Operator-valued Kernel-based Vector Autoregressive Models for Network Inference

Néhémy Lim, Florence d'Alché-Buc, Cédric Auliac, George Michailidis

Operator-valued Kernel-based Vector Autoregressive Models for Network Inference

Néhémy Lim *
IBISC EA 4526, Université d'Évry-Val d'Essonne,
23 Bd de France, 91000,Évry, France,
and CEA, LIST, 91191 Gif-sur-Yvette CEDEX, France,

Florence d'Alché-Buc*
INRIA-Saclay, LRI umr CNRS 8623, Université Paris Sud, France
and IBISC EA 4526, Université d'Évry-Val d'Essonne

Cédric Auliac
CEA, LIST, 91191 Gif-sur-Yvette CEDEX, France,

George Michailidis
Department of Statistics, University of Michigan,
Ann Arbor, MI 48109-1107

October 11, 2013

Abstract

Reverse-modeling of dynamical systems from time-course data still remains a challenging and canonical problem in knowledge discovery. For this learning task, a number of approaches primarily based on sparse linear models or Granger causality have been proposed in the literature. However when the dynamics are nonlinear, there does not exist a systematic answer that takes into account the nature of the underlying system. We introduce a novel family of vector autoregressive models based on a new operator-valued kernel to identify the dynamical system and retrieve the target network. As in the linear case, a key issue is to control the model's sparsity. This control is performed through the joint learning of the structure of the kernel and the basis vectors. To solve this learning task, we propose an alternating optimization algorithm based on proximal gradient procedures that learn both the structure of the kernel and the basis vectors. Results on the DREAM3 competition gene regulatory benchmark networks of size 10 and 100 show the new model outperforms existing methods. Another application of the model on climate data identifies

*The two first authors have equally contributed to the paper

interesting and interpretable interactions between natural and human activity factors thus confirming the ability of the learning scheme to retrieve dependencies between state-variables.

1 Introduction

In many scientific problems, *high dimensional data* with *network structure* play a key role in knowledge discovery (Kolaczyk, 2009). For example, recent advances in high throughput technologies have facilitated the simultaneous study of components of complex biological systems. Hence, molecular biologists are able to measure the expression levels of the entire genome and a good portion of the proteome and metabolome under different conditions and thus gain insight on how organisms respond to their environment. For this reason, reconstruction of gene regulatory networks from expression data has become a canonical problem in computational system biology (Lawrence et al, 2010). Similar data structures emerge in other scientific domains. For instance, political scientists have focused on the analysis of roll call data of legislative bodies, since they allow them to study party cohesion and coalition formation through the underlying network reconstruction (Poole and Rosenthal, 1997; Morton and Williams, 2010), while economists have focused on understanding companies creditworthiness or contagion (Gilchrist et al, 2009). Understanding climate changes implies to be able to predict the behavior of climate variables and their dependence relationship (Parry et al, 2007; Liu et al, 2010). Two classes of network inference problems have emerged simultaneously from all these fields: the inference of association networks that represent coupling between variables of interest (Meinshausen and Bühlmann, 2006; Kramer et al, 2009) and the inference of "causal" networks that describe how variables influence other ones (Murphy, 1998; Perrin et al, 2003; Auliac et al, 2008; Zou and Feng, 2009; Shojaie and Michailidis, 2010; Maathuis et al, 2010; Bolstad et al, 2011; Dondelinger et al, 2012; Chatterjee et al, 2012).

Over the last decade, a number of statistical techniques have been introduced for estimating networks from high-dimensional data in both cases. They divide into model-free approaches and model-driven approaches. Model-free approaches for association networks directly estimate information-theoretic measures, such as mutual information to detect edges in the network (Hartemink, 2005; Margolin et al, 2006). Among model-driven approaches, graphical models have emerged as a powerful class of models and a lot of algorithmic and theoretical advances have occured for *static* (independent and identically distributed) data under the assumption of *sparsity*. For instance, Gaussian graphical models have been thoroughly studied (see (Bühlmann and van de Geer, 2011) and references therein) to infer association networks using static data. Covariance estimation in linear models is the key component of these methods. Covariance selection has been addressed using different forms of norm-regularizers in the loss function to reinforce sparsity in linear models in an unstructured or a structured way. In order to infer causal relationship networks, Bayesian net-

works have been developed either from static data or time-series within the framework of dynamical Bayesian networks. In the case of continuous variables, linear multivariate autoregressive modeling has been developed with again an important focus on sparse models. In this latter framework, Granger causality models have attracted an increasing interest to capture causal relationships.

However very few works to date have focused on network inference for continuous variables in the presence of nonlinear dynamics. In this study, we start from a regularization theory perspective and introduce a general framework for nonlinear multivariate modeling and network inference. Our aim is to extend the framework of sparse linear modeling to that of *sparse nonlinear modeling*. In the machine learning community, a powerful tool to extend linear models to nonlinear ones is based on kernels. The famous kernel trick allows to deal with nonlinear learning problems by working implicitely in a new feature space, where inner products can be computed using a symmetric semi-definite positive function of two variables, called a kernel. In particular, a given kernel allows to build a unique Reproducing Kernel Hilbert Space (RKHS), e.g. a functional space where regularized models can be defined from data using representer theorems. The RKHS theory provides a unified framework for many kernel-based models and a principled way to build new (nonlinear) models. Since multivariate time-series modeling requires defining vector-valued models, we propose to build on operator-valued kernels and their associated reproducing kernel Hilbert space theory (Senkene and Tempel'man, 1973) that were recently introduced in machine learning by (Micchelli and Pontil, 2005) for the multi-task learning problem with vector-valued functions. Among different ongoing research on the subject (Alvarez et al, 2011), new applications concern vector field regression (Baldassarre et al, 2010), structured classification (Dinuzzo and Fukumizu, 2011), functional regression (Kadri et al, 2011) and link prediction (Brouard et al, 2011). However, their use in the context of time series is novel.

We introduce a new family of nonlinear vector autoregressive models based on a new operator-valued kernel. Once an operator-valued kernel-based model is learnt, we compute an empirical estimate of its Jacobian, providing a generic and simple way to extract dependence relationship among variables. Therefore, the operator-valued kernel proposed here was designed to produce not only a good approximation of the systems dynamics, but also a flexible and controllable Jacobian estimate. To obtain sparse networks, we focus on sparsity of the Jacobian, thus extending the sparsity constraint applied to the design matrix regularly employed in linear modeling. We minimize an elastic-net-like cost function to simultaneously control smoothing of the model and sparsity. Learning such models requires identification of both the kernel hyperparameter, here a semi-definite positive matrix, and the parameter vectors. We propose a novel learning and efficient strategy that alternatively uses a proximal gradient algorithm to learn the parameter vectors and an exponentiated gradient algorithm for learning the semi-definite positive matrix that characterizes the kernel. We show that without prior knowledge on the relationship between variables, the proposed algorithm is able to retrieve the network structure of a given underlying dynamical system from the observation of its behavior through time.

The structure of the paper is as follows: in Section 2, we present the general network inference scheme. In Section 3, we recall elements of RKHS theory devoted to vector-valued functions and introduce operator-valued kernel based autoregressive models. Section 4 presents the learning algorithm that estimates both the parameters of the model and the parameters of the kernel. Section 5 illustrates the performance of the model and the algorithm through extensive numerical work based on both synthetic and real data.

2 Network inference from nonlinear vector autoregressive models

Let $\mathbf{x}_t \in \mathbb{R}^d$ denote the *observed* state of a dynamical system comprising of d state variables. We are interested in inferring direct influences of a state variable i on another variable $j \neq i$, $(i,j) \in \{1,\ldots,d\}^2$. Further, we assume that a first-order stationary model is adequate to capture the temporal evolution of the system under study, which can exhibit nonlinear dynamics captured by a function $h : \mathbb{R}^d \to \mathbb{R}^d$:

$$\mathbf{x}_{t+1} = h(\mathbf{x}_t) + \mathbf{u}_t \tag{1}$$

where $\mathbf{u}_t$ is a noise term.

For models where the *network* matrix A is explicitly given (e.g. linear models $h(x_t) = Ax_t$ or parametric models), its estimation (possibly sparse) can be directly accomplished. However, for nonlinear models this is a more involved task. Our strategy is to first learn h from the data and subsequently estimate A by averaging the values of the empirical Jacobian matrix of h, over the whole set of time points. Specifically, denote by $\mathbf{x}_0, \ldots, \mathbf{x}_{N-1}$ the observed time series of the network state. Then, $\forall (i,j) \in \{1,\ldots,d\}^2$, an estimate of the adjacency matrix A is given by:

$$\hat{A}_{ij} = g\left(\frac{1}{N-1}\sum_{t=0}^{N-2}\frac{\partial h(\mathbf{x}_t)_i}{\partial (\mathbf{x}_t)_j}\right) \tag{2}$$

where g is a thresholding function that outputs 0 or 1. In the remainder of the paper, we note $J_{ij}(h)$ the (i,j) coefficient of the average Jacobian of h and $J_{ij}(h)(t)$ its value at a given time t.

Note that to obtain a high quality estimate of the network, we need a class of functions h whose Jacobian matrices can be controlled during learning in such a way that they could provide good continuous approximators of A. In this work, we propose a new class of nonparametric vector autoregressive models that exhibit such properties. Specifically, we introduce Operator-valued Kernel-based Vector AutoRegressive (OKVAR) models, that constitute a rich class as discussed in the next section.

3 Operator-valued kernels and vector autoregressive models

3.1 Elements of RKHS theory for vector-valued functions

We start by introducing the basic building blocks of our model and the necessary notation. In RKHS theory with operator-valued kernels, we consider functions with input in some set $\mathcal{X}$ and with vector values in some given Hilbert space $\mathcal{F}_y$. For completeness, we first describe the general framework and then come back to the case of interest, namely $\mathcal{X} = \mathcal{F}_y = \mathbb{R}^d$. Denote by $L(\mathcal{F}_y)$, the set of all bounded linear operators from $\mathcal{F}_y$ to itself. Given $A \in L(\mathcal{F}_y)$, A^* denotes its adjoint. Then, an operator-valued kernel K is defined as follows:

Definition 1 (Operator-valued kernel) *(Senkene and Tempel'man, 1973; Caponnetto et al, 2008)*
Let $\mathcal{X}$ be a set and $\mathcal{F}_y$ a Hilbert space. Then, $K : \mathcal{X} \times \mathcal{X} \to L(\mathcal{F}_y)$ is a kernel if:

- $\forall \ \ (x, z) \in \mathcal{X} \times \mathcal{X}$, $K(x, z) = K(z, x)^*$
- $\forall m \in \mathbb{N}$, $\forall \{(x_i, \mathbf{y}_i)\}_{i=1}^m \subseteq \mathcal{X} \times \mathcal{F}_y$, $\sum_{i,j=1}^m \langle \mathbf{y}_i, K(x_i, x_j)\mathbf{y}_j \rangle_{\mathcal{F}_y} \geq 0$

The following theorem whose proof can be found in Senkene and Tempel'man (1973); Micchelli and Pontil (2005) establishes that one can build a unique RKHS from a given operator-valued kernel.

Theorem 1 ((Senkene and Tempel'man, 1973; Micchelli and Pontil, 2005)) *Let $\mathcal{X}$ be a set and $\mathcal{F}_y$ be a Hilbert space. If $K : \mathcal{X} \times \mathcal{X} \to L(\mathcal{F}_y)$ is an operator-valued kernel, then there exists a unique RKHS $\mathcal{H}_K$ which admits K as the reproducing kernel; that is*

$$\forall x \in \mathcal{X}, \forall \mathbf{y} \in \mathcal{F}_y, \ \langle h, K(\cdot, x)\mathbf{y} \rangle_{\mathcal{H}} = \langle h(x), \mathbf{y} \rangle_{\mathcal{F}_y} . \tag{3}$$

The RKHS $\mathcal{H}_K$ is built by taking the closure of $span\{K(\cdot, x)\mathbf{y} | x \in \mathcal{X}, \mathbf{y} \in \mathcal{F}_y\}$ endowed with the scalar product $\langle f, g \rangle_{\mathcal{H}_K} = \sum_{i,j} \langle \mathbf{u}_i, K(r_i, s_j)\mathbf{v}_j \rangle_{\mathcal{F}_y}$ with $f(\cdot) = \sum_i K(\cdot, r_i)\mathbf{u}_i$ and $g(\cdot) = \sum_j K(\cdot, s_j)\mathbf{v}_j$. The corresponding norm $\| \cdot \|_{\mathcal{H}_K}$ is defined by $\| f \|^2_{\mathcal{H}_K} = \langle f, f \rangle_{\mathcal{H}_K}$.

For the sake of notational simplicity we omit K and use $\mathcal{H} = \mathcal{H}_K$ in the remainder of the paper. As in the scalar case, one of the most appealing features of RKHS is to provide a theoretical framework for regularization, e.g. representer theorems. Let us consider the case of regression with convex loss functions and denote by $\mathcal{S}_N = \{(x_i, \mathbf{y}_i)\}_{i=0}^{N-1} \subseteq \mathcal{X} \times \mathcal{F}_y$ the data set under consideration.

Theorem 2 ((Micchelli and Pontil, 2005)) *Let V be a convex loss function, and $\lambda > 0$ the regularization parameter. Let $\mathcal{S}_N$ be the data set $\{(x_0, \mathbf{y}_0), \ldots, (x_{N-2}, \mathbf{y}_{N-2})\}$. Then, the minimizer of the following optimization problem:*

$$\underset{h \in \mathcal{H}}{\operatorname{argmin}} \, \mathcal{L}(h) = \sum_{i=0}^{N-2} V(h(x_i), \mathbf{y}_i) + \lambda \|h\|^2_{\mathcal{H}} \ ,$$

admits an expansion:

$$\hat{h}(\cdot) = \sum_{\ell=0}^{N-2} K(\cdot, x_\ell)\mathbf{c}_\ell \ , \tag{4}$$

where the coefficients $\mathbf{c}_\ell, \ell = \{0, \cdots, N-2\}$ *are vectors in the Hilbert space* $\mathcal{F}_y$.

Such a result justifies a new family of models of the form (4) for vector regression in $\mathbb{R}^d$. Then, the operator-valued kernel (OVK) becomes a matrix-valued one. In case this matrix is diagonal, the model reduces to d *independent* models with scalar outputs and there is no need for a matrix-valued kernel. In other cases, when we assume that the different components of the vector-valued function are not independent and may share some underlying structure, a non-diagonal matrix-valued kernel allows to take into consideration similarities between the components of the input vectors. Initial applications of matrix-valued kernels deal with structured output regression tasks, such as multi-tasks learning and structured classification. For both tasks, a decomposable kernel based on the product of a scalar kernel k_1 and a positive semi-definite matrix B has been proposed.

3.2 The OKVAR model

The problem under consideration differs from structured prediction ones and hence some care is required to choose an appropriate kernel. Recall that the objective is to estimate an autoregressive model, which takes the following form: given the observed $d-$dimensional time series $\mathbf{x}_0, \ldots, \mathbf{x}_{N-1}$, h is defined as

$$h(\mathbf{x}_t; \mathcal{S}_N) = \sum_{\ell=0}^{N-2} K(\mathbf{x}_t, \mathbf{x}_\ell).\mathbf{c}_\ell \tag{5}$$

where $K(\cdot,\cdot)$ is a matrix-valued kernel and each $\mathbf{c}_\ell$ ($\ell \in \{0, \ldots, N-2\}$) is a vector of dimension d. In the following, we denote by $C \in \mathcal{M}^{N-1,d}$, the matrix composed of the $N-1$ row vectors $\mathbf{c}_\ell^T$ of dimension d.

As mentioned in the introduction, the network structure will be inferred by the empirical mean $\bar{J}$ of the instantaneous Jacobian matrices $J(t)$ of h over observed time-points. At any given time point t, for a given target state variable i and a matrix-valued kernel-based model h, we have:

$$\forall i \in \{1, \ldots, d\}, J_{ij}(t) = \sum_{\ell=0}^{N-2} \frac{\partial (K(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell)^i}{\partial \mathbf{x}_t^j} \tag{6}$$

Hence, each component of h should be a function of the state variables in such a way that the coefficients of the Jacobian reflect the dependence of the output component on some of the state variables. Due to our assumption of nonlinear dynamics of the underlying system, the kernel should contain nonlinear functions of the state variables. Moreover, a relevant matrix-valued kernel-based model should allow the sparsity to be controlled of the Jacobian through the

values of its parameters. As an example, let us take a look at a decomposable kernel. In our setting ($\mathcal{X} = \mathcal{F}_y = \mathbb{R}^d$), this kernel takes the following form:

$$\forall (\mathbf{x}, \mathbf{z}) \in \mathbb{R}^d \times \mathbb{R}^d, K_1(\mathbf{x}, \mathbf{z}) = k_1(\mathbf{x}, \mathbf{z})B \tag{7}$$

where B is a positive semi-definite matrix of size $d \times d$ and k_1 is a Gaussian kernel with hyperparameter γ_1: $k_1(\mathbf{x}, \mathbf{z}) = \exp(-\gamma_1 \parallel \mathbf{x} - \mathbf{z} \parallel^2)$. One could think that B could be used to encode the network. If we build a model h_1 from this kernel K_1, we see that the nonlinear term involved in the matrix-valued kernel does not differ from one pair (i, j) to another. Then, the corresponding (i, j)-th term of the Jacobian is given by:

$$J(h_1)_{ij}(t) = \sum_{\ell=0}^{N-2} \frac{\partial k_1(\mathbf{x}_t, \mathbf{x}_\ell)}{\partial \mathbf{x}_t^j} (B\mathbf{c}_\ell)^i \tag{8}$$

which implies that it is impossible to control specific values of the Jacobian matrix using B or the $\mathbf{c}$'s. To obtain a richer class of Jacobians than those induced by the decomposable kernels, more suitable for the inference task at hand, we therefore introduce a new matrix-valued kernel, based on the Hadamard product of two matrix-valued kernels, one being a decomposable kernel and the other, a transformable kernel. First defined in Caponnetto et al (2008), transformable kernels are based on a scalar kernel and a map T from $\mathcal{X}$ to some Haussdorf space on which the scalar kernel is defined. In this work, we focus on a Gaussian kernel k_2 with parameter γ_2 and a projection mapping. For $q \in \{1, \cdots, d\}$, let us define the projection on the q-th dimension as $T_q : \mathbb{R}^d \to \mathbb{R}$. For a given $\mathbf{x} \in \mathbb{R}^d$, then $T_q(\mathbf{x}) = x^q$, where x^q is the q-th component of vector $\mathbf{x}$. The following transformable kernel can be shown to be a matrix-valued one:$\forall \ \ (\mathbf{x}, \mathbf{y}) \in \mathbb{R}^d \times \mathbb{R}^d, \forall \ \ (p, q) \in \{1, \ldots, d\}^2$:

$$K_2(\mathbf{x}, \mathbf{y})_{pq} = k_2(T_p(\mathbf{x}), T_q(\mathbf{y})) = \exp(-\gamma_2(x^p - y^q)^2) \tag{9}$$

Note that an interesting feature of this kernel is that each coordinate i of the vector model $h_2(\mathbf{x}_t)^i$ can be expressed as a linear combination of nonlinear functions of variables j: $h_2(\mathbf{x}_t)^i = \sum_\ell \sum_q \exp(-\gamma_2(x_t^i - x_\ell^q)^2) c_\ell^q$. However, the (i, j)-th entry of the Jacobian at time t becomes

$$J(h_2)_{ij}(t) = 2\gamma_2(x_t^i - x_t^j) \exp\left(-\gamma_2(x_t^i - x_t^j)^2\right) c_t^j,$$

which implies that the c_t^j's have the same impact, no matter what the target variable i is. As a consequence, it becomes impossible to control those parameters for network inference purposes.

Nevertheless, if we combine those two kernels appropriately, we can both keep the nonlinear functions of single variables and use the matrices B and C to directly control the values of the Jacobian. The kernel K we propose is the Hadamard product of the decomposable kernel K_1 with the transformable kernel K_2:

$$\forall \ \ (\mathbf{x}, \mathbf{y}) \in \mathbb{R}^d \times \mathbb{R}^d, K(\mathbf{x}, \mathbf{y}) = k_1(\mathbf{x}, \mathbf{y})B \circ K_2(\mathbf{x}, \mathbf{y}) \tag{10}$$

The resulting kernel K possesses the kernel property, i.e:

Proposition 1 *The kernel defined by (10) is a matrix-valued kernel, where* $\circ$ *denotes the Hadamard product for matrices.*

Proof: A Hadamard product of two matrix-valued kernels is a matrix-valued kernel (proposition 4 in Caponnetto et al (2008)).

Next, we introduce the following OKVAR model based on the matrix-valued kernel K:

$$h(\mathbf{x}_t; \mathcal{S}_N) = \sum_{\ell=0}^{N-2} k_1(\mathbf{x}_t, \mathbf{x}_\ell) B \circ K_2(\mathbf{x}_t, \mathbf{x}_\ell).\mathbf{c}_\ell \tag{11}$$

and the entries of its Jacobian at time t $J(h)(t) = J(t)$ are given by:
$J_{ij}(t) = \sum_{\ell=0}^{N-2} \frac{\partial k_1(\mathbf{x}_t, \mathbf{x}_\ell)}{\partial \mathbf{x}_t^j} B \circ K_2(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell + k_1(\mathbf{x}_t, \mathbf{x}_\ell).\frac{\partial B \circ K_2(\mathbf{x}_t, \mathbf{x}_\ell)\mathbf{c}_\ell}{\partial \mathbf{x}_t^j}$
which after some calculations reduces to:

$$\begin{aligned} J_{ij}(t) &= 2\gamma_2 b_{ij}(x_t^i - x_t^j) \exp\left(-\gamma_2 (x_t^i - x_t^j)^2\right) c_t^j \\ &\quad -2\gamma_1 \sum_{\ell \neq t} k_1(\mathbf{x}_t, \mathbf{x}_\ell)(x_t^j - x_\ell^j) \sum_{p=1}^{d} b_{ip} \exp\left(-\gamma_2 (x_t^i - x_\ell^p)^2\right) c_\ell^p \end{aligned} \tag{12}$$

The obtained expression exhibits some interesting characteristics: if we choose γ_1 very close to 0 (for $\gamma_1 = 0$, k_1 is no more a kernel), then $k_1(\mathbf{x}_t, \mathbf{x}_\ell)$ is close to one for any pair $(\mathbf{x}_t, \mathbf{x}_\ell)$ and the second term in (12) is approximatively 0. Then, the value of the Jacobian for variables (i, j) is controlled by the value of b_{ij}: hence, B is capable of imposing structure in the model. For example, for a given pair of variables (i, j), if $b_{ij} = 0$, then irrespective of the values of c_ℓ^j, $\ell = 0, \cdots, N-2$, the corresponding Jacobian coefficient will be zero as well; i.e variable j does not influence variable i. Conversely, a non-zero coefficient b_{ij} does not reflect non-influence from j to i since the $\mathbf{c}$ parameters can still set the corresponding coefficient in the Jacobian to zero. Thus, the parameter B captures some of the structure of the underlying network, *together* with C. Note that the vectors $\mathbf{c}$'s and the cross-difference between coordinates in equation (12) allow us to have non-symmetric Jacobian matrices, suitable for reconstructing directed graphs.

4 Learning the OKVAR model

We employ a loss function analogous to the one used in elastic-net type regularized models in the scalar case. The main goal is to simultaneously learn the matrix of the model parameters C, as well as B, the positive semi-definite matrix underlying the kernel K. Thus, we aim to minimize the following loss function:

$$\mathcal{L}(B, C) = \sum_{t=0}^{N-2} ||h(\mathbf{x}_t; B, C) - \mathbf{x}_{t+1}||^2 + \Omega(B, C) \tag{13}$$

with $\Omega(B,C) = \lambda_h||h_{B,C}||^2_{\mathcal{H}} + \lambda_C||C||_1 + \lambda_B||B||_1$ and subject to the constraint that $B \in \mathcal{S}_d^+$ and $C \in \mathcal{M}^{N-1,d}$ where $\mathcal{S}_d^+$ denotes the cone of positive semi-definite matrices of size $d \times d$. Note that $||\cdot||_1$ denotes *both* the ℓ_1 norm of a vector and that of the vectorized form of a matrix.

For fixed B, the squared norm $||h_{B,C}||^2_{\mathcal{H}} = \sum_{i,j} k(\mathbf{x}_i, \mathbf{x}_j)\mathbf{c}_i^T \left(k_1(\mathbf{x}_t, \mathbf{x}_\ell)B \circ K_2(\mathbf{x}_i, \mathbf{x}_j)\right)\mathbf{c}_j$ plays the role of a weighted ℓ_2 norm on C, while the ℓ_1 norm of C controls the sparsity of the model, necessary for obtaining a sparse Jacobian. For fixed C, the squared norm of h imposes a smoothing constraint on B, while again the ℓ_1 norm of B aids in controlling the sparsity of the model and its Jacobian.

Further, for fixed B, the loss function $\mathcal{L}(B_{fixed}, C)$ is convex in C and conversely, for fixed C, $\mathcal{L}(B, C_{fixed})$ is convex in B. We propose an alternating optimization scheme to minimize the overall loss $\mathcal{L}(B,C)$. Since both loss functions $\mathcal{L}(B_{fixed}, C)$ and $\mathcal{L}(B, C_{fixed})$ involve a sum of two terms, one being differentiable and the other being sub-differentiable we employ proximal gradient algorithms (Beck and Teboulle, 2010) to achieve the minimization. Further, to increase the efficiency of the optimization procedure, we implement a block-coordinate descent strategy focusing on a given vector $\mathbf{c}_\ell$ at each round of the procedure, instead of learning the whole C in one step. Incorporating the minimization of the ℓ^{th} column vector of C implies that we allow the algorithm to incorporate (or not) the ℓ^{th} time-point, given the presence of the ℓ_1 constraint. Instead, we could have chosen to apply the block coordinate gradient descent to the j^{th} row of C; in that case, the emphasis is on eliminating the influence of the j^{th} variable through the corresponding Jacobian coefficient J_{ij}. We have chosen to favor parsimony in the data through the time dimension. For fixed B and for a specific index ℓ for the target coordinate $\mathbf{c}_\ell$ (a column of C) the loss function becomes:

$$\mathcal{L}(\hat{B}, C, \ell) = \sum_{t=0}^{N-2} ||h(\mathbf{x}_t; \hat{B}, C) - \mathbf{x}_{t+1}||^2 + \lambda_h||h_{\hat{B},C}||^2_{\mathcal{H}} + \lambda_C||\mathbf{c}_\ell||_1, \tag{14}$$

while for given $\hat{C}$, it is given by:

$$\mathcal{L}(B, \hat{C}) = \sum_{t=0}^{N-2} ||h(\mathbf{x}_t; B, \hat{C}) - \mathbf{x}_{t+1}||^2 + \lambda_h||h_{B,\hat{C}}||^2_{\mathcal{H}} + \lambda_B||B||_1 \tag{15}$$

In summary, the general form of the algorithm is given in Algorithm (1).

4.1 Learning a vector $\mathbf{c}_\ell$, for fixed B and index ℓ.

Note that the resulting convex loss function is a sum of two terms: $f_{\mathbf{c}_\ell}$ which is differentiable with respect to $\mathbf{c}_\ell$ and $g_{\mathbf{c}_\ell}$ which is non-smooth, but nevertheless convex and subdifferentiable with respect to $\mathbf{c}_\ell$:

$$\mathcal{L}(\hat{B}, C, \ell) = \underbrace{\sum_{t=0}^{N-2} \left\| \sum_{k=0}^{N-2} K(\mathbf{x}_t, \mathbf{x}_k)\mathbf{c}_k - \mathbf{x}_{t+1} \right\|^2 + \lambda_h \sum_{t,k=0}^{N-2} \mathbf{c}_t^T K(\mathbf{x}_t, \mathbf{x}_k)\mathbf{c}_k + \lambda_C \sum_{t \neq \ell} ||\mathbf{c}_t||_1}_{f_{\mathbf{c}_\ell}(\mathbf{c}_\ell)} + \underbrace{\lambda_C||\mathbf{c}_\ell||_1}_{g_{\mathbf{c}_\ell}(\mathbf{c}_\ell)} \tag{16}$$

Algorithm 1 Learning elastic-OKVAR (13)

Inputs : $B_0 \in \mathcal{S}_d^+; \epsilon_B; \epsilon_C$
Initialize : $m = 0$; $\ell = 1$ STOP=false
while STOP=false **do**
  Step 1: Given B_m and ℓ, minimize 14 and obtain $\mathbf{c}_{\ell,m}$
  Step 2: Update the ℓ^{th} column of C_m with $\mathbf{c}_{\ell,m}$
  Step 3: Given C_m, minimize the loss function (15) and obtain B_{m+1}
  if $m > 0$ **then**
    STOP:=$||B_m - B_{m-1}|| \leq \epsilon_B$ and $||C_m - C_{m-1}|| \leq \epsilon_C$
  end if
  Step 4: $m \leftarrow m + 1$
  Step 5: $\ell = \ell + 1$ mod[N-1]
end while

This leads us to employ a proximal gradient algorithm, which is appropriate for solving this subproblem. Its steps are outlined in Algorithm 2: The algorithm

Algorithm 2 Solve Problem (14)

Inputs : ℓ, $C_0 \in \mathcal{M}^{(N-1)d}$;M;$\epsilon_{\mathbf{c}}$;Lipschitz constant of $\nabla_{\mathbf{c}_\ell}$: $L_{\mathbf{c}_\ell} = 2\rho(\sum_{t=0}^{N-2} K(\mathbf{x}_\ell, \mathbf{x}_t)K(\mathbf{x}_t, \mathbf{x}_\ell) + \lambda_h K(\mathbf{x}_\ell, \mathbf{x}_\ell))$
Initialize : $m = 0; \mathbf{y}_\ell^{(1)} = \mathbf{c}_\ell^{(0)}; t^{(1)} = 1$; STOP=false
while $m < M$ and STOP=false **do**
  Step 0: $m \leftarrow m + 1$
  Step 1: $\mathbf{c}_\ell^{(m)} = \text{prox}_{\frac{1}{L_{\mathbf{c}_\ell}}}(g_{\mathbf{c}_\ell}) \left(\mathbf{y}_\ell^{(m)} - \frac{1}{L_{\mathbf{c}_\ell}} \nabla_{\mathbf{y}_\ell^{(m)}} f_{\mathbf{c}_\ell}(\mathbf{y}_\ell^{(m)})\right)$
  if $||\mathbf{c}_\ell^{(m)} - \mathbf{c}_\ell^{(m-1)}|| \leq \epsilon_{\mathbf{c}}$ **then**
    STOP:=true
  else
    Step 2: $t^{(m+1)} = \frac{1+\sqrt{1+4t^{(m)\,2}}}{2}$
    Step 3: $\mathbf{y}_\ell^{(m)} = \mathbf{c}_\ell^{(m)} + \frac{t^{(m)}-1}{t^{(m+1)}} \left(\mathbf{c}_\ell^{(m)} - \mathbf{c}_\ell^{(m-1)}\right)$
  end if
end while

relies on the following: the proximal operator applied to a function g for a given $\mathbf{y} \in \mathbb{R}^p$ is given by: $\text{prox}_t(g)(\mathbf{y}) = \text{argmin}_{\mathbf{u}} \left\{g(\mathbf{u}) + \frac{1}{2t}||\mathbf{u} - \mathbf{y}||^2\right\}$. Denote by $\mathcal{T}_\alpha : \mathbb{R}^d \to \mathbb{R}^d$ the shrinkage/soft threshold operator:
For $i = 1 \ldots d$,

$$\begin{aligned} T_\alpha(\mathbf{x})_i &= (|x_i| - \alpha)_+ \text{sgn}(x_i) \\ &= \begin{cases} x_i - \alpha & \text{if } x_i \geq \alpha \\ 0 & \text{if} -\alpha < x_i < \alpha \\ x_i + \alpha & \text{if } x_i \leq -\alpha \end{cases} \end{aligned}$$

Then, the proximal gradient term in the m^{th} iteration is given by:

$$\text{prox}_{\frac{1}{L_{\mathbf{c}_\ell}}}(g_{\mathbf{c}_\ell})\left(\mathbf{y}_\ell^{(m)} - \frac{1}{L_{\mathbf{c}_\ell}}\nabla_{\mathbf{y}_\ell^{(m)}} f_{\mathbf{c}_\ell}(\mathbf{y}_\ell^{(m)})\right) \quad = \quad \mathcal{T}_{\frac{\lambda_C}{L_{\mathbf{c}_\ell}}}\left(\mathbf{y}_\ell^{(m)} - \frac{1}{L_{\mathbf{c}_\ell}}\nabla_{\mathbf{y}_\ell^{(m)}} f_{\mathbf{c}_\ell}(\mathbf{y}_\ell^{(m)})\right)$$

We first need to calculate the derivative of $f_{\mathbf{c}_\ell}(\mathbf{c}_\ell)$ for variable $\mathbf{c}_\ell$:

- The derivative of the sum of squared errors with respect to $\mathbf{c}_\ell$ is given by: $\frac{\partial}{\partial \mathbf{c}_\ell}\sum_{t=0}^{N-2}\left\|\sum_{k=0}^{N-2} K(\mathbf{x}_t,\mathbf{x}_k)\mathbf{c}_k - \mathbf{x}_{t+1}\right\|^2 = \sum_{t=0}^{N-2} 2K(\mathbf{x}_\ell,\mathbf{x}_t)\left(\sum_{k=0}^{N-2} K(\mathbf{x}_t,\mathbf{x}_k)\mathbf{c}_k - \mathbf{x}_{t+1}\right)$
- Using the property of operator-valued kernels, $K(\mathbf{x}_t,\mathbf{x}_\ell)^T = K(\mathbf{x}_\ell,\mathbf{x}_t)$, the derivative of $||h_{B,C}||^2_{\mathcal{H}}$ with respect to $\mathbf{c}_\ell$ becomes: $\frac{\partial}{\partial \mathbf{c}_\ell}\sum_{t,k=0}^{N-2}\mathbf{c}_t^T K(\mathbf{x}_t,\mathbf{x}_k)\mathbf{c}_k = 2\sum_{t=0}^{N-2} K(\mathbf{x}_\ell,\mathbf{x}_t)\mathbf{c}_t$

Hence,

$$\nabla_{\mathbf{c}_\ell} f_{\mathbf{c}_\ell}(\mathbf{c}_\ell) = 2\sum_{t=0}^{N-2} K(\mathbf{x}_\ell,\mathbf{x}_t)\left(\sum_{k=0}^{N-2} K(\mathbf{x}_t,\mathbf{x}_k)\mathbf{c}_k - \mathbf{x}_{t+1} + \lambda_1\mathbf{c}_t\right) \tag{17}$$

Moreover, we need to define the Lipschitz constant $L_{\mathbf{c}_\ell}$ involved in the proximal gradient algorithm. Using some algebra and properties of the norm, we can establish that: For $\mathbf{c}_\ell^1, \mathbf{c}_\ell^2 \in \mathbb{R}^d$:

$$||\nabla_{\mathbf{c}_\ell} f_{\mathbf{c}_\ell}(\mathbf{c}_\ell^1) - \nabla_{\mathbf{c}_\ell} f_{\mathbf{c}_\ell}(\mathbf{c}_\ell^2)|| \leq \underbrace{2\rho\left(\sum_{t=0}^{N-2} K(\mathbf{x}_\ell,\mathbf{x}_t)K(\mathbf{x}_t,\mathbf{x}_\ell) + \lambda_1 K(\mathbf{x}_\ell,\mathbf{x}_\ell)\right)}_{L_{\mathbf{c}_\ell}} \cdot ||\mathbf{c}_\ell^1 - \mathbf{c}_\ell^2||$$

4.2 Learning the matrix B for fixed C

For given parameter matrix C, the loss function $\mathcal{L}(B,\hat{C})$ is minimized subjecto to the constraint that B is positive semi-definite.

$$\mathcal{L}(B,\hat{C}) = \underbrace{\sum_{t=0}^{N-2} ||h(\mathbf{x}_t;B,\hat{C}) - \mathbf{x}_{t+1}||^2}_{f_B(B)} + \underbrace{\lambda_h||h||^2_{\mathcal{H}}}_{g_B(B)} + \underbrace{\lambda_B||B||_1}_{h_B(B)} \tag{18}$$

Note that both f_B and g_B are differentiable with respect to B, while h_B is non-smooth, but convex and sub-differentiable with respect to B. In order to efficiently estimate B in the cone of semidefinite positive matrices, we adopt a strategy based on gradient updates for matrix exponentials (Tsuda et al, 2005), originally introduced for online learning problems. Instead of the gradient, we use the subgradient due to the presence of the ℓ_1 norm term. This approach solves the following surrogate optimization problem:

$$\min_{B\succ 0} \eta\mathcal{L}(B,\hat{C}) + \Delta(B,\tilde{B}), \tag{19}$$

where $\Delta(B, \tilde{B})$ denotes the Bregman divergence between the unknown B and a target $\tilde{B}$. In practice, $\tilde{B}$ corresponds to the value of B from the previous iteration of the algorithm. The role of the $\Delta(\cdot, \cdot)$ term is to keep the new value of B close to the previous update.

The solution to (19) is given by

$$B = \exp\big(\log \tilde{B} - \eta \text{sym}(\nabla_B \mathcal{L}(B, \hat{C}))\big). \tag{20}$$

The corresponding gradient descent algorithm is presented below. This algo-

Algorithm 3 Algorithm for solving problem (20)

Given an estimate for the matrix $\hat{C}$ we proceed as follows:
Inputs : η
Initialize : $\tilde{G} = \log B_{old}$
Step 1: For all $t \in \{0, \cdots, N-2\}$ predict $\hat{\mathbf{x}}_{t+1} = \sum_{\ell=0}^{N-2} K(\mathbf{x}_t, \mathbf{x}_\ell)\hat{\mathbf{c}}_\ell$
Step 2: Form $G = \tilde{G} - \eta \text{sym}(\nabla_B \mathcal{L}(B_{old}, \hat{C}))$
Step 3: Obtain the spectral decomposition $G = V \Lambda V^T$
Step 4: Update $B_{\text{new}} = \big(V \exp(\Lambda - \alpha I) V^T\big) / \big(\text{trace}(\exp(\Lambda - \alpha I)\big)$, where $\alpha = \max_i \Lambda_{i,i}$
where B_{old} denotes the value of B from the previous iteration.

rithm makes use of the following gradient computations. For ease of presentation, we derive the gradient $\nabla_B \mathcal{L}(\tilde{B}, \hat{C}))$ in element-wise form. We adopt the following notation: $(\cdot)_j$ denotes the $j-th$ row of the corresponding vector, $(\cdot)_{j,i}$ the i-th element of the j-th row and $(\cdot)_{ij}$ the (i,j)-th element of the corresponding matrix. Specifically, for the first term we get

$$\begin{aligned}
\frac{\partial f_B(B)}{\partial b_{ij}} &= \frac{\partial}{\partial b_{ij}} \sum_{p=1}^{d} \left(\mathbf{x}_{t+1} - \sum_{\ell=0}^{N-2} (k_1(\mathbf{x}_t, \mathbf{x}_\ell) B \circ K_2(\mathbf{x}_t, \mathbf{x}_\ell)) \mathbf{c}_\ell \right)_p^2 \\
&= -2 \left(\sum_{\ell=0}^{N-2} k_1(\mathbf{x}_t, \mathbf{x}_\ell) K_2(\mathbf{x}_t, \mathbf{x}_\ell)_{ij} c_\ell^j \right) \left(\mathbf{x}_{t+1}^i - \sum_{\ell=0}^{N-2} \sum_{q=1}^{d} k_1(\mathbf{x}_t, \mathbf{x}_\ell) b_{iq} K_2(\mathbf{x}_t, \mathbf{x}_\ell)_{iq} c_\ell^q \right) \\
&\quad -2 \left(\sum_{\ell=0}^{N-2} k_1(\mathbf{x}_t, \mathbf{x}_\ell) K_2(\mathbf{x}_t, \mathbf{x}_\ell)_{ji} c_\ell^i \right) \left(\mathbf{x}_{t+1}^j - \sum_{\ell=0}^{N-2} \sum_{q=1}^{d} k_1(\mathbf{x}_t, \mathbf{x}_\ell) b_{jq} K_2(\mathbf{x}_t, \mathbf{x}_\ell)_{jq} c_\ell^q \right)
\end{aligned}$$

For the second term, note that

$$||h||^2 = \sum_{t=1}^{N-1} \sum_{\ell=1}^{N-1} \mathbf{c}_t^T \left(k_1(\mathbf{x}_t, \mathbf{x}_\ell) B \circ K_2(\mathbf{x}_t, \mathbf{x}_\ell) \right) \mathbf{c}_\ell$$

to get

$$\frac{\partial g_B(B)}{\partial b_{ij}} = \lambda_h \sum_{t=0}^{N-2} \sum_{\ell=0}^{N-2} c_t^i k_1(\mathbf{x}_t, \mathbf{x}_\ell) K_2(\mathbf{x}_t, \mathbf{x}_\ell)_{ij} c_\ell^j + c_t^j k_1(\mathbf{x}_t, \mathbf{x}_\ell) K_2(\mathbf{x}_t, \mathbf{x}_\ell)_{ji} c_\ell^i$$

For the third term, a subgradient can be computed for the ℓ_1 norm term as:

$$\begin{aligned}\frac{\partial h_B(B)}{\partial b_{ij}} &= \lambda_B \frac{\partial ||B||_1}{\partial b_{ij}} \\ &= \begin{cases} \lambda_B, & \text{if } b_{ij} \geq 0 \\ -\lambda_B, & \text{otherwise} \end{cases}\end{aligned}$$

5 Performance Assessment

The performance of the developed OKVAR model and the proposed optimization algorithm was assessed on two tracks: using simulated data from a biological system (DREAM3 challenge data set) and real climate data (Climate data set).

DREAM3 dataset

We start our investigation by considering data sets obtained from the DREAM3 challenge (Prill et al (2010)). DREAM stands for Dialogue for Reverse Engineering Assessments and Methods `http://wiki.c2b2.columbia.edu/dream/index.php/The_DREAM_Project` and is a scientific consortium that organizes challenges in computational biology and especially for gene regulatory network inference. In a gene regulatory network, a gene i is said to regulate another gene j if the expression of gene i at time t influences the expression of gene j at time $t+1$. Its behavior as a dynamical system can be observed through time-series of gene expression. The DREAM3 project provides realistic simulated data for several networks corresponding to different organisms (e.g. E-Coli, Yeast, etc.) of different sizes and topological complexity. We focus here on size-10 and size-100 networks generated for the DREAM3 in-silico challenges. Each of these networks corresponds to a subgraph of the currently accepted *E. coli* and *S. cerevisiae* gene regulatory networks and exhibits varying patterns of sparsity and topological structure. They are referred to as Ecoli1, Ecoli2, Yeast1, Yeast2 and Yeast3 with an indication of their size. The data were generated by imbuing the networks with dynamics from a thermodynamic model of gene expression and Gaussian noise. Specifically, 4 and 46 time series consisting of 21 points corresponding respectively to size-10 and size-100 networks were selected for our assessment investigation.

In all the experiments conducted, we assess the performance of our model using the area under the ROC curve (AUROC) and under the Precision-Recall curve (AUPR) for regulation ignoring the sign (positive vs negative influence). For the DREAM3 data sets we also show the best results obtained from other competing teams using **only** time course data. The challenge made available other data sets, including ones obtained from perturbation (knock-out/knock-down) experiments, as well as observing the organism in steady state, but these were not considered in the results shown in the ensuing Tables. Hence, in our evaluation we address the challenging problem of network based only on time-series data.

The alternate optimization scheme to learn the elastic-OKVAR model depends on the positive semidefinite matrix B_0 given at the start of the algorithm, which may produce different solutions for each initialization. To address that issue, the algorithm was run $nRun = 10$ times. Hence, the predictions of each run are combined to build a *consensus* network. Specifically, for each pair of nodes, we compute the frequency that the edge appears over multiple runs and if the frequency exceeds a preset threshold, the edge remains in the final predicted network, otherwise it is discarded.

Further, multiple (L) time series may also be available, because of multiple related initial conditions and/or technical replicates. In this case, the procedure is repeated accordingly and the $L \cdot nRun$ obtained networks are combined as described above to provide a final **consensus** network. We set $\hat{A}_{ij} = 1$ if and only if $\sum_{r=1}^{L \cdot nRun} |\hat{A}_{ij}^{(r)}| \geq f_{cons} \cdot L \cdot nRun$, where $\hat{A}^{(r)}$ is the estimated adjacency matrix for run number r and $f_{cons} \in [0, 1]$ is the consensus threshold level for edge acceptance.

Table 1: AUROC for the OKVAR model ($\lambda_h = 1, \lambda_C = 1, \lambda_B = 0.1$), LASSO, Team 236 and Team 190 (DREAM3 competing teams) run on DREAM3 size-10 networks. Due to multiple runs and multiple time-series, average ± standard deviation (OKVAR (1TS)) and consensus (OKVAR (4TS)) results are given for the OKVAR model. The numbers in **boldface** are the maximum values of each method.

Size-10	Ecoli1	Ecoli2	Yeast1	Yeast2	Yeast3
OKVAR + *True B*	0.956	0.918	0.806	0.781	0.780
OKVAR (1TS)	0.619 ± 0.104	0.611 ± 0.140	0.549 ±0.070	0.658 ±0.068	0.650 ±0.100
OKVAR (4TS)	**0.717**	**0.724**	0.644	**0.687**	**0.705**
LASSO	0.500	0.547	0.528	0.627	0.582
Team 236	0.621	0.650	**0.646**	0.438	0.488
Team 190	0.573	0.515	0.631	0.577	0.603

The performance of the OKVAR algorithm for prediction of the network structure is presented in Tables 1,2,3,4. The entries of these Tables correspond to the following methods: the base OKVAR learner alone when the true B is provided, the OKVAR model with multiple runs using a single time series and all the available time series, a LASSO algorithm that aims to obtain a sparse network structure and finally the results from two competing teams that exhibited a very good performance based only on similar time-series data.

Note that the base learner corresponds to an elastic-OKVAR model with the true adjecency matrix given, and projected onto the positive semidefinite cone. The row presenting the results of the LASSO algorithm corresponds to an ℓ_1-regularized linear least squares regression model of the form $x_{t+1,i} = \mathbf{x}_t \beta$, applied to each dimension (gene i). An edge is assigned for each nonzero β coefficient. The LASSO algorithm employed all the available time series and a final consensus network is built in the same fashion as delineated above. Although

Table 2: AUPR for the OKVAR model ($\lambda_h = 1, \lambda_C = 1, \lambda_B = 0.1$), LASSO, Team 236 and Team 190 (DREAM3 competing teams) run on DREAM3 size-10 networks. Due to multiple runs and multiple time-series, average ± standard deviation (OKVAR (1TS)) and consensus (OKVAR (4TS)) results are given for the OKVAR model. The numbers in **boldface** are the maximum values of each method.

Size-10	Ecoli1	Ecoli2	Yeast1	Yeast2	Yeast3
OKVAR + *True B*	0.752	0.677	0.473	0.523	0.586
OKVAR (1TS)	0.182 ± 0.088	0.267 ±0.169	0.138 ±0.049	**0.414** ±0.104	0.376 ±0.117
OKVAR (4TS)	**0.385**	**0.678**	**0.430**	0.363	**0.447**
LASSO	0.119	0.531	0.244	0.305	0.255
Team 236	0.197	0.378	0.194	0.236	0.239
Team 190	0.152	0.181	0.167	0.371	0.373

there is no information on the structure of team 236's algorithm, its authors responded to the post-competition DREAM3 survey stating that their method employs Bayesian models with an in-degree constraint (Prill et al (2010)). Team 190 (Tables 1,2) reported in the same survey that their method is also Bayesian with a focus on nonlinear dynamics and local optimization.

The AUROC and AUPR values obtained for size-10 networks (Tables 1,2) strongly indicate that the OKVAR model outperforms the LASSO model and the teams that exclusively used the same set of time series data in the DREAM3 competition except for size-10 Yeast1 (equivalent AUROC). In particular, we note that the OKVAR consensus runs exhibited excellent AUPR values compared to those obtained by teams 236 and 190.

A comparison of competing algorithms for size-100 networks (Tables 3,4) shows that the OKVAR method again achieves superior AUROC results compared to Team 236, the only team that exclusively used time series data for the size-100 network challenge, since Team 190 did not submit any results. The OKVAR method only lags behind by a slight margin for size-100 Yeast1 and Yeast3 in terms of AUPR. It is noticeable that the AUPR values in all rows are rather small (lower than 10%) compared to their size-10 counterparts. Such a decrease suggests that the AUPR values can be impacted more strongly by the lower density of the size-100 networks, where the *non-edges* class severely outnumbers the *edges* class, rather than the choice of algorithm. Such difficult tasks require much more available time-series to achieve better results in terms of AUROC and AUPR. Finally, it is worth noting that that the OKVAR model would have ranked in the top five and ten, respectively for size-10 and size-100 challenges, while the best results employed knock-out/knock-down data in addition to time-series data, the latter being rich in information content Michailidis (2012).

Climate dataset

Our second example examines climate data, originally presented in Liu et al (2010). It contains measurements on climate forcing factors and feedback mechanisms obtained from different databases. In this study, we extracted monthly measurements for 12 variables for the year 2002 (i.e. 12 time-points) that include temperature (TMP), precipitation (PRE), vapor (VAP), cloud cover (CLD), wet days (WET), frost days (FRS), Methane (CH4), Carbon Dioxide (CO2), Hydrogen (H2), carbon monoxide (CO), solar radiation (SOL) and aerosols (AER). The measurements were obtained from 125 meteorological stations located throughout the United States, corresponding to an equally spaced 2.5 $\times$ 2.5 grid. We used the developed OKVAR model to identify and explore dependencies between natural and anthropogenic[1] factors. The model allows learning a separate causal model for each of the multivariate time series for a specific area in the United States. Therefore, for the sake of exposition clarity, we first

[1] linked to human activity

Table 3: AUROC for the OKVAR model ($\lambda_h = 0.01, \lambda_C = 0.01, \lambda_B = 0.01$), LASSO, Team 236 (DREAM3 competing team) run on DREAM3 size-100 networks. Due to multiple runs and multiple time-series, average $\pm$ standard deviation (OKVAR (1TS)) and consensus (OKVAR (46TS)) results are given for the OKVAR model. The numbers in **boldface** are the maximum values of each method.

Size-100	Ecoli1	Ecoli2	Yeast1	Yeast2	Yeast3
OKVAR + *True B*	0.962	0.971	0.958	0.906	0.897
OKVAR (1TS)	0.558 $\pm$ 0.051	0.555 $\pm$ 0.062	0.499 $\pm$ 0.025	0.525 $\pm$ 0.022	0.499 $\pm$0.021
OKVAR (46TS)	**0.618**	**0.620**	**0.537**	**0.553**	**0.522**
LASSO	0.519	0.512	0.507	0.530	0.506
Team 236	0.527	0.546	0.532	0.508	0.508

Table 4: AUPR for the OKVAR model ($\lambda_h = 0.01, \lambda_C = 0.01, \lambda_B = 0.01$), LASSO, Team 236 (DREAM3 competing team) run on DREAM3 size-100 networks. Due to multiple runs and multiple time-series, average $\pm$ standard deviation (OKVAR (1TS)) and consensus (OKVAR (46TS)) results are given for the OKVAR model. The numbers in **boldface** are the maximum values of each method.

Size-100	Ecoli1	Ecoli2	Yeast1	Yeast2	Yeast3
OKVAR + *True B*	0.432	0.516	0.279	0.407	0.364
OKVAR (1TS)	0.017 $\pm$ 0.004	0.016 $\pm$0.004	0.017 $\pm$ 0.002	0.042 $\pm$ 0.004	0.056 $\pm$0.005
OKVAR (46TS)	**0.029**	**0.093**	0.024	**0.052**	0.053
LASSO	0.016	0.057	0.016	0.044	0.044
Team 236	0.019	0.042	**0.035**	0.046	**0.065**

consider only a single location, in northern Texas.

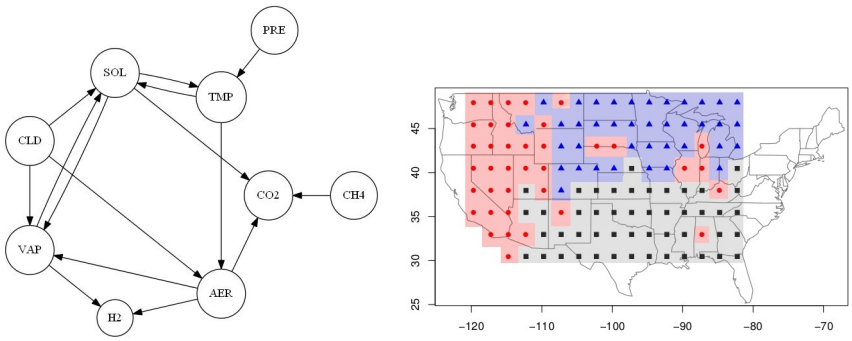
Table 5: Average($\times 10^6$)$\pm$standard deviation($\times 10^5$) BIC for the climate data set on 1 location (Northern Texas). The OKVAR algorithm was run 10 times for each couple of hyperparameters.

		λ_B				
		10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
λ_C	10^{-5}	1.53 ± 1.12	1.43 ± 1.47	1.45 ± 1.84	1.41 ± 1.20	1.43 ± 1.22
	10^{-4}	**1.38 ± 1.05**	1.45 ± 1.60	1.46 ± 1.21	1.54 ± 1.76	1.44 ± 1.02
	10^{-3}	1.50 ± 1.80	1.44 ± 1.26	1.44± 1.61	1.57 ± 1.27	1.43± 0.87
	10^{-2}	1.57 ±1.72	1.42 ± 1.78	1.41 ± 1.62	1.53 ± 1.16	1.39 ±0.90
	10^{-1}	1.46 ± 1.41	1.51 ± 1.95	1.40 ± 1.67	1.56 ± 1.68	1.50 ± 1.55

Similarly to our previous experimental work, we used a grid search in order to set up the hyperparameters of the model. We looked for a combination of λ_C and λ_B values that minimize the mean of the Bayesian Information Criterion (BIC) computed over ten independent runs, using the data of the northern Texas location. As can be seen in Table 5, we selected the values 10^{-4} and 10^{-5} for λ_C and λ_B, respectively. With this set up, we subsequently applied the OKVAR algorithm to the data sets for all available locations. A consensus graph was constructed by retaining only those directed edges whose frequency over multiple runs exceeded a predefined selection threshold. To ease the interpretation of the extracted graph, we considered a stringent selection threshold of 0.8. Indeed, considering the extreme complexity of the system under study, ultimately, dependencies might be found between most variables. However, in this experiment we are interested to identify the strongest and most robust relationships between the variables under consideration. For our first experiment focusing on the norte Texas dataset, this resulted in a parsimonious directed graph shown in Figure 1a.

In their work, Liu et al (2010) inspected more closely the general in-links to Temperature (TMP) and found that it was mostly affected by solar radiance (SOL), cloud cover (CLD), wet day (WET) and aerosols (AER). While the WET variable has not been retained in our final model, we also found that TMP is affected by SOL. It is directly related with AER while it is indirectly influenced by CLD through SOL.

Most of the edges that the OKVAR model identified are reasonable and supported by external knowledge about their interactions: specifically, Aerosols (AER) interact with Hydrogen (H2) through atmospheric chemistry and lower the presence of vapor (VAP) by favoring water condensation. Solar radiance (SOL) is impacted by both Cloud cover (CLD) and vapor concentration (VAP). Temperature is influenced by precipitation (PRE) and solar radiance (SOL). Finally, Carbon Dioxide concentrations (CO2) depends on Methane (CH4) and aerosols (AER) concentrations as eventually, those compounds will degrade into



(a) (b)

Figure 1: (a) Consensus graph (selection threshold = 0.8) extracted from the climate causal model applied to the north Texas data set, based on 10 independent runs of the OKVAR algorithm. (b) Partitioning of the causal models learned by the OKVAR model, over the United States. All extracted causal graphs have been clustered and labeled using spectral clustering.

CO2. Finally, Clouds (CLD) trap the aerosols (AER) which might be cleaned later on by rain. However, this causality might be reversed as the concentration in aerosols has an impact on the cloud cover.
On the contrary, the impact of solar radiance (SOL) on Carbon Dioxide (CO2) Concentration is unclear. Photosynthesis which is activated by the sun and consumes CO2, might explain this feature but this explanation is unlikely.
It is worth noting that errors tend to appear with direct loops. Indeed, edges with opposite direction compared to those we mentioned above (TMP $\Rightarrow$ SOL and SOL $\Rightarrow$ VAP) do not seem relevant.
In addition, some causal relationships such as TMP $\Rightarrow$ AER or TMP $\Rightarrow$ SOL would be more likely if they were reversed as one expects solar radiance (SOL) and aerosols (AER) to promote temperature increase. The same applies to CLD $\Rightarrow$ VAP as the likelihood of clouds (CLD) appearing increases with vapor (VAP) presence.

Of course, some causal influences are still missing in our final model. One would expect an influence of the cloud cover (CLD) on the precipitations (PRE), or an impact of the concentration of greenhouse gazes such as Carbon Dioxide (CO2) or Methane (CH4) on the temperature (TMP). However, most of these edges appeared in the initial consensus graph and would have been recovered with a lower selection threshold.

Since the physical and chemical processes at work in the atmosphere do not change drastically from one state to another, one would expect that the causal graphs learned across the US exhibits a certain degree of similarity. However, in the meantime, it is very likely that causal graphs corresponding to distant areas will show topological differences due to regional specificities regarding both

climate and human activity. In order to assess the stability of our approach, we repeated the former experiment by applying the OKVAR model to all 125 available data sets generated on different locations, thus obtaining 125 causal models.

We defined the structured similarity s between two graphs G_1 and G_2 based on the Hamming Distance between the corresponding adjacency matrices A_1 and A_2: $s(G_1, G_2) = 1 - \frac{1}{d^2} \sum_{i,j} |A_{1ij} - A_{2ij}|$. A spectral clustering algorithm (Ng et al, 2001) using this similarity matrix with the number of classes set to 3 was applied; the number of clusters selected a priori was based on the number of hidden variables considered in Liu et al (2010) for their latent variable model focusing on spatial interactions. Spectral clustering relies on the spectral decomposition of the Laplacian of the similarity matrix, followed by an application of a k-means algorithm in the new feature space. Figure 1b shows the labels of the resulting graphs and their corresponding location on map of the United States. A very clear segmentation of geographical locations emerges, exhibiting the same network structure. There is a split between the northern and southern parts of the country from the Atlantic coast all the way to Rocky Mountains possibly due to climate (typically temperature) differences, while a third zone covers the West, where high levels of CO2 concentration play a role.

6 Conclusion

Network inference from multivariate time-series represents a key problem in numerous scientific domains. In this paper, we address it by introducing and learning a nonlinear vector autoregressive model based on a novel operator-valued kernel. The model generalizes linear vector autoregressive models, and the proposed operator-valued kernel is governed by a semi-definite matrix which is learnt together with the other parameters of the model by minimizing an elastic-net cost function. An alternating optimization scheme based on a proximal gradient algorithm and a matrix exponentiated gradient algorithm is derived. Results obtained from benchmark size-10 and size-100 networks as well as from real datasets show very good performance of the OKVAR model. Future extensions include the use of the model in ensemble methods and also applications to other scientific fields.

Acknowledgements FA-B's work was supported in part by ANR (call SYSCOM-2009, ODESSA project). GM's work was supported in part by NSF grants DMS-1161838 and DMS-1228164.

References

Alvarez MA, Rosasco L, D LN (2011) Kernels for vector-valued functions: a review. Tech. rep., MIT_CSAIL-TR-2011-033

Auliac C, Frouin V, Gidrol X, d'Alché Buc F (2008) Evolutionary approaches for the reverse-engineering of gene regulatory networks: a study on a biologically realistic dataset. BMC Bioinformatics 9:91, DOI 10.1186/1471-2105-9-91, URL http://dx.doi.org/10.1186/1471-2105-9-91

Baldassarre L, Rosasco L, Barla A, Verri A (2010) Vector field learning via spectral filtering. In: Balczar J, Bonchi F, Gionis A, Sebag M (eds) Machine Learning and Knowledge Discovery in Databases, Lecture Notes in Computer Science, vol 6321, Springer Berlin / Heidelberg, pp 56–71

Beck A, Teboulle M (2010) Gradient-based algorithms with applications to signal recovery problems. In: Palomar D, Eldar Y (eds) Convex Optimization in Signal Processing and Communications, Cambridge press, pp 42–88

Bolstad A, Van Veen B, Nowak R (2011) Causal network inference via group sparsity regularization. IEEE Trans Signal Process 59(6):2628–2641

Brouard C, d'Alché Buc F, Szafranski M (2011) Semi-supervised penalized output kernel regression for link prediction. In: International Conference on Machine Learning, pp 593–600

Bühlmann P, van de Geer S (2011) Statistics for High-Dimensional Data: Methods, Theory and Applications. Springer

Caponnetto A, CA, Pontil M, Ying Y (2008) Universal multitask kernels. J Mach Learn Res 9

Chatterjee S, Steinhaeuser K, Banerjee A, Chatterjee S, Ganguly AR (2012) Sparse group lasso: Consistency and climate applications. In: SDM, SIAM / Omnipress, pp 47–58

Dinuzzo F, Fukumizu K (2011) Learning low-rank output kernels. In: Proceedings of the 3rd Asian Conference on Machine Learning, JMLR: Workshop and Conference Proceedings, vol 20

Dondelinger F, Lèbre S, Husmeier D (2012) Non-homogeneous dynamic bayesian networks with bayesian regularization for inferring gene regulatory networks with gradually time-varying structure. Machine Learning

Gilchrist S, Yankov V, Zakrajšek E (2009) Credit market shocks and economic fluctuations: Evidence from corporate bond and stock markets. Journal of Monetary Economics 56(4):471 – 493, DOI 10.1016/j.jmoneco.2009.03.017, URL http://www.sciencedirect.com/science/article/pii/S0304393209000440

Hartemink A (2005) Reverse engineering gene regulatory networks 23:554–555

Kadri H, Rabaoui A, Preux P, Duflos E, Rakotomamonjy A (2011) Functional regularized least squares classication with operator-valued kernels. In: International Conference on Machine Learning, pp 993–1000

Kolaczyk ED (2009) Statistical Analysis of Network Data: Methods and Models, vol In Press,. Springer

Kramer MA, Eden UT, Cash SS, Kolaczyk ED (2009) Network inference with confidence from multivariate time series. Physical Review E 79(6):061,916+, DOI 10.1103/physreve.79.061916, URL `http://dx.doi.org/10.1103/physreve.79.061916`

Lawrence N, Girolami M, Rattray M, Sanguinetti G (eds) (2010) Learning and Inference in Computational Systems Biology. MIT Press

Liu Y, Niculescu-Mizil A, Lozano A (2010) Learning temporal causal graphs for relational time-series analysis. In: Fürnkranz J, Joachims T (eds) Proceedings of the 27th International Conference on Machine Learning (ICML-10)

Maathuis M, Colombo D, Kalish M, Bühlmann P (2010) Predicting causal effects in large-scale systems from observational data. Nature Methods 7:247–248

Margolin I Aand Nemenman, Basso K, Wiggins C, Stolovitzky G, Favera R, Califano A (2006) Aracne: an algorithm for the reconstruction of gene regulatory networks in a mammalian cellular context. BMC Bioinf 7(Suppl 1):S7

Meinshausen N, Bühlmann P (2006) High dimensional graphs and variable selection with the lasso. Annals of Statistics 34:1436–1462

Micchelli CA, Pontil MA (2005) On learning vector-valued functions. Neural Computation 17:177–204

Michailidis G (2012) Statistical challenges in biological networks. Journal of Computational and Graphical Statistics 21(4):840–855

Morton R, Williams KC (2010) Experimental Political Science and the Study of Causality. Cambridge University Press

Murphy KP (1998) Dynamic bayesian networks: representation, inference and learning. PhD thesis, Computer Science, University of Berkeley, CA, USA

Ng AY, Jordan MI, Weiss Y (2001) On spectral clustering: Analysis and an algorithm. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS, MIT Press, pp 849–856

Parry M, Canziani O, Palutikof J, van der Linden P, Hanson C, et al (2007) Climate change 2007: impacts, adaptation and vulnerability. Intergovernmental Panel on Climate Change

Perrin BE, Ralaivola L, A M, S B, Mallet J, d'Alché Buc F (2003) Gene networks inference using dynamic bayesian networks. Bioinformatics 19(S2):38–48

Poole K, Rosenthal H (1997) Congress: A Political-Economic History of Roll Call Voting. Oxford University Press

Prill R, Marbach D, Saez-Rodriguez J, Sorger P, Alexopoulos L, Xue X, Clarke N, Altan-Bonnet G, Stolovitzky G (2010) Towards a rigorous assessment of systems biology models: The dream3 challenges. PLoS ONE 5(2):e9202

Senkene E, Tempel'man A (1973) Hilbert spaces of operator-valued functions. Lithuanian Mathematical Journal 13(4)

Shojaie A, Michailidis G (2010) Discovering graphical granger causality using a truncating lasso penalty. Bioinformatics 26(18):i517–i523

Tsuda K, Raetsch G, Warmuth M (2005) Matrix exponentiated gradient updates for online learning and bregman projection. Journal of Machine Learning Research 6::995–1018

Zou C, Feng J (2009) Granger causality vs. dynamic bayesian network inference: a comparative study. BMC Bioinformatics 10(1):122, DOI 10.1186/1471-2105-10-122, URL http://www.biomedcentral.com/1471-2105/10/122