Learning Graphical Models for Stationary Time Series

Francis R. Bach
Computer Science Division
University of California
Berkeley, CA 94114, USA
fbach@cs.berkeley.edu

Michael I. Jordan
Computer Science Division
and Department of Statistics
University of California
Berkeley, CA 94114, USA
jordan@cs.berkeley.edu

September 29, 2003

Technical Report 650
Department of Statistics
University of California, Berkeley

Abstract

Probabilistic graphical models can be extended to time series by considering probabilistic dependencies between entire time series. For stationary Gaussian time series, the graphical model semantics can be expressed naturally in the frequency domain, leading to interesting families of structured time series models that are complementary to families defined in the time domain. In this paper, we present an algorithm to learn the structure from data for directed graphical models for stationary Gaussian time series. We describe an algorithm for efficient forecasting for stationary Gaussian time series whose spectral densities factorize in a graphical model. We also explore the relationships between graphical model structure and sparsity, comparing and contrasting the notions of sparsity in the time domain and the frequency domain. Finally, we show how to make use of Mercer kernels in this setting, allowing our ideas to be extended to nonlinear models.

1 Introduction

Time series arise in many problems in signal processing, bioinformatics computer vision and machine learning. In the statistical modeling of time series, the assumption of stationarity makes possible the use of tools from spectral analysis [2]. Much of the algorithmic research effort in this area has dealt with making such tools scalable with the respect to the number of observed samples, for example via algorithms that exploit the Fast Fourier Transform, or forecasting algorithms such as the Durbin-Levinson algorithm [5].

Domains with large number of variables—in which Markovian or general graphical models have excelled—have not attracted the same attention in the time series field. Graphical models [23, 20] provide a general framework for defining probabilistic models over large numbers of variables, by building global models out of local interaction models. Numerous special cases are familiar in signal processing, including the Kalman filter, hidden Markov models and factor analysis. In addition, these models come equipped with standard, numerically-efficient learning and inference algorithms, algorithms that have had applications beyond signal processing and machine learning, such as in error-control coding [26].

Graphical models for time series are generally defined in the *time domain*. That is, they define a transition probability distribution on a set of state variables, conditioning on values of these variables at previous time steps. Such models, which are often referred to as *dynamic Bayesian networks*, have had significant applications in areas such as bioinformatics and speech processing [12, 24]. In this paper we consider an extension of the basic notion of graphical model which considers dependencies between entire time series [3, 11]. As we show, this extension can be naturally expressed in the frequency domain, making use of the *spectral representations* for stationary Gaussian time series. In this paper, we present an algorithm to learn the structure of directed graphical models for spectral representations of time series from data. The algorithm, presented in Section 5, is a direct extension of algorithms for learning directed graphical models for Gaussian data [14, 22].

We also discuss the problem of forecasting, the problem of predicting the future given the past. For stationary Gaussian time series, algorithms for forecasting can be defined in either the time or frequency domain. In Section 4 we present a novel algorithm for efficient forecasting with graphical models. While classical algorithms such as the Durbin-Levinson [5] algorithm incur a cubic time complexity, our new algorithm, by making use of the structure of the graphical model, has a quadratic complexity (in the number of variables). This new algorithm can also be used for factor analysis models for time series [4].

While our basic focus is on structured linear time series models, we also present an extension to nonlinear models in Section 6. In particular, we make use of our previous work in learning graphical models for hybrid data [1], enabling the fitting of augmented models that include nonlinearities or discrete variables. The principle of the algorithm is very simple: although variables may not be Gaussian, by mapping them into a high-dimensional feature space, they can be considered as Gaussian for the purpose of model selection. This mapping is made implicit and computationally efficient through the use of kernel methods [27].

The graphical models that we describe in this paper could have several possible applications, paralleling the many applications of graphical models for non-temporal data, such as feature selection for regression or classification, or sparse and statistically sound models for forecasting. In Section 7, we perform simulations on synthetic and real datasets, to illustrate the validity of our algorithm and compare them to other approaches for modeling stationary time series.

2 Stationary Gaussian processes

We consider a multiple time series $x(t)$, where for each $t \in \mathbb{Z}$, $x(t) = (x_1(t), \ldots, x_m(t))$ has m univariate real components[1]. Throughout this paper we assume that $x(t)$ is a zero-mean Gaussian process; that is, all finite marginals are jointly Gaussian. In addition, we assume that the process $x(t)$ is *stationary*: for Gaussian processes, $x(t)$ is stationary if and only if $Ex(t+h)x(t)^\top$ does not depend on t, or equivalently, all marginals are invariant by time translation. Given a stationary process, the *autocovariance function* is defined as the matrix-valued function $\Gamma(h)$ on $\mathbb{Z}$, defined as

$$\Gamma(h) = Ex(t+h)x(t)^\top = Ex(h)x(0)^\top.$$

For each $h \in \mathbb{Z}$, $\Gamma(h)$ is a symmetric $m \times m$ real matrix, and the function $h \mapsto \Gamma(h)$ is *nonnegative*, that is, for all sets of vectors $\alpha_i \in \mathbb{R}^m$, indexed by $i \in I$, we have $\sum_{i,j \in I} \alpha_i^\top \Gamma(i-j)\alpha_j \geqslant 0$. Essentially, the Gaussian stationarity assumption is equivalent to modeling the variables as jointly Gaussian with tied parameters; i.e., the covariance matrix of any successive variables $x(t), x(t+1), \ldots, x(t+h-1)$

[1]Note that the theory of stationary Gaussian processes and most of the results of this paper can be naturally extended to the complex case.

is *Toeplitz* by blocks:

$$T_h(\Gamma) = \begin{pmatrix} \Gamma(0) & \Gamma(-1) & \Gamma(-2) & \cdots & \Gamma(-h+1) \\ \Gamma(1) & \Gamma(0) & \Gamma(-1) & & \\ \Gamma(2) & \Gamma(1) & \Gamma(0) & & \vdots \\ \vdots & & & \ddots & \\ \Gamma(h-1) & & \cdots & & \Gamma(0) \end{pmatrix}. \tag{1}$$

2.1 Spectral density matrix

We assume that $\sum_{-\infty}^{\infty} ||\Gamma(h)|| < \infty$, so that the *spectral density matrix* $f(\omega)$ is well defined, as a matrix-valued function on $\mathbb{R}$:

$$f(\omega) = \frac{1}{2\pi} \sum_{h=-\infty}^{+\infty} \Gamma(h) e^{-ih\omega}.$$

For each ω, $f(\omega)$ is an $m \times m$ Hermitian matrix. In addition the function $\omega \mapsto f(\omega)$ is 2π-periodic. Also for real valued random variables, we have $f(-\omega) = f(\omega)^\top$, which implies that we only need to consider the spectral density matrix for $\omega \in [0, \pi]$.

Since the function $\Gamma(h)$ is nonnegative, the spectral density matrix is pointwise nonnegative, that is $\forall \omega \in [-\pi, \pi],\ \forall \alpha \in \mathbb{C}^m,\ \alpha^* f(\omega) \alpha \in \mathbb{R}^+$. Note that knowing the spectral density f is equivalent to knowing the autocovariance function Γ, since they form a Fourier transform pair. In particular, we have:

$$\Gamma(h) = \int_{-\pi}^{\pi} f(\omega) e^{ih\omega} d\omega. \tag{2}$$

The rate of decay of $\Gamma(h)$ as h grows is determined by the smoothness of the spectral density. In particular, when designing a spectral density, care must be taken regarding its smoothness so that the resulting autocovariance function has attractive decay properties (which themselves govern the complexity of prediction).

2.2 Spectral representation

Any Gaussian stationary time series with absolutely summable autocovariance function has a spectral representation of the following form [5]:

$$x(t) = \int_{-\pi}^{\pi} f(\omega) dZ(\omega), \tag{3}$$

where $Z(\omega)$ is a random process with orthogonal increments and such that for each ω, $EZ(\omega)Z(\omega)^* = f(\omega)$. In other words, $x(t)$ is a superposition of infinitely many independent random signals at different frequencies.

2.3 Autoregressive models

In this paper, we consider stationary autoregressive (AR) models. Stationary can be imposed either by assuming that the process is initialized according to the stationary distribution or that the process is causal and extends to negative infinity. An autoregressive model has the following formulation:

$$x(t) = \Phi_1 x(t-1) + \cdots + \Phi_p x(t-p) + z(t)$$

where the variables $z(t)$ are mutually independent, each with covariance matrix Σ, and also independent from $x(u), u < t$. Each Φ_i is an $m \times m$ matrix.

The parameters Φ_i and the order p of AR models can be efficiently estimated from data by using classical regression methods for parameter estimation and variable selection [5]. If sparsity is desired, one can attempt to find zeros in the covariance matrix Σ or its inverse, as well as in the regression weight matrices Φ_i, in a manner analogous to subset selection for linear regression [18]. In this paper we consider sparsity in the frequency domain, a notion that is complementary to sparsity in the time domain.

2.4 Finite sample

In this section, we briefly review methods for estimating the autocovariance function $\Gamma(h)$ and the spectral density matrix $f(\omega)$ from data $x(0), \ldots, x(T-1)$.

2.4.1 Sample autocovariance and periodogram

The *sample autocovariance function* is defined as

$$\hat{\Gamma}(h) = \frac{1}{T} \sum_{t=0}^{T-h-1} (x(t+h) - \bar{x})(x(t) - \bar{x})^\top$$

for $h \in [0, T-1]$, extended by symmetry for negative h and equal to zero for $|h|$ equal or greater than T (the vector $\bar{x} = \frac{1}{T}\sum_{t=1}^{T} x(t)$ is the sample mean of the data). The sample autocovariance function is consistent and asymptotically normal under weak assumptions [5].

The *periodogram* is defined as the Fourier transform of the sample autocovariance function. More precisely, let $d(0), \ldots, d(T-1)$ be the discrete Fourier transform of the data:

$$d(k) = \frac{1}{N} \sum_{t=0}^{T-1} x(t) e^{-ikt}.$$

At the frequencies $\omega_k = 2\pi k/T$, $\omega_k \in [-\pi, \pi]$, the periodogram is defined as $I(\omega_k) = d(k)d(k)^*$ and can readily be computed using m fast Fourier transforms (FFT). It is usually extended as a piecewise constant periodic function on $\mathbb{R}$.

The periodogram does not provide a consistent estimator of the spectral density and is a notoriously bad estimator of the spectral density. However when it is appropriately smoothed, it is a good estimator.

2.4.2 Smoothing the periodogram

The periodogram is smoothed by convolving it with a smoothing window $W_r(j)$, leading to the following estimator, at frequency ω_k:

$$\hat{f}^r(\omega_k) = \sum_{j=-\infty}^{\infty} W_r(j) I(\omega_{j+k}). \tag{4}$$

$W_r(j)$ is a smoothing window that is required to be symmetric, with finite support, and to sum to one. In our simulations, we used the window $W_r(j) = \frac{r\sqrt{2\pi}}{T} e^{-\omega_j^2 r^2/2}$, which has favorable properties both in time (it is positive) and frequency (it is smooth). Note that if the number of samples T tends to infinity with $r(T)$ tending to infinity such that $r(T)/T$ tends to zero, then Eq. (4) provides a consistent estimate of the spectral density matrix [5].

Note that by simply inverting the Fourier transform using the FFT, we can recover the autocovariance function. With the type of smoothing we chose, we know in advance an upper bound on the decay of the autocovariance as h grows. Indeed, we can simply verify that the autocovariance function Γ^r derived from the estimate $\hat{f}^r$, satisfies $||\Gamma^r(h)||_2 \leqslant e^{-h^2/r^2} ||\hat{\Gamma}(0)||_2$. Thus we can choose the time horizon H to be a constant times r (in simulations we took $H = 4r$). Limiting the time horizon is equivalent to limiting the resolution of the spectra; that is, the spectral density is represented by

```
Input: data $x_i(t)$, $i \in \{1,\dots,m\}$, $t \in \{0,\dots,T-1\}$
Algorithm:
1. Compute the $m$ discrete Fourier transforms using $m$ FFTs: $d_i(.) = \mathrm{FFT}(x_i(.))$
2. Compute the periodogram $I(k) = d(k)d(k)^*$
3. Determine optimal smoothing $r$ and degree of freedom $df$ by minimizing $S(r)$ in Eq. (6), for
   $r^{-1}$ ranging from $T^{1/5}$ to $T^{4/5}$ with a grid of step $T^{3/10}$, i.e. $O(T^{1/2})$ steps
4. Compute the spectral density matrix $f_k = \hat{f}(\omega)$ at frequencies $\omega_k = \frac{2k\pi}{T}$, for
   $k \in \{0,\dots,T-1\}$ using Eq. (4)
5. Take $H = 2^{\lfloor \log_2(4df) \rfloor}$
6. Subsample the sequence $f_k$ $T/H$ times.
Output: $df = r\sqrt{2\pi}$, $f_k$, $k \in \{0,\dots,H-1\}$
```

Figure 1: Periodogram smoothing with automatic selection of the bandwidth.

its values on the grid $\omega_k = 2\pi k/H$ for a given H. All integrals involving the density are computed using Riemannian sums. Note that by the Nyquist theorem, if $\Gamma(h)$ is equal to zero for $|h| > H$, then the spectral density can be exactly represented by a finite sample with even steps $2\pi/H$. The periodogram gives a sample with precision $2\pi/T$; to obtain a sample with precision $2\pi/H$, we need to subsample the periodogram with a positive linear filter (in order to maintain its positivity).

2.4.3 Selecting the bandwidth

We use the Akaike information criterion (AIC) to select the optimal bandwidth for a given dataset. This requires an efficient computation of the likelihood, as well as the effective number of parameters or degrees of freedom df [18].

We use the *Whittle approximation* of the likelihood:

$$\ell_w = -\frac{1}{2}\sum_{k=0}^{T-1}\Big(\log|\hat{f}(\omega_k)| + \mathrm{tr}\{\hat{f}(\omega_k)^{-1}I(\omega_k)\}\Big) - \frac{1}{2}Tm\log 2\pi, \tag{5}$$

where $|A|$ denotes the determinant of the matrix A. The Whittle approximation relies on the fact that the discrete Fourier transform of the data is asymptotically normal with independent components and with variance the spectral density.

In order to determine df, we notice that the smoothing defined in Eq. (4) is a linear smoothing; that is, $\hat{f}(\omega_k)$ is obtained from $I(\omega_k)$ by applying a linear matrix H_r. The effective number of parameters is the trace of H_r, which in our case this is simply obtained as $df_r = \mathrm{tr}\, H_r = T \times W(0) = r\sqrt{2\pi}$.

Thus, in order to find the optimal bandwidth, we minimize the following AIC criterion with respect to the smoothing parameter r:

$$S(r) = -\ell_w + \frac{df_r}{2}m^2. \tag{6}$$

(m^2 is the number of real parameters necessary to encode an $m \times m$ Hermitian matrix, and the 1/2 comes from the fact that since our signals are real, we only need the spectral density on $[0,\pi]$). The resulting algorithm is presented in Figure 1. Note that we estimate the joint spectral density matrix first and then learn the structure of the graphical model (the topic of Section 5). In Section 5.3, we show how adaptive smoothing of the periodogram can be achieved in a manner that depends on the structure of the learned graphical model.

2.5 Prediction - forecasting

Given a finite sample from time 1 to H and a model (an autocovariance function or a spectral density), forecasting is the task of predicting values for times $H+1$ and later. In the Gaussian case, the best prediction is a linear approximation of $x(H+1)$, based on $x(1),\ldots,x(H)$. More precisely, we look for matrix coefficients Φ_j such that the expected error

$$e(\Phi) = E\left\|x(H+1) - \textstyle\sum_{j=1}^{H}\Phi_j x(H+1-j)\right\|^2 \tag{7}$$

is minimal.

Minimizing $e(\Phi)$ in Eq. (7) is equivalent to solving the following system of equations, the so-called Yule-Walker equations (obtained by setting the derivatives to zero in Eq. (7)):

$$\forall j \geqslant 1, \sum_{i=1}^{H}\Psi_i\Gamma(i-j) = \Gamma(j). \tag{8}$$

We denote by γ_H the $H\times m$ vector such that $\gamma_H^\top = (\Gamma(1)\ \Gamma(2)\ \cdots\ \Gamma(H))$, and Ψ the $H\times m$ vector such that $\Psi^\top = (\Psi_1\ \Psi_2\ \cdots\ \Psi_H)$. We also use the notation

$$Q_H(\Gamma) = \begin{pmatrix} \Gamma(0) & \Gamma(1) & \cdots\ \Gamma(H-1) \\ \Gamma(-1) & \Gamma(0) & \\ \vdots & & \ddots \\ \Gamma(-H+1) & \cdots & \Gamma(0) \end{pmatrix}.$$

Then Eq. (8) can be written as:

$$Q_H\Psi = \gamma_H. \tag{9}$$

In order to solve the system in Eq. (8) or Eq. (9), the Toeplitz structure of the problem can be exploited, to avoid the $O(m^3H^3)$ complexity associated with the unstructured linear system. In particular, the innovations algorithm or the Durbin-Levinson algorithm can be used to iteratively compute the prediction and the error covariance in $O(m^3H^2)$ operations [5].

When the model is defined through the spectral density sampled at H points, then we first compute the autocovariance function using the FFT in time $O(m^2H\log H)$, and incur a cubic time complexity for inference. In Section 4, we use a different technique to solve the Yule-Walker equations in time $O(m^2H(d+\log H))$, for spectral densities that factorize in a graphical model with maximum fan-in d.

3 Graphical models for time series

The graphical model framework can be extended to multivariate time series in several ways. We follow [3] and consider dependencies between whole time series, that is between the entire sets $x_i \triangleq \{x_i(t), t\in\mathbb{Z}\}$, for $i=1,\ldots m$. Thus the time series x_i and x_j are independent if and only if the random infinite vectors $\{x_i(t), t\in\mathbb{Z}\}$ and $\{x_j(t), t\in\mathbb{Z}\}$ are independent. Similarly, time series x_i and x_j are conditionally independent given x_k if and only if $\{x_i(t), t\in\mathbb{Z}\}$ and $\{x_j(t), t\in\mathbb{Z}\}$ are conditionally independent given $\{x_k(t), t\in\mathbb{Z}\}$.

For classical graphical models and Gaussian variables, marginal and conditional independence statements can be read out from zeros in the covariance matrix or its inverse [23]. It turns out that for Gaussian stationary time series, these results can be naturally extended, essentially replacing the covariance matrix by the spectral density matrix, as we now describe.

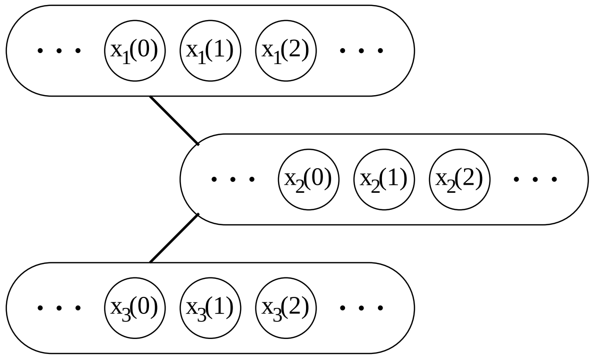

Figure 2: Graphical model for three time series: x_1 is independent from x_3 given x_2

3.1 Gaussian time series and independence

We consider a Gaussian stationary multivariate time series $x(t)$ with m components and with positive (i.e. invertible) spectral density matrix $f(\omega)$. Marginal independence and conditional independence are easily characterized in the frequency domain, as the following proposition shows [3]:

Proposition 1: *The time series x_i and x_j are* marginally independent *if and only if*

$$\forall \omega \in [0, 2\pi],\ \ f_{ij}(\omega) = 0.$$

The time series x_i and x_j are conditionally independent *given all other time series $x_k,\ k \neq i, j$, if and only if*

$$\forall \omega \in [0, 2\pi],\ \ (f(\omega)^{-1})_{ij} = 0.$$

■

Intuitively, this proposition enables us to consider each frequency component independently of the other components.

As a final step towards full equivalence between a Gaussian stationary time series and the concatenation of independent variables at each frequency, we have the following proposition, which gives a closed form expression for the KL divergence [21], paralleling the expression for the KL divergence between zero mean Gaussian vectors [29]:

Proposition 2: *The KL divergence between two zero-mean stationary vector-valued Gaussian processes U and V, with invertible spectral density matrices $f(\omega)$ and $g(\omega)$, is equal to*

$$J(f(\omega)||g(\omega)) = \frac{1}{2\pi}\int_0^{2\pi} I(f(\omega)||g(\omega))d\omega, \tag{10}$$

where $I(F||G) = -\frac{1}{2}\log\det[FG^{-1}] - \frac{1}{2}\operatorname{tr}(I - FG^{-1})$ is the KL divergence between two covariance matrices. ■

As we will see in Section 5, what is needed when learning a graphical model from data is the expression of the entropy of the random variable defined through a spectral density. The expression of interest is the entropy rate, defined as $h = \lim_{T\to\infty}\frac{1}{T}H(x(1), \ldots, x(T))$. For Gaussian stationary time series, it can be readily computed:

$$h = \frac{1}{4\pi}\int_{-\pi}^{\pi} \log\det[4\pi^2 e f(\omega)]d\omega, \tag{11}$$

a result known as Szëgo's theorem [17].

Note that h is also equal to the conditional entropy of $x(0)$ given the infinite past [10], that is, if G is the covariance matrix of $x(0)$ given the infinite past, we have:

$$h = \lim_{T\to\infty}\frac{1}{T}H(x(0)|x(-1), \ldots, x(-T)) = \frac{1}{2}\log\det 2\pi e G$$

3.2 Directed graphical models for Gaussian time series

In this paper, we consider directed graphical model representations for time series. This section reviews the semantics of directed graphical models [23], so as to make the paper self-contained. We first review the classical results for Gaussian vectors and covariance matrices and present extensions of these to Gaussian stationary time series and spectral densities.

3.2.1 Directed models and conditional independence

Let x be a Gaussian variable with covariance matrix Σ. The variable x, or the covariance matrix Σ, are said to factorize in a directed acyclic graph G, if and only if for all i, x_i is independent from its non-descendant variables given its parents [23]. For a characterization in terms of the covariance matrix, see Section 3.2.3.

We generalize this definition to stationary time series: the time series $x(t)$, with spectral density $f(\omega)$, is said to factorize in G, if for all $\omega \in [-\pi, \pi]$, the (complex) covariance matrix $f(\omega)$ factorizes in G. This is exactly equivalent to the following property: for all i, x_i is independent from its non-descendant variables given its parents, where conditional independence between time series should be understood as in the previous section.

3.2.2 Factorization of the KL divergence

Since the KL divergence factorizes in a directed graphical model for Gaussian variables [23], the previous propositions immediately imply that the KL divergence factorizes in directed models for Gaussian stationary time series. Thus parameter estimation in a directed graphical model with m time series decouples into m distinct conditional density estimates for univariate time series. This makes possible the efficient learning of the structure of a directed graphical model for time series from data, as we show in Section 5.

3.2.3 Sparse representation of the covariance and precision matrices

In this section, we show how matrix-vector products $\Sigma\alpha$ and $\Sigma^{-1}\alpha$ can be computed efficiently, when a covariance matrix Σ factorizes in a directed graphical model. This is easily seen from the "regression view" of Gaussian directed graphical models [28], which implies that Σ^{-1} and Σ can be written as

$$\begin{aligned}\Sigma^{-1} &= (I-W)^* D^{-1} (I-W) \\ \Sigma &= (I-W)^{-1} D (I-W)^{-*}\end{aligned}$$

where W is a strictly lower triangular matrix (i.e. with zero diagonal) and D is diagonal. In this interpretation, W are the regression weights and D the conditional variances. In other words, a distribution that factorizes in G is defined by the parameters W and D. Note that these parameters can be easily found from the full joint covariance matrix as follows:

$$\begin{aligned}W_i &= \Sigma_{i,\pi_i} \Sigma_{\pi_i,\pi_i}^{-1} \\ d_i &= \Sigma_{i,i} - \Sigma_{i,\pi_i} \Sigma_{\pi_i,\pi_i}^{-1} \Sigma_{\pi_i,i}\end{aligned}$$

If the maximal fan-in of the graph G is d, then for any vector α, we can compute $\Sigma^{-1}\alpha$ in $O(dm)$ operations, because the product $W\alpha$ can be computed in $O(md)$ operations. In order to be able to compute $\Sigma\alpha$, we just need to be able to solve the systems of the form $(I-W)x = \alpha$, which can be done in $O(md)$ operations, since the matrix W is triangular and has at most md nonzero elements.

4 Efficient forecasting with directed graphical models

In this section, we describe a novel algorithm for efficient forecasting when the spectral density matrix has a sparse structure. We learn the one-step-ahead predictor from the potentially infinite past, that is we learn the predictor from the H previous time steps, where the required "horizon" H is determined by the smoothness of the spectral density. We essentially need to solve the Yule-Walker equation of order H. Direct methods such as the Durbin-Levinson algorithm do not scale well with the number of variables m. However, by solving the linear system using the conjugate gradient technique, we can make use of the structure of our spectral density.

4.1 Problem set-up

We assume that the spectral density is known through H samples f_k at frequencies $\omega_k = 2k\pi/H$, and that H is large enough so that the autocovariance function is equal to the discrete Fourier transform of the sequence f_k. Also we assume that for each k, f_k has a *sparse structure*. By sparse structure we mean that for each vector $\alpha \in \mathbb{R}^m$, the product $f_k\alpha$ and f_k^{-1} can be computed in $O(sm)$ operations, where s is a constant independent of m, instead of $O(m^2)$. A first example is when f_k factorizes in a directed graphical model with maximal fan-in less than d, where we have $s = O(d)$, as shown in Section 3.2.3. A second example is where the covariance $f = f_k$ follows a factor analysis model, that is $f = WW^\top + \psi$, where W is an $m \times p$ matrix and ψ is diagonal. In that case it is easy to show that $s = O(p^3)$ (the argument is trivial for $f\alpha$, and it follows by the matrix inversion lemma for $f^{-1}\alpha$).

In this section, our interest is the design of an algorithm to determine the one-step-ahead predictor from the last H steps, so we need to solve the following Yule-Walker equations $Q_H(\Gamma)\Psi = \gamma_H$, as defined in Eq. (9). We assume that the values $\Gamma(h)$, for $h = 0, \dots, H/2$, are obtained as the first $H/2+1$ values of the FFT of order H of the sequence $(f_0, \dots, f_{H-1})$.

4.2 Conjugate gradient methods and Toeplitz systems

Conjugate gradient methods can be used to solve large linear systems of equations $Ax = b$ of size n, where the matrix A is Hermitian positive, and where the evaluation of the product Ax can be performed in $O(n)$ operations (instead of $O(n^2)$). It is an iterative method where at each iteration one matrix-vector product has to be performed. The number of iterations is governed by the condition number of the matrix A—the ratio between its largest eigenvalue and its smallest eigenvalue. Unfortunately, in many cases A is ill-conditioned and the number of iterations can be very large. A common solution is to precondition the matrix A: instead of solving $Ax = b$, the system $C^{-1}AC^{-1}y = C^{-1}b$ is solved, which yields a solution for the original system through $x = C^{-1}y$. In order to make conditioning practical there are two requirements for the matrix C: the linear system involving C^2 must be solved cheaply and the condition number of $C^{-1}AC^{-1}$ must be small.

For Toeplitz matrices, there is a natural choice of C obtained through the approximate diagonalization of Toeplitz matrices in the Fourier basis [16]. Indeed, any block Toeplitz matrix of the form

$$\mathcal{T} = \begin{pmatrix} T_0 & T_{-1} & \cdots & T_{-H+1} \\ T_1 & T_0 & & \\ \vdots & & \ddots & \\ T_{H-1} & \cdots & & T_0 \end{pmatrix}$$

can be approximated by a circulant matrix of the form

$$\mathcal{C} = \begin{pmatrix} C_0 & C_{-1} & \cdots & C_{-H+1} \\ C_1 & T_0 & & \\ \vdots & & \ddots & \\ C_{H-1} & \cdots & & C_0 \end{pmatrix}$$

where $C_k = T_k$ for $|k| \leqslant \frac{H}{2} - 1$ and $C_k = C_{n-k}$ (which defines the values for $|k| \geqslant \frac{H}{2}$).

The circulant approximation is very useful because it can be diagonalized in a fixed basis, that is

$$\mathcal{C} = \mathcal{F}\mathcal{D}\mathcal{F}^*$$

where $\mathcal{F}$ is defined by blocks, that is $\mathcal{F}_{jk} = \frac{1}{\sqrt{H}} \exp(2ijk\pi/H)I$ and $\mathcal{D}$ is block diagonal with diagonal blocks $\widehat{C}_k = \frac{1}{\sqrt{H}} \sum_{j=0}^{H-1} C_k \exp(-2ijk\pi/H)$. Since circulant matrices can be diagonalized in the Fourier basis $\mathcal{F}$, they can be easily inverted as

$$\mathcal{C}^{-1} = \mathcal{F}\mathcal{D}^{-1}\mathcal{F}^*.$$

Also the computation of the product between a circulant matrix and an $Hm \times Hm$ matrix requires $O(H \log Hm^2 + Hm^3)$ operations, or only $O(H \log Hm^2 + Hm^2 s)$ operations if matrices with sparse structure are used. Finally any Toeplitz matrix can be embedded in a circulant matrix of size $2H$, which makes it possible to multiply Toeplitz matrices in time $O(2H \log(2H)) = O(H \log H)$.

The circulant matrix obtained when embedding the Toeplitz matrix Q_H is exactly the $2H$-th order circulant matrix with generating sequence $\bar{f}_k$. To precondition, we have two choices, either (a) to precondition with the Toeplitz matrix R_H with generating function $1/\bar{f}_k$, or (b) to precondition directly with the inverse circulant matrix associated with T_H. Both choices have been studied [6, 7, 8] and we chose the first option, since it does not require subsampling the spectral density. In Figure 3, we give an outline of the resulting algorithm.

4.3 Computing ν-step-ahead predictors

Using the concept of spectral factorization, we can compute any ν-step-ahead predictor and error covariances efficiently. The solution of the linear system is the optimal one-step-ahead filter, with transfer function $A_1(\omega) = \sum_{k=0}^{\infty} \Psi_{k+1} e^{-ik\omega}$. Let A_ν be the transfer function of the ν-step-ahead filter, and G_ν its error. We have the following theorem [31, 32]:

Theorem 1 : *The error covariance $G = G_1$ of the one-step-ahead predictor is*

$$G = \int_{-\pi}^{\pi} (I - e^{-i\omega} A_1(\omega)) f(\omega) d\omega.$$

If $\psi(\omega) = I - e^{-i\omega} A_1(\omega)$ and $\phi(\omega) = \psi^{-1}(\omega) = I - \sum_{k=1}^{\infty} \phi_k e^{-ik\omega}$, the ν-step-ahead filter can be written as:

$$A_\nu(\omega) = e^{i\nu\omega}(I - \sum_{k=0}^{\nu-1} \phi_k \phi^{-1}(\omega))$$

and the error covariance is $G_\nu = \sum_{k=0}^{\nu-1} \phi_k G \phi_k^\top$ ■

Thus, once the one-step-ahead filter is found, all other filters and errors can be computed. However, this requires inverting H matrices of size m. This can be avoided by noticing that $\phi(\omega)$ can be obtained by computing the one-step-ahead predictor corresponding to the spectral density $\bar{f}^{-1}(\omega)$. If m is very large, such that multiplying two full matrices of size m is prohibitive, we can obtain the the ν-step-ahead predictor and error covariance by solving the Toeplitz system $Q_H \Psi_\nu = \gamma_{H,\nu}$, where $\gamma_{H,\nu} = (\Gamma(\nu), \ldots, \Gamma(\nu + H - 1))$.

4.4 Comments

4.4.1 Convergence analysis

Results from [8] show that the number of iterations is independent of the order H, and depends on the smoothness of the spectral density as well as how far $f(\omega)$ is from singular.

4.4.2 Interpretation in the frequency domain

The previous algorithm has an interpretation in the frequency domain which links it to previous approaches [31, 32] that do not rely on efficient methods for linear systems.

4.4.3 Total complexity

Let κ be the condition number of the matrices $f(\omega)$, H the number of samples required for the computation of the Fourier transforms, m the number of samples, d the maximum fan-in. We have:

- Each iteration: $O(m^2 H \log H)$ for the FFTs, $O(m^2 dH)$ for the multiplication by $f(\omega)$.
- Overall complexity of computing the predictor: $O(m^2 H(\log H + d))$.

5 Learning directed graphical models

In this section, we present our algorithm for learning directed graphical models for stationary Gaussian time series. Not surprisingly, we make heavy use of the corresponding algorithms for the Gaussian temporally independent case. We cast the structure learning as a model selection problem where the model is defined by the directed graph G. We use the Akaike Information Criterion (AIC) in order to define the score $J(G)$ to be minimized. Once the score is determined, the task of minimizing it is performed with greedy algorithms, as detailed in Section 5.2.

5.1 AIC score for time series

We are given T consecutive samples of m univariate times series, $x_i(t)$, $i \in \{1, \dots, m\}$ and $t \in \{1, \dots, T\}$. We first estimate the joint spectral density matrix $\hat{f}(\omega)$ as well as the estimated degrees of freedom df, as shown in Section 2.4. The spectral density is represented as a set of H samples f_k at frequencies $\omega_k = 2k\pi/H$.

For directed models, the AIC score $J(G)$ is equal to the maximum of the likelihood plus a penalty score. It is known to factorize [19, 13]:

$$J(G) = \sum_{i=1}^{m} J_i(\pi_i(G)) \tag{12}$$

where $\pi_i = \pi_i(G)$ are the parents of node i in G and

$$J_i(\pi_i(G)) = -T\hat{H}(x_i | x_{\pi_i}) + \#\{i, \pi_i(G)\}$$

where $\#\{i, \pi_i(G)\}$ is the number of parameters required for encoding the conditional probability distribution of x_i given the parents $x_{\pi_i(G)}$, and $\hat{H}(x|y)$ is conditional entropy of the random vector x given the random vector y, computed using the estimated distribution of the joint (spectral) density.

Using the expression of the KL divergence and entropy rate in Eq. (10) and Eq. (11), we get the following expression for the local score (where we omit the terms that are independent of the choice of the graph G):

$$J_i(\pi_i) = \frac{-T}{4\pi} \int_{-\pi}^{\pi} \log \frac{|\hat{f}_{\{i\}\cup\pi_i}(\omega)|}{|\hat{f}_{\pi_i}(\omega)|} d\omega + (2|\pi_i| + 1)\frac{df}{2} \tag{13}$$

where $|\pi_i|$ is the cardinality of π_i (i.e. the number of parents) and $f_A(\omega)$ denotes the square block (A, A) of the $m \times m$ matrix $f(\omega)$. The previous local score is approximated using the samples of $\hat{f}(\omega)$ as

$$J_i(\pi_i) = \frac{-T}{2H} \sum_{k=0}^{H-1} \log \frac{|(f_k)_{\{i\}\cup\pi_i}|}{|(f_k)_{\pi_i}|} + (2|\pi_i| + 1)\frac{df}{2}. \tag{14}$$

```
Input: Spectral density values $f_k \in \mathbb{R}^{m\times m}$, at frequencies $\omega_k = k\pi/2H$, $k \in \{0,\dots,H-1\}$,
       precision $\varepsilon$

Algorithm:

1. Computing covariances:
       $\forall h \in \{0,\dots,H-1\}$, $r(:,:,h) \leftarrow \bar{f}(:,:,h)$
       $\forall (i,j) \in \{1,\dots,m\}^2$, $r(i,j,:) \leftarrow \mathrm{IFFT}(r(i,j,:))$
                                $r(i,j,:) \leftarrow r(i,j,0:H/2-1)$

2. Initialization: $k \leftarrow 0$, $\gamma_1 \leftarrow 0$, $\rho \leftarrow \sum_{i,j,t} |r(i,j,t)|^2$
3. Compute $\Psi$: while $\sqrt{\rho} > \varepsilon$ :
       $k \leftarrow k+1$
       $\forall (i,j) \in \{1,\dots,m\}^2$, $z(i,j,:) \leftarrow [r(i,j,:);0\dots0]$
                                $z(i,j,:) \leftarrow \mathrm{IFFT}(z(i,j,:))$

       $\forall h \in \{0,\dots,H-1\}$, $z(:,:,h) \leftarrow \bar{f}(:,:,h)^{-1} z(:,:,h)$
       $\forall (i,j) \in \{1,\dots,m\}^2$, $z(i,j,:) \leftarrow \mathrm{FFT}(z(i,j,:))$
                                $z(i,j,:) \leftarrow z(i,j,0:H/2-1)$

       $\gamma_0 \leftarrow \gamma_1, \gamma_1 \leftarrow \sum_{i,j,t} \bar{r}(i,j,t) z(i,j,t)$
       if $k = 1$ then $p = z$ else $\beta \leftarrow \gamma_1/\gamma_0$, $p \leftarrow z + \beta p$
       $\forall (i,j) \in \{1,\dots,m\}^2$, $w(i,j,:) \leftarrow [p(i,j,:);0\dots0]$
                                $w(i,j,:) \leftarrow \mathrm{IFFT}(w(i,j,:))$

       $\forall h \in \{0,\dots,H-1\}$, $w(:,:,h) \leftarrow \bar{f}(:,:,h) w(:,:,h)$
       $\forall (i,j) \in \{1,\dots,m\}^2$, $w(i,j,:) \leftarrow \mathrm{FFT}(w(i,j,:))$
                                $w(i,j,:) \leftarrow w(i,j,0:H/2-1)$

       $\alpha \leftarrow \sum_{i,j,t} \bar{p}(i,j,t) w(i,j,t)$, $\alpha \leftarrow \gamma_1/\alpha$
       $g \leftarrow g + \alpha p$
       $r \leftarrow r - \alpha w$
       $\rho \leftarrow \sum_{i,j,t} |r(i,j,t)|^2$

4. Compute $G$ and $\Psi_i$ :
       $\forall h \in \{0,\dots,H/2-1\}$, $g(:,:,h) \leftarrow g(:,:,h)^\top$
       $\forall (i,j) \in \{1,\dots,m\}^2$, $w(i,j,:) \leftarrow [0;g(i,j,:);0\dots0]$
                                $w(i,j,:) \leftarrow \mathrm{FFT}(w(i,j,:))$

       $\forall h \in \{0,\dots,H-1\}$, $w(:,:,h) \leftarrow (I - w(:,:,h))\bar{f}(:,:,h)$
       $G \leftarrow \frac{1}{H}\sum_t w(:,:,t)$            $\forall h \in \{1,\dots,H/2\}$, $\Psi_h = g(:,:,h-1)$

Output: $G$, $\Psi_i$, $h \in \{1,\dots,H/2\}$
```

Figure 3: Solving the Yule-Walker equations using the spectral density.

```
Input: data $x_i(t)$, $i \in \{1,\ldots,m\}$, $t \in \{1,\ldots,T\}$
Algorithm:
1. Estimate the joint spectral density $f_k \hat{f}(\omega_k)$ and the degree of freedom $df$ using the algorithm
   in Figure 1, for $\omega_k = 2k\pi/H$
2. Minimization of the AIC score $J(G)$
   a. Initialization: $G = \varnothing$
   b. while no decrease $J(G)$, select best local moves: addition, deletion or reversal
3. Compute optimal smoothing $r_i$ separately for each clique $\{i\} \cup \pi_i(G)$ using local smoothing
4. Compute the spectral density matrix $f_k^G$ that factorizes in $G$ using the local sufficient statis-
   tics
Output: $G$, $f_k^G$
```

Figure 4: Learning graphical models for stationary Gaussian time series.

5.2 Learning algorithm

In order to learn the graph structure G, we need to minimize the score $J(G)$ defined by Eq. (12) and Eq. (13). This minimization problem is known to be NP-complete [9], and greedy algorithms are commonly used. In particular, since the score $J(G)$ factorizes as a sum of local scores, hill-climbing search using local moves—edge addition, deletion or removal—is computationally efficient (in particular through the caching of already computed local scores) and, with the possible use of random restarts, yields good local minima [19, 13]. An outline of the final algorithm is presented in Figure 4. The exact complexity of the search procedure which uses caching of local scores is presented in Section 5.4.

5.3 Local smoothing and efficiency

One of the major gains from learning a sparse structure for the spectral density matrix is that we can perform (and optimize) the smoothing of the periodogram locally in the graph, restricting ourselves to elements that share a clique; that is, instead of applying the algorithm of Figure 1 once with m variables, we can apply it m times with $|\pi_i(G)| + 1$ variables. In the algorithm presented in Figure 4, a joint pilot estimate of the spectral density is used to determine the graph G, then the local smoothing is performed. We now explore some of the issues, both numerical and statistical, associated with such smoothing.

Numerically, learning the best smoothing parameter for the joint spectral density is an $O(m^3 T \log T)$ operation. In order to overcome the cubic time complexity in m, it is possible to learn a different smoothing parameter r_i for each local potential that is required, which makes the algorithm only $O(d^3 T \log T)$ where d is the maximal fan-in of the directed graph. Note that in the algorithm of Figure 4, it is possible to avoid the cubic time complexity of the computation of the pilot estimate simply by considering an amount of smoothing that is the maximum of the amounts of smoothing for each of the variables taken separately (note that this has a tendency to undersmooth and make the graphical model sparser).

Statistically, learning a smoothing parameter for the joint density when there is a strong local structure would lead to oversmoothing (since the AIC penalty becomes relatively too important): local smoothing is substantially more efficient. In the algorithm presented in Figure 4, the local smoothing is performed once the structure is learned, thus requiring m distinct smoothing parameter searches.

Finally, alternating between learning local smoothing parameters and directed graphs is possible, in a manner analogous to the procedure of [13]. In that situation the global score to be optimized is the concatenation of Eq. (6), Eq. (12) and Eq. (13). More precisely, if r_i denotes the smoothing parameters for the clique $\{i\} \cup \pi_i(G)$ and $\hat{f}^{r_i}$ denotes the smoothed periodogram with parameter r_i, we need to compute the Whittle likelihood, and then add the number of parameters, as shown in the next theorem.

Theorem 2: *The Whittle likelihood for a spectral density that factorizes in a directed graph G, where the conditional spectral density are computed using a smoothing constant r_i and with resulting estimates $\hat{f}^{r_i}_{\{i\}\cup\pi_i}(\omega)$ for the cliques $\{i\} \cup \pi_i$ is approximately equal to:*

$$\ell_W(G,r) = -\frac{T}{2H}\sum_{k=0}^{H-1}\sum_{i=1}^{m}\left(\log\frac{|(f_k^{r_i})_{\{i\}\cup\pi_i}|}{|(f_k^{r_i})_{\pi_i}|} - \operatorname{tr}\left\{(f_k^{r_i})^{-1}_{\{i\}\cup\pi_i} I_{\{i\}\cup\pi_i}(\omega_k)\right\} + \operatorname{tr}\left\{(f_k^{r_i})^{-1}_{\pi_i} I_{\pi_i}(\omega_k)\right\}\right) \tag{15}$$

■

The overall AIC score is thus equal to:

$$J(G,r) = \ell_W(G,r) + \sum_{i=1}^{m}(2|\pi_i|+1)\frac{df_i}{2}. \tag{16}$$

Note that the optimal local smoothing that results from optimizing r_i independently from G and other smoothing constants r_j, $j \neq i$, is different from the optimal smoothing on the clique $\pi_i \cup \{i\}$. In simulations on a wide variety of examples, it turns out that the difference between the two types of smoothing is very small, with a slight advantage for the conditional smoothing.

5.4 Running time complexity

In this section, we study more closely the computational aspects of our algorithm in cases where the number of variables is very large, potentially on the order of thousands. We denote by m the number of these variables, d the maximum fan-in that we impose on our networks, T the number of observations and H the time horizon. As seen in Figure 4, the structure learning has several successive stages, whose running time complexities are:

1. Estimate (marginal) smoothing parameters r_i for each of the m variables. Cost: $O(mT^{3/2}\log T)$.
2. Take $r = \frac{1}{m}\sum_i r_i$, and compute/smooth the periodogram for the joint density. Cost: $O(m^2 T\log T)$.
3. Structure learning using greedy search: when using the efficient scheme of [15], it can be made $O(m^3d^2 + m^2d^4H)$.
4. Estimate local smoothing parameters. Cost: $O(md^3T^{3/2}\log T)$.
5. Overall cost: $O(md^3T^{3/2}\log T + m^2T\log T + m^3d^2 + m^2d^4H))$.

6 Non-linearity through Mercer kernels

In this section, we show how the methods presented thus far can be extended to nonlinear models. In particular, in Section 6.2, we show how these methods can be "kernelized"; based on this we develop algorithms for learning in Section 6.3 and prediction in Section 6.4. As already mentioned, the basic principle underlying our approach is to first map the data x into a "feature space" $\mathcal{F}$ using a "feature map" $\Phi(x)$, and to use the algorithms developed in earlier sections on the transformed data $\Phi(x)$. What makes the approach feasible is that only dot products between data points are needed by these algorithms, as seen in Section 6.2 and Section 6.3. Thus, the algorithm only involves manipulation of the values $k(x,y) = \langle\Phi(x),\Phi(y)\rangle$ for x and y being two data points in the original space. $k(x,y)$ is usually referred to as the *kernel function.* A function $k(x,y)$ is a Mercer kernel if and only there exists a feature space $\mathcal{F}$ and a feature map $\Phi(x)$ such that $k(x,y) = \langle\Phi(x),\Phi(y)\rangle$.

Since the algorithm only involves dot products and thus kernel values, algorithms can benefit from the expressive power of the feature space, without incurring the computational cost of actually mapping the points into this feature space, a method usually referred to as the "kernel trick" [27]. Mercer kernels can be defined on many different input spaces, thus making the applicability of the presented technique wide.

6.1 Notation

We assume that the variables $x_i(t)$ are mapped to $\varphi_i(t) = \Phi_i(x_i(t)) \in \mathcal{F}_i$, where Φ_i are given feature maps from the "input space" $\mathcal{X}_i$ to the feature space $\mathcal{F}_i$. Let $k_i(x_i, y_i) = \langle \Phi_i(x_i), \Phi_i(y_i) \rangle$ be the dot product between two mapped elements. We are given T samples, from $t = 0$ to $t = T - 1$. We can build m "Gram matrices" K_i (one per variable) as $T \times T$ matrices composed of the pairwise kernel values $k_i(x_i(u), x_i(v))$ between two data points.

6.2 Kernelization of the periodogram

In this section, we show how we can find a basis in which the periodogram has a particularly simple expression involving products of Gram matrices. Since the model selection scores are independent of the basis, we can use these expressions in Section 6.3 to construct our model selection criteria.

We assume that the data are centered[2], that is, $\sum_t \varphi_i(t) = 0$. The sample covariance matrix $\widehat{\Gamma}(h)$, which is defined by blocks, has its (i, j)-block equal to, for $h \geqslant 0$:

$$\widehat{\Gamma}_{ij}(h) = \frac{1}{T} \sum_{t=h}^{T-1} \varphi_i(t) \varphi_j(t-h)^\top.$$

For $s, u \in \{0, \ldots, T-1\}$, we have

$$\begin{aligned} \varphi_i(s)^\top \widehat{\Gamma}_{ij}(h) \varphi_j(u) &= \frac{1}{T} \sum_{t=h}^{T-1} \varphi_i(s)^\top \varphi_i(t) \varphi_j(t-h)^\top \varphi_j(u) \\ &= \frac{1}{T} \sum_{t=h}^{T-1} (K_i)_{s,t} (K_i)_{t-h,u} = \frac{1}{T} \delta_s K_i J_h K_j^\top \delta_u \end{aligned}$$

where δ_s is the vector in $\mathbb{R}^T$ with only zeros except a one at index s, K_i is the Gram matrix associated with variable i, and J_h is the $T \times T$ matrix such that $(J_h)_{ab} = \delta(a - b = h)$. The only non-zero elements of J_h belong to the h-th lower diagonal. Thus, in the *data basis* defined by the data points, the autocovariance function has its (i, j)-th block equal to $\frac{1}{T} K_i J_h K_j$. The Fourier transform of the autocovariance has the following expression:

$$\begin{aligned} I_{ij}(\omega) &= \sum_{h=-(T-1)}^{T-1} \Gamma_{ij}(h) e^{-ih\omega} \\ &= \frac{1}{T} \sum_{h=-(T-1)}^{T-1} K_i J_h e^{-ih\omega} K_j^\top \\ &= \frac{1}{T} K_i \left(\sum_{h=-(T-1)}^{T-1} J_h e^{-ih\omega} \right) K_j^\top \\ &= \frac{1}{T} K_i f_\omega f_\omega^* K_j^\top = \frac{1}{T} K_i f_\omega (K_i f_\omega)^* \end{aligned}$$

[2]The data can be implicitly centered in feature space by replacing Gram matrices K by $(I - \frac{1}{T}\mathbf{1})\ K\ (I - \frac{1}{T}\mathbf{1})$, where $\mathbf{1}$ is a $T \times T$ matrix composed of ones [27].

where f_ω is the vector in $\mathbb{R}^T$ with components $e^{-ih\omega}$, $h \in \{0, \ldots, T-1\}$. The periodogram is exactly $I_{ij}(\omega_k)$ for $\omega_k = 2k\pi/T$.

Since the feature space might have infinite dimension, smoothing (in space) is usually required to limit the number of parameters that are implicitly estimated. Smoothing (in space) the periodogram by an isotropic Gaussian is equivalent to adding κI; in the data basis, the regularized (cross)-periodogram is thus

$$I_{ij}(\omega) = \frac{1}{T} K_i f_\omega f_\omega^* K_j^\top + \kappa \delta_{ij} K_i.$$

Smoothing (in frequency) can be done as in the non-kernelized version by averaging consecutive values of the periodogram. We let denote $(f_k)_{ij}$, $k = 1, \ldots, H$ the values of the estimated spectral density after smoothing. Each $(f_k)_{ij}$ is an $T \times T$ matrix.

6.3 Model selection criteria

Following [1], the effective dimension of variables i is taken to be $d_i = \operatorname{tr}(K_i + \kappa I)^{-1} K_i$ and we use the following AIC score:

$$J(G) = \sum_{i=1}^{m} \left\{ \frac{-T}{2H} \sum_{k=0}^{H-1} \log \frac{|(f_k)_{\{i\}\cup\pi_i, \{i\}\cup\pi_i}|}{|(f_k)_{\pi_i,\pi_i}|} + (2 \sum_{j\in\pi_i} d_j + 1) \frac{df}{2} \right\} \tag{17}$$

The same optimization techniques used for classical graphical models can be used to optimize this score. Efficient implementation techniques based on low-rank approximations are presented in [1].

6.4 Prediction with kernels

Once we have a model, we would like to perform prediction. Given our framework, we obtain a predicted value $\hat{\phi} = (\hat{\phi}_1, \ldots, \hat{\phi}_m)$ in the feature space, and this predicted value might not correspond to an an x_i such that $\hat{\phi}_i = \Phi_i(x_i)$. As for the covariance error, it is not defined in input space. In order to get an estimate and an error, we propose the following two-stage procedure. Once the predicting filters $\hat{\phi}_i(t) = f_i(\phi(t-1), \ldots, \phi(t-H))$ are computed:

1. For each variable i and each index $t \in [H, T-1]$, and find

$$\hat{x}_i(t) = \arg\min_{x_i} ||\Phi_i(x_i) - f_i(\phi(t-1), \ldots, \phi(t-H))||.$$

   This is usually referred to as the "pre-image" and several techniques are available [30, 27].

2. Compute the average errors. In particular, if all variables are continuous, we get:

$$G = \frac{1}{T-H} \sum_{t\in[H,T-1]} (\hat{x}(t) - x(t))(\hat{x}(t) - x(t))^\top$$

7 Simulations

7.1 Synthetic examples

Our first goal is to see whether, when the data are generated from a sparse graphical model, the search procedure can find this model. We use the following procedure: we generate a random spectral density matrix, project the spectral density matrix function to a random graph with maximal fan-in equal to two, generate data from the new spectral density matrix with various numbers of samples, learn the model from data using the algorithm described in Figure 4. We then compute the KL divergence from the generating model as well as the number of undirected edges not found by the

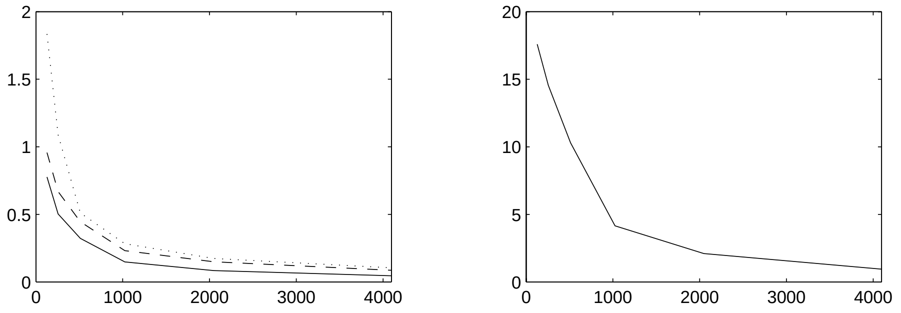

Figure 5: Synthetic examples. Left: KL divergence from the truth vs. number of samples. (dotted) fully connected graphical model, (dashed) learned graphical model without post-smoothing, (plain) learned graphical model with post-smoothing. Right: number of non recovered edges vs. number of samples.

number of variables	20	40	80
iid Gaussian graphical model	20.6	36.5	68.2
Sparse AR model	14.8	32.0	68.0
Independent spectral densities	20.4	41.9	84.7
Graphical model for time series	13.7	26.2	50.7

Figure 6: Results for the climate datasets: negative log likelihood

learning algorithm. We compare the performance in KL divergence for the following models: a fully-connected graphical model, a learned graphical model without post-smoothing, a learned graphical model with post-smoothing (as described in Section 5.3). We can see in Figure 5 that the structure learning algorithm manages to predict significantly better than the fully-connected graphical model and that the post-smoothing is essential.

7.2 Real-life datasets

We compare our algorithms to a classical approach that exhibits reasonable scaling of computing time to large datasets. Note in particular that with large datasets such as climate data, two issues need to be addressed: (a) there are a very large number of variables m, (b) there are relatively few time samples compared to the number of variables. The model we compare to is the sparse autoregressive model, SAR(p), of maximal order p. These are AR(p) models which favor zeros in the weights Φ_i and are such that the error matrix Σ factorizes in a sparse graphical model G. They are learned using forward selection of edges with the AIC criterion.

We used climate datasets extracted from the National Climatic Data Center (NCDC) "summary of the day" data, which are composed of daily mean temperatures and precipitation for more than 10,000 stations across the United States. We subsampled this dataset by chosing random locations and choosing the closest stations. The number of samples was 1024 consecutive days. Each of the datasets that was so constructed was divided in a training set (of size 1024 days) and a testing set. For each of these sets, the obvious seasonal component (year) is removed by using the first six corresponding Fourier components. We report the negative log likelihood of the one-step-ahead prediction of the test data. We normalize the result by subtracting the log likelihood of the corresponding independent Gaussian variables. We also report the average prediction error. We report results in Figure 6 and 7: we can see that the approach based on sparsity in the frequency domain outperforms the approaches based on sparsity in the time domain.

number of variables	20	40	80
iid Gaussian graphical model	2.5	2.6	2.6
Sparse AR model	0.55	0.57	0.58
Independent spectral densities	0.72	0.77	0.78
Graphical model for time series	0.56	0.55	0.53

Figure 7: Results for the climate datasets: average prediction error

8 Conclusions

Probabilistic graphical models provide an elegant general framework for the representation of complex sets of dependencies among an interacting set of variables. Standard algorithms are available for probabilistic inference, and a variety of parameter estimation and model selection algorithms have been devised. The graphical structure of these models is essential for making these algorithms computationally efficient. It has important statistical implications as well, yielding models in which the graphical structure corresponds directly to a natural notion of sparsity of the representation.

Applications of graphical models to time series analysis have generally taken the form of state space models. In this setting, the graphical model machinery is aimed at capturing structure in the state transition matrix, and sparsity in the graph has an interpretation in the time domain—a given state variable has a probabilistic dependence on a limited number of variables in the past.

In the current paper, we have described an alternative methodology for making use of graphical models in time series analysis. In particular, we have developed a frequency domain approach in which the structure captured by a graphical model is related to sparseness in the spectral density matrix. We have described parameter estimation and structure learning algorithms that are geared for this setting. We have provided computational complexity analyses throughout, emphasizing the development of methods that are appropriate for large-scale time series models.

As seen in the experiments with climate data, methods that attempt to uncover structure in the frequency domain can lead to better predictive performance than analogous methods operating in the time domain.

Finally, it is worth noting that while we have restricted ourselves to regularly sampled time series in this paper, our algorithms apply immediately to irregularly sampled time series, once the spectral density is estimated [25].

References

[1] F. R. Bach and M. I. Jordan. Learning graphical models with Mercer kernels. In *Advances in Neural Information Processing Systems 15*, 2003.

[2] P. Bloomfield. *Fourier Analysis of Time Series: An Introduction.* Wiley & Sons, 2000.

[3] D. R. Brillinger. Remarks concerning graphical models for time series and point processes. *Rev. Econ.*, 16:1–23, 1996.

[4] D. R. Brillinger. *Time Series: Data Analysis and Theory.* SIAM, 2001.

[5] P. J. Brockwell and R. A. Davis. *Time Series: Theory and Methods*. Springer-Verlag, 1991.

[6] R. H. Chan. Toeplitz preconditioners for Toeplitz systems with nonnegative generating functions. *IMA J. Numer. Anal.*, 11(3):333–345, 1991.

[7] R. H. Chan, J. G. Nagy, and R. J. Plemmons. Fft-based preconditioners for Toeplitz-block least squares problems. *SIAM J. Numer. Anal.*, 30(6):1740–1768, 1993.

[8] R. H. Chan and M. K. Ng. Conjugate gradient methods for Toeplitz systems. *SIAM Review*, 38(3):427–482, 1996.

[9] D. M. Chickering. Learning Bayesian networks is NP-complete. In *Learning from Data: Artificial Intelligence and Statistics 5*. Springer-Verlag, 1996.

[10] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* Wiley & Sons, 1991.

[11] R. Dahlhaus. Graphical interaction models for multivariate time series. *Metrika*, 51:157–172, 2000.

[12] T. Dean and K. Kanazawa. A model for reasoning about persistence and causation. *Comput. Intel.*, 5(3):142–150, 1989.

[13] N. Friedman and M. Goldszmidt. Learning Bayesian networks with local structure. In *Learning in Graphical Models.* MIT Press, 1998.

[14] D. Geiger and D. Heckerman. Learning Gaussian networks. In *Proc. UAI*, 1994.

[15] P. Giudici and R. Castelo. Improving Markov chain Monte-Carlo model search for data mining. *Machine Learning*, 50:127–158, 2003.

[16] R. M. Gray. Toeplitz and circulant matrices: A review. Technical report, Information Systems Laboratory, Stanford Univ., 2002.

[17] E. J. Hannan. *Multiple Time Series.* John Wiley & Sons, 1970.

[18] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning.* Springer-Verlag, 2001.

[19] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[20] M. I. Jordan. Graphical models. *Statistical Science (Special Issue on Bayesian Statistics)*, 2002. In press.

[21] D. Kazakos and P. Papantoni-Kazakos. Spectral distances between Gaussian processes. *IEEE Trans. Aut. Cont.*, AC-25:950–959, 1980.

[22] W. Lam and F. Bacchus. Learning Bayesian belief networks: An approach based on the MDL principle. *Comput. Intel.*, 10(4):269–293, 1994.

[23] S. L. Lauritzen. *Graphical Models.* Clarendon Press, 1996.

[24] K. Murphy. *Dynamic Bayesian Networks: Representation, Inference and Learning.* PhD thesis, UC Berkeley, Computer Science Division, 2002.

[25] E. Parzen. *Time Series Analysis of Irregularly Observed Data.* Springer-Verlag, 1983.

[26] T. J. Richardson and R. L. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Trans. Inform. Theory*, 47(2):599–618, 2001.

[27] B. Schölkopf and A. J. Smola. *Learning with Kernels.* MIT Press, 2001.

[28] R. Shachter and R. Kenley. Gaussian influence diagrams. *Management Science*, 35(5):527–550, 1989.

[29] T. P. Speed and H. T. Kiiveri. Gaussian Markov distributions over finite graphs. *Annals of Statistics*, 14(1):138–150, 1986.

[30] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. In *Adv. NIPS 15*, 2003.

[31] N. Wiener and P. Masani. The prediction theory of multivariate stochastic processes. I. The regularity conditions. *Acta Math.*, 98:111–150, 1957.

[32] N. Wiener and P. Masani. The prediction theory of multivariate stochastic processes. II. The linear predictor. *Acta Math.*, 99:93–137, 1958.