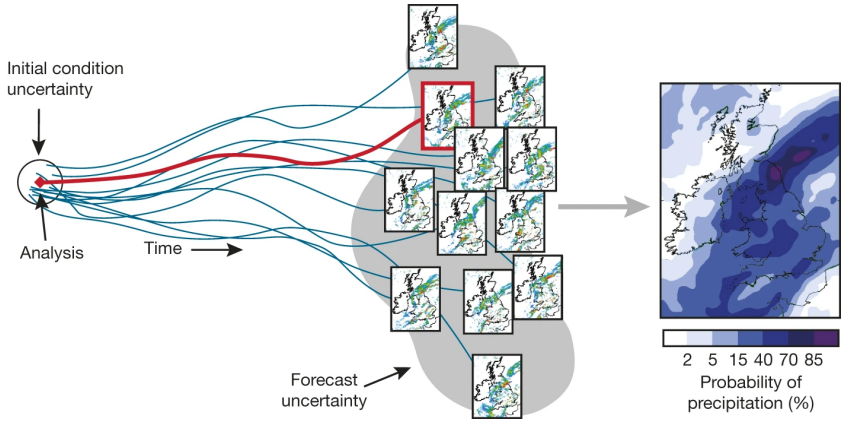


Figure 1.5. Ensemble forecast schematic for precipitation prediction over Europe. From Bauer et al. 2015 [3]

Our research investigates if DL weather forecasting models can be used in ensembles to give a similar probabilistic forecast to NWP. The focus will be on the predictability of a DL forecast. This research is a start-up as the aim is not to analyze the computationally expensive global weather forecasting model but to analyze a chaotic toy model. The Kuramoto-Sivashinsky system is picked for the endeavour because it is thoroughly analyzed in the literature [15–17].

The goal is to verify that the DL forecasting model can reproduce the predictability of a known chaotic system. We will go into extra depth by comparing the long-term chaotic dynamics between our DL forecaster and the ground truth.

In chapter 2, the recent background of DL forecasting will be given. Followed by a background of predictability problems in weather forecasting. In chapter 3, a more rigorous mathematical description of Chaos Theory is given, after which the Kuramoto-Sivashinsky system is introduced. Lastly, a description of the DL architecture used is given. Chapter 4 details the training of the DL algorithm, together with the methods used for our results. Subsequently, the results, conclusion and discussions are given in chapters 5, 6, 7.

2

Background

Our research contains elements of four diverse topics: numerical weather prediction, the Kuramoto-Sivashinsky system, deep learning surrogacy and Chaos theory. On each topic, there is a wealth of knowledge available in the form of papers and books. We aim to provide a surface-level insight into them all. This chapter builds upon the introduction while chapter 3 will give a mathematical insight.

With respect to DL, the topics it includes are such niches that a comprehensive review of previously introduced deep-learning techniques would be impossible. We also do not recommend starting with the papers cited but with the wide variety of online free courses surrounding machine learning and deep learning.

The following sections are read mostly separately.

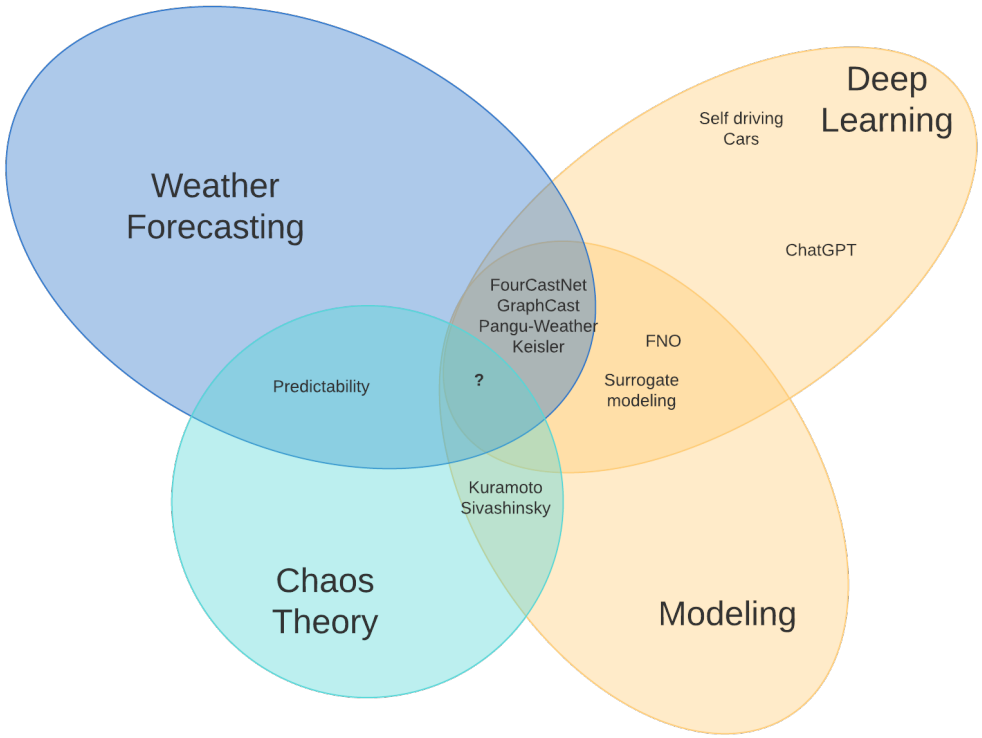


Figure 2.1. Sketching out the overlap of different topics.

2.1. Deep learning in atmospheric sciences

In recent years the amount of Deep Learning (DL) papers has skyrocketed. DL originally a subtopic of Machine Learning (ML) has overtaken the majority of other ML disciplines. Figure 2.2 shows the growth of DL compared to other ML proficiencies [18]. The growth is mainly due to the interest in computer vision and extensive language modelling. Large transformers can generate images out of thin air with Stable Diffusion and Dall-E on the forefront, while as recently as 2023, ChatGPT-4 seems to be a constant table topic due to the perceived knowledge it contains.

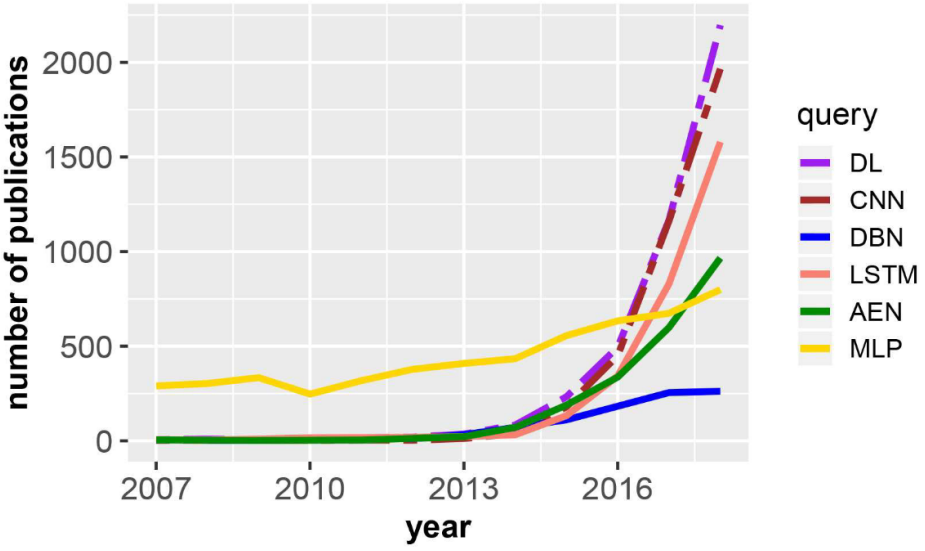


Figure 2.2. Number of publications per year including the term DL; deep learning, CNN; convolutional neural network; DBN, deep belief network; LSTM, long short-term memory; AEN, autoencoder; and MLP, multilayer perceptron. The two dashed lines are scaled down by 5 (deep learning) and 3 (convolutional neural network) factors. The legend shows the search terms used to query the Web of Science publication database. Copied from Emmert-Streib et al. 2020 [18]

DL, or more generally ML, learns to solve complicated problems requiring only data and a framework. The framework comprises an algorithm/network, a loss function and a learning method. The network is a set of parameters linked together that can be moulded into learning a solution to a problem. The loss function penalizes the network when it makes wrong predictions. The loss function is adapted to the dataset and can be modified to include artificial constraints. The learning method is the teacher that modifies the network parameters according to the loss function.

Figure 2.3 shows an intercomparison in methodology between ML and classical models. The DL paradigm is fundamentally different from a classical physical approach. In physics-based model building, a model is built based on observation from the ground up. Mathematics describes individual parts of the observed physical processes on which basis the model is built. After which, the model is used to generalise for different situations. In one, the model is built to fit the dataset, while in the other, a predefined model learns how to approximate the problem defined on the dataset.

DL can often be seen as a statistical approximation. It learns approximate trends from a dataset with which it predicts the data points it has not seen before. The universal approximation theorem states that even the simplest Multi-Layer Perceptron (MLP) could approximate any solution with a large enough network size.

Even though DL is just the tip of the iceberg when it comes to ML, there are many types of DL specialities. The general idea is that all DL networks can be traced back to a group of artificial neurons connected, similar to how brains work is outdated at best. It is still often possible to consider some parts of a DL network as made up of neurons resembling an MLP. However, the connection between the neurons is significantly more complex in recent advanced DL networks than previously imagined.

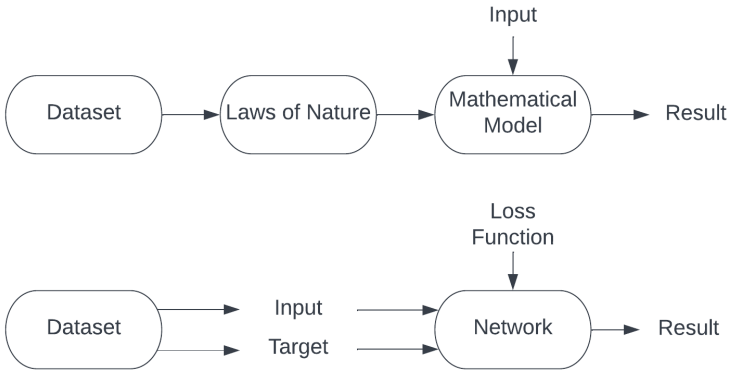


Figure 2.3. Schematic depiction of the difference between ML and physical methodology.

2.1.1. Surrogates

On the back of the exploding diversity in new DL algorithms, a new paradigm for scientific DL is slowly evolving. The scientific world is slowly getting saturated with DL looking around the corner in many disciplines. ML and, to an extent DL, has been used for many classification and regression tasks for some time now. In the atmospheric sciences, it can help scientists classify cloudy satellite imagery or give an estimate of the shortwave radiation at the surface. Until recently more complicated topics, such as time series prediction, have not been tackled to satisfactory levels.

DL surrogates are brand new in the scientific community. Conceived as a way to save on expensive time-consuming numerical simulations are now used for predictive task as well. A surrogate model is a model that mimics another model. The original model is considered the ground truth relating to phenomena. Figure 2.4 shows a surrogate model for non-chaotic Kolmogorov flow [19]. Even though the flow predictions happen on a coarser scale, it still accurately predicts the vorticities without being specified in the system equations.

As an example surrogate, imagine a model for the radiative transfer through an atmospheric column. The physics concerning radiation is known, and a model can numerically calculate the radiation, but it is time-consuming to run the model. Therefore a DL model is trained on a small generated dataset, after which the DL model generalises to situations outside of the dataset. The DL surrogate is faster and generalizes accurately enough for the team to produce more results.

Data-driven surrogacy is a type of surrogate model that trains solely on the data produced by a numerical model, while not applying extra physical constraints. Data-driven surrogacy is often less computationally expensive but means that a surrogate could hallucinate physically incoherent situations.

Surrogates for dynamical systems can also be trained to forecast an approximate trajectory. Forecasting surrogates usually use an autoregressive method to forecast the future. In autoregressive models, previous model states are input to predict the future state(s).

$$\mathbf{X}^{T+N}, \dots, \mathbf{X}^{T+2}, \mathbf{X}^{T+1} = M(\mathbf{X}^{T}, \mathbf{X}^{T-1}, \mathbf{X}^{T-2}, \dots)$$

Often it is simplified to a Markov operator, where the prediction only depends on the current state, and only one state ahead in time is predicted $\mathbf{X}^{T+1} = M(\mathbf{X}^{T})$.

We aim to apply data-driven surrogate modelling to the Kuramoto-Sivashinsky (KS) dynamical system. A system where chaos limits the predictability for a forecasting surrogate. The KS system has been studied thoroughly so that a comparison can be made between the chaotic dynamics of the exact and surrogated model.

Previously, the KS model has been surrogated by a reservoir computer [20], a Fourier Neural Operator and a few other DL methods [21]. Visual and statistical comparisons have been made between the surrogates and exact models, but a comparison between dynamics in the case of FNO surrogacy remains an open question.

In NWP, few groups have taken surrogate modelling to the next stage. Instead of using mathematically defined systems, such as Kolmogorov flow, where creating new data by solving the systems

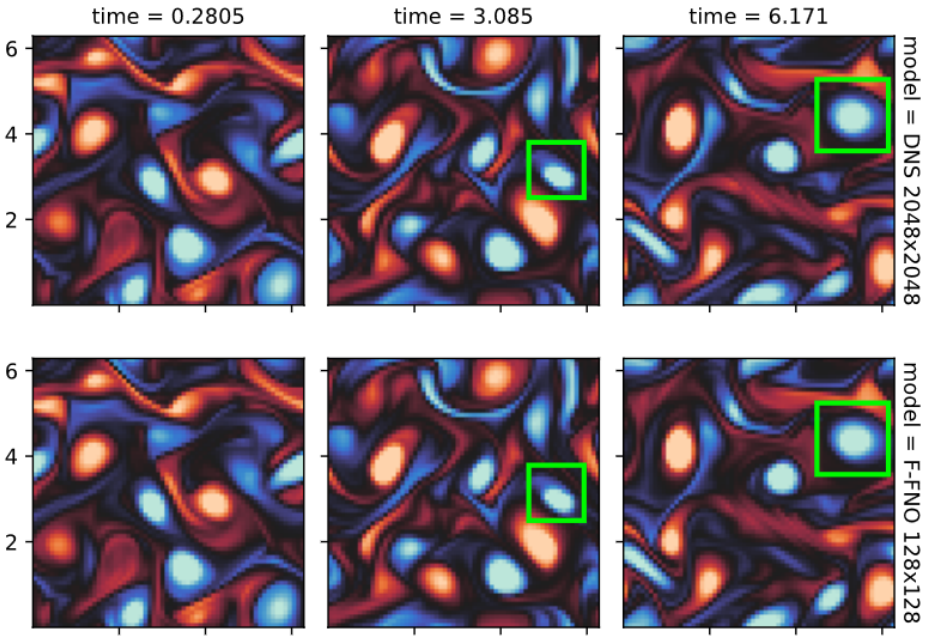


Figure 2.4. Vorticity fields of a Direct Numerical simulation of 2d-Kolmogorov flow on a torus (top) and a factorized FNO surrogate model (bottom). Factorized FNO (F-FNO) factorizes a two-dimensional domain into two separate Fourier domains on which FNO learns. The exact model uses Direct Numerical Simulation on a 2048x2048 grid. The F-FNO DL model infers on a 128x128 grid. Copied from Tran et al. 2021 [19]

numerically is possible. They proposed using the same surrogate modelling methods on a reanalysis dataset. Hereby the data generation step can be bypassed but is also limited by the number of observations. The associated model biases and parameterization problems from a generated dataset can be avoided. In the global NWP sphere, five surrogates are published (table 1.1). These surrogates aim to replace the entire forecasting step in global NWP with a DL network.

The observations used are those from the ECMWF ReAnalysis dataset v5 (ERA5). The schematics of how ERA5 is created are shown in figure 2.5. ERA5 combines vast historical observations into structured global estimates. Satellites, weather stations, weather balloons, aeroplane and marine observations are all combined with the highest resolution ECMWF model HRES to create the dataset that most accurately represents historical weather. ECMWF uses advanced modelling and data assimilation systems to create these estimates.[1] ERA5 raw data is available hourly from 1979 to today on a 0.25° latitude-longitude grid (721×1440 grid points) with 37 vertical levels [22]

The DL surrogates train directly on the ERA5 dataset without extra constraints. In figure 2.6, one of the DL global weather forecasting networks FourCastNet [6] is shown. The initial condition is a test sample from the ERA5 dataset. Both the HRES and FourCastNet forecast the next 96 hours. The near-surface wind speed is plotted with insets zooming into typhoon Manghkut and three Hurricanes (Florence, Issac and Helene). The agreement between the locations of the hurricanes in the ERA5 and the FourCastNet forecast cannot be understated. The forecasting performance of FourCastNet is comparable, if not better than HRES in short-range forecasts up to 48 hours. After 48 hours, the FourCastNet becomes less performant than HRES.

Other global weather forecasting surrogates that are worth mentioning are, Weyn et al. 2021 [5] who was one of the first to show the capability of DL in weather forecasting using a Convolutional Neural Network. Keisler et al. 2022 [7] used a Graph Neural Network first, Pangu-Weather [4] showed better performance than ECMWF in forecasts up to 120 hours ahead with a transformer-based architecture. And most recently GraphCast [8] a Google initiative showed lower RMS error on 10-day forecasts compared to HRES.

[1] Using the HRES NWP model for the reanalysis means that model errors can still seep through to the ERA5 dataset. ERA5 is not perfect, but it is the best type of structured weather observation.

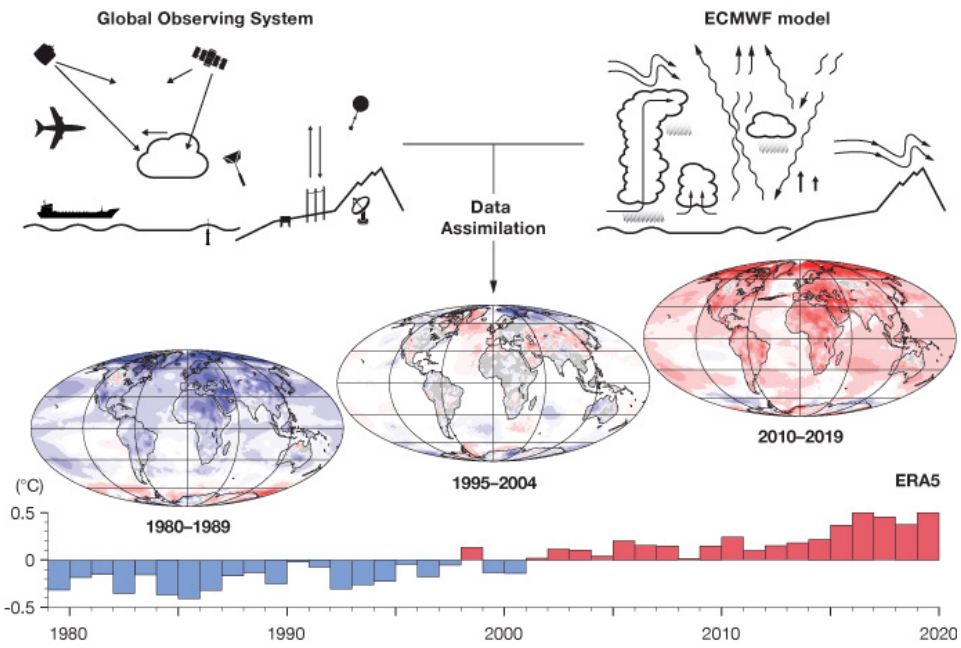


Figure 2.5. Schematic representation of how the ERA5 reanalysis dataset is created. Acquired from ECMWF on 03-2023.

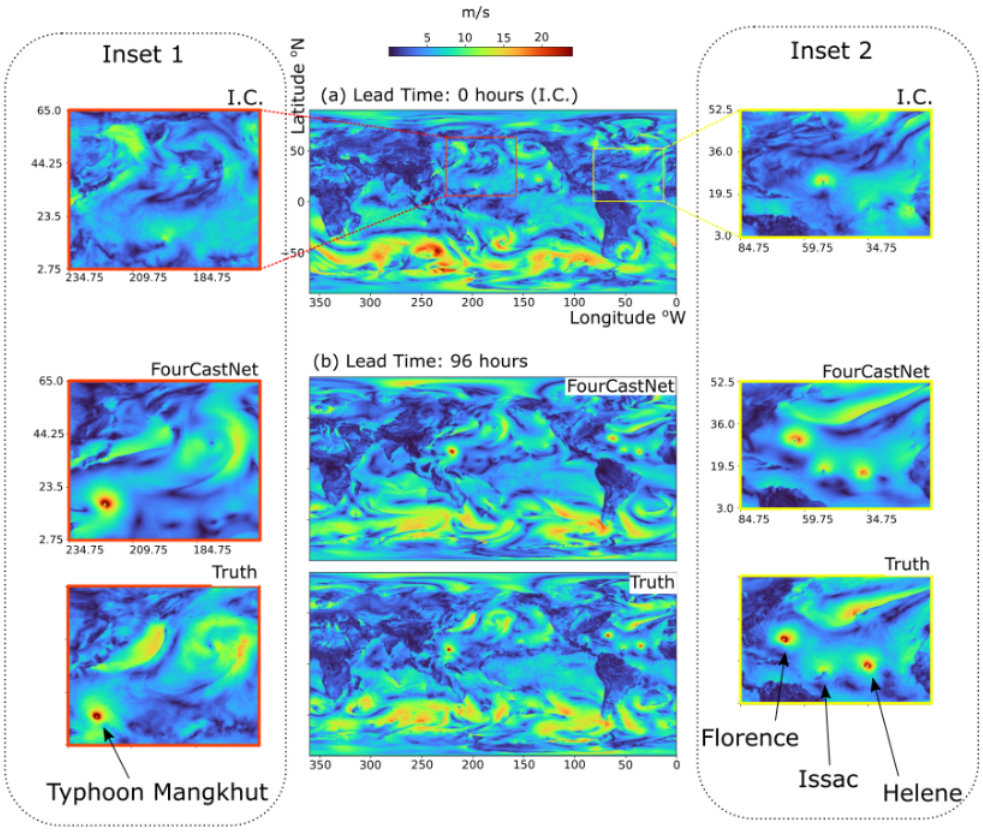


Figure 2.6. A FourCastNet forecast of the global near-surface wind speed over the globe at a resolution of 0.25° on an out-of-sample test initial condition. Copied from Pathak et al. 2022 [6]

2.2. Predictability

To assess a surrogate model on predictability, it is first necessary to specify what predictability means. Predictability was already introduced as a 'sensitivity to initial conditions'. This section will expand the concept towards the rate of error growth in a system.

Normally the predictability of the weather needs to be estimated. This is done by setting up an ensemble. In the ensemble there is a control simulation with an initial condition, the initial weather state. Around the control, a number of ensemble members are set up with a perturbed initial condition. The ensemble is propagated in time and the difference between the control and ensemble member is plotted in a predictability window.

In figure 2.7, the average and the standard deviation of the Root Mean Square (RMS) difference between the control and 107 ensembles are plotted for the NMC DERF experiments [23]. The predictability window has three distinct regions. The first region shows the exponential perturbation growth of the initial perturbation. In the second part, the growth slows down showing linear growth. After which it reaches a plateau which is called the saturation level. When the saturation level is reached, there is no correlation left between the ensemble and the control. If the deviation between the ensemble and control reaches the saturation level, then the ensemble forecast is not worth anything more than the climatological average. The time at which the average error reaches the saturation level is seen as the predictability in NWP. In figure 2.7 the saturation level is at 130 RMS difference and is reached in about 20 days.

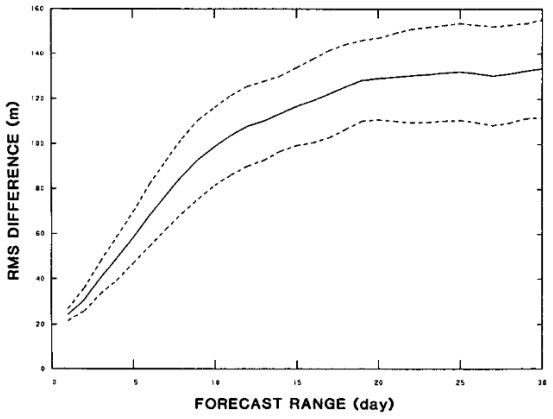


Figure 2.7. RMS error for 107 ensembles in the NMC DERF experiments. Copied from Chen et al. 1989 [23]

Predictability windows in NWP do not only deal with the uncertainty of initial conditions but also uncertainty in model error and boundary conditions. Our research disregards the model uncertainty and uncertainty in boundary conditions. The perfect model assumption is used. The only uncertainty left is the sensitivity to initial conditions inherent to the KS system. The predictability derived from the perfect model assumption is the intrinsic predictability of the model. As no other uncertainty apart from the initial condition uncertainty affects this predictability.

The KS model shows spatiotemporal chaos, meaning chaos in a spatially extended domain. A small perturbation will grow exponentially. In KS system the rate of exponential growth in the respective predictability shows a certain consistency. The rate of exponential growth after a small perturbation can be described by a Lyapunov Exponent (LE). The rate of divergence between control and perturbed is defined as the LE λ_i. A schematic representation of the perturbation growth is shown in figure 2.8. A control flow and a perturbed flow are evolved over time showing a growth of the perturbation.

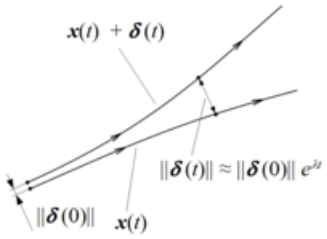


Figure 2.8. Given two initially close flows, the exponential rate at which the flows diverge is given by the Lyapunov Exponent.

LEs give the doubling rate of different orthogonal perturbation vectors for a specific initial condition. The maximum LE describes the maximum growth in perturbation and is dominant in the long term. Due to the fact that a random perturbation often has a component in the direction of maximum perturbation growth, the mLE is the only LE considered in predictability plots, such as figure 2.7. In that figure, the first region is expected to show a constant doubling of error growth associated with the LE. Here it is difficult to see the exponential growth due to the fact the RMS starts at 25 and quickly slows to a linear rate.

When comparing a surrogate with the ground truth, it is expected that the surrogate shows a similar exponential rate and that the saturation level is reached at the same time. Additionally, it is expected that both the exact model and surrogate model reach the same saturation level. The saturation level has nothing to do with predictability, but it is the first sign that the surrogate model approximates the same climate as the ground truth. The saturation level gives the approximate standard deviation of weather in the climate.

Extrapolating the exponential trend to an infinitesimally small error and measuring how many days it would take to reach the saturation level gives the limit of predictability for this model. In the case of the NMC DERF experiments, the limit of predictability was found to be 24 days. So assuming there is a perfect model out there and the exact state of the weather was known around NMC DERF experiments, an infinitesimally small error would make it impossible to forecast the weather more than 24 days in advance [23].

In the case of the KS system, there is no limit of predictability. The KS system is deterministic; without a perturbation in the initial condition, the KS model is completely predictable.

A last property of the error growth can be deduced from the spread in error growths. A large spread in error growth between ensembles represents a low predictability of the error growth. Combining both properties of the intrinsic predictability plot and the spread becomes the predictability of predictability. In other words how well can the ensemble predict the perturbation growth? Does each ensemble follow the same mLE or do they differ?

2.3. Dynamical climate reproduction

The exponential growth rate of small perturbation of the KS system becomes a constant in the long-time average. In other words, the average growth of a infinitesimally small perturbation follows a Characteristic LE (CLE). The CLE is defined the expectation of exponential error growth of the KS system. With the CLE something can be said about the average predictability of the KS system because it directly relates to the time in which the saturation level is reached. The CLEs are constant for the entire 'climate' of the KS system. They will be expanded on in chapter 3, where an introduction to chaos theory is given.

As the climate is the probability distribution of the weather state, climate dynamics is the probability distribution of the weather's time-dependent behaviour. *What is the average predictability of the weather with time?*

Here the analogue between the weather and KS systems has reached its limit. For NWP, it is of limited usefulness to find the average predictability. There exists no perfect model of the weather, and even if it did, the predictability of weather is so dependent on external conditions that there would be limited use of an average with time [24]. Neither can we assume that climate is constant due to all the

external influences, which is a required assumption for calculating the long-term time averages. Nor would it be practical for the weather forecasts made.

There is a reason they are used calculated in the report as they are an in-depth dynamical metric for the performance of the surrogate model. Similar comparisons have been made for simpler chaotic systems such as the Röossler and Lorenz63 system [25] but have only been done once with Kuramoto-Sivashinsky in Pathak et al. 2018 [20] using a reservoir computer DL model.

A more extensive explanation of the dynamical properties of the KS system is given in section 3.1. The idea of LEs will be expanded to local and characteristic LEs, together with the advanced concepts of attractors that are used for comparison in the climate dynamics.

3

Theory

3.1. Nonlinear systems, chaos, attractors and Lyapunov exponents

Our research will compare the dynamics of the one-dimensional Kuramoto-Sivashinsky model with its surrogate. Before something diving into the detailed dynamics, a refresher in mathematical theory is expected. The following sections will introduce concepts used in Chaos theory from the ground up. A Lorenz system will be used as an example to aid mathematics. Afterwards, the concepts will be expanded to KS.

Nonlinearity is at the basis of chaos. In most engineering careers, non-linear systems are discussed sparsely. linear systems are, by definition, easier to predict and, thus, easier to understand. While in general linear systems are the exception to the rule. Polish scientist and mathematician Stanislaw Ulam once remarked:

> "Using a term like nonlinear science is like referring to the bulk of zoology as the study of non-elephant animals."

All chaotic systems are non-linear and out of all non-linear systems almost all are chaotic. Therefore to understand our world better it is important to study the chaos that is spawned by non-linearity.

3.1.1. Sensitivity on initial conditions

Chaotic systems are everywhere around us. The most well-known chaotic system is the weather. As we all know, a weather prediction further than two weeks ahead is useless due to the chaotic nature of the atmosphere. The definition of chaos is "small perturbations in the initial conditions lead to vastly different solutions".

There is a need to approach chaotic systems with caution. Often there are no generalized analytical solutions to chaotic systems, so numerical solutions are the only ones we have. Still, numerical models are often flawed when used on chaotic systems. Numerical errors might cause error propagation and trigger the sensitivity to initial conditions down the line. The solution would then differ from when the discretization resolution is increased.

3.1.2. Lorenz system

The Lorenz system will be used as an illustrative example of chaotic systems. The three-dimensional Ordinary Differential Equation (ODE) is given by

$$\frac{dx}{dt} = \sigma(y - x), \tag{3.1}$$

$$\frac{dy}{dt} = x(\rho - z) - y, \tag{3.2}$$

$$\frac{dz}{dt} = xy - \beta z. \tag{3.3}$$

With parameters $\sigma = 10$, $\beta = \frac{8}{3}$ and $\rho = 28$. The second line in the equations contains the non-linearity causing the chaotic behaviour in the Lorenz system.

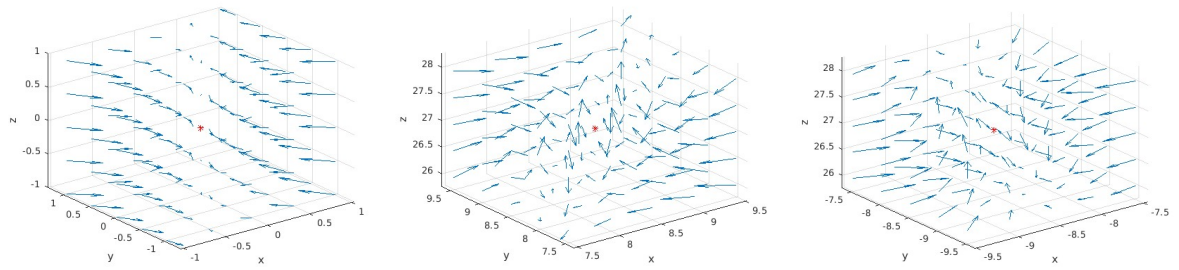
The three variables x, y, z and all the **states** they can take up is what we call the **phase space** of the Lorenz system. $\mathbf{u} = (x, y, z)$ can take up any real values; thus the phase space can be represented by a real three-dimensional space $\mathbb{R}^3$. We are most interested in the structures that form in the phase space and the behaviour of trajectories in the phase space.

Fixed/equilibrium points are points in the phase space where the time derivatives are equal to zero. Theoretically, if the system reached such a state, it would never leave it again. The Lorenz system with previously given parameters has three fixed points, $p_1 := (0, 0, 0)$, $p_2 := (\sqrt{72}, \sqrt{72}, 27)$ and $p_3 := (-\sqrt{72}, -\sqrt{72}, 27)$. These states alone cannot describe the phase space structure, although they give us the first puzzle piece. The second comes from describing the **flow** around the fixed points. The flow is the trajectory a point with specific initial conditions will follow with respect to time. In general, flows are derived by numerically integrating the ODE with time. Do these points attract the flow or repulse the flow? Or maybe something in between?

A **stable** point denotes a point where if the flow is in the neighbourhood of the fixed point, it will stay in the neighbourhood indefinitely. This does not mean that surrounding flows will converge to the stable point; the flows might be periodic around the stable point.

An **attractor** point is where every neighbourhood flow converges to the stable point. An attractor point is always stable, but a stable point does not have to be an attractor point. Attractors have a basin of attraction, meaning that within a domain in the phase space, all flows will end up on this attractor. Analogously you could think of river basins often divided by mountain ridges; if it rains on one side of a mountain ridge, it will flow to a distinct basin. The divisions (ridge) between multiple attractor basins are also structures of interest in chaos theory. Lastly, there are unstable fixed points where the flows are repulsed in *one or more* directions. These are coined unstable since if a flow starting at an unstable point is perturbed only slightly, it will diverge from the fixed point.

In three-dimensional space figure 3.1, the flows around fixed points of the Lorenz system can be visualized by a vector field of the differential around the fixed point. The arrows are indicative of where the flow wants to move to. Due to the three dimensions, it is difficult to conclude from figure 3.1 alone.



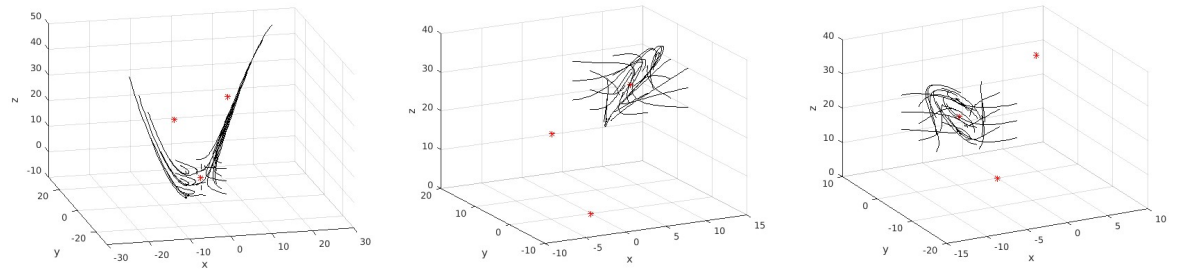
(a) $(0, 0, 0)$ (b) $(\sqrt{72}, \sqrt{72}, 27)$ (c) $(-\sqrt{72}, -\sqrt{72}, 27)$

Figure 3.1. The differential calculated on a uniform grid around the fixed points of the Lorenz system with $\sigma = 10,\ \beta = 8/3,\ \rho = 28$. The arrow length is proportional to the magnitude of the differential.

p_2 and p_3 seem to have some cyclic behaviour around them. Integrating multiple solutions on a uniform grid around the fixed points shows a complete picture, figure 3.2. p_1 is an unstable point, repulsing flows in an approximate plane $y = -x$ while attracting flows in other directions. p_2 and p_3 show cyclic behaviour in a plane around the fixed points. So do all flows converge to periodic solutions around p_2 or p_3? Not quite.

3.1.3. Chaotic attractor

The Lorenz system is dissipative by nature. Due to this, we could have guessed that one of the fixed points must be an attractor. The argument is that if the system loses energy over time, the system must also reach an equilibrium point. The Lorenz system (with these parameters) has a chaotic attractor. As the name suggests, chaotic attractors show high sensitivity to initial conditions. p_2 and p_3 are unstable points but do not repulse the flows like p_1; instead, the flow erratically visits the neighbourhood of both points. In figure 3.3, a single flow is integrated for 100 seconds, showing the Lorenz attractor's two wings.



(a) $(0, 0, 0)$ (b) $(\sqrt{72}, \sqrt{72}, 27)$ (c) $(-\sqrt{72}, -\sqrt{72}, 27)$

Figure 3.2. Multiple flows integrated over 0.2 seconds on a uniform grid around the fixed points of the Lorenz system with $\sigma = 10,\ \beta = 8/3,\ \rho = 28$. **a.** shows an unstable fixed point with the flows being repulsed in the approximate directions towards fixed point b or c. **b.** shows flows attracted to a cyclic plane around the fixed point. **c.** idem.

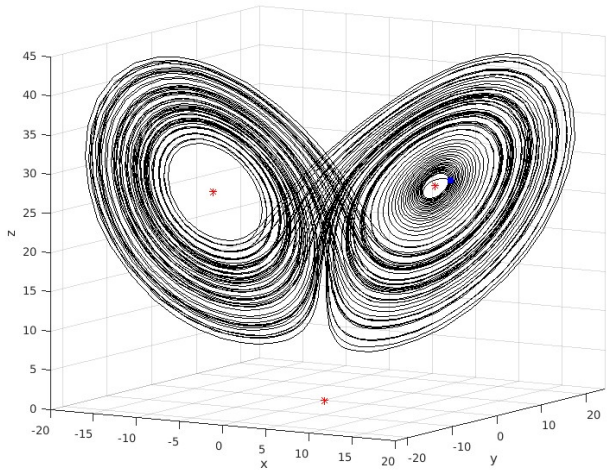


Figure 3.3. Lorenz flow integrated for 100 seconds with i.c. (blue point) close to the fixed point $p_2 = (\sqrt{72}, \sqrt{72}, 27)$ with $\sigma = 10,\ \beta = 8/3,\ \rho = 28$.

The ellipsoidal flows around p_2 and p_3 are aperiodic but will always stay on one of the wings. The wings together are the structure that is called a Chaotic attractor. All neighbourhood flows are attracted to this structure and are chaotic because they behave erratically with high sensitivity on initial conditions. The chaotic attractor is the only attracting structure in the phase space because there are no other fixed points to consider. Thus, the basin of attraction is all points in $\mathbb{R}^3$.

The second part of chaos; *vastly different solutions* can be illustrated best by integrating two infinitesimally close flows with time shown in figure 3.4. Within 10 seconds, both flows show little correlation to each other.

The separation between initially close flows is described by the Lyapunov Exponents (LEs). The LEs represent the expansion/contraction in phase space where there is an LE for each dimension in the phase space. Combined, they are called the LE spectrum. Imagine if we have an infinitesimally small separation vector $\delta\mathbf{u}_0$ in the direction of λ. Eqn 3.4 can approximate the separation between the two flows.

$$||\delta\mathbf{u}(t)||_2 \approx e^{\lambda t}||\delta\mathbf{u}_0||_2 \tag{3.4}$$

The LE changes within the phase space, but it has a structure. Every LE describes the expansion or contraction in a specific direction within the phase space. While the LE magnitude and associated direction are variable in phase space, averaged over the entire attractor, the magnitude of each LE is

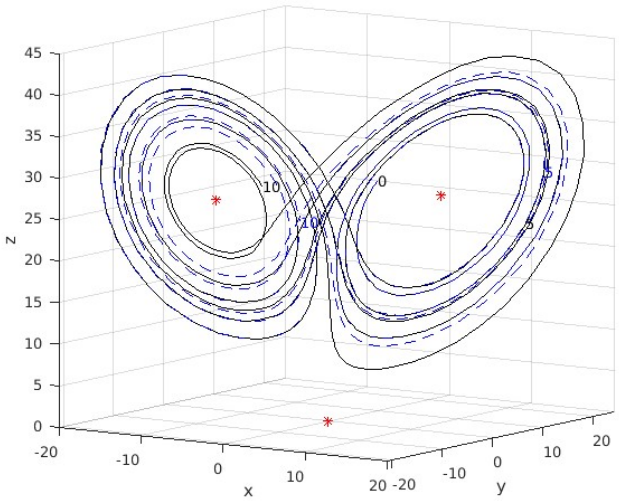


Figure 3.4. Two Lorenz flows integrated for 10 seconds with infinitesimally close i.c. on the Attractor with $\sigma = 10,\ \beta = 8/3,\ \rho = 28$.

constant. The average separation rates are sought after because they are characteristic of the system's dynamics.

The averaged separation rate between two infinitesimally close flows is described by the **Characteristic Lyapunov Exponents** (CLEs). Due to the exponential nature, there can be made a distinction between three types of CLEs:

$\lambda > 0$	Expansion between infinitesimally close trajectories. Positive CLE is an important sign of chaos.
$\lambda = 0$	Invariance, meaning the infinitesimally close trajectories theoretically stay the same distance apart.
$\lambda < 0$	Contraction between infinitesimally close trajectories. Theoretically, the flows should converge to a common flow.

The CLEs for the Lorenz system are given in table 3.1.

Lorenz system CLE	
λ_1	0.906
λ_2	0
λ_3	-14.572

Table 3.1. The characteristic Lyapunov exponents for the Lorenz system with $\sigma = 10,\ \beta = 8/3,\ \rho = 28$.

In practice, the effect of a positive CLE will dominate the flow on the attractor. While the CLEs are global, the direction in which the flow contracts or expands is not and changes locally. Often a tiny part of the separation vector will be in the direction of a positive LE, causing the separation to grow in an unstable direction. Estimating the CLE is a non-trivial task involving an algorithm countering the abovementioned problem. In section 4.3, the CLE spectrum estimation algorithm is introduced.

Another property of the chaotic attractor can be estimated using the CLE spectrum. The **Kaplan-Yorke** conjecture concerns the dimension of an attractor. Here the dimension used is the Hausdorf dimension. The Hausdorf dimension of a point is 0, a line 1, a square surface 2, a cube 3 and so on. The dimension of an attractor does not have to be an integer, but can also be a fractal number. The fractal dimension can appear when the topological shape of the attractor is somewhere between two

fundamental shapes, such as a line and a surface. These shapes can be found often in nature, such as the outline of a snow crystal or a coastline. The fractal dimension of Britain is estimated at 1.21 [26].

The Kaplan-Yorke dimension eqn. 3.5 is an estimate of the Hausdorf dimension based only on the CLE spectrum. The estimate is defined as a fractal number where the sum of CLEs crosses zero. Visually it can be seen as the dimension where expansion and contraction balance each other.

$$D_{\mathrm{KY}} = j + \frac{\sum_j \lambda_i}{|\lambda_j|} \tag{3.5}$$

where for j, $\sum_k \lambda_i \geq 0$ and $\sum_{k+1} \lambda_i < 0$.

The Kaplan-Yorke dimension of the Lorenz system is $D_{\mathrm{KY}} = 2 + 0.906/14.572 = 2.06$.

3.2. Kuramoto-Sivashinsky

The Kuramoto-Sivashinky equation is a famous example of Chaos. Yoshike Kuramoto and Gregory Sivashinksy [27] derived the Partial Differential Equation (PDE) 3.6 in the 1970s to model flame fronts in combustions. The Kuramoto-Sivashinsky (KS) equation models diffusive-thermal laminar flame fronts. Figure 3.5 shows one of such flame fronts in a weakly turbulent flow. The cells of the flame front show chaotic self-motion in the form of rising and dispersing peaks. The variable $u(x,t)$ can be seen as the height of the flame at the front of the combustion with peaks and troughs.

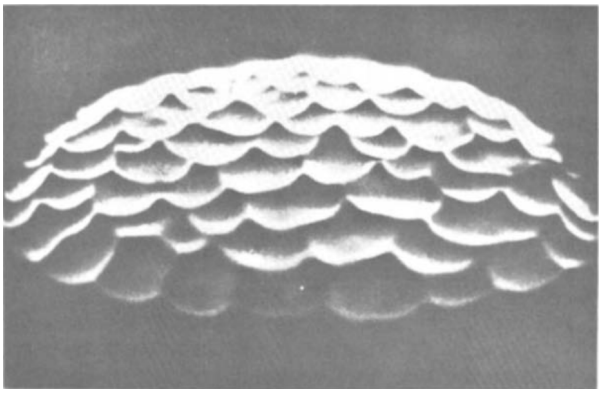


Figure 3.5. Rich propane-air cellular flame showing chaotic motions. Copied from Sivashinsky et al. 1998 [28], originally appeared in Sabathier et al. 1980.

The KS is also used to model plasmas, chemical reaction dynamics and two-phase flows in cylindrical pipes [29]. For these purposes, it has also been extended to two and higher dimensions, but for our purpose, the 1d-KS eqn. 3.6 is studied,

$$\partial_t u + \partial_x^2 u + \partial_x^4 u + u\partial_x u = 0, \tag{3.6}$$

$$u(x + L, t) = u(x, t) \quad \forall \quad 0 \leq x \leq L. \tag{3.7}$$

A periodic domain eqn. 3.7 is normally considered, but different boundary conditions, such as odd-periodic [30], constant $u(0,t) = c$ and Neumann b.c. $\frac{(0,t)}{x} = c$ can be considered. The periodic boundary conditions make it suitable for spectral integration methods and are similar to how many atmospheric variables are solved in NWP. The periodic domain size L is an important parameter which changes the KS dynamics.

KS is well known for its chaotic behaviour and is one of the simplest chaotic PDEs. Due to this fact, it is also well studied in literature [15, 31, 32]. The KS shows a smooth, coherent spatial structure together with temporal chaos. In figure 3.6a Wittenberg et al. 1999 [33] shows the smooth spatial KS structure with its erratic behaviour with time.

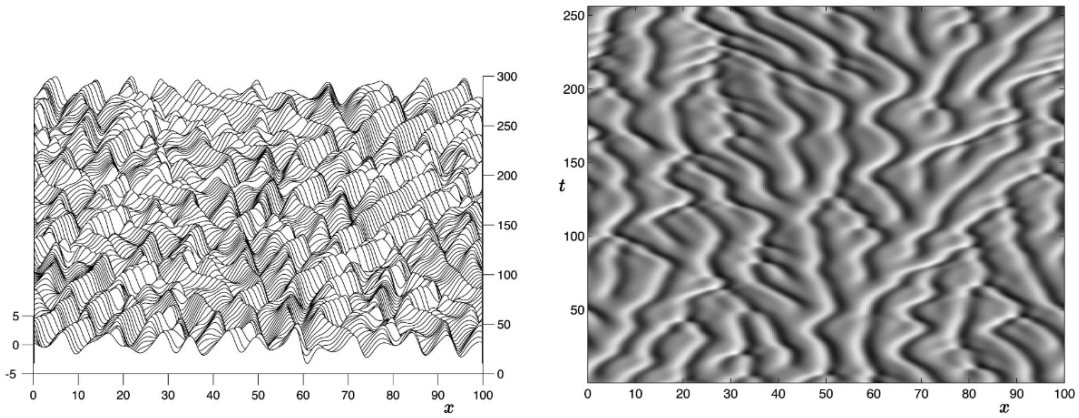


Figure 3.6. A line plot of the periodic Kuramoto-Sivashinsky equation with L=100. Each line is separated by $\Delta t = 1$. Copied from Wittenberg et al. 1998 [33]

KS is well posed, meaning for every initial condition $\mathbf{u}_0$, there is a unique deterministic solution dependent on the initial condition [15]. In other words, the KS can never revisit a state it has already been to.

In figure 3.7 there is an intercomparison between four KS flows with increasing domain sizes. With $L = 12$, the KS shows a travelling wave. Increasing L to 13.5 intermittent bursts interrupt the pattern. A complex periodic system can still be seen in its dynamics. $L = 36$ shows chaotic cellular structures interacting. Lastly, with $L = 100$ the spatiotemporal chaotic pattern is set.

At those large L, the KS generally shows aperiodic behaviour in time. This aperiodic behaviour follows from what is named a period-doubling cascade [16]. A period-doubling cascade is when the solution doubles in a period following a parameter (L) increase. It is as if a delivery driver has to follow a route with 5 stops, and then his route is doubled the next time. He now has to go to 10 stops, and the next day again doubles the number of stops until you can say that it is not a delivery round anymore but a route that he will follow for eternity. KS evolves with the increments in L to an erratic behaviour that signifies chaos.

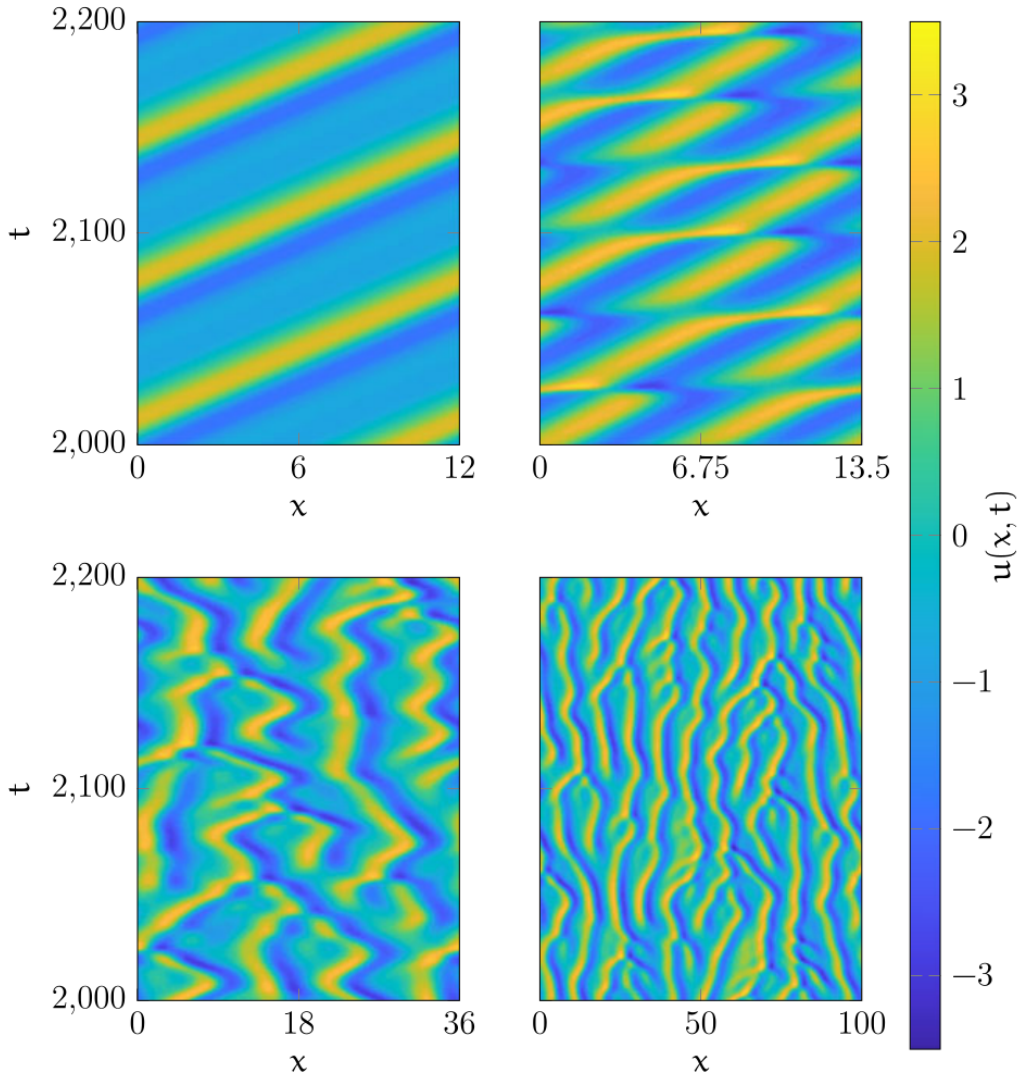


Figure 3.7. KS trajectories with periodic b.c. for increasing domain size L. Copied from Edson et al. 2019 [30]

The KS eqn.3.6 is a fourth-order PDE with a negative diffusion term ($\partial_x^2 u$) and a negative hyper-diffusion term ($\partial_x^4 u$). The convection term ($u\partial_x u$) connects the larger scales to the smaller scales in a non-linearly similar to turbulent fluids. The negative diffusion term destabilises solutions, which shows in spontaneous energy bursts in the smaller spatial scales [16]. During one of these bursts, a new wave is formed. The convection term quickly disperses the bursts to form a smooth sinusoidal wave.

KS contains multiple symmetries, (i) temporal translation invariance $u(x,t) \rightarrow u(x,t+\tau)$ for arbitrary τ, (ii) spatial translation invariance $u(x,t) \rightarrow u(x+l,t)$ for arbitrary l, (iii) parity $u(x,t) \rightarrow -u(-x,t)$ and (iv) Galilean invariance $u(x,t) \rightarrow u(x,x-ct,t)-c$ where c is a constant. These invariances break with the onset of chaos but are still seen in a time-averaged sense [30].

The Power Spectral Density (PSD) of KS is shown in figure 3.8. KS contains a peak power around $q \approx 1/\sqrt{2}$. At lower frequencies $q < 0.3$, the power is relatively constant. At higher frequencies, the power decays exponentially.

The PSD is similar to those of turbulent fluid flows as most energy is included in the lower frequencies, but the high-frequency tail differs from atmospheric fluid flows. In turbulence, there are high energy low-frequency ranges in which turbulence is created by larger-scale effects, after which the energy cascades downwards to the inertial subrange with a range of higher frequencies. The slope in the inertial subrange is considered constant $q^{-5/3}$. In KS, there is a short inertial subrange $1/\sqrt{2} < q < 1.5$. Afterwards, the energy decreases exponentially to zero. The exponential decrease is associated with the hyperdiffusion

term, which quickly dampens the higher frequencies.

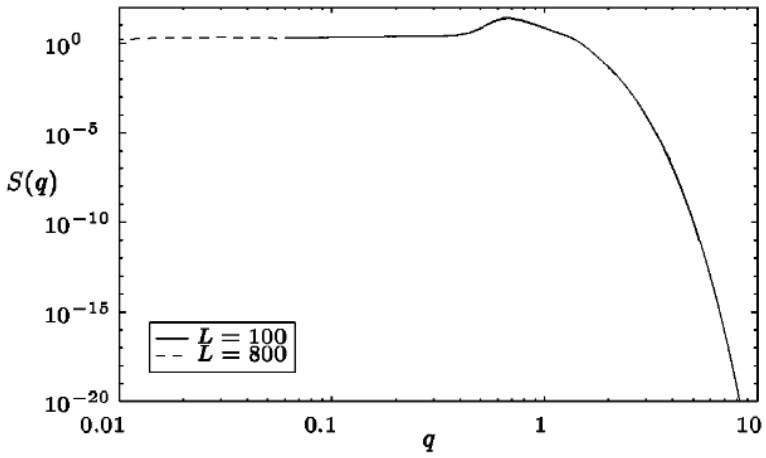


Figure 3.8. Power Spectrum Density $S(q)$ of the Kuramoto-Sivashinsky Equation for two different domain sizes. Copied from Wittenberg et al. 1998 [33]

3.2.1. Chaos for the Kuramoto-Shivashinsky equation

The KS system is fundamentally different from the Lorenz system as the solution lives not only on a temporal continuum but also on a spatial continuum. Therefore, the phase space for KS is infinite-dimensional compared to the three-dimensional phase space for the Lorenz system. The usual way to deal with this is to discretize the system into an M-dimensional spatial system.

To extend the chaotic analysis analogous to the Lorenz system, there need to be a few compromises. It will not be possible to present even an m-dimensional phase space in 2d figures; 2d/3d slices of the phase space will be something we'd have to settle on. Flows will be shown in image maps, such as figure 3.6 & 3.7, where the colour denotes $u(x,t)$.

For the KS system, it is more difficult to identify the fixed points apart from the trivial solution $u(x,t) = 0 \; \forall \; 0 \leq x \leq L$. The usual method for finding fixed points is doing an initial condition grid search to find periodic solutions or attracting points [16]. Unstable fixed points will not be identified in this fashion, although it can be deduced from the flow around zero that the trivial solution $u(x,t) = 0$ is an unstable point.

Although there might be more fixed points, analysing the KS system further is a job for seasoned mathematicians [34]. It is enough to state that the KS system contains a global attractor where all states will evolve to. The chaotic attractor of the KS can be thought of as a low dimensional structure on the entire phase space [31]. Equivalent to how the Lorenz attractor is contained to a subset of the entire phase space. The Lorenz attractor can be visualized in 2d images as the butterfly wings, which is impossible to do for the KS attractor.

The attractor can be seen as the climate of the KS system. Without an external force or change in parameters, the KS flow will never leave this collection of states. One can make the parallel to the weather climate here as well as if the weather lives on this gigantic weather attractor [35, 36]. but the attractor is variable due to changing boundaries, forcings and seasonality.

It is important to note that because the KS attractor is finite-dimensional, the solution for the KS system can be completely and accurately simulated by a discretized numerical method.

The long-term dynamics of the KS attractor can be described with the CLEs. As KS is infinite-dimensional there are also an infinite number of CLEs. In general, we are only interested in the most positive CLE or mCLE. For the comparison study, we will only compute at most 48 CLE. In table 3.2 the computed CLE spectra together with the estimated Kaplan-Yorke dimension for a few different domain sizes are shown as they are presented in the literature [30]. The mCLE increases with L until approximately 0.09 where it flattens out. The CLE spectra contain three approximately zero CLE which come forth out of the three spatial invariances in the KS system. The Kaplan-Yorke dimension follows an approximately linear trend with L, $D_{\mathrm{KY}} \approx 0.226L - 0.160$ [30].

	$L = 12$	$L = 13.5$	$L = 22$	$L = 36$	$L = 60$	$L = 100$
λ_1	0.003	0.059	0.043	0.080	0.089	0.088
λ_2	-0.005	0.004	0.003	0.056	0.067	0.082
λ_3	-0.088	-0.004	0.002	0.014	0.055	0.070
λ_4	-0.089	-0.227	-0.004	0.003	0.041	0.061
λ_5	-0.186	-0.730	-0.008	-0.003	0.030	0.048
λ_6	-3.524	-1.467	-0.185	-0.004	0.005	0.041
λ_7	-3.525	-1.529	-0.253	-0.021	0.003	0.033
λ_8	-9.835	-6.956	-0.296	-0.088	0.000	0.028
λ_9	-9.849	-6.963	-0.309	-0.160	-0.004	0.018
λ_{10}	-9.959	-7.977	-1.965	-0.224	-0.009	0.012
λ_{11}	-10.01	-7.993	-1.967	-0.309	-0.029	0.005
λ_{12}	-10.12	-9.199	-5.599	-0.373	-0.066	0.003
D_{KY}	1.663	3.259	5.198	8.229	13.56	22.44

Table 3.2. Computed characteristic Lyapunov exponents for the periodic Kuramoto-Sivashinsky models. Copied from Edson et al. 2019 [30]

3.3. Fourier Neural Operator

The research focuses on the Fourier Neural Operators (FNO), the backbone of FourCastNet. FNO is a relatively new type of transformer network introduced by Li et al. in 2020 [10]. FNO works by learning resolution-independent convolutions over the entire domain. It is claimed to be the fastest Neural Operator (NO) by leveraging the speed of the Fast Fourier Transform when converting spatial data to the Fourier domain [37]. The advantage of learning an operator in the Fourier domain is that the operator becomes resolution invariant for forecasting. The disadvantages are that FNO can not deal appropriately with discontinuities because the periodic functions it learns are global.

FNO shows excellent promise in modelling continuous fluid dynamics [10, 19]. In figure 3.9 Li et al. 2020 [10] simulated viscous incompressible 2-d Navier-Stokes flow on a unit torus with $\nu = 1e-4$. The FNO was trained on a 64x64x20 ($N_x \times N_y \times N_t$) dataset and set to predict all 30 steps in time in one FNO run.

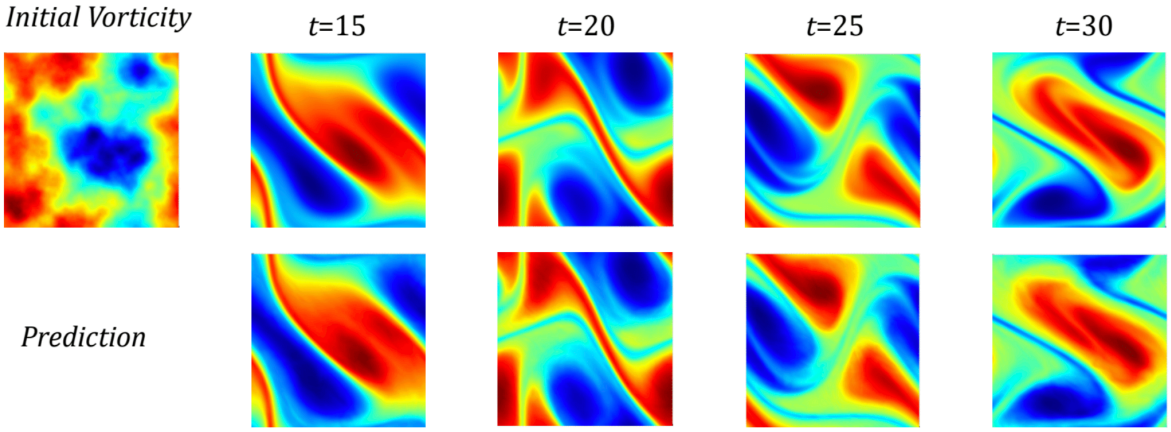


Figure 3.9. 2-d Navier-Stokes equation for a viscous, incompressible fluid. The top row is a CFD simulation, and the bottom is an FNO prediction. Taken from Li et al. 2020 [10]

Firstly the problem description of operator learning will be given. The structure of a generalized NO will be defined. Thirdly, the Fourier variant of NO will follow from a few assumptions made, and lastly, the architecture of the FNO will be expanded on.

3.3.1. Operator learning

Many types of DL networks including Convolutional Neural Networks, Graph Neural Networks and Transformers could be considered learning operators, although it is sparsely mentioned in the computer

science world. For surrogate modelling, it is convenient to name them operators, due to the fact that operators are a fundamental part of the mathematical models that describe physics. FNOs are special in that they aim to find an operator parametrized in the Fourier domain. It assumes the operator can be divided into a collection of sinusoid functions of which the amplitude and phase are taught through deep learning.

Operator learning's premise is finding approximations to mathematical operators in continuous space. Operator learning aims to approximate an infinite-dimensional nonlinear mapping between function spaces. Figure 3.10 visualizes what is meant with the operator over function spaces with the most basic example. With $a(x) = \cos x$ and $u(x) = \sin x$ we want to approximate the continuous operator $G^\dagger$ so that $u(x) = G^\dagger(a(x))$. More generally, the set of all potential input functions describe a function space $\mathcal{A}$ is to be mapped into a specific output function from the output function space $\mathcal{U}$, $G^\dagger : \mathcal{A} \to \mathcal{U}$. With a single input/output function pair, there might be multiple operators which can transform between them. As an example, the spatial derivative operator $\frac{\partial}{\partial x}$ is a candidate that fits the requirements, but a translation operator $u(x) = a(x - \pi/2)$ is a possibility as well. $G^\dagger$ does not have to be a unique operator that maps between the function spaces. The concept is generally applicable to function spaces with all different domains, but in this report, it is sufficient to consider the domain where x lives on a 1-dimensional line $x \in \mathbb{R}$.

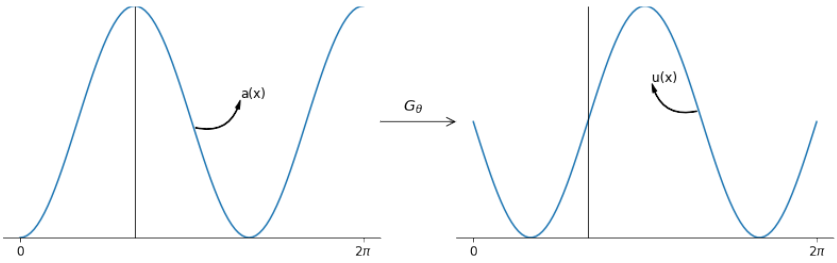


Figure 3.10. An example of a group of operators acting between a coefficient function $a(x) \in \mathcal{A}(D; \mathbb{R})$ and a solution function $u(x) \in \mathcal{U}(D'; \mathbb{R})$ defined on the same domain $D = D' := (-\pi, 2\pi)$. In this example, The mapping operator $G^\dagger$ can be a differentiation or translation operator. Properties of the function spaces $\mathcal{A}$ and $\mathcal{U}$ are essential to narrow down the operator, but there may be no unique operator $G^\dagger : \mathcal{A}(D; \mathbb{R}) \to \mathcal{U}(D; \mathbb{R})$.

Mathematical description: Assume we have a $u \in \mathcal{U}(D; \mathbb{R}^{d_u})$ solution function specified on domain D with the function living in R^{d_u} and $a \in \mathcal{A}(D; \mathbb{R}^{d_a})$ coefficient function defined on the domain D. Both $\mathcal{A}$, $\mathcal{U}$ are infinite-dimensional Banach function spaces. Assume there is/are an/multiple operator(s), which map between the function spaces $G : \mathcal{A} \to \mathcal{U}$. The Neural Operator is an approximate (non-linear) mapping $G^\dagger$ between infinite-dimensional function spaces defined on open bounded domains $D \subset \mathbb{R}^d$ and $D' \subset \mathbb{R}^{d'}$ so that $G^\dagger \approx G$.

From now on we assume $a(x)$ and $u(x)$ are defined on the same domain if not explicitly said otherwise.

3.4. Neural Operator

The general formulation of a NO is given in eqn. 3.8. Firstly, the input is locally transformed by a linear mapping P, $v_1(x) = P(a(x))$ to a higher channel width (as an example a single function $u(x) \to v_1^1(x), ..., v_1^{d_w}(x)$ uplifted to d_w functions). It is assumed that operator G_θ is easier to learn in a higher dimensional latent space. Secondly comes an iterative process $v_0 \mapsto \ldots \mapsto v_Z$ for $Z \in \mathbb{N}$ operator iterations. Every iteration can be split into two parallel operations, an integral kernel operator and a 1d-convolutional operation with kernel width 1. The results are combined and a non-linear activation function is applied. After the iterative layers, v_Z is projected to the output dimension $u(x) = Q(v_Z(x))$

with a linear mapping Q.

$$\begin{aligned} v_1(x) &:= P(a(x)) \\ v_{z+1}(x) &:= \sigma(W_z(v_z(x)) + b_z(x) + (\mathcal{K}_z v_z)(x)) \\ u(x) &:= Q(v_Z(x)), \end{aligned} \tag{3.8}$$

where $1 \leq z \leq Z$, $\sigma : \mathbb{R} \to \mathbb{R}$ is a non-linear local activation function, $W_z(v_z(x)) + b_z(x)$ is a linear transform given by a 1d-convolutional layer with kernel width 1, $W : \mathbb{R}^{d_v} \to \mathbb{R}^{d_v}$ with a bias term b_z.

3.4.1. Integral kernel operator

Operator learning's most important component, the integral kernel operator, is based on Green's function [37]. Green's function describes that for any linear ordinary differential operator (ODE) $\mathcal{L}$, written as $\mathcal{L}u(x,t) = f(x)$, there exists a kernel function $\kappa(x,s)$ so that

$$u(x) = \int_D \kappa(x,s) f(s) ds \tag{3.9}$$

with $u(x)$ being a solution function, $f(x)$ a continuous forcing function and D specifying integration over the entire domain of x. Applying $\mathcal{L}$ on both sides of Eqn. 3.9 would lead to the original ODE. The integral kernel operator Eqn. 3.9 could be seen as the inverse operation of $\mathcal{L}$. In mathematics, the goal often is to find $\kappa(x,s)$ analytically; for a NO, the goal is to fit the kernel function to the training data. The kernel function $\kappa_\phi(x,s)$ becomes a learnable function. Green's function only applies to linear ODEs but not non-linear PDEs such as Kuramoto-Sivashinsky eqn. 3.6.

The integral kernel operator is used within the hidden layers of the algorithm sequentially, with a local non-linear activation function between every integral kernel operator layer. The idea is that combining a linear operator with a non-linear activation function will be able to approximate non-linear operators, similar to how a classical multi-layer perceptron uses linear matrix operations in combination with a non-linear activation function [37].

Leaving out the W_z and b_z in eqn. 3.8 the NO can be written as

$$G_\theta := Q \circ \sigma(\mathcal{K}_Z) \circ \ldots \circ \sigma(\mathcal{K}_1) \circ P,$$

where every $\mathcal{K}_z$ for $1 \leq z \leq Z$ is an integral kernel operator.

$$(\mathcal{K}_z v)(x) = \int_D \kappa_\phi(x,s) v(s) ds$$

The different types of NOs are characterized by how the kernel function $\kappa(x,s)$ is defined or how the integral kernel operator is approximated [37].

3.4.2. Fourier transform

A simplification for the kernel function is to assume that $\kappa_\phi(x,s)$ can be represented by a convolutional kernel function $\kappa_\phi(x-s)$ [10]. This assumes that $v(s)$ is invariant on the domain $x \in D$.

$$(\mathcal{K}_z v)(x) = \int_D \kappa_{\phi,z}(x-s) v(s) ds = (\kappa_{\phi,z} * v)(x) \tag{3.10}$$

Let $\mathcal{F}$ denote the Fourier transform and $\mathcal{F}^{-1}$ the inverse Fourier transform in space. Then Eqn. 3.10 can be written as

$$(\mathcal{K}_z v)(x) = \mathcal{F}^{-1}(\mathcal{F}(\kappa_{\phi,z}) \cdot \mathcal{F}(v))(x)$$

Instead of parametrizing the kernel function in the spatial domain, the parametrization is done in the frequency domain $R_\phi := \mathcal{F}(\kappa_{\phi,z})$.

Definition 1 (Fourier integral kernel)**.** The definition of a Fourier integral operator is given by Eqn. 3.11

$$(\mathcal{K}_z v)(x) = \mathcal{F}^{-1}(R_{\phi,z} \cdot \mathcal{F}(v))(x) \tag{3.11}$$

assuming $\kappa_{\phi,z}$ is a periodic function thus R_ϕ conforms a Fourier series expansion.

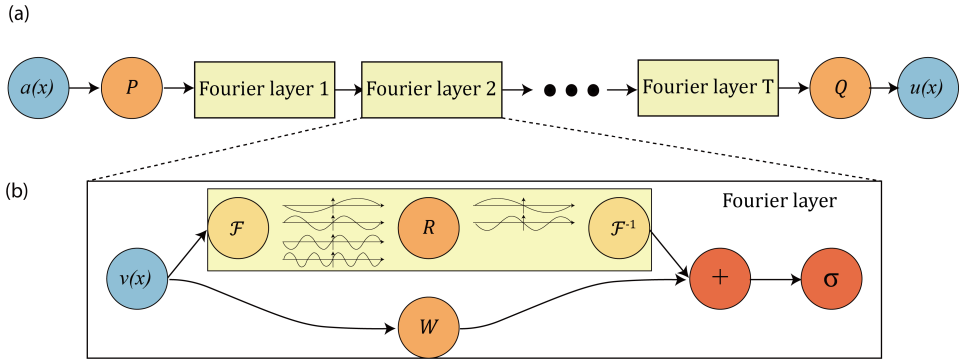


Figure 3.11. Architecture of the Fourier neural operator

3.4.3. Architecture

The FNO architecture consists of 3 distinct parts. It is schematically depicted in figure 3.11.

- **Lifting** operation P in the form of a Fully Connected Neural Network (FCNN) $v_1(x) = P(a(x))$. The dimensionality is increased by increasing the channel width. In general, the input consists of two channels $d_a = 2$, $a : D \to \mathbb{R}^{d_a \times n}$ of which the function and its grid $[u(x), x]^T$. P increases the dimension by increasing the channel width to d_w, $P : \mathbb{R}^{d_a \times n} \to \mathbb{R}^{d_w \times n}$ with $d_w > d_a$ so that $v_1 : D \to \mathbb{R}^{d_w \times n}$.
- **Iterative kernel integration** for Z layers. In parallel, we have:
  - $W_z(v_z(x)) + b_z(x)$ is defined as a 1d Convolutional operation with kernel size 1 together with a bias function, $W_z : \mathbb{R}^{d_w \times n} \to \mathbb{R}^{d_w \times n}$ and $b_z : D \to \mathbb{R}^{d_w \times n}$. The convolutional layer collapses all information across the channel dimension d_w together to a new 1d function. With d_w amount of filters, the size is returned to the original size of $v_z(x)$. It could be seen as a fully connected neural network solely working in the channel dimension.
  - $(\mathcal{K}_{\phi,z} v_z)(x)$ computes a 1d-Fourier transform along the spatial dimension n for all channels, $\mathcal{F}(v_z(x)) \in \mathbb{C}^{d_w \times n/2+1}$ (Only the positive modes k are calculated because of conjugate symmetry). The Fourier output is truncated to $|k| \leq k_{max}$ modes. After which it is multiplied by the complex weights $R_{\phi,z} \in \mathbb{C}^{d_w \times d_w \times k_{max}}$. Lastly, it is transformed back to the spatial domain with an inverse Fourier transform using zero padding, ensuring the output recovers the same discretization size n.

  The results are combined, and a local non-linear activation function $\sigma : \mathbb{R} \to \mathbb{R}$ is applied to the combined output.
- **Projecting** operation Q similarly to P a fully connected neural network projecting the channel width back to appropriate dimension d_u, $u(x) = Q(v_Z(x))$ with $Q : \mathbb{R}^{d_w \times n} \to \mathbb{R}^{d_u \times n}$

The architecture of the FNO is shown in figure 3.11. The FNO architecture contains 3 hyperparameters: the number of layers $\mathbf{Z}$, the channel width/lifting dimension $\mathbf{d_v}$ and the number of modes $\mathbf{k_{max}}$ to truncate in the frequency domain.

4

Methodology

4.1. FNO training

4.1.1. Dataset

The training methodology used here is meant to compare with training a DL model on a single flow. Other training methodologies use multiple flows [38] or even Δt segments initialized with a random u_0 following a distribution function [21]. The latter is preferred for PDE emulation as the operator model will see a larger region of the phase space.

For the dataset, three individual KS flows were simulated using the pseudo-spectral collocation method, appendix B, for training, validation and testing. For Training, validation and testing, 5000, 1000 and 1000 seconds long flows are used respectively. All flows are initialized with an i.i.d. Gaussian random field. The integration parameters are described in table 4.1. The first 500 seconds is considered a spin-up period where the flow follows its transient to the KS attractor. It is not disregarded for training nor testing/validation as the FNO attempts to learn an operator. The operator should not differ between transient and attractor regions. The transitional part of the flow is expected to improve the ability to generalize because the FNO sees a more significant portion of the phase space than when trained solely on an attractor dataset.

Five surrogate models were created for differing $L = 14, 32, 60, 100, 200$. These L were picked to show a progression to more chaotic KS models as the complexity of the attractor increases with L. The KS14 and KS32 models were selected as a control because they show periodic attractors for this L [16]. Increasing the L from 60 to 200 increases the dimension of the chaotic attractor, which is estimated by the D_{ky} Kaplan-Yorke dimension. Edson et al. 2019 [30] found an empirical relationship as $D_{ky} = 0.226L - 0.160$.

Models were trained on an RTX2070 super with double precision. The models used in hyperparameter tuning were trained on single precision due to time constraints.

L	14	32	60	100	200
Nx	32	64	128	254	512
dt	0.0001	0.0001	0.0001	0.0001	0.0001

Table 4.1. Settings used for the pseudo-spectral method in generating the dataset.

4.1.2. Training

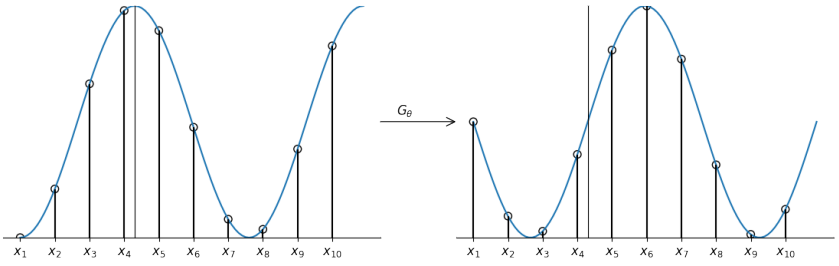
The FNO is trained as a Markov predictor because it is the simplest formulation of an operator. The Markov predictor showed good results during testing.

$$u(x, t + \Delta t) := (G^\dagger u)(x, t) \approx (G_\theta u)(x, t). \tag{4.1}$$

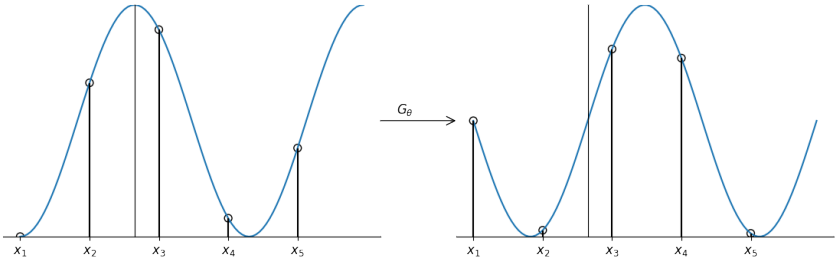
Chaining the predictor $G_\theta^N \circ G_\theta^{N-1} \circ \cdots \circ G_\theta^1$ approximates a KS flow with a Δt timestepping.

FNO is trained as a continuous operator between a function spaces. Considering a discretized function is used, our input is considered as a pointwise subsampled function $a(x_i) = a_i$, where we have

a uniformly spaced subdomain of the original domain $D_M = x_0, \ldots, x_M \subset D$. This seems unimportant mathematical correctness, but it is needed to consider G_θ as a spatial resolution invariant operator thus it should not matter how frequently both functions are sampled. Figure 4.1 illustrates the fact that FNO could be used for different discretization sizes due to it assuming the discretized function is subsampled from a continuous function. It is possible to train the FNO on lower-resolution data but infer on a higher resolution. Data loss might occur if there are features present with a higher characteristic frequency than the Nyquist frequency. We were not limited by computing power, thus the FNO was trained on the full spatial resolution dataset.



(a)



(b)

Figure 4.1. Illustration of how the FNO should work on different discretizations, due to the assumption the discretized points are subsamples of a continuous function. (a) and (b) are discretized with 10 and 5 domain points, respectively.

The normalized L2-loss function is used for training

$$L_\theta = \frac{1}{N_b} \sum^{N_b} \frac{||\mathbf{u}^j - \hat{\mathbf{u}}^j||_2}{||\mathbf{u}^j||_2} \tag{4.2}$$

where $\hat{\mathbf{u}}^j$ is the predicted output and N_b denotes the batchsize.

Deciding on a good Δt is fundamental to an excellent performing surrogate. A too-large timestep and the correlation between the input and output becomes too low. A too-small timestep and the model error would quickly propagate with time. Through testing, it was found that using a small Δt made the FNO unstable, while a large timestep ($\Delta t = 10$) worked fine.

A smaller $\Delta t = 1$ was picked rather due to the limitations involved with the CLE spectrum estimation algorithm. The timestep used for the algorithm is a tuning parameter set at $T = 2$. A larger $\Delta t > 2$ invalidates the CLE spectrum comparison between the pseudo-spectral and FNO surrogate. It should also be mentioned that Li et al. 2021 [21] showed that $\Delta t = 1$ is the optimal value for training with an L2-loss function.

10000 input-output pairs $\mathbf{a}^j$, $\mathbf{u}^j$ were sampled uniformly from the training dataset with $\Delta t = 1$ in between. Overlapping input-output pairs are allowed. The models are trained for 300 epochs using an ADAM optimizer with a variable learning rate and weight decay. The validation dataset is uniformly sampled in 2000 input-output pairs. After every epoch, the validation loss and training loss are logged.

The surrogate models generally show little overfitting after 300 epochs. Thus, the last model state is used for further evaluation.

Model	KS14	KS32	KS60	KS100	KS200
Nx	32	64	128	256	512
Δt			1		
Modes	8	8	16	32	32
Width			50		
Layers			4		

Table 4.2. Model parameters

Hyperparameter	
Epochs	300
Batchsize N_b	1000
Learning Rate	0.001
Stepsize	30
Gamma	0.75
Weight Decay	0.0001

Table 4.3. Hyperparameters

4.2. Intrinsic predictability window

The methodology that is used for calculating the intrinsic predictability window is based on ensembles which are used in weather forecasting.

In the case of weather forecasting ensembles perturbed initial states of a model are used to give an insight into the predictability at that initial state. A large spread in ensembles indicates a low predictability of the weather state and vice versa for a small spread. In the case of low predictability, the confidence that comes with a forecast is lower and can be adapted accordingly. The predictability of an initial state is given by the maximum Local Lyapunov Exponent (mLLE) [39, 40].

A brute-force method is applied to give a visual representation of intrinsic predictability. A KS control flow is simulated with initial conditions $\mathbf{u}_0$ for 600 seconds. 100 different ensembles are set up with a slightly perturbed initial condition $\mathbf{u}_0'$. The perturbation is in the form of point perturbation ϵ_0 on one of the $\mathbf{u}_0$ gridpoints, so that $||\mathbf{e}_0||_2 = ||\mathbf{u}_0 - \mathbf{u}_0'||_2 = 10^{-6}$. The ensembles are simulated for 600 seconds. The euclidean 2-norm $||\mathbf{u}_0 - \mathbf{u}_0'||_2$ is used as a measure for perturbation growth.

4.3. Characteristic Lyapunov spectrum

The characteristic Lyapunov exponents (CLEs) describe the averaged expansion of perturbations in different directions in phase space. In general, CLE cannot be calculated analytically and an algorithm based on Benettin et al. 1980 [41, 42], the differential formulation of the standard method [43], is used for estimation. The algorithm assumes the ergodicity of the KS system and uses the Oseledets multiplicative ergodic theorem [42, 43].

The algorithm 1 can theoretically also be used for the computation of CLEs in the case of a DL surrogate model because the concepts are similarly applicable. Pathak et al. 2017 [44] have used the algorithm on a reservoir computer surrogating KS. The propagation interval T does have a lower limit due to the time step on which the DL model is trained.

Firstly a higher-level interpretation of the algorithm is given along the lines of expanding and contracting spaces using the Lorenz system as an example. Then the more generalized higher dimensional approach is given. Lastly, in-depth details of the algorithm will be discussed and a pseudo algorithm is given.

4.3.1. Lorenz ball

Imagine an initial perturbation sphere around initial conditions in the Lorenz phase space. Perturbation ϵ being arbitrarily small, the maximum CLE describes the averaged magnitude of the largest expansion of the sphere, and the second CLE describes the averaged magnitude of expansion in an orthogonal direction to the first. And the third CLE gives the expansion in the last direction spanning the 3d space.

In 3d this can be visualized by ϵ-ball that non-linearly transforms into an ellipsoid. Figure 4.2 shows the deformation in the Lorenz system phase space. A unity ball is simulated for 1 second. It expands and contracts along the plane of the attractor forming an elongated deformed disc.

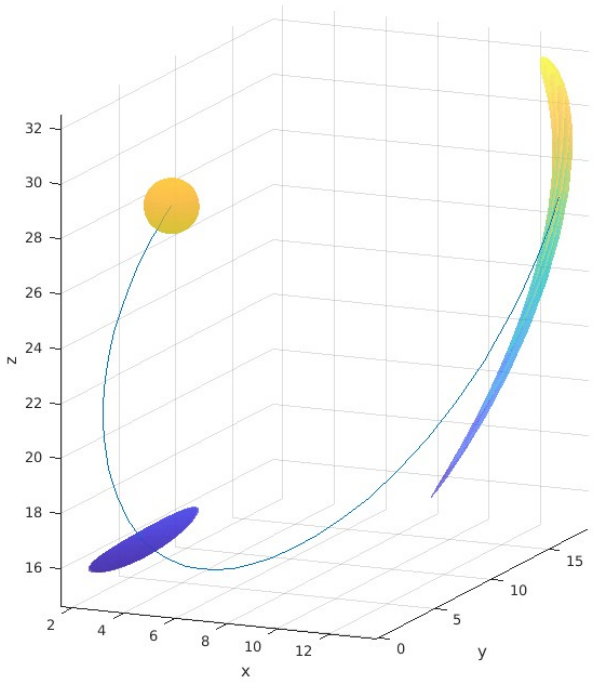


Figure 4.2. The deformation of an ϵ-ball in the Lorenz system.

From the deformed disc, a magnitude of maximum expansion can be estimated. The CLE algorithm follows a single flow and constantly creates balls around the flow. Every time a ball is deformed for T seconds, the deformations are stored, and a new round ball is created. From the stored deformation components, the CLE spectrum is estimated.

4.3.2. CLE estimation for an m-dimensional system

Given an m-dimensional explicit system $\dot{\mathbf{u}}(t) = F(\mathbf{u}(t))$. Consider $\mathbf{u}(t)$ the discretized KS solution. the procedure is as follows. Starting with initial conditions $\mathbf{u}(0)$ on the attractor.

1. Define a hypersphere around $\mathbf{u}(t)$ with radius ϵ.
2. Propagate $\mathbf{u}(t)$ and the ϵ-hypersphere in the phase space around $\mathbf{u}(t) \to \mathbf{u}(t+T)$.
3. Calculate from large to small m orthogonal vectors of deformation $\mathbf{v}_1, \ldots, \mathbf{v}_m$.
4. Store the expansion norms $\frac{||\mathbf{v}_i||_2}{\epsilon}$.
5. $\mathbf{u}(t+T)$ becomes $\mathbf{u}(t)$

Repeat N times.

Using Eqn. 3.4, a perturbation in a LE direction in the phase space grows approximately exponentially with LE λ_i, $||\delta\mathbf{u}(t+T)|| \approx e^{\lambda_i T}\delta\mathbf{u}(t)$. The expansion norm $\frac{||\mathbf{v}_i||_2}{\epsilon}$ is a good approximation to $\frac{||\delta\mathbf{u}(t+T)||_2}{||\delta\mathbf{u}(t)||_2}$ as it describes the magnitude of expansion in the ith largest direction of expansion in the phase space at time t. Thus

$$\frac{||\delta\mathbf{u}(t+T)||_2}{||\delta\mathbf{u}(t)||_2} \approx \frac{||\mathbf{v}_i||_2}{\epsilon} \approx e^{\lambda_i T}. \tag{4.3}$$

Rearranging Eqn. 4.3 and averaging over the N repetitions concludes the following estimator

$$\hat{\lambda}_i = \sum_j^N \log\left(\frac{||\mathbf{v}_i^{(j)}||_2}{\epsilon}\right)/(NT). \tag{4.4}$$

4.3.3. Expansion of the e-hypersphere and reorthonormalization

The ϵ-hypersphere can be defined as a set of orthogonal perturbation vectors. In 3d space, it would thus be made up of three orthogonal vectors scaled so that they are ϵ in length. It is possible to follow each of them to find linear expansion magnitudes between the original perturbation and time-propagated perturbation. This is an efficient method to represent a ϵ-hypersphere in a large dimensional system. There are two problems with this method:

1. From an i.c. $\mathbf{u}_0$ on the attractor it is unknown which perturbation directions are associated with which CLE.
2. In practice, any perturbation vector will eventually be corrupted towards the maximum CLE direction.

Due to the latter, a reorthonormalization procedure is used every T seconds in which secondary perturbation vectors are forced to become orthogonal to the primary larger perturbation vectors. The solution automatically solves the first problem if repeated for many iterations because the primary perturbation vector will eventually expand at a rate depicted by the maximum LE. Thus the primary vector becomes associated with the maximum LE and the second with the second largest LE.

Using orthonormalization steps, the initial orthogonal set of vectors $\{\mathbf{q}_i\}$ can be randomly selected, because after taking a finite number of steps the directions automatically align. Typically a QR decomposition procedure is used as a robust type of Gramm-Schmidt orthonormalization. The entire pseudo-algorithm is shown in alg. 1.

4.3.4. Tuning parameters

Three parameters in the algorithm require tuning: the number of reorthonormalization steps N, the interval for each reothonormalization step T and the perturbation size ϵ. N is discussed in the results.

ϵ is preferred to be as small as possible because the expansion coefficient is more accurate if solutions are considered close to the actual point.

T balances a more accurate large CLE or low CLE. Edson et al. 2019 [30] computed the CLE spectrum for many different values of domain length L in the KS eqn. 3.6. His findings are that for small T, the large CLE ($T < 0.1$ for $L = 10$ and $T < 1$ for $L = 100$) are inaccurate because T is too small to sufficiently capture the flow divergence, leading to an unstable QR decomposition. On the other hand, a large T leads to inaccurate most negative CLE because they propagate too long and get corrupted towards more positive CLE. In the case of a surrogate FNO model, the corruption towards more positive CLE directions happens faster. When the interval T is 5 times larger than the surrogate timestep, the results are corrupted towards the MLE.

Algorithm 1 Lyapunov Spectrum Estimation

$d\mathbf{u}/dt = F(\mathbf{u}, t)$ describes the ordinary differential equation (ODE).
ϵ: The perturbation magnitude.
N: The number of reorthonormalization steps.
T: The time between reorthonormalization steps.
m : The number of most positive Lyapunov Exponents to compute.
$Q^{(0)} \leftarrow \left[\mathbf{q}_1^{(0)} \quad \dots \quad \mathbf{q}_m^{(0)}\right]$ ▷ Initialize perturbation matrix with m orthogonal directions
Require: $\mathbf{u}^{(0)}$ on the Chaotic Attractor ▷ Simulate for $\tau = 2000$ seconds to get rid of transients.
for $j \leftarrow 1$ to N **do**
$\mathbf{u}^{(j-1)} \rightarrow \mathbf{u}^{(j)}$ ▷ Propagate for T seconds using ODE.
for $i \leftarrow 1$ to m **do**
$\mathbf{w}_1^{(j-1)} \leftarrow \mathbf{u}^{(j-1)} + \epsilon \mathbf{q}_i^{(i-1)}$ ▷ Create perturbed flows
$\mathbf{w}_i^{(j)} \leftarrow \mathbf{F}(\mathbf{w}_i^{(i-1)})$ ▷ Propagate for T seconds using ODE.
$\Psi \mathbf{q}_i^{(j)} \leftarrow (\mathbf{w}_i^{(j)} - \mathbf{u}^{(j)})/\epsilon$ ▷ Calculate expansion vectors
end for
$\Psi Q^{(j)} \leftarrow \left[\Psi \mathbf{q}_1^{(j)} \quad \dots \quad \Psi \mathbf{q}_m^{(j)}\right]$
$Q^{(j)} R^{(j)} \leftarrow QR(\Psi Q^{(j)})$ ▷ Gramm-Schmidt Orthogonalization using QR-decomposition.
end for
for $i \leftarrow 1$ to m **do**
$\hat{\lambda}_i = \sum_j^N \ln |R_{i,i}^{(j)}|/(NT)$ ▷ diag(R) estimates the expansion in orthogonal directions.
end for

5

Results

The results are divided into four general sections. Section 5.1 describes the performance of the surrogates KS14, KS32, KS60, KS100 and KS200 compared to the Computational Fluid Dynamics (CFDs). Section 5.2 & 5.3 compare the intrinsic predictability window and characteristic Lyapunov exponent spectrum respectively. Afterwards, section 5.4 is about hyperparameter tuning the FNO. And lastly, section 5.5 checks the zero-shot superresolution capability with respect to predictability.

5.1. Surrogates

In figure 5.2 a to e the performance of the surrogates is compared by a prediction of the last 200 seconds on the test flow. The initial conditions for the surrogates are set equal to the CFD flow at 800 seconds and propagated further to the end at 1000 seconds.

The last 200 seconds are used because they should be closest to the attractor of KS. The flows for KS14 and KS32 both show periodic behaviour. At L=32 the solution flow shows a travelling wave. The FNO shows a similar travelling wave but drifts slowly away from the control. With L=14 there is more complex periodic behaviour. A travelling wave, similar to L=32, switches to a multitude of states. The switching between one state and another has a periodicity itself. The FNO predicts the first switch between the travelling wave states slightly later than the control, but the error thereafter stays relatively low. The accumulation of smaller errors makes KS32 unstable after the first switch. From this point on the KS32 predicts the switching at a higher frequency than the CFD.

KS60, KS100 and KS200 are tasked with predicting more difficult flows. At these lengths, the non-linear nature of the KS system amplifies the errors greatly and the CFD and prediction diverge exponentially. All show error growth that reaches saturation levels around 25-50 seconds in. Figure 5.1 show the root mean square error growth with all of them reaching the saturation level around 50 seconds. The KS60, KS100 and KS200 had a loss of 0.0012, 0.0011 and 0.0056 on the validation dataset respectively. The model error for the KS200 is significantly higher which is seen in the error growth plot as a higher starting point. From $RMS > e^{-3.5}$, the error growths start to follow an exponential slope of 0.09 LE equivalent.

After the saturation level is reached the surrogate flows still look similar to the CFD simulation in a climatological sense.

Figure 29 shows a comparison between difference plots in Lyapunov time. Lyapunov time is the time scaled by the maximum Characteristic Lyapunov Exponent λ_1. The mCLEs are 0.084, 0.088, 0.092 for KS60, KS100 and KS200 respectively. They are computed with the CFD (shown in table 5.1) and checked with Edson et al. 2019 [30]. This plot shows a predictability score relative to how unstable each KS system is. KS200 shows a slightly faster divergence in Lyapunov time from the ground truth compared to KS60 and KS100.

Figure 5.4 shows the spatial PSD averaged over the last 200 seconds. The dots denote the surrogates and the dashed lines/crosses denote the CFD simulation. The PSD match greatly at large energies, but in the expected exponential tail the surrogates settle at a much higher noise level than the CFD simulations. With L=60, L=100 and L=200 the CFD noise level is at 10^{-16} while the noise levels for the surrogates are higher at 10^{-9} for KS60 and KS100 and 10^{-8} for KS200. The higher noise level

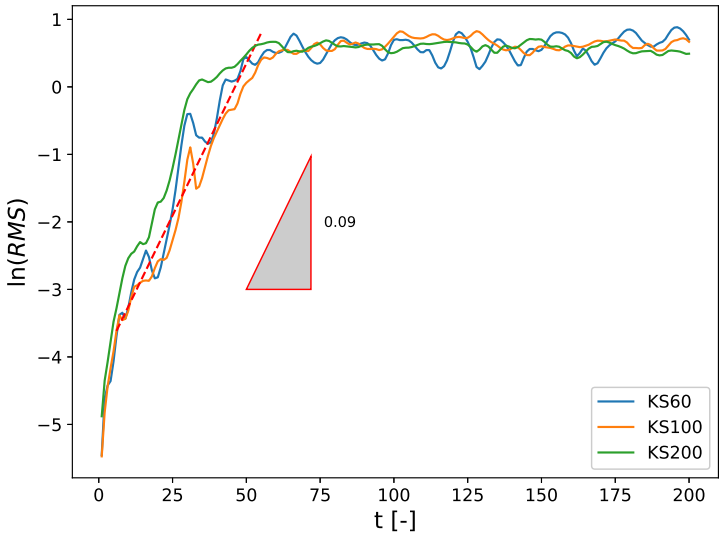
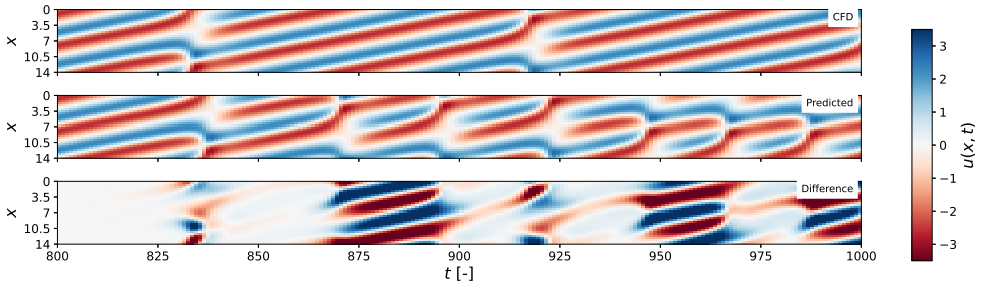


Figure 5.1. Root mean square error between the CFD control simulation and the FNO surrogate predictions.

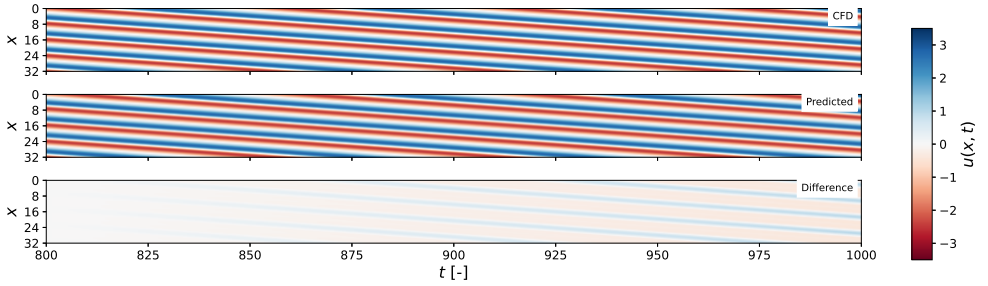
for KS200 concurs with the lower predictability score and a larger validation loss. In the case of KS14, the noise level is similar to KS200. With the unique travelling wave solution for KS32, there are clear peaks at modes where the travelling could be decomposed in a Fourier series. In between the peaks in power, there are noise level modes with the CFD simulation at 10^{-22} and KS32 at 10^{-8}.

The noise seems to set a precedent for the larger model error and thus the faster divergence of individual surrogates. Given the initial seed of noise for the KS200 is higher, the error growth starts at a higher point.

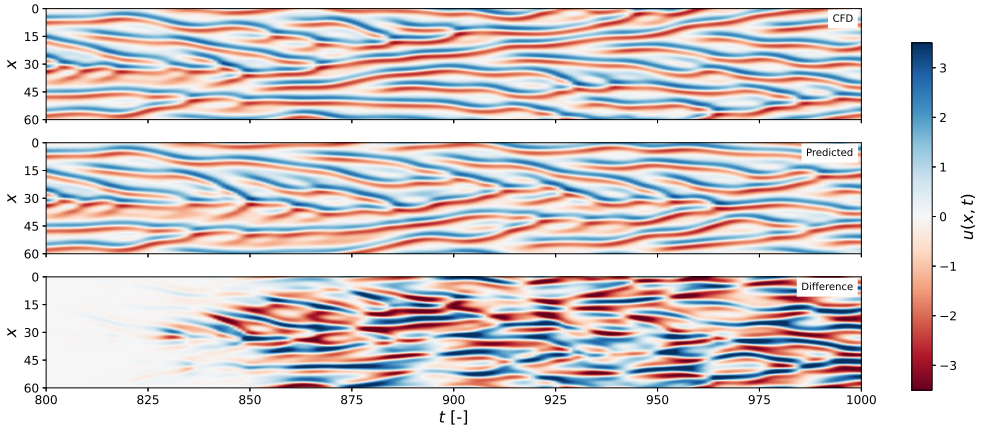
Although decreasing the surrogate model error/noise is interesting by itself. The focus of the results is diverted to the intrinsic predictability and dynamics of the surrogate.



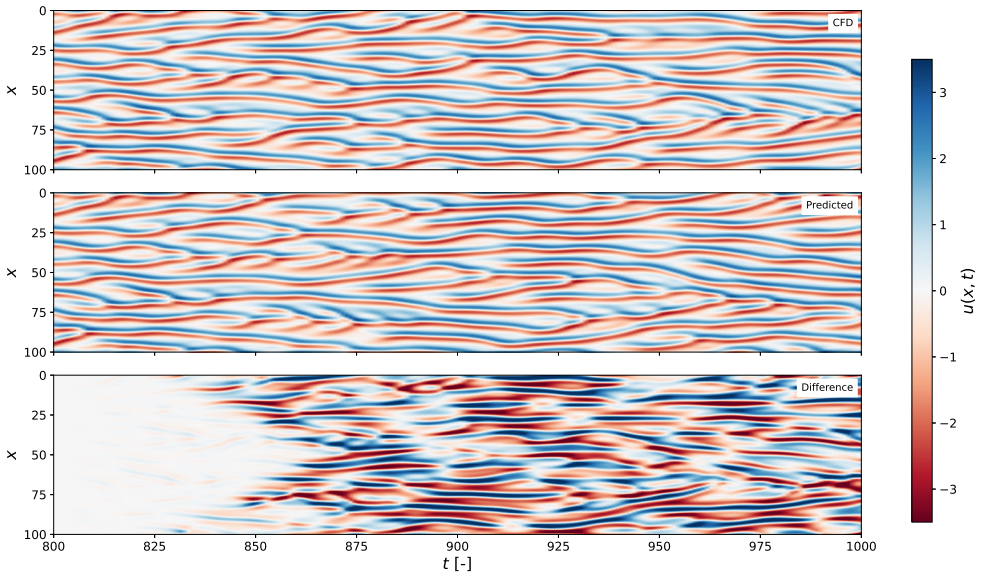
(a) L14



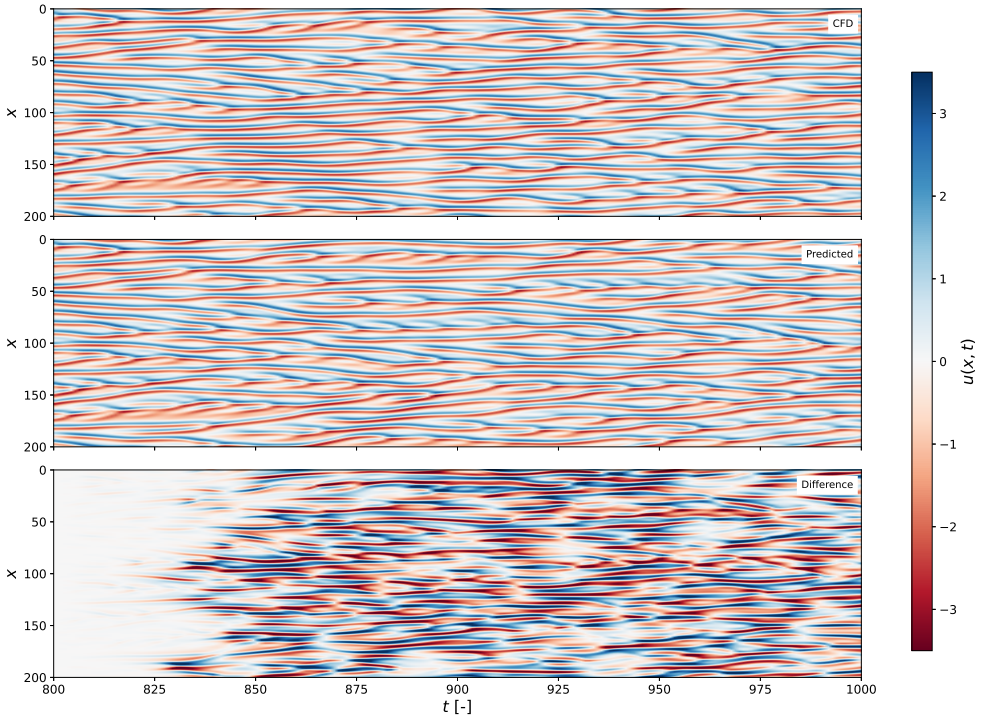
(b) L32



(c) L60



(d) L100



(e) L200

Figure 5.2. Comparison between the simulated 'exact' flow and the predicted flow by the FNO surrogate.

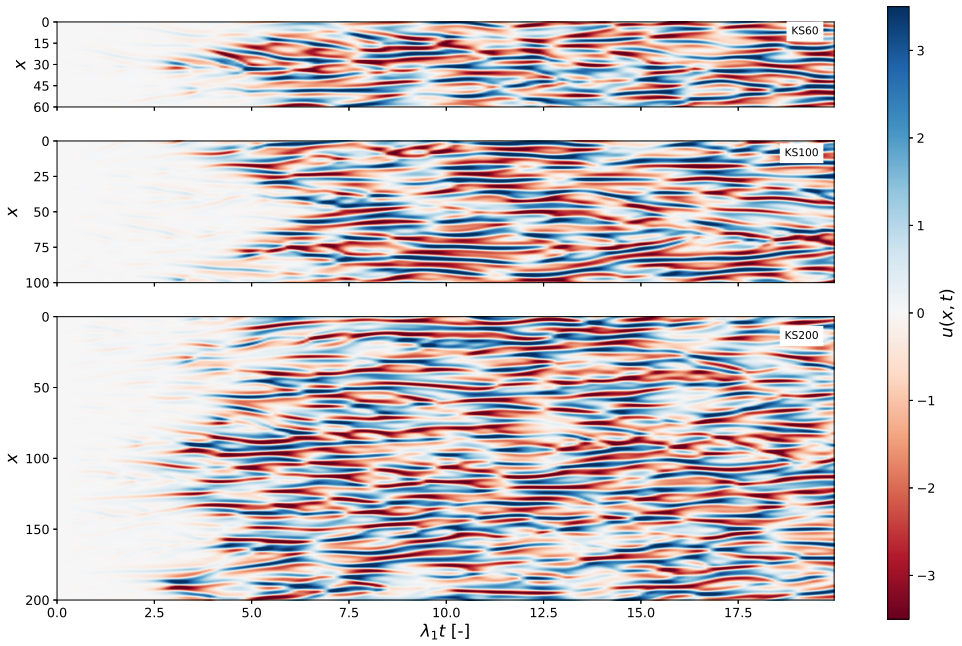
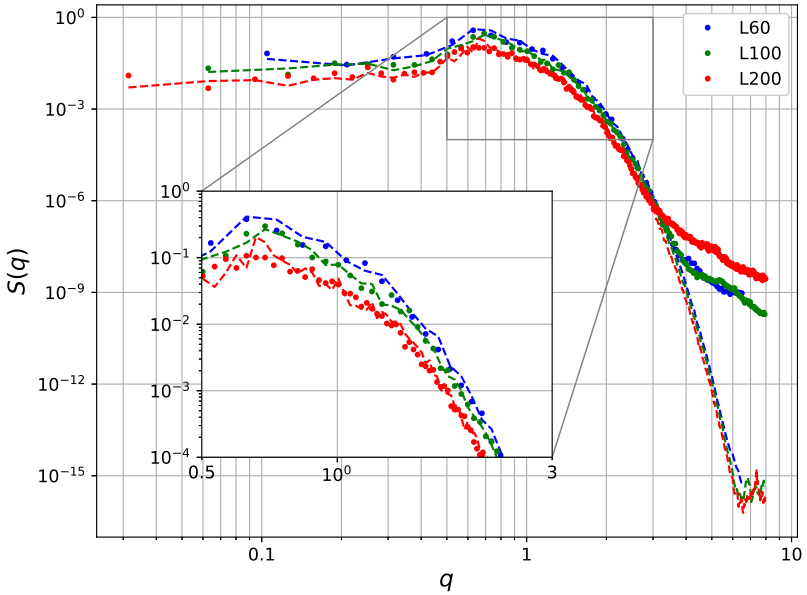
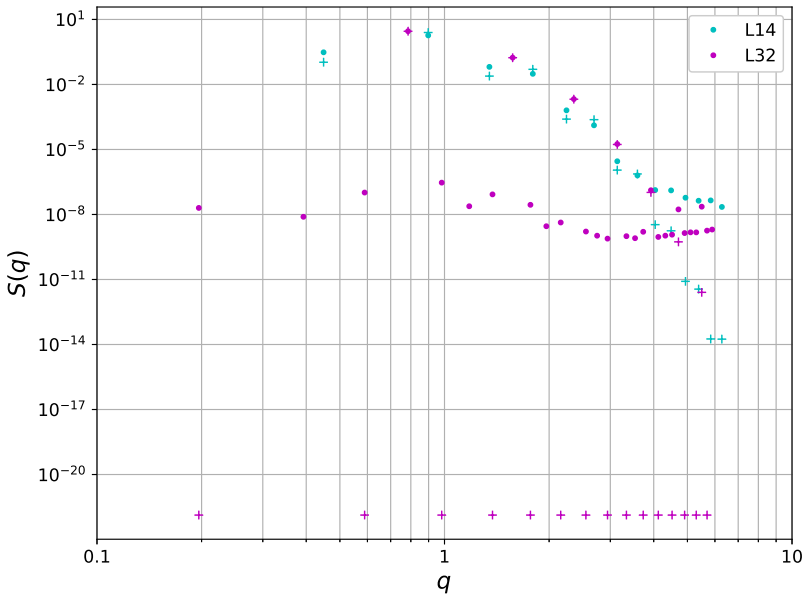


Figure 5.3. Comparing the differences between CFD and predicted in Lyapunov time. CLE $\lambda_1 = 0.084,\ 0.088$, and 0.092 for KS60, KS100 and KS200 respectively.



(a)



(b)

Figure 5.4. Power Spectral Density comparison between the FNO (dots) and the CFD (dashed line, crosses).

5.2. Intrinsic predictability window

The dynamics of a surrogate FNO are not adequately studied as far as we know. Here the first results related to KS dynamics should make us think about whether FNO as a Markov operator cannot only predict accurately but also give a reasonable estimate of the predictability. The intrinsic predictability window is an internal property of the surrogate model that will be compared with that of the CFD. The model error described previously can be disregarded in the following analysis because it is an intramodel analysis.

The reasoning is, given a 300 second flow $\hat{\mathbf{u}}^1, \hat{\mathbf{u}}^2 \ldots, \hat{\mathbf{u}}^{300}$ predicted by the surrogate from the initial condition $\mathbf{u}^0$. We compare it to a slightly perturbed prediction $\hat{\mathbf{u}}'^1, \hat{\mathbf{u}}'^2 \ldots, \hat{\mathbf{u}}'^{300}$ where the initial conditions are perturbed $\mathbf{u}'^0 = \mathbf{u}^0 + \epsilon$. The difference $\hat{\mathbf{u}}^{300} - \hat{\mathbf{u}}'^{300}$ solely dependent on the intrinsic predictability.

The perturbation growth between the two sequences gives the predictability of the model. Computing the same intrinsic predictability window gives a basis to compare the two without taking care of surrogate model errors. Creating an ensemble of 100 perturbed flows grants the necessary statistics to give a comparison. Ensembles are created equally with the same perturbed initial conditions. In figure 5.5 a single ensemble member and control for CFD and KS100 are shown. The intrinsic predictability is significantly increased compared to the intermodel prediction error. Both ensembles reach a saturation level at 130 seconds, compared to 25-50 seconds in figure 5.2. Figure 4.2 shows the perturbation growth of each ensemble vs. control in the form of a Euclidean norm. The perturbation is transformed with the natural logarithm to show the exponential growth. The right plots compare the mean error growth of the ensembles and the sample standard deviation between CFD and FNO. The spread in perturbation growth intramodel is found to be approximately lognormally distributed, which justifies the use of a sample standard deviation after the logarithmic transformation. The KS60 shows good agreement in both spread and mean with the CFD. KS100 shows good agreement of the mean up to 100 seconds, where the CFD becomes slightly more predictable. The saturation level is reached 10 seconds earlier by the FNO. In the case of KS200, the CFD shows a larger spread between the ensembles, while the mean is very similar. Overall the mean intrinsic predictability window is copied well by the surrogates. This implies that the surrogate correctly copies the predictability for these respective initial conditions.

The spread of the predictability window signifies how well the predictability can be predicted. If the low spread in predictability (such as KS200) is taken for granted, we might assume that the predictability is relatively constant no matter how the initial conditions are perturbed. But if, in reality, a spread is high, predictability is less 'predictable'.

In all plots from figure 5.6, the mean error is a straight line in a logarithmic plot. The growth follows from theory, where a perturbation is expected to grow approximately exponentially with mLE λ_1. Repeating the experiment at multiple initial conditions will lead to different mLE. Averaging the mLEs would lead to an estimate of the maximum Characteristic LE, as the CLE is by definition the average exponential expansion rate of perturbation over the whole attractor. In figure 5.7, the intrinsic predictability windows are plotted with the slope if it is assumed that the error grows with the mCLE. Here we see that the maximum Local LE already approaches the maximum CLE with some minor variations. We can do better than repeating the experiment above multiple times to find the CLE. The following section will approach the surrogate from a dynamical climate perspective. Computing the whole CLE spectrum with a computationally efficient method.

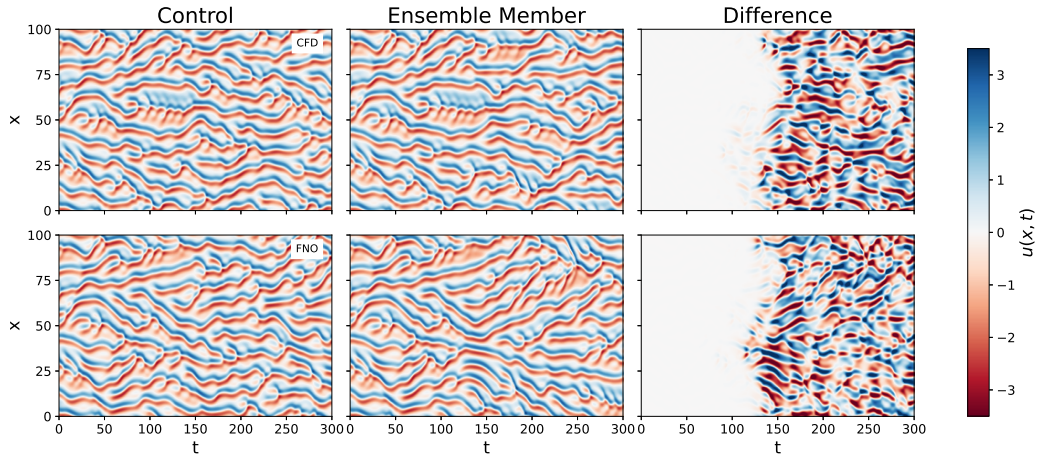
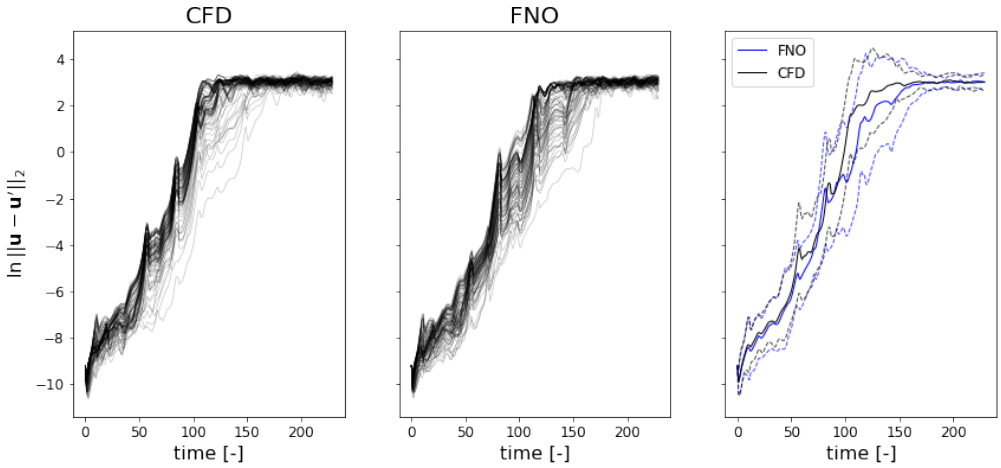
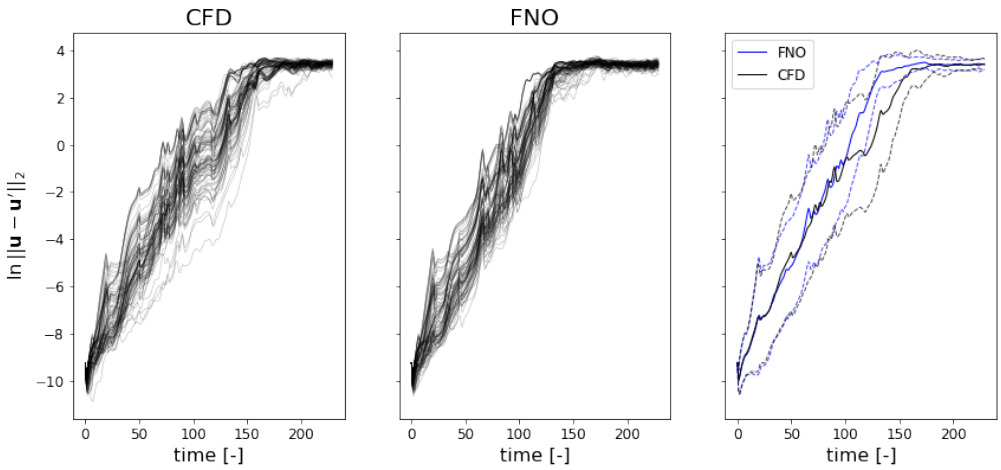


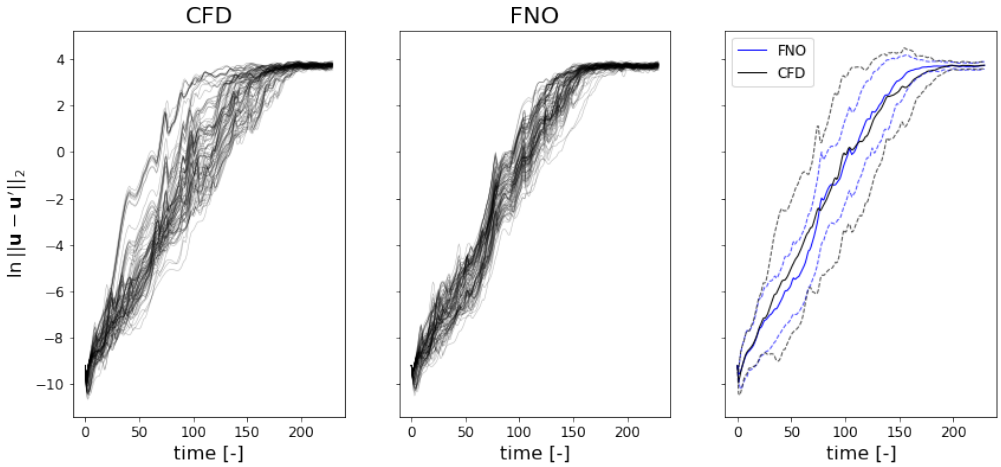
Figure 5.5. Comparison between a single ensemble member and the control flow for the CFD and FNO.



(a) L60



(b) L100



(c) L200

Figure 5.6. Intrinsic predictability comparison between the numerical and FNO models when faced with a perturbation in the initial conditions. 100 ensembles are computed against a control flow starting on the attractor. Each ensemble adds a different $\epsilon_0 = 10^{-4}$ point perturbation to the control initial conditions. The error growth compared to the control is computed with the Euclidean 2-norm. In the third plot, the sample means and sample standard deviations are plotted against each other.

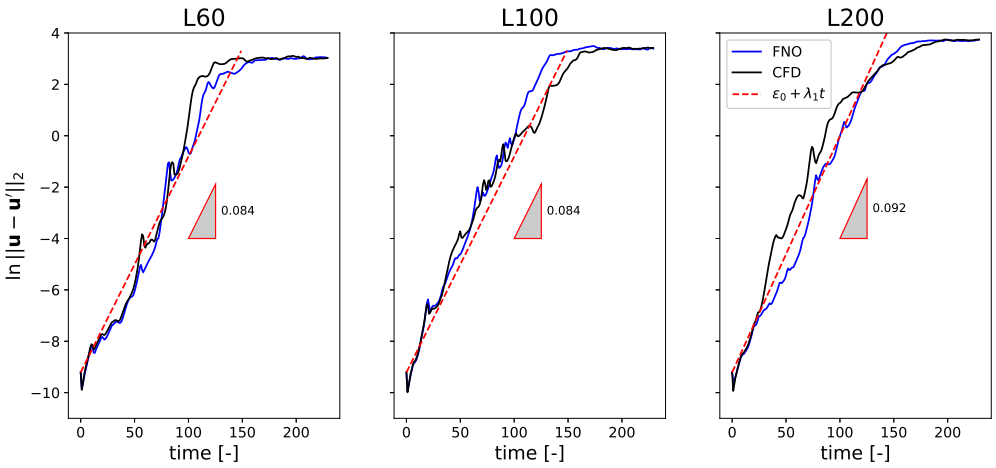


Figure 5.7. Average intrinsic predictability over 100 ensembles compared to the slope which the maximum characteristic Lyapunov exponent predicts.

5.2.1. Perturbation size and numerical precision

The initial perturbation size must be picked carefully. The following plot for L=200 in figure 5.8 is created the same as fig 5.6 but shows a rapid error growth at the start and a very small spread for the FNO. The predictability is now underestimated compared to the CFD. The perturbation size is $\epsilon_0 = 10^{-6}$ similar to before.

The difference is that 32-bit floating point numbers are used for the FNO network. PyTorch uses 32-bit floats by default for all its operations and models to save memory, while the CFD ran in 64-bit doubles. The perturbations in the FNO results were too small and caused numerical errors at the start of the ensemble. Increasing the perturbation size to 10^{-4} or forcing PyTorch to use 64-bit doubles solved the problem.

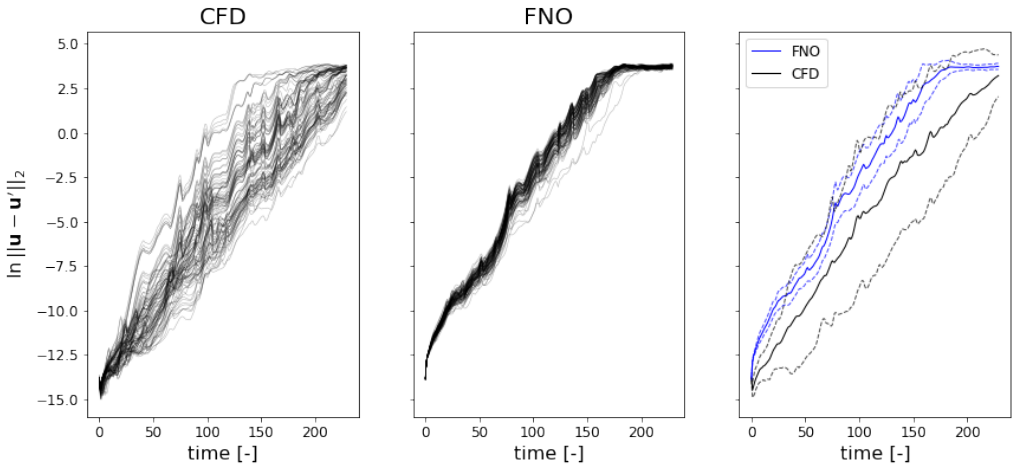


Figure 5.8. Intrinsic predictability comparison between the numerical and FNO models when faced with a small 10^{-6} perturbation in the initial conditions, while the FNO uses 32-bit precision.

5.3. Characteristic Lyapunov spectrum

The CLE spectrum is one of the dynamic constants within a model. It is expected that a surrogate should have similar dynamical properties and thus also the same CLE spectrum. The most important is the maximum CLE denoted by λ_1, which has been frequently used in previous results.

Algorithm 1 converges with $N \to \infty$, although a cut-off must be balanced between precision and compute time. Edson et al. 2019 [30] reported $N = 1000$ was generally precise enough; We found that $N = 2000$ gives better convergence. In figure 5.9, the positive CLEs are plotted against the number of orthonormalization steps N used in the CLE computation. It shows sufficient convergence for the mCLE, but the smaller the CLE, the longer it takes.

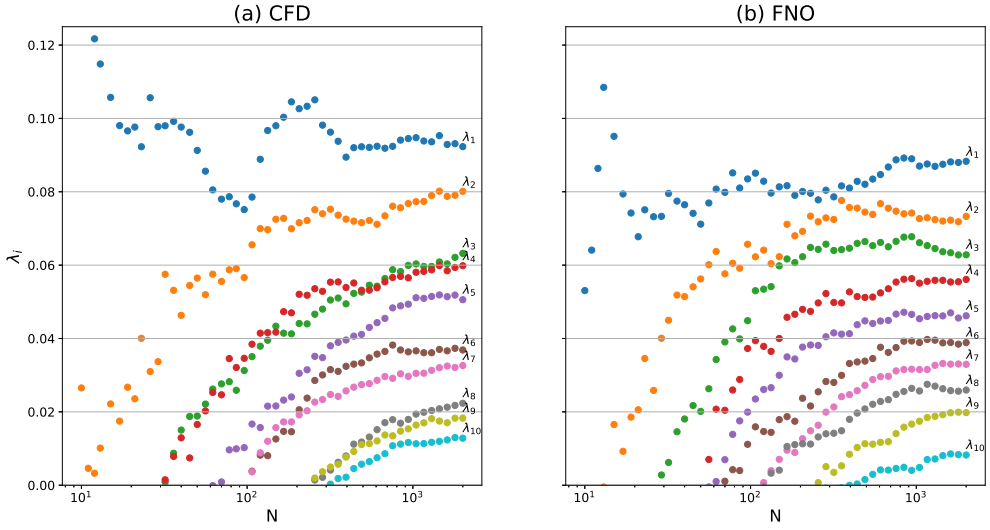
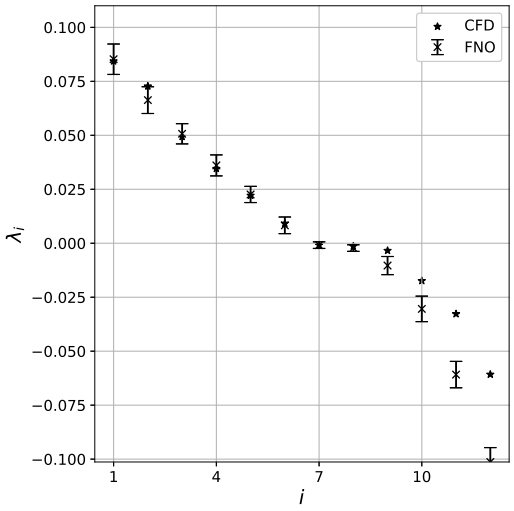


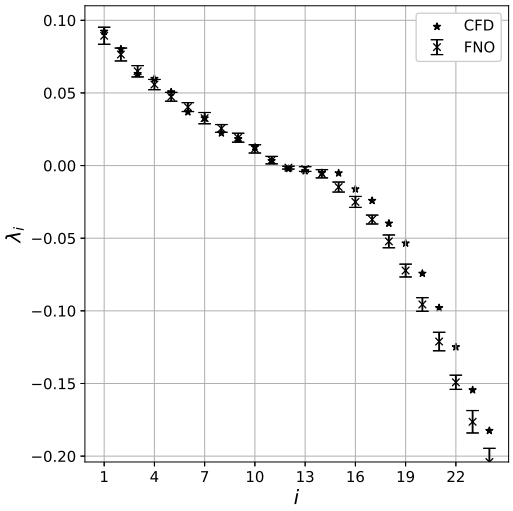
Figure 5.9. Estimates of the CLE spectrum for L100 for increasing size of N.

In the case of the FNO, the CLE spectrum computation is on the order of $O(10^4)$ times faster (single CPU vs GPU). Therefore, it is repeated 25 times to give a sample mean and standard deviation assuming the estimated CLE follows a normal distribution. The results are shown in figure 5.10 and table 5.1. The CLE spectra for KS60 and KS100 show a good match apart from a missing zero CLE in both. Shifting negative CLEs to the left once for the CFD and the entire spectra would line up. The CLE spectrum for KS200 shows a positive bias for the positive CLE. The CFD mCLE falls outside the lowest bound of the 95% confidence interval. Thus, it seems the KS200 generally has slightly lower intrinsic predictability than the CFD. For KS200, a zero CLE is missing as well (not shown). A zero CLE is missing for all models indicating FNO finds replicating one of the spatial invariances challenging. Pathak et al. 2017 [44] obtained the same result.

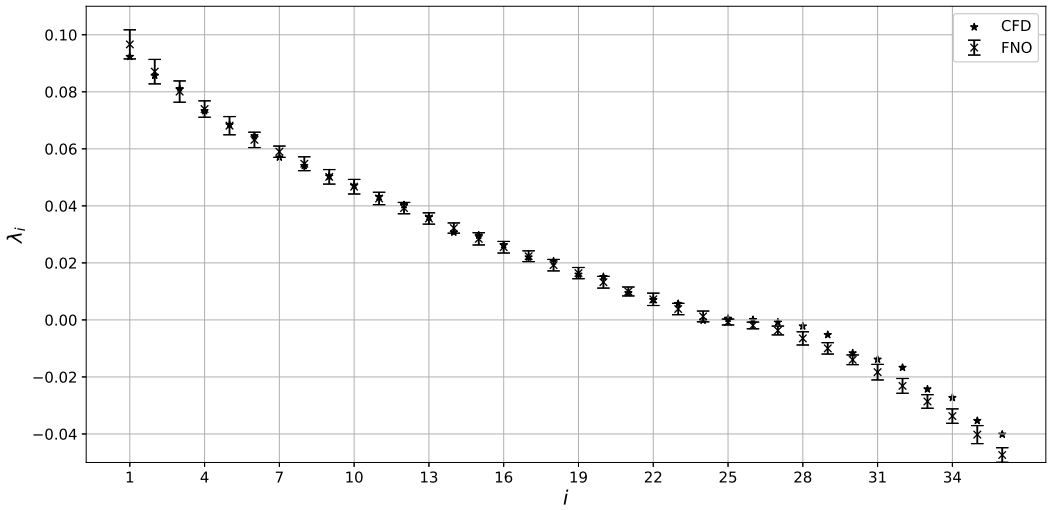
The missing zero CLE is seen again in the Kaplan-Yorke dimension; for all surrogates, the Kaplan-Yorke dimension is smaller by approximately one.



(a) L60



(b) L100



(c) L200

Figure 5.10. CLE spectrum comparison between FNO surrogate and truth using algorithm 1[30, 42, 43]. The surrogate FNO show the sample mean ($N_{sample} = 25$), and the error bars denote the 95% confidence interval assuming normality.

L	60			100			200	
i	Edson	CFD	KS60	Edson	CFD	KS100	CFD	KS200
1	0.089	0.084	0.085±0.0035	0.088	0.092	0.089±0.0029	0.092	0.097±0.0026
2	0.067	0.073	0.066±0.0031	0.082	0.080	0.076±0.0022	0.085	0.087±0.0021
3	0.055	0.049	0.051±0.0023	0.070	0.063	0.065±0.0019	0.079	0.080±0.0019
4	0.041	0.035	0.036±0.0024	0.061	0.060	0.056±0.0018	0.071	0.074±0.0014
5	0.030	0.022	0.023±0.0019	0.048	0.051	0.047±0.0015	0.066	0.068±0.0016
6	0.005	0.009	0.008±0.0019	0.041	0.037	0.040±0.0015	0.061	0.063±0.0013
7	0.003	-0.001	-0.001±0.0008	0.033	0.033	0.033±0.0019	0.054	0.059±0.0010
8	0.000	-0.001	-0.002±0.0008	0.028	0.022	0.026±0.0013	0.051	0.055±0.0012
9	-0.004	-0.003	-0.010±0.0021	0.018	0.018	0.019±0.0015	0.046	0.050±0.0013
10	-0.009	-0.017	-0.030±0.0030	0.012	0.013	0.011±0.0014	0.043	0.047±0.0013
D_{KY}	13.56	13.6	12.42±0.081	22.44	21.9	21.3±0.101	44.7	44.1±0.137

Table 5.1. CLE spectrum comparison of the 10 largest CLE and the Kaplan-Yorke dimension D_{KY} between the surrogate, CFD ground truth and Edson et al. 2019 [30] literature reference using algorithm 1 [30, 42, 43]. The surrogate shows a sample mean ($N_{sample} = 25$) with the sample standard deviation assuming normality.

5.4. Hyperparameter tuning

During the testing of the FNO surrogate models, a large amount of different hyperparameter configurations were checked. The loss on the validation dataset measures the performance of a model. In general, most hyperparameters from table 4.3 were not found to impact performance significantly. However, weight decay deserves special mention, as it prevents the model from overfitting.

From table 4.2, the N_x and Δt were considered fixed. The number of modes, width and layers were hyperparameter tuned for KS100. The number of modes was the most important for model performance. It represents the truncation of the intermediate solution in the Fourier domain, where #modes are the integer of the highest mode that is kept. Figure 5.11 shows the performance increase with an increasing number of modes. The colours denote the width used in the FNO model. With a low number of modes, the FNO can compensate for the performance using a larger model width, but at a larger number of modes, the performance is almost equal between different model widths.

The effect of a larger number of modes is shown well in figure 5.12. Here the time-averaged PSD is shown for models with a different number of modes. The vertical lines are the number of modes represented in the FNO. The PSD is expected to exponentially reduce quickly after the peak at $q \approx 1/\sqrt{2}$. The lowest #modes model discontinues the exponential decay after $q \approx 2$ where noise is seen to set in at $S(2) \approx 10^{-4}$. Large #modes models follow the exponential decay until $q \approx 3$, with a lower noise level at 10^{-8}. The #modes seem to suppress the onset of a noisy high-frequency range because models with a large truncation show a jump in noise level after the last represented mode.

The model performance has a positive impact on dynamical climate reproduction. Figure 5.13 plots the validation loss against the estimated Kaplan-Yorke dimension D_{KY}. Low performance correlates with an overestimation of the D_{KY}. The CFD reference value is $D_{\mathrm{KY}} = 20.9$ *or* 21.44, excluding the missing zero CLE. High-performing models consistently estimate a $D_{\mathrm{KY}} \approx 21.5$ while low-performing models overestimate the D_{KY}. The KS200 model seems to overestimate the D_{KY} similar to the lower performing models here.

In terms of surrogating the KS system, the hyperparameter optimization shows that FNO works best in the case where the highest frequency components can be accurately represented in the Fourier domain. The highest frequency components for KS lie within the inertial part of the PSD. In figures 5.12, 5.4 & 3.8 the cross-over after the inertial domain is roughly at $q \approx 1.5$. The increase of 16 to 32 #modes denotes the cross-over between a KS surrogate that does not represent all modes within the inertial domain to a surrogate that does. The performance in terms of validation loss shows diminishing

returns at the same moment. Although figure 5.13 shows that the dynamics are copied sufficiently for 16 #modes.

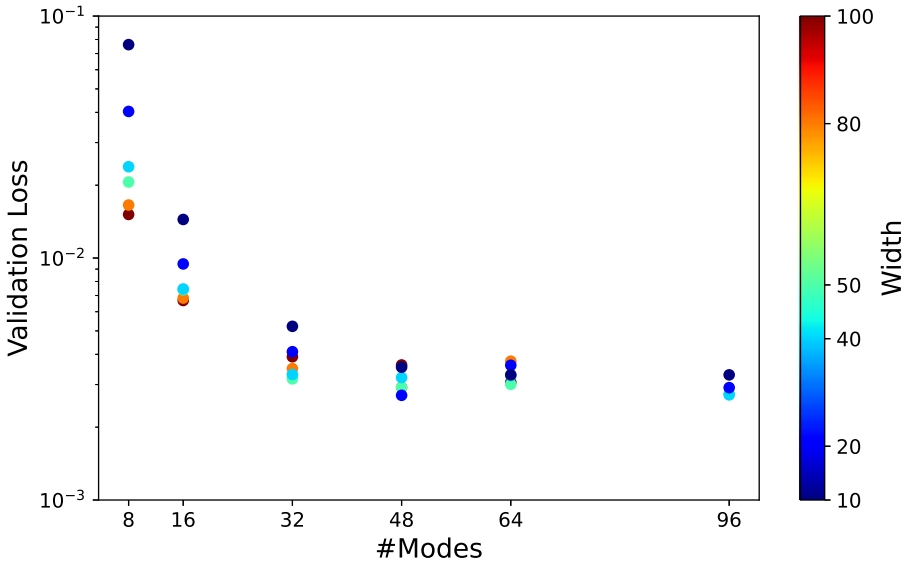


Figure 5.11. Hyper parameter tuning the number of modes represented in the FNO against the validation loss. The colour denotes the width used for the model.

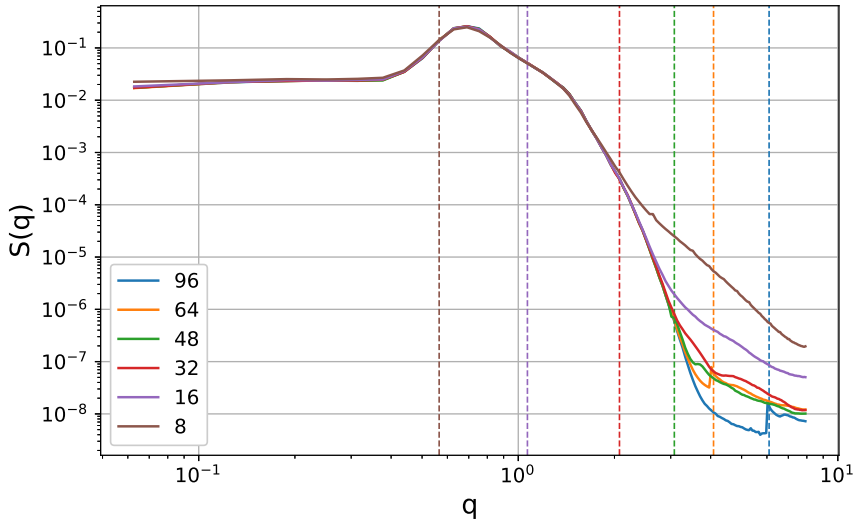


Figure 5.12. PSD for the different number of modes used. Vertical lines show the mode truncation used in the FNO.

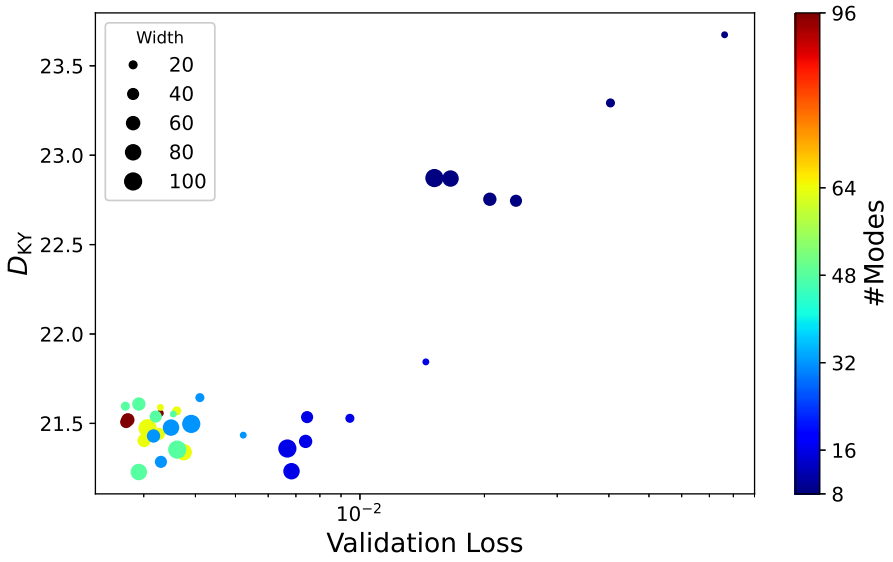


Figure 5.13. The Kaplan-Yorke dimension as a performance metric for the CLE spectrum plotted against the validation loss of the model. The ground-truth Kaplan-Yorke dimension is $D_{\mathrm{KY}} = 21.9$. Although the FNO is expected to have one less dimension due to a missing zero CLE. The size of the points denotes the FNO width. The colour denotes the number of FNO modes.

5.4.1. Lower amount of training data

The amount of training data is crucial to consider how well FNO can generalize to the entire phase space. The training dataset was truncated at the end to create shorter flows on which the FNO can train. At very short durations, the training dataset mainly comprises a transient flow towards the attractor. The validation loss against the training dataset length is shown in figure 5.14. Even though the FNO trains on short KS flows, it can still optimize an operator with high accuracy on the validation dataset. It should be noted that the model trained on 100 seconds became unstable when autoregressively predicting a longer KS flow. Thus, a flow of 200 seconds is necessary to train a working KS surrogate.

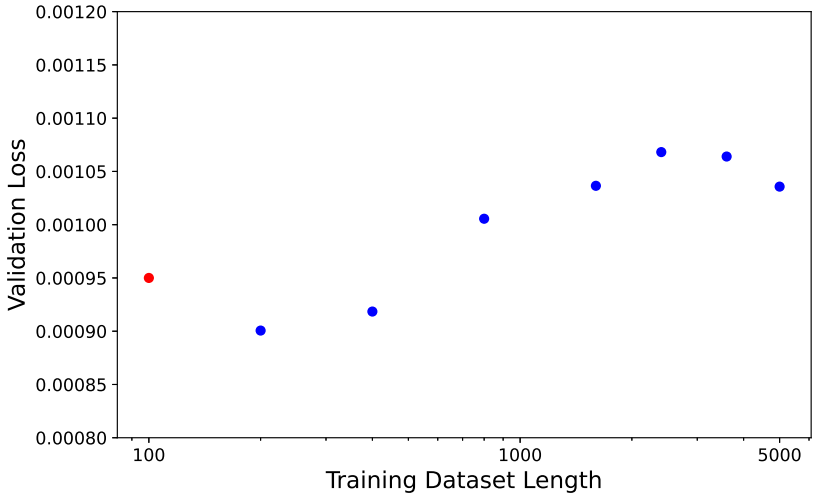


Figure 5.14. Validation loss found for different Training Dataset lengths. Using only 100 seconds for training made the model unstable. Every training dataset is divided into 10^4 input-output pairs. For low dataset lengths, these pairs comprise mostly overlapping pairs.

5.5. Invariance to discretization

To illustrate the property of invariance to discretization, the KS100 model was used for a zero-shot superresolution prediction. An initial condition was upscaled to 384, 512 and 1024 gridpoints, after which all were used to predict a flow in their respective spatial resolution. Figure 5.15 compares the different resolution flows. Figure 5.16 shows the divergence between the higher and original resolution flow. The error growth approaches the mCLE rate at $\lambda_1 \approx 0.09$ equivalent.

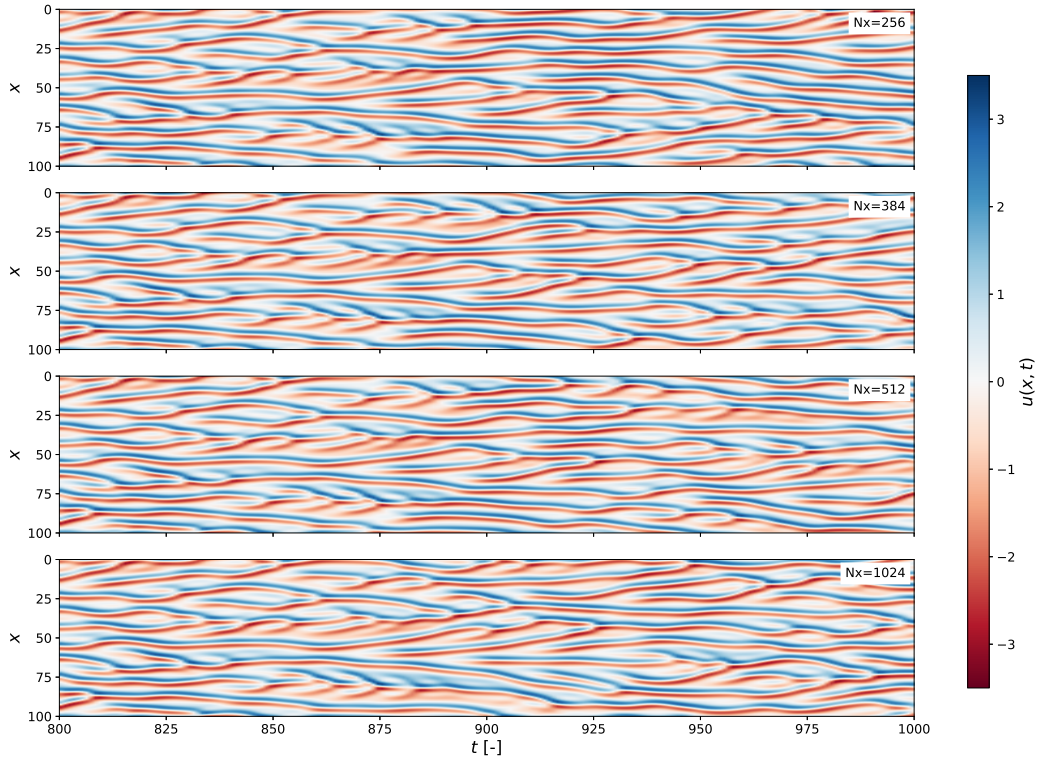


Figure 5.15. KSL100 is used to predict 3 flows with higher spatial resolutions at $N_x = 384,\ 512,\ 1024$ starting from the same initial conditions.

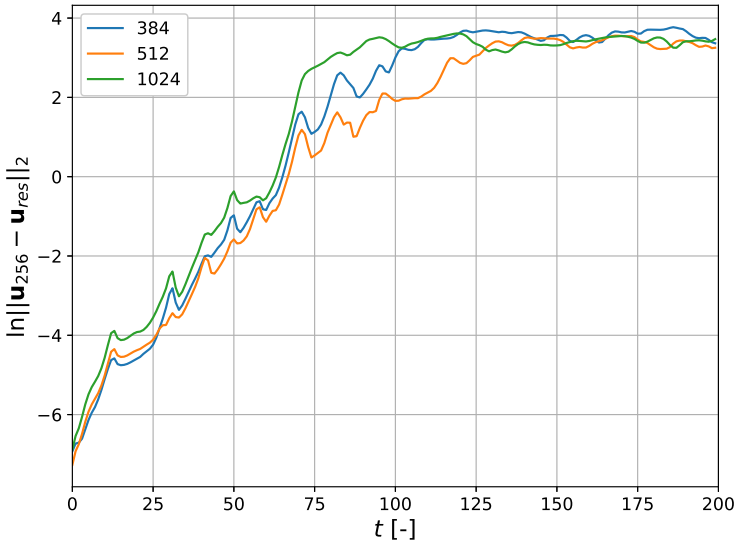


Figure 5.16. Error growth in the higher resolution flows compared to the original resolution. The flows diverge equivalent to mCLE ($\lambda_1 \approx 0.09$).

6

Conclusion

Global weather forecasting models using data-driven deep learning have shown excellent results, but do they show the same spatiotemporal chaotic dynamics as numerical weather prediction would? This research aimed to prove that Fourier Neural Operators (FNO) can replicate the intrinsic predictability of spatiotemporal chaos in the form of the Kuramoto-Sivashinsky (KS) equation.

The data-driven Markovian FNO surrogates predict the KS system well. The noise level in the power spectral density correlates negatively with the model performance and predictability score. The surrogates replicate intrinsic predictability. Even the predictability of the intrinsic predictability is accurately represented. The surrogate, with statistical certainty, reproduces the dynamical climate regarding the characteristic Lyapunov spectrum. The FNO surrogate cannot represent one of the invariant symmetries in KS, shown as a zero characteristic Lyapunov exponent in the spectrum.

Hyperparameter tuning correlates the surrogate performance to the Fourier modes represented in the FNO. The lower limit in training length trajectory for a surrogate lacking instabilities is 200 seconds.

7

Outlook & limitations

7.1. FNO

FNO is not the only DL network used in global weather forecasting. This section is dedicated to discussing different DL forecasting networks, the performance of the FNO, its limitations and other possible improvements to the network.

7.1.1. Limitations of FNO

In its current formulation, the FNO has a few limitations. FNO works only on uniform grids due to using the FFT algorithm. This might not be a problem for weather forecasting, but for other problems with non-uniform grids, it cannot work. An adaptation Geo-FNO [45] claims to solve the problem by transforming the complex grid to a uniform grid.

Another problem that Geo-FNO aims to solve is dealing with discontinuities. FNO uses global operations to predict a solution, which makes it difficult to approximate discontinuities. In many systems, neither the domain nor the solution can be considered continuous. An operator that considers only continuous Fourier series should then lead to errors in the surrogate. In Lu et al. 2022 [46], a fair comparison is made between FNO and DeepONet, another neural operator-type network that does not require strict conditions. The author ordered a list of differences shown in table 7.1. Notable limitations, but not of importance for weather forecasting are: the input and output domains have to be the same $D = D'$, discretization of output function is necessary, and prediction locations have to be on the grid because for forecasting with an autoregressive operator, all are necessary. The necessary full-field observation data is similarly not a problem if the FNO can train directly on reanalysis data of ERA5.

	DeepONet	FNO
Input domain D' & Output domain D'	Arbitrary	Cuboid, $D = D'$
Discretization of output function u	No	Yes
Mesh	Arbitrary	Grid
Prediction location	Arbitrary	Grid points
Full-field observation data	No	Yes
Discontinuous functions	Good	Questionable

Table 7.1. Comparison between vanilla DeepONet and vanilla FNO. Copied from Lu et al. 2022 [46]

The zero-shot superresolution claimed by FNO works well for smooth functions as the FNO can accurately predict the KS model using global convolutions. But, using the FNO to predict a process with a lot of energy in the high frequencies close to the Nyquist frequency might not work well. To the authors' knowledge, it has also not been studied yet. Although FourCastNet [6] in figure 2.6 has notable numerical diffusion with its smaller scale feature. Future research might apply FNO to non-smooth problems and determine which truncation in the Fourier domain can accurately predict them.

7.1.2. Larger timesteps

As shown in the results, the model error increases with larger timesteps, but that does not mean that large timestep models should perform worse with longer-term forecasts. A small error made by a small timestep model grows exponentially due to the chaos. Future research can try to find an optimal timestep for high predictability.

A greedy algorithm can be used when the forecasting frequency required is higher than the optimal forecasting timestep. A greedy algorithm uses multiple models trained on different timesteps, then uses the least steps necessary to forecast at T time. Pangu-Weather used a greedy algorithm with three models: a 1-hour, 6-hour and 12-hour step model [4]. For a prediction 20 hours ahead, the predictor uses one 12-hour step, one 6-hour step and two 1-hour steps.

7.2. Global weather forecasting

NWP agencies use different methodologies than Lyapunov time and Lyapunov exponent when describing predictability in the weather. The idea of error doubling with time loses relevance in practice. The KS model abides by chaos theory, but the weather is significantly more complex. The following sections will dissect the differences between a KS model and global weather.

7.2.1. Scale matters!

The weather consists of multi-scale processes. Global Weather Forecasting (GWF) mainly concerns the planetary- and synoptic-scale processes in the atmosphere. The smaller mesoscale and microscale on the order of 100-1km processes are mostly parametrized. In general, smaller-scale processes have a shorter timescale as well. The timescale impacts the predictability of the process. A synoptic-scale Rossby wave has a longer timescale; on its own, Rossby waves can be predicted multiple days into the future [24]. On the other hand, mesoscale convective clouds are harder to predict and have timescales of 1-10 hours. Assuming GWF could resolve both synoptic- and mesoscale processes, the question can be asked, which timescale would impact the predictability the most? The mesoscale convective cloud's predictability is much lower than the Rossby wave. However, the mesoscale process could appear and disappear simultaneously before the Rossby wave ends. The prediction error for a mesoscale convective cloud reaches saturation levels faster than the Rossby wave.

In a GWF error growth graph, such as figure 2.7, it is hard to find exponential growth as many different scale processes are superimposed within the model. Lorenz 2006 [24] illustrated the problem using a simple coupled system consisting of two timescales. The predictability of the total system showed exponential growth from the smallest scale first, after which it slowed down to quasi-exponential growth from the larger scale process. The quasi-exponential growth reached a saturation level associated with the larger scale process, while the initial exponential error growth indicates much lower predictability.

The assumption that predictability can be associated with the initial exponential error growth is only partly correct. Past studies from the '60s until the '90s have used the initial exponential error growth, or error rate doubling time, for predictability. Over those years, the resolution and accuracy of NWP models increased significantly.
With the increase in resolution, the estimated error doubling rate decreased slowly: In 1965 Mintz-Arakawa found a doubling rate of 4 days. Afterwards, Smagorinsky in 1969, Lorenz 1982, and Simmons et al. 1995 described lower rates of 3, 2.1-2.4 and 1.5 days, respectively [24]. Poorer physics may have overestimated the predictability, but a more significant factor was that the spatial resolution on which the models operated increased over the years. Smaller features could now be represented in the models. Errors in features formerly not captured amplified the initial error growth. Thus the smaller scale processes were included in the resolved scales and seemed to decrease the intrinsic predictability. Therefore using Lyapunov exponents to describe the predictability in the GWF was disregarded and other predictability measures were found. Finite-Size Lyapunov Exponent (FSLE) or Finite-Time Lyapunov Exponent (FTLE) are replacements which measure the expansion in phase space with a fixed time length or expansion size, to overcome the difficulties with the instantaneous perturbation growth of Lyapunov exponents. In the KS system, there is only one process scale influencing predictability. Thus, Lyapunov exponents remain effective.

Another reason the LE were disregarded was that the initial uncertainty in the weather state was relatively large. Therefore the initial exponential growth could not be assumed in ensembles. Lyapunov exponents do have a function in current-day ensemble forecasting, but the use is limited to the initial perturbation in which the ensemble is perturbed.

7.2.2. Randomness on top of complexity

An essential assumption in the research methodology is the deterministic nature of the toy model. Assuming a perfect model, a deterministic system should be entirely predictable using exact initial conditions. Uncertainties in initial conditions can then be considered the only source of error growth in KS.

In NWP, the parametrization of the small-scale processes has to be used. Proven is that the small-scale features add randomness to the system [24]. ECMWF leans into the randomness by using stochastic parametrization of the subgrid scales. The randomness increases the spread of ensemble predictions. The randomness in smaller scales is added as uncertainty to the total uncertainty of an ensemble. The randomness does not change the methodology of quantifying the predictability; it is simply another uncertainty added to the ensemble. The validity of a predictability window in a stochastic system should not change. Still, a surrogacy study of a stochastic chaotic toy model could be considered.

7.2.3. Climate change changes predictability?

In the analysis of CLEs, ergodicity is used to estimate the CLEs. By definition, CLEs are properties formed by long-time averages. Ergodicity in a dynamical system describes that flow will eventually visit all parts of the space it moves in. In more colloquial terms it describes the climate of a dynamical system being constant.

The CLE algorithm is based on multiplying enough (N) short-term (T) perturbation growths to estimate a long-term average. Averaging many local short-term dynamical values should approach the long-term average dynamics. The theorem proving the algorithm's validity, Osedelets multiplicative ergodic theorem, assumes the ergodicity of the dynamical system. In the global weather system, there are many outside forcings which do not depend on the weather itself but significantly impact predictability. Ergodicity cannot be assumed in the weather, because the space in which the weather moves around is time-dependent. The climate of the weather is not constant but is influenced by external processes with the increase in CO_2 levels being the obvious example. Therefore it proves challenging to describe long-term time averages in predictability.

However, that is not the only reason NWP would not use CLE or similar predictability metrics. The CLEs do not describe how predictable a system is in the short term. It is a coincidence that the local LE found in the intrinsic predictability window aligns precisely with the CLE. In general dynamical systems have wildly varying LE depending on where you are in the phase space. For example, figure 1.4 shows three wildly different predictabilities derived from the same dynamical system. NWP shows similar signs with local high or low predictability [12, 47, 48]. The interest in localized predictability is of greater importance to weather forecasting agencies than estimating a long-term average of predictability.

7.2.4. Random perturbations in a high dimensional phase space

Part of the methodology is to use a brute-force approach to create the ensemble for the intrinsic predictability window. A hundred random orthogonal perturbation vectors are picked for a good statistical representation. It works under the assumption that a random perturbation always has a component in the direction of the maximum LE. The assumption is valid due to the relatively limited 128, 256, and 512 gridpoint discretized models used. Compared to an NWP with model sizes on the order of a billion gridboxes with multiple variables, a random perturbation is not expected to grow according to the leading error growth vector. Random perturbations in an NWP model dissipate faster than perturbations can grow with the maximum LE [49]. Meaning that the ensemble will not spread enough to convey every possible forecast. A bad ensemble is schematically depicted in figure 7.1 with a control forecast, a negatively and positively perturbed ensemble member. In the case of a bad ensemble, the truth is not represented within the spread of the ensemble, while for a good ensemble, the spread is larger and contains the truth.

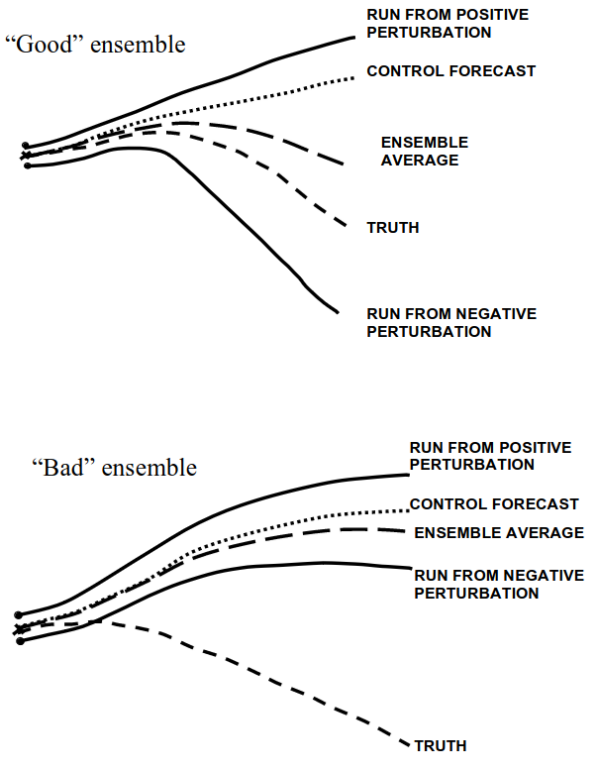


Figure 7.1. Schematic of a potential ensemble with (a) depicting a 'good' ensemble when the truth falls between the two perturbed ensemble members and (b) a 'bad' ensemble. Copied from Kalnay et al. 2006 [49]

In NWP, a different perturbation method is used. There are two main methods, the singular vector and the bred vector. In both, the approximate goal is to find the most unstable vector around the initial state and create perturbations around it. In the case of the KS model, these approaches (primarily bred vectors) are similar to iteratively finding the vector related to the maximum LE and perturbing the ensemble members in that direction.

7.3. Data-driven probabilistic forecasting

The theoretical implications of a fully data-based fully nonlinear network (FNO) inheriting the predictability of spatiotemporal chaos are noteworthy. Specifically, training an FNO using a global weather observational dataset could potentially enable accurate prediction of the intrinsic predictability of global weather. However, this conclusion must be approached with caution, as it overlooks the many differences between the KS model and the complex global weather system. Assuming that these discrepancies are reconciled, the potential for accurately predicting the global weather's intrinsic predictability using a data-based FNO becomes conceivable. In practical terms, predicting the likelihood of rain in the next three days is crucial for users seeking dependable weather forecasts, and this prediction directly correlates with the current level of weather state predictability. While current numerical weather prediction ensembles offer probabilistic forecasts, future advancements may include data-driven probabilistic forecasts in addition to traditional physics-based ones.

References

1. Richardson, L. F. & Lynch, P. *Weather Prediction by Numerical Process* ISBN: 9780521680448 (Cambridge University Press, Aug. 2007).
2. Lynch, P. The origins of computer weather prediction and climate modeling. *Journal of Computational Physics* **227,** 3431–3444. ISSN: 10902716 (Mar. 2008).
3. Bauer, P., Thorpe, A. & Brunet, G. *The quiet revolution of numerical weather prediction* Sept. 2015.
4. Bi, K. *et al.* Pangu-Weather: A 3D High-Resolution Model for Fast and Accurate Global Weather Forecast. http://arxiv.org/abs/2211.02556 (Nov. 2022).
5. Weyn, J. A., Durran, D. R., Caruana, R. & Cresswell-Clay, N. Sub-Seasonal Forecasting With a Large Ensemble of Deep-Learning Weather Prediction Models. *Journal of Advances in Modeling Earth Systems* **13.** ISSN: 19422466 (July 2021).
6. Pathak, J. *et al.* FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators. http://arxiv.org/abs/2202.11214 (Feb. 2022).
7. Keisler, R. Forecasting Global Weather with Graph Neural Networks. http://arxiv.org/abs/2202.07575 (Feb. 2022).
8. Lam, R. *et al. GraphCast: Learning skillful medium-range global weather forecasting* 2022. https://arxiv.org/abs/2212.12794.
9. Guibas, J. *et al.* Adaptive Fourier Neural Operators: Efficient Token Mixers for Transformers. http://arxiv.org/abs/2111.13587 (Nov. 2021).
10. Li, Z. *et al.* Fourier Neural Operator for Parametric Partial Differential Equations. http://arxiv.org/abs/2010.08895 (Oct. 2020).
11. Lorenz, E. N. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences* **20,** 130–141. ISSN: 0022-4928. http://journals.ametsoc.org/doi/10.1175/1520-0469(1963)020<0130:DNF>2.0.CO;2 (Mar. 1963).
12. Shukla, J. *Predictability in the Midst of Chaos: A Scientific Basis for Climate Forecasting* tech. rep. (). www.sciencemag.org.
13. *Predictability of Weather and Climate* (eds Palmer, T. & Hagedorn, R.) ISBN: 9780521848824. https://www.cambridge.org/core/product/identifier/9780511617652/type/book (Cambridge University Press, July 2006).
14. Palmer, T. The ECMWF ensemble prediction system: Looking back (more than) 25 years and projecting forward 25 years. *Quarterly Journal of the Royal Meteorological Society* **145,** 12–24. ISSN: 1477870X (2019).
15. Tadmor, E. *THE WELL-POSEDNESS OF THE KURAMOTO-SIVASHINSKY EQUATION** tech. rep. 4 (1986).
16. Hyman, J. M. & Nicolaenko, B. The Kuramoto-Sivashinsky equation: A bridge between PDE'S and dynamical systems. *Physica D: Nonlinear Phenomena* **18,** 113–126. ISSN: 0167-2789 (Jan. 1986).
17. Shibata, H. & Ishizaki, R. *Characterization of a system described by Kuramoto-Sivashinsky equation with Lyapunov exponent* tech. rep. (1999), 314–321. www.elsevier.com/locate/physa.
18. Emmert-Streib, F., Yang, Z., Feng, H., Tripathi, S. & Dehmer, M. *An Introductory Review of Deep Learning for Prediction Models With Big Data* Feb. 2020.
19. Tran, A., Mathews, A., Xie, L. & Ong, C. S. Factorized Fourier Neural Operators. http://arxiv.org/abs/2111.13802 (Nov. 2021).

20. Pathak, J., Hunt, B., Girvan, M., Lu, Z. & Ott, E. Model-Free Prediction of Large Spatiotemporally Chaotic Systems from Data: A Reservoir Computing Approach. *Physical Review Letters* **120.** ISSN: 10797114 (Jan. 2018).

21. Li, Z. *et al.* Learning Dissipative Dynamics in Chaotic Systems. http://arxiv.org/abs/2106.06898 (June 2021).

22. Rasp, S. *et al.* WeatherBench: A Benchmark Data Set for Data-Driven Weather Forecasting. *Journal of Advances in Modeling Earth Systems* **12.** ISSN: 1942-2466. https://onlinelibrary.wiley.com/doi/10.1029/2020MS002203 (Nov. 2020).

23. Chen, W. Y. Estimate of Dynamical Predictability from NMC DERF Experiments. *Monthly Weather Review* **117,** 1227–1236. ISSN: 0027-0644. http://journals.ametsoc.org/doi/10.1175/1520-0493(1989)117<1227:EODPFN>2.0.CO;2 (June 1989).

24. Lorenz, E. N. *Predictability—A problem partly solved. Predictability of Weather and Climate, T. Palmer and R. Hagedorn, Eds* 2006.

25. Ayers, D., Lau, J., Amezcua, J., Carrassi, A. & Ojha, V. Supervised machine learning to estimate instabilities in chaotic systems: estimation of local Lyapunov exponents. http://arxiv.org/abs/2202.04944 (Feb. 2022).

26. Mandelbrot, B. How Long Is the Coast of Britain? Statistical Self-Similarity and Fractional Dimension. *Science* **156,** 636–638. ISSN: 0036-8075 (May 1967).

27. Sivashinsky, G. I. Nonlinear analysis of hydrodynamic instability in laminar flames—I. Derivation of basic equations. *Acta Astronautica* **4,** 1177–1206. ISSN: 0094-5765 (Nov. 1977).

28. Sivashinsky, G. *INSTABILITIES, PATTERN FORMATION, AND TURBULENCE IN FLAMES* tech. rep. (1983), 179–99. www.annualreviews.org.

29. Kalogirou, A., Keaveny, E. E. & Papageorgiou, D. T. An in-depth numerical study of the two-dimensional Kuramoto-Sivashinsky equation. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* **471.** ISSN: 14712946 (July 2015).

30. Edson, R. A., Bunder, J. E., Mattner, T. W. & Roberts, A. J. Lyapunov exponents of the Kuramoto-Sivashinsky PDE. *The ANZIAM Journal* **61,** 270–285. ISSN: 1446-1811. http://arxiv.org/abs/1902.09651 (Feb. 2019).

31. Robinson, J. C. *Inertial manifolds for the Kuramoto-Sivashinsky equation* tech. rep. (1994), 190–193.

32. Jolly A'l, M. S., Kevrekidis A'b, I. G. & Titi, E. S. *APPROXIMATE INERTIAL MANIFOLDS FOR THE KURAMOTO-SIVASHINSKY EQUATION: ANALYSIS AND COMPUTATIONS* tech. rep. (1990), 38–60.

33. Wittenberg, R. W. & Holmes, P. *Scale and space localization in the Kuramoto-Sivashinsky equation* tech. rep. (1999). http://chaos.aip.org/chaos/copyright.jsp.

34. Zgliczynski, P. Attracting Fixed Points for the Kuramoto–Sivashinsky Equation: A Computer Assisted Proof. *SIAM Journal on Applied Dynamical Systems* **1,** 215–235. ISSN: 1536-0040 (Jan. 2002).

35. Zeng, X & Pielke, R. A. *North-Holland PHYSICS LETTERS A , What does a low-dimensional weather attractor mean?* tech. rep. (1993), 299–304.

36. Patil, D. J., Hunt, B. R., Kalnay, E., Yorke, J. A. & Ott, E. Local Low Dimensionality of Atmospheric Dynamics. *Physical Review Letters* **86,** 5878–5881. ISSN: 0031-9007 (June 2001).

37. Kovachki, N. *et al.* Neural Operator: Learning Maps Between Function Spaces. http://arxiv.org/abs/2108.08481 (Aug. 2021).

38. Li, Z. *et al.* Learning Dissipative Dynamics in Chaotic Systems. http://arxiv.org/abs/2106.06898 (June 2021).

39. Sterk, A. E., Holland, M. P., Rabassa, P., Broer, H. W. & Vitolo, R. Predictability of extreme values in geophysical models. *Nonlinear Processes in Geophysics* **19,** 529–539. ISSN: 10235809 (2012).

40. Li, X., Ding, R. & Li, J. A New Technique to Quantify the Local Predictability of Extreme Events: The Backward Nonlinear Local Lyapunov Exponent Method. *Frontiers in Environmental Science* **10.** ISSN: 2296665X (Mar. 2022).

41. Benettin, G., Galgani, L., Giorgilli, A. & Strelcyn, J.-M. Lyapunov Characteristic Exponents for smooth dynamical systems and for hamiltonian systems; a method for computing all of them. Part 1: Theory. *Meccanica* **15,** 9–20. ISSN: 0025-6455. http://link.springer.com/10.1007/BF02128236 (Mar. 1980).

42. Benettin, G., Galgani, L., Giorgilli, A. & Strelcyn, J.-M. Lyapunov Characteristic Exponents for smooth dynamical systems and for hamiltonian systems; A method for computing all of them. Part 2: Numerical application. *Meccanica* **15,** 21–30. ISSN: 0025-6455. http://link.springer.com/10.1007/BF02128237 (Mar. 1980).

43. Ramasubramanian, K & Sriram, M. *A comparative study of computation of Lyapunov spectra with different algorithms* tech. rep. (2000).

44. Pathak, J., Lu, Z., Hunt, B. R., Girvan, M. & Ott, E. Using machine learning to replicate chaotic attractors and calculate Lyapunov exponents from data. *Chaos* **27.** ISSN: 10541500 (Dec. 2017).

45. Li, Z., Huang, D. Z., Liu, B. & Anandkumar, A. Fourier Neural Operator with Learned Deformations for PDEs on General Geometries. http://arxiv.org/abs/2207.05209 (July 2022).

46. Lu, L. *et al.* A comprehensive and fair comparison of two neural operators (with practical extensions) based on FAIR data. *Computer Methods in Applied Mechanics and Engineering* **393.** ISSN: 00457825 (Apr. 2022).

47. Hoskins, B. J. *Predictability from a dynamical meteorology perspective.* eng. PhD thesis (Shinfield Park, Reading, Apr. 2003), 15–28.

48. Palmer, T. N. in *Predictability of Weather and Climate* 1–29 (Cambridge University Press, July 2006). https://www.cambridge.org/core/product/identifier/CBO9780511617652A008/type/book_part.

49. Kalnay, E., Hunt, B., Ott, E. & Szunyogh, I. in *Predictability of Weather and Climate* 157–180 (Cambridge University Press, July 2006). https://www.cambridge.org/core/product/identifier/CBO9780511617652A014/type/book_part.

A

Viscosity-independent FNO

This section is dedicated to demonstrating FNO can also estimate flows where the attractor is not constant. The viscosity constant ν is introduced as an extra variable. The ν directly impacts the attractor due to the hyperdiffusion being more or less active. The formulation of the KS eqn. 3.6 with a viscosity constant ν in front of the hyperdiffusion term becomes:

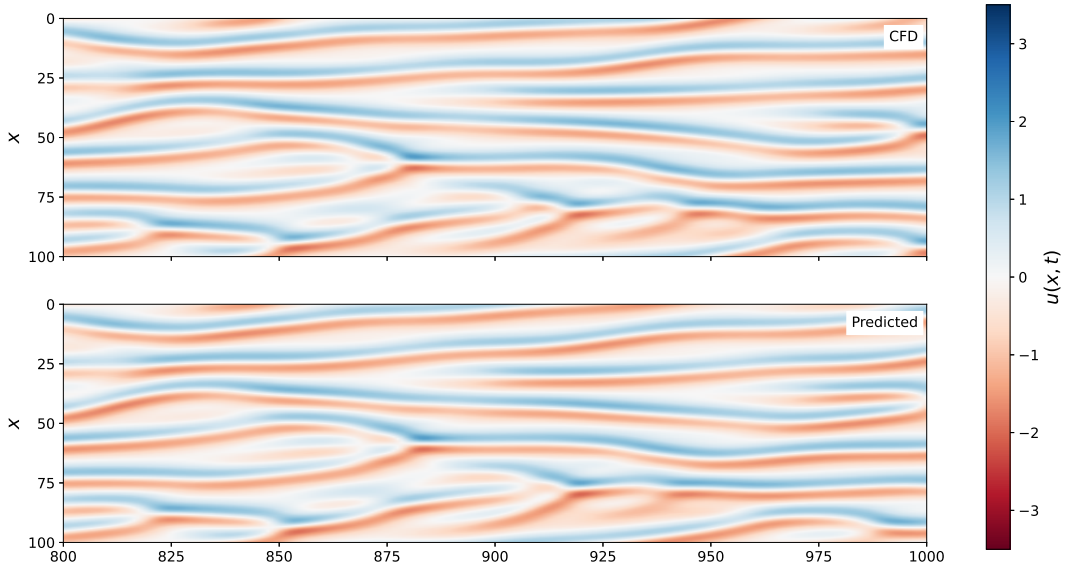
$$\partial_t u + \partial_x^2 u + \nu \partial_x^4 u + u \partial_x u = 0, \tag{A.1}$$

The combination of ν and L is found to dictate how the KS system bifurcates. The dimensionless bifurcation parameter is $\tilde{L} = \frac{L}{\sqrt{\nu}}$ [16]. In this experiment, the domain size L will be set at 100 to replicate constant geometry while the viscosity varies in value. Five 1000-second flows with different viscosities are created for training. The upper limit is an equivalent flow to what KS200 is trained on, and the lower limit is an equivalent flow to what KS60 is trained on. Four in-between values of viscosity are picked as interpolation flows. The FNO does not see these flows during training. The viscosities are given in table A.1.

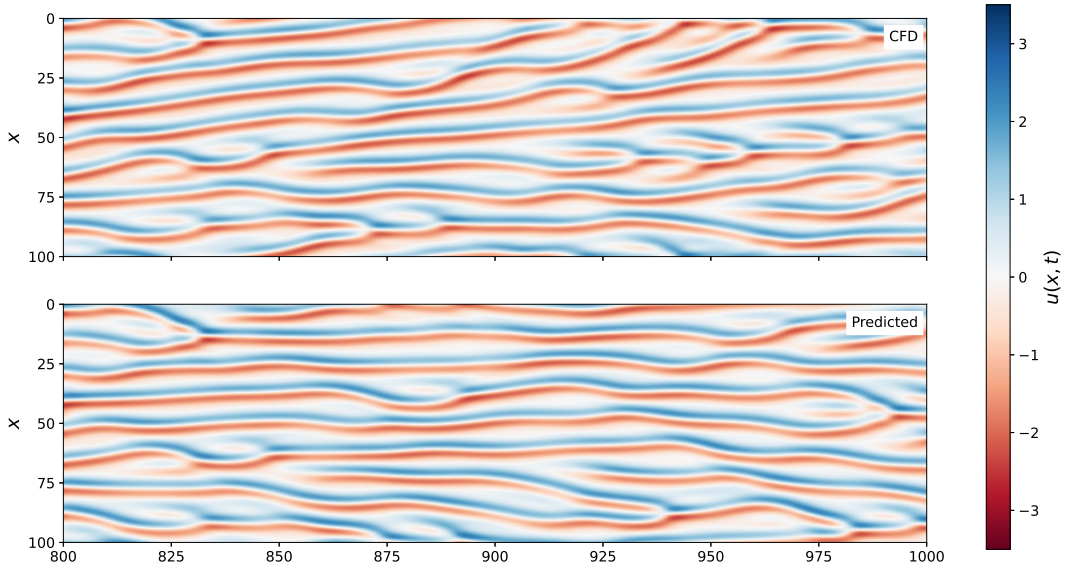
$\tilde{L}$		60	70	80	90	100	125	150	175	200
$\tilde{L} = L/\sqrt{\nu}$	ν	$\frac{25}{9}$	$\frac{100}{49}$	$\frac{25}{16}$	$\frac{100}{81}$	1	$\frac{16}{25}$	$\frac{4}{8}$	$\frac{16}{49}$	$\frac{1}{4}$

Table A.1. Viscosities used in training and testing (grey).

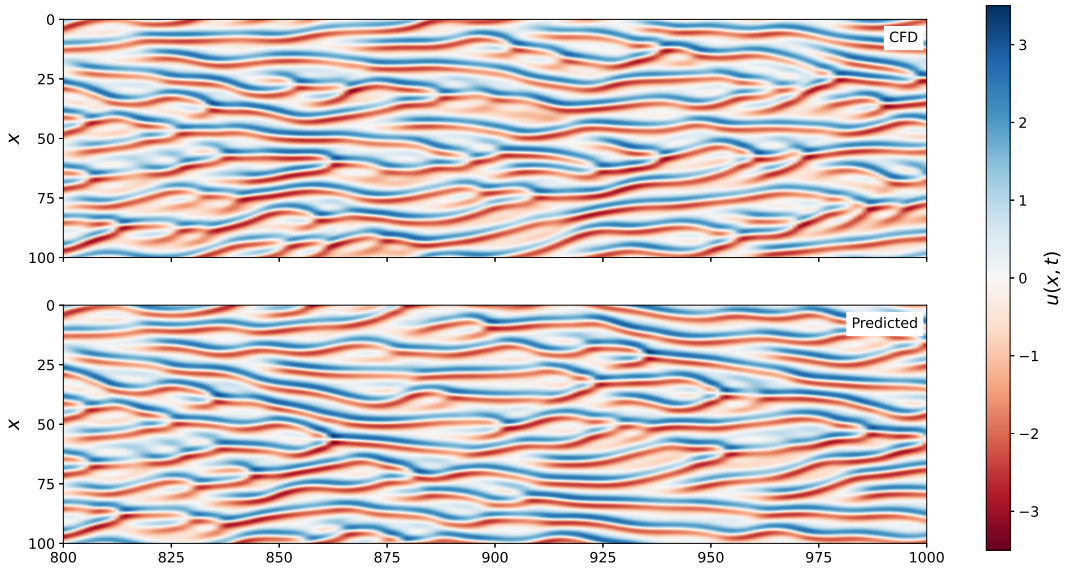
The $\sqrt{\nu}$ is appended to the input for each gridpoint. The input to the FNO becomes $[\mathbf{u}, \mathbf{x}, \sqrt{\nu}]$. In figures A.1 a to e and A.2 a to d, the predicted flows are plotted against the CFD ground truths. The viscosity has an impact on predictability. High viscosity shows a slow-changing solution with high predictability; the opposite is true for low viscosity. A PSD comparison is shown in figure A.3. The peak PSD moves to the higher frequencies with a lower viscosity. The cut-off mode for FNO can be seen at $q = 3$. At low viscosity, there is more energy in high frequencies above the cut-off mode, which adds a lot of noise to the predicted flow. Increasing the number of modes in the FNO should help it perform better based on the results shown in section 5.4.



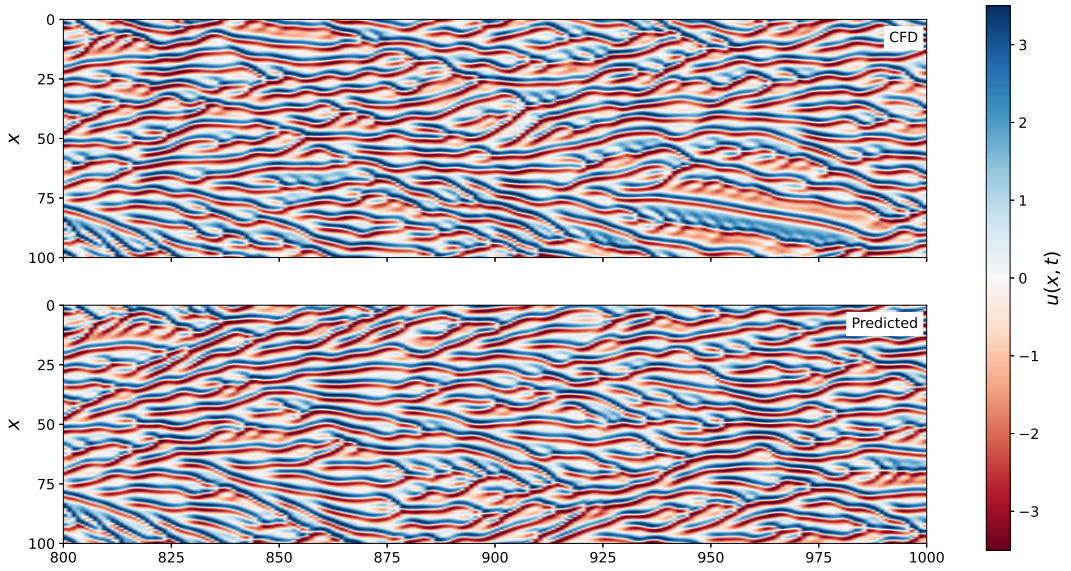
(a) $\nu = \frac{25}{9}$



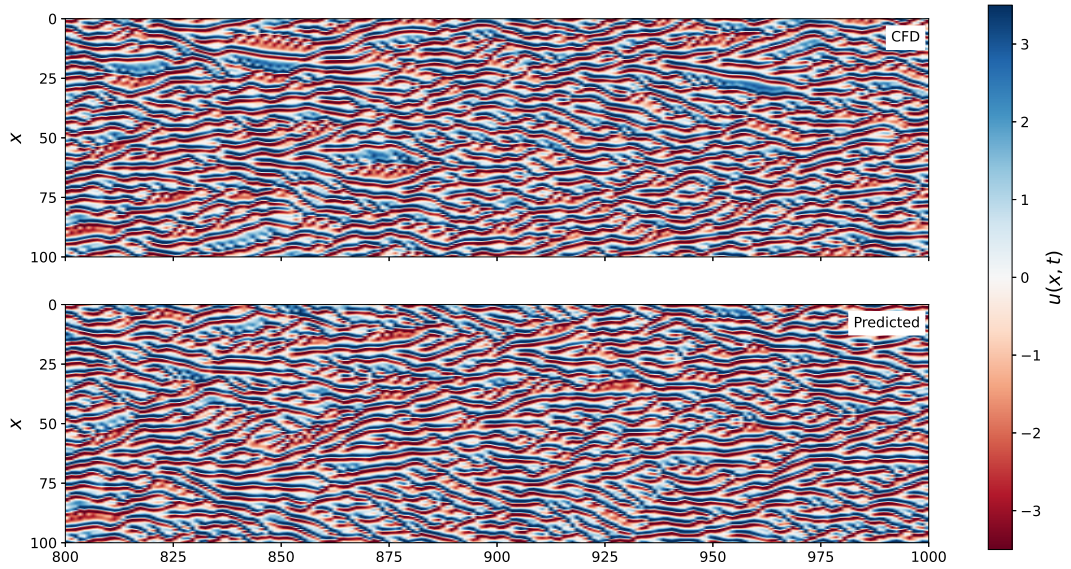
(b) $\nu = \frac{25}{16}$



(c) $\nu = 1$

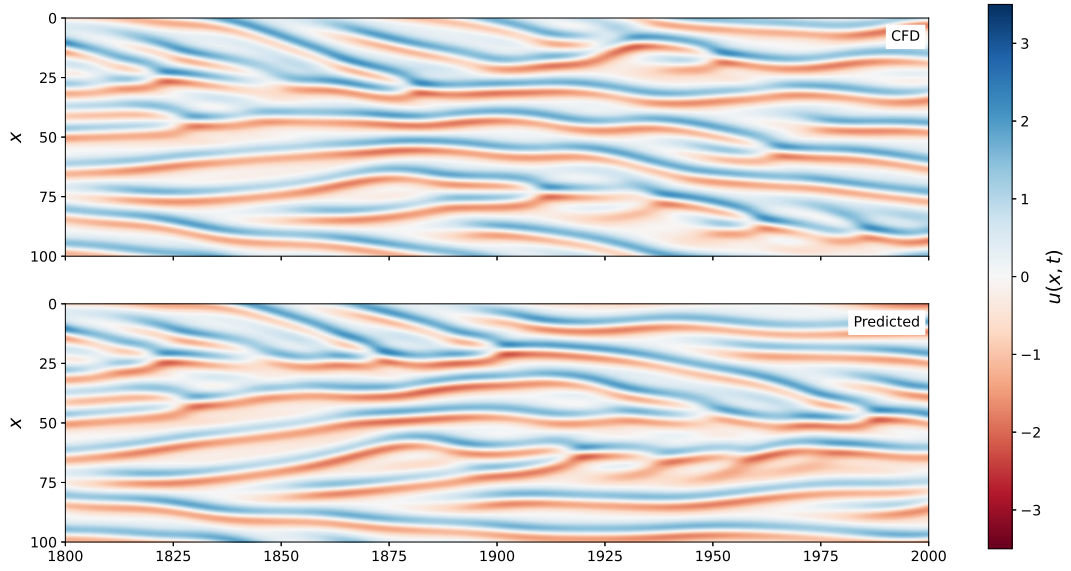


(d) $\nu = \frac{4}{9}$

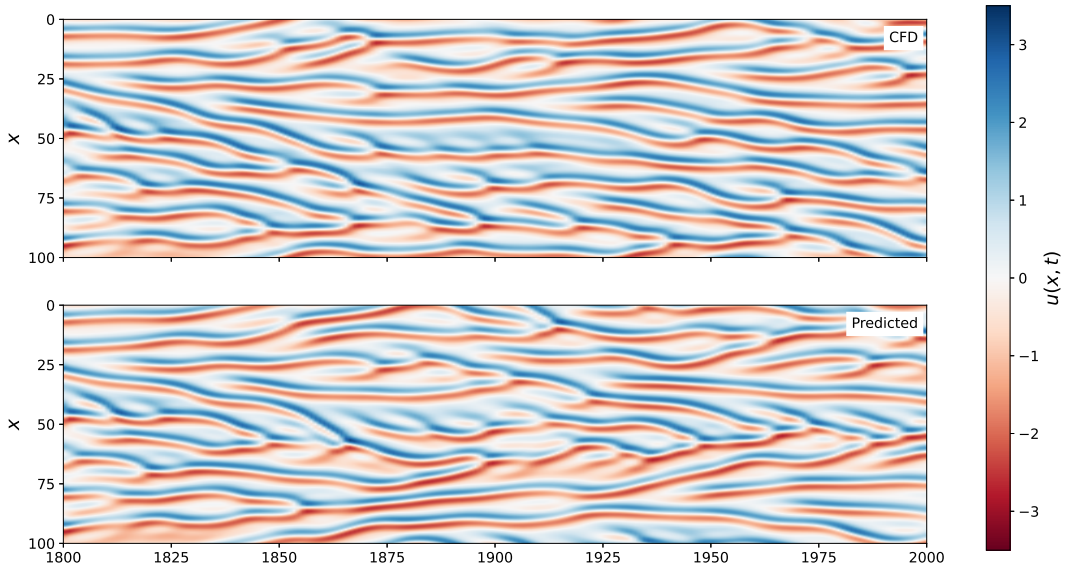


(e) $\nu = \frac{1}{4}$

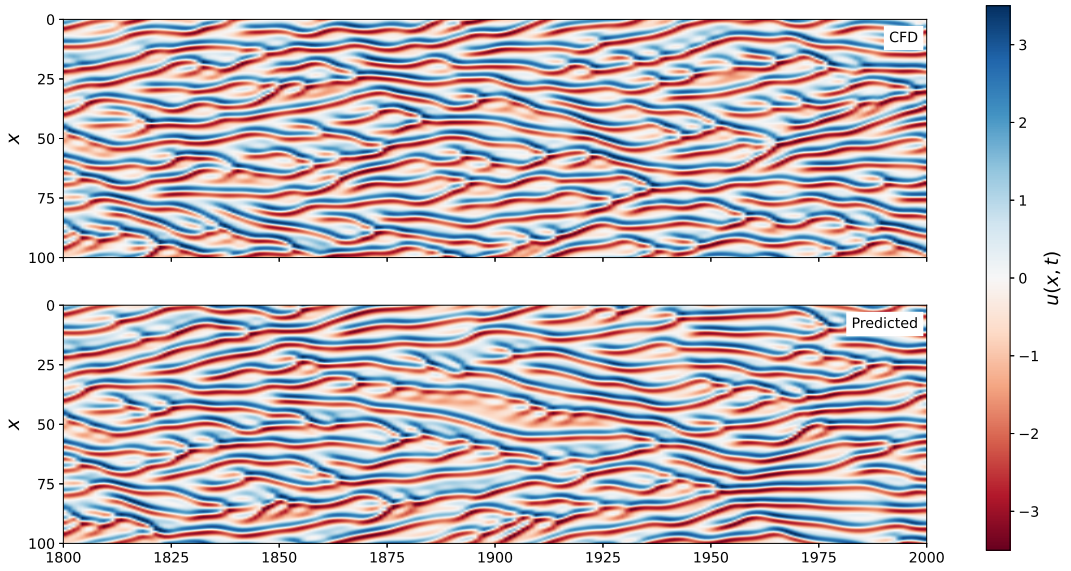
Figure A.1. Training dataset flows for the viscosity-independent FNO surrogate. Each shows a part of the training flow and the predicted flow.



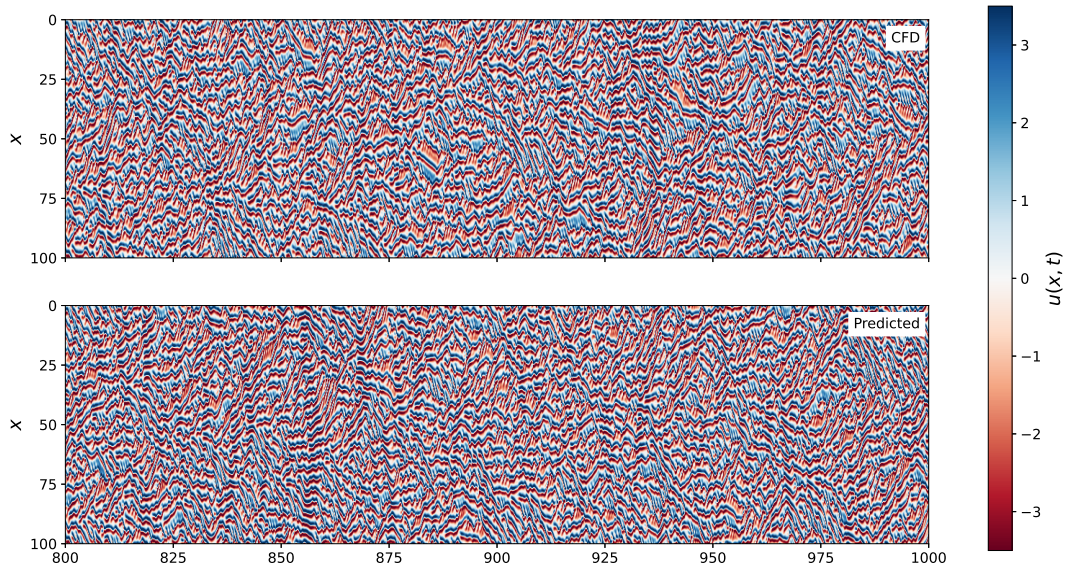
(a) $\nu = \frac{100}{49}$



(b) $\nu = \frac{100}{81}$



(c) $\nu = \frac{16}{25}$



(d) $\nu = \frac{16}{49}$

Figure A.2. Interpolation viscosity flows for the viscosity-independent FNO surrogate. Each shows a part of a sample flow with the predicted flow.

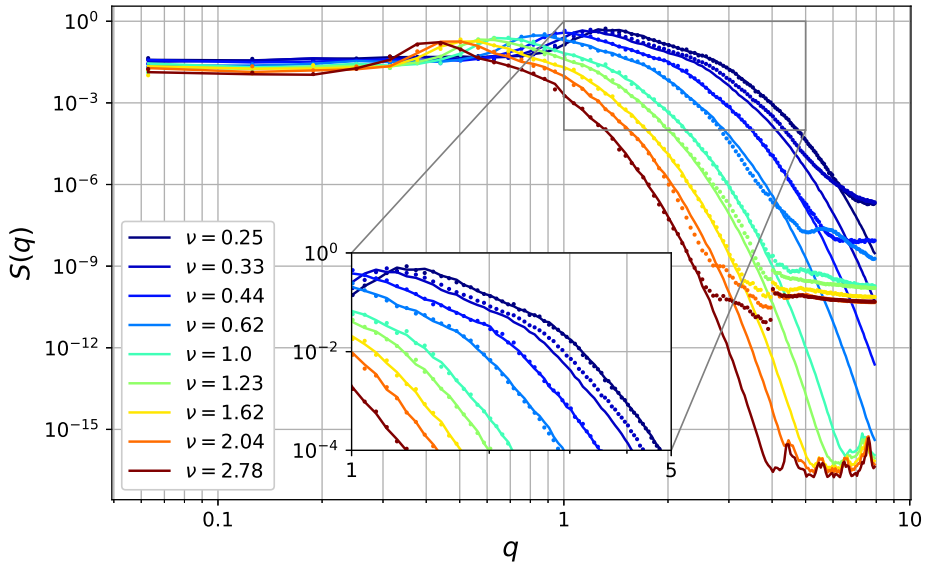


Figure A.3. PSD comparison between CFD and FNO predictions. The solid lines denote the CFD, and the dotted lines denote the FNO predictions.

B

Fourier collocation method

As a reference, we would need a good solution to the K-S equation. In literature, the periodic K-S system is often solved by a Fourier Collocation method in space with an appropriate time-marching time discretization. We will shortly address the method in the following section. To recall the K-S eqn. 3.6 in a conservative form where $\partial_x u^2 = 2u\partial_x u$

$$\partial_t u + \partial_x^2 u + \partial_x^4 u + \frac{1}{2}\partial_x u^2 = 0,$$
$$u(x+L,t) = u(x,t) \quad \forall \quad 0 \leq x \leq L.$$

The spatial Fourier transform will be denoted by $\tilde{u}_k(t)$ with k the wavenumber. Then $u(x,t)$ can be approximated with a truncated Discrete Fourier Transform over M collocation points

$$u(x,t) \approx \sum_{k=-M/2}^{M/2-1} \tilde{u}_k(t) e^{i\frac{2\pi}{L}kx}, \tag{B.1}$$

$$\tilde{u}_k(t) = \int_{\xi\in\mathrm{R}} u(\xi,t) e^{-i\frac{2\pi}{L}k\xi} d\xi \tag{B.2}$$

and similarly, u^2 can be written as:

$$u^2(x,t) \approx \sum_{k=-M/2}^{M/2-1} \tilde{v}_k(t) e^{i\frac{2\pi k}{L}x}, \tag{B.3}$$

$$\tilde{v}_k(t) = \int_{\xi\in\mathbb{R}} u^2(\xi,t) e^{-i\frac{2\pi k}{L}\xi} d\xi \approx \sum_{j=0}^{M} u^2(x_j,t) e^{-i\frac{2\pi k}{L}x_j}. \tag{B.4}$$

Substituting B.1 and B.3 into the K-S equation, taking the partial derivatives inside the summation and defining $\lambda_k := \frac{2\pi k}{L}$

$$\sum_{k=-M/2}^{M/2-1} \dot{\tilde{u}}_k(t) e^{i\lambda_k x} - \sum_{k=-M/2}^{M/2-1} \lambda_k^2 \tilde{u}_k(t) e^{i\lambda_k x} + \sum_{k=-M/2}^{M/2-1} \lambda_k^4 \tilde{u}_k(t) e^{i\lambda_k x} + \frac{1}{2}\sum_{k=-M/2}^{M/2-1} i\lambda_k \tilde{v}_k(t) e^{i\lambda_k x} = 0,$$

The dot notation denotes the partial time derivative. Due to the orthogonality of the basis functions $e^{i\lambda_k x}$, where $\overline{f(x)}$ denotes the conjugate:

$$< e^{i\lambda_p x}, e^{i\lambda_q x} >= \int_0^L \overline{e^{i\lambda_p x}} e^{i\lambda_q x} dx = 0 \quad if \quad p \neq q,$$

the terms in front of the basis functions should balance out for each individual wavenumber k in the summation. The end result becomes an ODE system

$$\dot{\tilde{u}}_k(t) + (\lambda_k^4 - \lambda_k^2)\tilde{u}_k(t) + i\frac{\lambda_k}{2}\tilde{v}_k(t) = 0,$$

This ODE system is solved numerically in one way or another. The approximations lie in calculating $\tilde{v}_k(t)$ and the numerical time integration.

It can be further simplified by multiplying by an integration factor e^{At} with $A = \lambda_k^4 - \lambda_k^2$,

$$\frac{d}{dt}(e^{At}\tilde{u}_k) = \frac{d}{dt}\tilde{w}_k(t) = -i\frac{\lambda_k}{2}\tilde{v}_k(t)e^{At}$$