ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/356818178

Learning dynamical systems from data: a simple cross-validation perspective, part II: nonparametric kernel flows

Preprint · December 2021

DOI: 10.13140/RG.2.2.16391.32164

CITATIONS
0

READS
75

5 authors, including:

Boumediene Hamzi
Imperial College London
68 PUBLICATIONS **422** CITATIONS

SEE PROFILE

Houman Owhadi
California Institute of Technology
162 PUBLICATIONS **2,922** CITATIONS

SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Kernel Methods for Dynamical Systems View project

Machine Learning and Dynamical Systems View project

All content following this page was uploaded by Boumediene Hamzi on 07 December 2021.

The user has requested enhancement of the downloaded file.

Learning dynamical systems from data: a simple cross-validation perspective, part II: nonparametric kernel flows

Matthieu Darcy[1], Boumediene Hamzi[2], Jouni Susiluoto[3], Amy Braverman[4], and Houman Owhadi[5]

[1]Department of Computing and Mathematical Sciences, Caltech, CA, USA. email: mdarcy@caltech.edu
[2]Department of Computing and Mathematical Sciences, Caltech, CA, USA. email: boumediene.hamzi@gmail.com
[3]Jet Propulsion Laboratory, Caltech, CA, USA. email: jouni.i.susiluoto@jpl.nasa.gov
[4]Jet Propulsion Laboratory, Caltech, CA, USA. email: Amy.Braverman@jpl.nasa.gov
[5]Department of Computing and Mathematical Sciences, Caltech, CA, USA. email: owhadi@caltech.edu

December 7, 2021

Abstract

In previous work, we showed that learning dynamical system [21] with kernel methods can achieve state of the art, both in terms of accuracy and complexity, for predicting climate/weather time series [20], when the kernel is also learned from data. While the kernels considered in previous work were parametric, in this follow-up paper, we test a non-parametric approach and tune warping kernels (with kernel flows, a variant of cross-validation) for learning prototypical dynamical systems.

1 Introduction

The ubiquity of time series in many domains of science has led to the development of diverse statistical and machine learning forecasting methods [22, 10, 11, 9, 38, 27, 1].

Amongst various learning-based approaches, kernel-based methods hold potential for considerable advantages in terms of theoretical analysis, numerical implementation, regularization, guaranteed convergence, automatization, and interpretability [12, 30]. Indeed, reproducing kernel Hilbert spaces (RKHS) [13] have provided strong mathematical foundations for analyzing dynamical systems [17, 14, 15, 4, 18, 23, 24, 2, 25, 6, 7, 8, 19] and surrogate modeling (cf. [43] for a survey). Yet, the accuracy of these emulators depends on the kernel, and the problem of selecting a good kernel has received less attention. Recently, the experiments by Hamzi and Owhadi [21] show that when the time series is regularly sampled, or is irregularly sampled [26], kernel flows [29] (an RKHS technique) can successfully reconstruct the dynamics of some prototypical chaotic dynamical systems. KFs have subsequently been applied to complex, large-scale systems, including climate data [28, 46]. A KFs version for SDEs is at [37].

In previous work [21, 26, 28], we used the parametric variant of kernel flows [29] (KF) to tune the kernel used to learn chaotic dynamical systems.

Given a family of kernels parameterized by α, and some input/output data, KF (in its original form [29]) finds a good α (a good kernel) by minimizing (with respect to α, via stochastic gradient descent) the loss

$$\rho = \mathbb{E}\left[\|u - v\|^2_{K_\alpha} / \|u\|^2_{K_\alpha}\right] \tag{1}$$

where $\|\cdot\|_{K_\alpha}$ is the RKHS norm defined by K_α, u is interpolates a random subset of the data, v interpolates a random subset of the previous subset of half-size and $\mathbb{E}$ represents the average with respect to subsampling. KF is a variant of cross-validation in the sense that it operates under the premise that a kernel must be good if the number of points used to interpolate the data can be halved without significant loss in accuracy. The method presented in [29] uses the regression relative error between two interpolants (measured in the RKHS norm of the kernel) as the quantity to minimize.

In this paper, we use the non-parametric version of KFs with this metric along a second metric based on the Maximum Mean Discrepancy (MMD) that is computed from two different samples of a time series or between a sample and a subsample of half-length that was introduced in [21]. The non-parametric version of KFs is essentially a method of kernel warping where samples are displaced along the direction that minimizes a certain metric which in our current work corresponds to the relative error in [29] or the metric based on the MMD introduced in [21].

Kernels of the form $K(\phi(x),\phi(x'))$ defined by a warping of the space ϕ have been employed in numerical homogenization [36] (where they enable upscaling with non separated scales), and in spatial statistics [41, 40, 45, 47] where they enable the nonparametric estimation of nonstationary and anisotropic spatial covariance structures.

Parametric vs. non-parametric. Statistical inference has an inherent tradeoff between robustness and accuracy: these are two conflicting requirements [34, 35, 32, 31, 3]. Indeed a model with a small number of parameters promotes accuracy when well-specified at the cost of a lack of robustness to misspecification. Conversely, a model with a large number of parameters is less likely to be misspecified at the cost of a loss of accuracy compared to a well-specified model with a small number of parameters. We observe this tradeoff when comparing the results of the non-parametric variants of KF to those obtained from parametric variants. [21] observed highly accurate learning of the underlying dynamical systems by using a parametric family of kernels able to capture long and short-range correlations. The warping kernels employed here do not have such information but are expressive to capture it.

Our contributions. The main contributions of this paper are as follows.

- We show that by using the Gaussian kernel and initially choosing the variance that characterizes it, combining (non-parametric) KF with the kriging of the vector field improves the accuracy of the prediction of chaotic time series.
- Training KFs based on the MMD as introduced in [21] gives better results than training KFs with the relative error in [29].
- The choice of the step size in training is important. If it's too large, the relative error diverges. If it's too small, convergence will be very slow.

Structure of the paper. The remainder of the manuscript is structured as follows. We describe the problem in Section 2 and describe two cross-validation metrics to learn the parameters of the kernel used for approximating the vector field of the dynamical system. In section 3, we investigate the performance of these methods for the Bernoulli map, the Hénon map, and the Lorenz system.

2 The problem and its proposed cross-validation solutions

Let $x_1,\dots,x_k,\dots$ be a time series in $\mathbb{R}^d$. Our goal is to forecast $x_{n+1},\dots,x_{n+m}$ given the observation of $x_1,\dots,x_n$. We work under the assumption that this time series can be approximated by a solution of a dynamical system of the form

$$z_{k+1} = f^\dagger(z_k,\dots,z_{k-\tau^\dagger+1}), \tag{2}$$

where $\tau^\dagger \in \mathbb{N}^*$ and $f^\dagger$ may be unknown. Given $\tau \in \mathbb{N}^*$, the approximation of the dynamical can then be recast as that of interpolating $f^\dagger$ from pointwise measurements

$$f^\dagger(X_k) = Y_k \text{ for } k = 1,\dots,N, \tag{3}$$

with $X_k := (x_{k+\tau-1},\dots,x_k)$, $Y_k := x_{k+\tau}$ and $N = n-\tau$. Given a reproducing kernel Hilbert space[1] of candidates $\mathscr{H}$ for $f^\dagger$, and using the relative error in the RKHS norm $\|\cdot\|_{\mathscr{H}}$ as a loss, the regression

[1] A brief overview of RKHSs is given in part I of this paper [21].

of the data (X_k, Y_k) with the kernel K associated with $\mathscr{H}$ provides a minimax optimal approximation [33] of $f^\dagger$ in $\mathscr{H}$. This regressor (in the presence of measurement noise of variance $\lambda > 0$) is

$$f(x) = K(x,X)(K(X,X)+\lambda I)^{-1}Y, \tag{4}$$

where $X = (X_1,\ldots,X_N)$, $Y = (Y_1,\ldots,Y_N)$, $k(X,X)$ is the $N \times N$ matrix with entries $k(X_i,X_i)$, $k(x,X)$ is the N vector with entries $k(x,X_i)$ and I is the identity matrix. This regressor has also a natural interpretation in the setting of Gaussian process (GP) regression: (i.) (4) is the conditional mean of the centered GP $\xi \sim \mathscr{N}(0,K)$ with covariance function K conditioned on $\xi(X_k) = Y_k + \sqrt{\lambda}Z_k$ where the Z_k are centered i.i.d. normal random variables.

Evidently, the accuracy of the proposed approach depends on the kernel K, and one of our goals is to also learn that kernel from the data (X_k, Y_k) with Kernel Flows (KF) [29].

Since the motivation of learning a dynamical system from data is not necessarily about only making predictions but also about emulating the qualitative behaviour of the dynamical systems, we also use a metric based on the Maximum Mean Discrepancy (MMD) to train KFs [21]. The MMD [16] is a distance on the space of probability measures with a representer theorem for empirical distributions which we recall in the appendix. Our strategy for learning the kernel K will then simply be to minimize the MMD

$$\rho_{\text{MMD}} = \mathbb{E}\text{MMD}(S_1,S_2), \tag{5}$$

between two different samples[2], $S_1 = x_{\sigma_1},\cdots,x_{\sigma_m}$ and $S_2 = x_{\mu_1},\cdots,x_{\mu_m}$, of the time series ($\mathbb{E}$ represents, a possibly approximated, average with respect to the subsampling).

2.1 Non-parametric KFs

Write $X := (X_1,\ldots,X_N)$ and $Y := (Y_1,\ldots,Y_N)$ for the input/output training data. Our goal is to learn a kernel of the form

$$K^\phi(x,x') = K(\phi(x,1),\phi(x',1)), \tag{6}$$

where K is a standard kernel (e.g. Gaussian or Matérn kernels) and ϕ maps the input space into itself. The warping of the input space ϕ satisfies the following ODE

$$\begin{cases} \dot{\phi}(x,t) = & v(\phi(x,t),t) \\ \phi(x,0) = & x \end{cases} \tag{7}$$

with

$$v(x,t) = \Gamma(x,q)\Gamma(q,q)^{-1}\dot{q}, \tag{8}$$

and

$$\dot{q} = -\nabla\big[\rho(q)\big], \tag{9}$$

where q corresponds to position variables in $\mathscr{X}^N$ started from $q(0) = X = (X_1,\cdots,X_N)$, Γ is an operator/vector-valued kernel, $\Gamma(q,q)$ is an $N \times N$ matrix with entries $\Gamma(q_i,q_j)$, $\Gamma(x,q)$ is a $1 \times N$ vector with entries $\Gamma(x,q_i)$, and ρ is the kernel flow loss (possibly randomized through-subsampling) (1) or (5) associated with the input/output data (q,Y).

Using an explicit Euler scheme to integrate (7) and regularizing with a nugget $\lambda > 0$ leads to an iteration of the form

$$\phi_{n+1}(x) = \phi_n(x) + \varepsilon v_n(\phi_n(x)). \tag{10}$$

with $\phi_0(x) = x$. Writing $X = (X_1,\ldots,X_N)$ for the training points and $q_n := \phi_n(X) := (\phi_n(X_1),\ldots,\phi_n(X_N))$, the discretized equations take the form

$$q_{n+1} = q_n - \varepsilon\nabla\rho(q_n) \tag{11}$$

and

$$v_n(x) = \Gamma(x,q_n)\big(\Gamma(q_n,q_n)+\lambda I\big)^{-1}(q_{n+1}-q_n)/\varepsilon \tag{12}$$

Note that $\rho(q_n)$ is the kernel flow loss (1) or (5) associated with the input/output data (q_n,Y) where the averaging operation $\mathbb{E}$ can be approximated (via Monte-Carlo sampling from the uniform distribution without replacement) using a finite number of subsamples (possibly reduced to one per iteration in n).

[2] One could also consider the MMD between a sample S_1 of size m and a subsample of S_1 of size $m/2$.

Note also that the proposed (Kernel Flow) algorithm produces a flow ϕ_n (randomized through a sampling of the training data) in the input space, a (stochastic) dynamical system $K(\phi_n(x),\phi_n(y))$ in kernel space and a (stochastic) dynamical system q_n in input space. Since learning becomes equivalent to integrating a dynamical system, it does not require back-propagation nor guessing the architecture of the network, which enables the construction of very deep networks and the exploration of their properties.

3 Numerical experiments

We now numerically investigate the efficacy of the approach described in the previous section in learning chaotic dynamical systems.

3.1 Example: Bernoulli map

We consider the Bernoulli dynamical system defined by

$$x(k+1) = 2x(k) \mod 1. \tag{13}$$

For training, we use a 200 point trajectory with $x_0 = \pi/3$. We choose both K and Γ to be the Gaussian kernel ($K = \Gamma$)

$$K(x,y) = \exp(-\frac{||x-y||^2}{\sigma^2}). \tag{14}$$

We set σ^2 to the average squared distance between all training points and the regularization (nugget) to $\lambda = 10^{-5}$. For testing we use three trajectories with initial values $x_0 = 0.1, 0.99, \pi/10$. We generate 500 points from each trajectory and measure the RMSE. The RMSE for each of the above initialization are denoted R_1, R_2, R_3. Our baseline is Kernel Regression with the kernel K. We first train Kernel Flows non-parametric (with ρ loss function) for 100000 iterations with learning rate $\varepsilon = 2\times 10^{-4}$. The learning rate ε must be chosen small enough so that the explicit Euler scheme (11) is stable and the flow of the dynamical system is averaged through subsampling and large enough so that the discretization remains computationally tractable (an adaptive integrator may be used here to adjust ε to the integration of the potential flow of $\rho(q)$). At inference time, we set $\lambda = 5\times 10^{-3}$. We set the batch size to $N_b = 100$ (50% of the training set).

The results are presented in table 1. We put the change compared to the baseline in parentheses (the baseline is presented in the first line) with the best result highlighted. We repeat the experiment with a larger training set of 800 points. The batch size is set to $N_b = 200$ (25% of the training set). The results are recorded in table 2. We also increase the training iterations to 500000 with a lower learning rate of $\varepsilon = \times 10^{-4}$. Finally, we run the same experiment with the ρ_{MMD} loss function. We train the kernel over 50000 iterations, with a training size of 800 ($N_b = 200$) and a learning rate of $\varepsilon = 1\times 10^{-5}$. This numerical experiment illustrates how the ρ_{MMD} can achieve better results with fewer iterations. Figure 1 illustrates the time dynamics of the first trajectory (for $R2$ and $R3$, see figure 9 in the appendix).

We also illustrate the deformation of the input space X by the flow function ϕ_t for different values of t in figure . We observe that in this simple 1-dimensional case, the flow rotates the data round the x axis(figure 2), which amounts to learning to separate between the two maps:

$$T(x) = \begin{cases} 2x, & \text{if } 0 \le x < \frac{1}{2} \\ 2x-1, & \frac{1}{2} \le x \le 1. \end{cases} \tag{15}$$

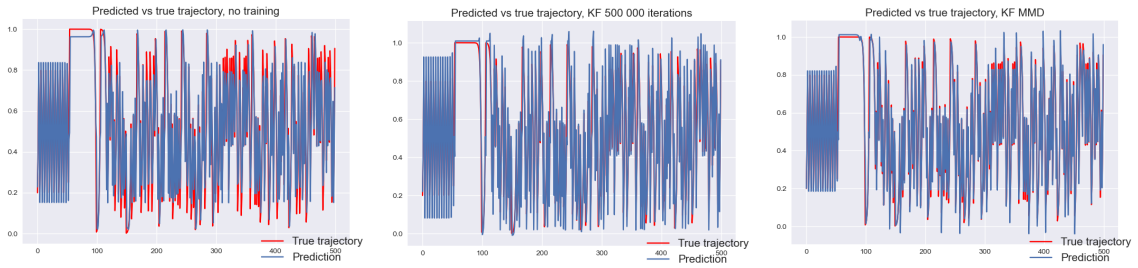
Finally, we illustrate the convergence properties of both loss functions in figure 3. We observe that while both loss functions converge, ρ_{MMD} does so much more quickly than the original ρ, possibly due to the lower noise in the loss. Hence, larger learning rates ε can be used for quicker convergence in a lower number of iterations. The effect of the learning rate is further discussed in section 3.3.

Table 1: Bernoulli map: 200 training points.

Method	R_1	R_2	R_3
Kernel Regression	0.107 (0.0%)	0.119(0.0%)	0.114 (0.0%)
Kernel Flows ρ	**0.102** (-5.3%)	**0.112** (-5.7 %)	**0.103** (-9.3 %)

Table 2: Bernoulli map: 800 training points.

Method	R_1	R_2	R_3
Kernel Regression	0.1053 (0.0%)	0.1168 (0.0%)	0.1125 (0.0%)
Kernel Flows ρ 1×10^5 iterations	0.0990 (-6.0%)	0.1010 (-13.5 %)	0.0994 (-11.7 %)
Kernel Flows ρ 5×10^5 iterations	0.0923 (-12.3%)	0.0924 (-20.8 %)	0.0900 (-20.0 %)
Kernel Flows ρ_{MMD} 5×10^4 iterations	**0.0562** (-46.6%)	**0.0528** (-54.8 %)	**0.0518** (-53.9 %)



(a) Time series (red) and the prediction (blue) without learning for $x_0 = 0.1$

(b) Time series (red) and the prediction (blue) by the learned kernel with ρ for $x_0 = 0.1$

(c) Time series (red) and the prediction (blue) by the learned kernel with ρ_{MMD} for $x_0 = 0.1$

Figure 1: Prediction results for the Bernoulli map $R1$

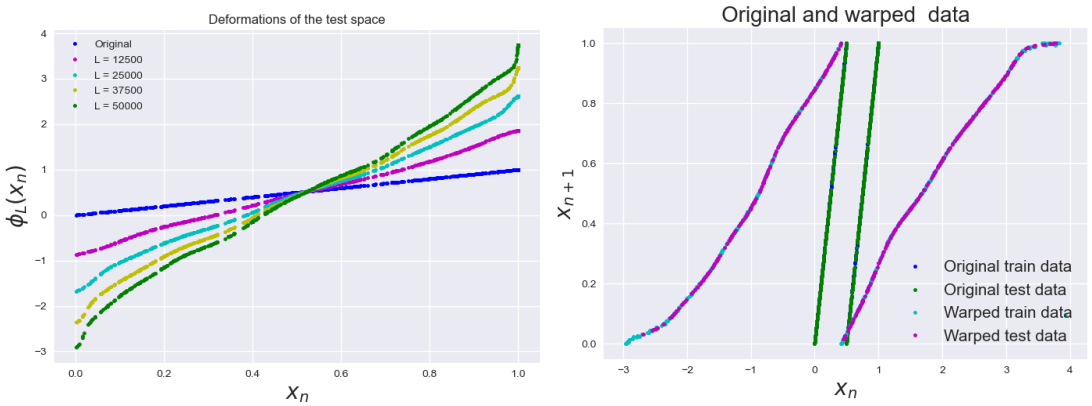


Figure 2: Deformation of input case for different iterations of the flow function ϕ_L (left) and deformed final data (right).

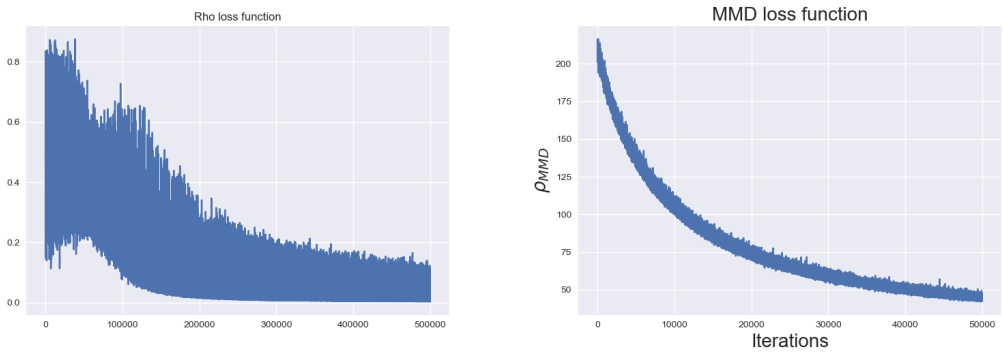


Figure 3: Convergence of the ρ and ρ_{MMD} losses.

3.2 Example: The Lorenz system

Consider the Lorenz system

$$\frac{dx}{dt} = s(y-x), \tag{16}$$
$$\frac{dy}{dt} = rx-y-xz, \tag{17}$$
$$\frac{dz}{dt} = xy-bz, \tag{18}$$

with $s=10$, $r=28$, $b=10/3$. We use the initial condition $(x(0),y(0),z(0)) = (0.0,1.0,1.05)$ and generate 3000 (training) points with a time step $h=0.01$. The test set consists of another trajectory of the same size with initial condition $(x(0),y(0),z(0)) = (0.5,1.5,2.5)$. For the Lorenz system, the inference task is separated into the prediction of the three functions:

$$\begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \mapsto x_{n+1}, \quad \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \mapsto y_{n+1}, \quad \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix} \mapsto z_{n+1} \tag{19}$$

We train and predict using Kernel Flows separately for each of these three functions, and the RMSE for each of the above functions are denoted as R_1, R_2, R_3 For training we use the Gaussian kernel (14) for K with $\sigma^2 = 1.0$ and regularization (nugget) of $\lambda = 10^{-5}$, with a batch size of $N_b = 300$ (10% of the training set). We choose Γ to be the vector valuated kernel $\Gamma = KI_3$ obtained by multiplying K with I_3, the 3×3 identity matrix.

We train KF with both the ρ loss function and ρ_{MMD} (without the averaging operation $\mathbb{E}$). The first is trained with a learning rate of $\varepsilon = 10^{-6}$, the second with a learning rate of 10^{-2}. We compare the performance of the two on all three dimensions of the Lorenz system. Our baseline is again Kernel Regression. The results of the experiment are recorded in Table 3 with the best performance highlighted. The recovered dynamics for the first dimension is provided in figure 4 (see figure 10 for dimensions 2 and 3), and the full dynamics is provided in figure 5.

We observed that in the case of the Lorenz system, the ρ_{MMD} loss function consistently outperformed the original loss ρ and the base estimator. These improvements may be due to the better convergence properties of ρ_{MMD} (in the absence of the averaging operation $\mathbb{E}$): for all three variables, Kernel Flows with ρ_{MMD} converged quickly whereas with a ρ loss function, no convergence was observed (see figure 6 for illustrations). Because of the lack of convergence, KF with ρ loss function does not improve over the base model, but it is possible that with a better learning rate schedule, the ρ loss function will yield better results. However, the choice ρ_{MMD} has the advantage of being much less sensitive to the choice of learning rate and will converge to a good solution for multiple values.

We also considered another way of running KFs with the ρ loss function. Since the Lorenz system has multivariate outputs, we apply a standard recipe to normalize the data so that it is zero mean and unit variance. More specifically, after subtracting the mean, we compute the eigendecomposition of the training (input) data covariance matrix and project our inputs on the three eigenvectors. We then scale each transformed dimension to have unit variance. The warping of the space with training data, and the predictions (warping training data and performing regression for the warped inputs) after that

are both run in this transformed space. This requires carrying out the transformation operation also for the testing data. Since we don't have a maximum covariance parameter in our model, we also perform the same transformation for the training data labels. That way, the marginal variance of our data is consistent with that of the kernel.

As in the other examples, we use a spherical Gaussian kernel. The length scale of the kernel is computed from the average Euclidean distance between the training inputs in the transformed space, and comes down to 2.16. We use small nuggets for training and prediction, $1e-7$ and $1e-6$, respectively.

The warping is computed in non-parametric KF from the instantaneous gradients of the loss function with respect to the training inputs. In some cases, a few gradients dominate; we use gradient clipping to regularize the flow and clip the magnitudes of the gradient vectors at the 0.8th quantile of all gradient magnitudes. The learning rate is scaled adaptively so that at each step, the maximum distance traveled by any of the inputs is 1e-5 in the transformed space. We perform 2000 iterations of KF in this experiment. The results are reported in Table 3.

Table 3: Lorenz system: original data.

Method	R_1	R_2	R_3
Kernel regression.	2.15 (0.0%)	2.57(0.0%)	5.45 (0.0%)
Kernel Flows ρ	2.15 (0.0%)	2.57(0.0%)	3.81 (-30%)
Kernel flows ρ_{MMD}	**1.60** (-26%)	**2.29** (-11%)	**3.81** (-30%)

Table 4: Lorenz system: pre-processed data.

Variable	R'_1	R'_2	R'_3	R'_4
x	0.25788	0.29009	0.01351	0.01714
y	0.86578	1.14942	0.04364	0.05669
z	0.84734	1.23247	0.01543	0.02203

where R'_1 is the RMSE in the warped coordinates, R'_2 : is the RMSE in the original coordinates, R'_3 is the relative error in the warped coordinates, R'_4 is the relative error in the original coordinates. The relative errors are reduced by NPKF training by 21.179, 23.0148, 29.943 percents, due to the training.

3.3 Learning rate and convergence

We now discuss one possible advantage of the ρ_{MMD} loss function over the ρ loss function. As was seen in the previous sections, for a particular dynamical system (the Lorenz system), Kernel Flows with ρ did not improve over the base estimator in our tests, whereas Kernel Flows with ρ_{MMD} achieved significant improvements. For the Bernoulli map, while both versions improved over the base estimator, ρ_{MMD} yielded a better performance and converged faster. One possible explanation is that the algorithm is more sensitive to the choice of learning rate ϵ with ρ compared to ρ_{MMD}.

To illustrate this sensitivity, we retrain the network on the Bernoulli dynamical system with the same parameters as in section 3.1, but with higher learning rates of 2×10^{-4}, 3×10^{-4} for both loss functions and compare the results with the originally trained kernel in tables 5 and 6. For the ρ loss function, we note that higher learning rates do not necessarily yield better results, even though convergence of the loss is achieved in all cases. In some cases, the performance significantly worsened compared to the base estimator.

We suspect that this is due to the instability of the explicit Euler scheme (11) in integrating (9). Indeed there are two potential sources of instabilities when ρ is defined as in (1) and the time steps are too large: (1) The instability of the explicit Euler scheme itself, (2) the fact ρ is not averaged but randomized at each time step (which produces an effective dynamic with the correct average drift when the time steps are small enough but does add instability when they are not).

Due to these instabilities and the inherent stochasticity of the loss function, as observed in [29], it is possible for the learned kernel to converge to so-called "degenerate" kernels if the learning rate

is too high. This may explain why Kernel Flows with ρ did not perform well on the Lorenz system: a high learning rate leads to convergence to a poor solution. We hypothesize that a better choice of learning rate (such as an adaptive learning rate) is required for the kernel to converge to a good solution.

On the other hand, for ρ_{MMD} increasing the learning rate improves the results as long as the kernel achieves convergences. Hence, ρ_{MMD} yields a more stable system which is less sensitive to small changes in hyper-parameter choices. We suspect that this is due to the fact that ρ_{MMD} is not inherently stochastic.

Table 5: Bernoulli map learned with ρ for different learning rates

Learning rate	R_1	R_2	R_3
$\varepsilon = 1 \times 10^{-4}$	0.0923 (-12.3%)	0.0924 (-20.8%)	0.112 (-20.0%)
$\varepsilon = 2 \times 10^{-4}$	0.0940 (-10.9%)	0.159(+36.2%)	0.0938 (-16.6%)
$\varepsilon = 3 \times 10^{-4}$	0.1030 (-2.1%)	0.105(-10.1%)	0.104 (-7.5%)

Table 6: Bernoulli map learned with ρ_{MMD} for different learning rates

Learning rate	R_1	R_2	R_3
$\varepsilon = 1 \times 10^{-4}$	0.0562 (-46.6%)	0.0528 (-54.8 %)	0.0518 (-53.9 %)
$\varepsilon = 2 \times 10^{4}$	0.04883 (-53.6%)	0.0425(–63.6%)	0.0420 (-62.4%)
$\varepsilon = 3 \times 10^{-4}$	**0.0441** (-58.1%)	**0.0375** (-67.9%)	**0.0370** (-67.1%)

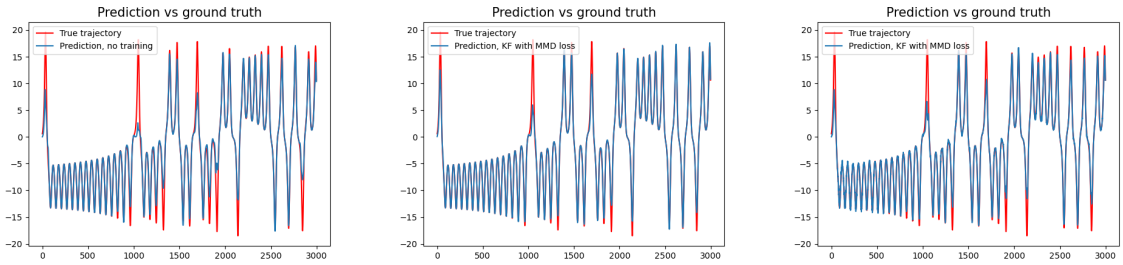


Figure 4: Comparison between the different predictions of the first variables for the baseline, ρ training and ρ_{MMD} training (left, middle, right).

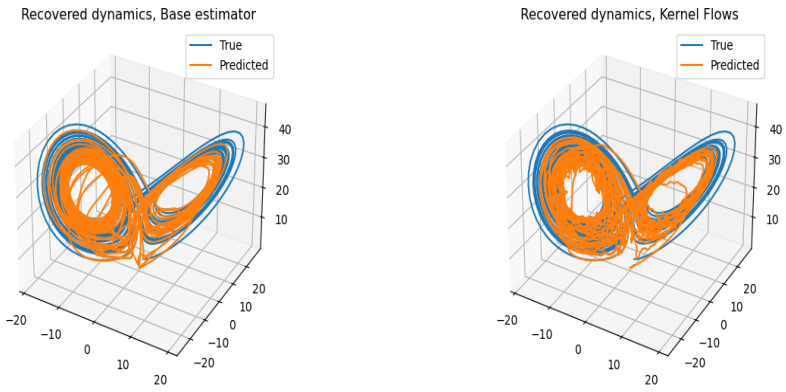


Figure 5: Recovered dynamics: base estimator (left), kernel flows with ρ_{MMD} (right).

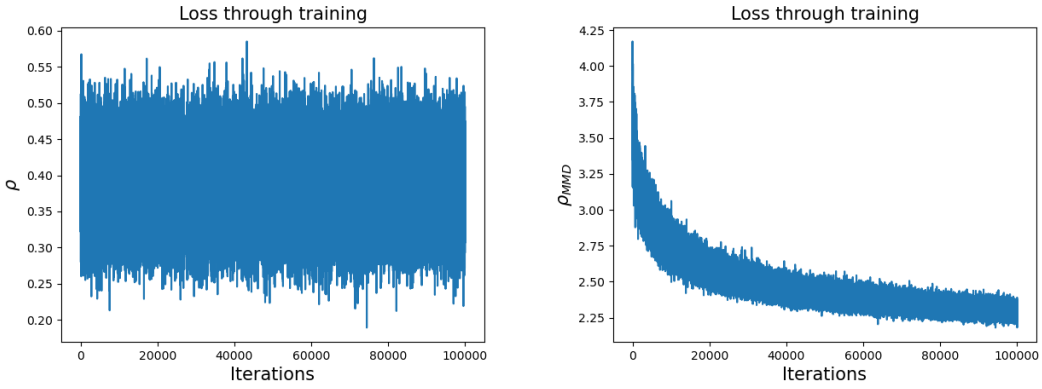


Figure 6: Loss functions through training of the first variable for ρ (left) and ρ_{MMD} (right).

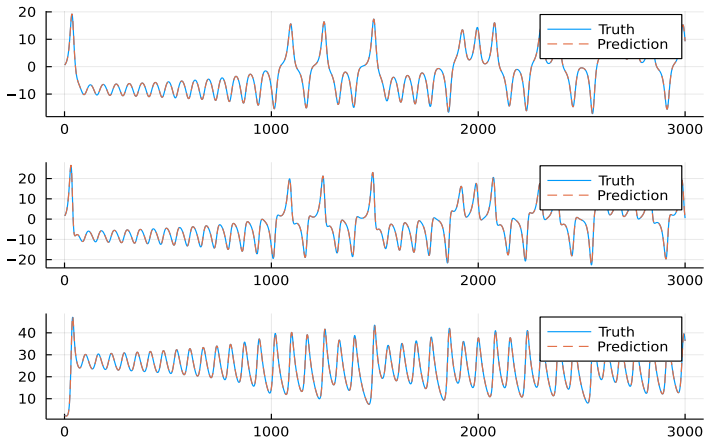


Figure 7: Times series predictions.

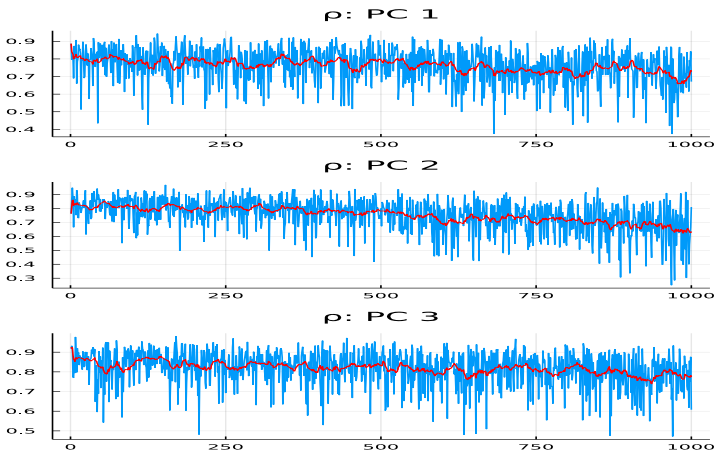


Figure 8: Losses.

Remarks

i. Convergence results that characterize the error estimates of the difference between a dynamical system and its approximation from data using kernel methods can be found in [5, 15].

ii. In the case of very large datasets, it is possible to reduce the number of points during training by considering greedy techniques as in [42, 44].

iii. It is possible to include new measurements when approximating the dynamics from data without repeating the learning process. This can be done by working in Newton basis as in [39] (check also section 4 of [43]). The Newton basis is just another basis for the space spanned by the kernel on the points, i.e., $\text{span}\{k(.,x_1),....,k(.,x_N)\} = \text{span}\{v_1,...,v_N\}$.

The kernel expansion of f writes as $f(x) = \sum_{i=1}^{N} c_i K(x,x_i) = \sum_{i=1}^{N} b_i v_i(x)$ with $< v_i, v_j >_H = \delta_{ij}$ (i.e., the basis is orthonormal in the RKHS inner product).

If we add a new point $x_{N+1},...,x_{N+m}$, we'll have corresponding elements $v_{N+1},...,v_{N+m}$ of the Newton basis, still orthonormal to the previous ones. So we will have a new interpolant $f_{\text{new}}(x) = \sum_{i=1}^{N+m} b_i v_i(x)$ that can be rewritten in terms of the old interpolant as

$$f_{\text{new}}(x) = \sum_{i=1}^{N+m} c_i v_i(x) = f(x) + \sum_{i=N+1}^{N+m} c_i v_i(x),$$

where f can still be written in terms of the basis K, but with different coefficients c'.

If A is the kernel matrix on the first N points, on can compute a Cholesky factorization $A = LL^T$ with L lower triangular. Let $B := L^{-T}$, then $v_j(x) = \sum_{i=1}^{N} (B)_{ij} K(x,x_i)$.

When we add new points, we have an updated kernel matrix A', and the Cholesky factor of A can be easily updated to the one of A'.

iv. In our simulations, the kernels in (6) and (8) are both Gaussian, a possible extension is to consider that they are different kernels and to learn them using parametric KFs.

4 Conclusion

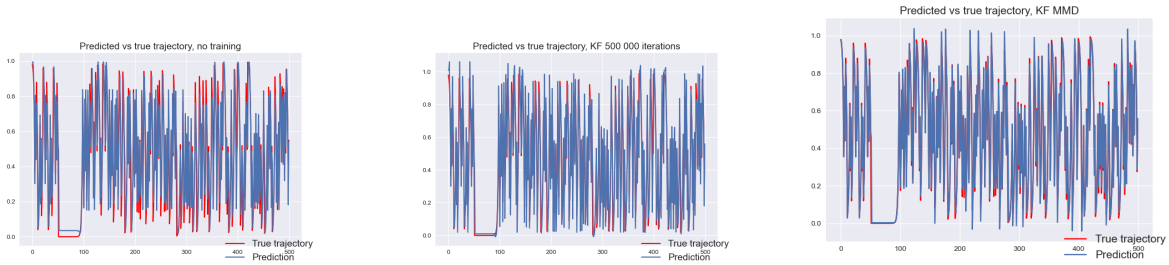
In this paper, numerical experiments show that by using the Gaussian kernel and initially choosing the variance that characterizes it, combining two variants of non-parametric KFs with the kriging of the vector field improves the accuracy of the prediction of chaotic time series. Extensions of this approach to other classes of dynamical systems remain open.

5 Acknowledgment

Parts of this work were done when B. H. was a Marie Curie fellow at Imperial College London. B. H. thanks the European Commission for funding through the Marie Curie fellowship STALDYS-792919 (Statistical Learning for Dynamical Systems). H. O. gratefully acknowledges support by the Air Force Office of Scientific Research under MURI award number FA9550-20-1-0358 (Machine Learning and Physics-Based Modeling and Simulation). A. B, J. S, and H. O. gratefully acknowledge support through the JPL Research and Technology Development program award: UQ-aware Machine Learning for Uncertainty Quantification (October 2020-September 2022).

6 Additional figures

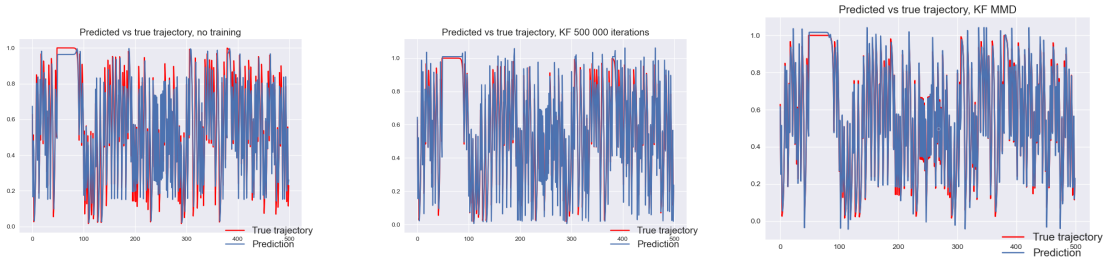
6.1 Bernoulli time series



(a) Time series (red) and the prediction (blue) without learning for $x_0 = 0.99$.

(b) Time series (red) and the prediction (blue) by the learned kernel with ρ for $x_0 = 0$.

(c) Time series (red) and the prediction (blue) by the learned kernel with ρ_{MMD} for $x_0 = 0.99$.



(d) Time series (red) and the prediction (blue) without learning for$x_0 = \pi/10$

(e) Time series (red) and the prediction (blue) by the learned kernel with ρ for $x_0 = \pi/10$.

(f) Time series (red) and the prediction (blue) by the learned kernel with ρ_{MMD} for $x_0 = \pi/10$.

Figure 9: Prediction results for the Bernoulli map for different initial conditions.

6.2 Lorenz time series

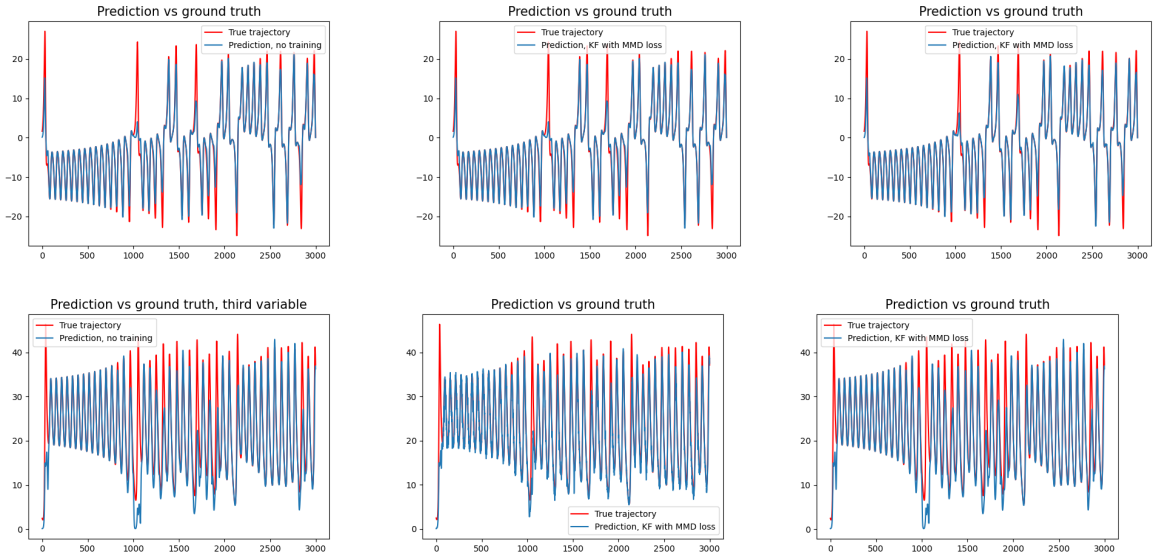


Figure 10: Lorenz system: comparison between the different predictions of the second and third variables (top, bottom) for the baseline, ρ_{MMD} training and ρ training (left, middle, right).

6.3 Convergence of the loss

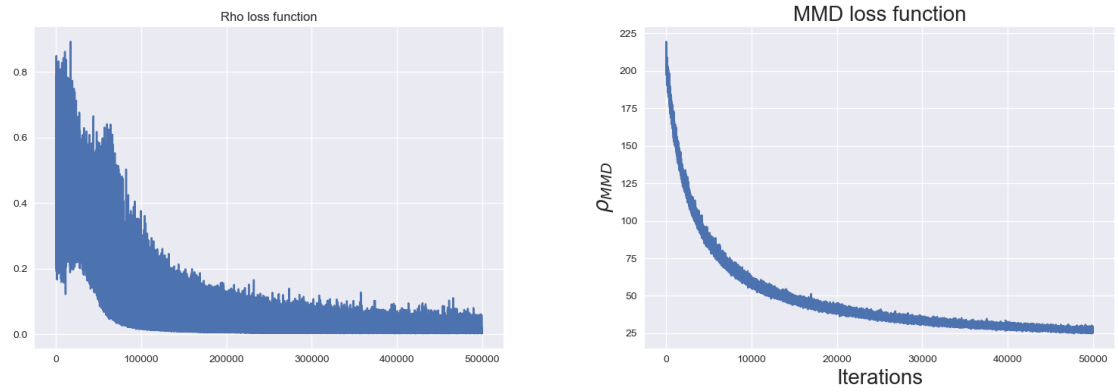


Figure 11: Convergence of the ρ (left) and ρ_{MMD} (right) for higher learning rates on the Bernoulli time series.

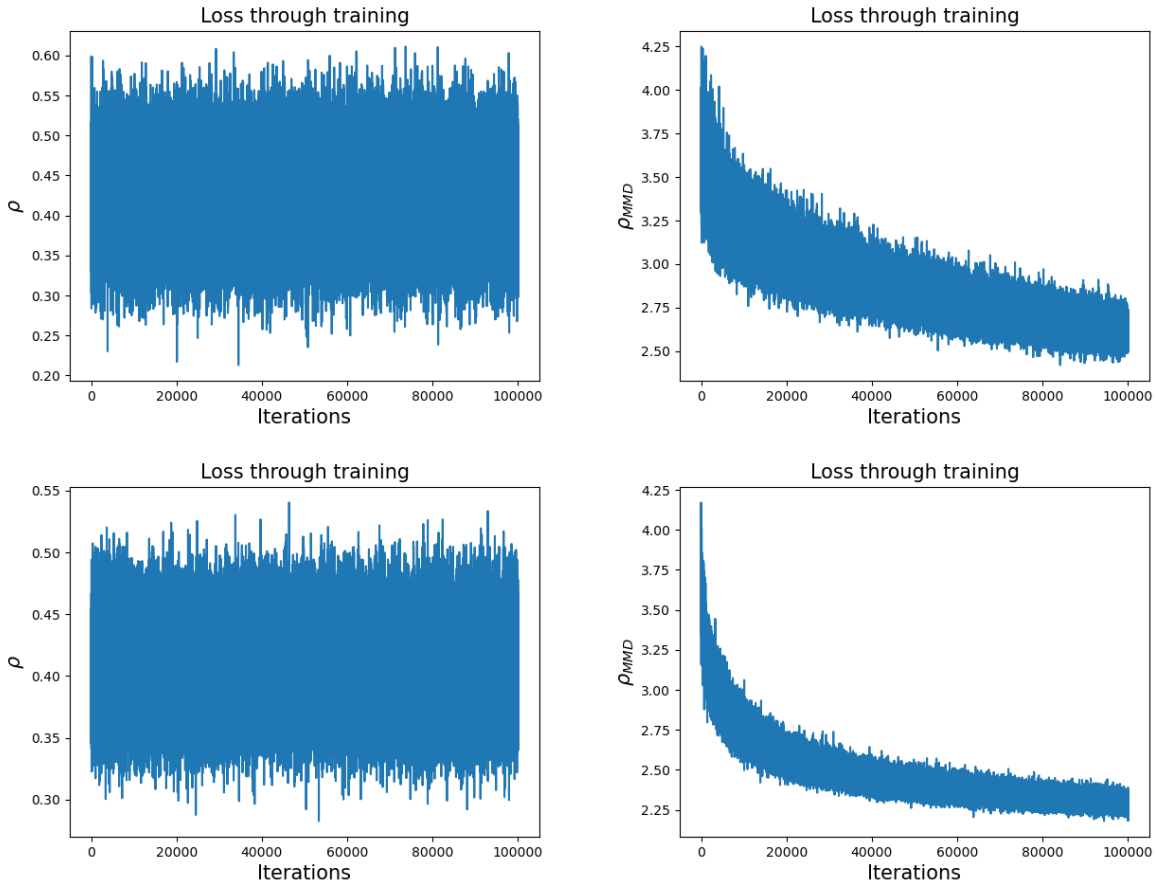


Figure 12: Loss functions through training for the second and third variables of the Lorenz times series (top, bottom) for ρ (left) and ρ_{MMD} (right).

References

[1] H. Abarbanel. *Analysis of Observed Chaotic Data*. Institute for Nonlinear Science. Springer New York, 2012.

[2] Romeo Alexander and Dimitrios Giannakis. Operator-theoretic framework for forecasting nonlinear time series with kernel analog techniques. *Physica D: Nonlinear Phenomena*, 409:132520, 2020.

[3] Hamed Hamze Bajgiran, Pau Batlle Franch, Houman Owhadi, Clint Scovel, Mahdy Shirdel, Michael Stanley, and Peyman Tavallali. Uncertainty quantification of the 4th kind; optimal posterior accuracy-uncertainty tradeoff with the minimum enclosing ball. *arXiv preprint arXiv:2108.10517*, 2021.

[4] Andreas Bittracher, Stefan Klus, Boumediene Hamzi, Péter Koltai, and Christof Schütte. Dimensionality reduction of complex metastable systems via kernel embeddings of transition manifolds. 2019. https://arxiv.org/abs/1904.08622.

[5] J. Bouvrie and B. Hamzi. Kernel methods for the approximation of nonlinear systems. *SIAM J. Control and Optimization*, 2017. https://arxiv.org/abs/1108.2903.

[6] Jake Bouvrie and Boumediene Hamzi. Empirical estimators for stochastically forced nonlinear systems: Observability, controllability and the invariant measure. *Proc. of the 2012 American Control Conference*, pages 294–301, 2012. https://arxiv.org/abs/1204.0563v1.

[7] Jake Bouvrie and Boumediene Hamzi. Kernel methods for the approximation of nonlinear systems. *SIAM J. Control and Optimization*, 2017. https://arxiv.org/abs/1108.2903.

[8] Jake Bouvrie and Boumediene Hamzi. Kernel methods for the approximation of some key quantities of nonlinear systems. *Journal of Computational Dynamics*, 1, 2017. http://arxiv.org/abs/1204.0563.

[9] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

[10] Martin Casdagli. Nonlinear prediction of chaotic time series. *Physica D: Nonlinear Phenomena*, 35(3):335 – 356, 1989.

[11] Ashesh Chattopadhyay, Pedram Hassanzadeh, Krishna V. Palem, and Devika Subramanian. Data-driven prediction of a multi-scale lorenz 96 chaotic system using a hierarchy of deep learning methods: Reservoir computing, ann, and RNN-LSTM. *CoRR*, abs/1906.08829, 2019.

[12] Yifan Chen, Bamdad Hosseini, Houman Owhadi, and Andrew M Stuart. Solving and learning nonlinear pdes with gaussian processes. *arXiv preprint arXiv:2103.12959*, 2021.

[13] Felipe Cucker and Steve Smale. On the mathematical foundations of learning. *Bulletin of the American Mathematical Society*, 39:1–49, 2002.

[14] B.Haasdonk ,B.Hamzi , G.Santin , D.Wittwar. Kernel methods for center manifold approximation and a weak data-based version of the center manifold theorems. *Physica D*, 2021.

[15] P. Giesl, B. Hamzi, M. Rasmussen, and K. Webster. Approximation of Lyapunov functions from noisy data. *Journal of Computational Dynamics*, 2019. https://arxiv.org/abs/1601.01568.

[16] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012.

[17] B. Haasdonk, B. Hamzi, G. Santin, and D. Wittwar. Greedy kernel methods for center manifold approximation. *Proc. of ICOSAHOM 2018, International Conference on Spectral and High Order Methods*, (1), 2018. https://arxiv.org/abs/1810.11329.

[18] Boumediene Hamzi and Fritz Colonius. Kernel methods for the approximation of discrete-time linear autonomous and control systems. *SN Applied Sciences*, 1(7):1–12, 2019.

[19] Boumediene Hamzi, Christian Kuehn, and Sameh Mohamed. A note on kernel methods for multiscale systems with critical transitions. *Mathematical Methods in the Applied Sciences*, 42(3):907–917, 2019.

[20] Boumediene Hamzi, Romit Maulik, and Houman Owhadi. Data-driven geophysical forecasting: Simple, low-cost, and accurate baselines with kernel methods.

[21] Boumediene Hamzi and Houman Owhadi. Learning dynamical systems from data: A simple cross-validation perspective, part i: Parametric kernel flows. *Physica D: Nonlinear Phenomena*, 421:132817, 2021.

[22] Holger Kantz and Thomas Schreiber. *Nonlinear Time Series Analysis*. Cambridge University Press, USA, 1997.

[23] Stefan Klus, Feliks Nuske, and Boumediene Hamzi. Kernel-based approximation of the koopman generator and schrödinger operator. *Entropy*, 22, 2020. https://www.mdpi.com/1099-4300/22/7/722.

[24] Stefan Klus, Feliks Nüske, Sebastian Peitz, Jan-Hendrik Niemann, Cecilia Clementi, and Christof Schütte. Data-driven approximation of the koopman generator: Model reduction, system identification, and control. *Physica D: Nonlinear Phenomena*, 406:132416, 2020.

[25] Andreas Bittracher, Stefan Klus, Boumediene Hamzi, Peter Koltai, and Christof Schutte. Dimensionality reduction of complex metastable systems via kernel embeddings of transition manifold, 2019. https://arxiv.org/abs/1904.08622.

[26] Jonghyeon Lee, Edward De Brouwer, Boumediene Hamzi, and Houman Owhadi. Learning dynamical systems from data: A simple cross-validation perspective, part iii: Irregularly-sampled time series, 2021.

[27] A. Nielsen. *Practical Time Series Analysis: Prediction with Statistics and Machine Learning*. O'Reilly Media, 2019.

[28] Boumediene Hamzi , Romit Maulik, Houman Owhadi. Simple, low-cost and accurate data-driven geophysical forecasting with learned kernels. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 477(2252), 2021.

[29] H. Owhadi and G. R. Yoo. Kernel flows: From learning kernels from data into the abyss. *Journal of Computational Physics*, 389:22–47, 2019.

[30] Houman Owhadi. Computational graph completion. *arXiv preprint arXiv:2110.10323*, 2021.

[31] Houman Owhadi and Clint Scovel. Brittleness of Bayesian inference and new Selberg formulas. *Commun. Math. Sci.*, 14(1):83–145, 2016.

[32] Houman Owhadi and Clint Scovel. Qualitative robustness in bayesian inference. *ESAIM: Probability and Statistics*, 21:251–274, 2017.

[33] Houman Owhadi and Clint Scovel. *Operator-Adapted Wavelets, Fast Solvers, and Numerical Homogenization: From a Game Theoretic Approach to Numerical Approximation and Algorithm Design*. Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, 2019.

[34] Houman Owhadi, Clint Scovel, and Tim Sullivan. On the brittleness of bayesian inference. *siam REVIEW*, 57(4):566–582, 2015.

[35] Houman Owhadi, Clint Scovel, Tim Sullivan, et al. Brittleness of bayesian inference under finite information in a continuous world. *Electronic Journal of Statistics*, 9(1):1–79, 2015.

[36] Houman Owhadi and Lei Zhang. Metric-based upscaling. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 60(5):675–723, 2007.

[37] M. Darcy , P. Tavallali , G. Livieri , B. Hamzi , H. Owhadi. Learning dynamical systems from data: a simple cross-validation perspective, part IV: Robust kernel learning of sdes. *preprint*, 2021.

[38] Jaideep Pathak, Zhixin Lu, Brian R. Hunt, Michelle Girvan, and Edward Ott. Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 27(12):121102, 2017.

[39] Maryam Pazouki and Robert Schaback. Bases for kernel-based spaces. *Journal of Computational and Applied Mathematics*, 236(4):575 – 588, 2011. International Workshop on Multivariate Approximation and Interpolation with Applications (MAIA 2010).

[40] O Perrin and P Monestiez. Modelling of non-stationary spatial structure using parametric radial basis deformations. In *GeoENV II—Geostatistics for Environmental Applications*, pages 175–186. Springer, 1999.

[41] Paul D Sampson and Peter Guttorp. Nonparametric estimation of nonstationary spatial covariance structure. *Journal of the American Statistical Association*, 87(417):108–119, 1992.

[42] G. Santin and B. Haasdonk. Kernel methods for surrogate modelling. *ArXiv e-prints arXiv:1907.10556*, 2019. https://arxiv.org/abs/1907.10556.

[43] Gabriele Santin and Bernard Haasdonk. Kernel methods for surrogate modeling. 2019. https://arxiv.org/abs/1907.105566.

[44] Florian Schäfer, Matthias Katzfuss, and Houman Owhadi. Sparse cholesky factorization by kullback-leibler minimization. 2020. https://arxiv.org/abs/2004.14455.

[45] 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.

[46] Sai Prasanth , Ziad S Haddad , Jouni Susiluoto , Amy J Braverman , Houman Owhadi, Boumediene Hamzi , Svetla M Hristova-Veleva , Joseph Turk. Kernel flows to infer the structure of convective storms from satellite passive microwave observations. *preprint*, 2021.

[47] Andrew Zammit-Mangion, Tin Lok James Ng, Quan Vu, and Maurizio Filippone. Deep compositional spatial models. *arXiv preprint arXiv:1906.02840*, 2019.

View publication stats