Scaling up Vector Autoregressive Models With Operator-Valued Random Fourier Features.

Romain Brault[1,3], Néhémy Lim[2], and Florence d'Alché-Buc[3]

[1] IBISC, Université d'Évry-Val-d'Essonne, Évry, 91000, France
romain.brault@telecom-paristech.fr
[2] Department of Statistics, University of Washington, Seattle, WA 98195, USA
nehemyl@uw.edu
[3] LTCI CNRS, Télécom ParisTech, Université Paris-Saclay, France
florence.dalche@telecom-paristech.fr

Abstract. We consider a nonparametric approach to Vector Autoregressive modeling by working in vector-valued Reproducing Kernel Hilbert Spaces (vv-RKHS). The main idea is to build vector-valued models (OKVAR) using Operator-Valued Kernels (OVK). As in the scalar case, regression with OVK boils down to learning as many weight parameters as data, except that here, weights are vectors. To avoid the inherent complexity in time and in memory to deal with kernels, we introduce Operator-Valued Random Fourier Features (ORFF) that extend Random Fourier Features devoted to scalar-valued kernels approximation. Applying the approach to decomposable kernels, we show that ORFFVAR is able to compete with OKVAR in terms of accuracy on stationary nonlinear time series while keeping low execution time, comparable to VAR. Results on simulated datasets as well as real datasets are presented.

1 Introduction

Time series are ubiquitous in various fields such as climate, biomedical signal processing, videos understanding to name but a few. When linear models are not appropriate, a generic nonparametric approach to modeling is relevant. In this work we build on a recent work about Vector Autoregressive models using Operator-Valued Kernels [1,2]. Vector autoregression is addressed in a vector-valued Reproducing Kernel Hilbert Space with the important property to allow for couplings between outputs. Given a d-dimensional time series of N data points $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, autoregressive models based on operator-valued kernels have the form $\hat{\mathbf{x}}_{t+1} = h(\mathbf{x}_t) = \sum_{\ell=1}^{N-1} K(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell$ where coefficients $\mathbf{c}_\ell \in \mathbb{R}^d, \ell = 1, \dots, N-1$ are the model parameters. A naive approach for training such a model requires a memory complexity $O(N^2d^2)$, which makes the method prohibitive for large-scale problems. To scale up standard algorithms, we define an approximated operator-valued feature map $\tilde{\Phi} : \mathbb{R}^d \to \mathbb{R}^D$ that allows to approximate the aforementioned model h in the RKHS by the following function: $\tilde{h}(\mathbf{x}_t) = \tilde{\Phi}(\mathbf{x}_t)^*\theta \approx h(\mathbf{x}_t)$. The features maps are matrices of size $D \times d$ where D controls the quality of the approximation, d is the dimension of the

inputs and θ is here the parameter vector to learn. This formulation allows to reduce the memory complexity to $O((N-1).D+(N-1).d)$ which is now linear w.r.t. the number of data points. The principle used for building the feature map extends the idea of Random Fourier Features to the operator-valued case [3,4].

2 Operator-Valued Kernels for Vector Autoregression

Assume that we observe a dynamical system composed of d state variables at N evenly-spaced time points. The resulting discrete multivariate time series is denoted by $\mathbf{x}_{1:N} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ where $\mathbf{x}_\ell \in \mathbb{R}^d$ denotes the state of the system at time t_ℓ, $\ell = 1, \ldots, N$. It is generally assumed that the evolution of the state of the system is governed by a function h, such that $\mathbf{x}_t = h(\mathbf{x}_{t-p}, \ldots, \mathbf{x}_{t-1}) + \mathbf{u}_t$ where t is a discrete measure of time and $\mathbf{u}_t$ is a zero-mean noise vector. Then h is usually referred to as a vector autoregressive model of order p. In the remainder of the paper, we consider first-order vector autoregressive models, that is $p = 1$. In a supervised learning setting, the vector autoregression problem consists in learning a model $\hat{h} : \mathbb{R}^d \to \mathbb{R}^d$ from a given training set $\mathcal{S}_N = \{(\mathbf{x}_1, \mathbf{x}_2), \ldots, (\mathbf{x}_{N-1}, \mathbf{x}_N)\} \subseteq \mathbb{R}^d \times \mathbb{R}^d$. In the literature, a standard approach to vector autoregressive modeling is to fit a VAR model. The VAR(1) model reads : $h(\mathbf{x}_t) = A\mathbf{x}_t$ where A is an $d \times d$ matrix whose structure encodes the temporal relationships among the d state variables.

However, due to their intrinsically linear nature, VAR models fail to capture the nonlinearities underlying realistic dynamical systems. This paper builds upon the recent work of [2] where the authors introduced a family of nonparametric nonlinear autoregressive models called OKVAR (*Operator-valued Kernel-based Vector AutoRegressive*). OKVAR models rely on the theory of operator-valued kernels [5], which provides a versatile framework for learning vector-valued functions [6,7,8]. Those models can be regarded as natural extensions of VAR models to the nonlinear case.

Next, we provide key elements of the theory of vector-valued Reproducing Kernel Hilbert spaces (vv-RKHS) of functions from $\mathbb{R}^d$ to $\mathbb{R}^d$.

Definition 1 (Matrix-valued kernels). *A function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is said to be a positive semidefinite $\mathbb{R}^{d\times d}$-valued kernel if :*

i) $\forall \mathbf{x}, \mathbf{z} \in \mathbb{R}^d,\ K(\mathbf{x}, \mathbf{z}) = K(\mathbf{z}, \mathbf{x})^*$,
ii) $\forall m \in \mathbb{N}, \forall \{(\mathbf{x}_i, \mathbf{y}_i) \in \mathbb{R}^d \times \mathbb{R}^d,\ i = 1, \ldots, m\},\ \sum_{i,j=1}^m \mathbf{y}_i^* K(\mathbf{x}_i, \mathbf{x}_j)\mathbf{y}_j \geq 0.$

Furthermore, for a given $\mathbb{R}^{d\times d}$-valued kernel K, we associate K with a unique vv-RKHS $(\mathcal{H}_K, \langle \cdot, \cdot \rangle_{\mathcal{H}_K})$ of functions from $\mathbb{R}^d$ to $\mathbb{R}^d$. The precise construction of $\mathcal{H}_K$ can be found in [6]. In this paper, we assume that all functions $h \in \mathcal{H}_K$ are continuous. Then K is called an $\mathbb{R}^d$-Mercer kernel. Similarly to the case of scalar-valued kernels, working within the framework of vv-RKHS allows to take advantage of representer theorems for a class of regularized loss functions such as ridge regression. More precisely, we consider h, a nonparametric vector autoregressive model of the following form assuming we have observed N data

points. Given $\mathbf{x}_t$ the state vector at time t, we have $\hat{\mathbf{x}}_{t+1} = \sum_{\ell=1}^{N-1} K(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell$, where $\mathbf{x}_{1:N} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\} \subseteq \mathbb{R}^d$ is the observed time series, $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ is a matrix-valued kernel and $\mathbf{c}_1, \ldots, \mathbf{c}_{N-1} \in \mathbb{R}^d$ are the model parameters. We call OKVAR any model of the above form. In [2], the authors developed a family of OKVAR models based on appropriate choices of kernels to address the problem of network inference where both the parameters $\mathbf{c}_\ell, \ell = 1, \ldots, N-1$ and the OVK itself are learned using a proximal block coordinate descent algorithm under sparsity constraints. In the following, we will not consider the kernel learning problem and will use a simple ridge loss. We will also illustrate our approach to a well known class of OVK, called *decomposable* or *separable* kernels [6,9] that were originally introduced to solve multi-task learning problems[10]. Other kernels may also be considered as developed in [11].

Proposition 1 (Decomposable kernels). *Let $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ be a scalar-valued kernel and $B \in \mathbb{S}_+^d$ a positive semidefinite matrix of size $d \times d$. Then function $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ defined for all $(\mathbf{x}, \mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d$ as $K(\mathbf{x}, \mathbf{z}) = k(\mathbf{x}, \mathbf{z})B$ is a matrix-valued kernel.*

A common choice for the scalar-valued kernel is the Gaussian kernel: $k_{\text{Gauss}}(\mathbf{x}, \mathbf{z}) = \exp(-\|\mathbf{x} - \mathbf{z}\|_2^2/(2\sigma^2))$ for any $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$ and $\sigma > 0$. The corresponding decomposable kernel is referred to as K_{dec} and is as follows: $K_{\text{dec}}(\mathbf{x}, \mathbf{z}) = k_{\text{Gauss}}(\mathbf{x}, \mathbf{z})B$ with $B \in \mathbb{S}_+^d$. While the model parameters $\mathbf{c}_\ell$'s are estimated under sparsity constraints in [2], here we consider the classic kernel ridge regression setting where the loss function to minimize is : $\mathcal{J}(h) = \frac{1}{N-1}\sum_{\ell=2}^{N} \|h(\mathbf{x}_{\ell-1}) - \mathbf{x}_\ell\|_2^2 + \lambda\|h\|_{\mathcal{H}_K}^2$ with $\lambda \geq 0$ and $\|h\|_{\mathcal{H}_K}^2 = \sum_{t,\ell=1}^{N-1} \mathbf{c}_t^* K(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell$. The optimization problem is solved using a L-BFGS-B which is well suited for optimization problems with a large number of parameters, and is wildly used as a training algorithm on small/medium-scale problems. However, like standard kernel methods, OKVAR suffers from unfavorable computational complexity both in time and memory since it needs to store the full Gram matrix, preventing its ability to scale to large data sets and making it really slow on medium scale problem. We argue that this obstacle can be effectively overcome: in the following we develop a method to scale up OKVAR to successfully tackle medium/large scale autoregression problems.

3 Operator-Valued Random Fourier Features

We now introduce our methodology to approximate OVKs. Given a translation-invariant kernel $K(\mathbf{x}, \mathbf{z}) = K_0(\mathbf{x} - \mathbf{z})$, we approximate K by finding an explicit feature map such that $\tilde{\Phi}(\mathbf{x})^*\tilde{\Phi}(\mathbf{z}) \approx K_0(\mathbf{x} - \mathbf{z})$. The idea is to use a generalization of Bochner's theorem for the OVK family that states that any translation-invariant OVK can be written as the Fourier transform of a positive operator-valued measure. More precisely, we build on the following proposition first proven in [7]. More details can be found in [11].

Proposition 2 ([7]). *Let K be a shift-invariant $\mathbb{R}^d$-Mercer kernel. Suppose that $\forall i, j = 1, \ldots, d,\ K_0(\cdot)_{ij} \in L^1(\mathbb{R}^d, dx)$ where dx denotes the Lebesgue measure on*

$\mathbb{R}^d$. *Define the matrix* $C(\omega)$ *such that for all* $\omega \in \mathbb{R}^d$,

$$C(\omega)_{ij} = \int_{\mathbb{R}^d} \exp\left(i\langle\delta, \omega\rangle\right) K_0(\delta)_{ij} d\delta = \mathcal{F}^{-1}\left[K_0(\cdot)_{ij}\right](\omega). \tag{1}$$

Then (i) $C(\omega)$ *is a non-negative operator for all* $\omega \in \mathbb{R}^d$, *(ii)* $C(.)_{ij} \in L^1(\mathbb{R}^d, d\omega)$, *and (iii) for all* $\delta \in \mathbb{R}^d$, $K_0(\delta)_{ij} = \int_{\mathbb{R}^d} \exp\left(-i\langle\delta, \omega\rangle\right) C(\omega)_{ij} d\omega$.

In the following, suppose that $K_0 = k_0(\cdot)A$ is a decomposable kernel. Decomposable kernels belong to the family of translation-invariant OVK. From proposition 2 we see that $C(\omega)_{ij} = \mathcal{F}^{-1}\left[k_0(\cdot)\right](\omega)A_{ij}$. We decompose A as $A = BB^*$, note that A does not depend on ω, and we denote $\bigoplus_{j=1}^{D} \mathbf{z}_j$ the Dm-long column vector obtained by stacking vectors $\mathbf{z}_j \in \mathbb{R}^m$. Then we define an approximate feature map for K_0, called Operator-Valued Random Fourier Feature (ORFF) map [11] as follows: for all $\mathbf{x} \in \mathbb{R}^d$,

$$\tilde{\Phi}^{dec}(\mathbf{x}) = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle\mathbf{x}, \omega_j\rangle B^* \\ \sin\langle\mathbf{x}, \omega_j\rangle B^* \end{pmatrix}, \ \omega_j \sim \mathcal{F}^{-1}\left[k_0\right],$$

which can also be expressed as a Kronecker product of a scalar feature map with an operator:

$$\tilde{\Phi}^{dec}(\mathbf{x}) = \tilde{\phi}(\mathbf{x}) \otimes B^*, \text{ where } \tilde{\phi}(\mathbf{x}) = \frac{1}{\sqrt{D}} \bigoplus_{j=1}^{D} \begin{pmatrix} \cos\langle\mathbf{x}, \omega_j\rangle \\ \sin\langle\mathbf{x}, \omega_j\rangle \end{pmatrix}, \ \omega_j \sim \mathcal{F}^{-1}\left[k_0\right]$$

is a scalar-valued feature map. In particular, if k_0 is a Gaussian kernel with bandwidth σ^2, then $\mathcal{F}^{-1}\left[k_0\right] = \mathcal{N}(0, 1/\sigma^2)$ as proven in [3]. More examples on different OVK can be found in [11] as well as a proof of the uniform convergence of the kernel approximation defined by $\tilde{K}(\mathbf{x}, \mathbf{z}) = \tilde{\Phi}(\mathbf{x})^* \tilde{\Phi}(\mathbf{z})$ towards the true kernel. In the case of vector autoregression, we consider a model $\tilde{h}$ of the form: $\hat{\mathbf{x}}_{t+1} = \tilde{\Phi}(\mathbf{x}_t)^* \theta$. That model is referred to as ORFFVAR in the remainder of the paper. Now, given the operator-valued feature map, we get a linear model, and we want to minimize the loss function

$$\mathcal{J}(\theta) = \frac{1}{N-1} \sum_{\ell=2}^{N} \left\| (\tilde{\phi}(\mathbf{x}_{\ell-1})^* \otimes B)\theta - \mathbf{x}_\ell \right\|_2^2 + \lambda \|\theta\|_2^2 \tag{2}$$

with $\lambda \geq 0$. [11] proposed to formulate the learning problem as a Stein equation when dealing with decomposable kernels, and then used an appropriate solver [12]. We opted here for a more general algorithm, which is a variant of the doubly stochastic gradient descent [13]. In a few words, this algorithm is a stochastic gradient descent that takes advantage of the feature representation of the kernel allowing the number of features to grow along with the number of points. [13] show that the number of iterations needed for achieving a desired accuracy ε using a stochastic approximation is $\Omega(1/\varepsilon)$, making it competitive compared to other stochastic methods for kernels such as NORMA [14] and its OVK adaptation ONORMA [15]. We propose here in Algorithm 1, an extension of the

doubly stochastic gradient descent of [13] to OVK. Additionally we consider a batch approach w.r.t. the data and the features, and make it possible to 'cap' the maximum number of features. The inputs of the algorithm are: $\mathcal{X}$ the input data, $\mathcal{Y}$ the targets, $K_0(\cdot)$ the kernel used for learning, γ_t the learning rate (see [13] for a discussion on the selection of a proper learning rate), T the number of iterations, n the size of data batch, b the size of the feature batch, and D the maximum number of features. Note that if K_0 is a scalar kernel, $D = T$, $b = 1$ and $n = 1$, we retrieve the algorithm formulated in [13].

Data: $\mathcal{X}$, $\mathcal{Y}$, K_0, γ_t, λ, T, n, D, b
Result: Find θ
Let $D_b = D/b$ and find B and $\mu = \mathcal{F}^{-1}[k_0]$ from K_0;
for $i = 1$ **to** D_b **do**
  $\theta^1_{i,\cdot} = 0$;
end
for $t = 1$ **to** T **do**
  $\mathcal{A}_t = \mathcal{X}_t \times \mathcal{Y}_t$, a random subsample of n data from $\mathcal{X} \times \mathcal{Y}$;
  $h(\mathcal{X}_t) =$ **predict** $(\mathcal{X}_t, \theta^t, K_0)$; `// Make a prediction.`
  $\Omega_i \sim \mu(\omega)$ with seed i, where $i = ((t-1) \ \mathbf{mod} \ D_b) + 1$; `// Sample` b `features from` $\mu(\omega)$`.`
  for $\omega \in \Omega_i$ `// Update the parameters from the gradient.`
  do

$$\theta^{t+1}_{i,\omega} = \theta^t_{i,\omega} - \gamma_t \left(\frac{1}{|\mathcal{A}_t|} \sum_{(x,\mathbf{y}) \in \mathcal{A}_t} \frac{B \exp(i\langle \omega, x \rangle)(h(x) - \mathbf{y})}{\sqrt{D}} + \lambda \theta^t_{i,\omega} \right);$$

  end
end
return θ^{t+1}

Algorithm 1: Block-coordinate mini-batch doubly SGD.

In addition, the convergence of the algorithm can be speeded-up by preconditioning by the Hessian of the system.

4 Numerical Performance

Simulated data. To assess the performance of our models, we start our investigation by generating discrete d-dimensional time series $(\mathbf{x}_t)_{t \geq 1}$ as follows

$$\begin{cases} \mathbf{x}_1 \sim \mathcal{N}(0, \Sigma_{\mathbf{x}}) \\ \mathbf{x}_{t+1} = h(\mathbf{x}_t) + \mathbf{u}_{t+1}, \forall t > 0. \end{cases} \tag{3}$$

where the residuals are homoscedastic and distributed according to $\mathbf{u}_t \sim \mathcal{N}(0, \Sigma_{\mathbf{u}})$. We study two different kinds of noise: an isotropic noise with covariance $\Sigma_{\mathbf{u}} = \sigma^2_{\mathbf{u}} I_d$ and an anisotropic noise with Toeplitz structure $\Sigma_{\mathbf{u},ij} = \nu^{|i-j|}$, where ν lives in $(0, 1)$. We generated $N = 1000$ data points and used a sequential cross-validation (SCV) with time windows $N_t = N/2$ to measure the Mean Squared Error (SCV-MSE) of the different models. Next, we compare the performances

of VAR(1), OKVAR and ORFFVAR through three scenarios. Across the simulations, the topological structures of the underlying dynamical systems are encoded by a matrix A of size 5×5. All entries of A are set to zero except for the diagonal where all coefficients are equal to 0.9 for Settings 1 and 3 and 0.5 for Setting 2. Then five off-diagonal coefficients are drawn randomly from $\mathcal{N}(0, 0.3)$ for Settings 1 and 3 and $\mathcal{N}(0, 0.5)$ for Setting 2. We check that all the eigenvalues of A are less than one to ensure the stability of the system. More specifically, we picked the following values of parameters for each scenario:

- **Setting 1: Linear model.**: $h(\mathbf{x}_t) = A\mathbf{x}_t$, $\nu = 0.9$ and $\sigma_{\mathbf{u}} = 0.9$,
- **Setting 2: Exponential model.**: $h(\mathbf{x}_t) = A \exp(\mathbf{x}_t)$ where exp is the element-wise exponential function, $\nu = 0.09$ and $\sigma_{\mathbf{u}} = 0.09$,
- **Setting 3: Sine model.**: $h(\mathbf{x}_t) = A \sin(\mathbf{x}_t)$ where sin is the element-wise sine function, $\nu = 0.9$ and $\sigma_{\mathbf{u}} = 0.009$.

ORFFVAR is instantiated with $D = 25$ random features in presence of a white noise while we set $D = 50$ in case of a Toeplitz noise. We summarize the computational efficiency and the statistical accuracy of the models in Table 1.Throughout all the experiments, we set B as the identity matrix of size $d \times d$. This reflects the absence of a prior on the structure of the data. A further study on the influence of the choice of B can be found in [8].

In Setting 1, we observe that OKVAR does not provide any advantage over VAR(1) as expected since the data were generated according to a linear VAR(1) model. Note that OKVAR takes orders of magnitude more time to achieve the same performance as VAR(1) while ORFFVAR performs equally well with a competitive timing. In nonlinear scenarios (Settings 2 and 3), OKVAR and ORFFVAR consistently outperform VAR(1). Noticeably, ORFFVAR reaches the accuracy of OKVAR with the computation time of VAR(1).

Setting		1			2			3		
model	noise	SVC-MSE	variance	time	SVC-MSE	variance	time	SVC-MSE	variance	time
VAR(1)	White	**0.914979**	**0.572485**	0.002467(s)	0.001275	0.000994	0.002346(s)	0.009534	0.006003	**0.001697(s)**
	Toeplitz	**1.091096**	**1.267880**	0.004822(s)	0.017014	0.013498	**0.002050(s)**	**0.116901**	0.127396	0.001702(s)
ORFFVAR	White	0.919663	0.572936	**0.000994(s)**	**0.001003**	**0.000647**	**0.001284(s)**	0.009536	0.005998	0.002377(s)
	Toeplitz	1.097183	1.268978	**0.001022(s)**	**0.012635**	**0.008837**	0.012144(s)	0.116964	**0.127395**	**0.000934(s)**
OKVAR	White	0.958790	0.591934	0.104706(s)	0.001100	0.000731	0.027099(s)	**0.009227**	**0.005717**	0.014458(s)
	Toeplitz	1.410969	1.312243	0.289046(s)	0.013854	0.010977	1.856988(s)	0.160133	0.136570	0.019170(s)

Table 1. Sequential cross-validation MSE and computation times for VAR(1), ORFFVAR and OKVAR on synthetic data (Settings 1, 2 and 3).

Influence of the number of random features. Next, we investigate the impact of D, the number of random features for ORFFVAR. To this end, we generated $N = 10000$ data points following eq. (3), with exponential nonlinearities and white noise as in Setting 2. We performed a sequential cross-validation on a window of $N/2$ data. As expected the error decreases with the number of random features D (Table 2). For the same computation time ($D = 25$) as VAR(1), ORFFVAR achieves an SCV-MSE that is twice as small.

model	$D = 1$	$D = 5$	$D = 10$	$D = 25$	$D = 50$	$D = 100$	VAR(1)
SVC-MSE	0.005342	0.001111	0.000991	0.000962	0.000949	0.000944	0.001660
variance	0.008639	0.000793	0.000660	0.000618	0.000608	0.000605	0.001363
time	0.001191(s)	0.002384(s)	0.003614(s)	0.018469(s)	0.038229(s)	0.069294(s)	0.019634(s)

Table 2. SVC-MSE with respect to D the number of random features for ORFFVAR.

Real data. We now investigate three real datasets. The performances of the models on those datasets are recorded in Table 3. Throughout the experiments, the hyperparameters are set as follows: the bandwidth of the Gaussian kernel σ is chosen as the median of the Euclidean pairwise distances and the regularization parameter λ was tuned on a grid. The number of random features D and the parameters in Algorithm 1 were picked so as to reach the level of accuracy of OKVAR/VAR.**Macrodata:** this dataset is part of the python library Statmodels[4]. It contains 204 US macroeconomic data points collected on the period 1959–2009. Each data point represents 12 economic features. No preprocessing is applied before learning. We measure SVC-MSE using a window of 25 years (50 points). We instantiated Algorithm 1 as follows: $\gamma_t = 1$, $\lambda = 10^{-3}$, $D = 100$, $T = 2$ and $b = 50$ for ORFF and $\lambda = 0.00025$ and $\sigma = 11.18$ for OKVAR. **Gesture phase:** this dataset[5] is constructed using features extracted from seven videos with people gesticulating. We present the results for videos 1 and 4, consisting in 1069 data points and 31 features. Data are normalized prior to learning. We measure SVC-MSE using a time window of 200 points. We implemented ORFFVAR with $\gamma_t = 1$, $\lambda = 10^{-3}$, $D = 100$, $T = 2$ and $b = 50$. **Climate:** this dataset [16] contains monthly meteorological measurements of 18 variables (temperature, CO2 concentration,...) collected at 135 different locations throughout the USA and recorded over 13 years, thus resulting in 135 time series of dimension 18 and length 156. Data are standardized at each station. A unique model is learned for all stations. SVC-MSE is measured on a window of 1872 points, corresponding to the data of all the 135 stations over one year. Specifically, we set the parameters of ORFFVAR as follows: $\gamma_t = 1$, $\lambda = 10^{-6}$, $D = 100$, $T = 1$ and $b = 100$.

			ORFFVAR			VAR(1)			OKVAR		
Dataset	N	d	SVC-MSE	variance	time	SVC-MSE	variance	time	SVC-MSE	variance	time
Macrodata	#203	#12	**445.9**	784.5	0.014(s)	449.1	1021	0.0005(s)	499.8	793.0	0.641(s)
Gesture phase 1	#1743	#31	**0.741**	2.999	0.009(s)	0.980	3.370	0.0014(s)	NA	NA	NA
Gesture phase 4	#1069	#31	**0.473**	2.406	0.061(s)	0.768	6.49	0.0075(s)	NA	NA	NA
Climate	#19375	#18	**0.237**	0.2128	0.396(s)	0.266	0.218	0.0124(s)	NA	NA	NA

Table 3. SCV-MSE and computation times for ORFFVAR, VAR(1) and OKVAR on real datasets.

5 Discussion

Operator-Valued Random Fourier Feature provides a way to approximate OVK and in the context of time series, allows for nonlinear Vector Autoregressive

[4] `https://github.com/statsmodels/statsmodels`

[5] `https://archive.ics.uci.edu/ml/datasets/Gesture+Phase+Segmentation`

models that can be efficiently learned both in terms of computing time and memory. We illustrate the approach with a simple family of Operator-valued kernels, the so-called decomposable kernels but other kernels may be used. While we focused on first-order autoregressive models, we will consider extensions of our models for higher orders. In this work, the kernel hyperparameter B is given prior to learning, however it would be interesting to learn B as in OKVAR. Thus, a promising perspective is to use these models in tasks such as network inference and search for causality graphs among the state variables for large-scale time series [1,2].

References

1. Lim, N., Senbabaoglu, Y., Michailidis, G., d'Alché-Buc, F.: Okvar-boost: a novel boosting algorithm to infer nonlinear dynamics and interactions in gene regulatory networks. Bioinformatics **29**(11) (2013) 1416–1423
2. Lim, N., d'Alché-Buc, F., Auliac, C., Michailidis, G.: Operator-valued kernel-based vector autoregressive models for network inference. Machine Learning **99**(3) (2015) 489–513
3. Rahimi, A., Recht, B.: Random features for large-scale kernel machines. In: NIPS 2007. (2007) 1177–1184
4. Sutherland, D.J., Schneider, J.G.: On the error of random fourier features. In: Proc. of UAI 2015, July 12-16, 2015, Amsterdam, The Netherlands. (2015) 862–871
5. Senkene, E., Tempel'man, A.: Hilbert spaces of operator-valued functions. Lithuanian Mathematical Journal **13**(4) (1973) 665–670
6. Micchelli, C., Pontil, M.: On learning vector-valued functions. Neural Computation **17** (2005) 177–204
7. Carmeli, C., De Vito, E., Toigo, A., Umanità, V.: Vector valued reproducing kernel hilbert spaces and universality. Analysis and Applications **8** (2010) 19–61
8. Álvarez, M., Rosasco, L., Lawrence, N.: Kernels for vector-valued functions: a review. Foundations and Trends in Machine Learning **4**(3) (2012) 195–266
9. Caponnetto, A., Micchelli, C., Pontil, M., Ying, Y.: Universal multi-task kernels. Journal of Machine Learning Research **9** (2008) 1615–1646
10. Evgeniou, T., Micchelli, C., Pontil, M.: Learning multiple tasks with kernel methods. JMLR **6** (2005) 615–637
11. Brault, R., d'Alché-Buc, F., Heinonen, M.: Random fourier features for operator-valued kernels. CoRR **abs/1605.02536** (2016)
12. Sonneveld, P., van Gijzen, M.B.: Idr (s): A family of simple and fast algorithms for solving large nonsymmetric systems of linear equations. SIAM Journal on Scientific Computing **31**(2) (2008) 1035–1062
13. Dai, B., Xie, B., He, N., Liang, Y., Raj, A., Balcan, M.F.F., Song, L.: Scalable kernel methods via doubly stochastic gradients. In: NIPS. (2014) 3041–3049
14. Kivinen, J., Smola, A.J., Williamson, R.C.: Online learning with kernels. IEEE transactions on signal processing **52**(8) (2004) 2165–2176
15. Audiffren, J., Kadri, H.: Online learning with multiple operator-valued kernels. arXiv preprint arXiv:1311.0222 (2013)
16. Liu, Y., Niculescu-Mizil, A., Lozano, A.C., Lu, Y.: Learning temporal causal graphs for relational time-series analysis. In: (ICML-10). (2010) 687–694