

Forecasting and Granger Modelling with Non-linear Dynamical Dependencies

Magda Gregorová^{1,2}, Alexandros Kalousis^{1,2}, and Stéphane Marchand-Maillet²

¹ Geneva School of Business Administration, HES-SO University of Applied Sciences
of Western Switzerland

² University of Geneva, Switzerland

Abstract. Traditional linear methods for forecasting multivariate time series are not able to satisfactorily model the non-linear dependencies that may exist in non-Gaussian series. We build on the theory of learning vector-valued functions in the reproducing kernel Hilbert space and develop a method for learning prediction functions that accommodate such non-linearities. The method not only learns the predictive function but also the matrix-valued kernel underlying the function search space directly from the data. Our approach is based on learning multiple matrix-valued kernels, each of those composed of a set of input kernels and a set of output kernels learned in the cone of positive semi-definite matrices. In addition to superior predictive performance in the presence of strong non-linearities, our method also recovers the hidden dynamic relationships between the series and thus is a new alternative to existing graphical Granger techniques.

1 Introduction

Traditional methods for forecasting stationary multivariate time series from their own past are derived from the classical linear ARMA modelling. In these, the prediction of the next point in the future of the series is constructed as a linear function of the past observations. The use of linear functions as the predictors is in part based on the Wold representation theorem (e.g. [6]) and in part, probably more importantly, on the fact that the linear predictor is the best predictor (in the mean-square-error sense) in case the time series is Gaussian.

The Gaussian assumption is therefore often adopted in the analysis of time series to justify the simple linear modelling. However, it is indeed a simplifying assumption since for non-Gaussian series the best predictor may very well be a non-linear function of the past observations. A number of parametric non-linear models has been proposed in the literature, each adapted to capture specific sources of non-linearity (for example multiple forms of regime-switching models, e.g. [20]).

In this paper we adopt an approach that does not rely on such prior assumptions for the function form. We propose to learn the predictor as a general vector-valued function \mathbf{f} that takes as input the past observations of the multivariate series and outputs the forecast of the unknown next value (vector).

We have two principal requirements on the function \mathbf{f} . The first is the standard prediction accuracy requirement. That is, the function \mathbf{f} shall be such that we can expect its outputs to be close (in the squared error sense) to the true future observations of the process. The second requirement is that the function \mathbf{f} shall have a structure that will enable the analysis of the relationships amongst the subprocesses of the multivariate series. Namely, we wish to understand how parts of the series help in forecasting other parts of the multivariate series, a concept known in the time-series literature as graphical Granger modelling [11, 9].

To learn such a function \mathbf{f} we employ the framework of regularised learning of vector-valued functions in the reproducing kernel Hilbert space (RKHS) [17]. Learning methods based on the RKHS theory have previously been considered for time series modelling (e.g. [10, 19, 15]). Though, as Pillonetto et al. note in their survey [18], their adoption for the dynamical system analysis is not a commonplace.

A critical step in kernel-based methods for learning vector-valued functions is the specification of the operator-valued kernel that exploits well the relationships between the inputs and the outputs. A convenient and well-studied class of operator-valued kernels (e.g. in [7, 8, 12]) are those decomposable into a product of a scalar kernel on the input space (input kernel) and a linear operator on the output space (output kernel).

The kernel uniquely determines the function space within which the function \mathbf{f} is learned. It thus has significant influence on both our objectives described above. Instead of having to choose the input and the output kernels a priori, we introduce a method for learning the input and output kernels from the data together with learning the vector-valued function \mathbf{f} .

Our method combines in a novel way the multiple-kernel learning (MKL) approach [14] with learning the output kernels within the space of positive semidefinite linear operators on the output space [12]. MKL methods for operator-valued kernels have recently been developed in [13] and [19]. The first learns a convex combination of a set of operator-valued kernels fixed in advance, the second combines a fixed set of input kernels with a single learned output kernel. To the best of our knowledge, ours is the first method in which the operator-valued kernel is learned by combining a set of input kernels with a set of multiple learned output kernels.

In accordance with our second objective stated above, we impose specific structural constraints on the function search space so that the learned function supports the graphical Granger analysis. We achieve this by working with matrix-valued kernels operating over input partitions restricted to single input scalar series (similar input partitioning has recently been used in [19]).

We impose diagonal structure on the output kernels to control the model complexity. Though this has a cost in the inability to model contemporaneous relationships, it addresses the strong over-parametrisation in a principled manner. It also greatly simplifies the final structure of the problem, which, in result, suitably decomposes into a set of smaller independent problems solvable in parallel.

We develop two forms of sparsity-promoting regularisation approaches for learning the output kernels. These are based on the ℓ_1 and ℓ_1/ℓ_2 norms respectively and are motivated by the search for Granger-causality relationships. As to our knowledge, the latter has not been previously used in the context of MKL.

Finally, we confirm on experiments the benefits our methods can bring to forecasting non-Gaussian series in terms of improved predictive accuracy and the ability to recover hidden dynamic dependency structure within the time series systems. This makes them valid alternatives to the state-of-the-art graphical Granger techniques.

Notation We use bold upper case and lower case letters for matrices and vectors respectively, and the plain letters with subscripts for their elements. For any matrix or vector the superscript T denotes its transpose. Vectors are by convention column-wise so that $\mathbf{x} = (x_1, \dots, x_n)^T$ is the n -dimensional vector \mathbf{x} . $\mathbb{R}, \mathbb{R}^n, \mathbb{R}^{m \times n}$ are the sets of real scalars, n -dimensional vectors, and $m \times n$ dimensional matrices. $\mathbb{R}_+^{m \times n}$ is the set of non-negative matrices, \mathbb{S}_+^m the set of positive semi-definite $m \times m$ matrices and \mathbb{D}_+^m the set of non-negative diagonal matrices. \mathbb{N}_m is the set of positive integers $\{1, \dots, m\}$. For any vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, $\langle \mathbf{x}, \mathbf{y} \rangle, \|\mathbf{x}\|_1, \|\mathbf{x}\|_2$ are the standard inner product, ℓ_1 and ℓ_2 norms in the real Hilbert spaces. For any square matrix \mathbf{A} , $\text{Tr}(\mathbf{A})$ denotes the trace. For any two matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$, $\langle \mathbf{A}, \mathbf{B} \rangle_F := \text{Tr}(\mathbf{A}^T \mathbf{B})$ is the Frobenius inner product and $\|\mathbf{A}\|_F := \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle_F}$ the Frobenius norm. $\langle \cdot, \cdot \rangle_{\mathcal{F}}$ and $\|\cdot\|_{\mathcal{F}}$ are the inner product and norm in the Hilbert space \mathcal{F} .

2 Problem Formulation

Given a realisation of a discrete stationary multivariate time series process $\{\mathbf{y}_t \in \mathcal{Y} \subseteq \mathbb{R}^m : t \in \mathbb{N}_n\}$, our goal is to learn a vector-valued function $\mathbf{f} : \mathcal{Y}^p \rightarrow \mathcal{Y}$ that takes as input the p past observations of the process and predicts its future vector value (one step ahead). The function \mathbf{f} shall be such that

1. we can expect the prediction to be near (in the Euclidean distance sense) the unobserved future value
2. its structure allows to analyse if parts (subprocesses) of the series are useful for forecasting other subprocesses within the series or if some subprocesses can be forecast independently of the rest; in short, it allows Granger-causality analysis [11, 9].

For notational simplicity, from now on we indicate the output of the function \mathbf{f} as $\mathbf{y} \in \mathcal{Y} \subseteq \mathbb{R}^m$ and the input as $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^{mp}$ (bearing in mind that $\mathcal{X} = \mathcal{Y}^p$ is in fact the p -th order Cartesian product of \mathcal{Y} and that the inputs \mathbf{x} and outputs \mathbf{y} are the past and future observations of the same m -dimensional series). We also align the time indexes so that our data sample consists of input-output data pairs $\{(\mathbf{y}_t, \mathbf{x}_t) : t \in \mathbb{N}_n\}$.

Following the standard function learning theory, we will learn $\mathbf{f} \in \mathcal{F}$ by minimising the regularised empirical squared-error risk (with a regularization

parameter $\lambda > 0$)

$$\begin{aligned} \hat{\mathbf{f}} &= \arg \min_{\mathbf{f} \in \mathcal{F}} R(\mathbf{f}) \\ R(\mathbf{f}) &:= \sum_{t=1}^T \|\mathbf{y}_t - \mathbf{f}(\mathbf{x}_t)\|_2^2 + \lambda \|\mathbf{f}\|_{\mathcal{F}}^2 . \end{aligned} \quad (1)$$

Here \mathcal{F} is the reproducing kernel Hilbert space (RKHS) of \mathbb{R}^m -valued functions endowed with the norm $\|\cdot\|_{\mathcal{F}}$ and the inner product $\langle \cdot, \cdot \rangle_{\mathcal{F}}$. The RKHS is uniquely associated with a symmetric positive-semidefinite matrix-valued kernel $\mathbf{H} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}^{m \times m}$ with the reproducing property

$$\langle \mathbf{y}, \mathbf{g}(\mathbf{x}) \rangle = \langle \mathbf{H}_{\mathbf{x}} \mathbf{y}, \mathbf{g} \rangle_{\mathcal{F}} \quad \forall (\mathbf{y}, \mathbf{x}, \mathbf{g}) \in (\mathcal{Y}, \mathcal{X}, \mathcal{F}) ,$$

where the map $\mathbf{H}_{\mathbf{x}} : \mathcal{X} \rightarrow \mathbb{R}^{m \times m}$ is the kernel section of \mathbf{H} centred at \mathbf{x} such that $\mathbf{H}_{\mathbf{x}_i}(\mathbf{x}_j) = \mathbf{H}(\mathbf{x}_i, \mathbf{x}_j)$ for all $(\mathbf{x}_i, \mathbf{x}_j) \in (\mathcal{X}, \mathcal{X})$. From the classical result in [17], the unique solution $\hat{\mathbf{f}}$ of the variational problem (1) admits a finite dimensional representation

$$\hat{\mathbf{f}} = \sum_{t=1}^T \mathbf{H}_{\mathbf{x}_t} \mathbf{c}_t , \quad (2)$$

where the coefficients $\mathbf{c}_t \in \mathcal{Y}$ are the solutions of the system of linear equations

$$\sum_{t=1}^T (\mathbf{H}(\mathbf{x}_s, \mathbf{x}_t) + \lambda \delta_{st}) \mathbf{c}_t = \mathbf{y}_s, \quad \forall s \in \mathbb{N}_n , \quad (3)$$

where $\delta_{st} = 1$ if $s = t$ and is zero otherwise.

2.1 Granger-causality Analysis

To study the dynamical relationships in time series processes, Granger [11] proposed a practical definition of causality based on the accuracy of least-squares predictor functions. In brief, for two time series processes \mathbf{y} and \mathbf{z} , \mathbf{y} is said to Granger-cause \mathbf{z} ($\mathbf{y} \rightarrow \mathbf{z}$) if given all the other relevant information we can predict the future of \mathbf{z} better (in the mean-square-error sense) using the history of \mathbf{y} than without it.

Though the concept seems rather straightforward, there are (at least) three points worth considering. First, the notion is purely technical based on the predictive accuracy of functions with differing input sets; it does *not* seek to understand the underlying forces driving the relationships. Second, in practice the conditioning set of information needs to be reduced to all the *available* information instead of all the *relevant* information. Third, it only considers relationships between pairs of (sub-)processes and not the interactions amongst a set of series.

Eichler [9] extended the concept to multivariate analysis through graphical models. The discussion in the paper focuses on the notion of Granger non-causality rather than causality and describes the specific Markov properties (conditional non-causality) encoded in the graphs of Granger-causal relationships. In this sense, the absence of a variable in a set of inputs is more informative of the Granger (non-)causality than its presence. In result, graphical Granger methods are typically based on (structured) sparse modelling [4].

3 Function Space and Kernel Specification

The function space \mathcal{F} within which \mathbf{f} is learned is fully determined by the reproducing kernel \mathbf{H} . Its specification is therefore critical for achieving the two objectives for the function \mathbf{f} defined in Sect. 2. We focus on the class of matrix-valued kernels decomposable into the product of input kernels, capturing the similarities in the inputs, and output kernels, encoding the relationships between the outputs.

To analyse the dynamical dependencies between the series, we need to be able to discern within the inputs of the learned function \mathbf{f} the individual scalar series. Therefore we partition the elements of the input vectors according to the source scalar time series. In result, instead of a single kernel operating over the full vectors, we work with multiple partition-kernels, each of them operating over a single input series. We further propose to learn the partition-kernels by combining the MKL techniques with output kernel learning within the cone of positive semi-definite matrices.

More formally, the kernel we propose to use is constructed as a sum of kernels $\mathbf{H} = \sum_j^m \mathbf{H}^{(j)}$, where m is the number of the individual scalar-valued series in the multivariate process (dimensionality of the output space \mathcal{Y}). Each $\mathbf{H}^{(j)} : \mathcal{X}^{(j)} \times \mathcal{X}^{(j)} \rightarrow \mathbb{R}^{m \times m}$ is a matrix-valued kernel that determines its own RKHS of vector-valued functions. The domains $\mathcal{X}^{(j)} \subseteq \mathbb{R}^p$ are sets of vectors constructed by selecting from the inputs \mathbf{x} only the p coordinates $i^{(j)} \in \mathbb{N}_{mp}$ that correspond to the past of a single scalar time series j .

$$\mathcal{X}^{(j)} = \{\mathbf{x}^{(j)} : x_i^{(j)} = x_{i^{(j)}} \forall i, \mathbf{x} \in \mathcal{X}\}, \quad \cup_j \mathcal{X}^{(j)} = \mathcal{X}$$

Further, instead of choosing the individual matrix-valued functions $\mathbf{H}^{(j)}$, we propose to learn them. We construct each $\mathbf{H}^{(j)}$ again as a sum of kernels $\mathbf{H}^{(j)} = \sum_j^{s_j} \mathbf{H}^{(ji)}$ of possibly uneven number of summands s_j of matrix-valued kernels $\mathbf{H}^{(ji)} : \mathcal{X}^{(j)} \times \mathcal{X}^{(j)} \rightarrow \mathbb{R}^{m \times m}$. For this lowest level $\mathbf{H}^{(ji)}$ we focus on the family of decomposable kernels $\mathbf{H}^{(ji)} = k^{(ji)} \mathbf{L}^{(ji)}$. Here, the input kernels $k^{(ji)} : \mathcal{X}^{(j)} \times \mathcal{X}^{(j)} \rightarrow \mathbb{R}$ capturing the similarity between the inputs are fixed in advance from a dictionary of valid scalar-valued kernels. The set $\mathcal{L} = \{\mathbf{L}^{(ji)} : j = \mathbb{N}_m, i = \mathbb{N}_{s_j}, \sum_j^m s_j = l\}$ of output kernels $\mathbf{L}^{(ji)} : \mathcal{Y} \rightarrow \mathcal{Y}$ encoding the relations between the outputs is learned within the cone of symmetric positive semidefinite matrices \mathbb{S}_+^m .

$$\mathbf{H} = \sum_{j=1}^m \mathbf{H}^{(j)} = \sum_{j=1}^m \sum_{i=1}^{s_j} \mathbf{H}^{(ji)} = \sum_{j=1}^m \sum_{i=1}^{s_j} k^{(ji)} \mathbf{L}^{(ji)} \quad (4)$$

3.1 Kernel Learning and Function Estimation

Learning all the output kernels $\mathbf{L}^{(ji)}$ as full PSD matrices implies learning more than m^3 parameters. To improve the generalization capability, we reduce the complexity of the problem drastically by restricting the search space for \mathbf{L} 's to

PSD diagonal matrices \mathbb{D}_+^m . This essentially corresponds to the assumption of no contemporaneous relationships between the series. We return to this point in Sect. 5.

As explained in Sect. 2.1, Granger (non-)causality learning typically searches for sparse models. We bring this into our methods by imposing a further sparsity inducing regularizer $Q : (\mathbb{R}^{m \times m})^l \rightarrow \mathbb{R}$ on the set of the output kernels \mathcal{L} . We motivate and elaborate suitable forms of Q in Sect. 3.2.

The joint learning of the kernels and the function can now be formulated as the problem of finding the minimising solution $\mathbf{f} \in \mathcal{F}$ and $\mathbf{L}'s \in \mathbb{D}_+^m$ of the regularised functional

$$J(\mathbf{f}, \mathcal{L}) := R(\mathbf{f}) + \tau Q(\mathcal{L}), \quad \tau > 0, \quad (5)$$

where $R(\mathbf{f})$ is the regularised risk from (1). By calling on the properties of the RKHS, we reformulate this as a finite dimensional problem that can be addressed by conventional finite-dimensional optimisation approaches. We introduce the gram matrices $\mathbf{K}^{(ji)} \in \mathbb{S}_+^n$ such that $K_{ts}^{(ji)} = k^{(ji)}(\mathbf{x}_t^{(j)}, \mathbf{x}_s^{(j)})$ for all $t, s \in \mathbb{N}_n$, the output data matrix $\mathbf{Y} \in \mathbb{R}^{n \times m}$ such that $\mathbf{Y} = (\mathbf{y}_1, \dots, \mathbf{y}_n)^T$, and the coefficient matrix $\mathbf{C} \in \mathbb{R}^{n \times m}$ such that $\mathbf{C} = (\mathbf{c}_1, \dots, \mathbf{c}_n)^T$.

Using these and (2) it is easy to show that the minimisation of the regularised risk $R(\mathbf{f})$ in (1) with respect to $\mathbf{f} \in \mathcal{F}$ is equivalent to the minimisation with respect to $\mathbf{C} \in \mathbb{R}^{n \times m}$ of the objective

$$\tilde{R}(\mathbf{C}) := \|\mathbf{Y} - \sum_{ji} \mathbf{K}^{(ji)} \mathbf{C} \mathbf{L}^{(ji)}\|_F^2 + \lambda \sum_{ji} \langle \mathbf{C}^T \mathbf{K}^{(ji)} \mathbf{C}, \mathbf{L}^{(ji)} \rangle_F. \quad (6)$$

The finite dimensional equivalent of (5) is thus the joint minimisation of

$$\tilde{J}(\mathbf{C}, \mathcal{L}) := \tilde{R}(\mathbf{C}, \mathcal{L}) + \tau Q(\mathcal{L}). \quad (7)$$

3.2 Sparse Regularization

The construction of the kernel \mathbf{H} and the function space \mathcal{F} described in Sect. 3 imposes on the function \mathbf{f} the necessary structure that allows the Granger-causality analysis (as per our 2nd objective set-out in Sect. 2). As explained in Sect. 2.1, the other ingredient we need to identify the Granger non-causalities is sparsity within the structure of the learned function.

In our methods, the sparsity is introduced by the regularizer Q . By construction of the function space, we can examine the elements of the output kernels $\mathbf{L}^{(ij)}$ (their diagonals) to make statements about the Granger non-causality. We say the j -th scalar time series is non-causal for the s series (given all the remaining series in the process) if $\mathbf{L}_{ss}^{(ji)} = 0$ for all $i \in \mathbb{N}_{s_j}$.

Essentially, any of the numerous regularizers that exist for sparse or structured sparse learning [3] could be used as Q , possibly based on some prior knowledge about the underlying dependencies within the time-series process.

We elaborate here two cases that do not assume any special structure in the dependencies as the base scenarios. The first is the entry-wise ℓ_1 norm across all the output kernels so that

$$Q_1(\mathcal{L}) = \sum_{ji} \|\mathbf{L}^{(ji)}\|_1 = \sum_{ji} \sum_s^m |L_{ss}^{(ji)}| . \quad (8)$$

The second is the ℓ_1/ℓ_2 grouped norm

$$Q_{1/2}(\mathcal{L}) = \sum_{js} \sqrt{\sum_i (L_{ss}^{(ji)})^2} . \quad (9)$$

After developing the learning strategy for these in Sect(s). 4.1 and 4.2, we provide some more intuition of their effects on the models and link to some other known graphical Granger techniques in Sect. 5.

4 Learning Strategy

First of all, we simplify the final formulation of the problem (7) in Sect. 3.1. Rather than working with a set of diagonal matrices $\mathbf{L}^{(ji)}$, we merge the diagonals into a single matrix \mathbf{A} . We then re-formulate the problem with respect to this single matrix in place of the set and show how this reformulation can be suitably decomposed into smaller independent sub-problems.

We develop fit-to-purpose approaches for our two regularisers in Sect(s). 4.1 and 4.2. The first - based on the decomposition of the kernel matrices into the corresponding empirical features and on the variational formulation of norms [3] - shows the equivalence of the problem with group lasso [22, 23]. The second proposes a simple alternating minimisation algorithm to obtain the two sets of parameters.

We introduce the non-negative matrix $\mathbf{A} \in \mathbb{R}_+^{l \times m}$ such that

$$\mathbf{A} = (\text{diag}(\mathbf{L}^{11}), \dots, \text{diag}(\mathbf{L}^{ms_m}))^T \quad (10)$$

(each row in \mathbf{A} corresponds to the diagonal of one output kernel; if $s_j = 1$ for all j we have $A_{js} = \mathbf{L}_{ss}^{(j1)}$). Using this change of variable, the optimisation problem (7) can be written equivalently as

$$\begin{aligned} & \arg \min_{\mathbf{A}, \mathbf{C}} \ddot{J}(\mathbf{C}, \mathbf{A}) \\ \ddot{J}(\mathbf{C}, \mathbf{A}) & := \ddot{R}(\mathbf{C}, \mathbf{A}) + \tau \ddot{Q}(\mathbf{A}) \quad , \end{aligned} \quad (11)$$

where

$$\begin{aligned} \ddot{R}(\mathbf{C}, \mathbf{A}) &= \sum_s^m \left(\|\mathbf{Y}_{:s} - \sum_{ji} A_{(ji)s} \mathbf{K}^{ij} \mathbf{C}_{:s}\|_2^2 + \lambda \sum_{ji} A_{(ji)s} \mathbf{C}_{:s}^T \mathbf{K}^{(ji)} \mathbf{C}_{:s} \right) \\ &= \sum_s^m \left(\ddot{R}_s(\mathbf{C}_{:s}, \mathbf{A}_{:s}) \right) \quad , \end{aligned} \quad (12)$$

and $\ddot{Q}(\mathbf{A})$ is the equivalent of $Q(\mathcal{L})$ so that

$$\ddot{Q}_1(\mathbf{A}) = \|\mathbf{A}\|_1 = \sum_{rs} |A_{rs}| \quad (13)$$

and

$$\ddot{Q}_{1/2}(\mathbf{A}) = \sum_{js} \sqrt{\sum_i (A_{(ji)s})^2} \quad (14)$$

In equations (12) and (14) we somewhat abuse the notation by using $\sum_{ji} A_{(ji)s}$ to indicate the sum across the rows of the matrix \mathbf{A} .

From (12)-(14) we observe that, with both of our regularizers, problem (11) is conveniently separable along s into m smaller independent problems, one per scalar output series. These can be efficiently solved in parallel, which makes our method scalable to very large multivariate systems. The final complexity depends on the choice of the regulariser Q and the appropriate algorithm. The overhead cost can be significantly reduced by precalculating the gram matrices $\mathbf{K}^{(ij)}$ in a single preprocessing step and sharing these in between the m parallel tasks.

4.1 Learning with ℓ_1 Norm

To unclutter notation we replace the bracketed double superscripts (ij) by a single superscript $d = 1, \dots, l$. We also drop the regularization parameter τ (fix it to $\tau = 1$) as it is easy to show that any other value can be absorbed into the rescaling of λ and the \mathbf{C} and \mathbf{A} matrices. For each of the s parallel tasks we indicate $\mathbf{A}_{:,s} = \mathbf{a}$, $\mathbf{C}_{:,s} = \mathbf{c}$ and $\mathbf{Y}_{:,s} = \mathbf{y}$ so that the individual problems are the minimisations with respect to $\mathbf{a} \in \mathbb{R}_+^l$ and $\mathbf{c} \in \mathbb{R}^n$ of

$$P(\mathbf{c}, \mathbf{a}) := \|\mathbf{y} - \sum_d a_d \mathbf{K}^d \mathbf{c}\|_2^2 + \lambda \sum_{ji} a_d \mathbf{c}^T \mathbf{K}^d \mathbf{c} + \sum_d a_d . \quad (15)$$

We decompose (for example by eigendecomposition) each of the gram matrices as $\mathbf{K}^d = \mathbf{\Phi}^d (\mathbf{\Phi}^d)^T$, where $\mathbf{\Phi}^d \in \mathbb{R}^{n \times n}$ is the matrix of the empirical features, and we introduce the variables $\mathbf{z}^d = a_d (\mathbf{\Phi}^d)^T \mathbf{c} \in \mathbb{R}^n$ and the set $\mathcal{Z} = \{\mathbf{z}^d : \mathbf{z}^d \in \mathbb{R}^n, d \in \mathbb{N}_l\}$. Using these we rewrite³ equation (15)

$$\tilde{P}(\mathcal{Z}, \mathbf{a}) := \|\mathbf{y} - \sum_d \mathbf{\Phi}^d \mathbf{z}^d\|_2^2 + \sum_d \left(\frac{\lambda \|\mathbf{z}^d\|_2^2}{a_d} + a_d \right) . \quad (16)$$

We first find the closed form of the minimising solution for \mathbf{a} as $a_d = \sqrt{\lambda} \|\mathbf{z}^d\|_2$ for all d . Plugging this back to (16) we obtain

$$\min_{\mathbf{a}} \tilde{P}(\mathcal{Z}, \mathbf{a}) = \|\mathbf{y} - \sum_d \mathbf{\Phi}^d \mathbf{z}^d\|_2^2 + 2\sqrt{\lambda} \sum_d \|\mathbf{z}^d\|_2 . \quad (17)$$

³ We extend the function $x^2/y : \mathbb{R} \times \mathbb{R}_+ \rightarrow \mathbb{R}_+$ to the point $(0, 0)$ by taking the convention $0/0 = 0$.

Seen as a minimisation with respect to the set \mathcal{Z} this is the classical group-lasso formulation with the empirical features Φ^d as inputs. Accordingly, it can be solved by any standard method for group-lasso problems such as the proximal gradient descent method, e.g. [3], which we employ in our experiments. After solving for \mathcal{Z} we can directly recover \mathbf{a} from the above minimising identity and then obtain the parameters \mathbf{c} from the set of linear equations

$$\left(\sum_d a_d \mathbf{K}^d + \lambda \mathbf{I}_n\right) \mathbf{c} = \mathbf{y} . \quad (18)$$

The algorithm outlined above takes advantage of the convex group-lasso reformulation (17) and has the standard convergence and complexity properties of proximal gradient descent. The empirical features Φ^d can be pre-calculated and shared amongst the m tasks to reduce the overhead cost.

4.2 Learning with ℓ_1/ℓ_2 Norm

For the ℓ_1/ℓ_2 regularization, we need to return to the double indexation (ji) to make clear how the groups are created. As above, for each of the s parallel tasks we use the vectors \mathbf{a} , \mathbf{c} and \mathbf{y} . However, for vector \mathbf{a} we will keep the (ji) notation for its elements. The individual problems are the minimisations with respect to $\mathbf{a} \in \mathbb{R}_+^l$ and $\mathbf{c} \in \mathbb{R}^n$ of

$$P(\mathbf{c}, \mathbf{a}) := \|\mathbf{y} - \sum_{ji} a_{(ji)} \mathbf{K}^{(ji)} \mathbf{c}\|_2^2 + \lambda \sum_{ji} a_{(ji)} \mathbf{c}^T \mathbf{K}^{(ji)} \mathbf{c} + \sum_j \sqrt{\sum_i a_{(ji)}^2} \quad (19)$$

We propose to use the alternating minimisation with a proximal gradient step. At each iteration, we alternatively solve for \mathbf{c} and \mathbf{a} . For fixed \mathbf{a} we obtain \mathbf{c} from the set of linear equations (18). With fixed \mathbf{c} , problem (19) is a group lasso for \mathbf{a} with groups defined by the sub-index j within the double (ji) indexation of the elements of \mathbf{a} . Here, the proximal gradient step takes place to move along the descend direction for \mathbf{a} . Though convex in \mathbf{a} and \mathbf{c} individually, the problem (19) is jointly non-convex and therefore can converge to local minima.

5 Interpretation and Crossovers

To help the understanding of the inner workings of our methods and especially the effects of the two regularizers, we discuss here the crossovers to other existing methods for MKL and Granger modelling.

ℓ_1 Norm The link to group-lasso demonstrated in Sect. 4.1 is not in itself too surprising. The formulation in (15) can be recognised as a sparse multiple kernel learning problem which has been previously shown to relate to group-lasso (eg. [2], [21]). We derive this link in Sect. 4.1 using the empirical feature representation to i) provide better intuition for the structure of the learned function $\hat{\mathbf{f}}$, ii) develop an efficient algorithm for solving problem (15).

The re-formulation in terms of the empirical features Φ^d creates an intuitive bridge to the classical linear models. Each Φ^d can be seen as a matrix of features generated from a subset $\mathcal{X}^{(j)}$ of the input coordinates relating to the past of a single scalar time series j . The group-lasso regularizer in equation (17) has a sparsifying effect at the level of these subsets zeroing out (or not) the whole groups of parameters \mathbf{z}^d . In the context of linear methods, this approach is known as the grouped graphical Granger modelling [16].

Within the non-linear approaches to time series modelling, Sindhvani et al. [19] recently derived a similar formulation. There the authors followed a strategy of multiple kernel learning from a dictionary of input kernels combined with a single learned output kernel (as opposed to our multiple output kernels). They obtain their IKL model, which is in its final formulation equivalent to problem (15), by fixing the output kernel to identity.

Though we initially formulate our problem quite differently, the diagonal constraint we impose on the output kernels essentially prevents the modelling of any contemporaneous relationships between the series (as does the identity output kernel matrix in IKL). What remains in our methods are the diagonal elements, which are non-constant and sparse, and which can be interpreted as the weights of the input kernels in the standard MKL setting.

ℓ_1/ℓ_2 Norm The more complex ℓ_1/ℓ_2 regularisation discussed in Sect. 4.2 is to the best of our knowledge novel in the context of multiple kernel learning. It has again a strong motivation and clear interpretation in terms of the graphical Granger modelling. The norm has a sparsifying effect not only at the level of the individual kernels but at the level of the groups of kernels operating over the same input partitions $\mathcal{X}^{(j)}$. In this respect our move from the ℓ_1 to the ℓ_1/ℓ_2 norm has a parallel in the same move in linear graphical Granger techniques. The ℓ_1 norm Lasso-Granger method [1] imposes the sparsity on the individual elements of the parameter matrices in a linear model, while the ℓ_1/ℓ_2 of the grouped-Lasso-Granger [16] works with groups of the corresponding parameters of a single input series across the multiple lags p .

6 Experiments

To document the performance of our method, we have conducted a set of experiments on real and synthetic datasets. In these we simulate real-life forecasting exercise by splitting the data into a training and a hold-out set which is unseen by the algorithm when learning the function $\hat{\mathbf{f}}$ and is only used for the final performance evaluation.

We compare our methods with the output kernel ℓ_1 regularization (NVARL1) and ℓ_1/ℓ_1 (NVARL12) with simple baselines (which nevertheless are often hard to beat in practical time series forecasting) as well as with the state-of-the-art techniques for forecasting and Granger modelling. Namely, we compare with simple mean and univariate linear autoregressive models (LAR), multivariate linear vector autoregressive model with ℓ_2 penalty (LVARL2), the group-lasso

Granger method [16] (LVARL1), and a sparse MKL without the $\mathcal{X}^{(j)}$ input partitioning (NVAR). Of these, the last two are the most relevant competitors. LVARL1, similarly to our methods, aims at recovering the Granger structure but is strongly constrained to linear modelling only. NVAR has no capability to capture the Granger relationships but, due to the lack of structural constraints, it is the most flexible of all the models.

We evaluate our results with respect to the two objectives for the function $\hat{\mathbf{f}}$ defined in Sect. 2. We measure the accuracy of the one-step ahead forecasts by the mean square error (MSE) for the whole multivariate process averaged over 500 hold-out points. The structural objective allowing the analysis of dependencies between the sub-processes is wired into the method itself (see Sect(s). 3 and 2.1) and is therefore satisfied by construction. We produce adjacency matrices of the graphs of the learned dependencies, compare these with the ones produced by the linear Granger methods and comment on the observed results.

6.1 Technical Considerations

For each experiment we preprocessed the data by removing the training sample mean and rescaling with the training sample standard deviation. We fix the number of kernels for each input partition to six ($s_j = 6$ for all j) and use the same kernel functions for all experiments: a linear, 2nd order and 3rd polynomial, and Gaussian kernels with width 0.5, 1 and 2. We normalise the kernels so that the training Gram matrices have trace equal to the size of the training sample.

We search for the hyper-parameter λ by a 5-fold cross-validation within a 15-long logarithmic grid $\lambda \in \{10^{-3}, \dots, 10^4\} \sqrt{nl}$, where n is the training sample size and l is the number of kernels or groups (depending on the method). In each grid search, we use the previous parameter values as warm starts. We do not perform an exhaustive search for the optimal lag for each of the scalar input series by some of the classical testing procedures (based on AIC, BIC etc.). We instead fix it to $p = 5$ for all series in all experiments and rely on the regularization to control any excess complexity.

We implemented our own tools for all the tested methods based on variations of proximal gradient descent with ISTA line search [5]. The full Matlab code is available at <https://bitbucket.org/dmmlgeneva/nonlinear-granger>

6.2 Synthetic Experiments

We have simulated data from a five dimensional non-Gaussian time-series process generated through a linear filter of a 5-dimensional i.i.d. exponential white noise \mathbf{e}_t with identity covariance matrix (re-centered to zero and re-scaled to unit variance). The matrix $\Psi = [0.7, 1.3, 0, 0, 0; 0, 0.6, -1.5, 0, 0; 0, -1.2, 1.46, 0, 0; 0, 0, 0.6, 1.4; 0, 0, 0, 1.3, -0.5]$ in the filter $\mathbf{y}_t = \mathbf{e}_t + \Psi \mathbf{e}_{t-1}$ is such that the process consists of two independent internally interrelated sub-processes, one composed of the first 3 scalar series, the other of the remaining two series. This structural information, though known to us, is unknown to the learning methods (not considered in the learning process).

We list in Table 1 the predictive performance of the tested methods in terms of the average hold-out MSE based on training samples of varying size. Our methods clearly outperform all the linear models. The functionally strongly constrained linear LVARL1 performs roughly on par with our methods for the small sample sizes. But for larger sample sizes, the higher flexibility of the function space in our methods yields significantly more accurate forecasts (as much as 10% MSE improvement).

Table 1: Synthetic experiments: MSE _(std) for 1-step ahead forecasts (hold-out sample average)

Train size	300	700	1000
Mean	0.925 <small>(0.047)</small>	0.923 <small>(0.047)</small>	0.923 <small>(0.047)</small>
LAR	0.890 <small>(0.045)</small>	0.890 <small>(0.044)</small>	0.890 <small>(0.044)</small>
LVAR	0.894 <small>(0.045)</small>	0.836 <small>(0.041)</small>	0.763 <small>(0.035)</small>
LVARL1	0.787 <small>(0.037)</small>	0.737 <small>(0.031)</small>	0.722 <small>(0.030)</small>
NVAR	0.835 <small>(0.041)</small>	0.735 <small>(0.032)</small>	0.719 <small>(0.030)</small>
NVARL1	<i>0.754</i> <small>(0.034)</small>	0.706 <small>(0.030)</small>	0.679 <small>(0.028)</small>
NVARL12	0.808 <small>(0.040)</small>	0.710 <small>(0.031)</small>	0.684 <small>(0.029)</small>
Train size	1500	2000	3000
Mean	0.923 <small>(0.047)</small>	0.922 <small>(0.047)</small>	0.922 <small>(0.047)</small>
LAR	0.888 <small>(0.045)</small>	0.889 <small>(0.045)</small>	0.888 <small>(0.045)</small>
LVAR	0.751 <small>(0.034)</small>	0.741 <small>(0.033)</small>	0.687 <small>(0.028)</small>
LVARL1	0.710 <small>(0.029)</small>	0.701 <small>(0.028)</small>	0.693 <small>(0.028)</small>
NVAR	0.699 <small>(0.028)</small>	0.682 <small>(0.027)</small>	0.662 <small>(0.026)</small>
NVARL1	<i>0.654</i> <small>(0.026)</small>	<i>0.640</i> <small>(0.025)</small>	0.626 <small>(0.025)</small>
NVARL12	0.659 <small>(0.027)</small>	0.685 <small>(0.028)</small>	0.657 <small>(0.027)</small>

In brackets is the average standard deviation (std) of the MSEs. Results for NVARL1 and NVARL12 in **bold** are significantly better than **all** the linear competitors, in *italics* are significantly better than the non-linear NVAR (using one-sided paired-sample t-test at 10% significance level).

The structural constraints in our methods also help the performance when competing with the unstructured NVAR method, which has mostly less accurate forecasts. At the same time, as illustrated in Fig. 1, our methods are able to correctly recover the Granger-causality structure (splitting the process into the two independent subprocesses by the zero off-diagonal blocks), which NVAR by construction cannot.

6.3 Real Data Experiments

We use data on water physical discharge publicly available from the website of the Water Services of the US geological survey (<http://www.usgs.gov/>). Our dataset consists of 9 time series of daily rates of year-on-year growth at measurement sites along the streams of Connecticut and Columbia rivers.

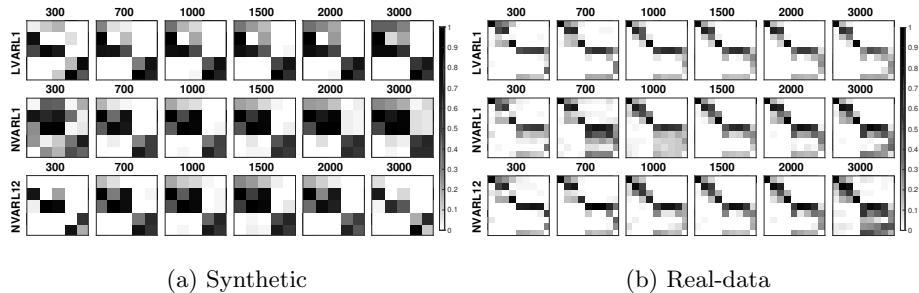


Fig. 1: Schematics of the learned adjacency matrices of Granger-causality graphs for the three sparse learning methods across varying training sample size. A scalar time series y_i does not Granger-cause series y_j (given all the other series) if the element e_{ij} in the adjacency matrix is zero (white). The displayed adjacency matrices were derived from the learned matrices \mathbf{A} by summing the respective elements across individual kernels. The values are rescaled so that the largest element in each matrix is equal to 1 (black).

The prediction accuracy of the tested methods is listed in Table 2. Our non-linear methods perform on par with the state-of-the-art linear models. This on one hand suggests that for the analysed dataset the linear modelling seems sufficient. On the other hand, it confirms that our methods, which in general have the ability to learn more complex relationships by living in a richer functional space, are well behaved and can capture simpler dependencies as well. The structure encoded into our methods, however, benefits the learning since the unstructured NVAR tends to perform less accurately.

The learned dynamical dependence structure of the time series is depicted in Fig. 1. In the dataset (and the adjacency matrices), the first 4 series are the Connecticut measurement sites starting from the one highest up the stream and moving down to the mouth of the river. The next 5 are the Columbia measurement sites ordered in the same manner.

From inspecting the learned adjacency matrices, we observe that all the sparse methods recover similar Granger-causal structures. Since we do not know the ground truth in this case, we can only speculate about the accuracy of the structure recovery. Nevertheless, it seems plausible that there is little dynamical cross-dependency between the Connecticut and Columbia measurements as the learned graphs suggest (the two rivers are at the East and West extremes of the US).

7 Conclusions

We have developed a new method for forecasting and Granger-causality modelling in multivariate time series that does not rely on prior assumptions about the shape of the dynamical dependencies (other than being sparse). The method

Table 2: Real-data experiments: MSE _(std) for 1-step ahead forecasts (hold-out sample average)

Train size	300	700	1000
Mean	0.780 (0.053)	0.795 (0.054)	0.483 (0.026)
LAR	0.330 (0.023)	0.340 (0.024)	0.152 (0.013)
LVARL2	0.302 (0.021)	0.311 (0.022)	0.140 (0.012)
LVARL1	0.310 (0.022)	0.310 (0.023)	0.140 (0.012)
NVAR	0.328 (0.023)	0.316 (0.023)	0.148 (0.012)
NVARL1	0.308 (0.023)	0.317 (0.024)	0.140 (0.012)
NVARL12	0.321 (0.023)	0.322 (0.024)	0.141 (0.012)
Train size	1500	2000	3000
Mean	0.504 (0.03)	0.464 (0.027)	0.475 (0.017)
LAR	0.181 (0.015)	0.179 (0.013)	0.187 (0.008)
LVARL2	0.167 (0.014)	0.164 (0.013)	0.170 (0.007)
LVARL1	0.165 (0.014)	0.163 (0.013)	0.170 (0.008)
NVAR	0.169 (0.014)	0.166 (0.012)	0.173 (0.007)
NVARL1	0.164 (0.014)	0.161 (0.013)	0.167 (0.007)
NVARL12	0.162 (0.014)	0.160 (0.012)	0.166 (0.007)

In brackets is the average standard deviation (std) of the MSEs.

is based on learning a combination of multiple operator-valued kernels in which the multiple output kernels are learned as sparse diagonal matrices. We have documented on experiments that our method outperforms linear competitors in the presence of strong non-linearities and is able to correctly recover the Granger-causality structure which non-structured kernel methods cannot do.

Acknowledgements

This work was partially supported by the research projects HSTS (ISNET) and RAWFIE #645220 (H2020). We thank Francesco Dinuzzo for helping to form the initial ideas behind this work through fruitful discussions while visiting in IBM Research, Dublin.

Bibliography

- [1] Arnold, A., Liu, Y., Abe, N.: Temporal causal modeling with graphical granger methods. Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '07 (2007)
- [2] Bach, F.: Consistency of the group lasso and multiple kernel learning. The Journal of Machine Learning Research (2008)
- [3] Bach, F., Jenatton, R., Mairal, J., Obozinski, G.: Optimization with sparsity-inducing penalties. Foundations and Trends in Machine Learning (2012)
- [4] Bahadori, M., Liu, Y.: An Examination of Practical Granger Causality Inference. SIAM Conference on Data Mining (2013)
- [5] Beck, A., Teboulle, M.: Gradient-based algorithms with applications to signal recovery. Convex Optimization in Signal Processing and Communications (2009)
- [6] Brockwell, P.J., Davis, R.A.: Time Series: Theory and Methods. Springer Science+Business Media, LLC, 2nd edn. (2006)
- [7] Caponnetto, A., Micchelli, C.A., Pontil, M., Ying, Y.: Universal Multi-Task Kernels. Machine Learning Research (2008)
- [8] Dinuzzo, F., Ong, C.: Learning output kernels with block coordinate descent. In: International Conference on Machine Learning (ICML) (2011)
- [9] Eichler, M.: Graphical modelling of multivariate time series. Probability Theory and Related Fields (2012)
- [10] Franz, M.O., Schölkopf, B.: A unifying view of wiener and volterra theory and polynomial kernel regression. Neural computation (2006)
- [11] Granger, C.W.J.: Investigating Causal Relations by Econometric Models and Cross-spectral Methods. Econometrica: Journal of the Econometric Society (1969)
- [12] Jawanpuria, P., Lapin, M., Hein, M., Schiele, B.: Efficient Output Kernel Learning for Multiple Tasks. In: NIPS (2015)
- [13] Kadri, H., Rakotomamonjy, A., Bach, F., Preux, P.: Multiple Operator-valued Kernel Learning. In: NIPS (2012)
- [14] Lanckriet, G.G.R., Cristianini, N., Bartlett, P., Ghaoui, L.E., Jordan, M.I.: Learning the kernel matrix with semidefinite programming. Journal of Machine Learning Research (2004)
- [15] Lim, N., D'Alché-Buc, F., Auliac, C., Michailidis, G.: Operator-valued Kernel-based Vector Autoregressive Models for Network Inference. Machine Learning (2014)
- [16] Lozano, A.C., Abe, N., Liu, Y., Rosset, S.: Grouped graphical Granger modeling for gene expression regulatory networks discovery. Bioinformatics (Oxford, England) (2009)
- [17] Micchelli, C.A., Pontil, M.: On learning vector-valued functions. Neural computation (2005)

- [18] Pillonetto, G., Dinuzzo, F., Chen, T., De Nicolao, G., Ljung, L.: Kernel methods in system identification, machine learning and function estimation: A survey. *Automatica* (2014)
- [19] Sindhvani, V., Minh, H.Q., Lozano, A.: Scalable Matrix-valued Kernel Learning for High-dimensional Nonlinear Multivariate Regression and Granger Causality. In: *UAI* (2013)
- [20] Turkman, K.F., Scotto, M.G., de Zea Bermudez, P.: *Non-Linear Time Series*. Springer (2014)
- [21] Xu, Z., Jin, R., Yang, H., King, I., Lyu, M.R.: Simple and efficient multiple kernel learning by group lasso. *International Conference on Machine Learning (ICML)* (2010)
- [22] Yuan, M., Lin, Y.: Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* (2006)
- [23] Zhao, P., Rocha, G.: Grouped and hierarchical model selection through composite absolute penalties (2006)