Annual Review of Chaos Theory, Bifurcations and Dynamical Systems
Vol. 9, (2020) 1-26, www.arctbds.com.

Regularized Kernel Machine Learning for Data Driven Forecasting of Chaos

Erik Bollt
Department of Electrical and Computer Engineering
Clarkson Center for Complex Systems Science (C^3S^2)
Clarkson University, Potsdam, NY, USA
e-mail: bolltem@clarkson.edu

Abstract: Forecasting outcomes from initial states continues to be an essential task in dynamical systems theory as applied across the sciences and engineering. The data-driven philosophy has become prevalent across the community. While geometric methods founded in time series to rebuild the underlying geometry based on Taken's embedding theorem have been popular and successful in previous decades, they are complex, computationally expensive, and parametrically intensive. The wave of machine learning methods have come to reveal that a black box oriented approach has a great deal to offer the fundamental problem of forecasting the future. Modelling the flow operator as a linear combination of nonlinear basis functions in terms of regression least squares fitting in a data-driven manner is straight forward to pose. However there are two major obstacles to overcome, which are first, the problem of model complexity may lead to either underfitting or overfitting, but these can be mitigated by Tikohonov regularization. Another serious issue regards computational complexity, which can be overcome by the kernel trick, where in all necessary inner products to be computed in a high dimensional feature (basis function) space occur implicitly within low-dimensional kernel operations. In particular kernel methods from the broader theory of support vector machines is founded in the functional analytic theory of Mercer's theorem and also reproducing kernel Hilbert spaces (RKHS), but practically this fundamental concept in machine learning has become central to many efficient algorithms. Putting these concepts together, the efficiency of kernel methods, and the robustness of regularized regression are both possible within an approach called kernelized ridge regression, that we show here makes for an especially useful way to carry forward time-series forecasting problems, as a simple to use and computationally efficient methodology. We demonstrate the utility of these concepts in terms of a progression of examples from low dimensional where direct analysis is possible, to high-dimensional and spatiotemporally chaotic, and then an experimental data set from physiology of heart rate and breathing interactions.

1 Introduction

Predicting the future is a classic problem in applied dynamical systems. Indeed in science more generally, the problem of forecasting is a central goal. Assuming an initial value problem (IVP),

$$\dot{x} = F(t, x), x(t_0) = x_0, \tag{1}$$

forecasts demand knowledge of the flow $x(t) = S_t(x_0)$, a function that relates outcomes in terms of inputs, $(t, x_0) \to x(t) = S_t(x)$. Whether a "closed form" solution is derived analytically, or an approximate solution is found by computational methods such as finite difference methods, the goal is to access the function $S_t(x_0)$ and we write $S_t : U \subset \Omega \to V \subset \Omega$, where U, V are subsets, $U = V$ if an invariant set, that may represent an attractor in a d-dimensional phase space Ω, for example, $\mathbb{R}^d$.

A data-driven approach to forecasting takes a somewhat different assumed starting point regarding the role of data, given or assumed as examples of outcomes, rather than relying directly on assuming the underlying differential equation. These examples on inputs and outputs may be collected from physical experiments, or otherwise from computational simulations, and they implicitly express the flow as many instances of inputs x_0 initial conditions, and outputs $S_t(x_0)$ downstream by the flow. Building a general model, based on fundamental laws, yielding a differential equation which describes the motion of states, may not be necessary given enough data. "This requires strong assumptions. A good fit of the data to the model validates the assumptions," [61]. Weigend and Gershenfeld make a distinction between weak modeling (data-rich and theory-poor) and strong modeling (data-poor and theory-rich). [61]. This is related to, "... the distinction between memorization and generalization . . ." It is always nice to have a general theory from which we may write down a global set of equations of motion. However, this is not always necessary if the goal is satisfied by producing forecasts. A scalar time series to represent the attractor when the time series is considered with enough delays, where enough may be taken in terms of the phase space, or even as related to the fractal dimension of the attractor, [48]. Forecasting can be done within this now classical framework by local-linear regression, [1, 41, 18, 6, 7, 20, 50], building an atlas of charts for an underlying manifold [40] in terms of a local "k-nearest" neighbor model for forecasting each point . While a data intensive approach may require many parameters to decide, such as embedding dimension, delay, k the number of near neighbors, and the many local polynomial models,[6], the method had some excellent success for forecasting, characterizing qualitative features [9], and popularity. Now, a new wave toward the same basic goal of forecasting enlists a modern machine learning approach which is able to skip many of these details which rather may well be implicit in the overall scheme.

There haven been several major thrusts into data-driven machine learning of dynamical systems, notably including, neural networks [15], and including a special case called reservoir computing or echo-state machines, [43, 31, 38]. This present work is premised on kernel learning methods [49] with ridge regression regularization [51, 25, 60]. Kernel learning allows for general all purpose machine learning methods that include the famous kernel support vector machines, [49, 28]. This approach has excellent and efficient properties that have proven useful in many scientific applications from image analysis, engineering, and neurology, to ecology, [5, 12, 62]. We describe here that this approach

from machine learning applies naturally to the problem of modelling the a flow of a dynamical system, as learned from input data. Notably we emphasized that regularized [56, 53] kernel methods offer benefits over other related works in forecasting for such systems [**?**]. The theoretical foundations are premised on an application of the Mercer's theorem [54, 49], and reproducing kernel Hilbert spaces (RKHS) [11]. As we recall here, the concept relies on feature maps whereby the observations of orbit segments in the phase space are lifted to a high-dimensional, or even infinite-dimensional feature space wherein the problem may be described as linear. This lifting can effectively overcome the otherwise detailed embedding part of time-series forecast if the data set is a scalar observation from a multivariate process. Generally, the computational danger of such a high-dimensional regression in practice is known to suffer from issues of over-fitting, where when too many model features are chosen. Therefore regularization is a concept of inverse problems theory by Tikhonov regularization [58] that generally mitigates the problem. Specifically, the ridge regression, that is a least squares objective together with a least squares regularization turns out to be computationally very convenient within the kernel learning theory formalism.

We assume data which may be the full state observation from the dynamical system, Eq. (1), or a time-delay embedding [55, 48] thereof. In either case, we require data as a large set of input-output pairs observed from the corresponding flow, $((t,x),(t',x')) = ((t,x),(t',S_{t'}(x))$. In practice, this will be a finite sample, $\{(t_i,t_i',x_i,S_{t_i'}(x_i)\}_{i=1}^N$, and we write, $x_i = x(t_i)$, $x_i \in \Omega$, the phase space. For convenience, this data may well be derived from one long single orbit, $t_1 < t_1' = t_2 < t_2' = t_3 < ... < t_{N-1}' = t_N$, or otherwise from several short orbit segments, or even just as stated in terms of single input output pairs of solution segments. We assume that the flow function, $S_t(x)$ has a power series representation, in terms of a basis of functions of monomials $g_n(x)$ of the vector of variables, $x \in \Omega$,

$$S_t(x) = \sum_n a_n g_n(x). \tag{2}$$

The coefficients a_n can be fitted simply within the kernel learning formalism, and the regularized version of the scheme will avoid the problem of overfitting. The data set must be "long enough" to support the method of estimation, as a general failing of any data-driven machine learning method is that it will tend to do much better in terms of interpolation than extrapolation. Generalizing to out of sample forecasts will tend to fare much better when the point to be forecasts is close to other observed inputs. So the quality of results can be quite brittle, depending as much upon curating a representative data set as the details of the method used, a statement that was also observed in [10].

In Sec. 2 we will review all the background of regression necessary for forecasting a flow. In Sec. 2.1 we will set the standard notation relevant for the standard least squares stated in matrix notation as we need here. In Sec. 2.2, multi-variate polynomial regression will be reviewed, with the description of how ridge regression regularization is standard to state in the matrix formulation Sec. 2.3 to avoid overfitting, and it is straightforward to do so also in the matrix formulation. Finally with this as background, we will review in Sec. 2.4 the kernel trick, and how the otherwise high-dimensional and computationally intensive version of ridge regression just reviewed can be simply posed to equivalently work the problem entirely in the observation space. With this theory reviewed and in hand,

we will move in Sec. 3 to cover several progressively more intensive examples, starting with logistic map where the input-output space each being one-dimensional allow for a complete plotting of the outcome and errors, and also closed form representation of moments allow for more detailed analysis of error. Then we offer the Lorenz equations as a 3-d ODE, the Kuramoto-Sivashinsky system as an example of spatiotemporal chaos in a PDE and finally a breathing-heart rate data set from experiments.

2 Kernel Ridge Regression

In this section, we review in turn material from very standard linear regression, multivariate regression, multivariate polynomial regression, then Tikhonov-ridge regularized regression, each of these as precursor to the method that is central here. This review of standard theory prepares for the presentation toward the kernelized formulation of the forecasting problem along with intrinsic efficiencies. Applications illustrating dynamical systems problems and forecasting will be described in the subsequent Sec. 3.

2.1 Least Squares

The standard linear regression problem, proceeds from input-output pairs, (x, y), first stated for simplicity in a scalar case, $x, y \in \mathbb{R}$. The goal is to derive parameters, intercept and slope b_0, b_1, so that a model line $y = L(x) = b_0 + b_1 x$ is fitted to best describe the data, $\{(x_i, y_i)\}_{i=1}^N$. Best fitting is stated in a least-squares sense by the loss function,

$$J(b) = \sum_{i=1}^{N} |y - b_0 - b_1 x_i|^2. \tag{3}$$

Similarly, for multivariate data, $x \in \mathbb{R}^d, y \in \mathbb{R}$, (which includes the scalar case if $d = 1$) then, for parameters column vector $b = (b_0; b_1; ..; b_d)$ the fitting hyperplane,

$$y = L(x) = b_0 + b_1 x_1 + ... + b_d x_d = b^t [1; x], \tag{4}$$

the last part written as a matrix product, or equivalent to a dot product $b \cdot [1; x]$ between the parameter column vector $b = (b_0; b_1; ..; b_d) \in \mathbb{R}^{d+1}$, and the data column vector $[1; x] = [1; x_1; x_2; ...; x_d] \in \mathbb{R}^{d+1}$. We have been overloading the notation on the subindex for the component index and also for the data vector number. For now forward, we will write x_i to refer to a single i^{th}-indexed vector of N data point $x_i = [x_{1_i}; x_{2_i}; ...; x_{d_i}] \in \mathbb{R}^d$. In matrix form, fitting all the data by a linear hyperplane model with respect to all the data, is collected in a data matrix,

$$X = [\mathbf{x}_1 | \mathbf{x}_2 | ... | \mathbf{x}_N] = [(1; x_1) | (1; x_2) | ... | (1; x_N)] \in \mathbb{R}^{d+1 \times N}, \tag{5}$$

whose columns are the column vector data points $x_i \in \mathbb{R}^d$, but with a scalar "1" appended, $\mathbf{x}_i := (1; x_i) \in \mathbb{R}^{d+1}$ to allow for the offset (bias) term associated with the b_0 parameter. Likewise, an output data matrix,

$$Y = [y_1 | y_2 | ... | y_N] \in \mathbb{R}^{1 \times N}, \tag{6}$$

allows the hyperplane for all the data, gives the model equations as,

$$Y = \beta^t X + \epsilon. \tag{7}$$

β is the $d+1 \times 1$ array of parameters. Since X is $(d+1) \times N$ and $N > d+1$, there generally does not exist a unique solution to $Y = \beta^t X$, The $1 \times N$ vector ϵ may be thought of either as noise, or errors. A typical analysis of variance analysis [17] assumes ϵ is normal. A general candidate vector of parameters b for β has least squares loss function,

$$J(b) = \frac{1}{2}\|Y - b^t X\|_2^2 = \frac{1}{2}\sum_{i=1}^{N}(y - b^t \mathbf{x}_i)^2. \tag{8}$$

If we state the optimal parameter,

$$b^* = \arg\min_b J(b), \tag{9}$$

then it is standard to show [17, 27] that the corresponding normal equations,

$$YX^t = b^t XX^t, \tag{10}$$

result in the least squares solution, written symbolically in terms of the inverse if it exists,

$$b^* = (XX^t)^{-1}XY^t. \tag{11}$$

The accepted way to solve the normal equations in a stable and generally well defined way is better to resort to the Penrose-pseudoinverse in terms of the singular value decomposition, [27], rather than forming the explicit inverse of a large matrix, $(XX^t)^{-1}$, which may well be ill-conditioned, so written, $X^\dagger$, [27]. However, with this understanding we will continue to write formally these pseudoinverses in terms the matrix inverse form of the statement.

2.2 Polynomial Regression

A polynomial least squares solution describes an optimal linear combination of monomial basis functions, $\phi_j(x) : \mathbb{R}^d \to \mathbb{R}$. For example,

$$\phi_j(x_i) = x_{1_i}^{j_1} x_{2_i}^{j_2} ...; x_{d_i}^{j_d},\ j_1 + j_2 + ...j_d = j. \tag{12}$$

Clearly, if $j = 0$, then follows the constant term from the basis function, $\phi_0(x) = 1$ as the full model. In this slightly more general form, a data point (a vector x) is observed by features as the functions collected into a $p \times 1$ column vector of p features,

$$x \mapsto \Phi(x) = [\phi_1(x); \phi_2(x); ...; \phi_p(x)]. \tag{13}$$

Using notation,

$$\Phi_i = \Phi(x_i), \tag{14}$$

and counting, $\Phi_i : \mathbb{R}^d \to \mathbb{R}^p$ from the d-dimensional data space to the p-dimensional feature space, then the data matrix of feature maps, stated in terms of the full data set is written,

$$\Phi = [\Phi_1|\Phi_2|...|\Phi_N], \tag{15}$$

as a $p \times N$ matrix.

From a polynomial model matrix equation, compare to Eq. (7),

$$Y = \beta^t \Phi + \epsilon, \tag{16}$$

the best least squares fit of a multi-variate polynomial therefore follows the loss function, compare to Eq. (8),

$$J(b) = \|Y - b^t\Phi\|_2^2 = \sum_{i=1}^{N} (y - b^t\Phi_i)^2. \tag{17}$$

Allowing $p = d + 1$ and $\phi_1(x) = 1$ and $\phi_j(x_i) = x_{i_{j-1}}$, then $1 < j \leq d + 1$ recovers the hyperplane special case. The general polynomial least squares solution follows by similar arguments as Eq. (11).

$$b^* = (\Phi\Phi^t)^{-1}\Phi Y^t. \tag{18}$$

Now contrast the sins choosing between underfitting and overfitting, and the difficulty of choosing the model of appropriate complexity, which leads to solution of this problem in inverse problems theory by regularization. The choice of how large to make p is very important to the quality and kind of output. See this issue highlighted in Fig. 1. If p is chosen too small, then the data is over simplified, and the optimal fit is nonetheless poor. The underlying process is misrepresented. But if p is chosen too large, and note that for a large enough p then there may be an exact fit, with a curve (hyper-surface if multi-variate) that "wiggles" so as to run through every point. In that case, we have literally modelled the details of a specific detailed sample of noisy data, rather than simplifying to the underlying process. The overfitted model will not generalize to other out of sample data.

2.3 Tikhonov Regularization and Ridge Regression

A good model fit requires that either the correct model complexity (polynomial degree) p should be correctly pre-chosen, which is not a simple task on its own, or otherwise an automatic way to compromise the model fit (called data fidelity) must be made to prevent overfitting so that models will generalize well to out of sample forecasts. See Fig. 1. We will pursue Tikhonov regularization from inverse problems theory [19] to balance between data fidelity and the competing regularity of the fit. A Tikhonov regularization is written as a loss function compromising between two terms,

$$J(b) = D(b) + \lambda R(b). \tag{19}$$

For the multivariate polynomial least squares problem, the choice $D(b)$ is by Eq. (17),

$$J(b) = \frac{1}{2}\|Y - b^t\Phi\|_2^2 + \lambda R(b). \tag{20}$$

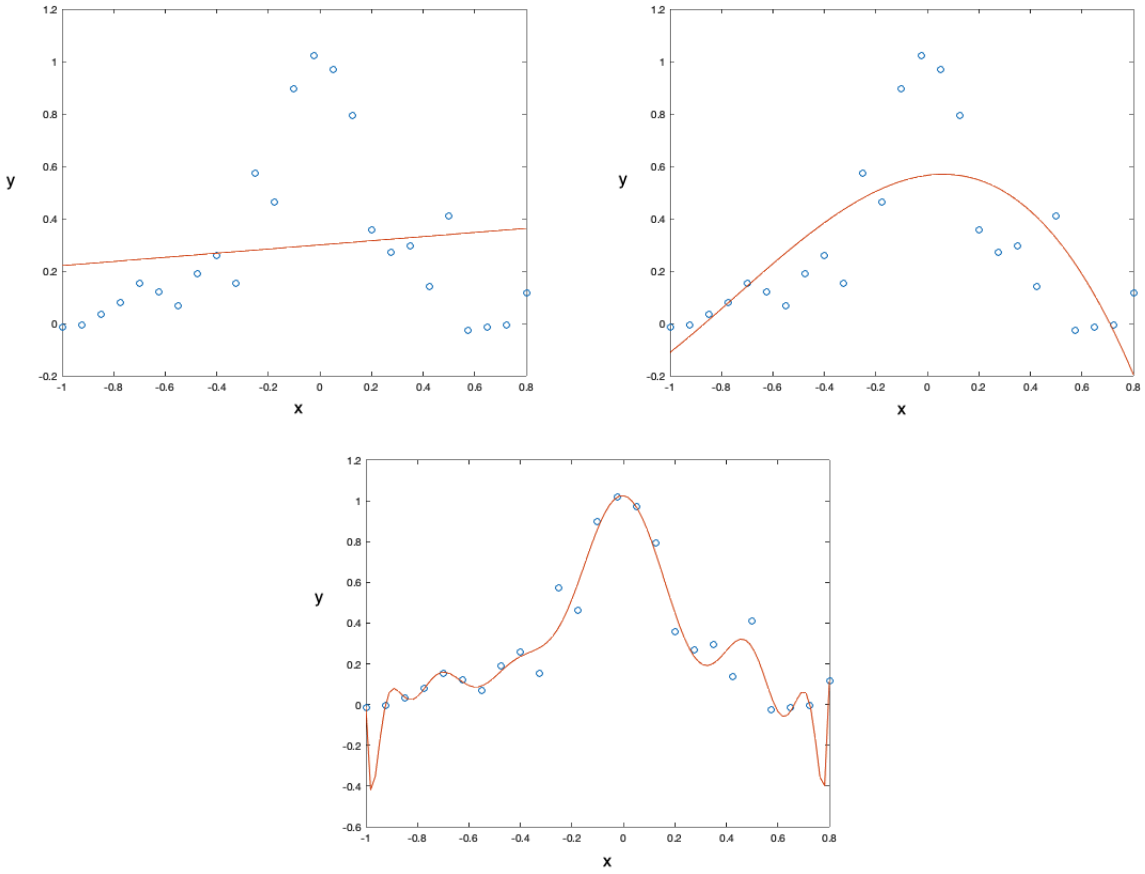

Figure 1: Overfitting is a classic problem in regression modelling. Here, data from a source, $y = \frac{1}{1+25x^2} + \epsilon$, for a normal random variable, $\epsilon \sim \mathcal{N}(0, 0.1)$ generates the $N = 25$ data points shown. (Top Left) Underfitting, due to a polynomial model that is too low. Linear. (Upper Right) Overfitting due to a high degree polynomial model, $d = 16$ begins to attempt to run through all the data points, modelling the noise rather than the process, specializing to the data, but the fitted model will not do as well with out of sample (new) data. (Bottom) A "reasonably" good fit.

The specific form of the regularization term is usually amongst a number of popular choices, each emphasizing different traits of what may be considered a likely model, such as continuity, smoothness, sparsity, or total variation, [59], and each leading to different specific solution. Notably, one popular choice is Lasso regression that emphasizes sparsity, $R(b) = \|b\|_1$, [47]. Instead, however, we will focus here on ridge regression, [27, 39], that can emphasize smoothness,

$$R(b) = \|Cb\|_2, \tag{21}$$

in terms of a Laplacian matrix C. For now we will focus on $C = I$, the identity matrix, so that,

$$J(b) = \frac{1}{2}\|Y - b^t\Phi\|_2^2 + \frac{1}{2}\lambda\|b\|_2^2, \tag{22}$$

defines the ridge regression problem. This regularity term penalizes large b values, that result if p, the polynomial complexity, were chosen too large. A classic picture of this compromise in ridge regression is captured in Fig. 1, the intermediate fit due to intermediate $R(b) = \|b\|_2$.

Ridge regression is also popular over other regularizers for reasons of computational simplicity, in terms of elementary matrix computations. While $Y = b^t\Phi$ is generally ill-posed, adapting from [27] to state for the general polynomial problem here, since,

$$\|b^t\Phi - Y\|_2^2 = \|\Phi^t b - Y^t\|_2^2, \tag{23}$$

then,

$$b^* = \arg\min_b \|(\Phi^t; \sqrt{\lambda}I)b - (Y^t; \mathbf{0})\|_2^2 = \arg\min_b \|\Phi^t b - Y^t\|_2^2 + \lambda\|b\|_2^2, \tag{24}$$

has normal equations,

$$(\Phi\Phi^2 + \lambda I)b = \Phi^t Y^2, \tag{25}$$

whose solution is extremal when,

$$b^* = (\Phi\Phi^t + \lambda I)^{-1}\Phi Y^t. \tag{26}$$

Notice the similarity between this ridge regression solution, and the standard least squares solution, Eq. (17) and how they coincide if $\lambda = 0$ is chosen. Regularizing due to $\lambda > 0$, computationally has the effect stabilize the inverse, with eigenvalues bounded away from zero. Also, if instead of $y \in \mathbb{R}$ is a scalar, but a vector such as $y \in \mathbb{R}^d$, so that x and y might be in the same d-dimensional phase space, then the resulting least squares problem and solution retain the same forms, only in terms of the Frobenius-norm. The matrix version of the standard vector 2-norm, Eq. (26) becomes, $J(b) = \|y - b^t\Phi\|_F^2 + \lambda\|b\|_F^2$. Then follows, b^* from Eq. (26) but with y a matrix of data, and b^* is a matrix of parameters solving the Frobenius norm ridge problem.

2.4 The Kernel Trick

While the feature space may be very high-dimensional, and correspondingly the data matrices in Eq. (26) are very large, the matrix computations Eq. (26) can be inferred indirectly and much more efficiently through a kernel function. The famous "kernel trick," [49, 28] realizes that whenever Φ appears in Eq. (26), so does its transpose. Consequently, the fitting problem in a very high-dimensional feature space can be inferred without ever explicitly lifting the data beyond its relatively low-dimensional observation space.

First, Eq. (26) can be rewritten,

$$b^* = (\Phi\Phi^t + \lambda I)^{-1}\Phi Y^t = \Phi(\Phi^t\Phi + \lambda I)^{-1}Y^t. \tag{27}$$

This follows a matrix identity,

$$(\Phi\Phi^t + \lambda I)^{-1}\Phi = \Phi(\Phi^t\Phi + \lambda I)^{-1}, \tag{28}$$

shown by multiplying both sides on the left by $(\Phi\Phi^t + \lambda I)$ and both sides on the right by $(\Phi^t\Phi + \lambda I)$. The utility is that $\Phi\Phi^t$ is $p \times p$, and $\Phi^t\Phi$, called the Gram matrix, is $N\times$N,

and so there is a computational complexity savings if $p > N$. Furthermore, the Gram matrix need not be explicitly formed through lifting to features, $x \mapsto \Phi(x)$, but rather by the Gram matrix computed through a kernel function.

The ridge regularized least square forecast for a given x is,

$$y(x) = b^{*^t}\Phi(x) = Y(\Phi^t\Phi + \lambda I)^{-1}\Phi^t\Phi(x), \tag{29}$$

not assuming x is one of the data points, $x \in \{x_i\}$. Notation $\Phi(x)$ refers to Eq. (13), the p-dimensional feature map of x. This describes that the forecast $y(x)$ must lie in the span of the data observations,

$$b^* = \sum_i \alpha_i \Phi(x_i), \tag{30}$$

where α_i are the N component elements of the vector,

$$\alpha = (\Phi^t\Phi + \lambda I)^{-1}Y. \tag{31}$$

The beautiful kernel trick follows the observation that in Eq. (29), Φ, the data matrix of features never appears alone, but rather only in terms of the Gram matrix as $\Phi^t\Phi$ (sometimes called the kernel matrix).

Consequently $\Phi^t\Phi$ can be written as a kernel function, and decomposed as rows to columns as inner products in the feature space. The point is that all of the values resulting from inner products in the high dimensional feature space can be computed through a kernel function, and much more computationally efficiently and stability than explicitly lifting the vectors x to $\Phi(x)$.

Specifically, we rewrite Eq. (29),

$$y = Y(K + \lambda I)^{-1}\kappa(x), \tag{32}$$

where, $\kappa(x)$ is a vector,

$$[\kappa(x)]_i = k(x_i, x), i = 1, 2, .., N \tag{33}$$

of entries indexed across the data $\{x_i\}_{i=1}^N$ and K is an $N \times N$ matrix,

$$[K]_{i,j} = k(x_i, x_j). \tag{34}$$

Both of these are written in terms of a kernel function, $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$.

Underlying the utility of the concept of this kernel trick, and kernel learning theory (including the famous kernel support vector machines), [49, 28] are the functional analysis theoretical Mercer's theorem and also reproducing kernel Hilbert spaces. We review the relevant parts of this theory here. The outcome may be stated roughly that given a kernel function, evaluation of inputs into the function correspond to inner products in another, high-dimensional or even infinite dimensional Hilbert space. A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is defined to be a kernel if and only if it is continuous, symmetric and positive semi-definite. By Mercer's theorem, [49], such a function implicitly computes inner products,

$$k(u, v) =< u, v >= \Phi(u)^t\Phi(v), \tag{35}$$

corresponding to a feature map $\Phi : \mathcal{X} \to \mathcal{H}$ in a Hilbert space $\mathcal{H}$. Connecting to the discussion above, $\mathcal{X} = \mathbb{R}^d$ is chosen to be the observation space, and $\mathcal{H} = \mathbb{R}^p$. Mercer's theorem

gives that the feature maps are eigenfunctions of a functional, $T_k[f](u) = \int_{\mathcal{X}} k(u,v)f(v)dv$, and $f \in L^2(\mathcal{X})$, positive semi-definite in these terms means,

$$< f, T_k f >_{L_2(\mathcal{X})} = \int_{\mathcal{X}} k(u,v)f(v)dvdu \geq 0 \text{ for all } f \in L^2(\mathcal{X}), \tag{36}$$

The kernel function is symmetric if $k(u,v) = k(v,u)$, for all $u, v \in \mathcal{X}$, and it is positive semidefinite if the Gram matrix is positive semidefinite for every input data. Then there are eigenfunctions $\phi_i \in L_2(\mathcal{X})$ and eigenvalues $\lambda_i \geq 0$ of T_k, so that for all $u, v \in \mathcal{X}$,

$$k(u,v) = \sum_{i=1}^{\infty} \lambda_i \phi_i(u)\phi_i(v), \tag{37}$$

and the series converges uniformly, [49]. Notice this is the infinite dimensional analog of the spectral decomposition theorem for positive semi-definite matrices, [27]. The feature map,

$$\Phi(x) = [\phi_1(x); \phi_2(x); ...], \tag{38}$$

of eigenfunctions, $\phi_i : \mathcal{X} \to \mathbb{R}$ produces a mapping $\Phi : \mathcal{X} \to l^2$ when $\mathcal{H} = L_2(\mathcal{X})$.

For practical application of this kernel learning theory, the central point for us is that the feature map Φ need never be explicitly computed, formed, or even known to the user. We need only that it exists, thanks to Mercer's theorem. Likewise if $dim(\mathcal{H}) > d$ this may represent a significant computational savings. The specific feature map follows the chosen kernel. For example perhaps a most simple example is the polynomial kernel,

$$f(u,v) = (1 + u^t v)^q. \tag{39}$$

For example, if $u, v \in \mathbb{R}^2$ and $q = 2$, we see that,

$$f(u,v) = (1 + u_1v_1 + u_2v_2)^2 = (1 + 2u_1v_1 + 2u_2v_2 + 2u_1u_2v_1v_2 + u_1^2v_1^2 + u_2^2v_1^2) = \Phi(u)^t\Phi(v), \tag{40}$$

where,

$$\Phi(u) = [1; \sqrt{2}u_1; \sqrt{2}u_2; \sqrt{2}u_1u_2; u_2^2; u_2^2], \tag{41}$$

that is a $p = 6$ dimensional feature map corresponding to this choice of the polynomial kernel. The dimension of,

$$p = C(d+q, q) = \frac{(d+q)(d+q-1)...(d+1)}{q!}, \tag{42}$$

grows quickly with q, relative to d, as $\mathcal{O}d^q$). Another common and useful form is the Gaussian kernel,

$$k(u,v) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{\|u-v\|_2^2}{2\sigma^2}}, \tag{43}$$

that corresponds to an infinite dimensional feature space $\mathcal{H} = L_2(\mathcal{X})$, by a feature map that, as it turns out, has a known explicit form written in terms Hermite polynomials, [21], but again we do not need to know that Hermite polynomial form. That said, the

curse of overfitting is avoided by regularizing and eigenvalue stabilizing properties of $\lambda > 0$ due to ridge regression. We get the best of both worlds with ridge

In summary, the magic of the kernel trick reveals that computing data input into a positive semi-definite kernel is equivalent to computing inner products in what may be a very high or even infinite-dimensional feature space. This offers the flexibility for regression to fit complex functions, but the danger being that of overfitting. However in summary, kernel ridge regression enjoys the best of both worlds, flexibility of high-dimensional fitting and regularity to prevent overfitting. In the next section we test these methods for the target problem of this paper which is fitting and forecasting time series.

3 Example Results

We demonstrate the concepts in terms of forecasting data from several example dynamical systems of progressively higher dimensionality and complexity. These serve as benchmark problems, and then we conclude with demonstration of the method for experimental data for heart and breathing in human subjects.

3.1 Logistic Map

The logistic map,

$$x_{n+1} = f(x_n) = ax_n(1 - x_n) \tag{44}$$

is a discrete time map of the unit interval, $x_n \in [0, 1]$, that is widely used in the pedagogy of dynamical system, [16, 3, 8]. It makes a good first example here since the one-iterate discrete-time map is a quadratic, which composed makes higher order polynomials. That is, the r-th iterate, $f^r(x)$ is a 2^rth-degree polynomial. When $a = 4$ the map is well known to be chaotic and 2^rth-degree polynomial is 2^r-to-one.

For this simulation, we iterated the logistic map with a small noise added at each iterate $f(x)+\nu$, $\nu \sim 10^{-5}$ through $N = 5000$ iterates to create a sample data set as a long noisy orbit. See a sample of these in Figs. 2, 3. The by means of kernel ridge regression, we make $t = 1, 2, ..., 5$ progressively into the future forecasts, choosing $\lambda = 10.^{-4}$ to regularize the forecasts. These forecasts are shown in red (top) versus the true orbit in blue, and as the point and its kth-iterate forecast in the bottom. For $t = 1$ iterate, the forecast is almost perfect and the blue and red time series curves are almost overlaid, and the delay map appears almost like the parabola, $ax(1 - x)$. This illustration uses the Gaussian kernel which implicitly has many terms in the $\Phi(x)$ beyond just quadratic, even if we do not need to explicitly know what they are, but the regularizer prevents overfitting. As k is increased from 2 through 5 we see the forecasts becomes successively worse, but the regularizer causes a smoothing that prevents the large variations typical of overfitting. Results are very similar for a polynomial kernel of high degree.

Fig. 4 summarizes errors of forecasting into the future, as we see mean log error squared grows with t, $< |log(error)^2| >$. This is perhaps a rare example where we can exactly compute the expected value of this error in closed form, using the invariant density [36, 8],

$$\rho(x) = \frac{1}{\pi\sqrt{x(1-x)}}. \tag{45}$$

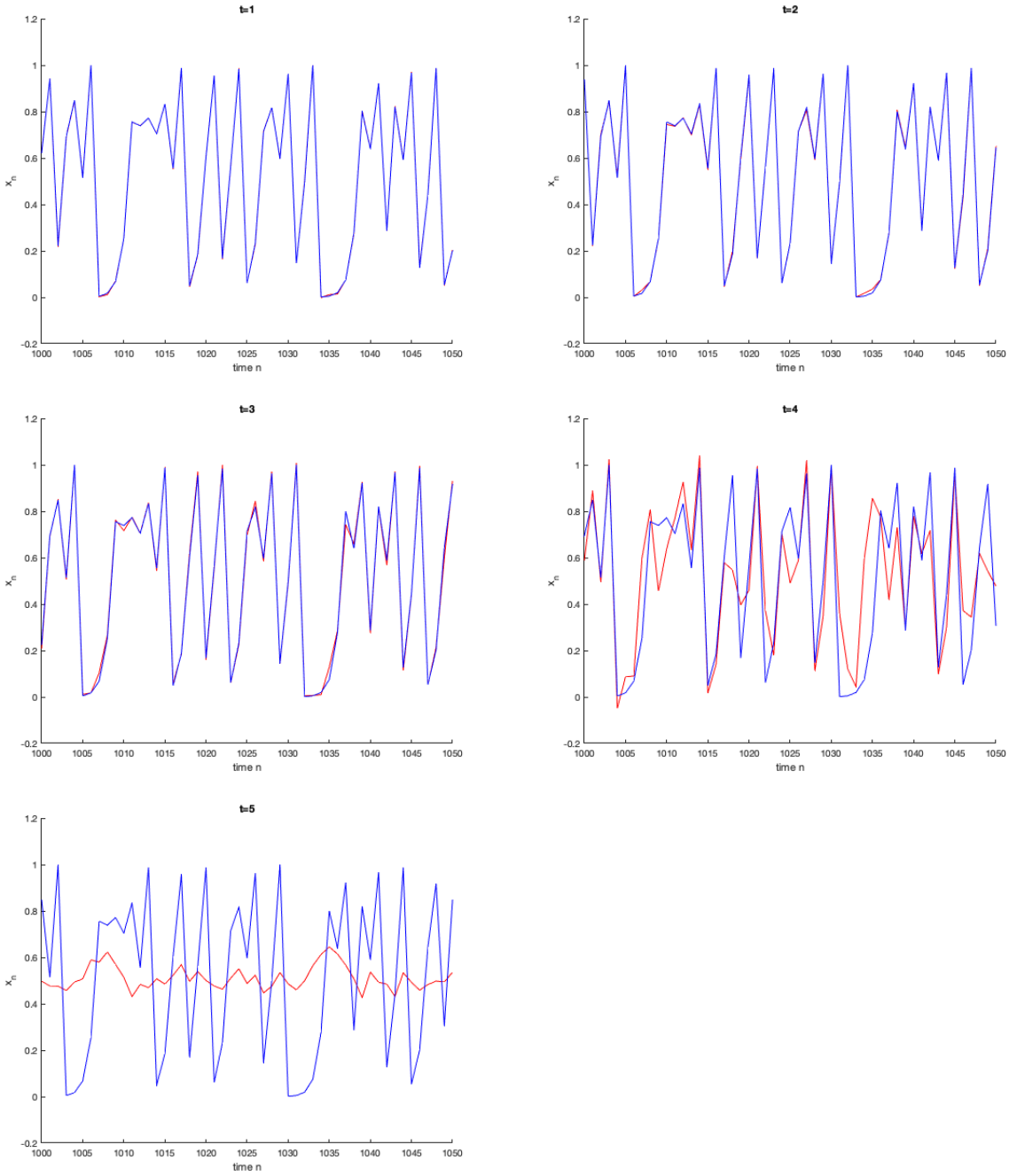

Figure 2: Forecast Logistic Map, Eq. (44) through increasing time. Time series data, x_n for time range shown, $n = 1000, ..., 1050$ from a $N = 1000$ time step sample. True data is shown in blue, and forecast in red. Forecasts for $t = 1, 2, .., 5$ steps into the future. Observe error grows with increasing time horizon, so that by $k = 5$ the forecast is poor. By $t = 4$, and then even more so by $k = 5$, the forecast becomes poor. Thereafter, the theoretical variance, $\sigma = \frac{1}{2\sqrt{2}} \sim 0.35$, is approximately realized related to Eq. (47). See Fig. 3 and also Fig. 4, showing an error plot.

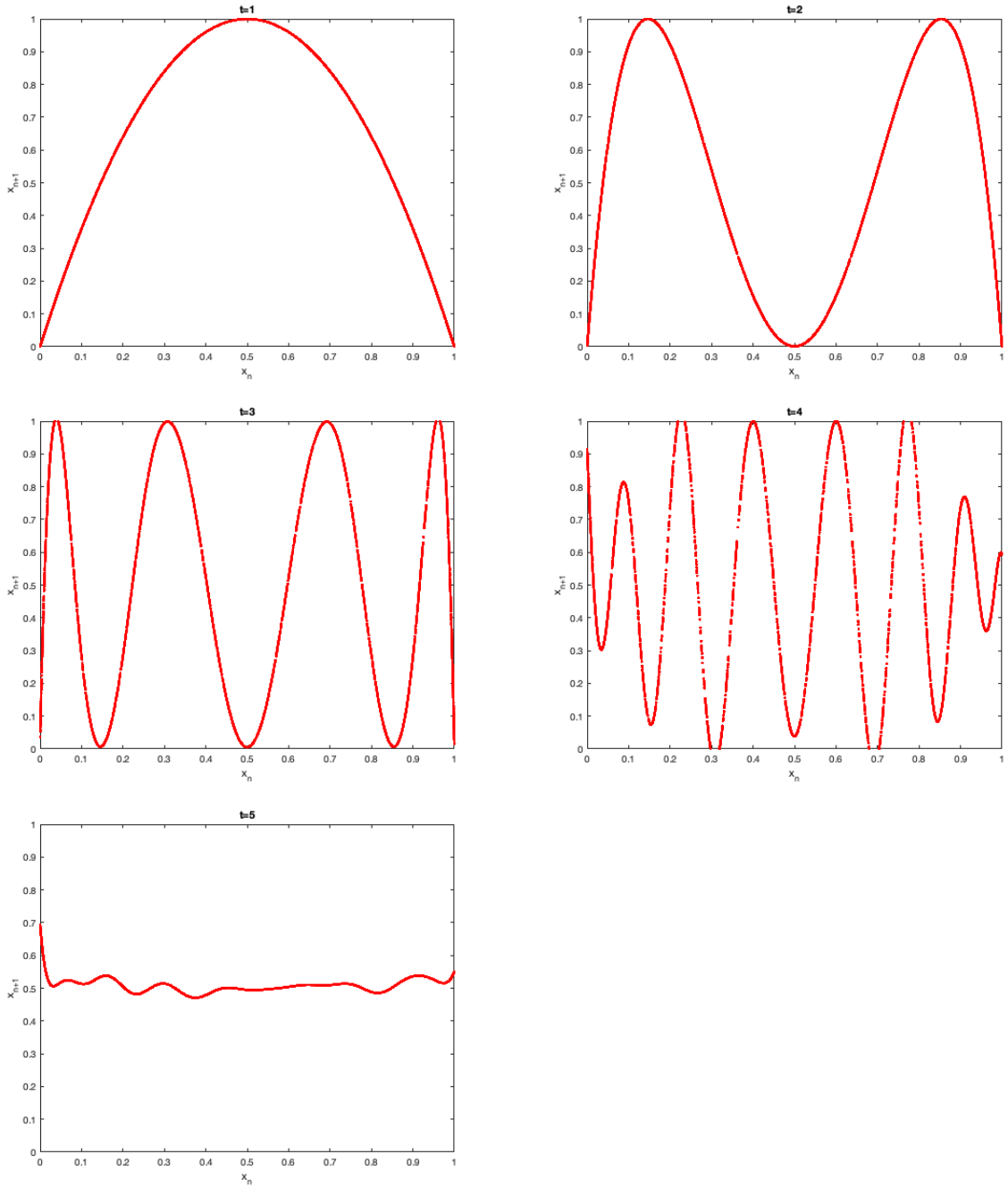

Figure 3: Forecast Logistic Map, Eq. (44) through increasing time. Fitting from time-series data shown in Fig. 3. The delay maps from the forecast (red) data shown. For $t = 1, 2, .., 5$ the delay maps look to be very close to the true quadratic, quintic, and octant, corresponding to iterates of the Logistic map. By $t = 4$, and then even more so by $k = 5$ the forecast becomes poor. Thereafter, the theoretical variance, $\sigma = \frac{1}{2\sqrt{2}} \sim 0.35$, is approximately realized related to Eq. (47). See Fig. 4, showing an error plot.

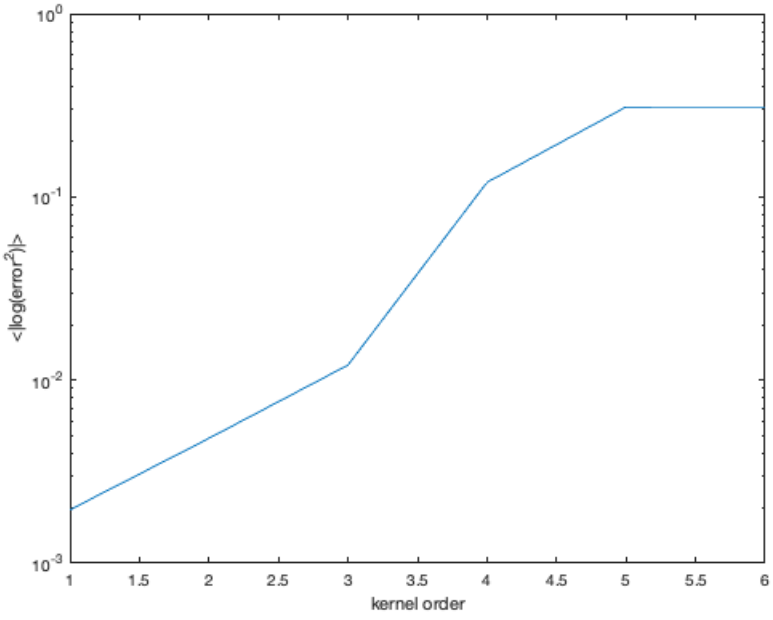

Figure 4: Forecast error growth of the logistic map, with increasing time horizon. k vs. $< log(|error|^2) >$, and average along the orbit. Compare to Figs. 2, 3. Also compare the limiting value to the theoretical variance, Eq. (47).

This is why we have chosen the example as the first benchmark problem. Then we can exchange time averages with space averages versus the invariant density since this example is an ergodic process, [45, 8] the Birkhoff-averages concept applies. The mean value along an orbit is,

$$< x >= \int_0^1 \frac{x}{\pi\sqrt{x(1-x)}} dx = \frac{1}{2}. \tag{46}$$

The theoretical standard deviation from the mean of a long orbit follows,

$$\sigma(x) = (\int_0^1 \frac{(x- < x >)^2}{\pi\sqrt{x(1-x)}} dx)^{\frac{1}{2}} = \frac{1}{2\sqrt{2}}. \tag{47}$$

In Fig. 4, we see just that, where by $k = 6$ the error has grown to full saturation, as the population numerical error, $< |error|^2 > \approx$ 0.3457 is close to theoretical standard deviation $\sigma = \frac{1}{2\sqrt{2}} \approx 0.353553...$ of a random orbit around its mean when all forecastability has been lost, by $t = 5$. This saturation is seen in both Figs. 2, 3 and 4.

3.2 Forecasting the Chaotic Lorenz System

We now consider the Lorenz system [37, 3]:

$$\begin{aligned} \dot{x} &= 10(y - x), \\ \dot{y} &= x(28 - z) - y, \\ \dot{z} &= xy - (8/3)z. \end{aligned} \tag{48}$$

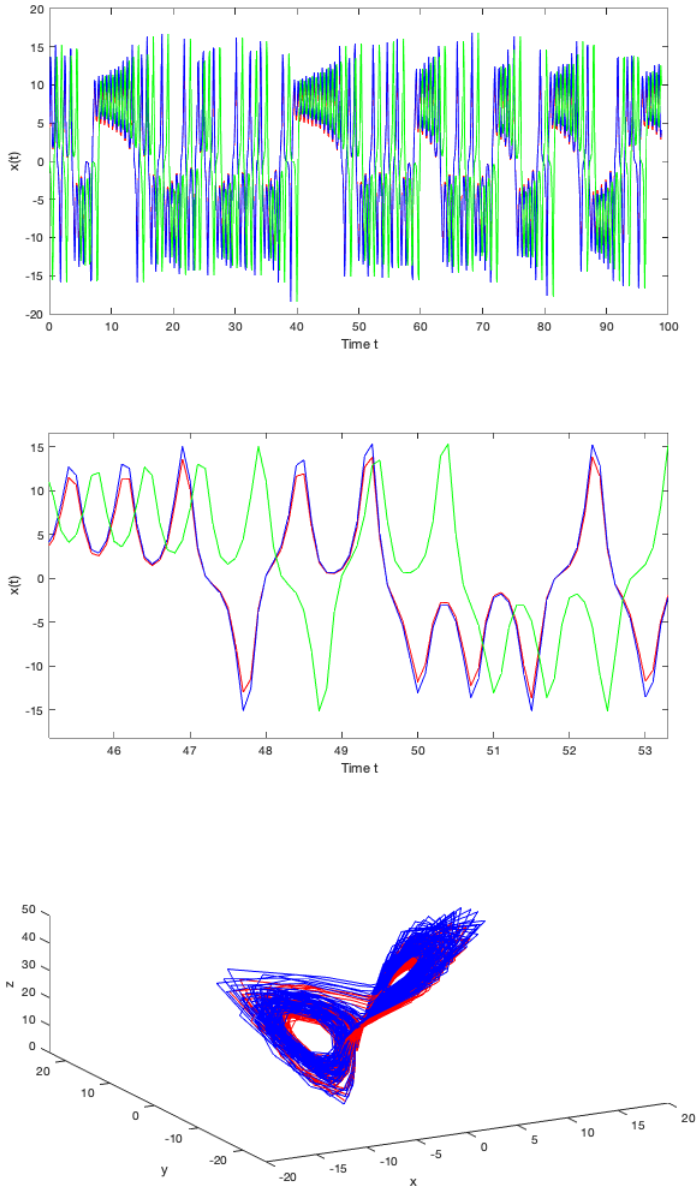

Figure 5: Forecasts of Lorenz Equations, Eq. (48). $\lambda = 10^{-5}$, (Top) Time-Series of $x(t)$ coordinate, blue is true solution at $x(t+\tau)$, $\tau = 10$ shown at time t. Green is $x(t+\tau)$ shown at time $t+\tau$, (appears shifted to the right), and red is forecast values which are seen near the true solution. (Middle) Zoom into smaller time range shown for detail, and errors can be seen more clearly as the difference between the blue and red curves. (Bottom) The full phase trajectory of the ODE shown in $(x(t), y(t), z(t))$ in the time range $t \in [0, 100]$, and the red forecast solution. Contrast to Fig. 6 where $\lambda = 10^{-3}$ plots are shown.

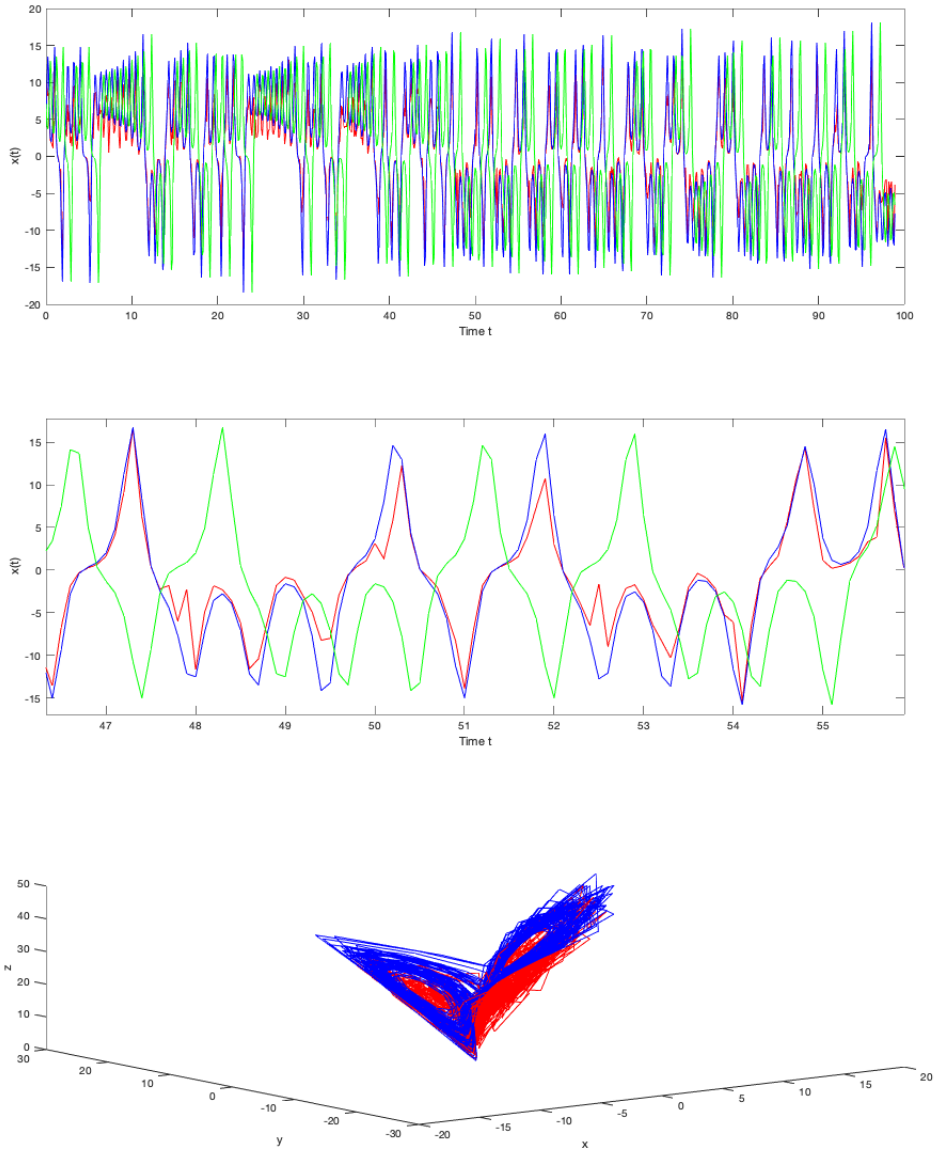

Figure 6: Forecasts of Lorenz Equations, Eq. (48). $\lambda = 10^{-3}$, (Top) Time-Series of $x(t)$ coordinate, blue is true solution at $x(t+\tau)$, $\tau = 10$ shown at time t. Green is $x(t+\tau)$ shown at time $t+\tau$, (appears shifted to the right), and red is forecast values which are seen near the true solution. (Middle) Zoom into smaller time range shown for detail, and errors can be seen more clearly as the difference between the blue and red curves. (Bottom) The full phase trajectory of the ODE shown in $(x(t), y(t), z(t))$ in the time range $t \in [0, 100]$, and the red forecast solution. Contrast to Fig. 5 where $\lambda = 10^{-5}$ plots are shown.

The Lorenz system has been a popular paradigm in the study of chaotic systems and in fact, it can even be physically realized by an electronic circuit [13]. The chaotic attractor in the phase space $\{x(t), y(t), z(t)\}$ is shown in Figs. 5-6. Also shown are regularized forecasts of the $x(t)$ time series, for two different choices of regularity, $\lambda = 10^{-5}, 10^{-3}$ chosen. We see that increasing the regularity less so emphasizes the variations of the true solution, as otherwise those high ordered polynomial terms that become de-emphasized are omitted in the compromise to prevent over-fitting.

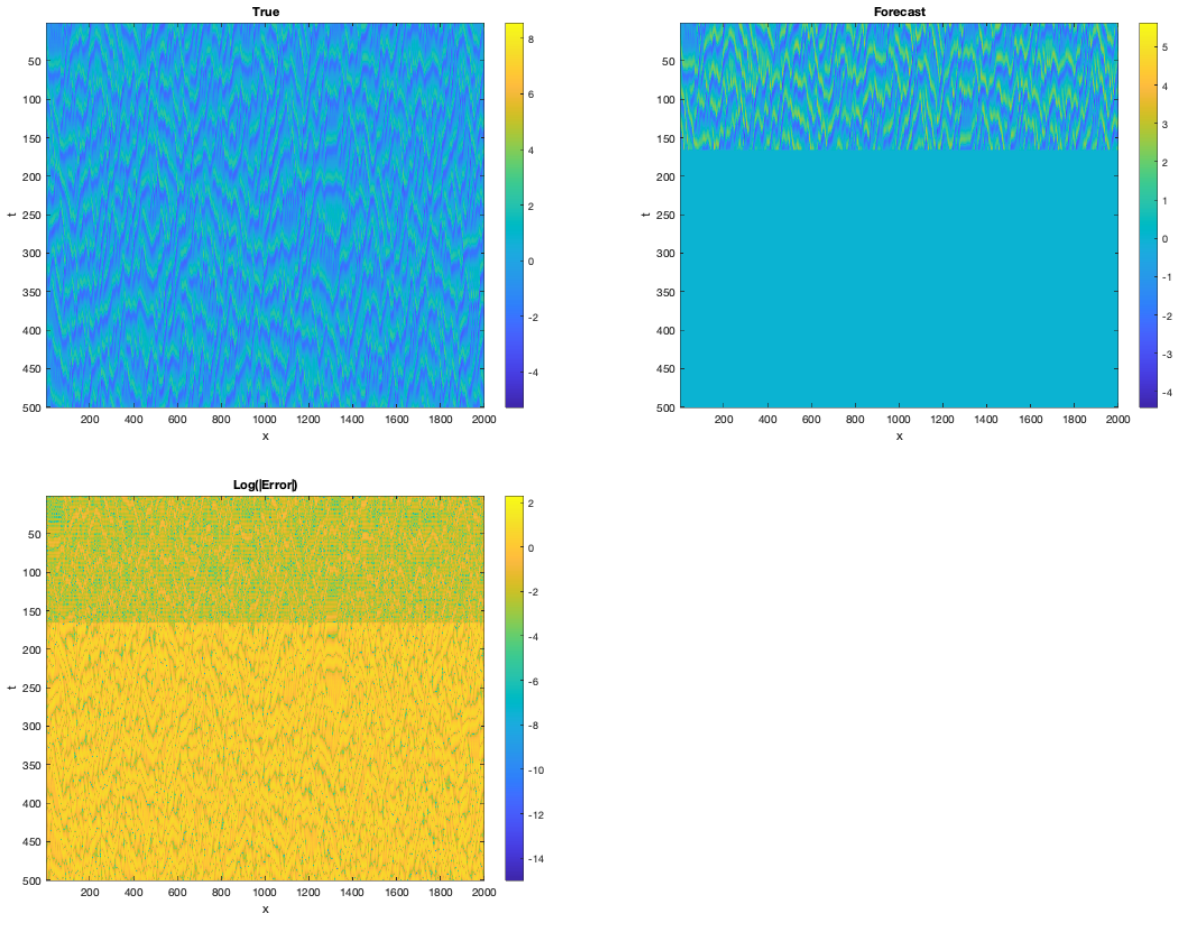

Figure 7: Forecasts of the Kuramoto-Shivasinsky equations, (Left) "True" computed solution form PDE solver of Eq. (49), x vs time t of the solution, but rather than $x \in [0, 2\pi]$ we index the horizontal axis by $N = 2000$ discretized sites as the dimensionality of the associated machine learning problem. (Middle) Forecast into the future from initial condition near that of solution on the left, tracks through 150 time units before amplitude death. (Right) $log(|error|)$.

3.3 Spatiotemporal Chaos in the Kuramoto-Sivashinsky System

The machine learning methods herein are also capable of handling a high-dimensional system, including a PDE which in principle is infinite-dimensional, but in practice the finite

data sampled there from is finite-dimensional. The Kuramoto-Sivashinsky PDE's are a classic example often put forward to the study of spatiotemporal chaos and pattern formation, which we use here to demonstrate forecasting, [14, 29]. The Kuramoto-Sivashinsky (KS) system was developed as a description of flame front flutter [35] has become somewhat of a paradigm of spatiotemporal dynamics, and also one used in particular the study of low-dimensional behavior in high dimensional systems [14, 29] and one for which the inertial manifold theory can be carried far forward analytically, [32, 34, 33, 46], and numerically with the approximate inertial manifold theory [23, 22]. As is typical with inertial manifold theory, there is an apparent very large gap between the minimally provable dimensionality of the inertial manifold, and the apparent dimension of the observed attractor.

The KS equations may be written,

$$u_t = (u^2)_x - u_{xx} - \nu u_{xxxx}, x \in [0, 2\pi], \tag{49}$$

and these may be periodically extended, $u(x,t) = u(x+2\pi, t)$. These equations represent an example of an evolution equation, PDE, which can be written formally as an ODE in a Banach space as follows [14, 29, 4]. Let,

$$u(x,t) = \sum_{k=-\infty}^{\infty} b_k(t) e^{ikx}. \tag{50}$$

Assuming a real u forces $b_k = \overline{b_k}$. Substitution yields infinitely many ODEs for the time varying Fourier coefficients,

$$\dot{b}_k = (k^2 - \nu k^4) b_k + ik \sum_{k=-\infty}^{\infty} b_m b_{k-m}. \tag{51}$$

Restricting to pure imaginary solutions yields, $b_k = i a_k$ for real a_k gives,

$$\dot{a}_k = (k^2 - \nu k^4) a_k + ik \sum_{k=-\infty}^{\infty} a_m a_{k-m}, \tag{52}$$

and restricting to odd solutions, $u(x,t) = -u(-x,t)$ gives $a_{-k} = a_k$. Finally, for computational reasons, it is always necessary to truncate at the N^{th} term, which here means defining $a_k = 0$ if $k > N$.

Assuming a sufficiently large number of modes in our truncation so as to ensure that the data is a good topological model of the attractor allows us to simulate realistic patterns, and so in Fig. 7, we show a 128 mode model simulation. In Fig. 7 are shown x space vs. t time and while $x \in [0, 2\pi]$, instead we index the horizontal axis by $N = 2000$ discretized sites as the dimensionality of the associated machine learning problem. We see the forecast well approximates the true solution for time, until roughly $t > 150$ when abruptly, the forecast becomes poor, and in a manner here of amplitude death. Correspondingly, the $log(|error|)$ shows an abrupt increase. In this example, the regularized kernel ridge regression methodology of machine learning as shown itself well capable of handling even very high dimensional spatiotemporal chaotic problems.

3.4 Nonlinear Time-Series Forecasting for Experimental Physiology Data-Set

Finally, consider a real world experimental data set, from a physiology application that has also become popular in the field of time-series analysis, [44, 30], consisting of heart rate, and breathing rate, over 2500 time samples. See Fig. 8. For each, the observed scalar measurement, $x(t)$ is developed into a time-delay coordinate embedding, $X(t) =< x(t), x(t-\tau), ..., x(t-d\tau) >$, as is standard practice in nonlinear forecasting, [20, 42, 26, 18, 6]. Standard practice in time-delay forecasting is to state an embedding coordinate system, with d "large enough" to embed, but not too large for fitting purposes, and τ chosen to "spread" the data into a useful informative observation, and that last phrase in practice has come to mean in terms of mutual information, [24]. The details of finding each of these important parameters has been the subject of a great deal literature, [57, 52, 1, 2].

While normally it would be important to not choose too large of an embedding dimension, as this too lends to a larger model complexity, the regularizer associated with the kernel method of this work automatically mitigates this issue. So in the spirit of choosing a delay dimension sufficiently large, we allow ourselves to overshoot, by choosing $d = 10$ likely larger than minimal, and τ was chosen as the sampling rate shown. Standard analysis, based on false nearest neighbors suggests 5 dimensions may be sufficient, so $d = 10$ is convenient. Normally this might make for difficulty in forecasting for overfitting reasons, due to the infinite series implicit in a Gaussian kernel method, but regularized here with $\lambda = 0.01$, discovered by cross validation, makes for excellent forecasting properties. See Fig. 8 where excellent forecasting of experimental time series is demonstrated.

4 Conclusions

Machine learning has become a major thrust in the data sciences and many have realized that these tools offer great promise in dramatically advancing what has always been a major goal in the scientific application of dynamical systems, which is a data-driven way to produce good forecasts of future outcomes. We have outlined here where one major thrust in machine learning, based in kernel learning theory can in particular be adapted to handle competing challenges between overfitting the inherent noise, and underfitting or otherwise oversimplifying models. That is, we show that methods of regularized kernel learning are especially well suited to the task of forecasting. Essentially the flow operator associated with a differential equation is a function, and that function may be well approximated in terms of linear combinations of features lifted into a high dimensional Hilbert space. Doing so without explicitly realizing those feature maps as a basis set, is the power promised by kernel trick, and doing so without overfitting is the special aspect associated with the regularized method. We have reviewed all the technical details associated with kernel methodology and the kernel trick, leading from standard regression, through polynomial regression, and Tikhonov regularization, and then the kernelized variant. We have illustrated the success of this approach across several examples of chaotic dynamical systems of increasing complexity and dimensionality. By illustrating in complete with a low dimensional logistic map example, we can show the nature and reason for progressive failure of fitting into forecasting into the future toward the Lyapunov time horizon, and we

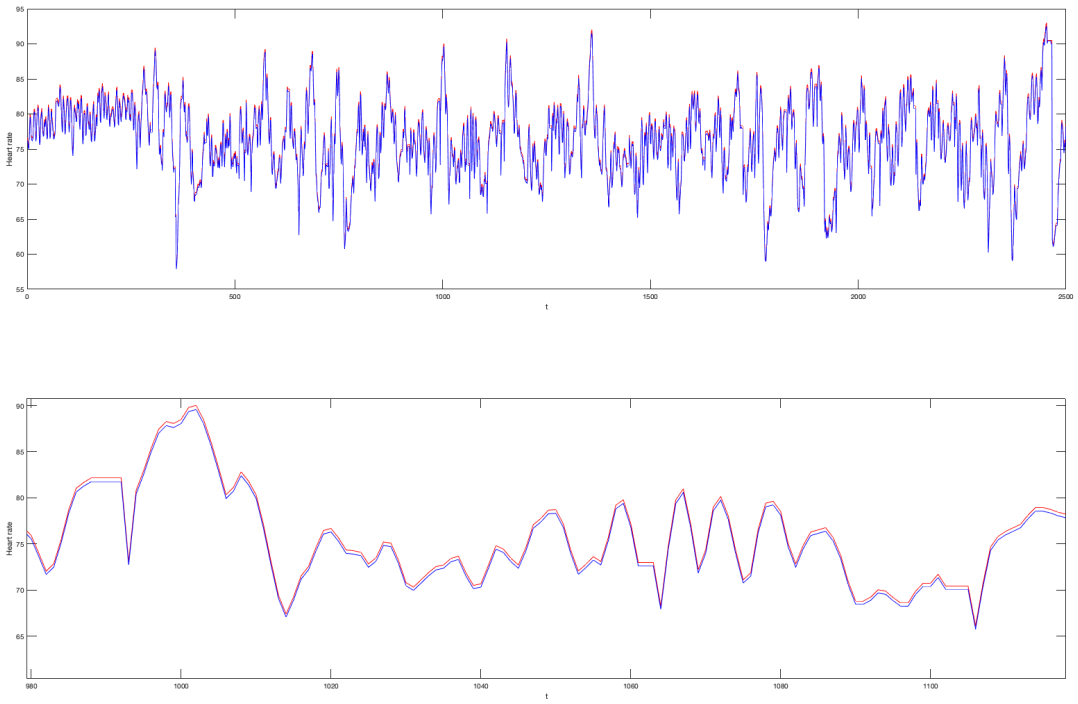

Figure 8: Heart rate and breathing rate forecasting. 2500 data points were forecast, from the time series in the nonlinear time series forecasting classic physiological data base, [44, 30]. Heart rate in this figure and simultaneous and coupled breathing rate data in Fig. 9: Top shows long time series, and bottom, a subset of the time samples for clarity of viewing. Using an over-embedded time delay embedding, and a regularized kernel method, forecasts are shown in red and are close versus the true observations in blue.

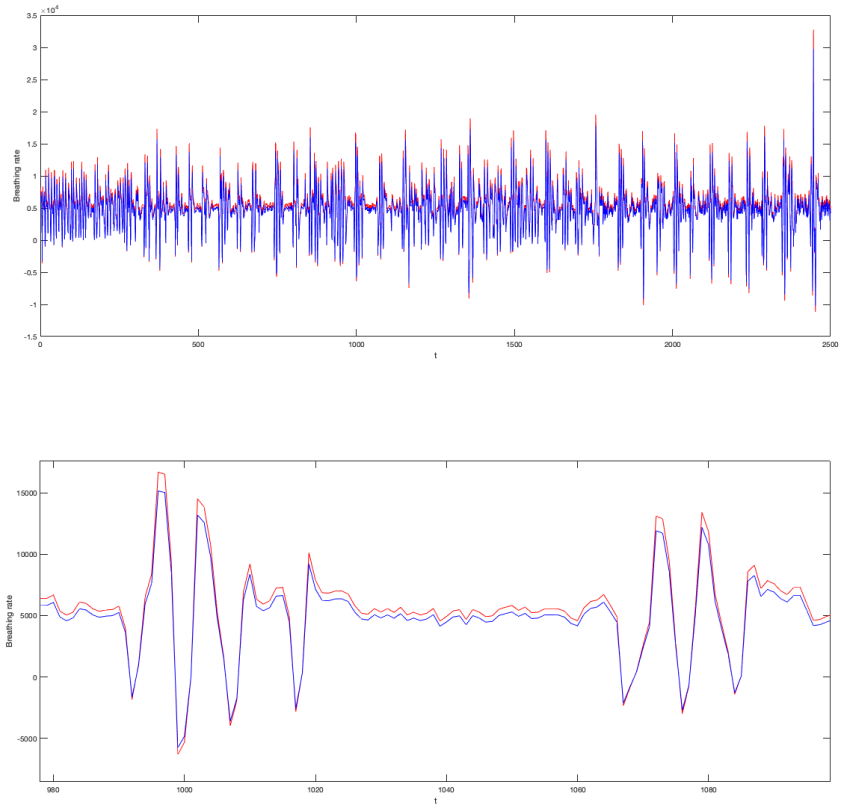

Figure 9: Heart rate and breathing rate forecasting. 2500 data points were forecast, from the time series in the nonlinear time series forecasting classic physiological data base, [44, 30]. Breathing rate in this figure and simultaneous and coupled heart rate data in Fig. 8: Top shows long time series, and bottom, a subset of the time samples for clarity of viewing. Using an over-embedded time delay embedding, and a regularized kernel method, forecasts are shown in red and are close versus the true observations in blue.

can compare to the theoretical limit associated with the ergodic average. Finally we have illustrated the method in terms of real world experimental data set derived from heart rate and breathing data collected from human subjects. We believe that the promise offered by this and other machine learning methods offers a bright new generation of tools beyond what was a first wave decades ago built explicitly upon embedding theory and local models, now with much more of a black box and simple approach. Future applications abound for forecasting across sciences, but also for phenomological modelling as well as possibly for control problems that always require an aspect of forecasting.

5 Acknowledgments

This work was funded by the Army Research Office (Grant No. W911NF-16-1- 0081), and by the DARPA.

References

[1] Henry DI Abarbanel. Modeling chaos. In *Analysis of Observed Chaotic Data*, pages 95–114. Springer, 1996.

[2] Henry DI Abarbanel, TA Carroll, LM Pecora, JJ Sidorowich, and L Sh Tsimring. Predicting physical variables in time-delay embedding. *Physical Review E*, 49(3):1840, 1994.

[3] Kathleen T Alligood, Tim D Sauer, and James A Yorke. *Chaos.* Springer, 1996.

[4] Abd AlRahman R AlMomani, Jie Sun, and Erik Bollt. How entropic regression beats the outliers problem in nonlinear system identification. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(1):013107, 2020.

[5] Senjian An, Wanquan Liu, and Svetha Venkatesh. Face recognition using kernel ridge regression. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–7. IEEE, 2007.

[6] Erik M Bollt. Model selection, confidence and scaling in predicting chaotic time-series. *International Journal of Bifurcation and Chaos*, 10(06):1407–1422, 2000.

[7] Erik M Bollt. Regularized forecasting of chaotic dynamical systems. *Chaos, Solitons & Fractals*, 94:8–15, 2017.

[8] Erik M Bollt and Naratip Santitissadeekorn. *Applied and computational measurable dynamics*, volume 18. SIAM, 2013.

[9] David S Broomhead and Gregory P King. Extracting qualitative dynamics from experimental data. *Physica D: Nonlinear Phenomena*, 20(2-3):217–236, 1986.

[10] Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics*, 52:477–508, 2020.

[11] Claudio Carmeli, Ernesto De Vito, and Alessandro Toigo. Reproducing kernel hilbert spaces and mercer theorem. *arXiv preprint math/0504071*, 2005.

[12] Carlton Chu, Yizhao Ni, Geoffrey Tan, Craig J Saunders, and John Ashburner. Kernel regression for fmri pattern prediction. *NeuroImage*, 56(2):662–673, 2011.

[13] Kevin M Cuomo and Alan V Oppenheim. Circuit implementation of synchronized chaos with applications to communications. *Physical review letters*, 71(1):65, 1993.

[14] Predrag Cvitanovic, Roberto Artuso, Ronnie Mainieri, Gregor Tanner, Gábor Vattay, Niall Whelan, and Andreas Wirzba. Chaos: classical and quantum. *ChaosBook. org (Niels Bohr Institute, Copenhagen 2005)*, 69, 2005.

[15] Kenya Andresia De Oliveira, Álvaro Vannucci, and Elton Cesar da Silva. Using artificial neural networks to forecast chaotic time series. *Physica A: Statistical Mechanics and its Applications*, 284(1-4):393–404, 2000.

[16] Robert Devaney. *An introduction to chaotic dynamical systems*. CRC Press, 2018.

[17] Norman R Draper and Harry Smith. *Applied regression analysis*, volume 326. John Wiley & Sons, 1998.

[18] J-P Eckmann and David Ruelle. Ergodic theory of chaos and strange attractors. In *The theory of chaotic attractors*, pages 273–312. Springer, 1985.

[19] Heinz Werner Engl, Martin Hanke, and Andreas Neubauer. *Regularization of inverse problems*, volume 375. Springer Science & Business Media, 1996.

[20] JD Farmer and JJ Sidorovich. Predicting chaotic attractors. *Phys. Rev. Lett.*, 59:845–848, 1987.

[21] Gregory E Fasshauer and Michael J McCourt. Stable evaluation of gaussian radial basis function interpolants. *SIAM Journal on Scientific Computing*, 34(2):A737–A762, 2012.

[22] C Foias, MS Jolly, IG Kevrekidis, George R Sell, and ES Titi. On the computation of inertial manifolds. *Physics Letters A*, 131(7-8):433–436, 1988.

[23] Ciprian Foias, George R Sell, and Roger Temam. Inertial manifolds for nonlinear evolutionary equations. 1986.

[24] Andrew M Fraser and Harry L Swinney. Independent coordinates for strange attractors from mutual information. *Physical review A*, 33(2):1134, 1986.

[25] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[26] Harold Froehling, James P Crutchfield, Doyne Farmer, Norman H Packard, and Rob Shaw. On determining the dimension of chaotic flows. *Physica D: Nonlinear Phenomena*, 3(3):605–617, 1981.

[27] GH Golub and CF Van Loan. Matrix computations 4th edn (baltimore, ma, 2012.

[28] Martin Hofmann. Support vector machines-kernels and the kernel trick. *Notes*, 26(3), 2006.

[29] James M Hyman and Basil Nicolaenko. The kuramoto-sivashinsky equation: a bridge between pde's and dynamical systems. *Physica D: Nonlinear Phenomena*, 18(1-3):113–126, 1986.

[30] Yuhei Ichimaru and GB Moody. Development of the polysomnographic database on cd-rom. *Psychiatry and clinical neurosciences*, 53(2):175–177, 1999.

[31] Herbert Jaeger. Echo state network. *scholarpedia*, 2(9):2330, 2007.

[32] Michael S Jolly, IG Kevrekidis, and Edriss S Titi. Approximate inertial manifolds for the kuramoto-sivashinsky equation: analysis and computations. *Physica D: Nonlinear Phenomena*, 44(1-2):38–60, 1990.

[33] Michael S Jolly, Ricardo Rosa, Roger Temam, et al. Evaluating the dimension of an inertial manifold for the kuramoto-sivashinsky equation. *Advances in Differential Equations*, 5(1-3):31–66, 2000.

[34] MS Jolly, IG Kevrekidis, and ES Titi. Preserving dissipation in approximate inertial forms for the kuramoto-sivashinsky equation. *Journal of Dynamics and Differential Equations*, 3(2):179–197, 1991.

[35] Yoshiki Kuramoto and Toshio Tsuzuki. Persistent propagation of concentration waves in dissipative media far from thermal equilibrium. *Progress of theoretical physics*, 55(2):356–369, 1976.

[36] Allan J Lichtenberg and Michael A Lieberman. *Regular and chaotic dynamics*, volume 38. Springer Science & Business Media, 2013.

[37] Edward N Lorenz. Deterministic nonperiodic flow. *Journal of the atmospheric sciences*, 20(2):130–141, 1963.

[38] Mantas Lukoševičius. A practical guide to applying echo state networks. In *Neural networks: Tricks of the trade*, pages 659–686. Springer, 2012.

[39] Donald W Marquardt and Ronald D Snee. Ridge regression in practice. *The American Statistician*, 29(1):3–20, 1975.

[40] James R Munkres. *Elementary Differential Topology.(AM-54)*, volume 54. Princeton University Press, 2016.

[41] Daniel B Murray. Forecasting a chaotic time series using an improved metric for embedding space. *Physica D: Nonlinear Phenomena*, 68(3-4):318–325, 1993.

[42] Lyle Noakes. The takens embedding theorem. *International Journal of Bifurcation and Chaos*, 1(04):867–872, 1991.

[43] Jaideep Pathak, Brian Hunt, Michelle Girvan, Zhixin Lu, and Edward Ott. Model-free prediction of large spatiotemporally chaotic systems from data: A reservoir computing approach. *Physical review letters*, 120(2):024102, 2018.

[44] David R Rigney. Multichannel physiological data description and analysis. *Time Series Prediction*, 1994.

[45] Clark Robinson. *Dynamical systems: stability, symbolic dynamics, and chaos*. CRC press, 1998.

[46] James C Robinson. Inertial manifolds for the kuramoto-sivashinsky equation. *Physics Letters A*, 184(2):190–193, 1994.

[47] Volker Roth. The generalized lasso. *IEEE transactions on neural networks*, 15(1):16–28, 2004.

[48] Tim Sauer, James A Yorke, and Martin Casdagli. Embedology. *Journal of statistical Physics*, 65(3-4):579–616, 1991.

[49] Bernhard Schölkopf, Alexander J Smola, Francis Bach, et al. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[50] Christian G Schroer, Tim Sauer, Edward Ott, and James A Yorke. Predicting chaos most of the time from embeddings with self-intersections. *Physical Review Letters*, 80(7):1410, 1998.

[51] John Shawe-Taylor, Nello Cristianini, et al. *Kernel methods for pattern analysis*. Cambridge university press, 2004.

[52] Michael Small and CK Tse. Optimal selection of embedding parameters for time series modelling. In *European Conference on Circuits Theory and Design (European Circuit Society and the Institute of Electrical and Electronic Engineers*, 2003.

[53] Ingo Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, 2005.

[54] Ingo Steinwart and Clint Scovel. Mercer's theorem on general domains: on the interaction between measures, kernels, and rkhss. *Constructive Approximation*, 35(3):363–417, 2012.

[55] Floris Takens. Detecting strange attractors in turbulence. In *Dynamical systems and turbulence, Warwick 1980*, pages 366–381. Springer, 1981.

[56] Ernesto De Vito, Lorenzo Rosasco, Andrea Caponnetto, Michele Piana, and Alessandro Verri. Some properties of regularized kernel methods. *Journal of Machine Learning Research*, 5(Oct):1363–1390, 2004.

[57] James B Vitrano and Richard J Povinelli. Selecting dimensions and delay values for a time-delay embedding using a genetic algorithm. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pages 1423–1430, 2001.

[58] Curtis R Vogel. *Computational methods for inverse problems*, volume 23. Siam, 2002.

[59] Curtis R Vogel and Mary E Oman. Iterative methods for total variation denoising. *SIAM Journal on Scientific Computing*, 17(1):227–238, 1996.

[60] Vladimir Vovk. Kernel ridge regression. In *Empirical inference*, pages 105–116. Springer, 2013.

[61] Andreas S Weigend and Neil A Gershenfeld. Results of the time series prediction competition at the santa fe institute. In *IEEE international conference on neural networks*, pages 1786–1793. IEEE, 1993.

[62] Pengbei Zhang, Chonghoon Lee, Henk Verweij, Sheikh A Akbar, Gary Hunter, and Prabir K Dutta. High temperature sensor array for simultaneous determination of o2, co, and co2 with kernel ridge regression data analysis. *Sensors and Actuators B: Chemical*, 123(2):950–963, 2007.