Evolving Gaussian Processes for Learning and Control

Tutorial with examples

Harshal Maske, Hassan Kingravi, Joshua Whitman, Girish Chowdhary,
POC email: girishc@illinois.edu

Introduction

Monitoring and modeling of large-scale stochastic phenomena with both spatial and temporal (spatiotemporal) evolution using a network of distributed sensors is a fundamental problem in the applied sciences. Consider for example a team of mechanical weeding robots managing herbicide resistant weeds on a farm (see Figure 1). This team of robots needs to predict the weed growth across the whole farm in real time to make intelligent decisions on robot coordination [1]. With rapid advances in robotics and the capability to process large volumes of data on compact computational packages, the applications for such distributed Cyber Physical Systems are rapidly expanding. Common examples include modeling and monitoring ocean heat content and acidification in oceanography using a network of satellites and surface sensors; prediction of traffic patterns using data from vehicles, cellphones, and traffic cameras; prediction of enemy movements through ground and aerial surveillance, and prediction of spatiotemporal evolution of extreme weather events using data from weather stations and aerial drones.

These types of problems are characterized by complex and stochastic dynamics that are distributed in space and time. While modeling such spatiotemporal phenomena has traditionally been the province of the field of geostatistics, it has in recent years gained more attention in the machine learning community [2]. The data-driven models developed through machine learning techniques provide a way to capture complex spatiotemporal phenomena which are not easily modeled by first-principles alone. However, machine learning models are limited by the data sets they are trained on, and the challenge with complex and distributed physical systems is that they have very high amounts of variability. For example, a model trained on weed growth over years of data in one field doesn't necessarily generalize to another, and a model trained on past few year's worth of data is still unreliable to predict weather variability in the following year. What is needed instead of just a *predict* system is a *predict-and-correct* system, we elucidate this point below in more detail.

In the machine learning community, kernel methods represent a class of extremely well-studied and powerful methods for regression in spatial domains. In these techniques, correlations between the input variables are encoded through a covariance kernels, and the model is formed through a linear weighted combination of the kernels [3]–[5]. In recent years, kernel methods have been applied to spatiotemporal regression problems with varying degrees of success [2], [3]. Many recent techniques in spatiotemporal modeling have focused on nonstationary covariance kernel design and associated hyperparameter learning algorithms [6]–[8]. These methods, which focus on the careful design of covariance kernels, have been proposed as an alternative to the naive approach of simply including time as an additional input variable in the kernel [9]. The careful design/optimization of covariance kernel avoids an explosion in the number of parameters (kernels utilized) of the model which would be inevitable in a model that simply adds time as an additional input variable, and has been shown to better account for spatiotemporal couplings.

However, there are two key challenges with existing kernel based, and for that matter, other machine learning approaches for prediction over complex spatiotemporally varying physical systems: The first is ensuring the scalability of the model to large scale phenomena, which manifests due to the fact that the problem of optimizing the covariance kernel (known as hyperparameter optimization in the ML community) is not convex in general, leading to methods that are difficult to implement especially in online settings, susceptible to getting stuck at local minima, and highly computationally demanding for large datasets [10]. This has been a key issue also in utilizing deep learning models for this problem. The second key challenge is that no matter how much data the model is trained on, it cannot completely capture the variability in the system. What is needed instead is a way to inform how sensing should be done, but current ML models do not provide such insight. In particular, there is not much literature providing insights onto how existing kernel-based machine learning models could be used for analysis and synthesis of observers and controllers for the large scale spatiotemporal phenoemena. While the first challenge can be addressed with increasing computational power and large datasets, addressing the latter (and vastly more fundamental) challenge is particularly important in the design of reliable engineering systems, such as distributed sensor/actuator networks intended for monitoring physical phenomena, autonomous soft-robots, or other physical systems with distributed sensing and actuation.

Contributions

In this article, we present a new perspective to solving the spatiotemporal monitoring problem that brings together kernel-based modeling, systems theory, and Bayesian filtering. We define the monitoring problem as follows: *Given an approximate predictive model of the spatiotemporal phenomena learned using historic data, estimate the current latent state of the*

phenomena in the presence of uncertainty using as few sensors as possible. Ideally, the solution to the problem should also provide guidance on how many sensors are needed and where to place them. In this paper we argue that when it comes to predictive inference over spatiotemporal phenomena, a Kalman-filter type approach of predicting and correcting with feedback from a set of minimal sensors is a robust way of dealing with real-world uncertainties and inherent modeling errors. In the context of this specific problem, our *main contributions are two-fold*: first, we demonstrate that spatiotemporal functional evolution can be modeled using stationary kernels with a linear dynamical systems layer on their mixing weights. In particular, in contrast with existing work, this approach does not necessarily require the design of complex spatiotemporal kernels, and can accommodate positive-definite kernels on any domain on which it is possible to define them, which includes non-Euclidean domains such as Riemannian manifolds, strings, graphs and images [11]. Second, we show that such a model can be utilized to determine sensing locations that guarantee that the hidden states of functional evolution can be estimated using a Bayesian state-estimator (Kalman filter) that is *embedded in the feature space of the kernel model* with very few sensors. A benefit of our solution approach is that it provide guidance on how many sensors are needed and where to place them. Accordingly, we provide sufficient conditions on the number and location of sensor measurements required and prove non-conservative lower bounds on the minimum number of sampling locations by developing fundamental results on observability of kernel based models. The validity of the presented model and sensing techniques is corroborated using synthetic and large real datasets.

Broader Context

The fundamental idea of building observers and controllers embedded in the feature spaces of machine learning models introduced in this paper is generalizable beyond the particular application of spatiotemporal monitoring. Indeed, the contributions of this paper demonstrate how machine learning machine learning theory can be fused with systems theory to address major challenges in distributed monitoring and control of complex spatiotemporally varying systems. We elucidate this point below.

Traditionally, the controls literature is strongest when the system dynamics can be represented as ODEs. As depicted in Figure 2, some of the major successes of controls have included results such as LQG, reinforcement learning, and adaptive control in state spaces with well-defined, finite, and physically meaningful state variables. The problem of state estimation of a temporally evolving, finite-dimensional state-space system for example has been extensively studied in the context of Kalman filtering and observer design [12]. Here, fundamental results in observability/controllability provide sufficient conditions on the structure of the state transition and measurement matrix such that the latent state can be estimated in the presence

of measurement and process noise. On the other hand, machine learning models operate in an abstract and learned function-space of *features* (see sidebar). Kalman filters can be naively extended to the functional domain (e.g. [13]), but have not typically been studied in context of the spatiotemporal monitoring problem studied here.

On the other hand, recent advances in machine learning are providing different and highly powerful ways of modeling complex spatiotemporal dynamical systems. Yet, the main challenge with using machine learning approaches such as deep learning or kernel based methods has been that these approaches lack interpretablity and analyzability, which makes it difficult to design robust engineered systems. This is mostly because machine learning works in abstract feature spaces that are only relatable to physical quantities through complex functional operations (see Sidebar *Feature Spaces in Machine Learning* for discussion on feature spaces in machine learning). This leads to a major challenge in using machine learning models in control: For example, when kernel based models are used for spatiotemporal systems, how does one answer fundamental questions such as 1) the least number of sensors required to observe a distributed system, 2) the placement of sensors/actuators to guarantee observability/controllability of the system, and 3) the effect of random sensor placement on system observability/controllability.

In this paper we present an approach that can provide one formal way of addressing these and other questions about complex systems that are modeled with machine learning. In particular, we demonstrate how linear dynamical systems can be embedded in the reproducing kernel Hilbert space (RKHS) [4], [14], [15] generated by features used in Gaussian Process modeling, and utilized to answer fundamental questions such as controllability and observability. This marriage of systems theory with machine learning pursued in this paper is exciting, and we expect that follow-up work will exploit the framework presented in this paper of utilizing linear models in RKHSs and feature spaces of other machine learning models to enable practical and analyzable data-driven engineering systems. To facilitate the development of the theory, we have focused this paper on the problem of monitoring spatiotemporal phenomena. However, the idea should be generalizable to any distributed cyber-physical system that is changing with space and time.

We begin the rest of the article by summarizing some related work in machine learning in this area. Sidebar *Feature Spaces in Machine Learning* outlines feature spaces in machine learning in a broader context. We then formulate the problem, introduce *Kernel Observers (KO)*, and develop the main theoretical and algorithmic results. This is followed by some results on the expected number of randomly placed sensors required to monitor a spatiotemporal process in the context of our model. An extension to the KO method called *Evolving Gaussian Processes (E-GP)* is the presented that learns one model for multiple, similar spatiotemporal processes, the efficacy of which on real-world CFD data is presented in Sidebar *Learning Fluid Flows with*

Evolving Gaussian Processes. The paper is then concluded. Elements of the work presented in this paper first appeared in the premier machine learning conference Neural Information Processing Systems (NIPS 2016) ([16], [17]), IEEE CDC 2015 conference [18], the Conference on Robot Learning (CoRL 2017) [19] and IEEE ACC 2018 conference [20]. This paper presents a comprehensive set of results and fills in the missing links in a single encompassing publication, and introduces new results on observability in the presence of random sensor placement. As such, we have focused in this article mostly on the fundamental theory and practical algorithms for modeling, estimation, and control, while the excruciating details of how to optimally implement the presented algorithms are omitted[1].

Related Work

There is a large body of literature on spatiotemporal modeling in geostatistics where specific process dependent kernels can be used [2], [21]. From the machine learning perspective, a naive approach is to utilize both spatial and temporal variables as inputs to a Mercer kernel [22]. However, this technique leads to an ever-growing kernel dictionary. Furthermore, constraining the dictionary size or utilizing a moving window will occlude learning of long-term patterns. Periodic or nonstationary covariance functions and nonlinear transformations have been proposed to address this issue [3], [7]. Work focusing on nonseparable and nonstationary covariance kernels seeks to design kernels optimized for environment-specific dynamics, and to tune their hyperparameters in local regions of the input space. Seminal work in [23] proposes a process convolution approach for space-time modeling. This model captures nonstationary structure by allowing the convolution kernel to vary across the input space. This approach can be extended to a class of nonstationary covariance functions, thereby allowing the use of a Gaussian process (GP) framework, as shown in [24]. However, since this model's hyperparameters are inferred using MCMC integration, its application has been limited to smaller datasets. To overcome this limitation, [8] proposes to use the mean estimates of a second isotropic GP (defined over latent length scales) to parameterize the nonstationary covariances. Finally, [6] considers nonistropic variation across different dimension of input space for the second GP as opposed to isotropic variation by [8]. Issues with this line of approach include the nonconvexity of the hyperparameter optimization problem and the fact that selection of an appropriate nonstationary covariance function for the task at hand is a nontrivial design decision (as noted in [25]).

Apart from directly modeling the covariance function using additional latent GPs, there exist several other approaches for specifying nonstationary GP models. One approach maps the

[1]Instead an open-source code-base is made available in MATLAB at http://daslab.illinois.edu/software.html and in Python on GitHub at https://github.com/hkingravi/funcobspy.

nonstationary spatial process into a latent space, in which the problem becomes approximately stationary [26]. Along similar lines, [27] extends the input space by adding latent variables, which allows the model to capture nonstationarity in original space. Both these approaches require MCMC sampling for inference, and as such are subject to the limitations mentioned in the preceding paragraph.

A geostatistics approach that finds dynamical transition models on the linear combination of weights of a parameterized model [2], [13] is advantageous when the spatial and temporal dynamics are hierarchically separated, leading to a convex learning problem. As a result complex nonstationary kernels are often not necessary (although they can be accommodated). The approach presented in this paper aligns closely with this vein of work. A systems-theoretic study of this viewpoint enables the fundamental contributions of the paper, which are 1) allowing for inference on more general domains with a larger class of basis functions than those typically considered in the geostatistics community, and 2) quantifying the minimum number of measurements required to estimate the state of functional evolution.

Lastly, sensor placement optimization is also a well-studied area. Examples include, but are not limited to 1) geometric approaches, which seek to provide a covering of the operating space without making assumptions about the spatiotemporal dynamics [28], and 2) information-theoretic approaches, which place their focus on sensor placement optimizing strategies based on mutual information and information entropy for Gaussian process models [29]. It should be noted that the contribution of the paper concerning sensor placement is to provide *sufficient conditions* for monitoring rather than optimization of the placement locations, and therefore a comparison with these approaches is not considered in the experiments.

Kernel Observers

This section outlines our modeling framework and presents theoretical results associated with the number of sampling locations required for monitoring functional evolution.

Problem Formulation

We focus on predictive inference of a time-varying stochastic process, whose mean f evolves temporally 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. Our approach builds on the fact that in several cases, temporal evolution can be hierarchically separated from spatial functional evolution. A classical and quite general example of this is the *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 $u \in \mathcal{B}$ [30]. This model of spatiotemporal

evolution is very general (AEOs, for example, model many PDEs), but working in Banach spaces can be computationally taxing. A simple way to make the approach computationally realizable is to place restrictions on $\mathcal{B}$: in particular, we restrict the sequence f_τ to lie in a reproducing kernel Hilbert space (RKHS), the theory of which provides powerful tools for generating flexible classes of functions with relative ease [3]. In a kernel-based model, $k : \Omega \times \Omega \to \mathbb{R}$ is a positive-definite Mercer kernel on a domain Ω that models the covariance between any two points in the input space, and 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 key insight behind the proposed model is that spatiotemporal evolution in the input domain corresponds to temporal evolution of the mixing weights of a kernel model alone in the functional domain. Therefore, f_τ can be modeled by tracing the evolution of its mean embedded in a RKHS using switched ordinary differential equations (ODE) when the evolution is continuous, or switched difference equations when it is discrete (Figure 3). The advantage of this approach is that it allows us to utilize powerful ideas from systems theory for deriving necessary and sufficient conditions for spatiotemporal monitoring.

In this paper, we restrict our attention to the class of functional evolutions $\mathbb{F}$ defined by linear Markovian transitions in an RKHS. While extension to the nonlinear case is possible (and non-trivial), it is not pursued in this paper to help ease the exposition of the key ideas. The class of linear transitions in RKHS is rich enough to approximately model many real-world datasets, as suggested by our experiments.

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:

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

where η_τ is a zero-mean stochastic process in $\mathcal{H}$, and ζ_τ is a Wiener process in $\mathbb{R}^N$. Classical treatments of kernel methods emphasize that for most kernels, the feature map ψ is unknown, and possibly infinite-dimensional; this forces practioners to work in the dual space of $\mathcal{H}$, whose dimensionality is the number of samples in the dataset being modeled. This conventional wisdom precludes the use of kernel methods for most tasks involving modern datasets, which may have millions and sometimes billions of samples [31]. An alternative is to work with a feature map $\widehat{\psi}(x) := \left[\widehat{\psi}_1(x) \cdots \widehat{\psi}_M(x) \right]^T$ to an approximate feature space $\widehat{\mathcal{H}}$, with the property that for every element $f \in \mathcal{H}$, $\exists \widehat{f} \in \widehat{\mathcal{H}}$ and an $\epsilon > 0$ s.t. $\|f - \widehat{f}\| < \epsilon$ for an appropriate function norm. A few such approximations are listed below.

Dictionary of atoms: Let Ω be compact. Given points $\mathcal{C} = \{c_1, \ldots, c_M\}$, $c_i \in \Omega$, we have a 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. Here,

$$\widehat{\psi}_i(x) := \langle \psi(x), \psi(c_i) \rangle_{\mathcal{H}} = k(x, c_i). \tag{2}$$

Low-rank approximations: Let Ω be compact, let $\mathcal{C} = \{c_1, \ldots, c_M\}$, $c_i \in \Omega$, and let $K \in \mathbb{R}^{M \times M}$, $K_{ij} := k(c_i, c_j)$ be the Gram matrix computed from $\mathcal{C}$. This matrix can be diagonalized to compute approximations $(\widehat{\lambda}_i, \widehat{\phi}_i(x))$ of the eigenvalues and eigenfunctions $(\lambda_i, \phi_i(x))$ of the kernel [32]. These spectral quantities can then be used to compute $\widehat{\psi}_i(x) := \sqrt{\widehat{\lambda}_i}\widehat{\phi}_i(x)$.

Random Fourier features: Let $\Omega \subset \mathbb{R}^n$ be compact, and let $k(x, y) = e^{-\|x-y\|^2/2\sigma^2}$ be the Gaussian RBF kernel. Then random Fourier features approximate the kernel feature map as $\widehat{\psi}_\omega : \Omega \to \widehat{\mathcal{H}}$, where ω is a sample from the Fourier transform of $k(x, y)$, with the property that $k(x, y) = \mathbb{E}_\omega[\langle \widehat{\psi}_\omega(x), \widehat{\psi}_\omega(y) \rangle_{\widehat{\mathcal{H}}}]$ [31]. In this case, if $V \in \mathbb{R}^{M/2 \times n}$ is a random matrix representing the sample ω, then $\widehat{\psi}_i(x) := \left[\frac{1}{\sqrt{M}} \sin([Vx]_i), \frac{1}{\sqrt{M}} \cos([Vx]_i) \right]$. Similar approximations exist for other radially symmetric kernels, as well as dot-product kernels.

In the approximate space case, we replace the transition operator $\mathcal{A} : \mathcal{H} \to \mathcal{H}$ in (1) by $\widehat{\mathcal{A}} : \widehat{\mathcal{H}} \to \widehat{\mathcal{H}}$. This approximate regime, which combines the flexibility of a truly nonparametric approach with computational realizability, still allows for the representation of rich phenomena, as will be seen in the sequel, and in Figure 5. The finite-dimensional evolution equations approximating (1) in dual form are

$$w_{\tau+1} = \widehat{A} w_\tau + \eta_\tau, \quad y_\tau = K w_\tau + \zeta_\tau, \tag{3}$$

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 $K_{(i)} = \widehat{\Psi}(x_i) = \left[\widehat{\psi}_1(x_i) \ \widehat{\psi}_2(x_i) \ \cdots \ \widehat{\psi}_M(x_i) \right]$. In systems-theoretic language, each row of K corresponds to a *measurement* at a particular location, and the matrix itself acts as a measurement operator.

The equations (1) suggest an immediate extension to functional control problems. Pick another basis for $\mathcal{H}$ as $\widetilde{\psi}(x) := \begin{bmatrix} \widetilde{\psi}_1(x) & \cdots & \widetilde{\psi}_{\ell'}(x) \end{bmatrix}^T$, where the functions $\widetilde{\psi}_j(x)$ are used to approximate the RKHS $\mathcal{H}$ generated by the kernel. We denote the span of these functions as $\widetilde{\mathcal{H}}$. In the dictionary of atoms case, an example would be another set of atoms $\mathcal{F}_D = \begin{bmatrix} \psi(d_1) & \cdots & \psi(d_{\ell'}) \end{bmatrix}$, $\psi(d_j) \in \mathcal{H}$, $d_j \in \Omega$, with $\widetilde{\mathcal{H}}$ being a strict subspace of the RKHS $\mathcal{H}$ generated by the kernel. The functional evolution equation is then as follows:

$$f_{\tau+1} = \mathcal{A} f_\tau + \mathcal{B} \delta_\tau + \eta_\tau, \quad y_\tau = \mathcal{K}_\tau f_\tau + \zeta_\tau, \tag{4}$$

where the control functions δ_τ evolve in $\widetilde{\mathcal{H}}$, and $\mathcal{B} : \widetilde{\mathcal{H}} \to \widehat{\mathcal{H}}$. To derive the finite-dimensional equivalent of $\mathcal{B}$, we have to work out the structure of the matrix B: since $\widehat{\mathcal{H}}$ is not, in general,

isomorphic to $\widetilde{\mathcal{H}}$, this imposes strict restrictions on B. We can derive B using least squares using the inner product of $\mathcal{H}$. An instructive example is where both $\widehat{\mathcal{H}}$ and $\widetilde{\mathcal{H}}$ are generated by dictionaries of atoms; recall that in this case, $\mathcal{F}^{\mathcal{C}} = \begin{bmatrix}\psi(c_1) & \cdots & \psi(c_M)\end{bmatrix}$ is the basis for $\widehat{\mathcal{H}}$, and let $\delta = \sum_{j=1}^{\ell'} \acute{w}_j \psi(d_j)$, and let $\mathcal{F}^{\mathcal{C}} = \begin{bmatrix}\psi(c_1) & \cdots & \psi(c_M)\end{bmatrix}$ be the basis for $\mathcal{H}^{\mathcal{C}}$. Then the projection of δ onto $\widehat{\mathcal{H}}$ can be derived as

$$\begin{bmatrix}\langle \delta, \psi(c_1)\rangle_{\mathcal{H}} \\ \vdots \\ \langle \delta, \psi(c_M)\rangle_{\mathcal{H}}\end{bmatrix} = \underbrace{\begin{bmatrix}\langle \psi(d_1), \psi(c_1)\rangle_{\mathcal{H}} & \cdots & \langle \psi(d_{\ell'}), \psi(c_1)\rangle_{\mathcal{H}} \\ \vdots & \ddots & \vdots \\ \langle \psi(d_1), \psi(c_M)\rangle_{\mathcal{H}} & \cdots & \langle \psi(d_{\ell'}), \psi(c_M)\rangle_{\mathcal{H}}\end{bmatrix}}_{K_{CD}} \begin{bmatrix}\acute{w}_1 \\ \vdots \\ \acute{w}_{\ell'}\end{bmatrix}. \tag{5}$$

Note that in the dictionary of atoms case, the entries of K_{CD} can be computed in closed form as $K_{CDij} := k(d_i, c_j)$, using the reproducing property. This derivation shows that the operator B is simply $K_{CD} \in \mathbb{R}^{M \times \ell'}$, the kernel matrix between the data C generating the atoms $\mathcal{F}^{\mathcal{C}}$ of $\widehat{\mathcal{H}}$ and the data D generating the atoms $\mathcal{F}_D$ of $\widetilde{\mathcal{H}}$. Thus, the finite-dimensional evolution equations equivalent to (4) are

$$w_\tau = \widehat{A} w_\tau + K_{CD} \acute{w}_\tau, \quad y_\tau = K_\tau w_\tau. \tag{6}$$

We define the *generalized observability matrix* [33] as $\mathcal{O}_\Upsilon = \begin{bmatrix} K\widehat{A}^{\tau_1} \\ \cdots \\ K\widehat{A}^{\tau_L}\end{bmatrix}$ where $\Upsilon = \{\tau_1, \ldots, \tau_L\}$ are the set of instances τ_i when we apply the operator K. A linear system is said to be *observable* if $\mathcal{O}_\Upsilon$ has full column rank (i.e. $\mathrm{Rank}(\mathcal{O}_\Upsilon) = M$) for $\Upsilon = \{0, 1, \ldots, M-1\}$ [33]. Observability guarantees two critical facts: firstly, it guarantees that the state w_0 can be recovered exactly from a finite series of measurements $\{y_{\tau_1}, y_{\tau_2}, \ldots, y_{\tau_L}\}$; in particular, defining $y_\Upsilon = \begin{bmatrix} y_{\tau_1}^T, y_{\tau_2}^T, \cdots, y_{\tau_L}^T\end{bmatrix}^T$, we have that $y_\Upsilon = \mathcal{O}_\Upsilon w_0$. Secondly, it guarantees that a feedback based *observer* can be designed such that the estimate of w_τ, denoted by $\widehat{w}_\tau$, converges exponentially fast to w_τ in the limit of samples. Note that all our theoretical results assume $\widehat{A}$ is available: while we perform system identification in the experiments ([16]), it is not the focus of the paper.

We are now in a position to formally state the spatiotemporal modeling, control, and inference problems being considered: given a spatiotemporally evolving system modeled using (3), 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*), can be predicted robustly (which corresponds to *Bayesian filtering*), and which can be controlled (which corresponds to *functional control*). Our approach to solve the monitoring and prediction problem relies on the design of the measurement operator K so that the pair $(K, \widehat{A})$ is observable: any Bayesian state estimator (e.g. a Kalman filter) utilizing this pair is denoted as a **kernel**

observer [2]. In the controls case, given a spatiotemporally evolving system modeled using (6), we need to choose a set of N sensing locations and ℓ' control locations, such that even with $N \ll M,\ \ell' \ll M$, the functional evolution of the spatiotemporal model can be controlled; in this case, we must design both a measurement operator K and a control operator K_{CD} such that the pair $(K_{CD}, \widehat{A})$ is controllable: a controls system utilizing this pair and the measurement operator K is denoted as a **kernel controller**.

Preliminaries on Rational Canonical Structures

We take a geometric approach towards the choice of sampling locations for inferring w_τ in (3); the extension for control is similar. We use the notation $\mathcal{V}$, with $\dim(\mathcal{V}) = M$, to emphasize the fact that these theorems hold for any finite-dimensional vector space. Consider the linear operator $\mathcal{A} : \mathcal{V} \to \mathcal{V}$, and recall that the definition of observability requires the construction of a linear operator $\mathcal{K} : \mathcal{V} \to \mathcal{U}$, with $\dim(\mathcal{U}) = N$, such that $\operatorname{rank}\left[\,(\mathcal{K})^T \cdots \left(\mathcal{K}\mathcal{A}^{M-1}\right)^T\,\right]^T = M$. In most applications, if $N \geq M$, and $\operatorname{rank}\mathcal{K} = N$, it is reasonable to expect that observability may be achieved. However, for our purposes, N must be *significantly* less than M. Therefore, we must design $\mathcal{K}$ with as small a rank as possible. To do so, we require a series of vectors v_i that, under repeated iterations of $\mathcal{A}$, can generate a basis for $\mathcal{V}$. For this task, we will use a fundamental decomposition result from the theory of modules, known as the *rational canonical structure* of $\mathcal{A}$ [34]. The intuition here is that if the sequence $\{v_i\}_i$ can generate this basis, it can be directly used to construct $\mathcal{K}$.

The linear operator $\mathcal{A} : \mathcal{V} \to \mathcal{V}$ has a characteristic polynomial $\pi(\lambda)$ such that $\pi(\mathcal{A}) = 0$ by the Cayley-Hamilton theorem. The minimal polynomial (MP) of $\mathcal{A}$ is the monic polynomial $\alpha(\cdot)$ of least degree (denoted by $\deg(\cdot)$) given as $\alpha(\lambda) = a_0 + a_1\lambda + \cdots + \lambda^{\deg(\alpha)} = 0$, such that $\alpha(\mathcal{A}) = a_0 I + a_1\mathcal{A} + \cdots + \mathcal{A}^{\deg(\alpha)} = \mathbf{0}$. The MP is unique and divides $\pi(\lambda)$, so that $\deg(\alpha) \leq \deg(\pi)$. The MP of a vector $v \in \mathcal{V}$ *relative to* $\mathcal{A}$ is the unique monic polynomial ξ_v of least degree such that $\xi_v(\mathcal{A})v = a_0 v + a_1\mathcal{A}v + \cdots + \mathcal{A}^{\deg(\alpha)}v = 0$. If $\deg(\alpha) = M$, then $\mathcal{A}$ is *cyclic* and $\exists v \in \mathcal{V}$, such that the vectors $\{v, \mathcal{A}v, \ldots, \mathcal{A}^{M-1}v\}$ form a basis for $\mathcal{V}$; this is the same as saying that the pair $(v^T, \mathcal{A}^T)$ is observable. A subspace $\mathcal{V}_\mathcal{S} \subset \mathcal{V}$ s.t. $\mathcal{A}\mathcal{V}_\mathcal{S} \subset \mathcal{V}_\mathcal{S}$ is *$\mathcal{A}$-cyclic* if $\mathcal{A}_{|\mathcal{V}_\mathcal{S}}$, the restriction of $\mathcal{A}$ to the subspace $\mathcal{V}_\mathcal{S}$, is cyclic. If $\alpha(\lambda)$ is the minimal polynomial of $\mathcal{A}$ and $\deg(\alpha) = m < M$, $\exists\ v \in \mathcal{V}$ such that $\{v, \mathcal{A}v, \ldots, \mathcal{A}^{m-1}v\}$ span an m-dimensional $\mathcal{A}$-cyclic subspace $\mathcal{V}_\mathcal{S}$, with v being the *cyclic generator* of $\mathcal{V}_\mathcal{S}$. The subspace $\mathcal{V}_\mathcal{S}$ decomposes $\mathcal{V}$ relative to $\mathcal{A}$. By the rational canonical structure theorem (Theorem 0.1 of [34]), $\mathcal{A}$ can be successively decomposed into subspaces $\mathcal{V}_i \subset \mathcal{V}$, $i \in \{1, \ldots, \ell\}$, s.t. $\mathcal{V} = \mathcal{V}_1 \oplus \ldots \oplus \mathcal{V}_\ell$, $\mathcal{A}\mathcal{V}_i \subset \mathcal{V}_i$,

[2] In the case where no measurements are taken, for the sake of consistency, we denote the state estimator as an **autonomous kernel observer**, despite this being somewhat of an oxymoron.

and $\mathcal{A}_{|\mathcal{V}_i}, i \in \{1, \ldots, \ell\}$, are cyclic[3]. The integer ℓ is unique and is called the *cyclic index of* $\mathcal{A}$.

One of our main results is to show that the cyclic index is a lower bound on the number of measurements required to reconstruct w_τ (see Prop. 3 and Alg. 1 in [16]). The matrix transform associated to this theorem is known as the *Frobenius normal form* (denoted by $C \in \mathbb{R}^{M\times M}$): for $\mathcal{A} \in \mathbb{R}^{M\times M}$, $\exists Q \in \mathbb{R}^{M\times M}$ invertible such that $\mathcal{A} = QCQ^{-1}$. We will also use the *Jordan decomposition*, where for $\mathcal{A} \in \mathbb{R}^{M\times M}$, $\exists P \in \mathbb{R}^{M\times M}$ invertible such that $\mathcal{A} = P\Lambda P^{-1}$, where Λ is a unique block diagonal matrix with Jordan blocks with λ_i along the diagonal. If all the eigenvalues λ_i are nonzero and real, we say the matrix has a *full-rank Jordan decomposition*.

Main Results

In this section, we prove results concerning the observability of spatiotemporally varying functions modeled by the functional evolution and measurement equations (3). In particular, observability of the system states implies that we can recover the current state of the spatiotemporally varying function using a small number of sampling locations N, which allows us to 1) track the function, and 2) predict its evolution forward in time. We work with the approximation $\widehat{\mathcal{H}} \approx \mathcal{H}$: given M basis functions, this implies that the dual space of $\widehat{\mathcal{H}}$ is $\mathbb{R}^M$. Proposition 1 shows that if $\widehat{A}$ has a full-rank Jordan decomposition, the observation matrix K meeting a condition called *shadedness* (Definition 1) is sufficient for the system to be observable. Proposition 2 provides a lower bound on the number of sampling locations required for observability which holds for any $\widehat{A}$. Proposition 3 constructively shows the existence of an abstract measurement map $\widetilde{K}$ achieving this lower bound. Since the measurement map does not have the structure of a kernel matrix, a slightly weaker sufficient condition for the observability of any $\widehat{A}$ is in Theorem 1. Finally, since both K and K_{CD} are kernel matrices generated from a shared kernel, these observability results translate directly into controllability results. Proofs of all claims are in the appendix.

Definition 1: **(Shaded Observation Matrix)** Given $k : \Omega \times \Omega \to \mathbb{R}$ positive-definite on a domain Ω, let $\{\widehat{\psi}_1(x), \ldots, \widehat{\psi}_M(x)\}$ be the set of bases generating an approximate feature map $\widehat{\psi} : \Omega \to \widehat{\mathcal{H}}$, and let $\mathcal{X} = \{x_1, \ldots, x_N\}$ be the set of sampling (or sensing) locations, with each $x_i \in \Omega$. Let $K \in \mathbb{R}^{N\times M}$ be the observation matrix, where $K_{ij} := \widehat{\psi}_j(x_i)$. For each row $K_{(i)} := \left[\, \widehat{\psi}_1(x_i) \cdots \widehat{\psi}_M(x_i) \,\right]$, define the set $\mathcal{I}_{(i)} := \{\iota_1^{(i)}, \iota_2^{(i)}, \ldots, \iota_{M_i}^{(i)}\}$ to be the indices in the observation matrix row i which are nonzero. Then if $\bigcup_{i\in\{1,\ldots,N\}} \mathcal{I}^{(i)} = \{1, 2, \ldots, M\}$, we denote K as a *shaded observation matrix* (see Figure 4a).

This definition seems quite abstract, so the following remark considers a more concrete

[3]In general, the subspaces $\mathcal{V}_i$ are not unique for a fixed $\mathcal{A}$.

example.

Remark 1: let $\widehat{\psi}$ be generated by the dictionary given by $\mathcal{C} = \{c_1, \ldots, c_M\}$, $c_i \in \Omega$. Note that since $\widehat{\psi}_j(x_i) = \langle \psi(x_i), \psi(c_j) \rangle_{\mathcal{H}} = k(x_i, c_j)$, K is the kernel matrix between $\mathcal{X}$ and $\mathcal{C}$. For the kernel matrix to be shaded thus implies that there does not exist an atom $\psi(c_j)$ such that the projections $\langle \psi(x_i), \psi(c_j) \rangle_{\mathcal{H}}$ vanish for all x_i, $1 \leq i \leq N$. Intuitively, the shadedness property requires that the sensor locations x_i are privy to information propagating from every c_j. As an example, note that, in principle, for the Gaussian kernel, a single row generates a shaded kernel matrix[4].

Proposition 1: Given $k : \Omega \times \Omega \to \mathbb{R}$ positive-definite on a domain Ω, let $\{\widehat{\psi}_1(x), \ldots, \widehat{\psi}_M(x)\}$ be the set of bases generating an approximate feature map $\widehat{\psi} : \Omega \to \widehat{\mathcal{H}}$, and let $\mathcal{X} = \{x_1, \ldots, x_N\}$, $x_i \in \Omega$. Consider the discrete linear system on $\widehat{\mathcal{H}}$ given by the evolution and measurement equations (3). Suppose that a full-rank Jordan decomposition of $\widehat{A} \in \mathbb{R}^{M \times M}$ of the form $\widehat{A} = P \Lambda P^{-1}$ exists, where $\Lambda = [\, \Lambda_1 \, \cdots \, \Lambda_O \,]$, and there are no repeated eigenvalues. Then, given a set of time instances $\Upsilon = \{\tau_1, \tau_2, \ldots, \tau_L\}$, and a set of sampling locations $\mathcal{X} = \{x_1, \ldots, x_N\}$, the system (3) is observable if the observation matrix K_{ij} is shaded according to Definition 1, Υ has distinct values, and $|\Upsilon| \geq M$.

When the eigenvalues of the system matrix are repeated, it is not enough for K to be shaded. In the next proposition, we take a geometric approach and utilize the rational canonical form of $\widehat{A}$ to obtain a lower bound on the number of sampling locations required. Let r be the number of unique eigenvalues of $\widehat{A}$, and let γ_{λ_i} denote the geometric multiplicity of eigenvalue λ_i. Then the *cyclic index* of $\widehat{A}$ is defined as $\ell = \max_{1 \leq i \leq r} \gamma_{\lambda_i}$ [34].

Proposition 2: Suppose that the conditions in Proposition 1 hold, with the relaxation that the Jordan blocks $[\, \Lambda_1 \, \cdots \, \Lambda_O \,]$ may have repeated eigenvalues (i.e. $\exists \Lambda_i$ and Λ_j s.t. $\lambda_i = \lambda_j$). Then there exist kernels $k(x, y)$ such that the lower bound ℓ on the number of sampling locations N is given by the cyclic index of $\widehat{A}$. In other words, the system in (3) is observable if $N \geq \ell$.

We will give a concrete example to build intuition regarding this lower bound below. For now, we note the following:

Proposition 3: Given the conditions stated in Proposition 2, it is possible to construct a measurement map $\widetilde{K} \in \mathbb{R}^{\ell \times M}$ for the system given by (3), such that the pair $(\widetilde{K}, \widehat{A})$ is observable.

The construction provided in the proof of Proposition 3 is utilized in Algorithm 1, which

[4]However, in this case, the matrix can have many entries that are extremely close to zero, and will probably be very ill-conditioned.

uses the rational canonical structure of $\widehat{A}$ to generate a series of vectors $v_i \in \mathbb{R}^M$, whose iterations $\{v_1, \ldots, \widehat{A}^{m_1-1}v_1, \ldots, v_\ell, \ldots, \widehat{A}^{m_\ell-1}v_\ell\}$ generate a basis for $\mathbb{R}^M$. Unfortunately, the measurement map $\widetilde{K}$, being an abstract construction unrelated to the kernel, does not directly select $\mathcal{X}$. We will show how to use the measurement map to guide a search for $\mathcal{X}$ in Remark 2 (in Appendix). For now, we state a sufficient condition for observability of a general system.

Theorem 1: Suppose that the conditions in Proposition 1 hold, with the relaxation that the Jordan blocks $\begin{bmatrix}\Lambda_1 & \cdots & \Lambda_O\end{bmatrix}$ may have repeated eigenvalues. Let ℓ be the cyclic index of $\widehat{A}$. Define

$$\mathbf{K} = \begin{bmatrix} K^{(1)^T} & \ldots & K^{(\ell)^T} \end{bmatrix}^T \tag{7}$$

as the ℓ-*shaded matrix* (see Figure 4b) which consists of ℓ shaded matrices with the property that any subset of ℓ columns in the matrix are linearly independent from each other. Then system (3) is observable if Υ has distinct values, and $|\Upsilon| \geq M$.

While Theorem 1 is a quite general result, the condition that any ℓ columns of $\mathbf{K}$ be linearly independent is a very stringent condition. One scenario where this condition can be met with minimal measurements is in the case when the feature map $\widehat{\psi}(x)$ is generated by a dictionary of atoms with the Gaussian RBF kernel evaluated at sampling locations $\{x_1, \ldots, x_N\}$ according to (2), where $x_i \in \Omega \subset \mathbb{R}^d$, and x_i are sampled from a non-degenerate probability distribution on Ω such as the uniform distribution. For a semi-deterministic approach, when the dynamics matrix $\widehat{A}$ is block-diagonal, we can utilize a simple heuristic:

Remark 2: Let Ω be compact, $\mathcal{C} = \{c_1, \ldots, c_M\}$, $c_i \in \Omega$, and let the approximate feature map be defined by (2). Consider the system (3) with $\widehat{A} = \Lambda$, and let $\Upsilon = \{0, 1, \ldots, M-1\}$. Then the measurement map $\widetilde{K}$'s values lie in $\{0, 1\}$; in particular, each row $\widetilde{K}^{(j)}$, $j \in \{1, \ldots, \ell\}$, corresponds to a subspace $\widehat{\mathcal{H}}_j$, generated by a subset of centers $\mathcal{C}^{(j)} \subset \mathcal{C}$. Generate samples $x_i^{(j)}$ to create a kernel matrix $K^{(j)}$ that is shaded only with respect to centers $\mathcal{C}^{(j)}$. Once this is done, move on to the next subspace $\widehat{\mathcal{H}}_{j+1}$. When all ℓ rows of $\widetilde{K}$ are accounted for, construct the matrix $\mathbf{K}$ as in (7). Then the resulting system $(\mathbf{K}, \widehat{A})$ is observable.

This heuristic is formalized in Algorithm 2 in the supplementary. Note that in practice, the matrix $\widehat{A}$ needs to be inferred from measurements of the process f_τ. If no assumptions are placed on $\widehat{A}$, it's clear that at least M sensors are required for the system identification phase. Future work will study the precise conditions under which system identification is possible with less than M sensors.

Discussion of Theoretical Results

The systems-theoretic approach taken in this paper reveals something rather surprising: functions with complex dynamics (with a small cyclic index) can be recovered with less sensor placements than functions with simpler dynamics. Although seemingly counterintuitive, it becomes clear that this is because complex dynamics, which are characterized by a lower geometric multiplicity of the eigenvalues, ensure that the orbit $\mathbf{\Theta} := \{\widehat{A}w_\tau\}_{\tau\in\Upsilon}$ traverses a greater portion of $\mathbb{R}^M \equiv \widehat{\mathcal{H}}$ and thus that fewer sensors can recover more geometric information. On the other hand, in 'simpler' functional evolution, $\mathbf{\Theta}$ evolves along strict subspaces of $\mathbb{R}^M$, and so more independent sensors are required to infer the same amount of information.

In the case described in Remark 2, we have a set of centers $\mathcal{C} = \{c_1, \ldots, c_M\}$, which generate the bases $\mathcal{F}^{\mathcal{C}} = \{\psi(c_1), \cdots, \psi(c_M)\}$. Let the cyclic index be ℓ: this implies that there exist ℓ subsets $\Psi^{(i)}$ of $\mathcal{F}^{\mathcal{C}}$ with at least one element $\psi(c_j)$ each, leading to $\binom{M}{\ell}$ possible choices: Figure 8 represents these choices as hyperplanes separating the subsets. The measurement map described in Alg. 1 in [16] induces this *decomposition of bases* $\mathcal{F}^{\mathcal{C}} = \{\Psi^{(1)}, \ldots, \Psi^{(\ell)}\}$ in polynomial time. Further, each subset $\Psi^{(i)}$ is directly associated to a subset of centers $\mathcal{C}^{(i)} \subset \mathcal{C}$, which allows us to pick targeted sensor locations $x_i \in \Omega$. In particular, for radially symmetric kernels such as the Gaussian, the centroid of the convex hull of $\mathcal{C}^{(i)}$ is sufficient for generating a sensor placement. The measurement map is a significant theoretical insight into sensor placement for dynamically changing environments, because it directly takes into account the dynamics of the process. Of course, in practice, this may be too expensive for approximate feature spaces with M very large, so one can use random sampling to generate the sensor locations instead, at the cost of N being larger than ℓ. The advantage here though is that since random sampling is computationally inexpensive, different choices of sensor placements can be generated and evaluated relatively quickly.

Another point to note is that since the collection of bases $\{\widehat{\psi}_i(x)\}_{i=1}^M$ determines the richness of the function space $\widehat{\mathcal{H}} \approx \mathcal{H}$ we operate in, it determines the fidelity of the model approximation to the true time-varying function. As a consequence, observability of the system in $\widehat{\mathcal{H}}$ refers to the best possible approximation in $\widehat{\mathcal{H}}$. The greater the number of bases, the higher the dimensionality, which results in greater model fidelity, but which may require a much greater number of measurements for state recovery. This is where the lower bounds presented in the paper are particularly useful, because they show that for functional evolutions corresponding to certain $\widehat{A}$, *the number of sensor placements are essentially independent of the dimensionality* M, but depend rather on the cyclic index of $\widehat{A}$.

Figure 7 gives an overall picture on the process of generating a kernel observer, while Figure 8 gives two approaches to sensor selection in our framework. The measurement map

approach can generate a smaller set of sensors than the random placement approach, but comes at an additional computational cost.

Random Sensor Placement

We now elaborate on how the challenging problem of sensor placement can be tackled through random selection. This process of random selection is a product of the kernel observer model described above. We present the theoretical background required to prove Theorem 2, which states the expected number of randomly placed sensors required to monitor a given spatiotemporal process, and Theorem 3, which determines the probability with which optimal sensor placement is ensured given that, N number of sensors have been placed.

As discussed earlier, we work with an approximate feature space $\widehat{\mathcal{H}}$, with the corresponding transition operator $\widehat{A} : \widehat{\mathcal{H}} \to \widehat{\mathcal{H}}$, representing finite-dimensional functional evolution. To achieve observability for the pair $(\widehat{A}, K)$, row vectors of the corresponding observability matrix, $\mathcal{O}$, should form the basis for the $\mathbb{R}^M$-dimensional space $\widehat{\mathcal{H}}$. According to the rational canonical structure Theorem [34], $\widehat{A}$ can successively decompose the dual space $\mathbb{R}^M$ into subspaces, $\mathcal{V}_i \subset \mathcal{V}$, $i \in \{1, \ldots, \ell\}$, with properties, i) $\mathcal{V} = \mathcal{V}_1 \oplus ... \oplus \mathcal{V}_\ell$, ii) $\widehat{A}\mathcal{V}_i \subset \mathcal{V}_i$, and iii) $\widehat{A}|\mathcal{V}_i, i \in \{1, \ldots, \ell\}$, are cyclic. The integer ℓ is unique and is called the *cyclic index of* $\widehat{A}$. Each of these properties contribute towards the theorem on the number of random samples required to achieve observability. The first property shows that the space $\mathbb{R}^M$ can be decomposed into ℓ independent subspaces. The second property shows that the vector $v_i \in \mathcal{V}_i$ stays in $\mathcal{V}_i$ even when operated upon by $\widehat{A}$. Thus, to generate bases for $\mathbb{R}^M$, one needs at least ℓ vectors, say, $v_1, \ldots, v_\ell$, with respect to each subspace $\mathcal{V}_1, \ldots, \mathcal{V}_\ell$. This holds due to the third property, but requires that the vectors $v_1, \ldots, v_\ell$, are the cyclic generators of their corresponding subspaces. Our analysis is based on whether a randomly selected sensor can generate a cyclic generator. To examine this, recall that a row vector $K_{(i)}$ generated by a randomly selected sensor location x_i takes the form,

$$K_{(i)} = \Big[k(x_i, c_1), \ldots, k(x_i, c_M)\Big]. \tag{8}$$

Here, for radial kernels for example, the entries corresponding to the centers closer to x_i tend to be non-zero, whereas the others tend to be zero. The rows $K_{(i)}$ from random sensor placement must be able to generate a basis for a subspace $\mathcal{V}_i$, and thus must be cyclic generators. We will derive the expected number of random sensor placements sufficient for observability for the case where $\widehat{A} = \Lambda$ and then attempt to generalize the result for any $\widehat{A}$. Note Λ is a block diagonal Jordan form. In this case, the cyclic generator for each subspace $\mathcal{V}_i$, is a vector v_i with non-zero entries corresponding to the leading entry of the Jordan blocks of $\mathcal{V}_i$.

Overall, our construction is as follows: for each subspace $\mathcal{V}_i$, let $\mathcal{C}_{\mathcal{V}_i} \subset \mathcal{C}$ be the centers

corresponding to those leading entry of Jordan blocks: then the minimum number of random samples required to generate the bases for $\mathcal{V}_i$ is equal to the number of Jordan blocks comprising $\mathcal{V}_i$. Altogether, the minimum number of random samples required to generate a basis for $\mathbb{R}^M$ is equal to the total number of Jordan blocks in $\widehat{A}$. Let ς be the total number of Jordan blocks in $\widehat{A}$, then

$$\varsigma = \sum_{\lambda \in \sigma(\widehat{A})} \gamma_{\widehat{A}}(\lambda) \tag{9}$$

where $\sigma(\widehat{A})$ represents the spectrum of $\widehat{A}$, whose elements are the eigenvalues of $\widehat{A}$, and $\gamma_{\widehat{A}}(\lambda)$ is the geometric multiplicity corresponding to the eigenvalue λ, which is also equal to the total number of Jordan blocks corresponding to the eigenvalue λ. Define a set of centers $\mathcal{C}_\varsigma$ with elements $\{c_1, c_2, \ldots, c_\varsigma\}$, to be the centers corresponding to the leading entries of the Jordan blocks. For sensor location $x \in \Omega$, and $\epsilon > 0$, let $k(x, c_j) > \epsilon$, denote the region $\Omega_j \subset \Omega$, such that the kernel evaluation with respect to center c_j is greater than ϵ, that is $\Omega_j \equiv \{x \in \Omega : k(x, c_j) > \epsilon\}$. We define p_ϵ as

$$p_\epsilon = \min_{c_j \in \mathcal{C}_\varsigma} \frac{\nu(k(x, c_j) > \epsilon)}{\nu(\Omega)}, \tag{10}$$

where ν is a measure in the real analysis sense. Hence, p_ϵ corresponds to a lower bound on the probability that a random sample lies within the $\epsilon-$shaded region of a particular center c_j. With all of this in place, we can prove the following theorem.

Theorem 2: Given the spatiotemporal function $f(x, \tau)$ with $x \in \Omega \subseteq \mathbb{R}^D, \tau \in \mathbb{Z}^+$ its kernel observer model (3), and a tolerance parameter $\epsilon > 0$, the expected number of randomly placed sensor locations required to achieve observability for the pair $(K, \widehat{A})$ is ς/p_ϵ where ς is the summation over geometric multiplicities of each $\lambda \in \sigma(\widehat{A})$ given by Equation (9).

Theorem 3: Given the spatiotemporal function $f(x, \tau)$ with $x \in \Omega \subseteq \mathbb{R}^D, \tau \in \mathbb{Z}^+$, its kernel observer model (3), a tolerance parameter $\epsilon > 0$, summation over geometric multiplicities of each $\lambda \in \sigma(\widehat{A})$ denoted by ς as in Equation (9), and a constant $\delta \in (0, 1]$, the probability that pair $(K, \widehat{A})$ is unobservable after the selection of N random sensors is at most $e^{\frac{-1}{2}(Np_\epsilon - 2\varsigma)}$, where p_ϵ is given by Equation (10) and $N \geq \varsigma/p_\epsilon$.

For the case when $\widehat{A} \neq \Lambda$, a change of basis can be used to obtain $\Lambda = P^{-1}\widehat{A}P$, where P is the projection map. There are two challenges in performing the above analysis for Λ so obtained: first, the leading entries of Jordan blocks do not directly correspond to the centers $\{c_1, \ldots, c_M\}$ which was the case for $\widehat{A} = \Lambda$. Second, although we can obtain the transformation of the row vector (Equation (8)) using the projection map P, we can no longer arrive at the definition of the probability p_ϵ as in Equation (10). The existence of the similarity transform hints that the results in Theorems 2-3 should hold for any $\widehat{A}$, but the mathematical tools utilized in the paper

seem to be insufficient to prove them. However, we present some empirical evidence for these claims for when $\widehat{A} \neq \Lambda$ in Section [20].

Generalizing Across Similar Spatiotemporally Evolving Systems

Building on the Kernel Observers method, let us introduce Evolving Gaussian Processes (E-GP). The primary novelty in this method of generating a model is learning an $\widehat{A}$ matrix for *multiple* systems. The ultimate goal of this research would be to generate highly efficient machine learning models that can be used instead of the costly numerical simulations for design and autonomy purposes. This would be a major success for the design and control of complex physical systems, such as soft robotics, as they would significantly reduce the cost and resources required in simulations. The ability to generalize across different physical situations, is critical. 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. 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. E-GP, as far as the authors know, was the first machine learning method to generalize across spatiotemporally evolving systems of such complexity using end-to-end data.

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 which govern fluid dynamics, 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 E-GP 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 [35].
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 this 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 quantity 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:

1) 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
2) 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) By performing an eigendecomposition of the $\widehat{A}$ matrix, and transforming the eigenvectors back from the weight space to the function space, one can obtain the Koopman modes (and associated eigenvalues) of the system.

Experimental Results

Sensing Locations for Synthetic Data Sets

The goal of this experiment is to investigate the dependency of the observability of system (3) on the shaded observation matrix and the lower bound presented in Proposition 2. The domain is fixed on the interval $\Omega = [0, 2\pi]$. First, we pick sets of points $\mathcal{C}^{(\iota)} = \{c_1, \ldots, c_{M_\iota}\},\ c_j \in \Omega$, $M = 50$, and construct a dynamics matrix $A = \Lambda \in \mathbb{R}^{M \times M}$, with cyclic index 5. We pick the RBF kernel $k(x, y) = e^{-\|x-y\|^2/2\sigma^2}$, $\sigma = 0.02$. Generating samples $\mathcal{X} = \{x_1, \ldots, x_N\}$, $x_i \in \Omega$ randomly, we compute the ℓ-shaded property and observability for this system. Figure 9a shows how shadedness is a necessary condition for observability, validating Proposition 1: the slight gap between shadedness and observability here can be explained due to numerical issues in computing the rank of $\mathcal{O}_\Upsilon$.

Next, we consider a system with a cyclic index $\ell = 18$ to verify random sensor placement results. We constructed the measurement operator K using the heuristic in Remark 2 (Algorithm **??**), and random sensor selection to generate the sampling locations $\mathcal{X}$. These results are presented in Figure 9b. The plot for random sampling which has been averaged over 100 runs, resembles a c.d.f function of an exponential distribution $F(X = x) = 1-\exp(-\lambda x)$. This verifies the claim made in Theorem 3, as the probability of becoming unobservable decays exponentially with the number of sensors placed. Also, fitting an exponential distribution we found that the mean λ^{-1} comes close to the ratio ς/p, which is the expected number of randomly placed sensors required for observability as per Theorem 2. Note that observability is not achieved if the number of samples $N < \ell$ verifying the result in Proposition 2.

Comparison With Nonstationary Kernel Methods on Real-World Data

We use three real-world datasets to evaluate and compare the kernel observer with the two different lines of approach for non-stationary kernels discussed in Section . For the Process Convolution with Local Smoothing Kernel (PCLSK) and Latent Extension of Input Space (LEIS) approaches, we compare with NOSTILL-GP [6] and [27] respectively, on the Intel Berkeley, Irish Wind and Ozone data-sets.

Model inference for the kernel observer involved three steps: 1) picking the Gaussian RBF kernel $k(x, y) = e^{-\|x-y\|^2/2\sigma^2}$, a search for the ideal σ is performed for a sparse Gaussian Process model (with a fixed basis vector set $\mathcal{C}$ selected using the method in [35]. For the data set discussed in this section, the number of basis vectors were equal to the number of sensing locations in the training set, with the domain for input set defined over $\mathbb{R}^2$; 2) having obtained σ, Gaussian process inference is used to generate weight vectors for each time-step in the training set, resulting in the sequence $w_\tau, \tau \in \{1, \ldots, T\}$; 3) matrix least-squares is applied to this sequence to infer $\widehat{A}$ (Algorithm 3 in the supplementary). For prediction in the autonomous setup, $\widehat{A}$ is used to propagate the state w_τ forward to make predictions with no feedback, and in the observer setup, a Kalman filter (Algorithm 4 in the supplementary) with N determined using Proposition 2, and locations picked randomly, is used to propagate w_τ forward to make predictions. We also compare with a baseline GP (denoted by 'original GP'), which is the sparse GP model trained using all of the available data.

Our first dataset, the Intel Berkeley research lab temperature data, consists of 50 wireless temperature sensors in indoor laboratory region spanning 40.5 meters in length and 31 meters in width[5]. Training data consists of temperature data on March 6th 2004 at intervals of 20 minutes (beginning 00:20 hrs) which totals to 72 timesteps. Testing is performed over another 72 timesteps

[5] http://db.csail.mit.edu/labdata/labdata.html

beginning 12:20 hrs of the same day. Out of 50 locations, we uniformly selected 25 locations each for training and testing purposes. Results of the prediction error are shown in box-plot form in Figure 10a and as a time-series in Figure 10b, note that 'Auto' refers to autonomous set up. Here, the cyclic index of $\widehat{A}$ was determined to be 2, so N was set to 2 for the kernel observer with feedback. Note that here, even the autonomous kernel observer outperforms PCLSK and LEIS overall, and the kernel observer with feedback with $N = 2$ significantly outperforms all other methods, which is why we did not include results with $N > 2$.

The second dataset is the Irish wind dataset, consisting of daily average wind speed data collected from year 1961 to 1978 at 12 meteorological stations in the Republic of Ireland[6]. The prediction error is in box-plot form in Figure 11a and as a time-series in Figure 11b. Again, the cyclic index of $\widehat{A}$ was determined to be 2. In this case, the autonomous kernel observer's performance is comparable to PCLSK and LEIS, while the kernel observer with feedback with $N = 2$ again outperforms all other methods.

Finally, the Ozone dataset measures ozone concentration (in parts per billion) measured at 60 stations by the United States Environmental Protection Agency [36] across USA. Due to missing measurements, we only selected data from year 1997 to 2013 for training and evaluation. For each station, we averaged ozone concentration over a period of three months, resulting in four quarters per year. Out of 60 sensor locations, we uniformly selected 30 for training and the remaining locations for testing purposes. The prediction error results are presented in box-plot form in Figure 12a and as a time-series in Figure 12b. Here, the cyclic index of $\widehat{A}$ was determined to be 1. In this case, the performance of autonomous kernel observer is comparable to PCLSK and LEIS, with kernel observer with feedback with $N = 1$ performing the best. Table 1 reports the total training and prediction times associated with PCLSK, LEIS, and the kernel observer. We observed that, 1) the kernel observer is an order of magnitude faster than the competing methods, and 2) even for small sets, competing methods did not scale well.

Prediction of Global Ocean Surface Temperature

We analyzed the feasibility of our approach on a very large dataset from the National Oceanographic Data Center: the 4 km AVHRR Pathfinder project, which is a satellite monitoring global ocean surface temperature (Figure 13a shows raw satellite data). This dataset is challenging, with measurements at over 37 million possible coordinates, but with only around 3-4 million measurements available per day, leading to a lot of missing data. The goal was to learn the day and night temperature models on data from the year 2011, and then to monitor thereafter for 2012. Success in monitoring would demonstrate two things: 1) the modeling process can

[6] http://lib.stat.cmu.edu/datasets/wind.desc

capture spatiotemporal trends that generalize across years, and 2) the observer framework allows us to infer the state using a number of measurements that are an order of magnitude fewer than available. Note that due to the size of the dataset and the high computational requirements of the nonstationary kernel methods, a comparison with them was not pursued. To build the autonomous kernel observer and general kernel observer models, we followed the same procedure outlined in Section , but with $\mathcal{C} = \{c_1, \ldots, c_M\}$, $c_j \in \mathbb{R}^2$, $|\mathcal{C}| = 300$. The Kalman filter for the general kernel observer model used $N \in \{250, 500, 1000\}$ at random locations to track the system state given a random initial condition w_0. As a fair baseline, the observers are compared to training a sparse GP model (labeled 'original') on approximately $400,000$ measurements per day. Figure 13b is an estimate of global ocean surface temperatures obtained using autonomous kernel observer. Figures 13c and 13d compare the autonomous and feedback approach with $1,000$ samples to the baseline GP; here, it can be seen that the autonomous does well in the beginning, but then incurs an unacceptable amount of error when the time series goes into 2012, i.e. where the model has not seen any training data, whereas FKO does well throughout. Figures 13e and 13f show a comparison of the RMS error of estimated values from the real data. This figure shows the trend of the observer getting better and better state estimates as a function of the number of sensing locations N [7]. Time required for kernel observer is much lesser than retraining the model every time step, see figures 13g-13h.

Weather Anomaly in 2012: We further investigated the poor performance of autonomous kernel observer in the year 2012 as observed in Figures 13c and 13d. Clearly, the error in prediction blows up at the start of May in the year 2012. Indicating that the autonomous model trained using the data of the year 2011 does well in capturing the annual weather dynamics up to the month of May 2012. We turned our attention towards the weather in May 2012, as changes in ocean temperatures are directly related to the weather. Surprisingly, severe weather onset was reported on the east coast of United States in May 2012, and this anomaly continued over the period of May to June 2012. Thus, the apparent poor performance of autonomous kernel observer can actually be a useful indicator in detecting the anomalous behavior, as was the case with the ocean temperature in May 2012 which had deviated from the nominal weather dynamics observed in the year 2011 in which no severe weather anomalies were reported. Further, we identified the locations at which the prediction error was two standard deviations above the mean error. These error locations are plotted in Figure 14. Error locations on 6-7th May coincide with the severe weather onset on the east coast of United states and that of 28-29th May were along the track of storm Beryl approaching the eastern coastline. This shows that the autonomous kernel observer

[7]Note that we checked the performance of training a GP with only $1,000$ samples as a control, but the average error was about 10 Kelvins, i.e. much worse than FKO.

model captures the dynamics of spatiotemporal evolution, and has the potential of identifying current behavior as anomalous to that observed in the training set.

Control of a linear PDE

We then employed kernel controllers for controlling an approximation to the scalar diffusion equation $u_t = bu_{xx}$ on the domain $\Omega = [0, 1]$, with $b = 0.25$. The solution to this equation is infinite-dimensional, so we chose a kernel $k(x, y) = e^{-(\|x-y\|^2/2\sigma^2)}$, and a set of atoms $\mathcal{F}^{\mathcal{C}} = \{c_1, \ldots, c_M\}$, $c_i \in \Omega$, with $M = 25$ generating $\mathcal{H}^{\mathcal{C}}$, the space approximating $\mathcal{H}$, and another set of atoms $\mathcal{F}_D = \{\psi(d_1), \ldots, \psi(d_{\ell'})\}$, $d_j \in \Omega$, $\ell' = 13$, generating the control space $\widetilde{\mathcal{H}}$. The number of, and the location of the observations was chosen to be the same as that of the actuation locations d_j. First, tests (not reported here) were conducted to ensure that the solution to the diffusion equation is well approximated in $\mathcal{H}^{\mathcal{C}}$. Matrix least-squares was used to infer $\widehat{A}$. Figure 15a shows an example of an initial function f_{init} evolving according to the PDE. A reference function $f_{\text{ref}} \in \mathcal{H}^{\mathcal{C}}$ was chosen to drive f_{init} to f_{ref} under the action of the PDE. Finally, Algorithm **??** was used to control the PDE, driving f_{init} to f_{ref}; Figure 15b shows the absolute value of the error between f_k and f_{ref} as a function of time.

Conclusions

In this paper we presented a systems-theoretic approach to the problem of predicting complex spatiotemporally evolving phenomena. Our approach focused on estimating the latent state of the spatiotemporal phenomena with a minimum number of sampling locations. Theoretical and empirical results demonstrate that the bounds concerning the number of sampling locations provided here are non-conservative, and as such, provide direct guidance in ensuring robust real-world sensor network design that must also account for sensor fault-tolerance and reliability considerations.

Conclusion

Machine learning techniques such as Gaussian Process regression and deep learning are providing increasingly more powerful ways of learning predictive models. However, these techniques are limited by the datasets that they are trained with. Hence, their predictions are not always reliable when predicting real-world complex spatiotemporally evolving phenomena with lots of variability. Indeed, years worth of weather data cannot be a reliable predictor of weather, nor can data from one farm directly relate to another. Hence, when engineering complex cyber physical systems such as team of mechanically weeding robots, engineers have to think of ways to supplement the predictions of the machine learning models with real measurements.

However, this is not always immediate when one works in the abstract feature spaces utilized in machine learning models. We presented here Evolving Gaussian Processes, which are formed by embedding a linear dynamical system in the reproducing kernel Hilbert spaces generated by the Gaussian Process model. The parameters of the linear dynamical system can be trained to approximately predict the evolution. In itself this can lead to powerful and generalizable models, as our results demonstrated in predicting solutions to Navier Stokes equations. Furthermore, a Kalman filter can be embedded with the linear dynamical system in the RKHS that can keep the predictions on track with real sensor measurements. This creates a very powerful prediction system that is capable of predicting nonlinear evolution using very few sensor measurements. We showed that the geometric properties of the linear dynamical system can be utilized to determine sensor numbers and locations. Looking forward, it would interesting to extend the idea to include nonlinear dynamical systems in the feature space. Such embeddings could improve the capability of the E-GP model. However, it seems from our results with complex datasets that the feedback with sensors creates a robust monitoring system with the predictions with a simple liner model. The theoretical limits of the tradeoff between improving the model versus increasing the sensing in the context of E-GPs are interesting to study. Finally we note also that deep neural networks have also been applied to the spatiotemporal modeling problem with significant empirical success [37], but because these methods are 1) nonlinear in their parameters, and 2) lack the spatial-encoding properties of some types of kernels, they are less amenable to analysis for the tasks we consider here. Yet, as sidebar indicates, there could be interesting future work in this area.

ACKNOWLEDGMENTS

The work presented was supported in part by AFOSR #FA9550-15-1-0146, and USDA/NSF grant CPS: Medium: Robust Deep Learning and Decision Making for Mechanical Weeding Agbots. We thank Adam Davis from UIUC Crop-Science, and Chinmay Soman from EarthSense in helping us understand the herbicide resistant weed problem and envisioning CPS and robotics solutions for it.

References

[1] Wyatt McAllistar, Denis Osipychev, Girish Chowdhary, and Adam Davis. Multi-agent planning for coordinated robotic weed killing. In *International Conference on Intelligent Robots and Systems (IROS)*, Madrid, Spain, 2018. IEEE.

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

[3] Carl E. Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, December 2005.

[4] K.-R. Müller, S. Mika, G. Rätsch, S. Tsuda, and B Schölkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–202, 2001.

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

[6] 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.

[7] Chunsheng Ma. Nonstationary covariance functions that model space–time interactions. *Statistics & Probability Letters*, 61(4):411–419, 2003.

[8] 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.

[9] G. Chowdhary, H. Kingravi, J.P. How, and P. Vela. Nonparametric adaptive control using gaussian processes. In *IEEE Conference on Decision and Control (CDC)*. IEEE, 2013.

[10] Suvrit Sra, Sebastian Nowozin, and Stephen J Wright. *Optimization for machine learning*. Mit Press, 2012.

[11] Sadeep Jayasumana, Richard Hartley, Mathieu Salzmann, Hongdong Li, and Mehrtash Harandi. Kernel Methods on Riemannian Manifolds with Gaussian RBF Kernels. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2015.

[12] A. Gelb. *Applied Optimal Estimation*. MIT press (QA402.A5), Cambridge, MA, 1974.

[13] Kanti V Mardia, Colin Goodall, Edwin J Redfern, and Francisco J Alonso. The kriged kalman filter. *Test*, 7(2):217–282, 1998.

[14] F. Cucker and S. Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39(1), 2002.

[15] Hassan A Kingravi, Girish Chowdhary, Patricio A Vela, and Eric N. Johnson. Reproducing kernel hilbert space approach for the online update of radial bases in neuro-adaptive control. *Neural Networks and Learning Systems, IEEE Transactions on*, 23(7):1130–1141, 2012.

[16] 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.

[17] Joshua Whitman, Harshal Maske, Girish Chowdhary, Hassan Kingravi, Chen Lu, and Balaji Jayaraman. Modeling nonlinear dynamical fluid flows with evolving gaussian process models. In *NIPS workshop on Machine Learning in the Wild*, Barcelona, Spain, 2016. NIPS.

[18] Hassan A. Kingravi, Harshal Maske, and Girish Chowdhary. Kernel controllers: A systems-theoretic approach for data-driven modeling and control of spatiotemporally evolving processes. *arXiv*, abs/1508.02086, 2015.

[19] Joshua Whitman and Girish Chowdhary. Learning dynamics across similar spatiotemporally-evolving physical systems. In *Conference on Robot Learning*, pages 472–481, 2017.

[20] Harshal Maske, Hassan Kingravi, and Girish Chowdhary. Sensor selection via observability analysis in feature space. In *American Control Conference*, 2018. accepted.

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

[22] Fernando Pérez-Cruz, Steven Van Vaerenbergh, Juan José Murillo-Fuentes, Miguel Lázaro-Gredilla, and Ignacio Santamaria. Gaussian processes for nonlinear signal processing: An overview of recent advances. *Signal Processing Magazine, IEEE*, 30(4):40–50, 2013.

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

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

[25] Amarjeet Singh, Fabio Ramos, H Durrant-Whyte, and William J Kaiser. Modeling and decision making in spatio-temporal processes for environmental surveillance. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 5490–5497. IEEE, 2010.

[26] 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.

[27] Tobias Pfingsten, Malte Kuss, and Carl Edward Rasmussen. Nonstationary gaussian process regression using a latent extension of the input space. *URL http://www. kyb. mpg. de/˜ tpfingst*, 2006.

[28] Mehran Mesbahi and Magnus Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, 2010.

[29] C. Guestrin, A. Krause, and A. Singh. Near-optimal sensor placements in gaussian processes. In *International Conference on Machine Learning*, 2005.

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

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

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

[33] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control*. Prentice Hall, Upper Saddle River, NJ, 1996.

[34] W Murray Wonham. *Linear multivariable control*. Springer, 1974.

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

[36] Lixin Li, Xingyou Zhang, and Reinhard Piltner. A spatiotemporal database for ozone in the conterminous us. In *Thirteenth International Symposium on Temporal Representation and Reasoning (TIME'06)*, pages 168–176. IEEE, 2006.

[37] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[38] Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[39] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[40] Y. Bengio. Learning deep architectures for ai. *Foundations and Trends in Machine Learning*, 2:1–127, 2009.

[41] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

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

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

[44] 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.

[45] 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.

Table 1: Total training and prediction times for Figs. 10 and 11

	Intel Berkeley	**Irish** Wind	**Ozone**
Data Size (*bases-timesteps*)	25-72	12-36	30-68
Kernel Observer	2.1 sec	0.1 sec	1.61 sec
PCLSK	121.4 sec	7.0 sec	91.90 sec
LEIS	43.8 sec	2.8 sec	37.41 sec

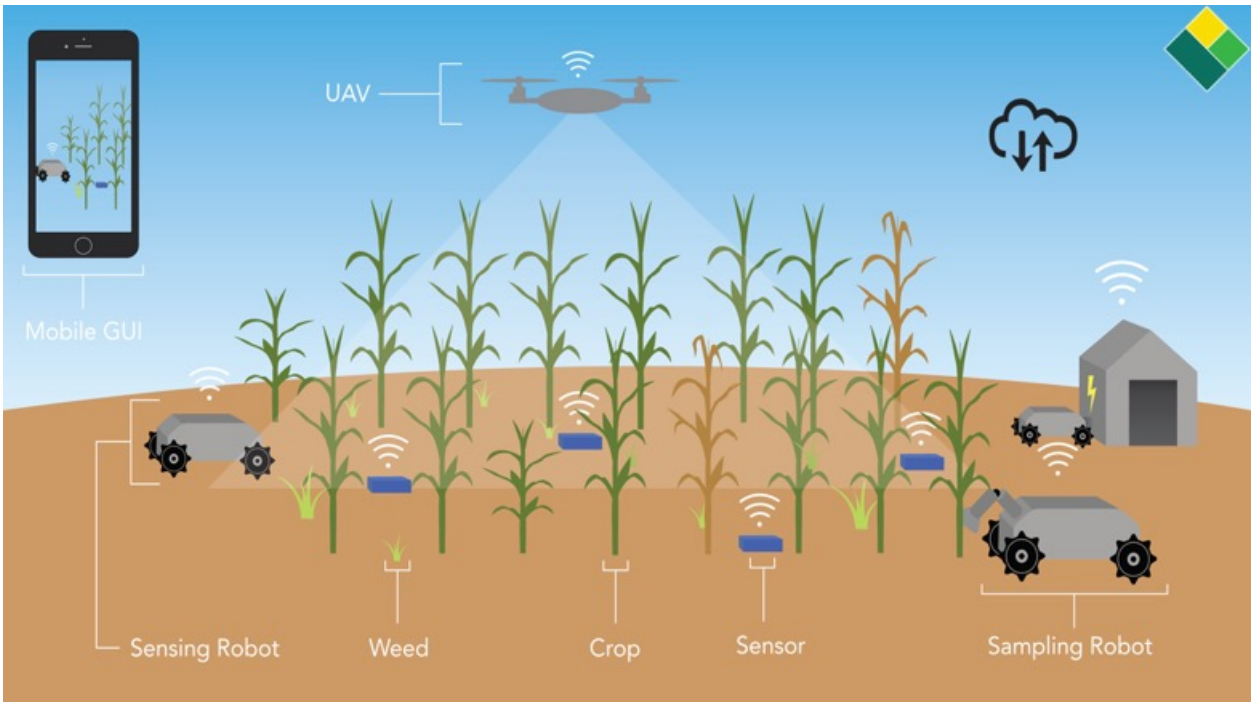


Figure 1: A Cyber Physical system consisting of a distributed team of robots for mechanical weed management on a farm. Image courtesy EarthSense inc.

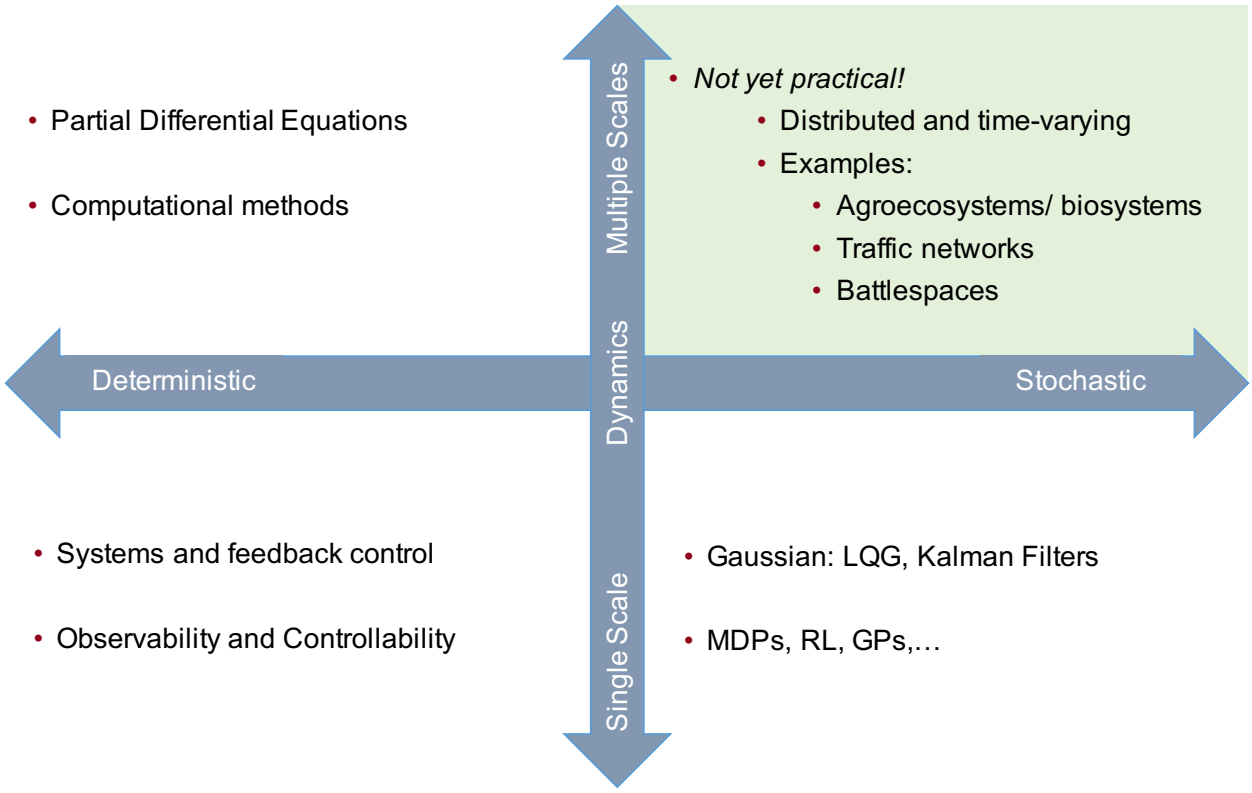


Figure 2: Modeling, monitoring, and controlling dynamical systems with complex and uncertain dynamics, such as agricultural, traffic, or weather monitoring systems, presents exciting open challenge for the controls community. The bottom left quadrant describes linear and time-invariant systems with single- scale dynamics, for which the theory of feedback control of dynamical systems is often sufficient. The bottom right quadrant shows stochastic single-scaled systems, where approaches such as Kalman Filters and Gaussian optimization have marked several successes of control systems enabling endeavors from Lunar landings to GPS navigation. The top left quadrant denotes systems with dynamics at multiple scales, where efficient computational solutions to Partial Differential Equations (PDEs) is an highly active area of research. However, fundamental theoretical advances and practical algorithms are needed to enable autonomous decision making for distributed stochastic Cyber Physical Systems with dynamics at multiple scales – shown in the top right quadrant – such as distributed agricultural robotic systems, traffic networks, and weather monitoring systems with mobile and stationary sensors.

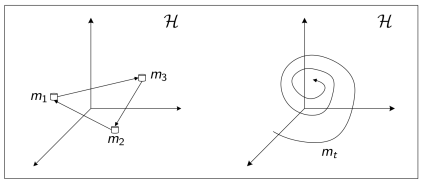
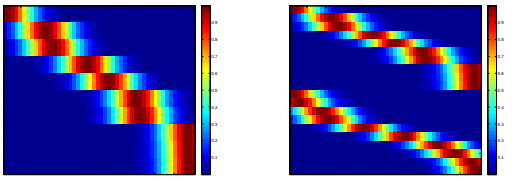
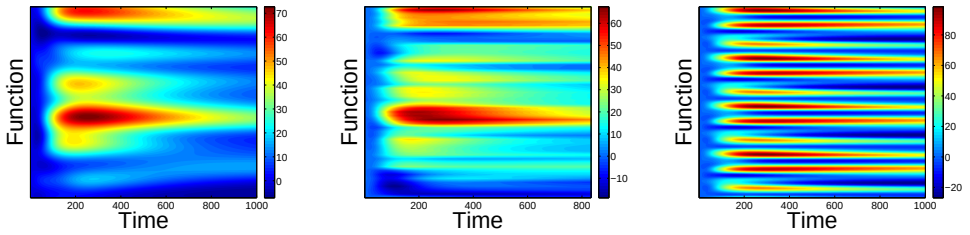


Figure 3: Two types of Hilbert space evolutions. Left: discrete switches in RKHS $\mathcal{H}$; Right: smooth evolution in $\mathcal{H}$.



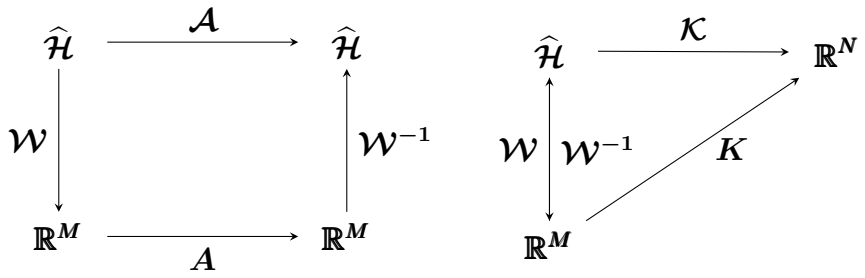
(a) 1-shaded (Def. 1) (b) 2-shaded (Eq. (7))

Figure 4: Shaded observation matrices for dictionary of atoms. Each row represents a sensing location with the color map indicating the evaluation of kernel function w.r.t the others points in the domain.

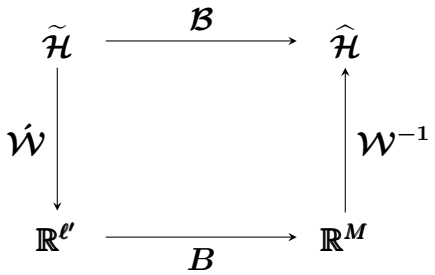


(a) Gaussian (b) Laplacian (c) Periodic

Figure 5: One-dimensional function evolution over a fixed transition matrix A, initial condition w_0 and centers $\mathcal{C}$, but with different kernels $k(x, y)$. Each y-vector at a given value of x represents the output of the function, which evolves from left to right. As seen, changing the kernel creates quite different dynamic behaviors.



(a) Relationship between $\mathcal{A}$ and A (b) Relationship between $\mathcal{K}$ and K



(c) Relationship between $\mathcal{B}$ and B

Figure 6: Commutative diagrams between primal and dual spaces

Algorithm 1 Measurement Map $\widetilde{K}$

Input: $\widehat{A} \in \mathbb{R}^{M \times M}$
Compute Rational Canonical Form, s.t. $C = Q^{-1}\widehat{A}^T Q$. Set $C_0 := C$, and $M_0 := M$.
for $i = 1$ **to** ℓ **do**
  Obtain MP $\alpha_i(\lambda)$ of C_{i-1}. This returns associated indices $\mathcal{J}^{(i)} \subset \{1, 2, \ldots, M_{i-1}\}$.
  Construct vector $v_i \in \mathbb{R}^M$ such that $\xi_{v_i}(\lambda) = \alpha_i(\lambda)$.
  Use indices $\{1, 2, \ldots, M_{i-1}\} \setminus \mathcal{J}^{(i)}$ to select matrix C_i. Set $M_i := |\{1, 2, \ldots, M_{i-1}\} \setminus \mathcal{J}^{(i)}|$
end for
Compute $\mathring{K} = [v_1^T, v_2^T, ..., v_\ell^T]^T$
Output: $\widetilde{K} = \mathring{K} Q^{-1}$

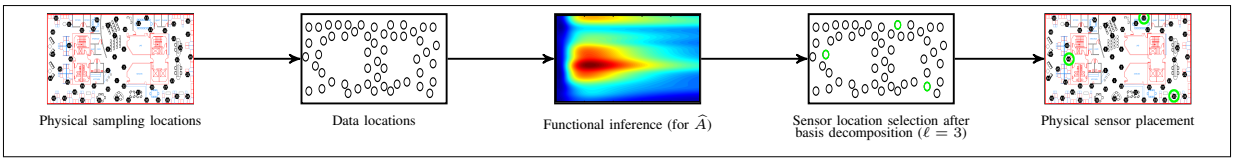


Figure 7: Overall description of how the kernel observer fits in the sensing framework. Physical locations are mapped to data locations, over which historical data is collected as a time series. Functional inference is performed over $\widehat{\mathcal{H}}$ to solve for $\widehat{A}$. The measurement operator K is then computed (see Figure 8), leading to sensor placement.

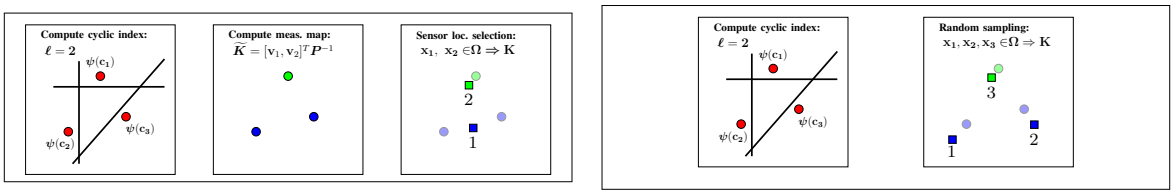
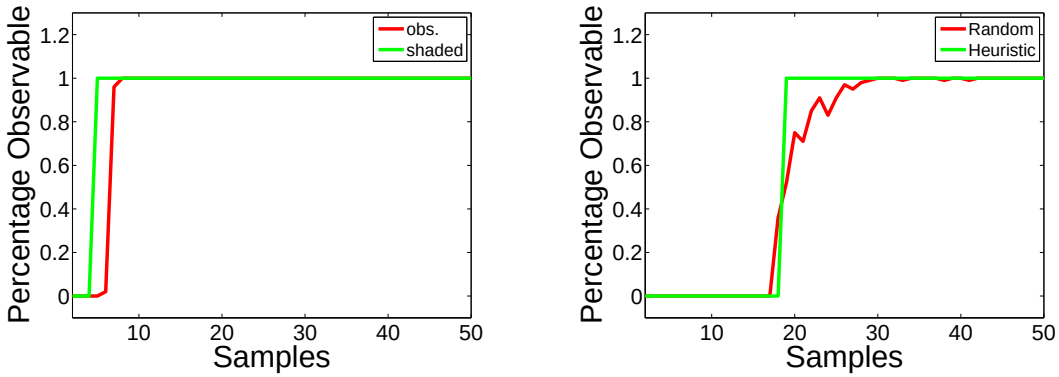


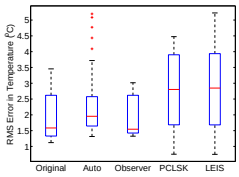
Figure 8: Diagram demonstrating sensor placement using the measurement map or random sampling approaches. The circles represent data locations associated to bases (e.g. $c_j \Leftrightarrow \psi(c_j)$) and the squares represent sensor locations (e.g. $x_i \Leftrightarrow \psi(x_i)$) . The cyclic index ($\ell = 2$) indicates how many possible couplings of bases exist, which can be represented as a choice of $\binom{M}{\ell}$ hyperplanes in Ω. If the measurement map is computed (left), the correct couplings are chosen (green vs. blue), and a smaller number of sensors (2) can be placed. Alternatively, random sampling (right) is more computationally efficient, but generally requires more sensors (3).



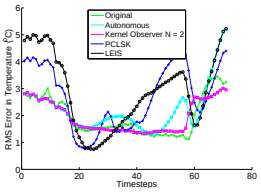
(a) Shaded vs. observability

(b) Heuristic vs. random

Figure 9: Kernel observability results.

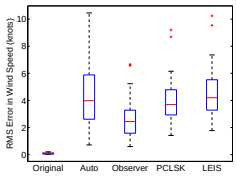


(a) Error (boxplot)

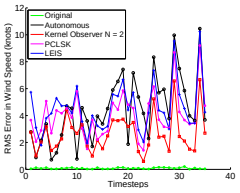


(b) Error (time-series)

Figure 10: Comparison of kernel observer to PCLSK and LEIS methods on Intel Berkeley dataset.

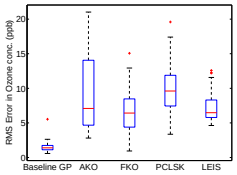


(a) Error (boxplot)

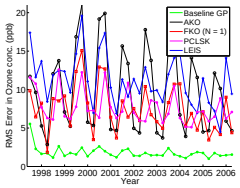


(b) Error (time-series)

Figure 11: Comparison of kernel observer to PCLSK and LEIS methods on Irish Wind dataset.

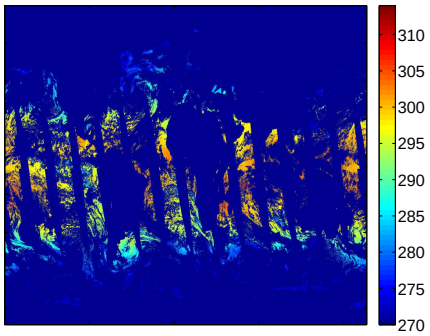


(a) Error (boxplot)

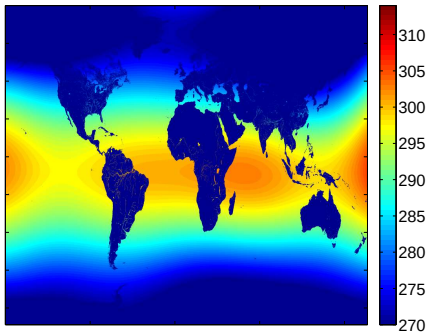


(b) Error (time-series)

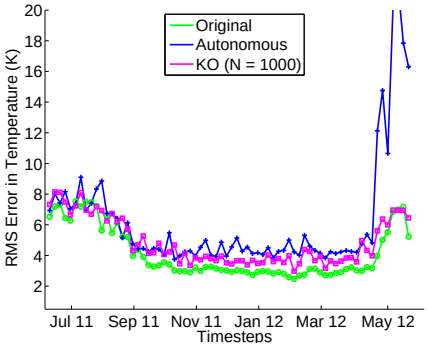
Figure 12: Comparison of kernel observer to PCLSK and LEIS methods on Irish Wind dataset.



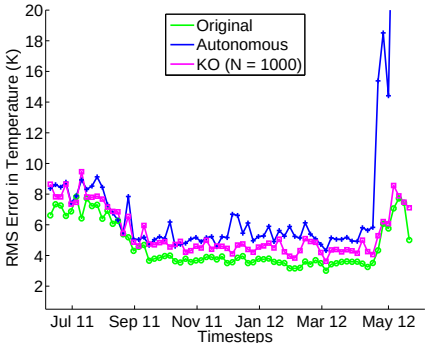
(a) Raw Satellite Data



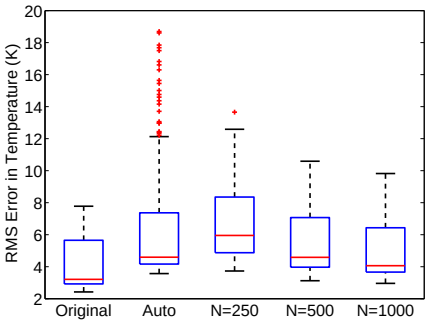
(b) AKO estimate



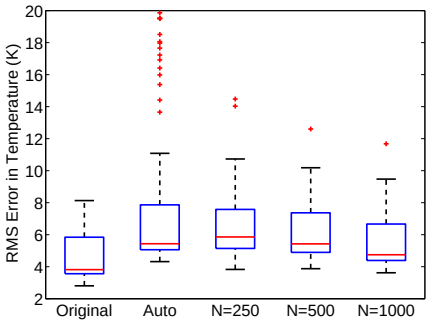
(c) Error-day (time-series)



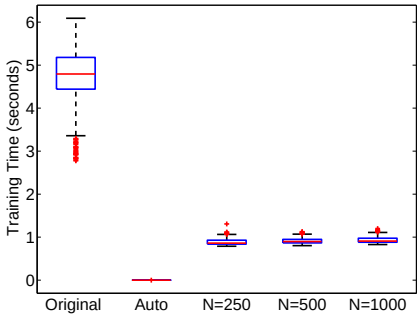
(d) Error-night (time-series)



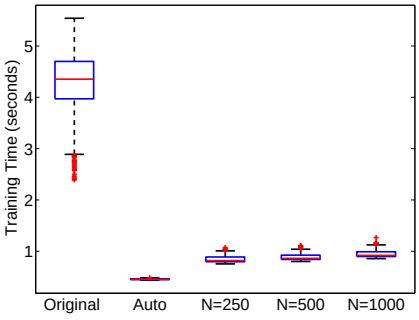
(e) Error-day (box-plot)



(f) Error-night (box-plot)

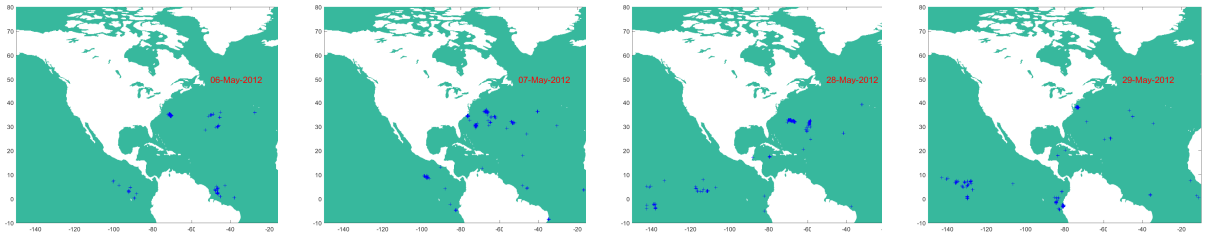


(g) Estimation time (day)



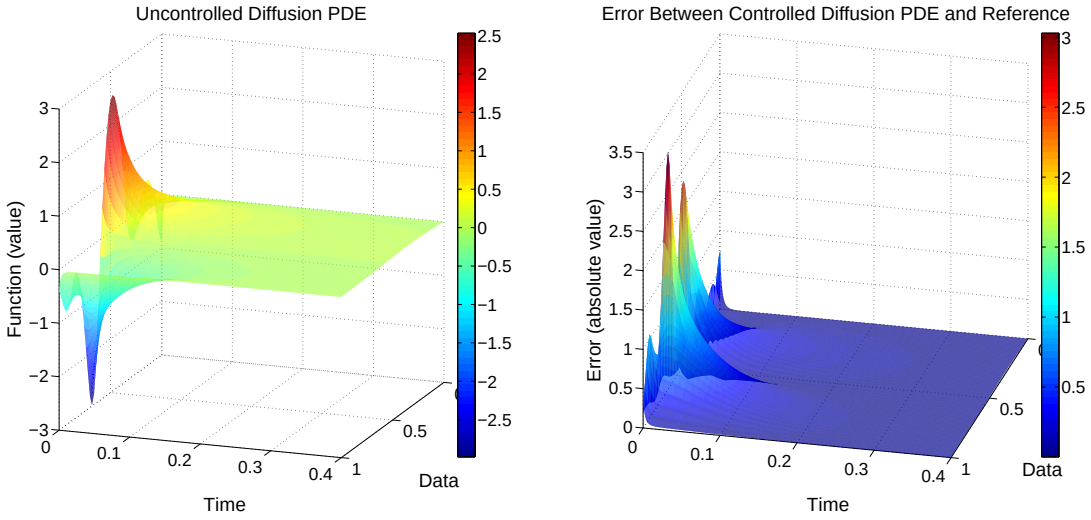
(h) Estimation time (night)

Figure 13: Performance of the kernel observer over AVVHR satellite 2012 data with different numbers of observation locations.



(a) 6th May (b) 7th May (c) 28th May (d) 29th May

Figure 14: Locations of weather anomaly obtained based on the error between the acutual temperature and the prediction of autonomous kernel observer. Landmass is shown in white and the ocean is in green. Locations marked have error greater than two standard deviations above the mean error.



(a) Evolution of initial function f_{init} according to diffusion equation.

(b) Error in absolute value between controlled pde and f_{ref}.

Figure 15: Demonstration of the control of a linear diffusion equation.

Sidebar: Feature Spaces in Machine Learning

Suppose we have data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$, where $x_i \in \Omega_I$ and $y_i \in \Omega_O$. Here, Ω_I is the input domain, and is generally a subset of $\mathbb{R}^D$, although more general sets such as discrete spaces, graphs, or text documents can be considered. Similarly, Ω_O, the output domain, can be just as general as Ω_I. We wish to solve for functions f in some space of functions $\mathcal{J}$ such that $f(x_i) = y_i \ \forall i$. Generally, to restrict the complexity of the space $\mathcal{J}$, a *loss function* $L(f, \mathcal{D}) \mapsto \mathbb{R}$ is chosen, which measures the error between a prediction $f(x_i)$ given a datapoint x_i, and y_i, averaged over the entire dataset $\mathcal{D}$, and the optimization problem becomes

$$f^* = \arg\min_{\mathcal{J}} L(f, \mathcal{D}) + \lambda g(f), \tag{S1}$$

where $\lambda \in \mathbb{R}$, and $g(f)$ represent some constraints on the function f, such as smoothness. Control theorists are most likely familiar with input-output pairs where $x_i \in \mathbb{R}^N$ and y_i is either in $\mathbb{R}$ or $\mathbb{R}^M$ (*regression*). In machine learning, the most common task is when the y_i are discrete (*classification*). Different combinations of task, loss functions, and spaces $\mathcal{J}$ result in different algorithms to solve these problems, which can sometimes form entire subfields of machine learning.

The choice of the function space $\mathcal{J}$ can be critical for the task we want to perform, similar to how the choice of the state space is in control theory. Let's consider a simple example. Suppose we have data from two *classes* $\mathcal{D}_A = \{(x_1^A, y_1^A), \ldots, (x_N^A, y_N^A)\}$ and $\mathcal{D}_B = \{(x_1^B, y_1^B), \ldots, (x_N^B, y_N^B)\}$, where $x_i^{\{A,B\}} \in \mathbb{R}^D$, and $y_i^{\{A,B\}} \in \{-1, +1\}$, shown in Figure **??**. Let f be chosen from the class of linear algorithms, i.e. $f = w^T x + b$, where $w \in \mathbb{R}^D, b \in \mathbb{R}$. We pick a loss L that returns a loss of zero when the prediction is the correct class, and if the prediction is the incorrect class, returns a higher value for misclassifications that are closer to the boundary. A classical example of such a loss is that used by the *perceptron algorithm*, which can be written as

$$L(f, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} \max(0, -y_i w^T x). \tag{S2}$$

This loss measures how accurate the prediction of the perceptron is on average. The general algorithm is as follows:

1) Initialize $w \in \mathbb{R}^D$ to all zeros.
2) For a fixed number of iterations, or until some stopping criterion is met:
    a) For each training example (x_i, y_i),
        i) Let $\hat{y}_i = \text{sgn}(w^T x_i)$.
        ii) If $y_i \neq \hat{y}_i$, update $w \leftarrow w + y_i x_i$.

The perceptron was one of the first machine learning models, and the genesis of modern neural networks [38]. Figure **??** shows a visual representation of where the perceptron algorithm can solve for the decision boundary with zero error. However, if the structure of the data has some nonlinearities, no solution will be found, as seen in Figure **??**. In this case, the original space the data resides in is, in some sense, not a rich enough representation. If we could construct a mapping of the data to a different space which gives a learning algorithm more degrees of freedom to work with, linear algorithms can still be deployed. If we map the same data using a nonlinear map $\phi(x, y) := (x^2, y^2, 2xy)$, the perceptron now finds a solution in 3 dimensions, as seen in Figure **??**. This example shows why so much of the work in machine learning focuses on learning the right representation for the data, for the right representation makes the classification task easy. Two major threads of research in the arena of feature maps over the last 40 years are kernel methods, and neural networks, the latter of which has gained remarkable notoriety in the last 10 years. These lines of research represent distinctly different strategies for generating feature maps from data.

In Figure **??**, the data was mapped using an explicit feature map. Kernel methods utilize an elegant strategy for generating feature maps from data, using a remarkably simple trick called *the kernel trick*. Given a positive-definite kernel function $k(x, y) : \Omega \times \Omega \to \mathbb{R}$, Mercer's theorem guarantees the existence of a feature map $\psi : \Omega \to \mathcal{H}$, where $\mathcal{H}$ is a *reproducing kernel Hilbert space (RKHS)*, and the map ψ obeys the property

$$k(x, y) = \langle \psi(x), \psi(y) \rangle_{\mathcal{H}}. \tag{S3}$$

Recall that since $\mathcal{H}$ is an RKHS, given $c \in \Omega$, $k(x, c) = \langle \psi(x), \psi(c) \rangle_{\mathcal{H}}$, and $k(x, c) := \psi_c \in \mathcal{H}$. Furthermore, $\mathrm{span}\{\psi_x\}_{x \in \Omega}$ is dense in $\mathcal{H}$. There exist kernels that generate $\mathcal{H}$s that are extremely high dimensional: for example, the RBF kernel $k(x, y) = e^{-\gamma \|x-y\|^2}$ is infinite-dimensional. This high degree of freedom enables the design of powerful learning algorithms that are linear in $\mathcal{H}$, but nonlinear in the input domain Ω. The canonical example of this is the support vector machine (SVM) [39], but a more instructive example for us is the perceptron algorithm.

Suppose we trained a percetron using $\mathcal{X} = \{x_1, \ldots, x_N\}$, $x_i \in \mathbb{R}^D$ for some D, $y_i \in \{-1, 1\}$. The prediction of the perceptron is $\hat{y} = \mathrm{sgn}(w^T x)$, where $w \in \mathbb{R}^D$. It can be shown that $w = \sum_{i=1} \alpha_i y_i x_i$, where α_i is the number of times x_i was misclassified. This allows us to derive the dual version of this algorithm, because

$$\hat{y} = \mathrm{sgn}(w^T x) = \mathrm{sgn} \sum_{i=1}^{n} \alpha_i y_i \langle x_i, x \rangle_{\mathbb{R}^D}.$$

The dot product $\langle x_i, x \rangle_{\mathbb{R}^D}$ can be replaced with the kernel, leading to the *kernel perceptron algorithm*:

1) Initialize $\alpha \in \mathbb{R}^N$ to all zeros.
2) For a fixed number of iterations, or until some stopping criterion is met:
    a) For each training example (x_j, y_j),
        i) Let $\hat{y}_i = \operatorname{sgn} \sum_{i=1}^n \alpha_i y_i k(x_i, x_j)$.
        ii) If $y_i \neq \hat{y}_i$, update $\alpha_j \leftarrow \alpha_j + 1$.

This algorithm is nonlinear in the input domain, but linear in the feature space $\mathcal{H}$, which led the deep learning community to, somewhat pejoratively, label this as an example of a *shallow learning architecture*. Kernel methods can also be used in a more direct fashion: if we have a subspace $\mathcal{H}' \subset \mathcal{H}$ with a basis generated from $\mathcal{C} = \{c_1, \ldots, c_M\}$, i.e. $\mathcal{H}' = \operatorname{span}\{\psi_{c_1}, \ldots, \psi(c_M)\}$, a linear model in $\mathcal{H}'$ is again given by a vector $w \in \mathbb{R}^M$. Suppose this weight vector represents a boundary in $\mathcal{H}'$: to compute which side of this boundary a point x would lie on in $\mathcal{H}'$, we simply compute $\operatorname{sgn}(\sum_{i=1}^M w_i \langle \psi(x), \psi(c_i) \rangle_{\mathcal{H}}) = \operatorname{sgn}(\sum_{i=1}^M w_i k(x, c_i))$. The choice of the kernel and its parameters depends on the dataset and the loss function. The kernel and the data together form the feature space. Because kernel methods are linear in their parameters and are restricted to RKHSs, they are amenable to somewhat straightforward mathematical analysis, and are very well studied because of this.

Deep neural networks (DNNs) are algorithms where the function f has nested nonlinearities. Fix a width $M \in \mathbb{N}$. Deep nets are parameterized models with weight matrices $W^l \in \mathbb{R}^{M \times M}$, bias vectors $b^l \in \mathbb{R}^M$, and a pointwise nonlinearity $\phi : \mathbb{R} \to \mathbb{R}$, with $l = 1, \ldots, L$. Vectors $h^l \in \mathbb{R}^M$ are called preactivations, and $x^l \in \mathbb{R}^M$ are called postactivations, each element of which is called a *neuron*. Let $h^0 \in \mathbb{R}^M$ be the input: then the canonical feedforward neural network is given by

$$x^l = \phi(h^l), \;\; h^l = W^l x^{l-1} + b^l. \tag{S4}$$

Therefore, we have that $f(h^0) = x^l$. The individual steps l are called the *layers* of the network, and the nesting property allows these networks to learn much more complicated functions than shallow architectures given the same number of nonlinearities [40]. Different choices of nonlinearities, connections, and layer architectures lead to different types of neural networks, which are used for different applications [41]. Deep learning has had an enormous impact on both the machine learning literature and industrial applications, and that impact has bled over rapidly to other fields. Due to the nested structure of nonlinearities in DNNs, they are more difficult to analyze using simple mathematical tools, and therefore most of the literature in the field has focused on the empirical performance of these methods, where they significantly outperform competing methods.

The signal acheivement of machine learning in general and deep learning in particular has been its ability to weaponize computation to obliterate the obstacles nexessary for industrial

progress. Control needs to provide insight into the structure of the spaces these models generate, to restrict certain forms of output, and to shape decision-making.

Sidebar: Learning Fluid Flows with Evolving Gaussian Processes

2nd order Partial Differential Equations (PDEs) are ubiquitous in practical science and engineering, from mechanics to transport phenomena to electromagnetics. In our view, the Navier-Stokes equations governing fluid dynamics represent most, if not all of the overall complexity of modeling these as (a) it exhibits of hybrid system behavior, e.g. elliptic-hyperbolic, and (b) the nonlinearity results in the complex spatio-temporal dynamics that are 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. [42]

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

We demonstrate the Evolving Gaussian Processes method on CFD data of flow over a bluff body (a cylinder) over a range of Reynolds numbers from 100 to 1000 (the Reynold number is dimensionless flow rate). This deterministic, high-dimensional spatiotemporal dynamical system is well-studied in the fluid dynamics literature, both experimentally and numerically [43]–[45]. The conventional wisdom would be to learn a separate model over each Reynolds number, but our results show that the E-GP method is capable of learning the dynamics of all the flow patterns at once. Using the learned dynamics over weights of successive kernel models, E-GP is capable of predicting the future states of functional evolution in a recursive manner. The key advantage of E-GP 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 values of the weights and the associated linear dynamical systems provide critical insights, such as spatial-correlations through the structure of the transition matrix, local modes of dynamic evolution through invariant subspaces of the transition matrix, and eigenmodes of evolution. The latter is of significant importance to the ongoing work in Koopman operator models of spatiotemporal phenomena.

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 for Re = 100, 300, 600, 800, and 1000. 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 S1,S2(a-d) 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.

We used the Gaussian RBF kernel $k(x,y) = e^{-\|x-y\|^2/2\sigma^2}$ in our E-GP model, with σ estimated to be 0.4. Using a budget of 600 kernel centers (see Figure S4a-S4b, and note how they cluster in the most dynamic regions), we find a 600×600 matrix $\widehat{A}$ which accurately (Figure S3a) 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 S3a. 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.

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 S3a), 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.

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 S3b. 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%.

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.

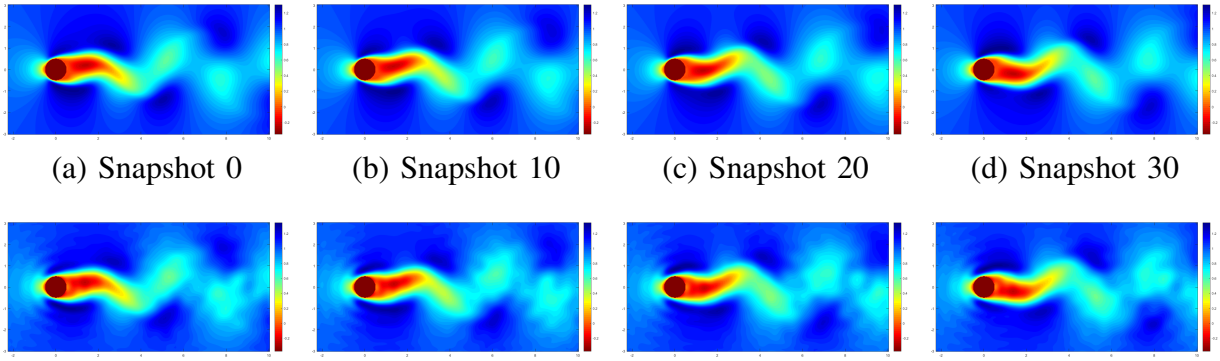
An invariant subspace of a linear operator is a subspace of the Hilbert space such that any vector in the subspace remains in the subspace under transformation by the operator. By marking which kernel centers are associated with different subspaces, we can spatially separate the space into multiple dynamic 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 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 S4a,S4b,S4c. 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 Figure **??**. 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

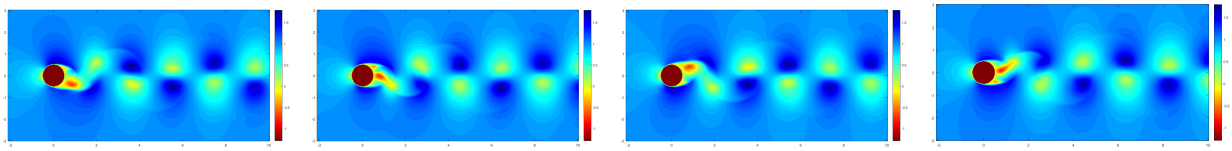
for each of the (relatively) largest elements a_{ij} of $\widehat{A}$, one can see how the system dynamics are coupled spatially (Figures **??**,**??**,**??**). We can also plot the magnitude of a_ij in a third axis for further insight into the most dominant dynamic connection in the system.



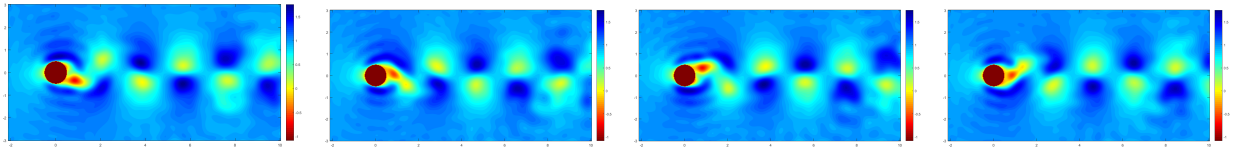
(a) Snapshot 0 (b) Snapshot 10 (c) Snapshot 20 (d) Snapshot 30

(e) Snapshot 0 (f) Snapshot 10 (g) Snapshot 20 (h) Snapshot 30

Figure S1: Visualization of Fluid Flow at Re = 100, CFD (a-d), E-GP (e-h)

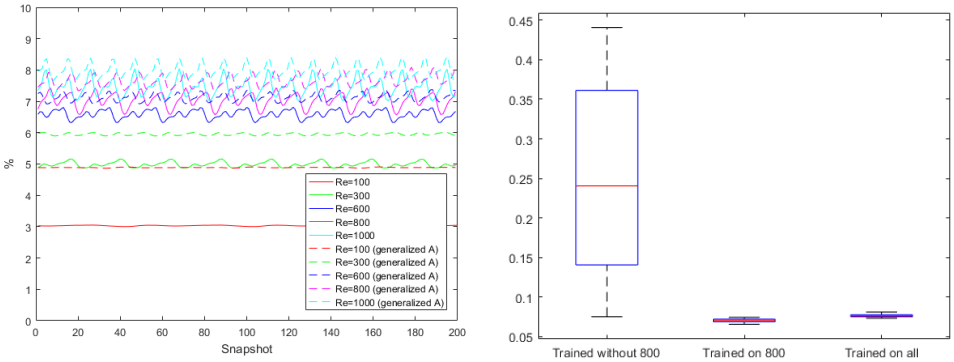


(a) Snapshot 0 (b) Snapshot 5 (c) Snapshot 10 (d) Snapshot 15



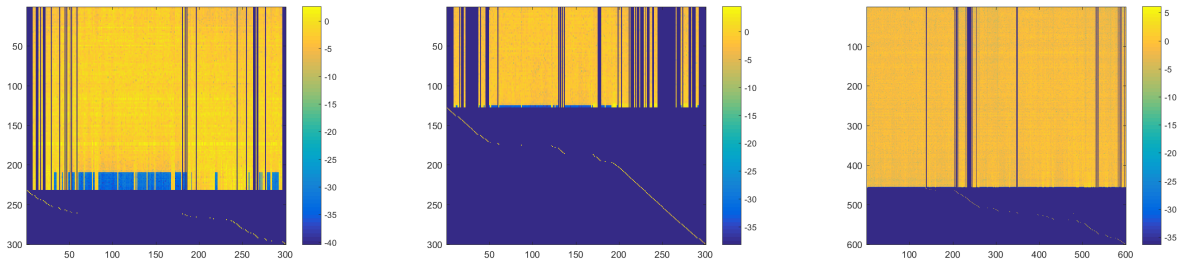
(e) Snapshot 0 (f) Snapshot 5 (g) Snapshot 10 (h) Snapshot 15

Figure S2: Visualization of Fluid Flow at Re = 1000, CFD (a-d), E-GP (e-h)



(a) Universal Generalizer vs Individual Models (b) Different Models Tested on Re=800

Figure S3: Total Percentage Errors



(a) Re = 100, ε = 0.005 (b) Re = 1000, ε = 0.05 (c) All Reynolds numbers, ε =0.069

Figure S4: Eigenvector Heat Maps

Author Biography

Hassan A. Kingravi is a senior research scientist at Pindrop, where he conducts research on machine learning and signal processing for the purposes of preventing fraud in the retail and financial sectors. His research interests revolve around the interplay of control theory, signal processing, and machine learning. He received his PhD in Electrical and Computer Engineering from the Georgia Institute of Technology in 2014.

Harshal Maske is a PhD candidate at the University of Illinois in Urbana-Champaign. He received his Masters and Bachelors integrated degree in Mechanical Engineering from the Indian Institute of Technology Kharagpur, India, in 2009. After completing his masters, he worked for three years at Defense R&D Organization (2009-2012), India and for one year at John Deere (2012-2013). His current research is grounded in the theory at the intersection of Controls and Machine learning with the aim of developing data driven models with theoretical guarantees that enable decision making and control in real-world, complex problems.

Joshua Whitman is a PhD candidate at the University of Illinois in Urbana-Champaign. He received Bachelors of Science degrees in Mechanical Engineering and Mathematics from Oklahoma State University in 2015, and his Masters of Science in Mechanical Engineering from the University of Illinois in 2018. His research is centered in the fields of machine learning, controls, and autonomy, and his work focuses on developing new methods for learning, monitoring, and controlling the dynamics of complex spatiotemporally-evolving systems.

Girish Chowdhary is an assistant professor at the University of Illinois at Urbana-Champaign and affiliated with Electrical and Computer Engineering, Agricultural and Biological Engineering, and the UIUC Coordinated Science Laboratory (CSL). He is the director of the Distributed Autonomous Systems laboratory at UIUC. He holds a PhD (2010) from Georgia Institute of Technology in Aerospace Engineering. He was a postdoc at the Laboratory for Information and Decision Systems (LIDS) of the Massachusetts Institute of Technology for about two years (2011-2013). He was an assistant professor at Oklahoma State University's Mechanical and Aerospace Engineering department (2013-2016). Prior to joining Georgia Tech, he also worked with the German Aerospace Center's (DLR's) Institute of Flight Systems for around three years (2003-2006). His undergraduate institution was the Royal Melbourne Institute of Technology in Australia. Girish's ongoing research interest is in theoretical insights and practical algorithms for adaptive autonomy.