Modeling changing dependency structure in multivariate time series

Xiang Xuan XXUAN@CS.UBC.CA
Kevin Murphy MURPHYK@CS.UBC.CA
Department of Computer Science, University of British Columbia, Vancouver, BC, Canada

Abstract

We show how to apply the efficient Bayesian changepoint detection techniques of Fearnhead in the multivariate setting. We model the joint density of vector-valued observations using undirected Gaussian graphical models, whose structure we estimate. We show how we can exactly compute the MAP segmentation, as well as how to draw perfect samples from the posterior over segmentations, simultaneously accounting for uncertainty about the number and location of changepoints, as well as uncertainty about the covariance structure. We illustrate the technique by applying it to financial data and to bee tracking data.

1. Introduction

Time series segmentation (also known as changepoint detection) has many applications, and a large number of techniques have been proposed to tackle this problem. One of the most difficult issues is estimating the number of segments. As in other examples of model selection, the Bayesian approach is particularly appealing, since it automatically captures a tradeoff between model complexity (number of segments) and model fit. It also allows one to express uncertainty about the number, and location, of changepoints.

In a series of papers (Fearnhead, 2004; Fearnhead & Liu, 2005; Fearnhead, 2006), Fearnhead developed efficient dynamic programming algorithms for exactly computing the posterior over the number and location of changepoints in time series. This improved upon earlier approaches, such as (Punskaya et al., 2002), which relied on reversible jump MCMC.

All of the examples that Fearnhead considered were univariate (one-dimensional) time series. In this paper, we show how to apply Fearnhead's algorithms to multidimensional time series. Specifically, we model the correlation structure of the vector-valued observations using sparse Gaussian graphical models, which scale to high dimensions better than full covariance matrices. We estimate the structures and the segmentation jointly. This allows us to segment based on a changing correlation structure, as well as changing mean, variance, etc, which is particularly useful in financial applications (Talih & Hengartner, 2005; Carvalho & West, 2006). Furthermore, the sparse structure within each segment is often interpretable.

Figure 1 illustrates the basic problem we are trying to solve. In the case of 1D time series, a segmentation might be induced by a change of mean or variance or model order (in the case of an autoregressive process). But in the case of multidimensional time series, we can additionally segment if the correlation structure changes. Such changes are often much harder to detect (see Figure 1(b) for example), but oftem arise in practice, especially in areas such as finance. Our model can segment data based on all of these kinds of changes, as we will see.

2. Previous work

A product partition model (PPM) (Barry & Hartigan, 1992; Barry & Hartigan, 1993; Denison et al., 2002) is a density model in which we assume we can partition the data into an unknown number K of partitions, $\pi_1, \ldots, \pi_K$, such that the data is independent across segments:

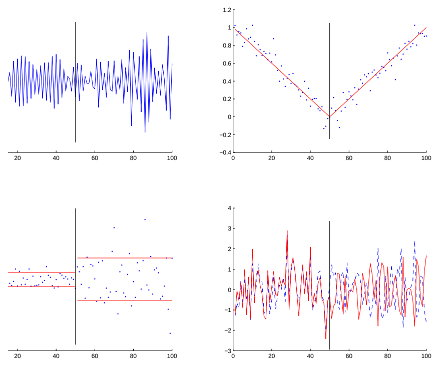
$$p(y_{1:T}|\pi) = \prod_{k=1}^{K} p(y_{\pi_k}).$$

(A dirichlet process mixture model is a special case of a PPM, in which we assume a specific form for $p(\pi)$ (Dahl, 2003).) If we assume that the partitions are non-overlapping partitions of the interval $1 : T$, then we can efficiently compute the posterior over K and π using dynamic programming, as Fearnhead showed; we shall call such a partitioning a segmentation.

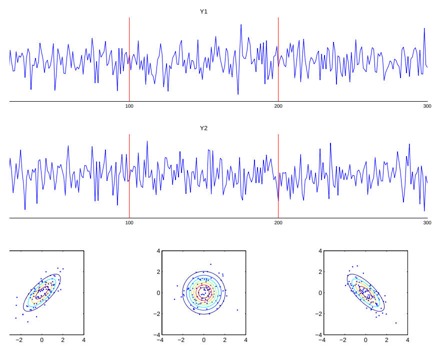
The marginal likelihood that data from s to t is produced by a single model m is given by

$$p(y_{s:t}|m) = \int [\prod_{i=s}^{t} p(y_i|y_{1:i-1}, \theta, m)] p(\theta|m) p(m) d\theta$$

Appearing in *Proceedings of the 24th International Conference on Machine Learning*, Corvallis, OR, 2007. Copyright 2007 by the author(s)/owner(s).



(a)



(b)

Figure 1. Some examples of changepoint detection. **(a)** Top Left: changing AR model order. Top right: changing regression slope. Bottom left: changing noise variance. Bottom right: changing correlation. **(b)** Changing correlation structure between two time series. In the first segment the components are positively correlated; in the middle segment, they are uncorrelated; in the final segment they are anti correlated. We visualize the empirical covariance matrix of each segment by plotting contours of constant probability density, assuming a Gaussian model.

Intuitively, if the segment grows to include data generated from different parameter regimes and/or different model types, the marginal likelihood will drop, and this will suggest that we should introduce a changepoint, and use two models. We will give some specific forms for the likelihood below. We shall assume that the prior $p(\theta)$ is conjugate, so that we can compute the marginal likelihood in closed form.

In addition to the data likelihood, we need to specify a prior on segment lengths, $p(\ell)$, which implicitly defines a prior on segmentations $p(\pi)$. Following Fearnhead, we shall assume a geometric distribution with parameter λ, so that the probability of a segment of length ℓ is $p(\ell) = \lambda(1-\lambda)^{\ell-1}$. The number of segments depends on λ, but also on the hyper-parameters α of $p(\theta)$, as we shall see below. We use $\lambda = 0.01$.

In (Fearnhead, 2004; Fearnhead, 2006), Fearnhead proposed a dynamic programming algorithm to compute the MAP segmentation

$$(K^*, \pi^*) = \arg\max_{K,\pi_{1:K}} p(y|\pi_{1:K})p(\pi_{1:K})$$

as well as a way to draw perfect samples from this posterior. (Conditioned on any segmentation, it is of course possible to compute the posterior over models and parameters in each segment, $p(m, \theta|y_{s:t})$.) The algorithm is very similar to the forwards-filtering backwards-sampling algorithm (Scott, 2002) for HMMs, except the "hidden variable" is not a discrete *state* index, but rather a *time* index encoding where the last change point occurred. Hence the algorithm takes $O(T^2)$ time and $O(T)$ space. In (Fearnhead & Liu, 2005), Fearnhead and Liu present an on-line algorithm that takes $O(T^2)$ time and space if solved exactly, but $O(T)$ time and space if solved approximately (by pruning hypotheses whose probability falls below a threshold, and only keeping the most probable hypotheses).

The input to all three algorithms is λ (or more generally $p(\ell)$), and a function that can compute the marginal likelihood $p(y_{s:t}|m)$ for any data segment $y_{s:t}$ given a model m; we shall denote this function by obslik$(y_{s:t}, m, \alpha)$, where α are hyper-parameters used by the observation model m. In addition, we need to specify the class of possible models $\mathcal{M}$, and a prior over models, $p(m)$, for $m \in \mathcal{M}$. Finally, for the approximate on-line algorithm, we need to specify a pruning threshold ϕ and a number of particles N_p (we use $\phi = 10^{-40}$ and $N_p = 50$). Hence the interface to the segmentation function is

$$(K, \pi) = \text{segment}(y_{1:T}, \lambda, p(m), \text{obslik}(\cdot,\cdot,\alpha), \phi, N_p)$$

Unfortunately we do not have the space to explain how this function works; see (Fearnhead & Liu, 2005) for details.

In this paper, we focus on the problems of specifying suitable observation likelihood functions obslik(), and of creating suitable hypothesis spaces $\mathcal{M}$. In particular, we propose to use Gaussian graphical models (GGMs) to represent $p(y_i|\theta, m)$, where $y_i \in \mathbb{R}^d$ and m is the graph structure, as we explain in Section 3 below. We estimate the graph structure using L1-penalized maximum likelihood (see Section 4).

A closely related paper by Talih and Hengartner (2005) also uses GGMs to segment multivariate time series, but they cannot use dynamic programming, since their model is not a PPM. Specifically, they assume that the graph structure changes slowly over time (one arc addition or deletion at each segment boundary), which violates the assumption that the parameters of each model in each segment are independent. They then use reversible jump MCMC to estimate the segmentation and the (non-decomposable) structures. Our technique is much faster and can draw perfect samples from the posterior (given $\mathcal{M}$). Also, we estimate the number of segments, whereas Talih and Hengartner assume this is known.

It is interesting to compare the product partition model (PPM) to a hidden Markov model (HMM) and an "infinite HMM" (IHMM) (Beal et al., 2002). In an HMM, we have a fixed number of states S; we label each time step with a state, which specifies which parameters to use to generate an observation. In a PPM, we have an unbounded number of states; we label each time step with the corresponding parameter to use, but we can never re-use an old state label once we have left a segment, so the label sequence is strictly increasing. An infinite HMM is a blend between these two models: we have an unbounded number of states/ parameters, but we can revisit an old state at any time.

Given these distinctions, it should be clear when to use — and when not to use — a PPM. In particular, if we think each segment is generated from a "fresh" set of parameters that we have not used before, a PPM is appropriate, but if we expect to return to an old parameter regime, an HMM or IHMM may be more appropriate. If the number of regimes is known, we can use an HMM, which supports dynamic programming for inference, but if the number of regimes is unknown, we have to use an IHMM, for which more expensive techniques such as Gibbs sampling must be used for inference. The GGM version of the PPM presented in this paper is an interesting compromise between these extremes, since it assumes you can "revisit" old *structures*, but you cannot revisit old parameters (since they are integrated out of each segment). This is possible because we condition on $\mathcal{M}$, and thus can share structures across time. We give more details below.

3. Observation likelihood models

We now discuss several approaches for modeling $p(y_{s:t})$ when $y_i \in \mathbb{R}^d$ is the d-dimensional observation at time i. We will compare their performance below.

3.1. Independent features model

When modeling multivariate time series, a simple approach is to assume each feature is independent (as in naive Bayes):

$$p(y_{s:t}) = \prod_{j=1}^{d} p(y_{s:t,j}) = \prod_{j=1}^{d} \left(\int [\prod_{i=s}^{t} p(y_{i,j}|\theta_j)] p(\theta_j) d\theta_j \right)$$

Suppose $y_{ij} \sim \mathcal{N}(0, \sigma_j^2)$ (we consider the case of non zero mean below) and $\sigma_j^2 \sim \chi^{-2}(N_0, V_{0j}) = IG(N_0/2, V_{0j}/2)$, where V_{0j} is our prior variance and N_0 is the strength of this prior. Then we can compute the inner integral to give

$$p(y_{s:t,j}) = \pi^{-n/2} \frac{V_{0j}^{N_0/2}}{V_{nj}^{(N_0+n)/2}} \frac{\Gamma(N_0/2)^{-1}}{\Gamma((N_0+n)/2)^{-1}}$$

where $n = t - s + 1$ is the length of the segment, $V_{nj} = V_{0j} + \sum_{i=s}^{t} y_{ij}^2$, and Γ is the gamma function. We call this the "independent features model". Unfortunately, this cannot capture correlations between the features, which is crucial for certain domains such as finance.

3.2. Full covariance model

A more expressive choice is to model y_i with a multivariate Gaussian, $y_i \sim \mathcal{N}(0, \Sigma)$. We will use an inverse Wishart prior, $\Sigma \sim IW(N_0, V_0)$ which is a generalization of inverse chi-squared. Here $N_0 > d - 1$ is the degrees of freedom and V_0 is the scale matrix. The marginal likelihood is as follows

$$\begin{aligned} p(y_{s:t}) &= \pi^{-\frac{nd}{2}} \frac{|V_0|^{N_0/2}}{|V_n|^{(N_0+n)/2}} \frac{\Gamma_d(N_0/2)^{-1}}{\Gamma_d((N_0+n)/2)^{-1}} \\ V_n &= V_0 + S, \quad S = \sum_{i=s}^{t} y_i y_i^T, \\ \Gamma_d(n) &= \pi^{d(d-1)/4} \prod_{i=0}^{n-1} \Gamma(n - \frac{i}{2}) \end{aligned}$$

where $\Gamma_d(n)$ is the multivariate gamma function (so $\Gamma_1(n) = \Gamma(n)$). When we increase/decrease the segment by one time step (as required by Fearnhead's algorithm), we can perform a rank-one update/downdate of S, so overall the cost is dominated by computing the matrix determinant $|V_n|$. We typically use the relatively uninformative hyper parameter values of $N_0 = d$ and $V_0 = \hat{\sigma}^2 I$, where $\hat{\sigma}^2$ is the mean of the empirical variance pooled across all the data. (Using this data-dependent prior is similar to preprocessing the data by scaling.) Note that $N_0 = d$ is the smallest value (weakest prior) we can use if the prior is to remain proper. We discuss the issue of hyperparameters in more detail in Section 5.2.

3.3. Gaussian graphical model

Although the full-covariance model works quite well for low-dimensional problems, the number of parameters needed by this model is $O(d^2)$. A fully diagonal approximation to Σ results in the independent features model. Gaussian graphical models provide a good compromise between these two extremes. We can either use directed graphs (i.e., Bayes nets; see e.g., (Geiger & Heckerman, 1994)) or undirected models (see e.g., (Lauritzen, 1996; Giudici & Green, 1999; Carvalho & West, 2006)). In this paper, we use undirected graphs, since there are very efficient procedures for estimating undirected graph structures (see Section 4), which is needed to compute the model space $\mathcal{M}$.

Computing the marginal likelihood for non decomposable graphical models cannot be done in closed form. Various approximations have been proposed (Roverato, 2002), but these are slow. So we shall assume that the graph structure is decomposable. Give a decomposable graph structure G, and assuming a hyper-inverse Wishart prior $\Sigma \sim HIW_G(b_0, V_0)$ (Dawid & Lauritzen, 1993), where $b_0 = N_0 + 1 - d > 0$ is the degree of freedom and V_0 is the location parameter, the marginal likelihood can be written as follows (Dawid & Lauritzen, 1993; Giudici & Green, 1999; Carvalho & West, 2006):

$$p(y_{s:t}|G) = (2\pi)^{-nd/2} \frac{h(G, b_0, V_0)}{h(G, b_0 + n, V_n)} \tag{1}$$

where

$$h(G, b, V) = \frac{\prod_{c \in C} |\frac{V_c}{2}|^{b_c/2} \Gamma_{|c|}(b_c/2)^{-1}}{\prod_{s \in S} |\frac{V_s}{2}|^{b_s/2} \Gamma_{|s|}(b_s/2)^{-1}} \tag{2}$$

where C are the cliques and S are the separators, and $b_c = b + |c| - 1$, $b_s = b + |s| - 1$. (If we use $C = \{\{1, \ldots, d\}\}$ and $S = \{\emptyset\}$, then Equation 1 reduces to Equation 1.) The cost of computing this equation is $O(w^3)$, where w is the treewidth of the graph. This can be much less than the $O(d^3)$ required by the full covariance model. We typically use the relatively uninformative hyperparameters $b_0 = 1$ (i.e., $N_0 = d$) and $V_0 = \hat{\sigma}^2 I$, as in the full covariance case.

If the structure of the graph is not known, we can marginalize it out, using

$$p(y_{s:t}) = \sum_{g \in \mathcal{M}} p(y_{s:t}|g) p(g) \tag{3}$$

where $\mathcal{M}$ is the space of possible models (graph structures). We discuss how to compute $\mathcal{M}$ in Section 4 below. Although computing this expression looks expensive, $p(y_{s:t}|g)$ decomposes into a sum of local terms (since the graphs are decomposable), so much of the computation can be shared across graphs.

3.4. Linear regression for the mean

So far we have assumed that each segment is zero mean, $y_i \sim \mathcal{N}(0, \Sigma)$, and have concentrated on modelling the covariance Σ. However, we can also model the mean μ using multivariate linear regression. Specifically, we assume $y_{s:t} = H\beta + \epsilon$, where H is a $n \times q$ design matrix (n is the length of the segment, q is the number of input features per time slice), β is a $q \times d$ matrix of regression parameters, and $\epsilon \sim \mathcal{N}(0, I_n, \Sigma)$, where $N(M, V, \Sigma)$ is the matrix Gaussian distribution (Dawid, 1981)

$$N(A; M, V, \Sigma) = \frac{|\Sigma|^{n/2}}{|2\pi V|^{d/2}} \times \exp(-\frac{1}{2} tr((A - M)^T V^{-1} (A - M)\Sigma)) \tag{4}$$

where M is a $n \times d$ matrix representing the means, V is a $n \times n$ matrix representing covariance amongst the rows (time slices), and Σ is a $d \times d$ matrix representing covariance amongst the columns (features). If we assume $\beta|\Sigma \sim \mathcal{N}(0, D, \Sigma)$ where $D = \text{diag}(\delta_1^2, \ldots, \delta_q^2)$, and $\Sigma \sim IW(N_0, V_0)$, then the marginal likelihood is given by (Minka, 2000)

$$\begin{aligned}
p(y_{s:t}) &= \pi^{-\frac{nd}{2}} \left(\frac{|M|}{|D|}\right)^{\frac{d}{2}} \frac{|V_0|^{N_0/2}}{|V_n|^{(n+N_0)/2}} \frac{\Gamma_d(N_0/2)^{-1}}{\Gamma_d((N_0+n)/2)^{-1}} \\
M &= (H^T H + D^{-1})^{-1},\ P = (I - HMH^T) \\
V_n &= V_0 + Y^T P Y
\end{aligned}$$

Using this framework, we can model multivariate autoregressive processes. It is straightforward to combine this approach of modeling the mean with the earlier methods for graphical modeling of Σ, by multiplying Equation 1 by $(|M|/|D|)^{d/2}$.

4. Estimating graph structures

How do we generate the hypothesis space $\mathcal{M}$? We face a chicken and egg situation: if we knew the segmentation, we could run a standard structure learning algorithm on each segment, but we need to know the structures in order to compute the segmentation.

We propose the following iterative solution. First we create $\mathcal{M}$ using a heuristic approach described below; we then perform a segmentation using Fearnhead's algorithm; we then re-compute $\mathcal{M}$, by applying the same structure learning algorithm used in initialization, but to the top N segmentations, as opposed to the heuristically chosen segmentations. This is summarized in Algorithm 1. The whole algorithm can be thought of as approximate inference in a hierarchical Bayesian model, where $\mathcal{M}$ is at the top of the hierarchy. The marginal likelihood of each segment is then given by integrating over the parameters and by marginalizing over all the models in $\mathcal{M}$.

To estimate a graph structure from a segment $y_{s:t}$, we use the fact that absent edges correspond to zeros in the precision matrix Λ (Lauritzen, 1996). We therefore compute the MAP estimate for Λ under a prior that encourages many entries to go to zero. We can pose this as the following

Algorithm 1 Sketch of overall algorithm

```
1: Input: data $y_{1:T}$, changepoint rate $\lambda$, pruning threshold $\phi$, number of particles $N_p$, regularization penalty $\rho$, observation model obslik(), observation hyperparameters $\alpha$
2: Output: $(K, \pi_{1:K}, m_{1:K})$.
3: Algorithm:
4: $\mathcal{S}$ = make overlapping segments from $y_{1:T}$
5: $\mathcal{M} = \{\text{estG}(y_s, \rho) : s \in \mathcal{S}\}$
6: **while** not converged **do**
7:     $(K, \pi) = \text{segment}(y_{1:T}, \lambda, 1/|\mathcal{M}|, \text{obslik}(), \phi, N_p)$
8:     $\mathcal{M} = \{\text{estG}(y_{\pi_i}, \rho) : \pi_i \in \pi\}$
9: **end while**
10: $m_i = \arg\max_m p(m|y_{\pi_i})$ for $i = 1 : K$
```

convex optimization problem:

$$\max_{\Lambda \succ 0} \log\det\Lambda - \text{trace}(\hat{\Sigma}\Lambda) - \rho||\Lambda||_1 \tag{5}$$

where $\hat{\Sigma}$ is the empirical covariance matrix derived from $y_{s:t}$, $||\Lambda||_1 = \sum_{ij} |\Lambda_{ij}|$, and $\rho > 0$ is a regularization parameter which controls the sparsity of the graph. (We set $\rho = 0.2$.) We solve this using the block coordinate descent algorithm of (Banerjee et al., 2006), and denote this procedure by $G = \text{estG}(y_{s:t}, \rho)$. This takes $O(Id^4)$ time, where I is the number of iterations.

Other, faster methods exist for estimating sparse undirected GGMs. One method that we tried is the following: compute a shrinkage estimate of $\hat{\Sigma}$, using the technique pf (Schaefer & Strimmer, 2005). From this, compute the precision $\hat{K} = \hat{\Sigma}^{-1}$ and the partial correlation coefficients

$$\rho_{ij} = -\frac{K_{ij}}{\sqrt{K_{ii}K_{jj}}} \tag{6}$$

Then set edge $G_{ij} = 0$ if $|\rho_{ij}| < \theta$, where θ is some threshold (we use 0.2); otherwise set $G_{ij} = 1$. A more principle approach would use a local false discovery rate criterion to pick the threshold adaptively (Schaefer & Strimmer, 2005). However, we have not yet tried this. Note that this thresholding-based approach takes $O(d^3)$ time, so scales much better than the block coordinate descent method. In the future, we would also like to try L1-based regression methods (Meinshausen & Buhlmann, 2006) for estimating K, which are also $O(d^3)$.

The above methods can learn arbitrary graph structures. Since our computation of the marginal likelihood assumes the graph is decomposable, we convert each non-decomposable graph in our set $\mathcal{M}$ into its "closest" decomposable approximation by computing a min-fill triangulation (Cowell et al., 1999).

To get the process started, we use the following heuristic. We slide a window of width $w = 0.2T$ across the data, shifting by $\sigma = 0.1w$ at each step, to create a set candidate segmentations. We then repeat this for different window sizes w and shifts σ. The hope is that this oversegmentation will contain the "true" segments, or at least something similar. We then apply the structure learning algorithm to all of these windows and use this as our initial guess of $\mathcal{M}$.

There is an obvious tradeoff between accuracy and speed. We set the w and σ parameters so that they generate about 100 segments; the number of unique graphs that results (and hence the size of $\mathcal{M}$) obviously depends on the data, but in our experiments is about 20–40. If the true structure is not in the initial model set because we did not guess the correct segmentation boundary, it is possible that the first iteration of the Fearnhead algorithm will nevertheless recover a good segmentation, and that the true model will be recovered in the second iteration. However, a limiting factor is the ability to detect the true structure even given the correct segmentation. Obviously we could consider other methods for structure learning besides MAP estimation with an L1 prior, and we could lift the decomposable graph assumption, but that is beyond the scope of this paper.

We use a uniform prior on structures, $p(g) = 1/|\mathcal{M}|$. Nevertheless, due to the Bayesian Occam's razor effect, the method will not always pick the most complex graph for each segment.

5. Experimental results

In the experiments below, we only performed one iteration of the algorithm, for simplicity. Preliminary results suggest that re-estimating the structures after segmentation did not improve results significantly, suggesting that our initial heuristic oversegmentation is adequate for recovering $\mathcal{M}$.

5.1. Bee data

We first applied our method to a 3 dimensional data set used in (Oh et al., 2006). This consists of the x and y coordinates of a honeybee, and its head angle θ, as it moves around an enclosure, as observed by an overhead camera.[1] Some examples of the data, together with a ground truth segmentation (created by human experts), are shown in Figure 2. We also show the results of segmenting this using a first-order auto-regressive AR(1) model, using independent features or with a full covariance model. (Using a GGM gives similar results to the full model in this low dimensional setting.) We see that the independent features model oversegments,

[1]We preprocessed the data by replacing θ with $\sin\theta$ and $\cos\theta$; this improves the results considerably, since there is no longer a discontinuity as the bee moves between $-\pi$ and π. Surprisingly, we found it slightly better to include both $\sin\theta$ and $\cos\theta$ than to include either one alone.

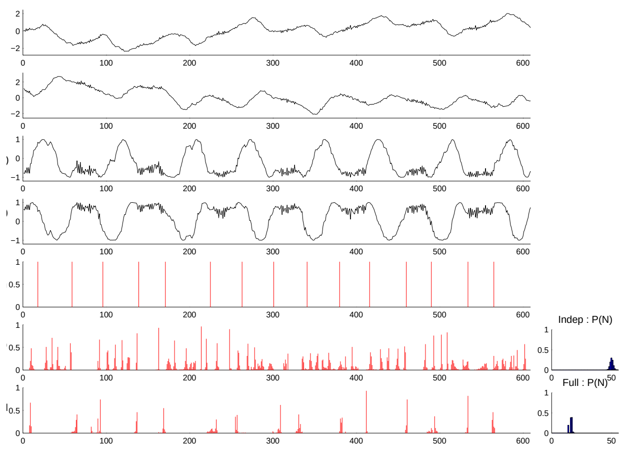


Figure 2. Bee sequence fit with AR(1) model. The intermittent high frequency oscillations correspond to the bees' "waggle dance". "Truth" refers to manual segmentation. "Indep" refers to the independent features model. "Full" refers to the full covariance model. The histogram on the right is an estimate of the number of segments.

because each signal has too many small changes, whereas what matters is the coordinated change.

Note that other techniques also perform well on this data. In particular, (Oh et al., 2006) use a switching latent state space model. This has the additional advantage that the posterior over the discrete switch states can be used to classify (label) each segment. This model assumes that parameters are stationary over time, so whenever a bee enters the "waggle" state, its waggle parameters are the same as the last time it entered that state. This seems like a reasonable assumption for this domain. In contrast, the PPM will use new parameters for each regime, which would be more appropriate in a non-stationary environment.

5.2. Synthetic data

To test the ability of the method to recover structure, as well as its ability to scale to larger dimensions, we applied it to a synthetic 20-dimensional dataset. The advantage of using synthetic data is that we know the ground truth about the location of the changepoints *and* the structure of the generating model. We sample $n = 100$ data points from 3 different sparse decomposable Gaussian graphical models, and then concatenate the data: see Figure 3. We then attempt to segment this data using Fearnhead's algorithm and 3 different likelihood models: the independent features model, the full covariance model, and the GGM, where the set of possible $\mathcal{M}$ is generated using the sliding window heuristic. As can be seen from Figure 3, the independent features model does very poorly (in this case, it thinks all the data comes from one segment), as does the full covariance model (which heavily oversegments the data) but the GGM model correctly identifies the three segments. Furthermore, the posterior over graph structures has very low entropy, and the resulting MAP graphs have structures that are reasonably close to the truth (in terms of number of correct/ incorrect edges).

We obtained similar results with other experiments of this kind. The full covariance method works for low dimensions (up to say $d = 5$), but in higher dimensions, it performs poorly, because there is not enough data to estimate a full Σ in each segment (unless each segment is very long). Note that by performing Bayesian estimation, we avoid the numerical problems encountered with a maximum likelihood estimate of a full Σ; however, the relatively uninformative prior we use does not overcome the statistical problem. The GGM approach is a more structured hypothesis space, which is helped by the fact that our heuristic way of creating $\mathcal{M}$ gets to look at the data first. In addition, we have found, in informal experiments, that the full covariance model is much more sensitive to the hyperparameter V_0 than the GGM approach. (Setting priors for variance parameters is known to be a delicate issue (Gelman, 2006), especially when considering model uncertainty.) In all the experiments, we used $V_0 = \hat{\sigma}^2 I$ as a reasonable default prior.

5.3. Financial data

Finally we applied the method to some financial data, the "30 industry portfolios" dataset.[2] This consists of the

[2]The data is available from `http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html`.

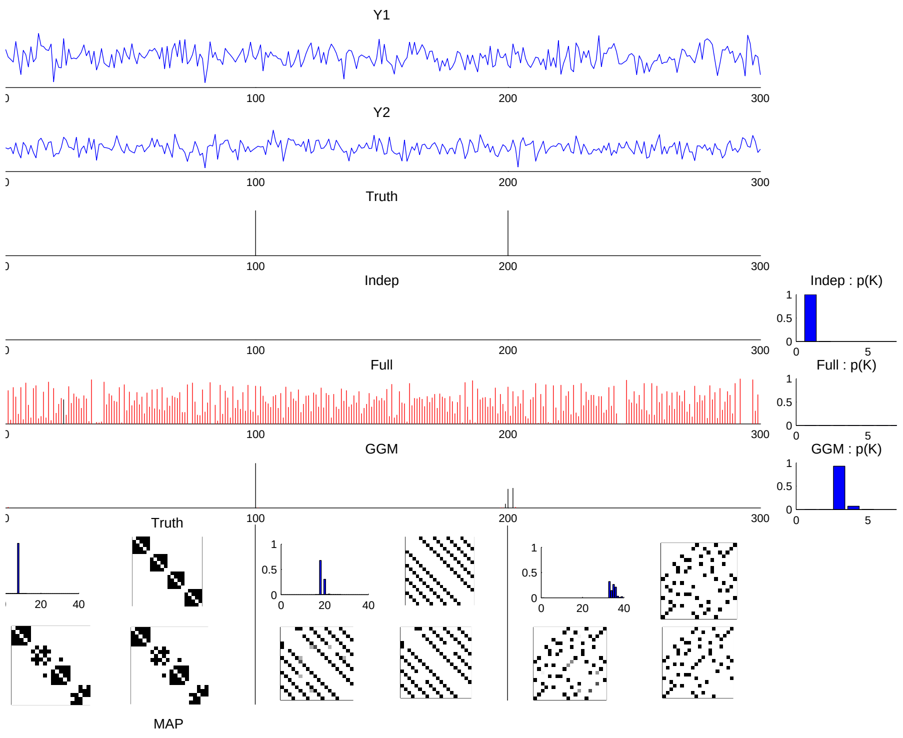


Figure 3. Results on synthetic 20 dimensional data. To save space, we only plot the first two variables, $y_{1,1:T}$ and $y_{2,1:T}$. We then show, in order, the true segmentation boundaries (at $t = 100$ and $t = 200$), and samples from the posterior over segmentations generated using the independent model, the full covariance model, and the Gaussian graphical model (GGM). To the right of each segmentation we plot the posterior over the number of segments, $p(K|y)$. We can see that the independence model thinks there is 1 segment (so no 'spikes' occur on the plot on the left); the full covariance thinks there are many segments, and the GGM (correctly) thinks there are 3 segments. For each of the three segments (as estimated by the GGM), we plot (clockwise from top left) the posterior over the $|\mathcal{M}| = 40$ possible model structures, $p(G|y_{s:t})$, the true structure (shown as an adjacency matrix), the MAP structure $G_{MAP} = \arg\max_G p(G|y_{s:t})$, and the marginal edge probabilities, $p(G_{ij} = 1|y_{s:t})$, computed using Bayesian model averaging. (Gray squares represent edges about which we are uncertain.)

valued-weighted monthly returns of 30 different industry sectors over the period 1927-2006. (A similar, but smaller (5 dimensional) dataset was analysed in (Talih & Hengartner, 2005).) The data is already approximately zero mean, but we take a log transform, to make the Gaussian assumption more reasonable. The results are shown in Figure 4, and take about 10 minutes to generate. (For this problem, we use the threshold method to generate the list of candidate models $\mathcal{M}$.) In this problem, we do not know the ground truth segmentation, let alone the true graph structure (if there is such a thing!). Nevertheless, our results seem qualitatively reasonable, although there is no obvious correspondence to known events of historical importance.

Since the truth is unknown, it is hard to assess the quality of these results. Ultimately the validity of a model is determined by its usefulness. Recent work by Carvahlo and West (Carvalho & West, 2006) has shown that using sparse GGMs for portfolio design can result in increased profits. They assumed the structure of the graph was constant over time, and they estimate it offline using all the data. It is possible that the techniques in this paper would provide a way to extend their results to non-stationary environments. However, we leave this to future work.

6. Discussion

We have shown how to apply Fearnhead's algorithms to multivariate time series, and also how to perform graphical model selection in each segment. In the future, we would like to investigate better ways to create the hypothesis space $\mathcal{M}$, as well as ways to make the techniques work in an online setting, which is required for financial (and other) applications.

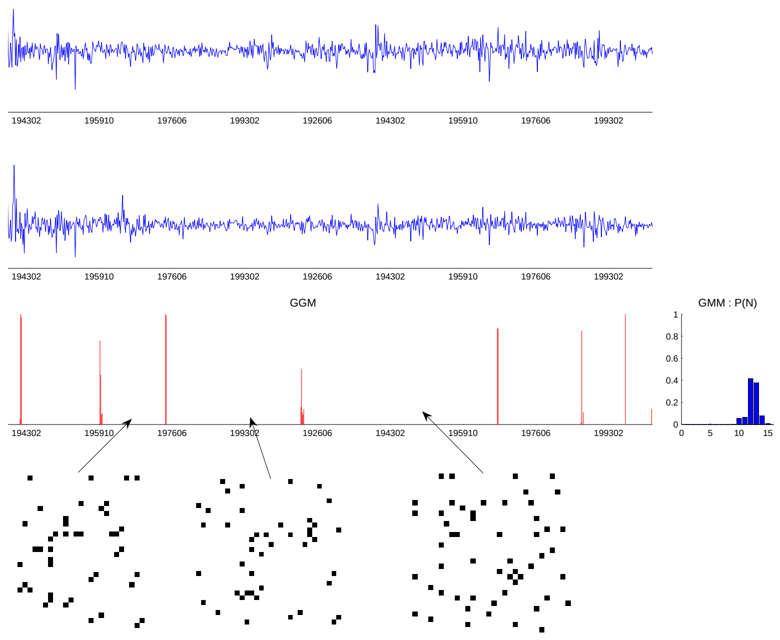


Figure 4. The portfolios data for 30 industry types. We only show the data for 2 industries, for clarity. We show the estimated graph structures for 3 of the segments, to illustrate their sparsity. The x-axis labels are in the form YYYYMM, for year and month.

References

Banerjee, O., Ghaoui, L. E., d'Aspremont, A., & Natsoulis, G. (2006). Convex optimization techniques for fitting sparse gaussian graphical models. *Intl. Conf. on Machine Learning*.

Barry, D., & Hartigan, J. (1993). A Bayesian analysis for change point problems. *J. of the Am. Stat. Assoc.*, *88*, 309–319.

Barry, D., & Hartigan, J. A. (1992). Product partition models for change point problems. *Annals of statistics*, *20*, 260–279.

Beal, M. J., Ghahramani, Z., & Rasmussen, C. E. (2002). The infinite hidden Markov model. *NIPS-14*.

Carvalho, C. M., & West, M. (2006). Dynamic matrix-variate graphical models. *Bayesian Analysis*. To appear.

Cowell, R. G., Dawid, A. P., Lauritzen, S. L., & Spiegelhalter, D. J. (1999). *Probabilistic networks and expert systems*. Springer.

Dahl, D. (2003). *Modal clustering in a univariate class of product partition models* (Technical Report). Dept. Statistics, Univ. Wisconsin.

Dawid, A. P. (1981). Some matrix-variate distribution theory: some notational considerations and a bayesian application. *Biometrika*, *68*, 265–274.

Dawid, A. P., & Lauritzen, S. L. (1993). Hyper-markov laws in the statistical analysis of decomposable graphical models. *The Annals of Statistics*, *3*, 1272–1317.

Denison, D., Holmes, C., Mallick, B., & Smith, A. (2002). *Bayesian methods for nonlinear classification and regression*. Wiley.

Fearnhead, P. (2004). Exact bayesian curve fitting and signal segmentation. *IEEE Trans. Signal Processing*, *53*, 2160–2166.

Fearnhead, P. (2006). Exact and efficient bayesian inference for multiple changepoint problems. *Statistics and computing*. To appear.

Fearnhead, P., & Liu, Z. (2005). *Online inference for multiple changepoint problems* (Technical Report). U. Lancaster.

Geiger, D., & Heckerman, D. (1994). Learning Gaussian networks. *UAI* (pp. 235–243).

Gelman, A. (2006). Prior distributions for variance parameters in hierarchical models. *Bayesian Analysis*.

Giudici, P., & Green, P. (1999). Decomposable graphical gaussian model determination. *Biometrika*, *86*, 785–801.

Lauritzen, S. (1996). *Graphical models*. OUP.

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

Minka, T. (2000). *Bayesian linear regression* (Technical Report). MIT.

Oh, S., Rehg, J., & Dellaert, F. (2006). Parameterized duration modeling for switching linear dynamic systems. *CVPR*.

Punskaya, E., Andrieu, C., Doucet, A., & Fitzgerald, W. (2002). Bayesian curve fitting using MCMC with applications to signal segmentation. *IEEE Trans. on Signal Processing*, *50*, 747–758.

Roverato, A. (2002). Hyper inverse wishart distribution for non-decomposable graphs and its application to bayesian inference for gaussian graphical models. *Scand. J. Statistics*, *29*, 391–411.

Schaefer, J., & Strimmer, K. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statist. Appl. Genet. Mol. Biol*, *4*.

Scott, S. (2002). Bayesian methods for hidden markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*.

Talih, M., & Hengartner, N. (2005). Structural learning with time-varying components: tracking the cross-section of financial time series. *J. Royal Stat. Soc. B*, *67*, 321–341.