Bayesian Inference on Change Point Problems

by

Xiang Xuan

B.Sc., The University of British Columbia, 2004

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

Master of Science

in

The Faculty of Graduate Studies

(Computer Science)

The University Of British Columbia

March, 2007

Abstract

Change point problems are referred to detect heterogeneity in temporal or spatial data. They have applications in many areas like DNA sequences, financial time series, signal processing, etc. A large number of techniques have been proposed to tackle the problems. One of the most difficult issues is estimating the number of the change points. As in other examples of model selection, the Bayesian approach is particularly appealing, since it automatically captures a trade off between model complexity (the number of change points) and model fit. It also allows one to express uncertainty about the number and location of change points.

In a series of papers [13, 14, 16], Fearnhead developed efficient dynamic programming algorithms for exactly computing the posterior over the number and location of change points in one dimensional series. This improved upon earlier approaches, such as [12], which relied on reversible jump MCMC.

We extend Fearnhead's algorithms to the case of multiple dimensional series. This allows us to detect changes on correlation structures, as well as changes on mean, variance, etc. We also model the correlation structures using Gaussian graphical models. This allow us to estimate the changing topology of dependencies among series, in addition to detecting change points. This is particularly useful in high dimensional cases because of sparsity.

Table of Contents

List of Figures

Acknowledgements

I would like to thank all the people who gave me help and support throughout my degree.

First of all, I would like to thank my supervisor, Professor Kevin Murphy for his constant encouragement and rewarding guidance throughout this thesis work. Kevin introduced me to this interesting problem. I am grateful to Kevin for giving me the Bayesian education and directing me to the research in machine learning. I am amazed by Kevin's broad knowledge and many quick and brilliant ideas.

Secondly, I would like to thank Professor Nando de Freitas for dedicating his time and effort in reviewing my thesis.

Thirdly, I would like to extend my appreciation to my colleagues for their friendship and help, and especially to the following people: Sohrab Shah, Wei-Lwun Lu and Chao Yan.

Last but not least, I would like to thank my parents and my wife for their endless love and support.

XIANG XUAN

The University of British Columbia
March 2007

Chapter 1

Introduction

1.1 Problem Statement

Change point problems are commonly referred to detect heterogeneity of temporal or spatial data. Given a sequence of data over time or space, change points split data into a set of disjoint segments. Then it is assumed that the data from the same segment comes from the same model. If we assume

$$data \quad = \quad model + noise$$

then the data on two successive segments could be different in the following ways:

- different models (model types or model orders)
- same model with different parameters
- same model with different noise levels
- in multiple sequences, different correlation among sequences

Figure 1.1 shows four examples of changes over successive segments. In all four examples, there is one change point at location 50 (black vertical solid line) which separates 100 observations into two segments. The top left panel shows an example of different model orders. The 1st segment is a 2nd order Autoregressive model and the 2nd segment is a 4th order Autoregressive model. The top right panel shows an example of same model with different parameters. Both segments are linear models, but the 1st segment has a negative slope while the 2nd segment has a positive slope. The bottom left panel shows an example

of same model with different noise level. Both segments are constant models which have means at 0, but the noise level (the standard deviation) of the 2nd segment is three times as large as the one on the 1st segment. The bottom right panel is an example of different correlation between two series. We can see that two series are positive correlated in the 1st segment, but negative correlated in the 2nd segment.

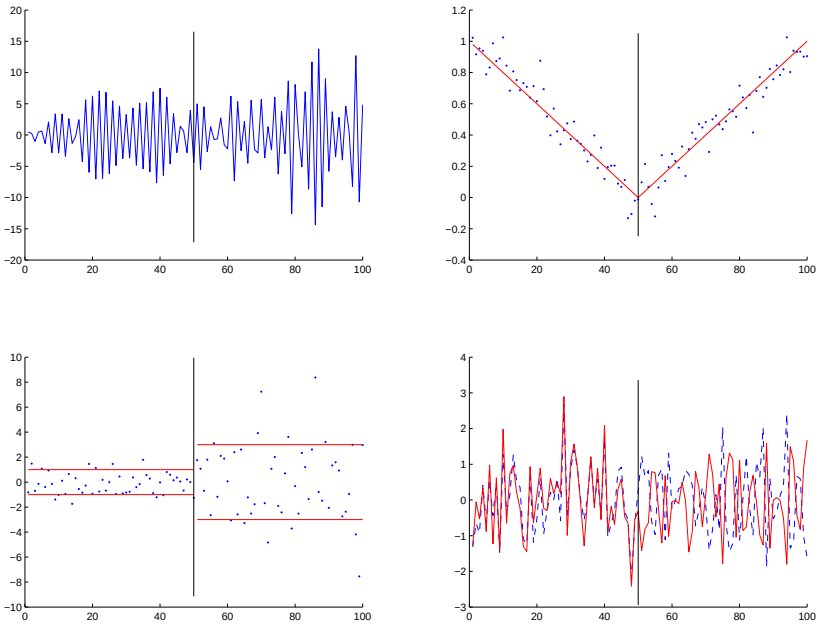

Figure 1.1: Examples show possible changes over successive segments. The top left panel shows changes on AR model orders. The top right panel shows changes on parameters. The bottom left panel shows changes on noise level. The bottom right panel shows changes on correlation between two series.

The aim of the change point problems is to make inference about the number and location of change points.

1.2 Related Works

Many works have been done by many researchers in different areas. This thesis is an extension based on the Fearnhead's work which is a special case of the Product Partition Models (PPM) (defined later) and using dynamic programming algorithms. Here I like to mention two approaches that are closely related to our works. The first approach is based on a different models Hidden Markov model (HMM), and the second approach is using a different algorithm Reversible Jump Monte Carlo Markov Chain.

1.2.1 Hidden Markov Models

A HMM is a statistical model where system being modeled is assumed to be Markov process with hidden states. In a regular Markov model, the state is directly visible. In a HMM, the state is not directly visible, but variables influenced by the hidden states are visible. The challenge is to determine the hidden states from observed variables.
In change point problems, we view the change is due to change on hidden states. Hence by inference on all hidden states, we can segment data implicitly. In HMM, we need to fix the number of hidden states, and often the number of state cannot be too large.

1.2.2 Reversible Jump MCMC

Reversible jump is a MCMC algorithm which has been extensively used in the change point problems. It starts by a initial set of change points. At each step, it can make the following three kinds of moves:

- Death move to delete a change point or merge two consecutive segments,
- Birth move to add a new change point or split a segment into two,
- Update move to shift the position of a change point

Each step, we will accept a move based on a probability calculated by the Metropolis-Hastings algorithm. We run until the chain converges. Then we can find out the posterior probability of the number and position of change points. The advantage of reversible jump MCMC is that it can handle a large family of distributions even when we only know the distribution up to a normalized constant. The disadvantages are slow and difficult to diagnose convergence of the chain.

1.3 Contribution

Our contributions of the thesis are:

- We extend Fearnhead's work to the case of multiple dimensional series which allows us to detect changes on correlation structures, as well as changes on means, variance, etc.
- We further model the correlation structures using Gaussian graphical models which allows us to estimate the changing topology of dependencies among series, in addition to detecting change points.
- We illustrate the algorithms by applying them to some synthetic and real data sets.

1.4 Thesis Outline

The remaining of the thesis is organized as follows. In Chapter 2, we will review Fearnhead's work in one dimensional series in details, and provide experimental results on some synthetic and real data sets. In Chapter 3, we will show how to extend Fearnhead's work in multiple dimensional series and how to use Gaussian graphical models to model and learn correlation structures. We will also provide experimental results on some synthetic and real data sets. Finally, our conclusions are stated in Chapter 4, along with a number of suggestions for future works.

Chapter 2

One Dimensional Time Series

Let's consider the change point problems with the following conditional independence property: given the position of a change point, the data before that change point is independent of the data after the change point. Then these models are exactly the Product Partition Models (PPM) in one dimension [3, 4, 11]. Here a dataset $Y_{1:N}$ is partitioned into K partitions, where the number of the partitions K is unknown. The data on the k-th segment Y^k is assumed to be independent with the data on the other segments given a set of parameters θ_k for that partition . Hence given the segmentation (partition) $S_{1:K}$, we can write

$$P(Y_{1:N}|K, S_{1:K}, \theta_{1:K}) = \prod_{k=1}^{K} P(Y^k|\theta_k) \tag{2.1}$$

Fearnhead [13, 14, 16] proposed three algorithms (offline, online exact and online approximate) to solve the change point problems under PPM. They all first calculate the joint posterior distribution of the number and positions of change points $P(K, S_{1:K}|Y_{1:N})$ using dynamic programming, then sample change points from this posterior distribution by perfect sampling [22].

$$K, S_{1:K} \sim P(K, S_{1:K}|Y_{1:N}) \tag{2.2}$$

After sampling change points, making inference on models and their parameters over segments is straight forward.

The offline algorithm and online exact algorithm run in $O(N^2)$, and the online approximate algorithm runs in $O(N)$.

2.1 Prior on Change Point Location and Number

To express the uncertainty of the number and position of change points, we put the following prior distribution on change points.

Let's assume that the change points occur at discrete time points and we model the change point positions by a Markov process. Let the transition probabilities of this Markov process be the following,

$$P(\text{next change point at } t|\text{change point at } s) = g(|t-s|) \tag{2.3}$$

We assume (2.3) only depends on the distance between two change points. Also we let the probability mass function for the distance between two successive change points s and t be $g(|t-s|)$. Furthermore, we define the cumulative distribution function for the distance as following,

$$G(l) \quad = \quad \sum_{i=1}^{l} g(i) \tag{2.4}$$

and assume that $g()$ is also the probability mass function for the position of the first change point. In general, $g()$ can be any arbitrary probability mass function with the domain over $1, 2, \cdots, N-1$. Then $g()$ and $G()$ imply a prior distribution on the number and positions of change points.

For example, if we use the Geometric distribution as $g()$, then our model implies a Binomial distribution for the number of change points and a Uniform distribution for the locations of change points. To see that, let's suppose there are N data points and we use a Geometric distribution with parameter λ. We denote $P(C_i = 1)$ as the probability of location i being a change point. By default, position 0 is always a change point. That is,

$$P(C_0 = 1) \quad = \quad 1 \tag{2.5}$$

First, we show that the distribution for the location of change points is Uniform by induction. That is, $\forall i = 1, \ldots, N$

$$P(C_i = 1) \quad = \quad \lambda \tag{2.6}$$

When $i = 1$, we only have one case: position 0 and 1 both are change points. Hence the length of the segment is 1. We have,

$$P(C_1 = 1) \quad = \quad g(1) = \lambda$$

Suppose $\forall\ i \leq k$, we have

$$P(C_i = 1) \quad = \quad \lambda \tag{2.7}$$

Now when $i = k+1$, conditioning on the last change point before position $k+1$, we have,

$$P(C_{k+1} = 1) \quad = \quad \sum_{j=0}^{k} P(C_{k+1} = 1 | C_j = 1) P(C_j = 1) \tag{2.8}$$

where

$$P(C_{k+1} = 1 | C_j = 1) \quad = \quad g(k+1-j) = \lambda(1-\lambda)^{k-j} \tag{2.9}$$

By (2.5), (2.7) and (2.9), (2.8) becomes,

$$\begin{aligned} P(C_{k+1} = 1) \quad &= \quad P(C_{k+1} = 1 | C_0 = 1) P(C_0 = 1) + \sum_{j=1}^{k} P(C_{k+1} = 1 | C_j = 1) P(C_j = 1) \\ (\text{let } t = k - j) \quad &= \quad \lambda(1-\lambda)^k + \sum_{j=1}^{k} \lambda(1-\lambda)^{k-j}\lambda = \lambda(1-\lambda)^k + \lambda^2 \sum_{t=0}^{k-1} (1-\lambda)^t \\ &= \quad \lambda(1-\lambda)^k + \lambda^2 \frac{1-(1-\lambda)^k}{1-(1-\lambda)} = \lambda(1-\lambda)^k + \lambda(1-(1-\lambda)^k) \\ &= \quad \lambda \end{aligned} \tag{2.10}$$

By induction, this proves (2.6). Next we show the number of change points follows Binomial distribution.

Let's consider each position as a trial with two outcomes, either being a change point or not. By (2.6), we know the probability of being a change point is the same. Then we only need to show each trial is independent. That is, $\forall i, j = 1, \ldots, N$ and $i \neq j$,

$$P(C_i = 1) \quad = \quad P(C_i = 1 | C_j = 1) = \lambda \tag{2.11}$$

When $i < j$, it is true by default, since future will not change history. When $i > j$, we show it by induction on j.

When $j = i-1$, we only have one case: position j and i both are change points. Hence the length of the segment is 1. We have,

$$P(C_i = 1|C_{i-1} = 1) \quad = \quad g(1) = \lambda$$

Suppose $\forall\ j \geq k$, we have

$$P(C_i = 1|C_j = 1) \quad = \quad \lambda \tag{2.12}$$

Now when $j = k-1$, conditioning on the next change point after position $k-1$, we have,

$$P(C_i = 1|C_{k-1} = 1) \quad = \quad \sum_{t=k}^{i} P(C_i = 1|C_t = 1, C_{k-1} = 1)P(C_t = 1|C_{k-1} = 1) \tag{2.13}$$

where

$$\begin{aligned} P(C_i = 1|C_t = 1, C_{k-1} = 1) \quad &= \quad P(C_i = 1|C_t = 1) = \begin{cases} \lambda & \text{if } t < i \\ 1 & \text{if } t = i \end{cases} \\ P(C_t = 1|C_{k-1} = 1) \quad &= \quad g(t-k+1) = \lambda(1-\lambda)^{t-k} \end{aligned} \tag{2.14}$$

Hence (2.13) becomes,

$$\begin{aligned} P(C_i = 1|C_{k-1} = 1) \quad &= \quad \sum_{t=k}^{i-1} \lambda\lambda(1-\lambda)^{t-k} + \lambda(1-\lambda)^{i-k} \\ \text{(let s = t-k)} \quad &= \quad \lambda^2 \sum_{s=0}^{i-k-1} (1-\lambda)^s + \lambda(1-\lambda)^{i-k} = \lambda^2 \frac{1-(1-\lambda)^{i-k}}{1-(1-\lambda)} + \lambda(1-\lambda)^{i-k} \\ &= \quad \lambda(1-(1-\lambda)^{i-k}) + \lambda(1-\lambda)^{i-k} = \lambda \end{aligned} \tag{2.15}$$

By induction, this proves (2.11). Hence the number of change points follows Binomial distribution.

2.2 Likelihood Functions for Data in Each Partition

If we assume that s and t are two successive change points, then data $Y_{s+1:t}$ forms one segment. We use the likelihood function $P(Y_{s+1:t})$ to evaluate how good the data $Y_{s+1:t}$ can be fit in one segment. Here there are two levels of uncertainty.

The first is the uncertainty of model types. There are many possible candidate models and we certainly cannot enumerate all of them. As a result, we will only consider a finite set of models. For example, polynomial regression models up to order 2 or autoregressive models up to order 3. Then if we put a prior distribution $\pi()$ on the set of models, and define $P(Y_{s+1:t}|q)$ as the likelihood function of $Y_{s+1:t}$ conditional on a model q, then we can evaluate $P(Y_{s+1:t})$ as following,

$$P(Y_{s+1:t}) = \sum_q P(Y_{s+1:t}|q)\pi(q) \tag{2.16}$$

The second is the uncertainty of model parameters. If the parameter on this segment conditional on model q is θ_q, and we put a prior distribution $\pi(\theta_q)$ on parameters, we can evaluate $P(Y_{s+1:t}|q)$ as following,

$$P(Y_{s+1:t}|q) = \int \prod_{i=s+1}^{t} f(Y_i|\theta_q, q)\pi(\theta_q|q)\, d\theta_q \tag{2.17}$$

We assume that $P(Y_{s+1:t}|q)$ can be efficiently calculated for all s, t and q, where $s < t$. In practice, this requires either conjugate priors on θ_q which allow us to work out the likelihood function analytically, or fast numerical routines which are able to evaluate the required integration. In general, for any data and models, as long as we can evaluate the likelihood function (2.17), we can use Fearnhead's algorithms. Our extensions are mainly based on this.

Now let's look at some examples.

2.2.1 Linear Regression Models

The linear regression models are one of the most widely used models. Here, we assume

$$Y_{s+1:t} = H\beta + \epsilon \tag{2.18}$$

where H is a $(t-s)$ by q matrix of basis functions, β is a q by 1 vector of regression parameters and ϵ is a $(t-s)$ by 1 vector of iid Gaussian noise with mean 0 and variance σ^2. Since we assume conjugate priors, σ^2 has an Inverse

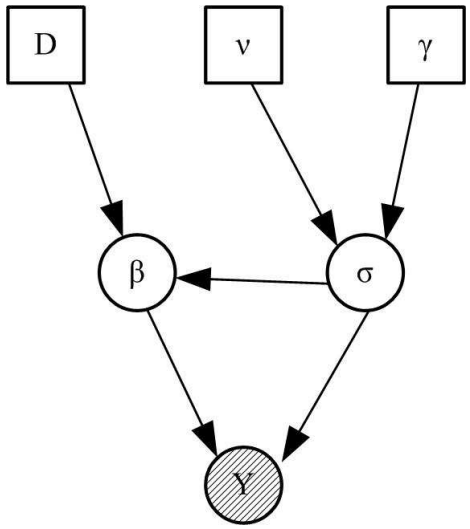

Figure 2.1: Graphical representation of Hierarchical structures of Linear Regression Models

Gamma distribution with parameters $\nu/2$ and $\gamma/2$, and the jth component of regression parameter β_j has a Gaussian distribution with mean 0 and variance $\sigma^2\delta_j^2$. This hierarchical model can be illustrated by Figure 2.1. For simplicity, we write $Y_{s+1:t}$ as Y and let $n = t - s$. And we have the following,

$$P(Y|\beta,\sigma^2) = \frac{1}{(2\pi)^{n/2}\sigma^n} exp\left(-\frac{1}{2\sigma^2}(Y-H\beta)^T I_n^{-1}(Y-H\beta)\right) \tag{2.19}$$

$$P(\beta|D,\sigma^2) = \frac{1}{(2\pi)^{q/2}\sigma^q|D|^{1/2}} exp\left(-\frac{1}{2\sigma^2}\beta^T D^{-1}\beta\right) \tag{2.20}$$

$$P(\sigma^2|\nu,\gamma) = \frac{(\gamma/2)^{\nu/2}}{\Gamma(\nu/2)}(\sigma^2)^{-\nu/2-1}exp\left(-\frac{\gamma}{2\sigma^2}\right) \quad (2.21)$$

where $D = diag(\delta_1^2,\ldots,\delta_q^2)$ and I_n is a n by n identity matrix.
By (2.17), we have,

$$\begin{aligned} P(Y_{s+1,t}|q) &= P(Y|D,\nu,\gamma) \\ &= \int\int P(Y,\beta,\sigma^2|D,\nu,\gamma)\,d\beta\,d\sigma^2 \\ &= \int\int P(Y|\beta,\sigma^2)P(\beta|D,\sigma^2)P(\sigma^2|\nu,\gamma)\,d\beta\,d\sigma^2 \end{aligned}$$

Multiplying (2.19) and (2.20), we have following,

$$\begin{aligned} P(Y|\beta,\sigma^2)P(\beta|D,\sigma^2) &\propto exp\left(-\frac{1}{2\sigma^2}((Y-H\beta)^T(Y-H\beta)+\beta^T D^{-1}\beta)\right) \\ &\propto exp\left(-\frac{1}{2\sigma^2}(Y^TY-2Y^TH\beta+\beta^TH^TH\beta+\beta^TD^{-1}\beta)\right) \end{aligned}$$

Now let

$$\begin{aligned} M &= (H^TH+D^{-1})^{-1} \\ P &= (I-HMH^T) \\ ||Y||_P^2 &= Y^TPY \\ (*) &= Y^TY-2Y^TH\beta+\beta^TH^TH\beta+\beta^TD^{-1}\beta \end{aligned}$$

Then

$$\begin{aligned} (*) &= \beta^T(H^TH+D^{-1})\beta-2Y^TH\beta+Y^TY \\ &= \beta^TM^{-1}\beta-2Y^THMM^{-1}\beta+Y^TY \\ &= \beta^TM^{-1}\beta-2Y^THMM^{-1}\beta+Y^THMM^{-1}M^TH^TY-Y^THMM^{-1}M^TH^TY+Y^TY \end{aligned}$$

Using fact $M = M^T$

$$\begin{aligned} (*) &= (\beta-MH^TY)^TM^{-1}(\beta-MH^TY)+Y^TY-Y^THMH^TY \\ &= (\beta-MH^TY)^TM^{-1}(\beta-MH^TY)+Y^TPY \\ &= (\beta-MH^TY)^TM^{-1}(\beta-MH^TY)+||Y||_P^2 \end{aligned}$$

Hence

$$P(Y|\beta,K)P(\beta|D,K) \quad \propto \quad exp\left(-\frac{1}{2\sigma^2}((\beta - MH^TY)^TM^{-1}(\beta - MH^TY) + ||Y||_P^2\right)$$

So the posterior for β is still Gaussian with mean MH^TY and variance σ^2M:

$$P(\beta|D,\sigma^2) \quad = \quad \frac{1}{(2\pi)^{q/2}\sigma^q|M|^{1/2}}exp\left(-\frac{1}{2\sigma^2}(\beta - MH^TY)^TM^{-1}(\beta - MH^TY)\right) \tag{2.22}$$

Then integrating out β, we have

$$\begin{aligned} P(Y|D,\sigma^2) &= \int P(Y|\beta,\sigma^2)P(\beta|D,\sigma^2)\,d\beta \\ &= \frac{1}{(2\pi)^{n/2}\sigma^n}\frac{(2\pi)^{q/2}\sigma^q|M|^{1/2}}{(2\pi)^{q/2}\sigma^q|D|^{1/2}}exp\left(-\frac{1}{2\sigma^2}||Y||_P^2\right) \\ &= \frac{1}{(2\pi)^{n/2}\sigma^n}\left(\frac{|M|}{|D|}\right)^{\frac{1}{2}}exp\left(-\frac{1}{2\sigma^2}||Y||_P^2\right) \end{aligned} \tag{2.23}$$

Now we multiply (2.21) and (2.23),

$$P(Y|D,\sigma^2)P(\sigma^2|\nu,\gamma) \quad \propto \quad (\sigma^2)^{-n/2-\nu/2-1}exp\left(-\frac{\gamma + ||Y||_P^2}{2\sigma^2}\right)$$

So the posterior for σ^2 is still Inverse Gamma with parameters $(n+\nu)/2$ and $(\gamma + ||Y||_P^2)/2$:

$$P(\sigma^2|\nu,\gamma) \quad = \quad \frac{((\gamma + ||Y||_P^2)/2)^{(n+\nu)/2}}{\Gamma((n+\nu)/2)}(\sigma^2)^{-(n+\nu)/2-1}exp\left(-\frac{\gamma + ||Y||_P^2}{2\sigma^2}\right) \tag{2.24}$$

Then integrating out σ^2, we have

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|D,\nu,\gamma) \\ &= \int P(Y|D,\sigma^2)P(\sigma^2|\nu,\gamma)\,d\sigma^2 \\ &= \left[\frac{1}{(2\pi)^{n/2}}\left(\frac{|M|}{|D|}\right)^{\frac{1}{2}}\right]\left[\frac{(\gamma/2)^{\nu/2}}{\Gamma(\nu/2)}\right]\left[\frac{\Gamma((n+\nu)/2)}{((\gamma + ||Y||_P^2)/2)^{(n+\nu)/2}}\right] \\ &= \pi^{-n/2}\left(\frac{|M|}{|D|}\right)^{\frac{1}{2}}\frac{(\gamma)^{\nu/2}}{(\gamma + ||Y||_P^2)^{(n+\nu)/2}}\frac{\Gamma((n+\nu)/2)}{\Gamma(\nu/2)} \end{aligned} \tag{2.25}$$

In implementation, we rewrite (2.25) in log space as following,

$$\begin{aligned} log(P(Y_{s+1:t}|q)) &= -\frac{n}{2}log(\pi) - \frac{1}{2}(log|M| - log|D|) + \frac{\nu}{2}log(\gamma) - \frac{n+\nu}{2}log(\gamma + ||Y||_P^2) \\ &+ log(\Gamma((n+\nu)/2)) - log(\Gamma(\nu/2)) \end{aligned} \tag{2.26}$$

To speed up, $-\frac{n}{2}log(\pi)$, $\frac{\nu}{2}log(\gamma)$, $-\frac{1}{2}log|D|$, $log(\Gamma((n+\nu)/2))$ and $log(\Gamma(\nu/2))$ can be pre-computed. At each iteration, M and $||Y||_P^2$ can be computed by the following rank one update,

$$\begin{aligned} H_{1:i+1,:}^T H_{1:i+1,:} &= H_{1:i,:}^T H_{1:i,:} + H_{i+1,:}^T H_{i+1,:} \\ Y_{1:i+1}^T Y_{1:i+1} &= Y_{1:i}^T Y_{1:i} + Y_{i+1}^T Y_{i+1} \\ H_{1:i+1,:}^T Y_{1:i+1} &= H_{1:i,:}^T Y_{1:i} + H_{i+1,:}^T Y_{i+1} \end{aligned}$$

We use the following notations: if Y is a vector, then $Y_{s:t}$ denotes the entries from position s to t inclusive. If Y is a matrix, then $Y_{s:t,:}$ denotes the s-th row to the t-th row inclusive, and $Y_{:,s:t}$ denotes the s-th column to the t-th column inclusive.

2.2.2 Poisson-Gamma Models

In Poisson Models, each observation Y_i is a non-negative integer which follows Poisson distribution with parameter λ. With conjugate prior, λ follows a Gamma distribution with parameters α and β. This hierarchical model can be illustrated by Figure 2.2. Hence we have the following,

$$P(Y_i|\lambda) = \frac{e^{-\lambda}\lambda^{Y_i}}{Y_i!} \tag{2.27}$$

$$P(\lambda|\alpha,\beta) = \lambda^{(\alpha-1)}\frac{\beta^\alpha e^{-\beta\lambda}}{\Gamma(\alpha)} \tag{2.28}$$

For simplicity, we write $Y_{s+1:t}$ as Y and let $n = t - s$. By (2.17), we have,

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|\alpha,\beta) \\ &= \int P(Y,\lambda|\alpha,\beta)\, d\lambda \\ &= \int P(Y|\lambda)P(\lambda|\alpha,\beta)\, d\lambda \\ &= \int \left(\prod_i P(Y_i|\lambda)\right) P(\lambda|\alpha,\beta)\, d\lambda \end{aligned}$$

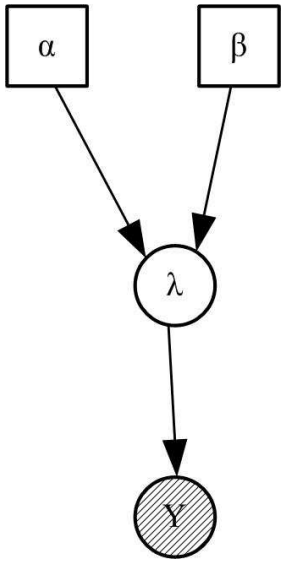

Figure 2.2: Graphical representation of Hierarchical structures of Poisson-Gamma Models. Note the Exponential-Gamma Models have the same Graphical representation of their Hierarchical structures.

By multiply (2.27) and (2.28), and let $SY = \sum_i Y_i$, we have,

$$P(Y|\lambda)P(\lambda|\alpha,\beta) \quad \propto \quad e^{-n\lambda-\beta\lambda}\lambda^{SY+\alpha-1}$$

So the posterior for λ is still Gamma with parameters $SY+\alpha$ and $n+\beta$:

$$P(\lambda|\alpha,\beta) \quad = \quad \lambda^{(SY+\alpha-1)}\frac{(n+\beta)^{SY+\alpha}e^{-(n+\beta)\lambda}}{\Gamma(SY+\alpha)} \tag{2.29}$$

Then integrating out λ, we have

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|\alpha,\beta) \\ &= \left(\prod_i \frac{1}{Y_i!}\right)\frac{\beta^\alpha}{\Gamma(\alpha)}\frac{\Gamma(SY+\alpha)}{(n+\beta)^{SY+\alpha}} \\ &= \left(\prod_i \frac{1}{Y_i!}\right)\frac{\Gamma(SY+\alpha)}{\Gamma(\alpha)}\frac{\beta^\alpha}{(n+\beta)^{SY+\alpha}} \end{aligned} \tag{2.30}$$

Similarly, in log space, we have the following,

$$log(P(Y_{s+1:t}|q)) \quad = \quad -\sum_i log(Y_i!) + log(\Gamma(SY+\alpha)) - log(\Gamma(\alpha)) + \alpha log(\beta)$$

$$- \quad (SY + \alpha) log(n + \beta) \tag{2.31}$$

Where $\sum_i log(Y_i!)$, $log(\Gamma(\alpha))$ and $\alpha log(\beta)$ can be pre-computed, and SY can use the following rank one update,

$$SY_{i+1} \quad = \quad SY_i + Y_{i+1}$$

2.2.3 Exponential-Gamma Models

In Exponential Models, each observation Y_i is a positive real number which follows Exponential distribution with parameter λ. With conjugate prior, λ follows a Gamma distribution with parameters α and β. This hierarchical model is the same as the one in Poisson-Gamma Models, which can be illustrated by Figure 2.2. Hence we have the following,

$$P(Y_i|\lambda) \quad = \quad \lambda e^{-\lambda Y_i} \tag{2.32}$$

$$P(\lambda|\alpha, \beta) \quad = \quad \lambda^{(\alpha-1)} \frac{\beta^\alpha e^{-\beta\lambda}}{\Gamma(\alpha)} \tag{2.33}$$

For simplicity, we write $Y_{s+1:t}$ as Y and let $n = t - s$. By (2.17), we have,

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|\alpha, \beta) \\ &= \int P(Y, \lambda|\alpha, \beta)\, d\lambda \\ &= \int P(Y|\lambda) P(\lambda|\alpha, \beta)\, d\lambda \\ &= \int \left(\prod_i P(Y_i|\lambda) \right) P(\lambda|\alpha, \beta)\, d\lambda \end{aligned}$$

By multiply (2.32) and (2.33), and let $SY = \sum_i Y_i$, we have,

$$P(Y|\lambda) P(\lambda|\alpha, \beta) \quad \propto \quad e^{-SY\lambda - \beta\lambda} \lambda^{n+\alpha-1}$$

So the posterior for λ is still Gamma with parameters $n + \alpha$ and $SY + \beta$:

$$P(\lambda|\alpha, \beta) \quad = \quad \lambda^{(n+\alpha-1)} \frac{(SY + \beta)^{n+\alpha} e^{-(SY+\beta)\lambda}}{\Gamma(n + \alpha)} \tag{2.34}$$

Then integrating out λ, we have

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|\alpha, \beta) \\ &= \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{\Gamma(n + \alpha)}{(SY + \beta)^{n+\alpha}} \end{aligned} \tag{2.35}$$

Similarly, in log space, we have the following,

$$log(P(Y_{s+1:t}|q)) \quad = \quad \alpha log(\beta) - log(\Gamma(\alpha)) + log(\Gamma(n+\alpha)) - (n+\alpha)log(SY+\beta) \tag{2.36}$$

Where $log(\Gamma(\alpha))$ and $\alpha log(\beta)$ can be pre-computed, and SY can use the rank one update as mentioned earlier.

2.3 Basis Functions

In the linear regression model (2.18), we have a basis function H.
Some common basis functions are the following.

2.3.1 Polynomial Basis

Polynomial model of order r is defined as following,

$$Y_i \quad = \quad \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \cdots + \beta_r X_i^r \tag{2.37}$$

Hence the polynomial basis H is following,

$$H_{i:j} \quad = \quad \begin{pmatrix} 1 & X_i & X_i^2 & \cdots & X_i^r \\ 1 & X_{i+1} & X_{i+1}^2 & \cdots & X_{i+1}^r \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_j & X_j^2 & \cdots & X_j^r \end{pmatrix} \tag{2.38}$$

where $X_i = \frac{i}{N}$.

2.3.2 Autoregressive Basis

Autoregressive model of order r is defined as following,

$$Y_i \quad = \quad \beta_1 Y_{i-1} + \beta_2 Y_{i-2} + \cdots + \beta_r Y_{i-r} \tag{2.39}$$

Hence the autoregressive basis H is following,

$$H_{i:j} = \begin{pmatrix} Y_{i-1} & Y_{i-2} & \cdots & Y_{i-r} \\ Y_i & Y_{i-1} & \cdots & Y_{i-r+1} \\ \vdots & \vdots & \ddots & \vdots \\ Y_{j-1} & Y_{j-2} & \cdots & Y_{j-r} \end{pmatrix} \tag{2.40}$$

There are many other possible basis functions as well (eg, Fourier Basis). Here we just want to point out that the basis function does provide a way to extend Fearnhead's algorithms (eg, Kernel Basis).

2.4 Offline Algorithms

Now we review the offline algorithm in detail. By its name, we know it works in batch mode. It first recurses backward, then simulates change points forward.

2.4.1 Backward Recursion

Let's define for $s = 2, \ldots, N$,

$$Q(s) = Prob(Y_{s:N}|\text{location } s-1 \text{ is a change point}) \tag{2.41}$$

and $Q(1) = Prob(Y_{1:N})$ since location 0 is a change point by default.
Offline algorithm will calculate $Q(s)$ for all $s = 1, 2, \cdots, N$.
When $s = N$

$$Q(N) = \sum_q P(Y_{N:N}|q)\pi(q) \tag{2.42}$$

When $s < N$

$$\begin{aligned} Q(s) &= \sum_{t=s}^{N-1} \sum_q P(Y_{s:t}|q)\pi(q)Q(t+1)g(t-s+1) \\ &+ \sum_q P(Y_{s:N}|q)\pi(q)(1-G(N-s)) \end{aligned} \tag{2.43}$$

where $\pi(q)$ is the prior probability of model q and $g()$ and $G()$ are defined in (2.4).

The reason behind (2.43) is that (drop the explicit conditioning on a change point at $s-1$ for notational convenience)

$$\begin{aligned} Q(s) &= \sum_{t=s}^{N-1} P(Y_{s:N},\ \text{next change point is at } t) \\ &+ P(Y_{s:N},\ \text{no further change points}) \end{aligned} \tag{2.44}$$

For the first part we have,

$$\begin{aligned} & P(Y_{s:N},\ \text{next change point is at } t) \\ = \ & P(\text{next change point is at } t) P(Y_{s:t}, Y_{t+1:N} | \text{next change point is at } t) \\ = \ & g(t-s+1) P(Y_{s:t} | \text{s,t form a segment}) P(Y_{t+1:N} | \text{next change point is at } t) \\ = \ & \sum_q P(Y_{s:t}|q)\pi(q)Q(t+1)g(t-s+1) \end{aligned}$$

For the second part we have,

$$\begin{aligned} & P(Y_{s:N},\ \text{no further change points}) \\ = \ & P(Y_{s:N} | \text{s,N form a segment}) P(\text{the length of segment} > N-s) \\ = \ & \sum_q P(Y_{s:N}|q)\pi(q)(1-G(N-s)) \end{aligned}$$

We will calculate $Q(s)$ backward for $s = N, \cdots, 1$ by (2.42) and (2.43). Since s is from N to 1, and at step s we will compute sum over t where t is from s to N, the whole algorithm runs in $O(N^2)$.

2.4.2 Forward Simulation

After we calculate $Q(s)$ for all $s = 1, \ldots, N$, we can simulate all change points forward. To simulate one realisation, we do the following,

1. Set $\tau_0 = 0$, and $k = 0$.

2. Compute the posterior distribution of τ_{k+1} given τ_k as following,
   For $\tau_{k+1} = \tau_k + 1, \tau_k + 2, \cdots, N-1$,

   $$P(\tau_{k+1}|\tau_k, Y_{1:N}) = \sum_q P(Y_{\tau_k+1:\tau_{k+1}}|q)\pi(q)Q(\tau_{k+1}+1)g(\tau_{k+1}-\tau_k)/Q(\tau_k+1)$$

For $\tau_{k+1} = N$, which means no further change points

$$P(\tau_{k+1}|\tau_k, Y_{1:N}) = \sum_q P(Y_{\tau_k+1:N}|q)\pi(q)(1 - G(N - \tau_k))/Q(\tau_k + 1)$$

3. Simulated τ_{k+1} from $P(\tau_{k+1}|\tau_k, Y_{1:N})$, and set $k = k + 1$.

4. If $\tau_k < N$ return to step (2); otherwise output the set of simulated change points, $\tau_1, \tau_2, \ldots, \tau_k$.

2.5 Online Algorithms

Now we discuss the online algorithms which have two versions (exact and approximate). Both work in online mode, and in the same way that first calculate recursion forward, then simulate change points backward.

We first review the online exact algorithm in detail. Let's introduce C_t as a state at time t, which is defined to be the time of the most recent change point prior to t. If $C_t = 0$, then there is no change point before time t. Then the state C_t can take values $0, 1, \ldots, t-1$, and $C_1, C_2, \ldots, C_t$ satisfy Markov property since we assume conditional independence over segments. Conditional on $C_{t-1} = j$, either $t-1$ is not a change point, which leads to $C_t = j$, or $t-1$ is a change point, which leads to $C_t = t-1$. Hence the transition probabilities of this Markov Chain can be calculated as following,

$$TP(C_t = j|C_{t-1} = i) = \begin{cases} \frac{1-G(t-i-1)}{1-G(t-i-2)} & \text{if } j = i \\ \frac{G(t-i-1)-G(t-i-2)}{1-G(t-i-2)} & \text{if } j = t-1 \\ 0 & \text{otherwise} \end{cases} \quad (2.45)$$

where G is defined in (2.4).

The reason behind (2.45) is : given $C_{t-1} = i$, we know there is no change point between i and $t-2$. This also means that the length of segment must be longer than $t-2-i$. At the same time, $G(n)$ means the cumulative probability of the distance between two consecutive change points no more than n, which can also be considered as the cumulative probability of the length of segment no more

than n. Hence We have $P(C_{t-1} = i) = 1 - G(t-2-i)$. Then if $j = i$, we have $P(C_t = i) = 1 - G(t-1-i)$. If $j = t-1$, which means there is a change point at $t-1$, and by definition of $g()$, the length of two change points is $t-1-i$, and this is just $g(t-1-i)$, which is also $G(t-i-1) - G(t-i-2)$.
If we use a Geometric distribution as $g()$, then $g(i) = (1-\lambda)^{i-1}\lambda$ and $G(i) = 1-(1-\lambda)^i$. Now if $j = i$,

$$\begin{aligned} TP(C_t = j|C_{t-1} = i) &= \frac{1-G(t-i-1)}{1-G(t-i-2)} = \frac{(1-\lambda)^{t-i-1}}{(1-\lambda)^{t-i-2}} \\ &= 1-\lambda \end{aligned}$$

And if $j = t-1$

$$\begin{aligned} TP(C_t = j|C_{t-1} = i) &= \frac{G(t-i-1) - G(t-i-2)}{1-G(t-i-2)} = \frac{(1-\lambda)^{t-i-2} - (1-\lambda)^{t-i-1}}{(1-\lambda)^{t-i-2}} \\ &= \lambda \end{aligned}$$

Hence (2.45) becomes,

$$TP(C_t = j|C_{t-1} = i) = \begin{cases} 1-\lambda & \text{if } j = i \\ \lambda & \text{if } j = t-1 \\ 0 & \text{otherwise} \end{cases}$$

where λ is the parameter of the Geometric distribution.

2.5.1 Forward Recursion

Let's define the filtering density $P(C_t = j|Y_{1:t})$ as: given data $Y_{1:t}$, the probability of the last change point is at position j. Online algorithm will compute $P(C_t = j|Y_{1:t})$ for all t, j, such that $t = 1, 2, \ldots, N$ and $j = 0, 1, \ldots, t-1$.
When $t = 1$, we have $j = 0$, hence

$$P(C_1 = 0|Y_{1:1}) = 1 \tag{2.46}$$

When $t > 1$, by standard filtering recursions, we have

$$\begin{aligned} P(C_t = j|Y_{1:t}) &= P(C_t = j|Y_t, Y_{1:t-1}) \\ &= \frac{P(C_t = j, Y_t|Y_{1:t-1})}{P(Y_t|Y_{1:t-1})} \end{aligned}$$

$$\begin{aligned} &= \frac{P(Y_t|C_t=j,Y_{1:t-1})P(C_t=j|Y_{1:t-1})}{P(Y_t|Y_{1:t-1})} \\ &\propto P(Y_t|C_t=j,Y_{1:t-1})P(C_t=j|Y_{1:t-1}) \end{aligned} \tag{2.47}$$

and

$$P(C_t=j|Y_{1:t-1}) = \sum_{i=0}^{t-2} TP(C_t=j|C_{t-1}=i)P(C_{t-1}=i|Y_{1:t-1}) \tag{2.48}$$

If we define $w_t^j = P(Y_t|C_t=j,Y_{1:t-1})$, and with (2.45), we have

$$P(C_t=j|Y_{1:t}) \propto \begin{cases} w_t^j \frac{1-G(t-i-1)}{1-G(t-i-2)} P(C_{t-1}=j|Y_{1:t-1}) & \text{if } j < t-1 \\ w_t^t \sum_{i=0}^{t-2} \left(\frac{G(t-i-1)-G(t-i-2)}{1-G(t-i-2)} P(C_{t-1}=i|Y_{1:t-1}) \right) & \text{if } j = t-1 \end{cases} \tag{2.49}$$

Now w_t^j can be calculated as following.

When $j < t-1$,

$$\begin{aligned} w_t^j &= \frac{P(Y_{j+1:t}|C_t=j)}{P(Y_{j+1:t-1}|C_t=j)} \\ &= \frac{\sum_q P(Y_{j+1:t}|q)\pi(q)}{\sum_q P(Y_{j+1:t-1}|q)\pi(q)} \end{aligned}$$

When $j = t-1$,

$$w_t^t = \sum_q P(Y_{t:t}|q)\pi(q) \tag{2.50}$$

where $\pi(q)$ is the prior probability of model q.

We will calculate $P(C_t|Y_{1:t})$ forward for $t = 1, \cdots, N$ by (2.46) and (2.49). Since t is from 1 to N, and at step t we will compute $C_t = j$ where j is from 0 to $t-1$, the whole algorithm runs in $O(N^2)$.

2.5.2 Backward Simulation

After we calculate the filtering densities $P(C_t|Y_{1:t})$ for all $t = 1, \ldots, N$, we can use idea in [9] to simulate all change points backward. To simulate one realisation from this joint density, we do the following,

1. Set $\tau_0 = N$, and $k = 0$.

2. Simulated τ_{k+1} from the filtering density $P(C_{\tau_k}|Y_{1:\tau_k})$, and set $k = k+1$.

3. If $\tau_k > 0$ return to step (2); otherwise output the set of simulated change points backward, $\tau_{k-1}, \tau_{k-2}, \ldots, \tau_1$.

2.5.3 Viterbi Algorithms

We can also obtain an online Viterbi algorithm for calculating the maximum a posterior (MAP) estimate of positions of change points and model parameters for each segment as following. We define M_i to be the event that given a change point at time i, the MAP choice of change points and model parameters occurs prior to time i.

For $t = 1, \ldots, n$, and $i = 0, \ldots, t-1$, and all q,

$$P_t(i,q) \quad = \quad P(C_t = i, modelq, M_i, Y_{1:t})$$

and

$$P_t^{MAP} \quad = \quad P(\text{change point at t}, M_t, Y_{1:t})$$

Then we can compute $P_t(i,q)$ using the following:

$$P_t(i,q) \quad = \quad (1 - G(t-j-1))P(Y_{j+1:t}|q)\pi(q)P_j^{MAP} \tag{2.51}$$

and

$$P_t^{MAP} \quad = \quad max_{j,q}\frac{P_t(j,q)g(t-j)}{1-G(t-j-1)} \tag{2.52}$$

At time t, the MAP estimates of C_t and the current model parameters are given respectively by the values of j and q which maximize $P_t(i,q)$. Given a MAP estimate of C_t, we can then calculate the MAP estimates of the change point prior to C_t and the model parameters of that segment by the value of j and q that maximized the right hand side of (2.52). This procedure can be repeated to find the MAP estimates of all change points and model parameters.

2.5.4 Approximate Algorithm

Now we look at the online approximate algorithm. The online approximate algorithm works in almost same way as the online exact algorithm. The only difference is the following. At step t, online exact algorithm stores the complete posterior distribution $P(C_t = j|Y_{1:t})$ for $j = 0, 1, \cdots, t-1$. We approximate $P(C_t = j|Y_{1:t})$ by a discrete distribution with fewer points. This approximate distribution can be described by a set of support points. We call them particles. We impose a maximum number (eg, M) of particles to be stored at each step. Whenever we have M particles, we perform resampling to reduce the number of particles to L, where $L < M$. Hence the maximum computational cost per iteration is proportional to M. Since M is not dependent on the size of data N, the approximate algorithm runs in $O(N)$.

There are many ways to perform resampling [7, 12, 15, 16]. Here, we use the simplest one. We let $L = M - 1$, hence each step we just drop the particle that has the lowest support. We will show later that this approximation will not affect the accuracy of the result much, but runs much faster.

2.6 Implementation Issues

There are two issues that need to be mentioned in implementation.

The first issue is numerical stability. We perform all calculations in log space to prevent overflow and underflow. We also use the functions logsumexp and logdet. In all three algorithms, we introduce a threshold (eg, 1×10^{-30}). During recursions, whenever the quantity that we need to compute is less than the threshold, we will set it to be $-\infty$, hence it will not be computed in the next iteration. This will provide not only stable calculation but also speed up because we will calculate less terms in each iteration. At the end of each iteration, we will rescale the quantity to prevent underflow.

The second one is rank one update. This is very important in term of speed. The way to do rank one update depends on the likelihood function we use.

2.7 Experimental Results

Now we will show experimental results on some synthetic and real data sets.

2.7.1 Synthetic Data

We will look at two synthetic data sets.

The first one is called Blocks. It has been previously analyzed in [10, 13, 21]. It is a very simple data set, and the results are shown in Figure 2.3. Each segment is a piecewise constant model, with mean level shifting over segments. We set the hyper parameter $\nu = 2$ and $\gamma = 2$ on σ^2. The top panel shows the raw data with the true change points (red vertical line). The bottom three panels show the posterior probability of being change points at each position and the number of segments that are calculated (from top to bottom) by the offline, the online exact and the online approximate algorithms. We can see that the results are essentially the same.

The second data set is called AR1, and results are shown in Figure 2.4. Each segment is an autoregressive model. We set the hyper parameter $\nu = 2$ and $\gamma = 0.02$ on σ^2. The top panel shows the raw data with the true change points (red vertical line). The bottom three panels show the posterior probability of being change points at each position and the number of segments that are calculated (from top to bottom) by the offline, the online exact and the online approximate algorithms. Again, we see the results from three algorithms are essentially the same. Henceforth only use online approximate method.

2.7.2 British Coal Mining Disaster Data

Now let's look at a real data set which records British coal-mining disasters [17] by year during the 112 year period from 1851 to 1962. This is a well-known data set and is previously studied by many researchers [6, 14, 18, 25]. Here Y_i is the number of disasters in the i-th year, which follows Poisson distribution with parameter (rate) λ. Hence the natural conjugate prior is a Gamma distribution.

We set the hyper parameter $\alpha = 1.66$ and $\beta = 1$, since we let the prior mean of λ is equal to the empirical mean ($\frac{\alpha}{\beta} = \bar{Y} = 1.66$) and the prior strength β to be weak. Then we use Fearnhead's algorithms to analyse the data, and the results are shown in Figure 2.5. In top left panel, it shows the raw data as a Poisson sequence. The bottom left panel shows the posterior distribution on the number of segments. It shows the most probable number of segments is four. The bottom right panel shows the posterior distribution of being change points at each location. Since we have four segments, we will pick the most probable three change points at location 41, 84 and 102 which corresponding to year 1891, 1934 and 1952. Then in the up right panel, it shows the resulted segmentation (the red vertical line) and the posterior estimators of rate λ on each segment (the red horizontal line). On four segments, the posterior rates are roughly as following, 3, 1, 1.5 and 0.5.

2.7.3 Tiger Woods Data

Now let's look at another interesting example. In sports, when an individual player or team enjoys periods of good form, it, is called 'streakiness'. It is interesting to detect a streaky player or a streaky team in many sports [1, 5, 26] including baseball, basketball and golf. The opposite of a streaky competitor is a player or team with a constant rate of success over time. A streaky player is different, since the associated success rate is not constant over time. Streaky players might have a large number of success during one or more periods, with fewer or no successes during other periods. More streaky players tend to have more change points. Here we use Tiger Woods as an example.

Tiger Woods is one of the most famous golf players in the world, and the first ever to win all four professional major championships consecutively. Woods turned professional at the Great Milwaukee Open in August 29, 1996 and play 230 tournaments, winning 62 championships in the period September 1996 to December 2005. Let $Y_i = 1$ if Woods won the i-th tournament and $Y_i = 0$ otherwise. Then the data can be expressed as a Bernoulli sequence with

parameter (rate) λ. We put a conjugate Beta prior with hyper parameters $\alpha = 1$ and $\beta = 1$ on λ since this is the weakest prior. Then we perform Fearnhead's algorithms to analyse the data, and the results are shown in Figure 2.6. In top left panel, it shows the raw data as a Bernoulli sequence. The bottom left panel shows the posterior distribution on the number of segments. It shows the most probable number of segments is three. The bottom right panel shows the posterior distribution of being change points at each location. Since we have three segments, we will pick the most probable two change points at location 73 and 167 which corresponding to May 1999 and March 2003. Then in the up right panel, it shows the resulted segmentation (the red vertical line) and the posterior estimators of rate λ on each segment (the red horizontal line). We can see clearly that Tiger Woods's career from September 1996 to December 2005 can be splitted into three periods. Initially, from September 1996 to May 1999, he was an golf player with winning rate lower than 0.2. However, from May 1999 to March 2003, he was in his peak period with winning rate nearly 0.5. After March 2003 till December 2005, his winning rate was dropped back to lower than 0.2. The data comes from Tiger Woods's official website http://www.tigerwoods.com/, and is previous studied by [26]. In [26], the data is only from September 1996 to June 2001.

2.8 Choices of Hyperparameters

Hyperparameters λ is the rate of change points and can be set as following,

$$\lambda = \frac{\text{the total number of expected segments}}{\text{the total number of data points}} \tag{2.53}$$

For example, if there are 1000 data points and we expect there are 10 segments, then we will set $\lambda = 0.01$. If we increase λ, we will encourage more change points. We use the synthetic data Blocks as an example. Results obtained by the online approximate algorithms under different values of λ are shown in Figure 2.7. From top to bottom, the values of λ are: 0.5, 0.1, 0.01, 0.001 and 0.0002. We see when $\lambda = 0.5$ (this prior says the length of segment is

2, which is too short.), the result is clearly oversegmented. Under other values of λ, results are fine.

Different likelihood functions have different hyperparameters. In linear regression, we have hyperparameters ν and γ on the variance σ^2. Since ν represents the strength of prior, we normally set $\nu = 2$ such that it is a weak prior. Then we can set γ to reflect our belief on how large the variance will be within each segment. (Note: we parameterize Inverse-Gamma in term of $\frac{\nu}{2}$ and $\frac{\gamma}{2}$.) When $\nu = 2$, the mean does not exist. We use the mode $\frac{\gamma}{\nu+2}$ to set $\gamma = 4\sigma^2$, where σ^2 is expected variance within each segment. For example, if we believe the variance is 0.01, then we will set $\gamma = 0.04$. Now we show results from synthetic data Blocks and AR1 obtained by the online approximate algorithms under different values of γ in Figure 2.8 and Figure 2.9. From top to bottom, the values of γ are: 100, 20, 2, 0.2 and 0.04. We see that the data Blocks is very robust to the choices of γ. For the data AR1, we see when $\gamma = 100$, we only detect 3 change points instead of 5. For other values of γ, results are fine.

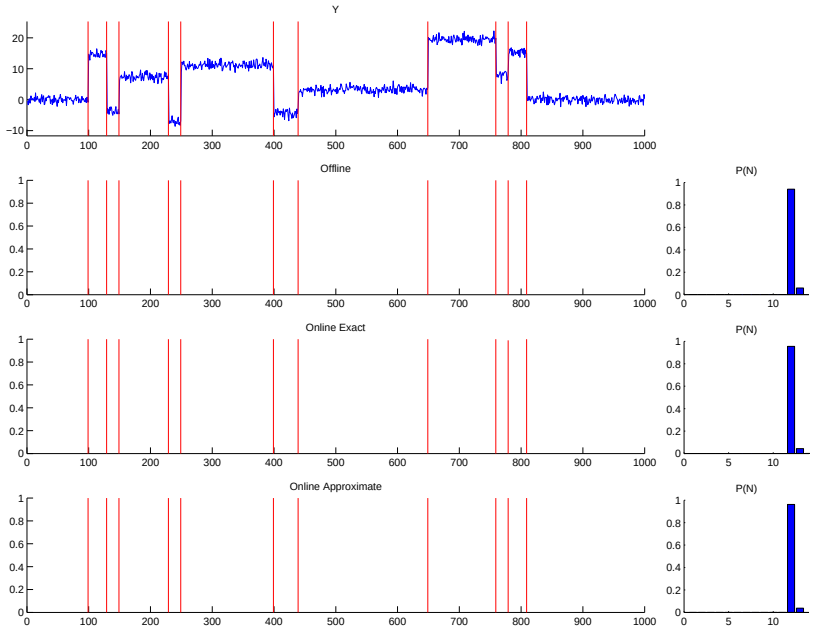

Figure 2.3: Results on synthetic data Blocks (1000 data points). The top panel is the Blocks data set with true change points. The rest are the posterior probability of being change points at each position and the number of segments calculated by (from top to bottom) the offline, the online exact and the online approximate algorithms. Results are generated by 'showBlocks'.

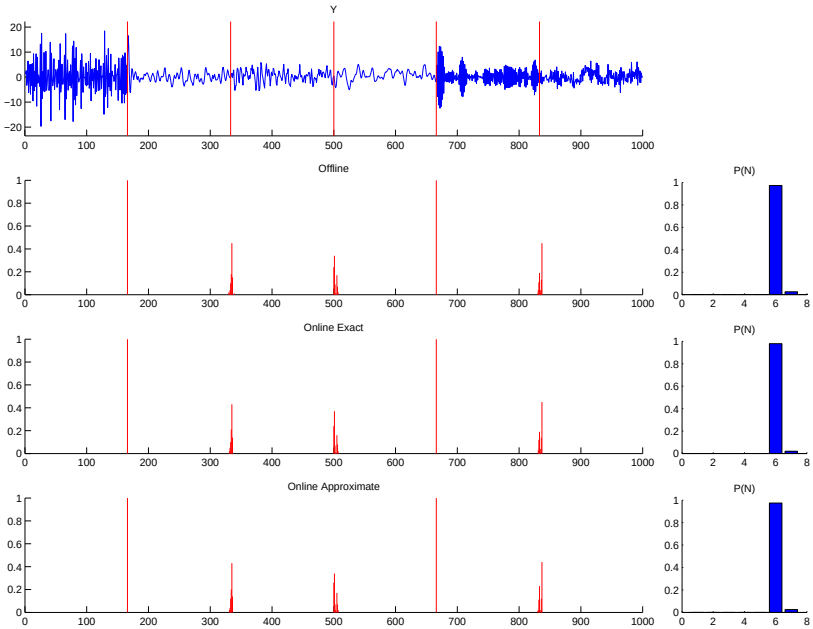

Figure 2.4: Results on synthetic data AR1 (1000 data points). The top panel is the AR1 data set with true change points. The rest are the posterior probability of being change points at each position and the number of segments calculated by (from top to bottom) the offline, the online exact and the online approximate algorithms. Results are generated by 'showAR1'.

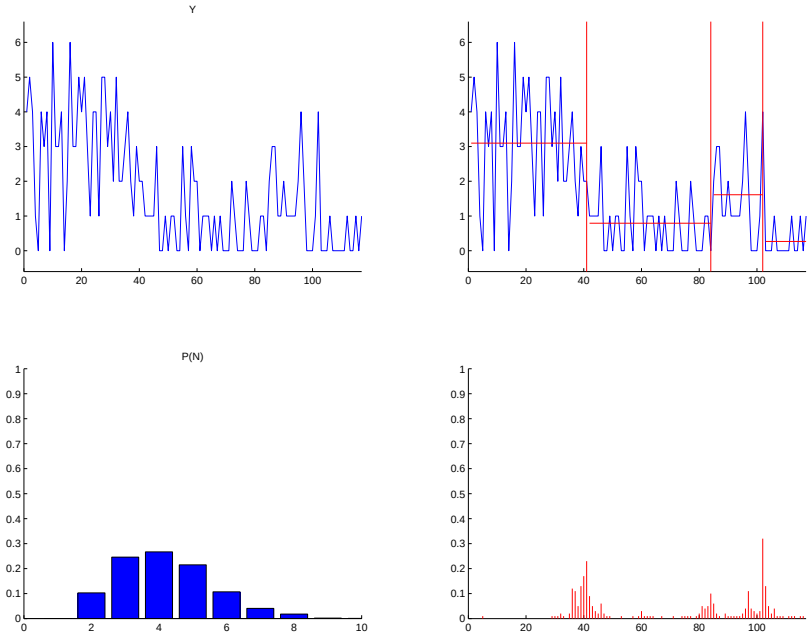

Figure 2.5: Results on Coal Mining Disaster data. The top left panel shows the raw data as a Poisson sequence. The bottom left panel shows the posterior distribution on the number of segments. The bottom right panel shows the posterior distribution of being change points at each position. The up right panel shows the resulted segmentation and the posterior estimators of rate λ on each segment. Results are generated by 'showCoalMining'.

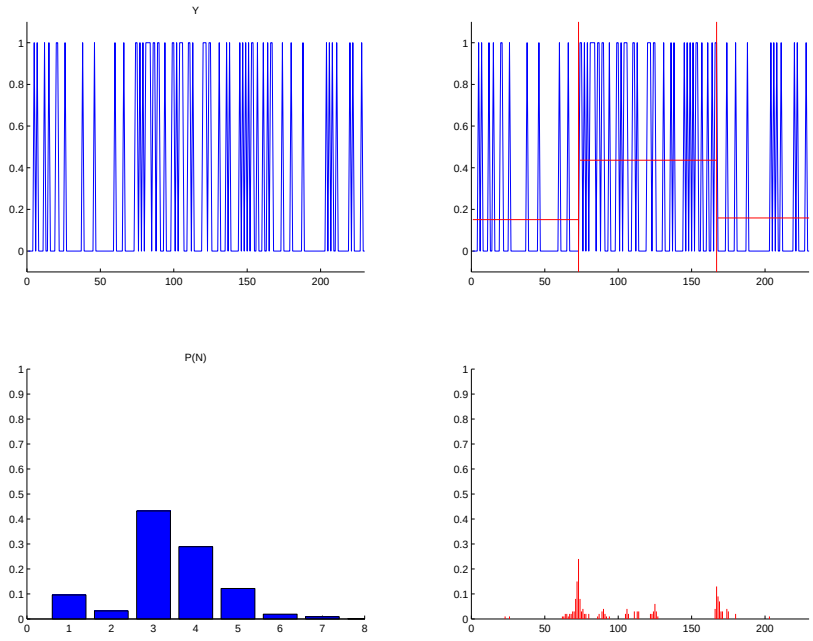

Figure 2.6: Results on Tiger Woods data. The top left panel shows the raw data as a Bernoulli sequence. The bottom left panel shows the posterior distribution on the number of segments. The bottom right panel shows the posterior distribution of being change points at each position. The up right panel shows the resulted segmentation and the posterior estimators of rate λ on each segment. Results are generated by 'showTigerWoods'. This data comes from Tiger Woods's official website http://www.tigerwoods.com/.

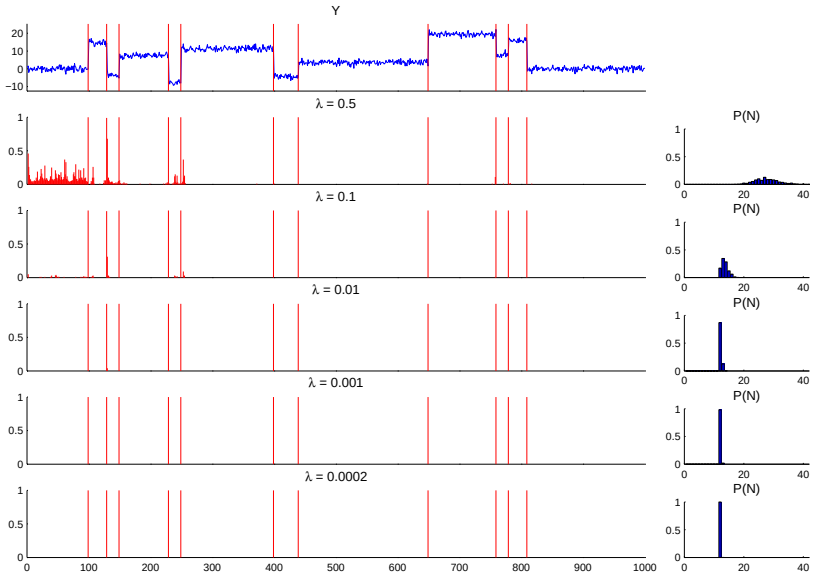

Figure 2.7: Results on synthetic data Blocks (1000 data points) under different values of hyperparameter λ. The top panel is the Blocks data set with true change points. The rest are the posterior probability of being change points at each position and the number of segments calculated by the online approximate algorithms under different values of λ. From top to bottom, the values of λ are: 0.5, 0.1, 0.01, 0.001 and 0.0002. Large value of λ will encourage more segments. Results are generated by 'showBlocksLambda'.

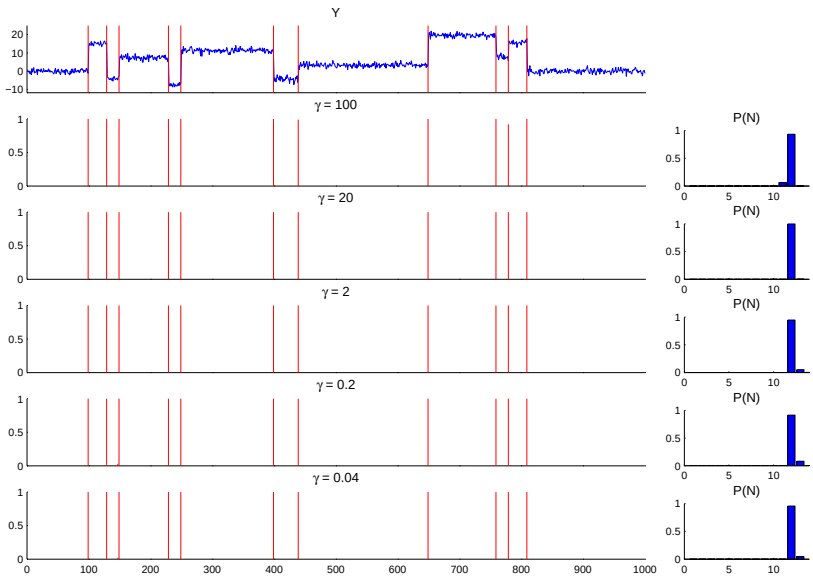

Figure 2.8: Results on synthetic data Blocks (1000 data points) under different values of hyperparameter γ. The top panel is the Blocks data set with true change points. The rest are the posterior probability of being change points at each position and the number of segments calculated by the online approximate algorithms under different values of γ. From top to bottom, the values of γ are: 100, 20, 2, 0.2 and 0.04. Large value of γ will allow higher variance in one segment, hence encourage less segments. Results are generated by 'showBlocks-Gamma'.

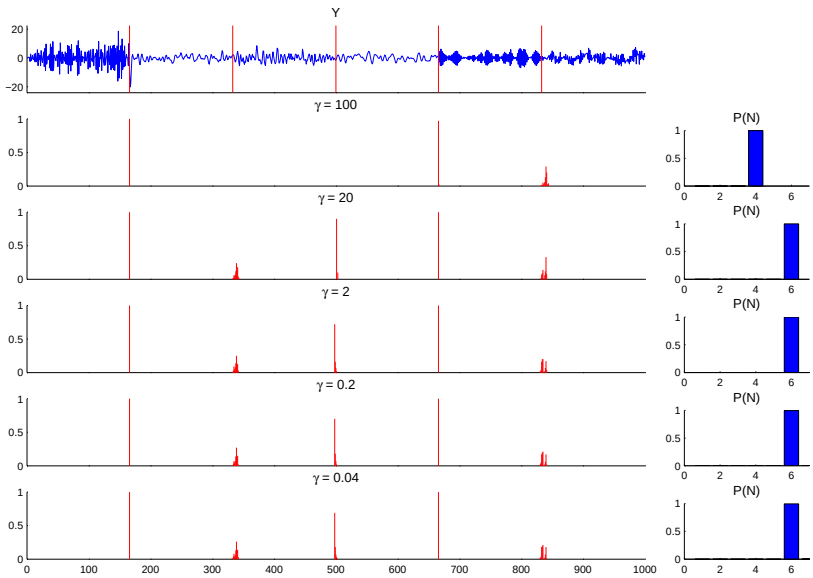

Figure 2.9: Results on synthetic data AR1 (1000 data points) under different values of hyperparameter γ. The top panel is the AR1 data set with true change points. The rest are the posterior probability of being change points at each position and the number of segments calculated by the online approximate algorithms under different values of γ. From top to bottom, the values of γ are: 100, 20, 2, 0.2 and 0.04. Large value of γ will allow higher variance within one segment, hence encourage less segments. Results are generated by 'showAR1Gamma'.

Chapter 3

Multiple Dimensional Time Series

We extend Fearnhead's algorithms to the case of multiple dimensional series. Now Y_i, the observation at time i, is a d-dimensional vector. Then we can detect change points based on changing correlation structure. We model the correlation structures using Gaussian graphical models. Hence we can estimate the changing topology of the dependencies among the series, in addition to detecting change points.

3.1 Independent Sequences

The first obvious extension is to assume that each dimension is independent. Under this assumption, the marginal likelihood function (2.16) can be written as the following product,

$$P(Y_{s+1:t}) \quad = \quad \prod_{j=1}^{d} P(Y^j_{s+1:t}) \tag{3.1}$$

where $P(Y^j_{s+1:t})$ is the marginal likelihood in the j-th dimension. Since now $Y^j_{s+1:t}$ is one dimension, $P(Y^j_{s+1:t})$ can be any likelihood function discussed in the previous chapter.

The independent model is simple to use. Even when independent assumption is not valid, it can be used as an approximate model similar to Naive Bayes when we cannot model the correlation structures among each dimension.

3.2 Dependent Sequences

Correlation structures only exist when we have multiple series. This is the main difference when we go from one dimension to multiple dimensions. Hence we should model correlation structures whenever we are able to do so.

3.2.1 A Motivating Example

Let's use the following example to illustrate the importance of modeling correlation structures. As shown in Figure 3.1, we have two series. The data on

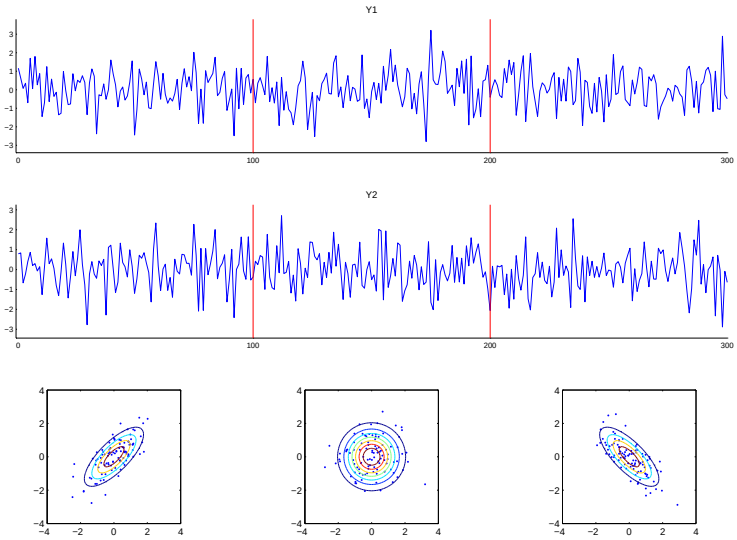

Figure 3.1: A simple example to show the importance of modeling changing correlation structures. We are unable to identify any change if we look at each series individually. However, their correlation structures are changed over segments(positive, independent, negative). Results are generated by 'plot2DExample'.

three segments are generated from the following Gaussian distributions. On the

k-th segment,

$$Y_k \sim N(0, \Sigma_k)$$

where $\Sigma_1 = \begin{bmatrix} 1 & 0.75 \\ 0.75 & 1 \end{bmatrix}, \Sigma_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \Sigma_3 = \begin{bmatrix} 1 & -0.75 \\ -0.75 & 1 \end{bmatrix}$

As a result, the marginal distribution on each dimension is, $N(0, 1)$, the same over all three segments. Hence if we look at each dimension individually, we are unable to identify any changes. However if we consider them jointly, then we find their correlation structures are changed. For example, in the first segment, they are positive correlated; in the second segment, they are independent; in the last segment, they are negative correlated.

3.2.2 Multivariate Linear Regression Models

For multivariate linear regression models, we know how to model correlation structures, and we can calculate the likelihood function (2.16) analytically.
On segment $Y_{s+1:t}$ we still have,

$$Y_{s+1:t} = H\beta + \epsilon \tag{3.2}$$

For simplicity, we write $Y_{s+1:t}$ as Y, and let $n = t - s$. Now Y is a n by d matrix of data, H is a n by q matrix of basis functions, β is a q by d matrix of regression parameters and ϵ is a n by d matrix of noise.
Here we need introduce the Matrix-Gaussian distribution. A random m by n matrix A is Matrix-Gaussian distributed with parameters M_A, V and W if the density function of A is

$$\begin{aligned} A &\sim N(M_A, V, W) \\ P(A) &= \frac{1}{(2\pi)^{mn/2}|V|^{n/2}|W|^{m/2}} exp\left(-\frac{1}{2} trace((A - M_A)^T V^{-1}(A - M_A)W^{-1})\right) \end{aligned}$$

where M is a m by n matrix representing the mean, V is a m by m matrix representing covariance among rows and W is a n by n matrix representing covariance among columns.

Similar as before we assume conjugate priors, ϵ has a Matrix-Gaussian distribution with mean 0 and covariance I_n and Σ, since we assume ϵ is independent across time(rows) but dependent across features(columns). And β has a Matrix-Gaussian distribution with mean 0 and covariance D and Σ. Finally covariance Σ has an Inverse-Wishart distribution with parameter N_0 and Σ_0. This hierarchical model can be illustrated by Figure 3.2.

Here $D = diag(\delta_1^2, \cdots, \delta_q^2)$ and I_n is a n by n identity matrix. Hence we have,

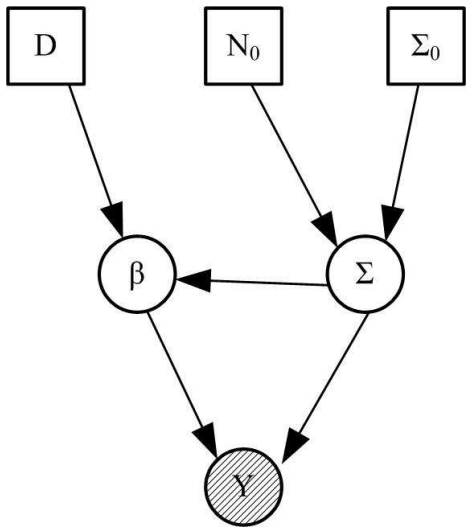

Figure 3.2: Graphical representation of Hierarchical structures of Multivariate Linear Regression Models

$$P(Y|\beta,\Sigma) = \frac{1}{(2\pi)^{nd/2}|I_n|^{d/2}|\Sigma|^{n/2}} exp\left(-\frac{1}{2}trace((Y-H\beta)^T I_n^{-1}(Y-H\beta)\Sigma^{-1})\right) \tag{3.3}$$

$$P(\beta|D,\Sigma) = \frac{1}{(2\pi)^{qd/2}|D|^{d/2}|\Sigma|^{q/2}} exp\left(-\frac{1}{2}trace(\beta^T D^{-1}\beta\Sigma^{-1})\right) \tag{3.4}$$

$$P(\Sigma|N_0,\Sigma_0) = \frac{|\Sigma_0|^{N_0/2}}{Z(N_0,d)2^{N_0 d/2}} \frac{1}{|\Sigma|^{(N_0+d+1)/2}} exp\left(-\frac{1}{2}trace(\Sigma_0\Sigma^{-1})\right) \tag{3.5}$$

where $N_0 \geq d$ and

$$Z(n,d) = \pi^{d(d-1)/4} \prod_{i=1}^{d} \Gamma((n+1-i)/2)$$

By (2.17), we have,

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|D, N_0, \Sigma_0) \\ &= \int\int P(Y, \beta, \Sigma|D, N_0, \Sigma_0)\, d\beta\, d\Sigma \\ &= \int\int P(Y|\beta, \Sigma) P(\beta|D, \Sigma) P(\Sigma|N_0, \Sigma_0)\, d\beta\, d\Sigma \end{aligned}$$

Now multiply (3.3) and (3.4),

$$\begin{aligned} P(Y|\beta, \Sigma)P(\beta|D, \Sigma) &\propto exp\left(-\frac{1}{2} trace(((Y - H\beta)^T(Y - H\beta) + \beta^T D^{-1}\beta)\Sigma^{-1})\right) \\ &\propto exp\left(-\frac{1}{2} trace((Y^TY - 2Y^TH\beta + \beta^T H^T H\beta + \beta^T D^{-1}\beta)\Sigma^{-1})\right) \end{aligned}$$

Now let

$$\begin{aligned} M &= (H^TH + D^{-1})^{-1} \\ P &= (I - HMH^T) \\ (*) &= Y^TY - 2Y^TH\beta + \beta^T H^T H\beta + \beta^T D^{-1}\beta \end{aligned}$$

Then

$$\begin{aligned} (*) &= \beta^T(H^TH + D^{-1})\beta - 2Y^TH\beta + Y^TY \\ &= \beta^T M^{-1}\beta - 2Y^THMM^{-1}\beta + Y^TY \\ &= \beta^T M^{-1}\beta - 2Y^THMM^{-1}\beta + Y^THMM^{-1}M^TH^TY - Y^THMM^{-1}M^TH^TY + Y^TY \end{aligned}$$

Using fact $M = M^T$

$$\begin{aligned} (*) &= (\beta - MH^TY)^T M^{-1}(\beta - MH^TY) + Y^TY - Y^THMH^TY \\ &= (\beta - MH^TY)^T M^{-1}(\beta - MH^TY) + Y^TPY \end{aligned}$$

Hence

$$P(Y|\beta, \Sigma)P(\beta|D, \Sigma) \propto exp\left(-\frac{1}{2} trace(((\beta - MH^TY)^T M^{-1}(\beta - MH^TY))\Sigma^{-1} + Y^TPY\Sigma^{-1})\right)$$

So the posterior for β is still Matrix-Gaussian with mean MH^TY and covariance M and W,

$$P(\beta|D, \Sigma) \sim N(MH^TY, M, \Sigma) \tag{3.6}$$

Then integrating out β, we have

$$\begin{aligned} P(Y|D,\Sigma) &= \int P(Y|\beta,\Sigma)P(\beta|D,\Sigma)\,d\beta \qquad (3.7)\\ &= \frac{1}{(2\pi)^{nd/2}|I_n|^{d/2}|\Sigma|^{n/2}}\frac{(2\pi)^{qd/2}|M|^{d/2}|\Sigma|^{q/2}}{(2\pi)^{qd/2}|D|^{d/2}|\Sigma|^{q/2}}exp(-\frac{1}{2}trace(Y^TPY\Sigma^{-1}))\\ &= \frac{1}{(2\pi)^{nd/2}|\Sigma|^{n/2}}\left(\frac{|M|}{|D|}\right)^{\frac{d}{2}}exp(-\frac{1}{2}trace(Y^TPY\Sigma^{-1})) \end{aligned}$$

Now multiply (3.5) and (3.7)

$$\begin{aligned} P(Y|D,\Sigma)P(\Sigma|N_0,\Sigma_0) &\propto \frac{1}{|\Sigma|^{n/2}|\Sigma|^{(N_0+d+1)/2}}exp(-\frac{1}{2}trace((Y^TPY+\Sigma_0)\Sigma^{-1}))\\ &\propto \frac{1}{|\Sigma|^{(n+N_0+d+1)/2}}exp(-\frac{1}{2}trace((Y^TPY+\Sigma_0)\Sigma^{-1})) \end{aligned}$$

So the posterior for Σ is still Inverse-Wishart with parameter $n+N_0$ and $Y^TPY+\Sigma_0$,

$$P(\Sigma) \sim IW(n+N_0, Y^TPY+\Sigma_0) \qquad (3.8)$$

Then integrating out Σ, we have

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|D,N_0,\Sigma_0)\\ &= \int P(Y|D,\Sigma)P(\Sigma|N_0,\Sigma_0)\,d\Sigma\\ &= \left(\frac{|M|}{|D|}\right)^{\frac{d}{2}}\frac{1}{(2\pi)^{nd/2}}\frac{|\Sigma_0|^{N_0/2}}{Z(N_0,d)2^{N_0d/2}}\frac{Z(n+N_0,d)2^{(n+N_0)d/2}}{|Y^TPY+\Sigma_0|^{(n+N_0)/2}}\\ &= \pi^{-nd/2}\left(\frac{|M|}{|D|}\right)^{\frac{d}{2}}\frac{|\Sigma_0|^{N_0/2}}{|Y^TPY+\Sigma_0|^{(n+N_0)/2}}\frac{Z(n+N_0,d)}{Z(N_0,d)} \qquad (3.9) \end{aligned}$$

When $d=1$ and by setting $N_0=\nu$ and $\Sigma_0=\gamma$, (2.25) is just a special case of (3.9), since $InverseWishart(\nu,\gamma) \equiv InverseGamma(\frac{\nu}{2},\frac{\gamma}{2})$. In implementation, we rewrite (3.9) in log space as following,

$$\begin{aligned} log(P(Y_{s+1:t}|q)) &= \frac{N_0}{2}log(|\Sigma_0|) - \frac{n+N_0}{2}log(|Y^TPY+\Sigma_0|) - \frac{d}{2}(log|M| - log|D|)\\ -\frac{nd}{2}log(\pi) &+ \sum_{i=1}^{d}log(\Gamma((n+N_0+1-i)/2)) - \sum_{i=1}^{d}log(\Gamma((N_0+1-i)/2)) \end{aligned} \qquad (3.10)$$

To speed up, $-\frac{nd}{2}log(\pi)$, $\frac{N_0}{2}log(|\Sigma_0|)$, $-\frac{d}{2}log|D|$, $\sum_{i=1}^{d} log(\Gamma((n+N_0+1-i)/2))$ and $\sum_{i=1}^{d} log(\Gamma((N_0+1-i)/2))$ can be pre-computed. At each iteration, M and Y^TPY can be computed by the following rank one update,

$$\begin{aligned} H_{1:i+1,:}^T H_{1:i+1,:} &= H_{1:i,:}^T H_{1:i,:} + H_{i+1,:}^T H_{i+1,:} \\ Y_{1:i+1,:}^T Y_{1:i+1,:} &= Y_{1:i,:}^T Y_{1:i,:} + Y_{i+1,:}^T Y_{i+1,:} \\ H_{1:i+1,:}^T Y_{1:i+1,:} &= H_{1:i,:}^T Y_{1:i,:} + H_{i+1,:}^T Y_{i+1,:} \end{aligned}$$

3.3 Gaussian Graphical Models

Gaussian graphical models are more general tools to model correlation structures, especially conditional independence structures. There are two kinds of graphs: directed and undirected. Here we use undirected graphs as examples, and we will talk about directed graphs later.

3.3.1 Structure Representation

In an undirected graph, each node represents a random variable. Define the precision matrix $K = \Sigma^{-1}$, which is the inverse of covariance matrix Σ. There is no edge between node i and node j if and only if $K_{ij} = 0$. As shown in Figure 3.3, there are four nodes. There is an edge between node 1 and node 2, since $K_{12} \neq 0$. ("X" denotes non-zero entries in the matrix.)

Then the independent model and the multivariate linear regression model are just two special cases of Gaussian graphical models. In independent model, K is a diagonal matrix, which represents the graph with all isolated nodes.

In multivariate linear regression model, K is a full matrix, which represents the graph with fully connected nodes.

Unlike independent models which are too simple, full covariance models could be too complex when the dimension d is large since the number of parameters needed by the models are $O(d^2)$. Gaussian graphical models provide a more flexible and better solution between these two extremes, especially when d becomes large, since in higher dimensions, sparse structures are more important.

$$\sum \xrightarrow{K=\sum^{-1}} K$$

$$K = \begin{bmatrix} X & X & 0 & X \\ X & X & 0 & 0 \\ 0 & 0 & X & 0 \\ X & 0 & 0 & X \end{bmatrix}$$

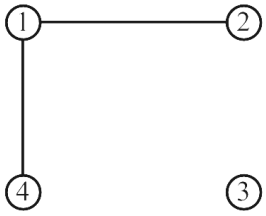

Figure 3.3: A simple example to show how to use a graph to represent correlation structures. We first compute precision matrix K. Then from K, if $K_{ij} = 0$, then there is no edge on from node i to node j. "X" represent non-zero entries in the matrix.

3.3.2 Sliding Windows and Structure Learning Methods

In order to use Fearnhead's algorithms, we need to do the following two things:

- generate a list of possible graph structures,
- compute the marginal likelihood given a graph structure.

For the first one, the number of all possible undirected graphs is $O(2^{d^2})$. Hence it is infeasible to try all possible graphs. We can only choose a subset of all graphs. How to choose them is a chicken and egg problem: if we knew the segmentation, then we could run a fast structure learning algorithm on each segment, but we need to know the structures in order to compute the segmentation. Hence we propose the following sliding window method to generate the

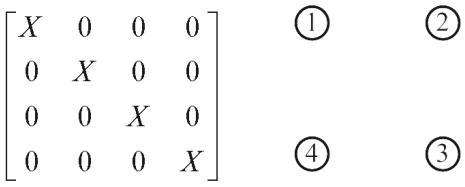

Figure 3.4: Gaussian graphical model of independent models.

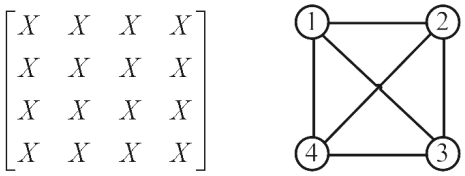

Figure 3.5: Gaussian graphical model of full models.

set of graphs. We slide a window of width $w = 0.2N$ across the data, shifting by $s = 0.1w$ at each step. This will generate a set of about 50 candidate segmentations. We can repeat this for different setting of w and s. Then we run a fast structure learning algorithm on each windowed segment, and sort resulting set of graphs by frequency of occurrence. Finally we pick the top $M = 20$ to form the set of candidate graphs. We hope this set will contain the true graph structures or at least the ones that are very similar.

As shown in Figure 3.6, suppose the vertical pink dot lines are true segmentations. When we run a sliding window inside of a true segment, (eg, the red window), we hope the structure we learn from this window is similar to the true structure of this segment. And when we shift the window one step, (eg, shifting to the blue window), if it is still inside of the same segment, we hope the structure we learn is the same or at least very similar to the one we learn in the previous window. Of course we will have the window that overlaps two segments (eg, the black window), then we know the structure we learn from this window will represent neither segments. However, this brings no harm since these "wrong" graph structures will later receive negligible posterior probabil-

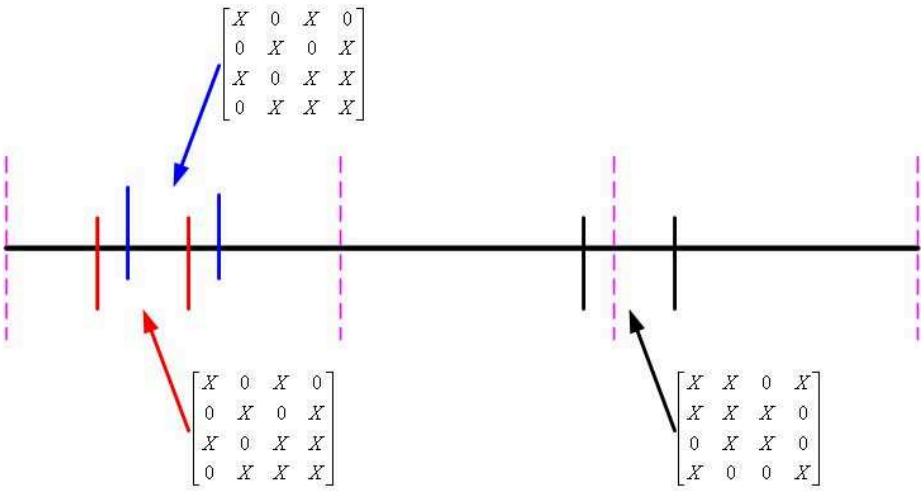

Figure 3.6: An example to show sliding window method.

ity. We can choose the number of the graphs we want to consider based on how fast we want the algorithms to run.

Now on each windowed segment, we need to learn the graph structure.

The simplest method is the thresholding method. It works as following,

1. compute the empirical covariance Σ,
2. compute the precision $K = \Sigma^{-1}$,
3. compute the partial correlation coefficients by $\rho_{ij} = \frac{K_{ij}}{\sqrt{K_{ii}K_{jj}}}$,
4. set edge $G_{ij} = 0$ if $|\rho_{ij}| < \theta$ for some threshold θ (eg, $\theta = 0.2$).

The thresholding method is simple and fast, but it may not give a good estimator. We can also use the shrinkage method discussed in [23] to get a better estimator of Σ, which helps regularize the problem when the segment is too short and d is large.

If we further pose the sparsity on the graph structure, we can use convex optimization techniques discussed in [2] to compute the MAP estimator for the precision K under a prior that encourage many entries to go to 0. We first form

the following problem,

$$\max_{\Sigma \succ 0} \quad log(det(K)) - trace(\Sigma K) - \rho ||K||_1 \tag{3.11}$$

where Σ is the empirical covariance, $||K||_1 = \Sigma_{ij}|K_{ij}|$, and $\rho > 0$ is the regularization parameter which controls the sparsity of the graph. Then we can solve (3.11) by block coordinate descent algorithms.
We summarize the overall algorithm as following,

1. **Input:** data $Y_{1:N}$, hyperparameters λ for change point rate and θ for likelihood functions, observation model obslik and parameter ρ for graph structure learning methods.
2. S = make overlapping segments from $Y_{1:N}$ by sliding windows
3. G = estimate graph structure $estG(Y_s, \rho) : s \in S$
4. **while** not converged **do**
5. $(K, S_{1:K})$ = segment($Y_{1:N}$, obslik, λ, θ)
6. $G = estG(Y_s, \rho) : s \in S_{1:K}$
7. **end while**
8. Inference model $m_i = \max_m P(m|Y_s)$ for $i = 1 : K$
9. **Output:** the number of segments K, the set of segments $S_{1:K}$, the model inferences on each segment $m_{1:K}$

3.3.3 Likelihood Functions

After we get the set of candidate graphs, we need to be able to compute marginal likelihood for each graph. However, we can only do so for decomposable graphs in undirected graphs. For non-decomposable graphs, we will use approximation by adding minimum number of edges to make it decomposable.
Given a decomposable graph, we will assume the following conjugate priors. Comparing with the multivariate linear regression models, everything is the

same except the prior on Σ is now a Hyper-Inverse-Wishart instead of Inverse-Wishart.

Hence we still have the following model, (let $Y_{s+1:t}$ as Y and $n = t - s$)

$$\begin{aligned} Y &= H\beta + \epsilon \\ \epsilon &\sim N(0, I_n, \Sigma) \end{aligned}$$

The priors are,

$$\begin{aligned} \beta &\sim N(0, D, \Sigma) \\ \Sigma &\sim HIW(b_0, \Sigma_0) \end{aligned} \tag{3.12}$$

where D, I_n are defined before, and $b_0 = N_0 + 1 - d > 0$.

When a graph G is decomposable, by the standard graph theory [8], G can be decomposited into a list of components and a list of separators, and each component and separator itself is a clique. Then the joint density of Hyper-Inverse-Wishart can be decomposed as following,

$$P(\Sigma|b_0, \Sigma_0) = \frac{\prod_C P(\Sigma_C|b_0, \Sigma_0^C)}{\prod_S P(\Sigma_S|b_0, \Sigma_0^S)} \tag{3.13}$$

where C is each component and S is each separator.

For each C, we have $\Sigma_C \sim IW(b_0 + |C| - 1, \Sigma_0^C)$ and Σ_0^C is the block of Σ_0 corresponding to Σ_C. The same applies to each S. By (2.17), we have,

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|D, b_0, \Sigma_0) \\ &= \iint P(Y, \beta, \Sigma|D, b_0, \Sigma_0)\, d\beta\, d\Sigma \\ &= \iint P(Y|\beta, \Sigma) P(\beta|D, \Sigma) P(\Sigma|b_0, \Sigma_0)\, d\beta\, d\Sigma \end{aligned}$$

Since $P(Y|\beta, \Sigma)P(\beta|D, \Sigma)$ are the same as before, by integrating out β, we have,

$$\begin{aligned} P(Y|D, \Sigma) &= \int P(Y|\beta, \Sigma) P(\beta|D, \Sigma)\, d\beta \\ &= \frac{1}{(2\pi)^{nd/2}|\Sigma|^{n/2}} \left(\frac{|M|}{|D|}\right)^{\frac{d}{2}} exp(-\frac{1}{2} trace(Y^T P Y \Sigma^{-1})) \end{aligned} \tag{3.14}$$

where $M = (H^T H + D^{-1})^{-1}$ and $P = (I - HMH^T)$ are defined before.

When P is positive definite, P has the Cholesky decomposition $Q^T Q$. (In

practice, we can let the prior $D = c * I$, where I is the identity matrix and c is a scalar. By setting c into a proper value, we can make sure P is always positive definite. This is very important in high dimension.) Now let $X = QY$, then (3.14) can rewrite as the following,

$$\begin{aligned} P(Y|D,\Sigma) &= \frac{1}{(2\pi)^{nd/2}|\Sigma|^{n/2}} \left(\frac{|M|}{|D|}\right)^{\frac{d}{2}} exp(-\frac{1}{2} trace(X^T X \Sigma^{-1})) \\ &= \left(\frac{|M|}{|D|}\right)^{\frac{d}{2}} \left(\frac{1}{(2\pi)^{nd/2}|I_n|^{d/2}|\Sigma|^{n/2}} exp(-\frac{1}{2} trace(X^T I_n^{-1} X \Sigma^{-1}))\right) \\ &= \left(\frac{|M|}{|D|}\right)^{\frac{d}{2}} P(X|\Sigma) \end{aligned} \tag{3.15}$$

where $P(X|\Sigma) \sim N(0, I_n, \Sigma)$, or just $P(X|\Sigma) \sim N(0, \Sigma)$.

Since we use decomposable graphs, this likelihood $P(X|\Sigma)$ can be decomposited as following [8],

$$P(X|\Sigma) = \frac{\prod_C P(X_C|\Sigma_C)}{\prod_S P(X_S|\Sigma_S)} \tag{3.16}$$

where C is each component and S is each separator.

For each C, we have $X_C \sim N(0, \Sigma_C)$ and Σ_C is the block of Σ corresponding to X_C. The same applies to each S.

Now multiply (3.13) and (3.16), we have,

$$P(X|\Sigma)P(\Sigma|b_0,\Sigma_0) = \frac{\prod_C P(X_C|\Sigma_C)P(\Sigma_C|b_0,\Sigma_0^C)}{\prod_S P(X_S|\Sigma_S)P(\Sigma_S|b_0,\Sigma_0^S)} \tag{3.17}$$

Then for each C, the posterior of $\Sigma_C \sim IW(b_0 + |C| - 1 + n, \Sigma_0^C + X_C^T X_C)$. The same holds for each S. As a result, the posterior of $\Sigma \sim HIW(b_0+n, \Sigma_0+X^T X)$. Hence by integrating out Σ, we have,

$$\begin{aligned} P(Y_{s+1:t}|q) &= P(Y|D, b_0, \Sigma_0) \\ &= \int P(Y|D,\Sigma)P(\Sigma|b_0,\Sigma_0)\, d\Sigma \\ &= \left(\frac{|M|}{|D|}\right)^{\frac{d}{2}} (\pi)^{-nd/2} \frac{h(G, b_0, \Sigma_0)}{h(G, b_n, \Sigma_n)} \end{aligned} \tag{3.18}$$

where

$$h(G, b, \Sigma) = \frac{\prod_C |\Sigma_C|^{\frac{b+|C|-1}{2}} Z(b_0 + |C| - 1, |C|)^{-1}}{\prod_S |\Sigma_S|^{\frac{b+|S|-1}{2}} Z(b_0 + |S| - 1, |S|)^{-1}}$$

$$\begin{aligned} Z(n,d) &= \pi^{d(d-1)/4} \prod_{i=1}^{d} \Gamma((n+1-i)/2) \\ \Sigma_n &= \Sigma_0 + Y^T P Y \\ b_n &= b_0 + n \end{aligned}$$

In log space, (3.18) can re-write as following,

$$log(P(Y_{s+1:t}|q)) = -\frac{nd}{2} log(\pi) - \frac{d}{2}(log|M| - log|D|) + log(h(G, b_0, \Sigma_0)) - log(h(G, b_n, \Sigma_n)) \tag{3.19}$$

We will also use the similar rank one update as mentioned earlier. Also notice that $h(G, b, \Sigma)$ contains many local terms. When we evaluate $h(G, b, \Sigma)$ over different graphs, we will cache all local terms. Then later when we meet the same term, we don't need to re-evalue it.

3.4 Experimental Results

Now we will show experimental results on some synthetic and real data sets.

3.4.1 Synthetic Data

First, we revisit the 2D synthetic data mentioned in the beginning of this chapter. We run it with all three different models (independent, full and Gaussian graphical models). We set the hyper parameter $\nu = 2$ and $\gamma = 2$ on σ^2 for each dimension in the independent model; and set the hyper parameter $N_0 = d$ and $\Sigma_0 = I$ on Σ where d is the number of the dimension (in this case d = 2) and I is the identity matrix in the full model; and set the hyper parameter $b_0 = 1$ and $\Sigma_0 = I$ on Σ in the Gaussian graphical model. We know there are three segments. From Figure 3.7, the raw data are shown in the first two rows. The independent model thinks there is only one segment, since the posterior probability of the number of segments is mainly at 1. Hence it detects no change points. The other two models both think there are three segments. Both models detect positions of change points that are close to the ground truth with some

uncertainty.

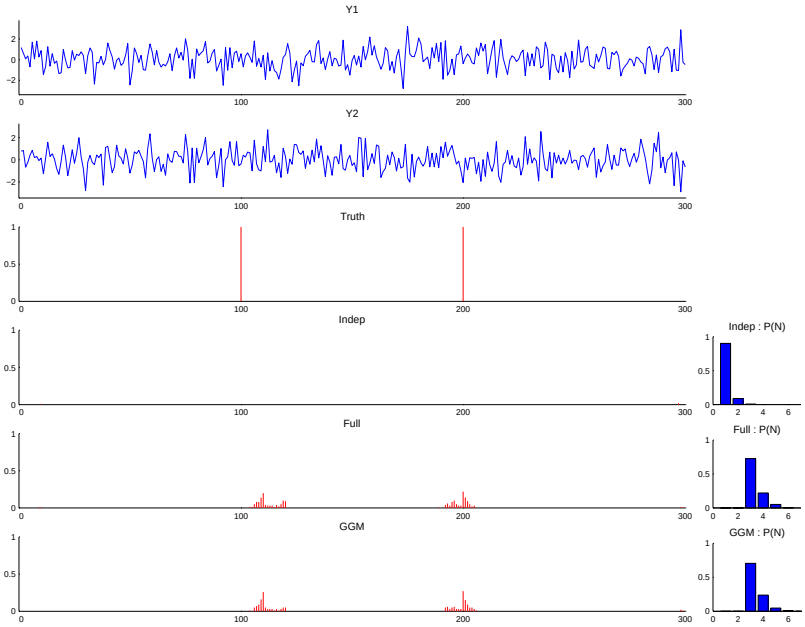

Figure 3.7: Results on synthetic data 2D. The top two rows are raw data. The 3rd row is the ground truth of change points. From the 4th row to the 6th row, results are the posterior distributions of being change points at each position and the number of segments generated from the independent model, the full model and the Gaussian graphical model respectfully. Results are generated by 'show2D'.

Now let's look at a 10D synthetic data. We generate data in a similar way, but this time we have 10 series. And we set the hyper parameters in a similar way. To save space, we only show the first two dimensions since the rest are similar. From Figure 3.8, we see as before the independent model thinks there is only one segment hence detects no change point. The full model thinks there might be one or two segments. Also it detects the position of change point is close to

two ends which is wrong. In this case, only the Gaussian graphical model think there are three segments, and detects positions that are very close to the ground truth. Then based on the segments estimated by Gaussian graphical model, we plot the posterior over all graph structures, $P(G|Y_{s:t})$, the true graph structure, the MAP structure $G_{MAP} = argmax_G P(G|Y_{s:t})$, and the marginal edge probability, $P(G_{i,j} = 1|Y_{s:t})$, computed using Bayesian model averaging (Gray squares represent edges about which we are uncertain) in the bottom two rows of Figure 3.8. Note, we plot the graph structure by its adjacency matrix, such that if node i and j are connected, then the corresponding entry is a black square; otherwise it is a white square.

We plot all candidate graph strutures selected by sliding windows in Figure 3.9. In this case, we have 30 graphs in total. For the 1st segment, we notice that the true graph structure is not included in the candidate list. As a result, GGM picks the graphs that are most close to the true structure. In this case, there are two graphs that both are close. Hence we see some uncertainty over the edges. However on the 3rd segment, the true structure is included in the candidate list. In this case, GGM can identify it correctly, and we see very little uncertainty in the posterior probability. As mentioned earlier, the candidate list contains 30 graphs. Some are very different from the true structures which might be generated by the window overlapping two segments. However, we find these graphs only slow the algorithms but won't hurt their results, because these "useless" graphs all get very low posterior probabilities.

Finally let's look at a 20D synthetic data. We generate data in a similar way and set the hyper parameters in a similar way. To save space, we only show the first two dimensions since the rest are similar. From Figure 3.10, we see as before the independent model thinks there is only one segment hence detects no change point. The full model clearly oversegments the data. Again, only the Gaussian graphical model correctly segments, and detects positions that are very close to the ground truth. Comparing with the results from 2d and 10d cases, we find that the independent model fails to detect in all cases since the changes are on the correlation structures, and the independent model cannot model it. The

full model can detect in low dimensional case, but fails in high dimension cases, because the full model has more parameters to learn. The Gaussian graphical model can detect on all cases, and it is more confident in high dimension cases since the sparsity structures are more important in high dimension.

3.4.2 U.S. Portfolios Data

Now let's look at two real data sets which record annually rebalanced value-weighted monthly returns from July 1926 to December 2001 total 906 month of U.S. portfolios data. The first data set has five industries (manufacturing, utilities, shops, finance and other) and the second has thirty industries.
The first data set has been previously studied by Talih and Hengartner in [24]. We set the hyper parameter $\nu = 2$ and $\gamma = 2$ on σ^2 for each dimension in the independent model; and set the hyper parameter $N_0 = 5$ and $\Sigma_0 = I$ on Σ in the full model; and set the hyper parameter $b_0 = 1$ and $\Sigma_0 = I$ on Σ in the Gaussian graphical model. The raw data are shown in the first five rows of Figure 3.11. Talih's result is shown on the 6th row. From the 7th row to the 9th row, results are from independent model, full model and GGM respectfully.

We find that full model and GGM have similar results since they think there are roughly 7 segments, and they agree on 4 out of 6 change points. Independent model seems to be over-segmented. In this problem, we do not know the ground truth and the true graph structures. Note, only 2 change points that we discovered coincide with the results of Talih (namely 1959 and 1984). There are many possible reasons for this. First, they assume a different prior over models (the graph changes by one arc at a time between neighboring segments); second, they use reversible jump MCMC; third, their model requires to pre-specify the number of change points. In this case, the positions of change points are very sensitive to the number of change points. We think their results could be over-segmented. In this data, sliding windows generates 17 graphs. As usual, based on the segmentation estimated by GGM, we plot the posterior over all 17 graphs

structures, $P(G|Y_{s:t})$, the MAP graph structure $G_{MAP} = argmax_G P(G|Y_{s:t})$, and the marginal edge probability, $P(G_{i,j} = 1|Y_{s:t})$, computed using Bayesian model averaging (Gray squares represent edges about which we are uncertain) in the bottom three rows of Figure 3.11. From the MAP graph structures detected from our results, we find Talih's assumption on the graph structure changing by one arc at a time between neighboring segments may not be true. For example, between the third segment and the fourth segment in our results, the graph structure is changed by three arcs, rather than one.

The second data set is similar to the first one, but has 30 series. Since from the results of synthetic data, we know that in higher dimension, the Gaussian graphical model performs much better than the independent model and the full model, we only show the result from the Gaussian graphical model in Figure 3.12.

We show the first two industry raw data in the top two rows. In the third row, we show the resulted posterior distribution of being change points at each position and the number of segments generated by the Gaussian graphical model. We still don't know the ground truth and the true graph structures. We also show the MAP graph structures detected on three consecutive regions. We can see that the graph structures are very sparse and they differ more than one arcs.

3.4.3 Honey Bee Dance Data

Finally, we analyse the honey bee dance data set used in [19, 20]. This consists of the x and y coordinates of a honey bee, and its head angle θ, as it moves around an enclosure, as observed by an overhead camera. Two examples of the data, together with a ground truth segmentation (created by human experts) are shown in Figure 3.13 and 3.14. We also show the results of segmenting this using a first-order auto-regressive AR(1) model, using independent model or with full covariate model. We preprocessed the data by replacing θ with $sin\theta$ and $cos\theta$ to overcome the discontinuity as the bees moves between $-\pi$ to π. We set the hyper parameter $\nu = 2$ and $\gamma = 0.02$ on σ^2 for each dimension in the

independent model; and set the hyper parameter $N_0 = 4$ and $\Sigma_0 = 0.01 * I$ on Σ in the full model.

From both Figures, the results from full covariate model are very close to the ground truth, but the results from independent model are clearly over segmented.

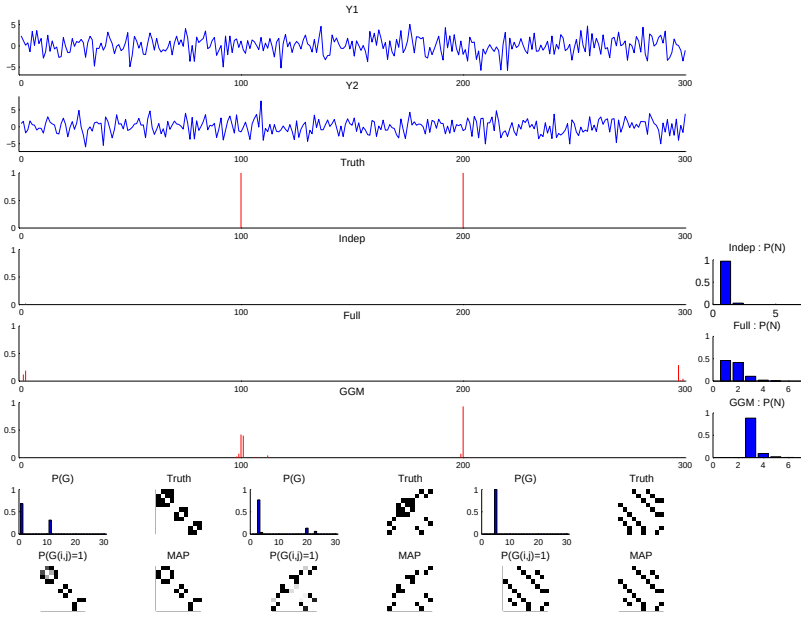

Figure 3.8: Results on synthetic data 10D. The top two rows are the first two dimensions of raw data. The 3rd row is the ground truth of change points. From the 4th row to the 6th row, results are the posterior distributions of being change points at each position and the number of segments generated from the independent model, the full model and the Gaussian graphical model respectfully. In the bottom 2 rows, we plot the posterior over all graph stuctures, $P(G|Y_{s:t})$, the true graph structure, the MAP structure $G_{MAP} = argmax_G P(G|Y_{s:t})$, and the marginal edge probability, $P(G_{i,j} = 1|Y_{s:t})$ on 3 segments detected by the Gaussian graphical model. Results are generated by 'show10D'.

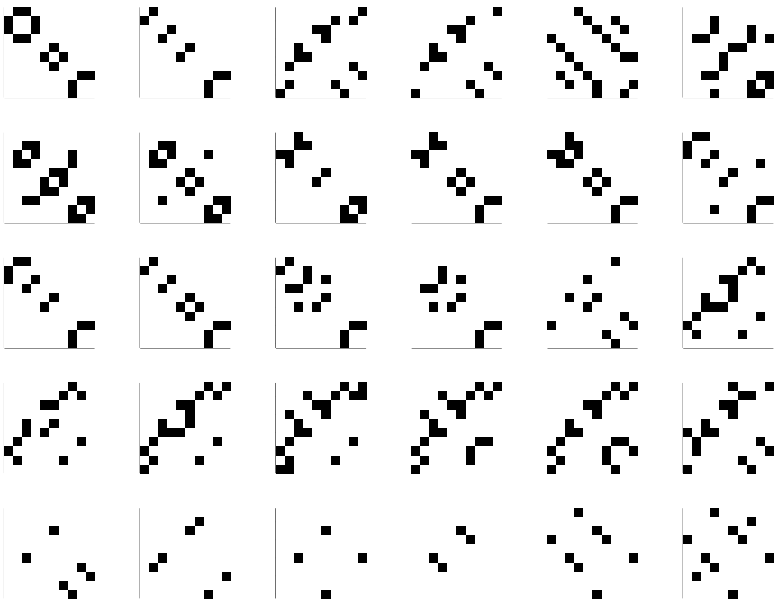

Figure 3.9: Candidate list of graphs generated by sliding windows in 10D data. We plot the graph structure by its adjacency matrix, such that if node i and j are connected, then the corresponding entry is a black square; otherwise it is a white square. Results are generated by 'show10D'.

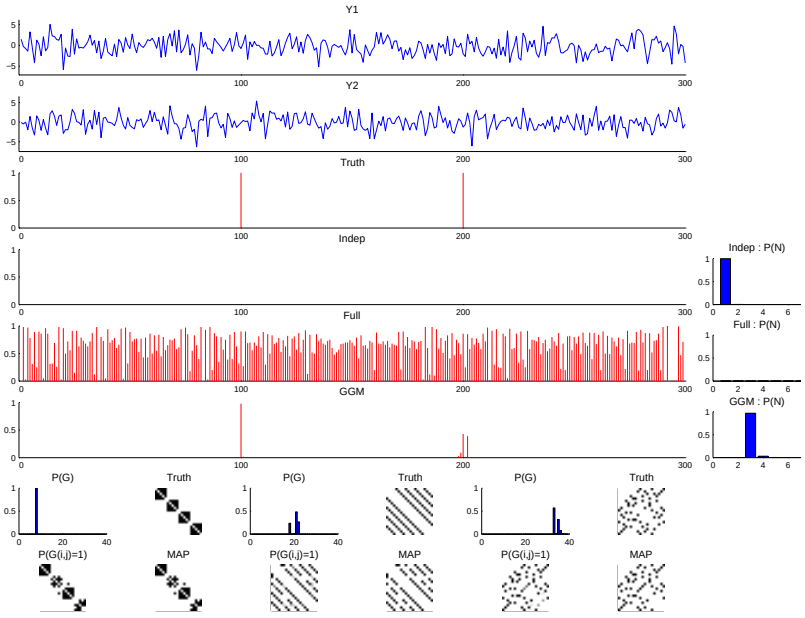

Figure 3.10: Results on synthetic data 20D. The top two rows are the first two dimensions of raw data. The 3rd row is the ground truth of change points. From the 4th row to the 6th row, results are the posterior distributions of being change points at each position and the number of segments generated from the independent model, the full model and the Gaussian graphical model respectfully. In the bottom 2 rows, we plot the posterior over all graph stuctures, $P(G|Y_{s:t})$, the true graph structure, the MAP structure $G_{MAP} = argmax_G P(G|Y_{s:t})$, and the marginal edge probability, $P(G_{i,j} = 1|Y_{s:t})$ on 3 segments detected by the Gaussian graphical model. Results are generated by 'show20D'.

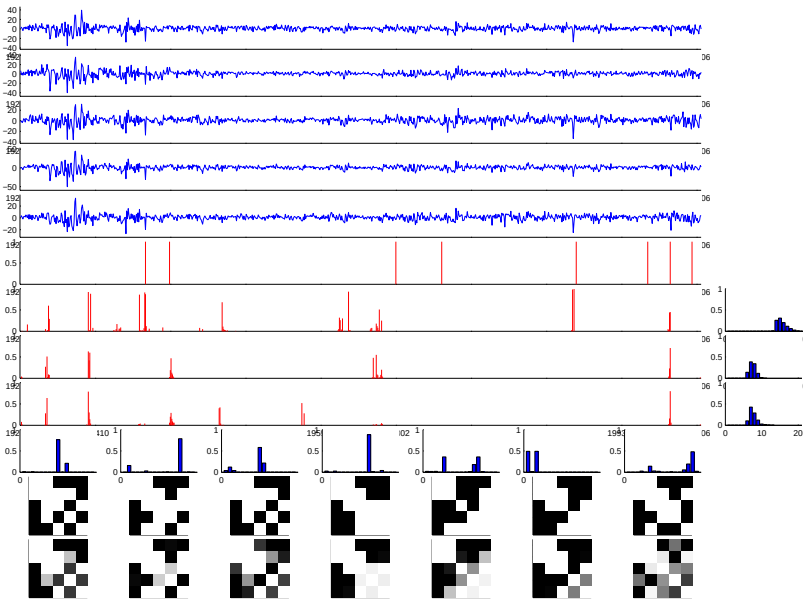

Figure 3.11: Results on U.S. portfolios data of 5 industries. The top five rows are the raw data representing annually rebalanced value-weighted monthly return in 5 industries from July 1926 to December 2001. Total there are 906 month. The 6th row is the result by Talih. From the 7th row to the 9th row, results are the posterior distributions of being change points at each position and the number of segments generated from the independent model, the full model and the Gaussian graphical model respectfully. In the bottom three rows, we plot the posterior over all graph structures, $P(G|Y_{s:t})$, the MAP graph structure $G_{MAP} = argmax_G P(G|Y_{s:t})$, and the marginal edge probability, $P(G_{i,j} = 1|Y_{s:t})$. Results are generated by 'showPortofios'.

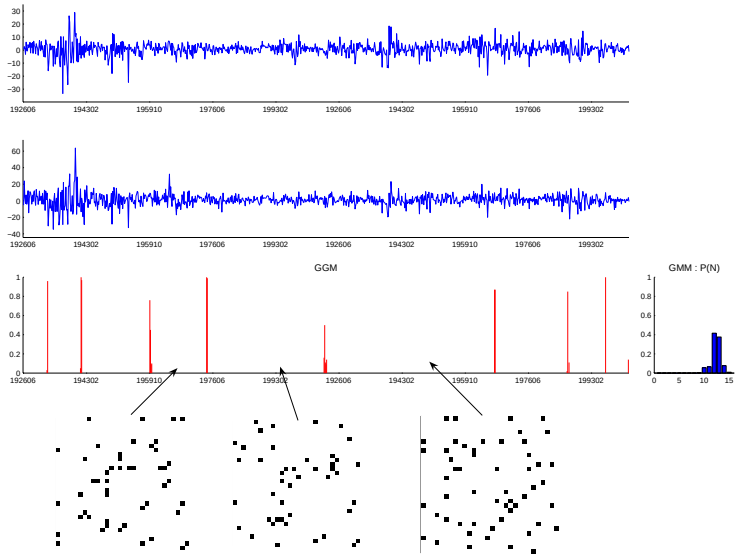

Figure 3.12: Results on U.S. portfolios data of 30 industries. We show the first two industry raw data in the top two rows. The third row is the result of the posterior distribution of being change points at each position and the number of segments generated from the Gaussian graphical model. In the fourth row, we show the MAP graph structures in 3 consecutive regions detected by the Gaussian graphical model. Results are generated by 'showPort30'.

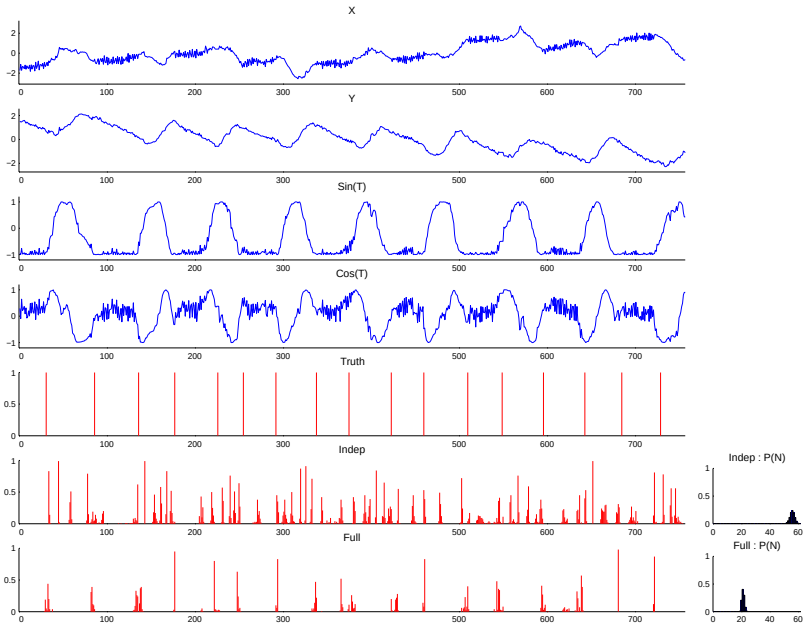

Figure 3.13: Results on honey bee dance data 4. The top four rows are the raw data representing x, y coordinates of honey bee and *sin*, *cos* of its head angle θ. The 5th row is the ground truth. The 6th row is the result from independent model and the 7th row is the result from full covariate model. Results are generated by 'showBees(4)'.

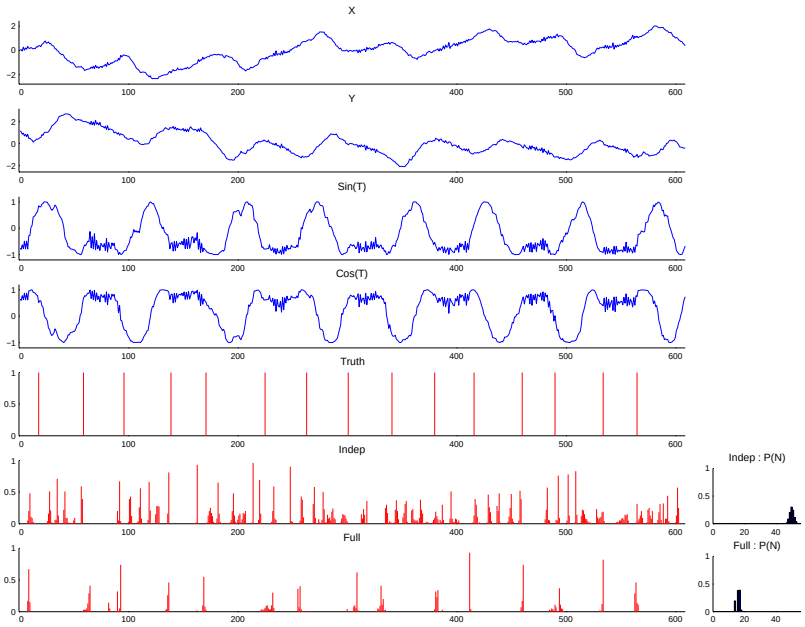

Figure 3.14: Results on honey bee dance data 6. The top four rows are the raw data representing x, y coordinates of honey bee and *sin*, *cos* of its head angle θ. The 5th row is the ground truth. The 6th row is the result from independent model and the 7th row is the result from full covariate model. Results are generated by 'showBees(6)'.

Chapter 4

Conclusion and Future Work

In this thesis, we extended Fearnhead's algorithms to the case of multiple dimensional series. This allowed us to detect changes on correlation structures, as well as changes on mean, variance, etc. We modeled the correlation structures using undirected Gaussian graphical models. This allowed us to estimate the changing topology of dependencies among series, in addition to detecting change points. This is particularly useful in high dimensional cases because of sparsity. We can also model the correlation structures by directed graphs. In directed graphs, we can compute the marginal likelihood for any graph structures. However, we don't have a fast way to generate candidate list of graph structures since currently all structure learning algorithms for directed graphs are too slow. Hence if there is a fast way to learn structure of directed graphs, we can use directed graphs.

Finally, the conditional independence property sometime might not be reasonable. Or we might want to model other constrains on changing over consecutive segments. This requires us to come up with new models.

Bibliography

[1] Jim Albert and Patricia Williamson. Using model/data simulation to detect streakiness. *The American statistician*, 55:41–50, 2001.

[2] Onureena Banerjee, Laurent El Ghaoui, Alexandre d'Aspremont, and Georges Natsoulis. Convex optimization techniques for fitting sparse gaussian graphical models. *Proceedings of the 23rd international conference on Machine learning*, pages 89–96, 2006.

[3] Daniel Barry and J. A. Hartigan. Product partition models for change point problems. *The Annals of Statistics*, 20(1):260–279, 1992.

[4] Daniel Barry and J. A. Hartigan. A bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88(421):309–319, 1993.

[5] Albright S. C., Albert J., Stern H. S., and Morris C. N. A statistical analysis of hitting streaks in baseball. *Journal of the American Statistical Association*, 88:1175–1196, 1993.

[6] Bradley P. Carlin, Alan E. Gelfand, and Adrian F. M. Smith. Hierarchical bayesian analysis of changepoints problems. *Applied Statistics*, 41(2):389–405, 1992.

[7] J. Carpenter, Peter Clifford, and Paul Fearnhead. An improved particle filter for non-linear problems. *IEE Proceedings of Radar and Sonar Navigation*, 146:2–7, 1999.

[8] Carlos Marinho Carvalho. *Structure and sparsity in high-dimensional multivariate analysis*. PhD thesis, Duke University, 2006.

[9] Nicolas Chopin. Dynamic detection of change points in long time series. *The Annals of the Institute of Statistical Mathematics*, 2006.

[10] D. G. T. Denison, B. K. Mallick, and A. F. M. Smith. Automatic bayesian curve fitting. *Journal of Royal Statistical Society Series B*, 60:333–350, 1998.

[11] David G. T. Denison, Christopher C. Holmes, Bani K. Mallick, and Adrian F. M. Smith. *Bayesian methods for nonlinear classification and regression*. John Wiley & Sons, Ltd, Baffins, Chichester, West Sussex, England, 2002.

[12] Arnaud Doucet, Nando De Freitas, and Neil Gordon. *Sequential Monte Carlo methods in practice*. Springer-Verlag, New York, 2001.

[13] Paul Fearnhead. Exact bayesian curve fitting and signal segmentation. *IEEE Transactions on Signal Processing*, 53:2160–2166, 2005.

[14] Paul Fearnhead. Exact and efficient bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16(2):203–213, 2006.

[15] Paul Fearnhead and Peter Clifford. Online inference for hidden markov models via particle filters. *Journal of the Royal Statistical Society: Series B*, 65:887–899, 2003.

[16] Paul Fearnhead and Zhen Liu. Online inference for multiple change point problems. 2005.

[17] R. G. Jarrett. A note on the intervals between coal-mining disasters. *Biometrika*, 66(1):191–193, 1979.

[18] Peter J.Green. Reversible jump markov chain monte carlo computation and bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[19] Sang Min Oh, James M. Rehg, Tucker Balch, and Frank Dellaert. Learning and inference in parametric switching linear dynamic systems. *IEEE International Conference on Computer Vision*, 2:1161–1168, 2005.

[20] Sang Min Oh, James M. Rehg, and Frank Dellaert. Parameterized duration modeling for switching linear dynamic systems. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2:1694–1700, 2006.

[21] Elena Punskaya, Christophe Andrieu, Arnaud Doucet, and William J. Fitzgerald. Bayesian curve fitting using mcmc with application to signal segmentation. *IEEE Transactions on Signal Processing*, 50:747–758, 2002.

[22] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods*. Springer, New York, 2004.

[23] Juliane Schafer and Korbinian Strimmer. A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, 4, 2005.

[24] Makram Talih and Nicolas Hengartner. Structural learning with time-varying components: Tracking the cross-section of financial time series. *Journal of Royal Statistical Society Series B*, 67:321–341, 2005.

[25] T. Y. Yang and L. Kuo. Bayesian binary segmentation procedure for a poisson process with multiple changepoints. *Journal of Computational and Graphical Statistics*, 10:772–785, 2001.

[26] Tae Young Yang. Bayesian binary segmentation procedure for detecting streakiness in sports. *Journal of Royal Statistical Society Series A*, 167:627–637, 2004.