MODELING AND INFERENCE OF THE DYNAMICS OF SPATIOTEMPORALLY EVOLVING SYSTEMS USING EVOLVING GAUSSIAN PROCESSES

BY

JOSHUA EARL WHITMAN

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Mechanical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2018

Urbana, Illinois

Advisor:

Professor Girish Chowdhary

Abstract

In this work, we present a new differentially-constrained machine learning model, termed Evolving Gaussian Processes (E-GP), for modeling and inference of spatiotemporally evolving dynamical systems. We show that an E-GP model can be used to estimate the latent state of large-scale physical systems of this type, and furthermore that a single E-GP model can generalize over multiple physically-similar systems over a range of parameters using only a few training sets. It is also shown that an E-GP model provides access to practical physical insights into the dynamic structure of the system(s) it is trained on. In particular, from spectral analysis of the linear dynamic layer in the top level of the E-GP model, one may derive the Koopman modes and eigenvalues of the system. We are also able to derive the spatial distribution of the invariant subspaces of a system using a new clustering method. This information can be used for sensor placement and/or mobile agent path planning for robust inference of the state of the system using few measurements. We primarily demonstrate our method on computational flow dynamics (CFD) data sets on fluid flowing past a cylinder at different Reynolds numbers. Though these systems are governed by highly nonlinear partial differential equations (the Navier-Stokes equations), we show that their major dynamical modes can be captured by a linear dynamical layer over the temporal evolution of the weights of stationary kernels.

To my Mom and Dad.

Acknowledgments

This project would not have been possible without the support of many people. Many thanks to my advisor, Girish Chowdhary, whose Principles of Autonomous Decision Making class inspired me to enter this field, and who has always been an incredible guide in pushing this research forward to the next level. Thank you to my colleagues in the Distributed Autonomous Systems Lab for their ideas, criticisms, thoughts, and camaraderie. Thanks to my parents and to numerous friends whose encouragement and advice has been invaluable in supporting me through my years of study. Soli Deo Gloria.

Table of Contents

List of Figures

List of Abbreviations

E-GP	Evolving Gaussian Process.
GP	Gaussian Process.
RBF	Radial Basis Function.
CFD	Computational Flow Dynamics.
PDE	Partial Differential Equations.
NSE	Navier-Stokes equations.
RKHS	Reproducing kernel Hilbert space.
AEO	Abstract evolution equation.
KoT	Koopman Operator Theory
DMD	Dynamic Mode Decomposition.
SVD	Singular Value Decomposition.
POD	Proper Orthogonal Decomposition.
PCA	Principal Component Analysis.
RNN	Recurrent Neural Network.

List of Symbols

f	A single-valued function over a spatial ($\Omega \subseteq \mathbb{R}^n$) domain.
τ	Time.
η	Exogenous inputs (noise).
k	A positive-definite Mercer kernel, usually based on RBF.
$\mathcal{H}$	The RKHS which f belongs to, with an inner product defined by k and ψ.
ψ	Smooth map from a point in the spatial domain to $\mathcal{H}$.
N	Dimension of measurements, i.e. number of sensors.
$\mathcal{A}$	Linear transition operator in the RKHS $\mathcal{H}$.
$\mathcal{K}$	Linear measurement operator.
ζ	Measurement noise.
M	Dimension of approximate feature map, i.e. number of centers
$\widehat{\psi}$	Approximate feature map.
$\widehat{\mathcal{H}}$	Approximate feature space.
c_i	A center for a kernel.
$\widehat{A}$	Linear transition matrix in the weight space of the model.
K	Kernel matrix, which is a measurement operator on the approximate feature space.
w	A weight vector.
Re	Reynolds number.
S	State space of the system (KoT).
s	State of the system (KoT).
T	Transition operator for the system (KoT).
ϕ	Scalar observable of the system (KoT).
U	Koopman (or composition) operator (KoT).

Chapter 1

Introduction

1.1 Challenges

One of the most difficult and important problems in bringing machine learning to physical domains is modeling and tracking large-scale stochastic phenomena with both spatial and temporal (spatiotemporal) evolution [1]. Examples of such phenomena include temperature variation, CO_2 flux over large areas, extreme weather events [12] like wildfires, pedestrian traffic patterns, and fluid dynamics.

The last example is a classic physics problem for numerical analysis, and is an ongoing subject of research in the field of Computational Fluid Dynamics (CFD). This field's aim is to model fluid flow using the first principles of fluid mechanics, e.g., by using numerical methods to solve the nonlinear Navier-Stokes partial differential equations. These simulations are costly and resource-intensive, sometimes requiring days on a supercomputer to generate. This means they are ill-suited for machine-learning tasks that require access to dozens or hundreds of simulations of different but similar situations. They are even more poorly suited for online robotic applications, such as autonomous aerial, ground, or water vehicles.

In contrast to first-principles approaches, data-driven models of spatiotemporally evolving phenomena have been gaining more attention in the machine learning and statistics communities [8]. The ultimate goal of this approach would be to generate highly efficient machine learning models that can be used instead of the costly numerical simulations for design and autonomy purposes. Success of this technique could revolutionize design and control of complex physical systems, such as soft robotics, as they would significantly reduce the cost and resources required in simulations. However, in order to be successful, these models need to be able to generalize across different physical situations. For example, in the context of fluid flows, these models must be able to predict fluid dynamics at different conditions (e.g. Reynolds number) than the training data. This is a difficult problem, as it requires that the model have the capability to actually learn the underlying physics and not just input-output relationships. As far as this author knows, there is no machine learning method currently in the field which is capable of producing physically-meaningful models of such complex dynamic systems using only raw measurements. It is the aim of this

work to change that.

1.2 Contribution

1.2.1 Generalizing Across Similar Spatiotemporally Evolving Systems

In this work, we demonstrate that a machine learning model can learn *and* generalize over the physics of the Navier-Stokes partial differential equation (PDEs) using only raw measurements, which we believe is unique in our field. We believe that this model can generalize over other similarly parameterized PDEs. We also show that the E-GP model works because it is deeply connected with Koopman operator theory. We leverage these results, along with new methods for analyzing the E-GP transition matrix, to produce new methods for path planning for mobile agents seeking to discover the true state of a changing system.

This work is built on results originally published in [15], and uses the predictive mechanism therein to perform inference with a single machine learning model over multiple systems. We term this model Evolving Gaussian Processes (E-GPs), which is a differentially-constrained hierarchic modeling method that layers a linear dynamic transition model on the weights of a kernel-based model (such as a Gaussian Process (GP) or a Gaussian Radial Basis Function (RBF) Neural Network). The advantage of E-GPs is that by separating the spatial and temporal dynamics hierarchically and using a linear transition model on the weights, the learning problem becomes more tractable while complex spatiotemporal behaviors can still be captured in a relatively low-complexity model. Furthermore, the linear transition model not only provides physical insights into the system, but also potentially enables the design of observers and controllers [15, 14]. In this work, we show that E-GP is powerful enough to generalize over multiple similar PDE-governed systems, and refine it by doing an in-depth analysis of the resulting model and showing how it can be used for online inference. Our approach leaves open the possibility of modeling nonlinear behavior in the weight-space evolution using neural networks as the transition models, especially as the flow becomes turbulent at higher Reynolds numbers. However, as shown in the latter part of this work, the ability to perform spectral analysis on the linear operator in the weight space is very important. Therefore, we consider that the better potential alternative would be to replace the kernel-based model with some other machine learning model, preferable something whose parameters are also spatially encoded like RBFs.

We demonstrate our results using CFD data of flow over a bluff body over a range of Reynolds numbers from 100 to 1000. The conventional wisdom would be to learn a separate model over each Reynolds number, but our results show that this is not necessary if one leverages our spatially encoded hierarchic Evolving

Gaussian Process model. Using the learned dynamics over weights of successive kernel models, E-GP is capable of predicting the future states of functional evolution in a recurrent manner. The key benefit is that evolution of large function spaces can be transformed into learning the evolution of a relatively smaller Hilbert space which is encoded by the kernels and the associated weight vector. Furthermore, the E-GP model's transition matrix provides valuable insights into the system, showing spatial correlations in the dynamics, local modes of dynamic evolution through invariant subspaces, and eigenmodes of evolution. The latter, importantly, connects this work to ongoing work in Koopman operator theory in CFD communities.

Koopman operator theory is a way of defining an linear operator over functions, or "observables", of the state space of a dynamic system which may be nonlinear. By finding the eigenfunctions (aka Koopman modes) and corresponding eigenvalues of the operator, one obtains a description of the dynamics that can be used to model the evolution of a system given a snapshot of its measurables. In this work, we show that the eigenvalues and eigenvectors of the linear transition model in the weight space of the E-GP model correspond directly with the eigenvalues and eigenfunctions of the Koopman operator via the kernel operator. Currently, the most popular data-driven technique in CFD literature for discovering Koopman modes is Dynamic Mode Decomposition (DMD), which is what we use for comparison.

Our analysis of the transition matrix, particularly study of the Koopman modes, eigenvalues, and the invariant subspaces, has led to breakthroughs in choosing optimal paths for a mobile agent to take in a domain in order to infer the state of a system that is changing in both space and time. We have developed two new algorithmic contributions towards this end which we present here. First, we present a method, inspired by k-means clustering, for identifying areas of the domain which correspond to invariant subspaces in the linear transition matrix. Secondly, we present a method for identifying which path amongst a library of paths is most likely to provide the most useful data for a moving agent in the system.

1.3 Related Work

In the machine learning community, kernel methods constitute a very well-studied and powerful class of methods for inference in spatial domains [31], in which correlations between input variables are encoded via a covariance kernel, and the model is formed through a weighted sum of the kernels [26]. There is a significant body of literature on extending these methods to spatiotemporal modeling [33, 26]. A naive approach is to utilize both spatial and temporal variables as inputs to the kernel. However, this technique leads to an ever-growing kernel dictionary, which is computationally taxing. In recent years, some degree of success has been found [8] by focusing on designing nonseperable

and nonstationary covariance kernels for environment-specific dynamics and optimizing/learning associated hyperparameters in local regions of the input space (ours being a significant exception). The Process Convolution with Local Smoothing Kernels (PCLSK) approach [13] captures nonstationary structures by allowing variation in kernel hyperparameters across the input space, which can be modeled using additional latent Gaussian processes [20, 23, 10]. Other such methods map the nonstationary process into a latent space where the problem becomes approximately stationary [30, 22]. However, there are a few major drawbacks to this approach. First of all, these methods currently have limited scalability to large-scale phenomena, due to the fact that the hyperparameter optimization problem is not convex in general, leading to methods that are difficult to implement (like MCMC), susceptible to local minima, and can become computationally intractable for large datasets like those generated by CFD. The scalability issue is only exacerbated by the fact that data is typically retained across both space and time. Just as importantly, the models generated by these methods do not lend themselves well to addressing the important challenges of monitoring systems with sensor feedback and designing controllers.

The geostatistics community literature has several examples of the dynamical spatiotemporal modeling approach, where the focus is on finding good dynamical transition models on the linear combination of weights in a parameterized model [7]. The advantage of this approach is that when the spatial and temporal dynamics are hierarchically separated, the learning problem can be made convex if linear transition models are used; as a result, complex nonstationary kernels are often not necessary. The approach presented in this paper aligns closely with this vein of work. This was the inspiration for the Kernel Observers papers [15, 14] which were able to determine the optimal number and location of static sensors for monitoring and predicting the state of a distributed system using feedback, by constructing an observer in a reproducing kernel Hilber space. If feedback is allowed, monitoring (state recovery) and prediction (filtering) can be made more efficient than other nonstationary kernel methods [6]. This thesis builds on that work by considering the much more challenging problem of predicting the state of the system using only a *single* measurement every time step from a mobile agent.

Within the CFD community literature, a new framework for data-driven analysis of nonlinear fluid flow was introduced in the 90s, called Koopman operator theory. A Koopman operator is a linear, infinite-dimensional operator that is defined for an autonomous dynamical system and governs the evolution of its *observables*, which are scalar functions of the state [36]. Although the Koopman operator is linear with respect to the infinite-dimensional space of observable functions, it does not rely on linearizing the underlying dynamic system. Indeed, this operator has been used to analyze nonlinear fluid flows [19, 18]. If the most significant eigenfunctions, eigenvalues, and eigenmodes of the Koopman operator can be approximated from the data, many of the same

advantages of E-GP are realized: the ability to transform the state space so the dynamics appear linear, to predict the temporal evolution of the linear system, and reconstruct the state of the original nonlinear system. Dynamic Mode Decomposition is the most widely used method for finding a finite-dimensional subspace of the Koopman operator's infinite-dimensional domain to work in, first introduced in 2008 [29] and shown to be connected with the Koopman operator not long after [28]. There are many modifications to DMD in CFD literature, all aiming to improve either the theoretical or practical value of the method, but the two most common variants are the Arnoldi method and the another using singular value decomposition (SVD). Williams et al., recently applied the kernel trick to DMD, allowing the algorithm to be extended to systems with much larger dimensions [35]. However, this method is restricted to approximating the Koopman operator and is only indirectly concerned with generative models, whereas our method is concerned with the evolution of the weights which can be directly used to predict observables (we also use Gaussian kernels instead of polynomial). Most importantly, our method has the capacity to can *generalize* across similar systems, and lends itself well toward designing observers and controllers for those system(s).

For the purpose of aerospace design tasks, feedforward neural networks have been used to model a highly restricted subset of the CFD output, such as the pressure at a couple of points on an object surface [5]. Convolutional neural networks (CNNs) have been used to model either simple, low-Reynolds number, steady flows at low resolution [11], or (more recently) certain costly components of CFD simulators [32] for turbulent flows. However, as far as we know, no end-to-end neural network has been used to learn and generalize about dynamics of a complex dynamic CFD system, nor used as the basis for online inference of the state of system using sparse/mobile measurements. We hope this work provides inspiration for future research in that area.

Chapter 2

Background

2.1 Partial Differential Equations and the Navier-Stokes Equations

Partial differential equations are ubiquitous in science and engineering and have their origins in multivariate calculus with functions that operate in continuous space. In this space, any change in functional values can be represented as a combination of partial derivatives of this multivariable function with respect to the independent variables (usually time and space). Examples of such PDEs litter the various areas of science and engineering as a means to describe evolutionary dynamics of many complex systems, including areas of mechanics (solids, fluids, gases), transport phenomena in general (waves, information), electrostatics & electromagnetics, circuits, thermal sciences, quantum mechanics, transmission lines and more.

The predominant class of PDEs encountered in practical science and engineering are of the second order. These second order PDEs, especially when linear, can be classified into elliptic, parabolic or hyperbolic depending on the signature of eigenvalues of the coefficients of the PDE system. However, the generic PDEs encountered in practice are non-linear, i.e. the coefficients of the PDE system are functions of the independent variables or dependent variables or both. For example, the Navier-Stokes equations (NSE) can be classified as mixed type, i.e they can behave as hyperbolic or parabolic or elliptic systems in different regimes of the non-linear coefficients, depending on the boundary and initial conditions specified. The other consequence of the nonlinearity is chaotic dynamics, commonly referred to as turbulence, wherein any small disturbance evolve the system along bifurcations to excite unstable modes and new physical scales, in a cumulative cascading effect.

In our view, the Navier-Stokes equations represents most, if not all of the overall complexity of modeling 2nd order PDEs as it (a) allows of hybrid system behavior, i.e. elliptic-hyperbolic etc. and (b) the nonlinearity results in complex spatio-temporal dynamics that is prevalent in many practical situations. The form of the NSE for compressible Newtonian fluids is expressed below, where $\mathbf{u}$ is the fluid velocity, p is the fluid pressure, ρ is the fluid density, and μ is the fluid dynamic viscosity. [21]

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \nabla\cdot\left(\mu(\nabla\mathbf{u} + (\nabla\mathbf{u})^T)\right) + \nabla\left(-\frac{2\mu}{3}\nabla\cdot\mathbf{u}\right) + \rho\mathbf{g}$$

2.2 Kernel Observers

As presented in [15], the problem is predictive inference of a time-varying stochastic process, whose mean f evolves as $f_{\tau+1} \sim \mathbb{F}(f_\tau, \eta_\tau)$, where $\mathbb{F}$ is a distribution varying with time τ and exogenous inputs η. The goal of our approach is to hierarchically separate temporal evolution from spatial functional evolution. Our prototype is the classical and quite general *abstract evolution equation* (AEO), which can be defined as the evolution of a function u embedded in a Banach space $\mathcal{B}$: $\dot{u}(t) = \mathcal{L}u(t)$, subject to $u(0) = u_0$, and $\mathcal{L} : \mathcal{B} \to \mathcal{B}$ determines spatiotemporal transitions of $u \in \mathcal{B}$ [3]. To make this approach computationally realizable, we restrict the sequence f_τ to lie in a reproducing kernel Hilbert space (RKHS) [26]. Let $k : \Omega \times \Omega \to \mathbb{R}$ be a positive-definite Mercer kernel on a domain Ω, modeling the covariance betwen any two points in the input space. This also implies the existence of a smooth map $\psi : \Omega \to \mathcal{H}$, where $\mathcal{H}$ is an RKHS with the property $k(x, y) = \langle\psi(x), \psi(y)\rangle_{\mathcal{H}}$. The insight of the proposed model is in assuming spatiotemporal evolution in the input domain corresponds to temporal evolution of the mixing weights of a kernel model alone in the functional domain.

Let $y \in \mathbb{R}^N$ be the measurements of the function available from N sensors, $\mathcal{A} : \mathcal{H} \to \mathcal{H}$ be a linear transition operator in the RKHS $\mathcal{H}$, and $\mathcal{K} : \mathcal{H} \to \mathbb{R}^N$ be a linear measurement operator. The model for the functional evolution and measurement studied in this paper is:

$$f_{\tau+1} = \mathcal{A}f_\tau + \eta_\tau, \quad y_\tau = \mathcal{K}f_\tau + \zeta_\tau, \tag{2.1}$$

where η_τ is a zero-mean stochastic process in $\mathcal{H}$, and ζ_τ is a Wiener process in $\mathbb{R}^N$. To avoid working in dual space and have the parameters grow with the data, we work with an approximate feature map $\widehat{\psi}(x) := [\,\widehat{\psi}_1(x) \cdots \widehat{\psi}_M(x)\,]$ to an approximate feature space $\widehat{\mathcal{H}}$. Typical examples of such maps include random Fourier features [24], FastFood [16], A la Carte [37], and the Nyström method [34]. Here we use the dictionary of atoms approach as follows: let Ω be compact. Given points $\mathcal{C} = \{c_1, \ldots, c_M\}$, $c_i \in \Omega$, define the dictionary of atoms $\mathcal{F}^{\mathcal{C}} = \{\psi(c_1), \cdots, \psi(c_M)\}$, $\psi(c_i) \in \mathcal{H}$, the span of which is a strict subspace $\widehat{\mathcal{H}}$ of the RKHS $\mathcal{H}$ generated by the kernel, where $\widehat{\psi}_i(x) := k(x, c_i)$. In the approximate space case, we replace the transition operator $\mathcal{A} : \mathcal{H} \to \mathcal{H}$ in (2.1) by $\widehat{\mathcal{A}} : \widehat{\mathcal{H}} \to \widehat{\mathcal{H}}$. The finite-dimensional evolution equations approximating (2.1) in approximate dual form are

$$w_{\tau+1} = \widehat{A}w_\tau + \eta_\tau, \quad y_\tau = Kw_\tau + \zeta_\tau, \tag{2.2}$$

where we have matrices $\widehat{A} \in \mathbb{R}^{M\times M}$, $K \in \mathbb{R}^{N\times M}$, the vectors $w_\tau \in \mathbb{R}^M$, and where we have slightly abused notation to let y_τ, η_τ and ζ_τ denote their $\widehat{\mathcal{H}}$ counterparts. Here K is the matrix whose rows are of the form $K_{(i)} = [\,\widehat{\psi}_1(x_i)\ \widehat{\psi}_2(x_i)\ \cdots\ \widehat{\psi}_M(x_i)\,]$. In systems-theoretic language, the matrix acts as a measurement operator.

Modeling the system as a linear time-invariant dynamic system in the weight space enables the use of several important and useful techniques from control theory. For example, it was demonstrated in [14] that given a spatiotemporally evolving system modeled using (2.2), under certain conditions one may choose a set of N sensing locations such that even with $N \ll M$, the functional evolution of the spatiotemporal model can be estimated (which corresponds to *monitoring*) and can be predicted robustly (which corresponds to *Bayesian filtering*). The key to solving this problem is designing the measurement operator K so that the pair $(K, \widehat{A})$ is observable. By taking the Jordan decomposition of the $\widehat{A}$ and looking at the geometric multiplicities of the eigenvalues, one can determine the *cyclic index* of $\widehat{A}$. The cyclic index is a nonconservative lower bound on the number of distinct sampling locations required for the observability of system (2.2), and is equal to the number of *invariant subspaces* $\mathcal{H}_i \subset \mathcal{H}$ into which $\mathcal{A}$ can be uniquely decomposed. It is formally defined as the largest geometric multiplicity of an eigenvalue in $\widehat{A}$.

An invariant subspace of a linear mapping from some vector space to itself is a subspace W of $\mathbb{R}^M$ that is preserved by $\widehat{A}$, that is $\widehat{A}(W) \subseteq W$. Trivial examples include $\{0\}$ and $\mathbb{R}^M$, but we are interested in subspaces that correspond with a set of regions dividing the domain. Unlike neural networks, the weights in an E-GP do not exist in some abstract, difficult-to-comprehend space, but are associated with kernel centers in specific locations in the domain. We refer to this very important attribute of E-GPs as the *spatial encoding* property.

2.3 Koopman Operator Theory

Koopman operator theory is formulated in the context of studying the "dynamics of observables", that is, the dynamics of the system are analyzed by studying the evolution of functions on the state space, rather than the state space trajectories themselves.

Our formulation will follow that found in the exemplary [4]. The state space shall be denoted as S and the dynamics on S defined by the transition operator $T : M \to M$. A *scalar observable* is defined as a function $\phi : S \to \mathbb{C}$, where ϕ belongs to a function space $\mathcal{F}$ which we will specify later. A real-life interpretation of an observable function would be a sensor measuring some quantity related to the system. Instead of tracking the state trajectory $s, T(s), T^2(s), ...$, we track the trace $\phi(s), \phi(T(s)), \phi(T^2(s)), ...$ This corresponds with the familiar state

and output equation format used in the control theory community:

$$s_{n+1} = T(s), \quad y_n = \phi(s_n) \tag{2.3}$$

The Koopman operator, also known as the composition operator, is defined on the space of observables ($U_T : \mathcal{F} \to \mathcal{F}$) as follows:

$$(U\phi)(s) = (\phi \circ T)(s) = \phi(T(s)) \tag{2.4}$$

for all $s \in M$. When $\mathcal{F}$ is a vector space, then U is linear. When S is a finite set, then U is a finite-dimensional operator and can be represented by a matrix; however, in the usual case that the state space is not finite (either finite- or infinite-dimensional), then U is infinite-dimensional. Often, we would like to extend the space of scalar observables to a vector space. A natural example of this is when we have N sensors $\phi_1, ..., \phi_N$ collecting measurements which we might store as a vector to consider writing the measurements of N particular sensors as a vector

2.3.1 Dynamic Mode Decomposition

As noted before, Dynamic Mode Decomposition is the most popular method for approximating the Koopman modes of a system from a discrete time series of observations, and there exist many variants of DMD. We describe the Arnoldi method below.

The data is in the form of a sequence of snapshots: $Y_1^N = \{y_1, y_2, \ldots, y_t\}$ where $y_i \in \mathbb{R}^N$ is the i-th snapshot of the flow field measurements, and $Y_1^N \in \mathbb{R}^{N \times t}$ is a matrix whose columns are the individual snapshots. The subscript and superscript denote the index of the snapshot in the first and last columns respectively. The presumption of DMD is that we can approximately relate all of these snapshots via the linear mapping

$$y_{i+1} = Ay_i$$

This implies that

$$Y_2^t = AY_1^{t-1} + re_{t-1}^T$$

where r is the vector of residuals that accounts for behaviors that cannot be described completely by A, $e_{t-1} = \{0, 0, \ldots, 1\} \in \mathbb{R}^{t-1}$. The DMD algorithm outputs the eigenvalues and eigenvectors of A.

The Arnoldi approach, noting that the number of measurements N is often much larger than the number of snapshots t, is to rewrite the above matrix equation as

$$Y_2^t = AY_1^{t-1} = Y_1^{t-1}B + re_{t-1}^T$$

where B is the companion matrix

$$B = \{e_2, e_3, \ldots, e_{t-1}, a\}, \qquad y_t = a_1 y_1 + \cdots + a_{t-1} y_{t-1} + r$$

The vector a can be computed by solving a least squares problem. The eigenvalues of A are approximated by the eigenvalues of S, and the eigenvectors are approximated by $Y_1^{T-1} v$ where v is an eigenvector of S.

2.4 k-Means Clustering

k-means clustering is an algorithm [17] that aims to partition M vectors into k clusters in which each vector belongs to the cluster with the nearest mean, with the result that the vector space is divided into cells. The problem is NP-hard, however there are efficient heuristic algorithms that widely employed and converge quickly to local optimums.

Specifically, given a set of vectors $\{x_1, \ldots, x_n\}$, k-means clustering aims to partition the n observations into $k \leq n$ sets $S = S_1, \ldots, S_k$ so as to minimize the within-cluster sum of squares (WCSS) (i.e. variance), which handily equals the pairwise squared deviations of points in the same cluster. That is, the problem is

$$\arg\min_S \sum_{i=1}^{k} \sum_{x \in S_i} \|x - \mu_i\|^2 = \arg\min_S \sum_{i=1}^{k} |S_i| \mathrm{Var} S_i = \arg\min_S \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{x,y \in S_i} \|x - y\|^2$$

The standard, iterative algorithm for solving this problem is as follows:

1. Assignment step: assign each vector x_i to the cluster whose mean has the least squared Euclidean distance (the "nearest" one).
2. Calculate the new means to be the centroids of the new clusters, $m_i^{(t+1)} = \frac{1}{|S_i^t|} \sum_{x_j \in S_j^t} x_j$

The algorithm is done when the assignments no longer change, though there is no guarantee that the optimum is found. The most common way to initialize the algorithm is to choose k vectors from the data set and use these as the initial means (Forgy method).

Chapter 3

Methods

3.1 Evolving Gaussian Processes

The Evolving Gaussian Processes method builds on the Kernel Observers method. The primary novelty in our method of generating a model is learning an $\widehat{A}$ matrix for *multiple* systems. We found that the class of functional evolutions $\mathbb{F}$ defined by linear Markovian transitions in a RKHS is still sufficient to model the nonlinear Navier Stokes equations, since the unknown map ψ allows us to model highly nonlinear dynamics in the input space. However, we do expect that phenomena such as bifurcation or turbulence will require nonlinear mappings $\mathcal{H}$. There are three steps to generate an Evolving Gaussian Process model:

1. After picking the kernel and estimating the bandwidth hyperparameter σ (we utilize the maximum likelihood approach, although other approaches can be used), find an optimal basis vector set $\mathcal{C}$ using the algorithm in [9].

2. Use Gaussian process inference to find weight vectors for each time-step in the training set(s), generating the sequence $w_\tau, \tau = 1, \ldots, T$ for each system. A uniform time-step makes next step easier but can be worked around for non-uniform data sets

3. Using the weight trajectory, use matrix least-squares with the equation $\widehat{A}[w_1, w_2, ..., w_{T-1}] = [w_2, w_3, ..., w_T]$ to solve for $\widehat{A}$.

4. To generate a multi-system model, concatenate the weight trajectories from each similar system in the least-squares computation of $\widehat{A}$. That is, let $W_\theta = [w_1^{(\theta)}, w_2^{(\theta)}, ..., w_{n-1}^{(\theta)}]$ and $W'_\theta = [w_2^{(\theta)}, w_3^{(\theta)}, ..., w_n^{(\theta)}]$ be the weight trajectory and next weight trajectory for some parameter . Then we solve the least-squares problem $\widehat{A} = [W_{\theta_1}, \ldots, W_{\theta_n}] = [W'_\theta, \ldots, W'_{\theta_n}]$

For the sake of defining when it is appropriate to expect our method to be able to generalize across different spatiotemporally evolving systems, we shall define what it means for two fluid flows to be *similar*. In configuring a fluid dynamics simulation, a set of quantifiable parameters are defined. Two dynamical fluid systems S_1 and S_2 are considered *similar* if they have the same configuration of parameters and differ only in the value of at most one parameter. Furthermore, we require that the parameter be continuously variable and that any observable

data point in the domain of the system vary smoothly as that parameter varies from its value in S_1 to its value in S_2. For example, for fluids flowing past identical cylinders, the Reynolds number associated with the free stream velocity may be varied to produce similar systems. However, to replace the system's cylinder with a triangle would be to qualitatively change the configuration of the system parameters, and thus would produce a non-similar system.

Unlike neural networks, the weights in an E-GP do not exist in some abstract, difficult-to-comprehend space, but are associated with kernel centers in specific locations in the domain. We refer to this attribute of E-GPs as the *spatial encoding* property. This property is an extremely valuable tool for gaining insight into the learned model works. For example, by plotting which kernel centers are associated with which invariant subspaces in the transition matrix, one can visualize where the eigenfunctions are found and how the dynamic modes are separated spatially. For another example, by plotting arrows from center c_j to c_i for each of the largest elements $\hat{a}_{ij}$ of $\widehat{A}$, one can visualize how different areas of the domain influence each other's evolution.

3.2 Theoretical Connection between Koopman & E-GP

Having formulated the Koopman operator as we did previously, we are prepared to examine why and how the eigenvalues and eigenvectors of the matrix $\widehat{A}$ can be related to the Koopman operator. In our system, the states are the mean functions f_τ which exist in the state space $\mathcal{H}$, which is an RKHS generated by the positive definite Mercer kernel k. The system operator $\mathcal{A}$ takes the place of the state transformation, and the measurement operator $\mathcal{K}$ is in essence an observable of the state. From this perspective, the finite-dimensional approximation of the Hilbert space is really a dimensionality-reducing convolution operator. Given that in the limit of centers used, $\widehat{\mathcal{H}}$ becomes dense in $\mathcal{H}$, we can identify every function with the set of weight vectors in $\mathbb{R}^M$. Since we still want the corresponding operators in approximate space, $\widehat{A}$ and K, to be linear, they become matrices. We can now write the Koopman operator U as a matrix as well, defined by

$$UK = K\widehat{A}$$

Supposing v is an eigenvector of $\widehat{A}$ with eigenvalue λ, $\widehat{A}v = \lambda v$, we find that

$$U(Kv) = K\widehat{A}v = \lambda Kv$$

which implies that Kv is a Koopman mode with eigenvalue λ.

3.3 Spectral Analysis of E-GP Model and Resulting Algorithms

One way of viewing the invariant subspaces concept (as described above) is say that information contained in an invariant subspace never leaves that invariant subspace. We hypothesize that the kernel centers associated with the invariant subspaces of a spatiotemporally evolving system are generally associated with spatial regions in the domain (and not just homogeneously spread all over and or mixed with the other invariant subspaces). This hypothesis makes sense both physically and mathematically. In physics, the *principle of locality* states that an object is only directly influenced by its immediate surroundings [**?**]. If this is the case (and there is no reason to believe that it is not, apart from certain quantum dynamics situations), and the E-GP model accurately captures the physics of the system, then information (measurable phenomena) may only travel continuously from one point in the domain to another. Mathematically, since a value at any one point in the domain is influenced by the weights of multiple nearby centers, we would expect nearby centers to be connected dynamically, unless separated by "plains" where the values are indistinguishable from noise.

When a square matrix is nicely formed, the Jordan form of a $n \times n$ matrix $\widehat{A}$ is block diagonal, and therefore gives a decomposition of the n dimensional Euclidean space into invariant subspaces of $\widehat{A}$. The cyclic index, which can be found by counting the geometric multiplicities of eigenvalues in $\widehat{A}$, gives the number of invariant subspaces. In reality, data-driven approximations of $\widehat{A}$ rarely give such easily interpretable properties. Hence, the need for an algorithm with divides invariant subspaces.

Each block in the Jordan normal form has a set of corresponding eigenvectors and an eigenvalue with geometric multiplicity. When we transform the former back into the domain space, we obtain complex-valued functions which are the Koopman modes of the system. These provide an image of what kind of structures we see in the dynamics. The eigenvalues describe the frequency with which these structures oscillate between their real and imaginary forms, as well as the exponential growth or decay of their magnitudes. In this paper, we demonstrate our method on systems with eigenvalues primarily on the unit circle (neither growing nor decaying).

3.3.1 k-Invariant Subspaces Clustering

The intuition behind our k-invariant subspaces clustering algorithm is to replace the Euclidean distance with a different metric of "nearness", namely one corresponding with the dynamic connections in the space. The $\widehat{A}$ matrix provides easy access to these: its rows $\widehat{A}_{i*}$ indicate which centers inform the i^{th} value of $w_{tindex+1}$, and its columns $\widehat{A}_{*j}$ indicate what centers will be informed by the j^{th} value of w_τ. However, in order to cluster without bias, we need to control for

the differing frequencies of the eigenmodes. If we do the eigen-decomposition of the matrix as $\widehat{A} = UDU^{-1}$, then let $\bar{D}$ be zeroed out except for its eigenvalues on the unit circle, and let $\bar{U}$ be truncated to only the corresponding eigenvectors. Then let $\bar{A} = UDU^{\dagger}$, where $U^{\dagger}$ is the pseudo-inverse. Much like the pairwise squared deviations of points formulation of the k-means clustering problem, we can now write our problem as

$$\arg\max_{S} \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{\substack{x_i \neq x_j \\ x_j, x_i \in S_i}} \bar{A}_{ij}^2$$

This problem can be solved with the following algorithm: Note the differences

Algorithm 1 k-Invariant Subspaces Algorithm

while clusters have changed **do**
  for each center i **do**
    find cluster k which maximizes the score

$$\frac{1}{|S_k| + 1} \left(\|\bar{A}_{i, S_k \setminus \{i\}}\|^2 + \|\bar{A}_{S_k \setminus \{i\}, i}\|^2 \right)$$

    reassign center i to cluster with highest score
  end for
end while
return clusters

between this and the k-means clustering algorithm: first, we are maximizing since the terms represent influnce rather than nearness. Secondly, we use $|S_k| + 1$ in the denominator to avoid division by zero. Thirdly, we exclude centers influence on themselves from the score by taking $S_k \setminus \{i\}$.

3.3.2 Scoring Paths

We have already described some of the important properties of the dynamics of the system. Now we discuss their relevance for path planning. We postulate that the value of taking a measurement at a point in the domain with respect to a particular eigenvector is proportional to the following factors:

1. The spatial extent of the eigenvector. This can be taken to be equivalent to the area under the curve of the magnitude of the complex function corresponding to that eigenvector (normalized by the function peak)

$$SE_i = \frac{1}{\sup_x v_i \hat{\Phi}(x)} \int_X |v_i \hat{\Phi}(x)| dx$$

2. The expected size of the measurement itself, which is roughly equal to $v_i \hat{\Phi}(x)$.

3. The ratio of the frequency with which that Koopman mode is visited to

the frequency of its eigenvalue. This factor is included due to the aliasing effect

4. A discount factor equal to a decaying exponential of the number of times that eigenmode has been visited

This scoring method allows one to decide the likeliest candidate amongst a large family of paths

3.3.3 Generating High-Scoring Random Paths

We propose the following algorithm for generating high-scoring paths:

1. Cluster the centers into invariant subspaces according to the k-invariant subspaces algorithm, with the best reasonable guess(es) for k

2. In each cluster, select a waypoint where the sum of all Koopman modes is maximized

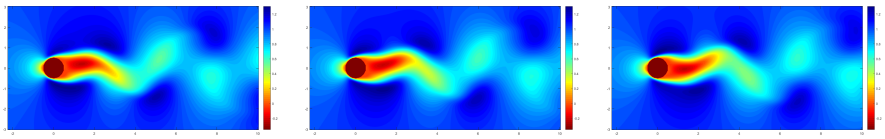
3. Generate a path by selecting waypoints randomly, where the selection is randomly weighted. The weights are equal to the spatial extent of the clusters multiplied by an exponential decay factor for the number of times each has been visited

Chapter 4

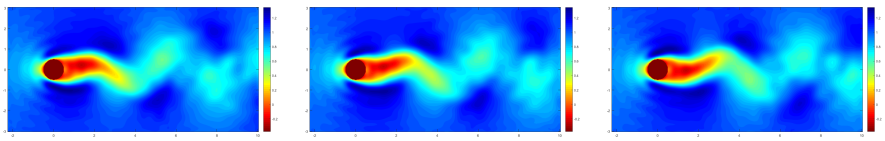
Experimental Results

4.1 Modeling the Individual Flows

We used CFD methods to generate the states for a canonical fluid mechanics problem: flow past a cylinder at various Reynolds numbers, namely Re = 100, 300, 600, 800 and 1000. This deterministic, high-dimensional spatiotemporal dynamical system is well-studied in the fluid dynamics literature, both experimentally and numerically [27, 2, 25]. In our CFD simulation, we used a 4th-order polynomial expansion with spectral element method on the incompressible Navier-Stokes equation to generate the cylinder flow data. The spatial domain is $[-2, 10] \times [-3, 3]$, excluding the diameter-1 cylinder at the origin. Neumann boundary conditions are applied to the far-field of the cylinder in the y-direction and the outlet of the flow field; and a Dirichlet boundary condition is applied to the inlet. Each data set contains at least 200 snapshots with a uniform time step of $\tilde{0}.03$ sec. Each snapshot contains 24,000 velocity data points for Re=100 or 95,000 velocity data points for Re=300,600,800,1000. Each data set took at least 10 hours in a high performance computer cluster to generate. Figures 4.1,4.2(a-c) visualize the horizontal velocity for Re=100 and Re=1000, with red being the greatest negative velocity and blue the greatest positive velocity. The flow is unstable, periodic, and clearly nonlinear.



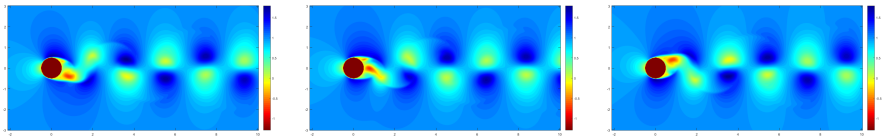
(a) Snapshot 0 (b) Snapshot 10 (c) Snapshot 20



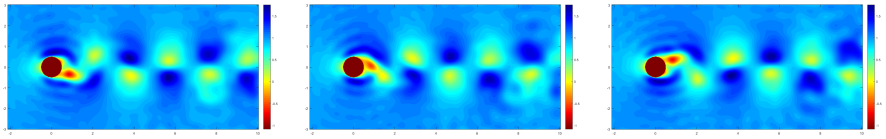
(d) Snapshot 0 (e) Snapshot 10 (f) Snapshot 20

Figure 4.1: Visualization of Fluid Flow at Re = 100, CFD (a-c), E-GP (d-f)

We used the Gaussian RBF kernel $k(x, y) = e^{-\|x-y\|^2/2\sigma^2}$ in our E-GP model,



(a) Snapshot 0 (b) Snapshot 5 (c) Snapshot 10



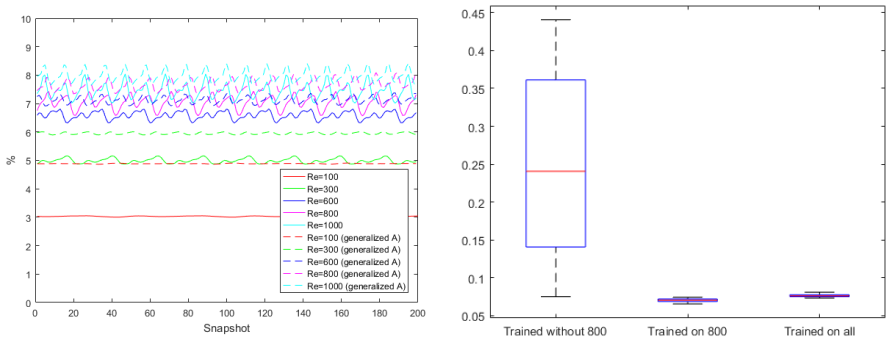
(d) Snapshot 0 (e) Snapshot 5 (f) Snapshot 10

Figure 4.2: Visualization of Fluid Flow at Re = 1000, CFD (a-c), E-GP (d-f)

with σ estimated to be 0.4. Using a budget of 600 kernel centers (see Figure 4.4(a)-4.4(b), and note how they cluster in the most dynamic regions), we find a 600×600 matrix $\widehat{A}$ which accurately (Figure 4.3(a)) captures the dynamics of the nonlinear system. We can use this to propagate single initial condition w_0 forward to make predictions, then compare the predictions to the original training data. We found total percentage errors between 3% for Re=100 and 7-8% for Re=1000, as can be seen in the solid lines in Figure 4.3(a). We define the total percentage errors as $E_\tau = \frac{\|y_\tau - \bar{y}_\tau\|_2}{\|\bar{y}_\tau\|_2}$ where $\bar{y}_\tau$ is the output vector for time τ and y_τ is the E-GP estimate at that time. Note that *the size of the model has been reduced by almost two orders of magnitude* from the original CFD data. This process takes about 13 minutes in MATLAB for a 200 snapshot by 95,000 point set on an ordinary Intel i7 4.00 GHz processor.

4.2 One Transition Matrix for Everything

In order to approach the challenge of generalizing across similar spatiotemporally evolving systems, the first question we had to answer is whether we can find an $\widehat{A}$ matrix that accurately captures the dynamics of multiple similar flows. The answer to that question is *yes*, using the trajectory concatenation method. Amazingly, a single model generated this way works almost as well on all five data sets as do five individual models trained on each data set separately. This is confirmed by both the total error plots (Figure 4.3(a)), which show only slight increases in each of the total percentage error plots, and visual inspection of the dynamic modes displayed. This result is even more surprising in light of the fact that the rate of vortex shedding for each Reynolds number is different. By taking a Fourier transform of the time evolution of a data point located at (0.5,8), we find that for the original data sets the vortex shedding frequency is 0.448 Hz, 1.260 Hz, 1.380 Hz, 1.388, and 1.401 Hz for Re=100, 300, 600, 800, and 1000 respectively, and for the E-GP models the frequencies are 0.452 Hz, 1.21 Hz, 1.36 Hz, 1.36 Hz, and 1.36 Hz respectively.



(a) Universal Generalizer vs Individual Models

(b) Different Models Tested on Re=800

Figure 4.3: Total Percentage Errors

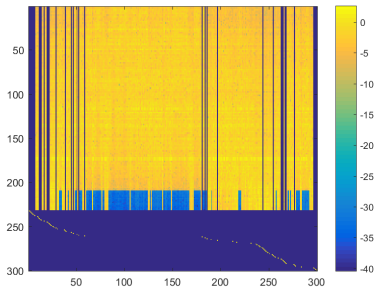
4.3 Generalizing from Learned Dynamics to Unknown Dynamics

Having seen that it is possible to find a single transition in the weight space that models the dynamics systems over a range of parameters, the next challenge is to be able to model flows with parameters that the model has not been trained on. We derived an $\widehat{A}$ matrix from the Re=100, 300, 600, and 1000 data sets and tested it against the Re=800 data set. The results are below in Figure 4.3(b). For the first 120 snapshots, the total percentage error remains under 10%, which is satisfactory. After this, however, the total percentage error curves upwards as the slight errors in the transition matrix compound. Over 800 snapshots, we found an average total percentage error of less than 25%.

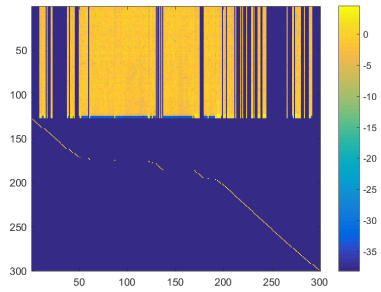
4.4 Linear Dynamical Layer Analysis & Insights

Due to the spatial encoding of the weights which the linear transition model operates on, we are able to analyze the dynamics and find physical insights into the process. We demonstrate two techniques: (1) using eigendecomosition of the transition matrix to discover the eigenfunctions and invariant subspaces of system, and (2) visualizing the most significant spatial interactions in the system.

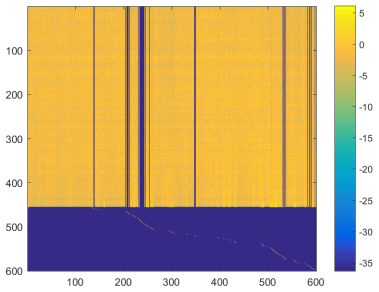
An invariant subspace of a linear operator is a subspace of the Hilbert space such that any vector in the subspace remains in the subspace under transformation by the operator. By marking which kernel centers are associated with different subspaces, we can spatially separate the space into multiple dynamic modules. The physical insight is some areas of the space are dynamically entangled with each other, and other are independent of each other. For those interested in monitoring spatiotemporally evolving systems, the number and



(a) Re = 100, $\varepsilon = 0.005$



(b) Re = 1000, $\varepsilon = 0.05$



(c) All Reynolds numbers, ε =0.069

Figure 4.4: Eigenvector Heat Maps

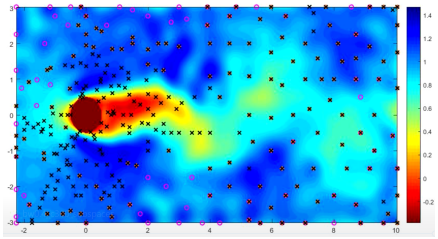
location of the invariant subspaces determines how many and where feedback sensors ought to be for robust prediction of the weights.

Before doing the Jordan decomposition of $\widehat{A}$, we zero any elements smaller than some small ε in order to stabilize the algorithm for matrices with many elements close to zero. Afterwards we visualize the eigenvector matrix using a *logarithmic* color chart, as seen in Figures 4.4(a),4.4(b),4.4(c). These plots are for models trained individually on Re=100 and Re=1000 with 300 kernels, and on all five with 600 kernels, for comparison. We see three categories of eigenvector in the rows: (1) Rows at the bottom that have exactly one non-zero elements, (2) In the middle, a couple rows with a dozen significant elements, and (3) at the top a number of rows that affect the majority of the kernel centers in the space.

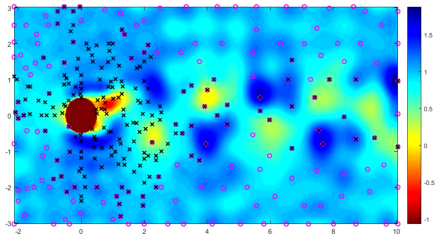
Each eigenvector of (1) spans its own invariant subspace, and is depicted in magenta circles in Figures 4.5(a),4.5(b),4.5(c). Category (3) is one invariant subspace, depicted with black crosses. Category (2) is subsumed in Category (3). The figures show that the dynamics near/around the cylinder and in its wake are so entangled that a single sensor measurement in that area may be sufficient to estimate over that entire subspace. On the other hand, areas far from the core of dynamic excitement are their own independent, invariant subspaces, and thus must be monitored locally.

Another way to visualize the operation of the linear transition matrix is to

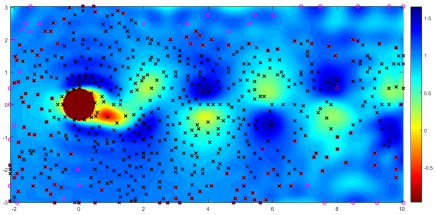
plot lines between kernel centers that are influencing each other strongly. That is, if we draw a line center c_j to c_i for each of the (relatively) largest elements a_{ij} of $\widehat{A}$, one can see how the system dynamics are coupled spatially (Figures 4.6(a),4.6(b),4.6(c)). We can also plot the magnitude of a_ij in a third axis for further insight into the most dominant dynamic connection in the system.



(a) Re = 100, $\varepsilon = 0.005$

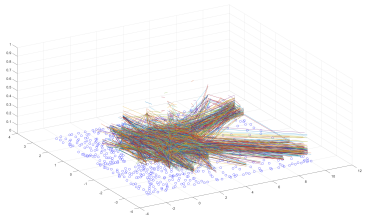


(b) Re = 1000, $\varepsilon = 0.05$

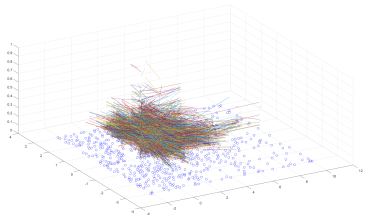


(c) All Reynolds numbers, $\varepsilon = 0.069$

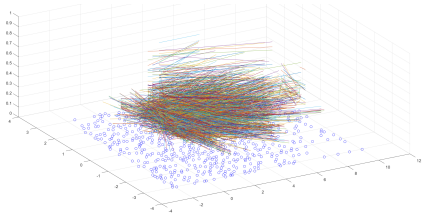
Figure 4.5: Invariant Subspaces



(a) Re=100

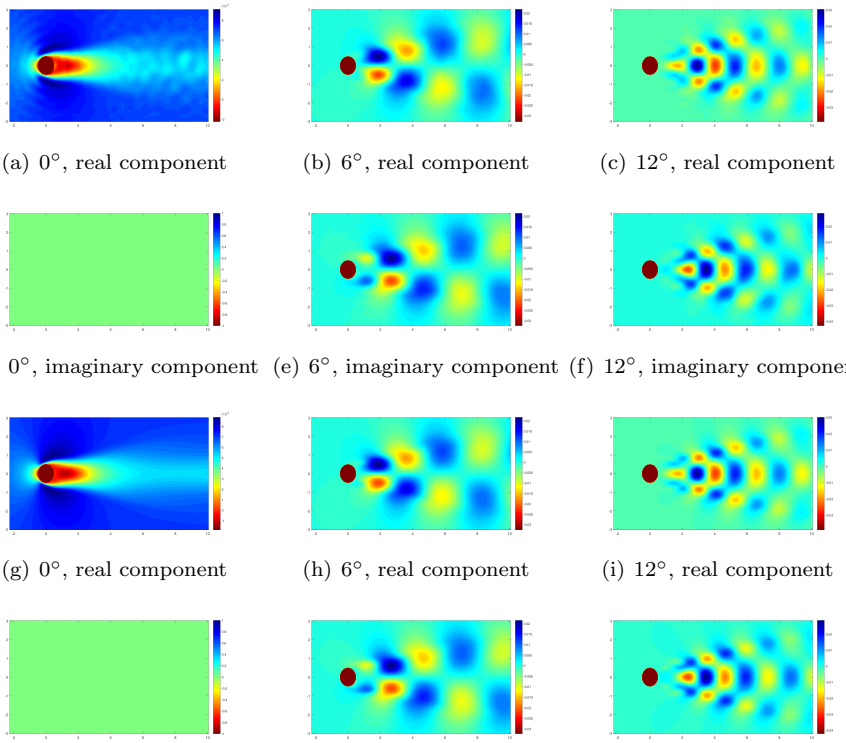


(b) Re=1000



(c) Trained on all 5 data sets

Figure 4.6: Visualization of Co-Relations in Transition Matrix



(a) 0°, real component (b) 6°, real component (c) 12°, real component

(d) 0°, imaginary component (e) 6°, imaginary component (f) 12°, imaginary component

(g) 0°, real component (h) 6°, real component (i) 12°, real component

(j) 0°, imaginary component (k) 6°, imaginary component (l) 12°, imaginary component

Figure 4.7: Primary Koopman Modes, Re=100. (a-f) E-GP, (g-l) DMD

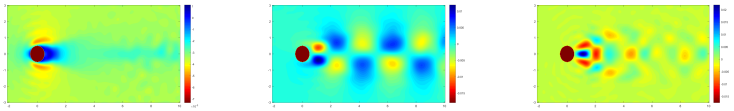
4.5 Koopman Modes & Eigenvalues

In order to derive the eigenvalues and Koopman modes of the system, one must simply take the eigenvalues and eigenvectors of the $\widehat{A}$ matrix, and transform the latter through the observation matrix K. As shown in figures 4.7 through 4.11, the eigenmodes corresponding with the eigenvectors of the $\widehat{A}$ matrix correspond exactly with the eigenmodes from the DMD method. As shown in 4.12, the eigenvalues also match along the unit circle – orange represents E-GP, blue DMD. The values inside the unit circle exponentially decay to zero.

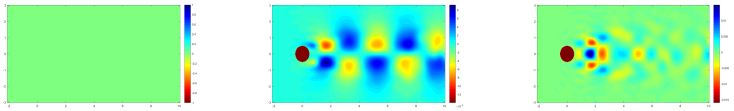
4.6 Mobile Path Planning & Inference

To begin our investigation into mobile path planning & inference, we constructed a number of simple synthetic spatiotemporally evolving systems. One of these is displayed below 4.13. This is a system with two invariant subspaces, in each of which, two hills oscillate up and down in sync at a different frequency than the other invariant subspace. This is ideal for testing our methods. In the example shown we used 100 centers with an RBF kernel with $\sigma = 0.3$, however in testing we also went as low as 20 centers.

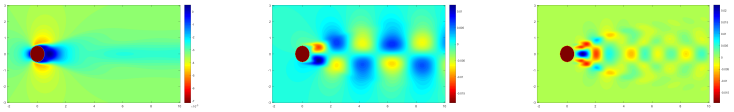
The k-means algorithm divides the system nicely through the middle, as



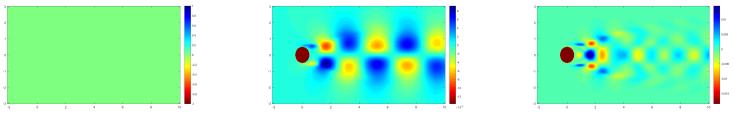
(a) 0°, real component (b) 15°, real component (c) 30°, real component



(d) 0°, imaginary component (e) 15°, imaginary component (f) 30°, imaginary component

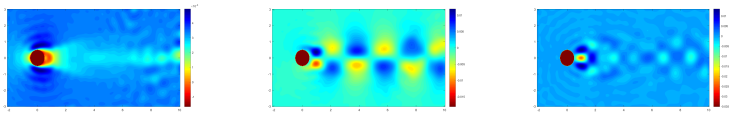


(g) 0°, real component (h) 15°, real component (i) 30°, real component

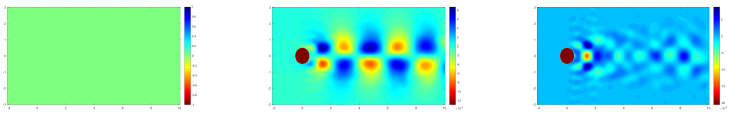


(j) 0°, imaginary component (k) 15°, imaginary component (l) 30°, imaginary component

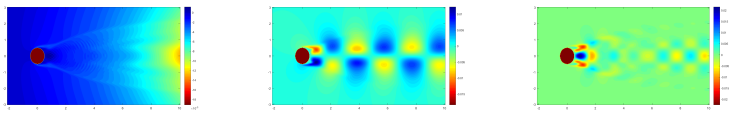
Figure 4.8: Primary Koopman Modes, Re=300. (a-f) E-GP, (g-l) DMD



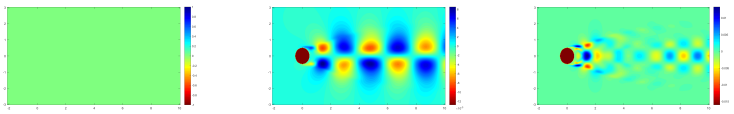
(a) 0°, real component (b) 16°, real component (c) 33°, real component



(d) 0°, imaginary component (e) 16°, imaginary component (f) 33°, imaginary component

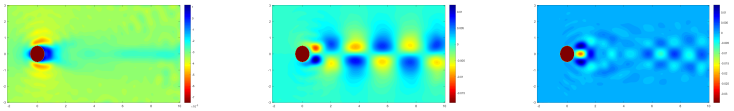


(g) 0°, real component (h) 16°, real component (i) 33°, real component

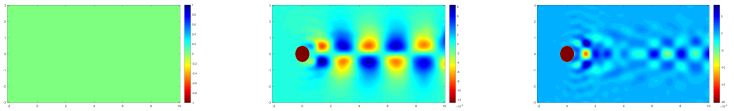


(j) 0°, imaginary component (k) 16°, imaginary component (l) 33°, imaginary component

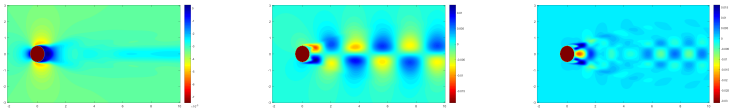
Figure 4.9: Primary Koopman Modes, Re=600. (a-f) E-GP, (g-l) DMD



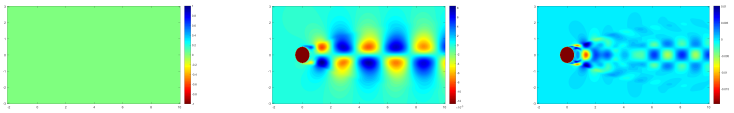
(a) 0°, real component (b) 17°, real component (c) 34°, real component



(d) 0°, imaginary component (e) 17°, imaginary component (f) 34°, imaginary component

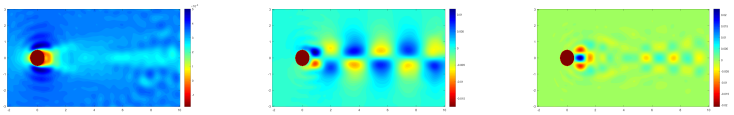


(g) 0°, real component (h) 17°, real component (i) 34°, real component

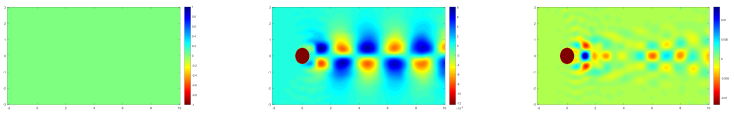


(j) 0°, imaginary component (k) 17°, imaginary component (l) 34°, imaginary component

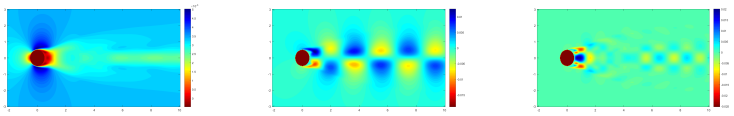
Figure 4.10: Primary Koopman Modes, Re=800. (a-f) E-GP, (g-l) DMD



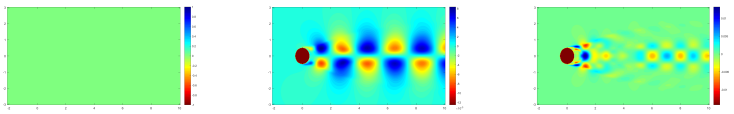
(a) 0°, real component (b) 17°, real component (c) 34°, real component



(d) 0°, imaginary component (e) 17°, imaginary component (f) 34°, imaginary component

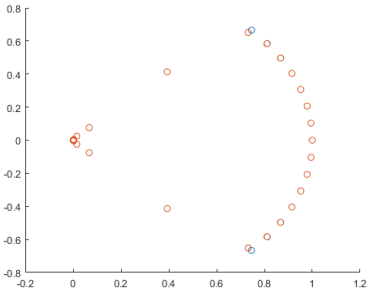


(g) 0°, real component (h) 17°, real component (i) 34°, real component

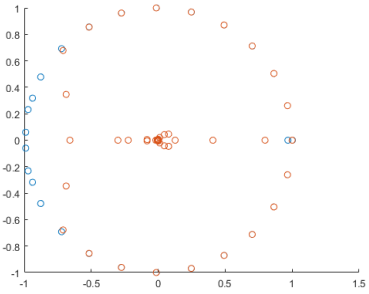


(j) 0°, imaginary component (k) 17°, imaginary component (l) 34°, imaginary component

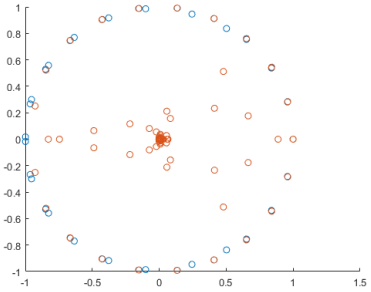
Figure 4.11: Primary Koopman Modes, Re=1000. (a-f) E-GP, (g-l) DMD



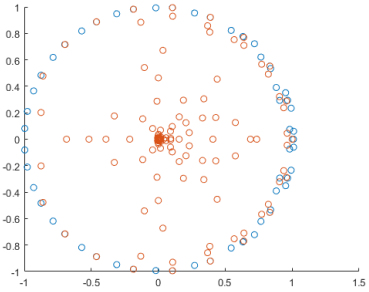
(a) Re=100



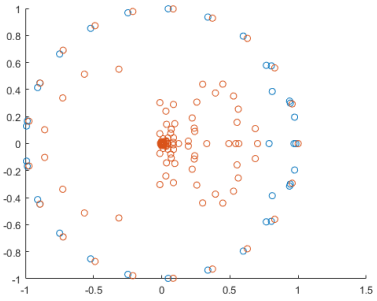
(b) Re=300



(c) Re=600



(d) Re=800



(e) Re=1000

Figure 4.12: Eigenvalue Comparison

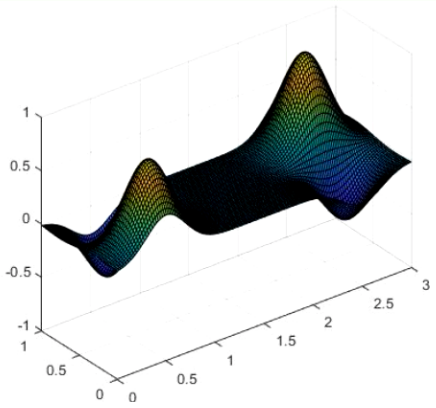
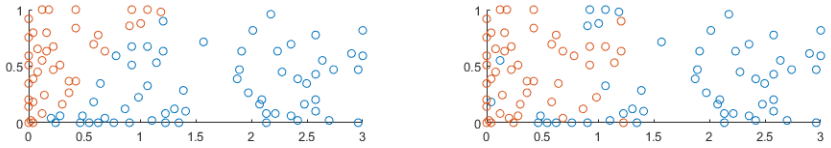


Figure 4.13: Simple Synthetic System with Two Invariant Subspaces



(a) Initial clustering of centers

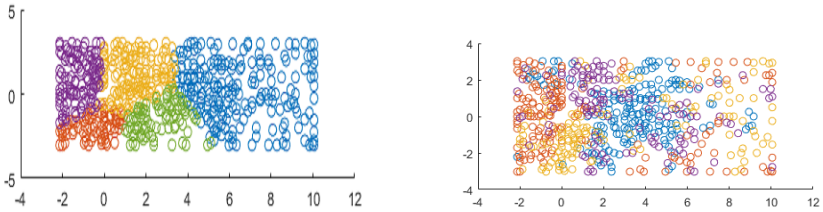
(b) Final clustering of centers

Figure 4.14: Synthetic System Invariant Subspaces

expected.

We found that few purely random paths would converge quickly for the synthetic data set, due to the large plain of zero values in between the two invariant subspaces. Our method for generating and selecting paths with waypoints resulted in paths that traveled between the four humps and quickly converged.

Since the CFD data set has dynamics almost everywhere, most paths are able to converge to the state. However, it is notable that a pre-designed lawnmower path failed to do as well as most random paths. The random paths generated and chosen by our methods did the best.



(a) Initial clustering of centers

(b) Final clustering of centers

Figure 4.15: CFD System Invariant Subspaces

Chapter 5

Conclusion

In this work, we presented a systems-theoretic approach to the problem of modeling complex spatiotemporally evolving phenomena and generalizing across continuously similar systems. Our approach focused on deriving a linear transition matrix in a space of weights layered over a kernel-based model. We also showed that spectral analyis of the linear transition matrix in the weight space of the model reveals connections to Koopman operator theory and provides key insights into the physics of the system, which may be used to design paths for mobile agents seeking to infer the state of the system. These methods were demonstrated on computational flow dynamics data of a fluid moving past a cylinder at various Reynolds numbers. We found that a single model could predict the evolution of the system at five very different Reynolds numbers with almost the same accuracy as a model of the same size trained on only one of the data sets. We found that our model was able to predict the evolution of similar systems that it had never been trained on. We found that the eigenvectors of the transition matrix, transformed into the input space, are identical to the Koopman modes found through Dynamic Mode Decomposition, which illustrate elegantly the dynamic structures which compose the wake and the vortices being shed from the cylinder. We demonstrated a new algorithm for determining the spatial regions associated with different invariant subspaces in linear transition matrix, and described how mobile agents can plan paths based on these results.

References

[1] Tim P Barnett, David W Pierce, and Reiner Schnur. Detection of anthropogenic climate change in the world's oceans. *Science*, 292(5515):270–274, 2001.

[2] Chassaing P.H.H.M. Braza, M. and H.H. Minh. Numerical study and physical analysis of the pressure and velocity fields in the near wake of a circular cylinder. *Journal of fluid mechanics*, 165(130), 1986.

[3] Haim Brezis. *Functional analysis, Sobolev spaces and partial differential equations*. Springer Science & Business Media, 2010.

[4] Marko Budivsic, Ryan Mohr, and Igor Mezic. Applied koopmanism. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(4):047510, 2012.

[5] et al Cao, Yi. Prediction of convergence dynamics of design performance using differential recurrent neural networks. In *IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, 2008.

[6] Andrea Carron, Marco Todescato, Ruggero Carli, Luca Schenato, and Gianluigi Pillonetto. Machine learning meets kalman filtering. In *Decision and Control (CDC), 2016 IEEE 55th Conference on*, pages 4594–4599. IEEE, 2016.

[7] Noel Cressie. *Statistics for spatial data*. John Wiley & Sons, 2015.

[8] Noel Cressie and Christopher K Wikle. *Statistics for spatio-temporal data*. John Wiley & Sons, 2011.

[9] Lehel Csatö and Manfred Opper. Sparse on-line gaussian processes. *Neural computation*, 14(3):641–668, 2002.

[10] Sahil Garg, Amarjeet Singh, and Fabio Ramos. Learning non-stationary space-time models for environmental monitoring. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, July 22-26, 2012, Toronto, Ontario, Canada.*, 2012.

[11] Wei Li Guo, Xiaoxiao and Francesco Iorio. Convolutional neural networks for steady flow approximation. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016.

[12] Matthew J Heaton, Matthias Katzfuss, Shahla Ramachandar, Kathryn Pedings, Eric Gilleland, Elizabeth Mannshardt-Shamseldin, and Richard L Smith. Spatio-temporal models for large-scale indicators of extreme weather. *Environmetrics*, 22(3):294–303, 2011.

[13] David Higdon. A process-convolution approach to modelling temperatures in the north atlantic ocean. *Environmental and Ecological Statistics*, 5(2):173–190, 1998.

[14] H. A. Kingravi, H. Maske, and G. Chowdhary. Kernel controllers: A systems-theoretic approach for data-driven modeling and control of spatiotemporally evolving processes. In *2015 54th IEEE Conference on Decision and Control (CDC)*, pages 7365–7370, Dec 2015.

[15] Hassan Kingravi, Harshal Maske, and Girish Chowdhary. Kernel observers: Systems theoretic modeling and inference of spatiotemporally varying processes. In *Advances in Neural Information Processing Systems (NIPS)*, Barcelona, Spain, 2016. accepted.

[16] Quoc Le, Tamás Sarlós, and Alex Smola. Fastfood-approximating kernel expansions in loglinear time. In *Proceedings of the International Conference on Machine Learning*, 2013.

[17] James MacQueen et al. Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, pages 281–297. Oakland, CA, USA, 1967.

[18] Igor Mezić. Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics*, 41(1-3):309–325, 2005.

[19] Igor Mezić and Andrzej Banaszuk. Comparison of systems with complex behavior. *Physica D: Nonlinear Phenomena*, 197(1-2):101–133, 2004.

[20] C Paciorek and M Schervish. Nonstationary covariance functions for gaussian process regression. *Advances in neural information processing systems*, 16:273–280, 2004.

[21] Ronald L Panton. *Incompressible flow*. John Wiley & Sons, 2006.

[22] Tobias Pfingsten, Malte Kuss, and Carl Edward Rasmussen. Nonstationary gaussian process regression using a latent extension of the input space, 2006.

[23] Christian Plagemann, Kristian Kersting, and Wolfram Burgard. Nonstationary gaussian process regression using point estimates of local smoothness. In *Machine learning and knowledge discovery in databases*, pages 204–219. Springer, 2008.

[24] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *NIPS*, pages 1177–1184, 2007.

[25] Kandasamy A. Rajani, B.N. and S. Majumdar. Numerical simulation of laminar flow past a circular cylinder. *Applied Mathematical Modelling*, 33(3):1228–1247, 2009.

[26] C.E. Rasmussen and C.K.I. Williams. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005.

[27] A. Roshko. On the development of turbulent wakes from vortex streets. *California Institute of Technology*, Report 1191, 1954.

[28] Clarence W Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, and Dan S Henningson. Spectral analysis of nonlinear flows. *Journal of fluid mechanics*, 641:115–127, 2009.

[29] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

[30] Alexandra M Schmidt and Anthony O'Hagan. Bayesian inference for non-stationary spatial covariance structure via spatial deformations. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(3):743–758, 2003.

[31] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[32] Jonathan Tompson, Kristofer Schlachter, Pablo Sprechmann, and Ken Perlin. Accelerating eulerian fluid simulation with convolutional networks. *arXiv preprint arXiv:1607.03597*, 2016.

[33] Christopher K Wikle. A kernel-based spectral model for non-gaussian spatio-temporal processes. *Statistical Modelling*, 2(4):299–314, 2002.

[34] Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In *NIPS*, pages 682–688, 2001.

[35] Clarence W. Rowley Williams, Matthew O. and Ioannis G. Kevrekidis. A kernel-based method for data-driven koopman spectral analysis. *Journal of Computational Dynamics*, 2(2), 2015.

[36] Ioannis G. Kevrekidis Williams, Matthew O. and Clarence W. Rowley. A datadriven approximation of the koopman operator: Extending dynamic mode decomposition. *Journal of Nonlinear Science*, 25(6):1307–1346, 2015.

[37] Zichao Yang, Andrew Wilson, Alex Smola, and Le Song. {A la Carte–Learning Fast Kernels}. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pages 1098–1106, 2015.