# Direct Simulation Methods for Multiple Changepoint Problems

**Zhen Liu, B.Sc.**

Submitted for the degree of Doctor of Philosophy

at Lancaster University,

September 2007.



ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

# Direct Simulation Methods for Changepoints Problems

**Zhen Liu B.Sc**

Department of Mathematics and Statistics

Fylde College, Lancaster University

Submitted for the degree of Doctor of Philosophy

at Lancaster University, September 2007.

## Abstract

The multiple changepoint model has been considered in a wide range of statistical modelling, as it increases the flexibility to simple statistical applications. The main purpose of the thesis enables the Bayesian inference from such models by using the idea of particle filters. Compared to the existed methodology such as RJMCMC of Green (1995), the attraction of our particle filter is its simplicity and efficiency.

We propose an on-line algorithm for exact filtering for a class of multiple changepoint problems. This class of models satisfy an important conditional independence property. This algorithm enables simulation from the true joint posterior distribution of the number and position of the changepoints for a class of changepoint models. The computational cost of this exact algorithm is quadratic in the number of observations. We further show how resampling ideas from particle filters can be used to reduce the computational cost to linear in the number of observations, at the expense of introducing small errors; and propose two new, optimum resampling algorithms for this problem. In practice, large computational savings can be obtained whilst introducing negligible error. We demonstrate how the resulting particle filter is practicable for segmentation of human GC content.

We then generalise our method to models where the conditional independence

property does not hold. In particular we consider models with dependence of the parameters across neighbouring segments.

Examples of such models are those with unknown hyper-parameters, and piecewise polynomial regression models which assume continuity of the regression function. The particle filter we propose is based on a simple approximation to the filtering recursion. We show that the error introduced by the approximation can be small.

We demonstrate our method on the problem of Bayesian curve fitting. The novelty of our model is that we fit a piecewise polynomial function and allow for both discontinuity and continuity at changepoints. This method is compared to existing Bayesian curve fitting method, and applied to the analysis of well-log data.

# Declaration

I declare that the work presented in this thesis is my own, except where stated otherwise. The work of Chapter 4 is based on the publication of Fearnhead and Liu (2007).

The algorithms developed in Chapter 4 and 5 are coded by myself in a combination of R and C++ programming languages. The template numerical toolkit (TNT) developed by National Institute of Science and Technology (NIST) is used for matrices manipulation.

Zhen Liu

# Acknowledgments

I am very grateful to my supervisor, Professor Paul Fearnhead. I feel very lucky to be supervised by him in the past three years. His innovative thoughts, modest attitude and remarkable achievement in his research work has been inspired me throughout my PhD study. He has been extremely supportive of me, as he has shown great patience to check every line of my code and thesis. He also provided me a lot of chances to attend different training and conferences.

I am also very grateful to Dr Joe Whittaker, for his encouragement and guidance in the first year of my PhD study. When I arrived this country that is more than 10,000 miles away from home for the first time,it was him who made me confident to finish this thesis.

My life at Lancaster would have been different without a friendly environment. In particular, I would like to thank Dongfang Shi, David Yen, Hongsheng Dai, Yanchun Bao, Peter Henrys, Tristan Marshall, Ting-li Su and Vilda Purutcuoglu for their friendship, advice and invaluable discussions. I would also like to thank the football teammates in the department. The game with them made me very energetic in the academic research.

Of course, I would like to thank my parents for their supports in all aspects. They spent countless time and energy to make me a doctor without thinking of any reward. However, it is time to reward them now. Two friends from the other side of the Atlantic Ocean are worth mentioning: Dan Liao and Junling Xiong. I would

like to take the chance to thank they two.

Finally, I gratefully acknowledge the financial support for this project given by the EPSRC.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Retrospect

Sampling-based Bayesian statistical methods have been very popular in the last 20 years because of its simplicity in approximating the intractable integrals involved with the inferential problem, particularly in high dimensions. All these methods are based on the Monte Carlo integration, in which a set of samples $x^{(1)}, \ldots, x^{(N)}$ are independently simulated from a target distribution of random variable $X$ with probability density function $p(x)$. Then we can use these samples to approximate the expectation of any function $h(\cdot)$ of $X$, provided the expectation exists. That is if we want to calculate

$$\vartheta = \int h(x)p(x)\mathrm{d}x = \mathrm{E}_p(h(X)), \tag{1.1}$$

we can approximate it by a sample mean

$$\hat{\vartheta}(p) = \frac{1}{N}\sum_{i=1}^{N} h(x^{(i)}). \tag{1.2}$$

The approach is remarkably easy to use and gives an unbiased estimate with variance proportional to $1/N$ (i.e. $\text{var}(h(x))/N$).

In particular, if we take $h(x) = I_A(x)$ where $I_A(x)$ is an indicator function so that it takes value 1 if $x \in A$ and value 0 otherwise, then the probability $\Pr(x \in A)$ are approximated only by the proportion of samples in $A$:

$$\Pr(x \in A) = \mathrm{E}(I_A(x)) \approx \frac{1}{N} \sum_{i=1}^{N} I_A(x^{(i)}).$$

### 1.1.1 Rejection sampling

However, It is often the case that we are unable to simulate directly from the target density $p(\cdot)$. A sensible method to overcome the problem is to simulate from another density $q(\cdot)$ which is easy to simulate from, but then to only accept those samples with a probability $p_{accept}$. This is the basic idea of *rejection sampling* (Hammersley and Handscomb, 1964).

To run the method, we only need to know the target density $p(\cdot)$ up to a normalising constant, and have to set an upper bound $K$ such that

$$p(x)/q(x) \leq K \qquad \text{for all } x,$$

therefore the support of $q(\cdot)$ contains all the support of $p(\cdot)$. The sampling procedure is done as follows:

**Algorithm 1.1** *Rejection sampling*

**Step 1** *Simulate $\tilde{x}$ from the proposal density $q(x)$;*

**Step 2** *Calculate the accept probability as $p_{accept} = p(\tilde{x})/(Kq(\tilde{x}))$;*

**Step 3** *Generate a random variable $U$ uniformly from the interval* $[0, 1]$*;*

**Step 4** *If $U < p_{accept}$ accept $\tilde{x}$; otherwise repeat.*

Then $\tilde{x}$s accepted by this algorithm are independent identically distributed (i.i.d) samples from the target distribution. Furthermore, the average acceptance probability is $1/K$.

The efficiency of the rejection sampling is dependent on the upper bound $K$, and particularly the dimensions of the target distribution. The acceptance probability decreases exponentially as the dimension increases.

### 1.1.2 Markov chain Monte Carlo

If we run the rejection sampling iteratively over an irreducible and aperiodic Markov chain whose equilibrium distribution is the target distribution, this is the intuitive idea behind *Markov chain Monte Carlo* (MCMC).

The main difficulty of MCMC is how to construct a suitable Markov chain to enable a simulation from the target distribution. A general algorithm which they call *Metropolis-Hasting algorithm* is proposed by Metropolis et al. (1953) and then generalised by Hasting (1970). The algorithm requires a transition kernel $k(x, x')$ for the Markov chain, which is a proposal density function of $x'$ for each given value of $x$. Thus at each iteration, a sample $x'$ is drawn from the kernel $k(x, x')$, and the new value $\tilde{x}$ in the chain is

$$\tilde{x} = \begin{cases} x' & \text{with probability } p_{accept}, \\ x & \text{with probability } 1 - p_{accept}, \end{cases} \tag{1.3}$$

where

$$p_{accept} = \min\left\{1, \frac{k(x', x)p(x')}{k(x, x')p(x)}\right\}. \tag{1.4}$$

The initial value of $x$ can be chosen arbitrarily. Then Tierney (1994) has proved that the Markov chain obtained by the above algorithm is time-reversible and has an equilibrium distribution $p(\cdot)$.

The transition kernel can be chosen arbitrarily as well, in principle, so any choice should work. However, not all kernels are equally good with respect to the convergence property (or mixing property) of the algorithm. Common choices include fully conditional distribution (in *Gibbs sampling*) and random walk with normal increment (in *random walk Metropolis algorithm*). For a complete review of MCMC, see Gilks et al. (1996); Robert and Casella (1999). Note that MCMC does not provide independent draws from $p(\cdot)$; but Monte Carlo estimators such as (1.2) will still be consistent.

## 1.2 Motivation

The MCMC method has been very successful since the beginning of 1990s, because of its flexibility to a lot of statistical models. It is a popular approach to sample different complicated probability distributions. However, there are still some limitations of MCMC method in some situations. For example, it is inefficient for the recursive estimation problems. Hence, we introduce in the thesis another sampling method based on the importance sampling.

### 1.2.1 Importance sampling

There might be another problems with $\hat{\vartheta}(p)$ as an estimator of $\vartheta$: although sampling from $p(\cdot)$ is possible, the estimator $\hat{\vartheta}(p)$ might have very high variance.

Instead, an *importance sampling* technique (see Geweke, 1989, for example) can be used to overcome the problem. We can choose another distribution of the random variable $X$ with density $q(x)$, from which, the samples $x^{(1)}, \ldots, x^{(N)}$ can be easily simulated. Thus, we can rewrite $\vartheta$ as

$$\vartheta = \int h(x)\frac{p(x)}{q(x)}q(x)\mathrm{d}x, \tag{1.5}$$

and it can be approximated by

$$\hat{\vartheta}(q) = \sum_{i=1}^{N} w^{(i)}h(x^{(i)}), \tag{1.6}$$

where we define the (normalised) *importance weight* $w^{(i)}$ as

$$w^{(i)} \propto \frac{p(x^{(i)})}{q(x^{(i)})}, \qquad \sum_{i=1}^{N} w^{(i)} = 1. \tag{1.7}$$

Thus the importance sampling is basically choosing the samples concentrated on the area where there is greatest variation in the integrand so that each simulated value contains greatest information. If we choose $q(\cdot)$ so as to make $h(x)p(x)/q(x)$ nearly constant, the variance of $\hat{\vartheta}(q)$ will be much lower than the variance of $\hat{\vartheta}(p)$.

Note that it is also possible to use unnormalised weights in the approximation such that

$$\hat{\vartheta}(q) = \frac{1}{N}\sum_{i=1}^{N} \tilde{w}^{(i)}h(x^{(i)}), \tag{1.8}$$

where the importance weight $\tilde{w}^{(i)}$ is

$$\tilde{w}^{(i)} = \frac{p(x^{(i)})}{q(x^{(i)})} \tag{1.9}$$

instead. However, in many applications, the target probability $p(\cdot)$ and the proposal density $q(\cdot)$ may be known only up to a normalising constant. This is always true when applying the importance sampling to the state space model and, particularly in Bayesian statistics. Hence the use of normalised importance weights is more general.

### 1.2.2 Sequential Monte Carlo method

Importance sampling has a wider scope than reducing the variance of Monte Carlo estimators. This thesis will concentrate on the importance sampling in the sequential settings, which is also known as *particle filters* (Doucet et al., 2001; Liu, 2001).

The technique has been commonly used in time series model for some dynamic problems such as target tracking (Gordon et al., 1993), signal deconvolution (Liu and Chen, 1995), speech recognition (Godsill and Clapp, 2001), oil drilling (Fearnhead and Clifford, 2003) and stock pricing (Kitagawa, 1996), amongst others. In such cases, the new observation becomes available at each time, thus a real time inference or prediction is required. In other words, a sequence of distributions $\pi_t$, which is the posterior of the underlying states given the observations in the dynamic system, needs to be estimated at each time $t$. A typical example of $\pi_t$ is the position and speed of a target at time $t$ in the target tracking problem.

The reason why the importance sampling can be used efficiently to estimate these posteriors is that the approximate samples from the distribution $\pi_t$ can be recycled by importance sampling to produce approximate samples from the distribution $\pi_{t+1}$, provided the two distributions share same supports. Even if the supports

are different, we can augment the supports of $\pi_t$ to the supports of $\pi_{t+1}$, and simulate the imputed samples to approximate $\pi_{t+1}$ (Kong et al., 1994). The biggest advantage of this sequential computing is that the importance weights at each time $t$ do not need to be re-computed from the scratch. The dynamic updating produces a reduction on the computational cost.

The motivation of the thesis is to consider and develop particle filters for analysis of multiple changepoint problems. With particle filters, we aim to draw samples directly from the posterior distribution of changepoints.

The multiple changepoints model we use here consists of a sequence of changepoints occurring at discrete positions. Both the number and positions of them are unknown. The MCMC method has been dominant in the Bayesian analysis of the changepoints models. If the number of changepoints is known, the method can be directly used for inference in the models (e.g. Stephens, 1994; Chib, 1996). If the number of changepoints is unknown, a common approach is the *reversible jump Markov chain Monte Carlo* (RJMCMC) method of Green (1995). However, RJMCMC can suffer from poor mixing, and hence a high CPU cost, unless efficient MCMC move can be designed. But this is generally very hard, particularly for the move between different models in RJMCMC (see Brooks et al., 2003, for guidelines on how to design these moves).

By contrast, the particle filtering approach to changepoints model is less obvious. An artificial time has to be given so that a pseudo sequence of the posterior distributions of changepoints can be fed into the particle filters as it were the target distributions arising in a dynamic problem. The particle filtering approach avoids the diagnosis of the convergence of Markov chains and hence the design of moves in the MCMC. It is also believed that the particle filter approach provides better estimates in terms of robustness and effectiveness.

Although we introduced the particle filter as an alternative to the MCMC method,

the two are not that separated. Instead, we can even embed one algorithm into the other, to improve the performance of the algorithm.

## 1.3 Outline of the thesis

The theme of this thesis is the construction of a direct simulation methodology based on the particle filters, and the application to the multiple changepoint problems. The method is proposed to enable inference for the changepoint model to be made more efficiently. The outline of the subsequent chapters is as follows:

In Chapter 2, the basic structure of particle filter including sampling, resampling and smoothing is introduced. A very simple example which is known as the SIR filter or Bootstrap filter (Gordon et al., 1993) is given immediately to demonstrate how the particle filter works on the non-linear/non-Gaussian state space model. Motivated from the demonstrative example, a number of literature focusing on improving the performance of the particle filters are reviewed. The improvements cover all aspects of the particle filters (e.g. sampling, resampling and smoothing).

In Chapter 3, we describe the multiple changepoint problem through a state space model so that the on-line inference can be made. We adapt a point process of Barry and Hartigan (1993) to model the distribution of the positions of changepoints and the number of changepoints is automatically implied. The underlying states have a hierarchy with the changepoints and the associated parameters, which will make the particle filters introduced in Chapter 2 less accurate and efficient. So it is advantageous to marginalise the parameter state sequence as nuisance parameters and focus on the on-line inference of changepoints first. Two specific examples given by Chen and Liu (2000) and Chopin (2007) respectively are reviewed. The approach of Chen and Liu (2000) is a special case of Rao-Blackwellised particle filter when the state space model is linear/Gaussian conditional on the change-

points.

The innovative part of the thesis is Chapter 4. We propose an on-line algorithm for exact filtering of the multiple changepoint problems. This algorithm enables simulation from the true joint posterior distribution of the number and position of the changepoints for a class of changepoint models. The algorithm is constructed with in a particle filter framework, and we demonstrate how the resulting particle filter is practicable for segmentation of human GC content.

In Chapter 5, we extend the multiple changepoint model to allow for dependencies across segments and apply it to the curve fitting examples. We propose an algorithm for approximated filtering of the multiple changepoints model and a smoothing algorithm to detect both the positions and types of changepoints. We demonstrate the performance of our algorithm on both smooth and unsmooth curves, and compare the it with some MCMC method. Practically, we use the algorithm to analyse well log data from the oil industry. The results are presented there as well.

In the final chapter, we present some conclusions and point out some further research in this field.

# Chapter 2

# Particle filters

## 2.1 Introduction

Particle filters are sequential Monte Carlo methods based upon point mass (or "particle") representation of probability densities, which are widely applied for on-line inference of state space models:

$$\begin{aligned} X_t &= \mathbf{f}(X_{t-1}, W_t) \\ Y_t &= \mathbf{g}(X_t, V_t). \end{aligned} \tag{2.1}$$

Here $W_t$ and $V_t$ are sequences of mutually independent random variables of known distribution. To enable the inferences of the underlying states $X_t$ to be made, the measurements $Y_t$ are taken at each discrete time $t = 1, 2, \ldots, n$. The underlying states $X_t$ follow a *Markov process*. We denote the transition probabilities implied by (2.1) as $p(x_{t+1}|x_t)$; and assume a prior distribution for the state at time 1, $p(x_1)$.

If (2.1) are linear equations, and $W_t$ and $V_t$ have Gaussian distributions, the Kalman filter (Kalman and Bucy, 1961) can be used to calculate the posterior

distribution of the states. If the assumptions fail to hold, some other sub-optimal algorithm needs to be used, like the particle filter.

The particle filter gives a Monte Carlo approximation to the distributions of interest. A set of comprehensive reviews of particle filters can be found in Liu and Chen (1998); Doucet et al. (2001); Arulampalam et al. (2002). The use of Monte Carlo methods in filtering can be traced back to the pioneering contribution of Handschin and Mayne (1969) and Handschin and Mayne (1970), in which the Monte Carlo methods are used only to estimate the mean and covariance of the posterior. Another earlier sequential Monte Carlo methods was proposed by West (1992) when filtering with the mixture probability densities. Alternatives to the particle filters include the extended Kalman Filter (Jazwinski, 1973; Anderson and Moore, 1979), the Gaussian sum filter (Sorenson and Alspach, 1971) and the approximate grid-based methods (Bucy and Senne, 1971). See also West and Harrison (1997) for a complete review.

### 2.1.1 The basis of particle filters

The aim of the particle filter is to estimate recursively in time the posterior distribution of states $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$ (where $\mathbf{x}_{1:t} := (x_1, \ldots, x_t)$ and $\mathbf{y}_{1:t} := (y_1, \ldots, y_t)$), or the marginal distribution $p(x_t|\mathbf{y}_{1:t})$ (also known as the *filtering distribution*), and consequently, some functions of the states, e.g. the expectations $E_p(h(X_t))$. We focus on the filtering distribution in this thesis.

At any time $t$, the marginal distribution $p(x_t|\mathbf{y}_{1:t})$ is given by *Bayes' theorem*

$$p(x_t|\mathbf{y}_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|\mathbf{y}_{1:t-1})\mathrm{d}x_{t-1}, \tag{2.2}$$

$$p(x_t|\mathbf{y}_{1:t}) = \frac{p(y_t|x_t)p(x_t|\mathbf{y}_{1:t-1})}{\int p(y_t|x_t)p(x_t|\mathbf{y}_{1:t-1})\mathrm{d}x_t}. \tag{2.3}$$

Although the recursions of posterior $p(x_t|\mathbf{y}_{1:t})$ are easily obtained, solving them

is normally intractable, as it involves the evaluation of complex high-dimensional integrals in calculating $\int p(y_t|x_t)p(x_t|\mathbf{y}_{1:t-1})\mathrm{d}x_t$. The basic idea of particle filter is to use importance sampling sequentially to approximate the intractable integrals appearing in equations (2.2) and (2.3).

The particle filter is based on the assumption that the probability density function is able to be approximated by a swarm of weighted particles. Given that there has been a discrete set of particles and associated weights $\left(x_{t-1}^{(i)}, w_{t-1}^{(i)}\right)$ at time $t-1$, for $i = 1, \ldots, N$, the posterior distribution of $x_{t-1}$ therefore can be approximated by:

$$\hat{p}(x_{t-1}|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} w_{t-1}^{(i)} \delta(x_{t-1} - x_{t-1}^{(i)}), \tag{2.4}$$

where $\delta(\cdot)$ is the Dirac-Delta function. Substituting it into (2.2) and (2.3), the density function at next time $t$ can be approximated as:

$$\hat{p}(x_t|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} p(x_t|x_{t-1}^{(i)}) w_{t-1}^{(i)}, \tag{2.5}$$

$$\hat{p}(x_t|\mathbf{y}_{1:t}) \propto \sum_{i=1}^{N} p(y_t|x_t) p(x_t|x_{t-1}^{(i)}) w_{t-1}^{(i)}. \tag{2.6}$$

One iteration of the particle filter produces an approximation of (2.6) by a set of weighted particles. One possible approach is to draw the particles $x_t^{(i)}$ from the transitional probability $p(x_t|x_{t-1}^{(i)})$, for $i = 1, \ldots, N$ and approximate (2.6) by these particles with weights

$$w_t^{(i)} \propto w_{t-1}^{(i)} p(y_t|x_t^{(i)}) \quad \text{and} \quad \sum_{i=1}^{N} w_t^{(i)} = 1. \tag{2.7}$$

Thus the weights have been easily updated from the previous weights. So the calculation of $p(x_t|\mathbf{y}_{1:t})$ is a completely sequential update.

Similarly, if the joint posterior distribution $p(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1})$ is approximated by

$$\hat{p}(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} w_{t-1}^{(i)} \delta(x_{1:t-1} - x_{1:t-1}^{(i)}).$$

Due to the Markov property, we can still draw $x_t^{(i)}$ from $p(x_t|x_{t-1}^{(i)})$ and attach it to the particle $\mathbf{x}_{1:t-1}^{(i)}$. Then the associated weight of the new particle $\mathbf{x}_{1:t}^{(i)}$, $w_t^{(i)}$, is updated in the same way as (2.7).

### 2.1.2 Resampling in particle filters

The problem of the updating process is that the variance of the weights increases exponentially over time (Kong et al., 1994; Doucet, 1998), which means that after a few iterations, the distribution of importance weights becomes more and more skewed. As time increases, all but one particle has negligible weights. This is known as *degeneracy*. The algorithm, consequently, fails to give a good approximation to the true posterior distributions.

Simply increasing the sample size can not solve the degeneracy problem. Instead, resampling can be used to reduce the effect of degeneracy. The key point of resampling is to eliminate the particles having small weights and concentrate on the particles which have large weights. Thus, only those particles with significant weights will be selected and propagated to the next time. A typical resampling method is the multinomial sampling (Gordon et al., 1993), in which all the particles produced at time $t$ will be resampled according to their weight $w_t^{(i)}$, i.e.

$$\Pr(x_t^{new} = x_t^{(i)}) = w_t^{(i)} \qquad i = 1, \ldots, N.$$

We can also select the particles deterministically so that each particle $x_t^{(i)}$ has a

$N_i$ copies after resampling, where

$$N_i = \left[ N w_t^{(i)} \right].$$

$[\cdot]$ stands for the integer part.

Resampling also bring some undesired effects: (i) it increase the variance of any estimator, so estimation should be done before the resampling (Liu and Chen, 1998); (ii) resampling limits the use of parallel computing since all particles have to be combined together before resampling; (iii) if we focus on estimating $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$ other than $p(x_t|\mathbf{y}_{1:t})$, resampling can lead to *sample impoverishment* when the number of distinct values $x_k$ for $k << t$ that are stored will be small.

### 2.1.3 Sampling importance resampling

The above procedure naturally leads to a very simple particle filter called *sampling importance resampling* (SIR) filter (Gordon et al., 1993). The algorithm has also been developed independently by Kitagawa (1996) and Isard and Blake (1996), where it is called *Monte Carlo filter* and *Condensation algorithm*, respectively.

The innovation of the SIR filter is that a resampling step is introduced at every time to overcome the degeneracy problem. At each time $t$, the particles will be resampled $N$ times by multinomial sampling and all selected particles will have equal weights, $1/N$ say. This swarm of particles is assumed to be an approximate sample from the true posterior at that time. According to (2.7), the weight at the next time before the resampling is

$$w_{t+1}^{(i)} \propto \frac{1}{N} p(y_{t+1}|x_{t+1}^{(i)}) \propto p(y_{t+1}|x_{t+1}^{(i)}). \tag{2.8}$$

We list the steps of SIR as follows:

**Algorithm 2.1** *The SIR filter*

*For* $t = 1, \ldots, n$

**Sampling** *For each* $x_{t-1}^{(i)}$, *if* $t = 1$, *draw* $\tilde{x}_1^{(i)}$ *directly from* $p(x_1)$; *if* $t \geq 2$, *draw a new particle* $\tilde{x}_t^{(i)}$ *independently from* $p(x_t|x_{t-1}^{(i)})$.

**Weighting** *Assign each particle* $\tilde{x}_t^{(i)}$ *a (normalised) weight which is calculated as*

$$w_t^{(i)} = \frac{p(y_t|\tilde{x}_t^{(i)})}{\sum_{i=1}^{N} p(y_t|\tilde{x}_t^{(i)})}, \tag{2.9}$$

*for* $i = 1, \ldots, N$.

**Resampling** *Resample the* $N$ *particles independently* $N$ *times, with replacement, according to the associated weights. Then assign the newly simulated particles equal weights* $\frac{1}{N}$. *Denote them by* $x_t^{(i)}$.

The term "importance resampling" comes from the weighting stage where an empirical distribution that approximates the target distribution $p(x_t|\mathbf{y}_{1:t})$ is generated by essentially using importance sampling approach.

The SIR filter is very convenient to use, as (i) it uses the transition probability $p(x_t|x_{t-1})$ as proposal density which is easily sampled; (ii) it uses the likelihood of each particle as the weight, which is easily evaluated; and (iii) it uses multinomial sampling to select the particles.

But the cost to pay for the convenience is expensive: (i) the proposal density function ignores the information of the observations, which makes the filter inefficient and sensitive to outliers; (ii) multinomial sampling can add substantial Monte Carlo variation to the algorithm. (iii) the resampling at each time may be unnecessary.

We now look at ideas suggested to address these problems. The first two problems

are quite fundamental. They are related to two major aspects of particle filters: (i) how to sample the particles and (ii) how to resample these particles. Many ideas have been made to stress these two problem in the past 15 years, and these will be briefly reviewed later. The third problem can be overcome by doing resampling only if the weights become sufficiently skewed. We discuss this problem first.

### 2.1.4 When to resample

A number of papers look at when to resample (Kong et al., 1994; Liu and Chen, 1995, 1998). The idea is that the effect of resampling is greatest when the weights are highly skewed. This can be measured via *effective sample size* (ESS), which is originally used to measure the efficiency of importance sampling (Liu, 1996; Neal, 1998). The ESS answers the question that is how large a simple random sample from a target distribution would be required to estimate the function of interest in the importance sampling. This is similar to the idea of auto-correlation time for MCMC method, which is used to measure the effective sample size of a sample of size $N$ generated from the Markov chain.

Liu (1996) has derived an analytical approximation result to the ESS at each time $t$ in particle filters:

$$N_{ess} = \frac{N}{1 + \mathrm{Var}_{p(x_t|x_{t-1})}(r_t)}, \tag{2.10}$$

where $r_t = p(x_t|\mathbf{y}_{1:t})/p(x_t|x_{t-1})$ and the $\mathrm{Var}_{p(x_t|x_{t-1})}(r_t)$ is the variance of $w_t$ with respect to the transition probability function. However, it is in general hard to obtain $\mathrm{Var}_{p(x_t|x_{t-1})}(r_t)$, Kong et al. (1994) therefore suggested to use the sample variance of $w_t$ to approximate it so that (2.10) can be therefore reformulated as

$$N_{ess} = \frac{1}{\sum_{i=1}^{N} \left(w_t^{(i)}\right)^2}. \tag{2.11}$$

From (2.11), we know that the ESS is determined by the distribution of importance weights. So the ESS can be used to measure the degree of degeneracy in the particle filter. A small value of $N_{ess}$ indicates that distribution of weights is quite skewed, which is caused by a severe degeneracy. Liu and Chen (1995, 1998) use this idea to monitor the ESS, and resample when the value falls below a pre-fixed threshold, $N_T$.

Note that the intuitive interpretation of ESS does not hold when resampling is used in the particle filters, as the particles after resampling are no longer independent. However, ESS still gives a natural condition for when to resample. An alternative Monte Carlo procedure for estimating an ESS is given by Carpenter et al. (1999).

## 2.2 Sampling algorithms

The SIR filter can be viewed as using importance sampling to approximate (2.6) with proposal distribution

$$\hat{p}(x_t|\mathbf{y}_{1:t-1}) \approx \sum_{i=1}^{N} w_{t-1}^{(i)} p(x_t|\tilde{x}_{t-1}^{(i)}), \tag{2.12}$$

Samples are generated from this by (i) simulating particles $\tilde{x}_{t-1}^{(i)}$ with weight $w_{t-1}^{(i)}$ in the resampling stage; and (ii) propagation each resampled particle using $p(x_t|x_{t-1})$ in the sampling stage. However, the importance sampling approximation may be poor when $p(x_t|\mathbf{y}_{1:t-1})$ is quite different from $p(x_t|\mathbf{y}_{1:t})$. This case would be happened when the likelihood $p(y_t|x_t)$ is very peaked or when there is little overlap between the likelihood $p(y_t|x_t)$ and the prior approximation $p(x_t|\mathbf{y}_{1:t-1})$. In such cases, particles near the very narrow likelihood peak will be given much bigger weights than any others so that only a small fraction of particles simulated from $p(x_t|\mathbf{y}_{1:t-1})$ will be selected by the resampling. An alternative way of viewing this problem is that since the posterior very closely resembles the likelihood in both

the shape and position, most support of the prior, from which we have simulated, plays a minor role in the support of the posterior, and the corresponding particles are relatively unimportant.

A more fundamental weakness of the SIR filter is that the empirical approximation of (2.12) has poor performance in the tails, so the target distribution can be only poorly approximated when there are outliers.

To overcome these problems, a number of variations to the SIR filter have been developed to improve the filter's performance on the sampling aspect, which include using other importance sampling proposal density (Liu and Chen, 1995; Pitt and Shephard, 1999), using rejection sampling (Hurzeler and Kunsch, 1998) or using MCMC methods (Berzuini et al., 1997). For more separate reviews, see Liu and Chen (1998) and Doucet (1998).

### 2.2.1 Sequential importance sampling

Liu and Chen (1995) suggested to use importance sampling instead, which they call *sequential importance sampling* (SIS), so that the particle $x_t^{(i)}$ is able to be easily generated from a proposal density function $q(x_t|x_{t-1}^{(i)}, y_t)$, so equation (2.6) becomes:

$$\hat{p}(x_t|\mathbf{y}_{1:t}) \propto \sum_{i=1}^{N} \frac{p(y_t|x_t)p(x_t|x_{t-1}^{(i)})w_{t-1}^{(i)}}{q(x_t|x_{t-1}^{(i)}, y_t)} q(x_t|x_{t-1}^{(i)}, y_t), \tag{2.13}$$

then the corresponding weight is

$$w_t^{(i)} \propto w_{t-1}^{(i)} \frac{p(y_t|x_t^{(i)})p(x_t^{(i)}|x_{t-1}^{(i)})}{q(x_t^{(i)}|x_{t-1}^{(i)}, y_t)}. \tag{2.14}$$

The SIS filter is given below:

**Algorithm 2.2** *The SIS filter*

**Initialisation** *Sample* $x_1^{(i)} \sim q(x_1|y_1)$ *for* $i = 1, \ldots, N$ *at time* $t = 1$, *assigning weight* $w_1^{(i)} = p(x_1^{(i)}|y_1)/q(x_1^{(i)}|y_1)$ *to it.*

**Importance sampling (at time** $t$**)** *Assume we have particles* $\left(x_{t-1}^{(i)}, w_{t-1}^{(i)}\right)$, *then sample*

$$x_t^{(i)} \sim q(x_t|x_{t-1}^{(i)}, y_t).$$

*Assign* $x_t^{(i)}$ *a new weight, according to (2.14)*

**End** *Time goes to* $t+1$, *and algorithm goes to importance sampling step.*

Note that if we choose proposal density as

$$q(x_t|x_{t-1}^{(i)}, y_t) = p(x_t|x_{t-1}^{(i)}),$$

the SIS filter becomes the SIR filter without resampling step. If we choose

$$q(x_t|x_{t-1}^{(i)}, y_t) = p(x_t|x_{t-1}^{(i)}, y_t), \tag{2.15}$$

then (2.14) becomes

$$w_t^{(i)} \propto w_{t-1}^{(i)} p(y_t|x_{t-1}^{(i)}). \tag{2.16}$$

So given a value of $x_{t-1}^{(i)}$, the importance weights $w_t^{(i)}$ are the same no matter what values of $x_t$s are drawn from this proposal density function, which amounts to saying that variance of the importance weights conditional on the $x_{t-1}^{(i)}$, i.e. $\text{var}(w_t^{(i)}|x_{t-1}^{(i)})$ is equal to zero. For this reason, (2.15) is known as optimal proposal density function (Arulampalam et al., 2002). However, calculating $p(x_t|x_{t-1}^{(i)}, y_t)$ involves an integration over $x_t$, which is typically impossible in most non-linear

non-Gaussian cases. Alternative choices of the proposal density are available. See Liu and Chen (1998) and references therein for more details.

One fundamental problem with the SIS approach is that no resampling is used. The algorithm will inevitably suffer sample degeneracy. A remedy to this problem is using resampling when the ESS falls below a certain threshold.

### 2.2.2 Auxiliary SIR filter

Pitt and Shephard (1999) also suggested to use importance sampling. Their work known as *auxiliary SIR* (ASIR) filter is motivated from the equation (2.6), which can be re-written as

$$\begin{aligned}\hat{p}(x_t|\mathbf{y}_{1:t}) &\propto \sum_{i=1}^{N} p(y_t|x_t)p(x_t|x_{t-1}^{(i)})w_{t-1}^{(i)} \\ &= \sum_{i=1}^{N} \frac{p(y_t|x_t)p(x_t|x_{t-1}^{(i)})}{\int p(y_t|x_t)p(x_t|x_{t-1}^{(i)})\mathrm{d}x_t} \int p(y_t|x_t)p(x_t|x_{t-1}^{(i)})\mathrm{d}x_t w_{t-1}^{(i)} \\ &= \sum_{i=1}^{N} p(x_t|x_{t-1}^{(i)}, y_t)p(y_t|x_{t-1}^{(i)})w_{t-1}^{(i)}. \end{aligned} \tag{2.17}$$

So the target distribution $p(x_t|\mathbf{y}_{1:t})$ is a mixture distribution with distributions $p(x_t|x_{t-1}^{(i)}, y_t)$, each assigned weights

$$\lambda_{t-1}^{(i)} \propto p(y_t|x_{t-1}^{(i)})w_{t-1}^{(i)} \tag{2.18}$$

The ASIR filter aims to approximate this mixture density and use the approximation as a proposal distribution in an importance sampling method.

A simple implementation of such an importance sampling approach would lead to an $O(N^2)$ algorithm. So Pitt and Shephard (1999) aimed to calculate the joint posterior distribution of both the state $x_t$ and the indices $i$ instead, for which the resulting importance sampling approach only has a computational complexity

$O(N)$. Thus we define

$$\hat{p}(x_t, i|\mathbf{y}_{1:t}) = \lambda_{t-1}^{(i)} p(x_t|x_{t-1}^{(i)}, y_t). \tag{2.19}$$

The corresponding proposal density could be defined as

$$q(x_t, i|\mathbf{y}_{1:t}) = \hat{\lambda}_{t-1}^{(i)} q(x_t|x_{t-1}^{(i)}, y_t), \tag{2.20}$$

where $\hat{\lambda}_{t-1}^{(i)}$ is an approximation to $\lambda_{t-1}^{(i)}$. Thus the marginal posterior of the index is

$$q(i|\mathbf{y}_{1:t}) = \int \hat{\lambda}_{t-1}^{(i)} q(x_t|x_{t-1}^{(i)}, y_t) \mathrm{d}x_t = \hat{\lambda}_{t-1}^{(i)}, \tag{2.21}$$

so we may choose the index $i$ with respect to the weight $\hat{\lambda}_{t-1}^{(i)}$, and then simulate $x_t$ from the transition probability $q(x_t|x_{t-1}^{(i)}, y_t)$ given the index and the latest observation $y_t$. Then each pair $\left(x_t^{(j)}, i^{(j)}\right)$ will be reweighted:

$$\begin{aligned} w_t^{(j)} &\propto \frac{p(x_t^{(j)}, i^{(j)}|\mathbf{y}_{1:t})}{q(x_t^{(j)}, i^{(j)}|\mathbf{y}_{1:t})} \\ &= \frac{\lambda_{t-1}^{(i^{(j)})} p(x_t^{(j)}|x_{t-1}^{(i^{(j)})}, y_t)}{\hat{\lambda}_{t-1}^{(i^{(j)})} q(x_t^{(j)}|x_{t-1}^{(i^{(j)})}, y_t)} \\ &= \frac{w_{t-1}^{(i^{(j)})} p(y_t|x_t^{(j)}) p(x_t^{(j)}|x_{t-1}^{(i^{(j)})})}{\hat{\lambda}_{t-1}^{(i^{(j)})} q(x_t^{(j)}|x_{t-1}^{(i^{(j)})}, y_t)}. \end{aligned} \tag{2.22}$$

The complete ASIR filter is given below:

**Algorithm 2.3** *ASIR filter*

**Preliminary at time** $t-1$ $N$ *weighted particles* $\left(x_{t-1}^{(i)}, w_{t-1}^{(i)}\right)$ *are generated;*

**Selection** *Simulate* $i^{(j)}$ *from* $\{1, 2, \ldots, N\}$ *according to the weight* $\hat{\lambda}_{t-1}^{(j)}$, *for* $j = 1, \ldots, N$;

**Prediction** *Simulate* $x_t^{i(j)}$ *from* $q(x_t|x_{t-1}^{(i(j))}, y_t)$ *independently for* $j = 1, \ldots, N$;

**Filtering** *Assign each pair* $\left(x_t^{(i(j))}, i^{(j)}\right)$ *a weight* $w_t^{(j)}$ *which is (2.22);*

**End** *Time goes to* $t$.

In the non-linear Gaussian state space model, the optimal proposal density, i.e. $q(x_t|x_{t-1}^{(i(j))}, y_t) = p(x_t|x_{t-1}^{(i(j))}, y_t)$ and $\lambda_{t-1}^{(i)} = w_{t-1}^{(i)} p(y_t|x_{t-1}^{(i)})$, can be chosen, then (2.22) is a constant. If we consider $p(y_t|x_t)$ to be log-concave, the $q(x_t|x_{t-1}^{(i(j))}, y_t)$ can be approximated by the optimal density, so near-optimal results are obtained.

More generally, we can take $q(x_t|x_{t-1}^{(i(j))}, y_t) = p(x_t|x_{t-1}^{(i(j))})$, and approximate $\hat{\lambda}_{t-1}^{(i)} = p(y_t|\mu_t^{(i)}) w_{t-1}^{(i)}$ where $\mu_t^{(i)}$ is a statistic of $x_t|x_{t-1}^{(i)}$ such as mean, mode, or a sample, then the new weight is

$$\begin{aligned} w_t^{(j)} &= \frac{w_{t-1}^{(i(j))} p(y_t|x_t^{(j)}) p(x_t^{(j)}|x_{t-1}^{(i(j))})}{\hat{\lambda}_{t-1}^{(i)} q(x_t^{(j)}|x_{t-1}^{(i(j))}, y_t)} \\ &= \frac{p(y_t|x_t^{(j)})}{p(y_t|\mu_t^{(i(j))})} \end{aligned} \tag{2.23}$$

Compared to the SIR and SIS filter, the ASIR filter can produce much less variable weights because there is a preliminary selection of the particles before the resampling step by using the predictive information $p(y_t|\mu_t)$ at time $t-1$ so that the new weight is dependent on where the particles are sampled from and therefore potentially on the whole trajectory through $w_{t-1}^{i(j)}$. Note that the selection step is performing a similar function to resampling, but allowing the resampling probabilities to depend on $y_t$. Thus no further resampling step is needed. The original ASIR filter did include a resampling step (Pitt and Shephard, 1999), but see Carpenter et al. (1999) for an example of how much this can make worse the performance of the filter.

### 2.2.3 Other sampling algorithms

Hurzeler and Kunsch (1998) advocated the use of *rejection sampling* to draw the particles, which has the most attractive property that it produces independent samples. The basic idea of the method is to simulate $x_t^{(i)}$ from a proposal probability $q(x_t|x_{t-1}^{(i)}, \mathbf{y}_{1:t})$ (normally taken to be the transition probability), then accept $x_t^{(i)}$ with probability

$$p_{accept} = \frac{p(y_t|x_t^{(i)})}{\max_{x_t^{(i)}} p(y_t|x_t^{(i)})}.$$

If the $\max_{x_t^{(i)}} p(y_t|x_t^{(i)})$ is not available, an upper bound $c$ of $p(y_t|x_t)$ can be used instead. The likelihood still dominates the selection of particles. But this time, because the rejection sampling is used, it is impossible to know exactly how many particles have to be simulated to achieve the required accuracy.

Berzuini et al. (1997) proposed an MCMC method to simulate the particles from the optimal distribution $p(x_t|x_{t-1}^{(i)}, y_t)$, which they call *Metropolis-Hasting importance resampling* (MHIR). The particle $x_t^{(i)}$ is simulated within a single iteration of any Metropolis-Hastings algorithm (Metropolis et al., 1953; Hasting, 1970) having $p(x_t|x_{t-1}^{(i)}, y_t)$ as its equilibrium distribution, and accept it with a corresponding probability. See the reference for more details.

Like any MCMC method, the MHIR filter always needs a long burn-in and thinning period to make sure that the Markov chain will converge to the target distribution $p(x_t|x_{t-1}^{(i)}, y_t)$. So the MHIR is, in general, less efficient than the SIR filter when the approximation (2.6) to the equilibrium distribution works quite well.

These two methods can be also embedded into the SIS filter and the ASIR filter.

## 2.3 Resampling algorithms

The SIR filter has given an initial impression of the effect of resampling step in evolving the state space model over time. It sampled the particles $x_t^{(i)}$s at each time according to the multinomial distribution of the importance weights $w_t^{(i)}$s. However, as we have mentioned, one disadvantage of the resampling is that it introduces extra random variation to the estimator. Thus, other methods for resampling which introduce less variability have been introduced. Most variance reduction techniques in Monte Carlo integration (see Fishman, 1996, for a complete reference) can be applied in the context of particle filters. The main one is *stratified sampling*; though similar ideas are behind the *residual sampling* of Crisan and Lyons (1997); Liu and Chen (1998).

### 2.3.1 Stratified sampling

A low-variance resampling method is the *stratified sampling* proposed by Carpenter et al. (1999), which is based on the idea of stratification (Cochran, 1963). In the particle filters, assume we have $N$ particles at time $t$ denoted by $x_t^{(i)}$ with weights $w_t^{(i)}$. Consider resampling a set of $M$ particles. The basic idea is to resample particle $x_t^{(i)}$ $N_i$ times, where

$$E(N_i) = Nw_t^{(i)},$$

and $N_i$ takes the value $\left[Nw_t^{(i)}\right]$ or $\left[Nw_t^{(i)}\right] + 1$.

The specific stratified algorithm of sampling $M$ times from $N$ particles is given below. A demonstrative example of this algorithm is also given in Figure 2.1.

**Algorithm 2.4** *Stratified sampling*

*Given that there are $N$ particles with associated weights $\left(x_t^{(i)}, w_t^{(i)}\right)$;*

*Set $U = U(0,1)/M$; $i = 1$;*

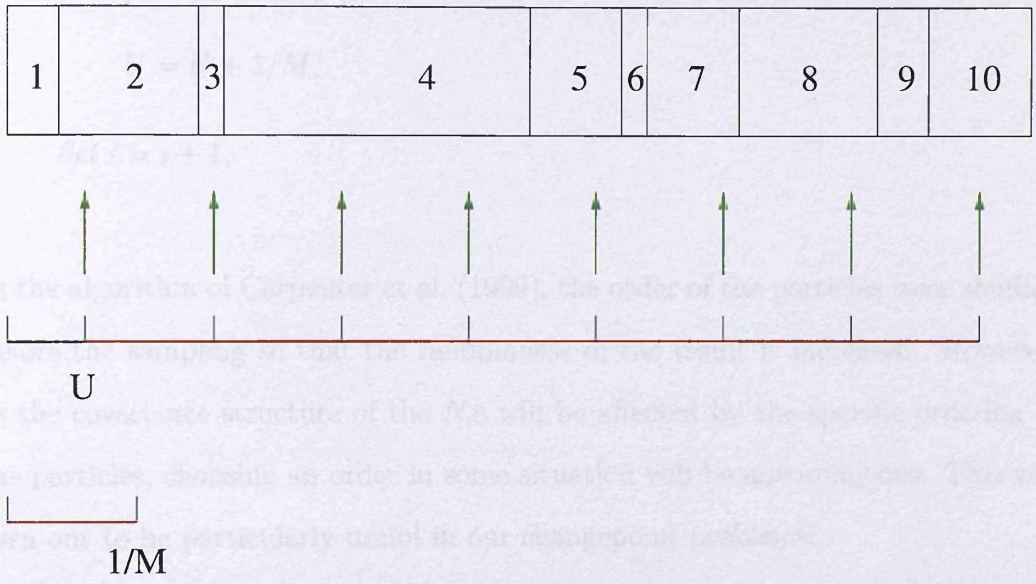


Figure 2.1: The diagram of the stratified sampling with $M = 8$ and $N = 10$. The 10 particles are denoted by the rectangles with different lengths. Each length is proportional to the particle's weight. Firstly a random variable $U$ is uniformly simulated from $U[0, \frac{1}{M}]$. $M$ points are also set in $[0, 1]$ by the algorithm, each a distance $1/M$ apart. For the $i$th particle, the sum of the associated weights of previous particles (i.e. $x_t^{(1)}, \ldots, x_t^{(i)}$) is obtained as $Q_i = \sum_{j=1}^{i} w_t^{(i)}$. Simultaneously, calculate the accumulated value $U_i = U + i/M$. When $U_i$ lies in between $Q_{i-1}$ and $Q_i$, the particle $x_t^{(i)}$ is kept. By this way, the number of times each particle $x_t^{(i)}$ is selected proportional to the weight $w_t^{(i)}$ on average. The selected particles are indicated by the green arrow.

*While* $i < N$

*Set* $U = U - w_t^{(i)}$;

*While* $U < 0$ *then*

*pick particle* $x_t^{(i)}$;

$U = U + 1/M$;

*Set* $i = i + 1$;

In the algorithm of Carpenter et al. (1999), the order of the particles were shuffled before the sampling so that the randomness of the result is increased. However, as the covariance structure of the $N_i$s will be affected by the specific ordering of the particles, choosing an order in some situation will be advantageous. This will turn out to be particularly useful in our changepoint problems.

### 2.3.2 Rejection control

Most resampling methods produce a sample of over-correlated particles, which give a greater Monte Carlo variation. To overcome the problem, Liu et al. (1998) presented an idea of *rejection control* (RC) to combine the rejection sampling with the importance sampling in the state space model, by which independent particles are simulated at each time.

The method is to set a series of threshold value $\alpha_1, \alpha_2, \ldots, \alpha_s$ at some checking points $t_1, t_2, \ldots, t_s$, at each of which the particles $x_{t_s}^{(1)}, \ldots, x_{t_s}^{(N)}$ are drawn from the sampling distribution $q_{t_s}$, with associated weights $w_{t_s}^{(1)}, \ldots, w_{t_s}^{(N)}$, then the following rejection procedure is:

**Algorithm 2.5** *Rejection control algorithm*

**Step 1** *Compute the control threshold* $\alpha_s$ *at time* $t_s$,

**Step 2** *For* $i = 1, \ldots, N$, *accept particle* $x_{t_s}^{(i)}$ *with probability*

$$r_{t_s}^{(i)} = \min\left\{1, w_{t_s}^{(i)}/\alpha_s\right\}$$

**Step 3** *If the* $j$*th particle is accepted, renew its weight as*

$$W_{t_s}^{(i)} = C_n w_{t_s}^{(i)}/r_{t_s}^{(i)},$$

*where* $C_n$ *is a normalising constant.*

**Step 4** *If the* $j$*th particle is rejected, simulate a new particle from time* $t = 0$ *and make sure it passes all the checking points at* $t_1, t_2, \ldots, t_s$. *Failure on passing at any checking time will result in a new simulation from the scratch.*

In step 1, the values of the threshold $\alpha_s$ has to be decided in advance. Although an arbitrary value can be chose, Liu et al. (1998) suggested a particular way to choose these. Step 2 uses the rejection sampling to select the particles. As the variability of weights increase over time, this step together with step 3 will potentially remove particles having low weights. The renewing of weights in step 3 ensures that the resampling is unbiased, that is

$$E(W_{t_s}^{(i)}) = w_{t_s}^{(i)}. \tag{2.24}$$

This can be easily shown. If $w_{t_s}^{(i)} > \alpha_s$ then $W_{t_s}^{(i)} = w_{t_s}^{(i)}$; otherwise,

$$E(W_{t_s}^{(i)}) = r_{t_s}^{(i)}\alpha_s + (1 - r_{t_s}^{(i)}) \times 0 = w_{t_s}^{(i)}.$$

Finally, it is natural, in the case that some particles are deleted, to replenish the particles in step 4, and since the algorithm forces the regeneration of the particle from the beginning instead of the previous time, it guarantees to simulate the

particles independently.

A computational limitation of the RC is that the computing cost increase very rapidly (even exponentially) over time, due to the replenishment in step 4. This could cause the RC method to be very impractical to be implemented in many situations.

Liu et al. (2001) implemented a *partial rejection control* (PRC) method to ease the computation. By partial, they mean the resampling on a rejected particle at time $t_s$ only start from the previous check point $t_{s-1}$ instead of time $t = 0$. Obviously, the PRC doesn't produce the independent samples due to the sampling procedure it uses.

### 2.3.3 Optimal resampling

Most resampling methods in the particle filters produce multiple copies of the particles, according to the associated weights. A typical example of this is the multinomial sampling. However, for situations where (2.6) is a discrete distribution which can be calculated exactly, storing these multiple copies is a waste of memory because all the information in multiple copies of a particle can be carried by one particle where weight is equal to the sum of the weights of the copies of particles. Strictly for such models, (2.6) can be represented exactly by a set of weighted particles. However, resampling is needed as otherwise the number of particles needed will increase exponentially with time.

The *optimal resampling* (OR) proposed by Fearnhead and Clifford (2003) then only eliminates the particles that have very small weights and does not have any multiple copies of a particle. The OR algorithm is optimal in terms of minimising

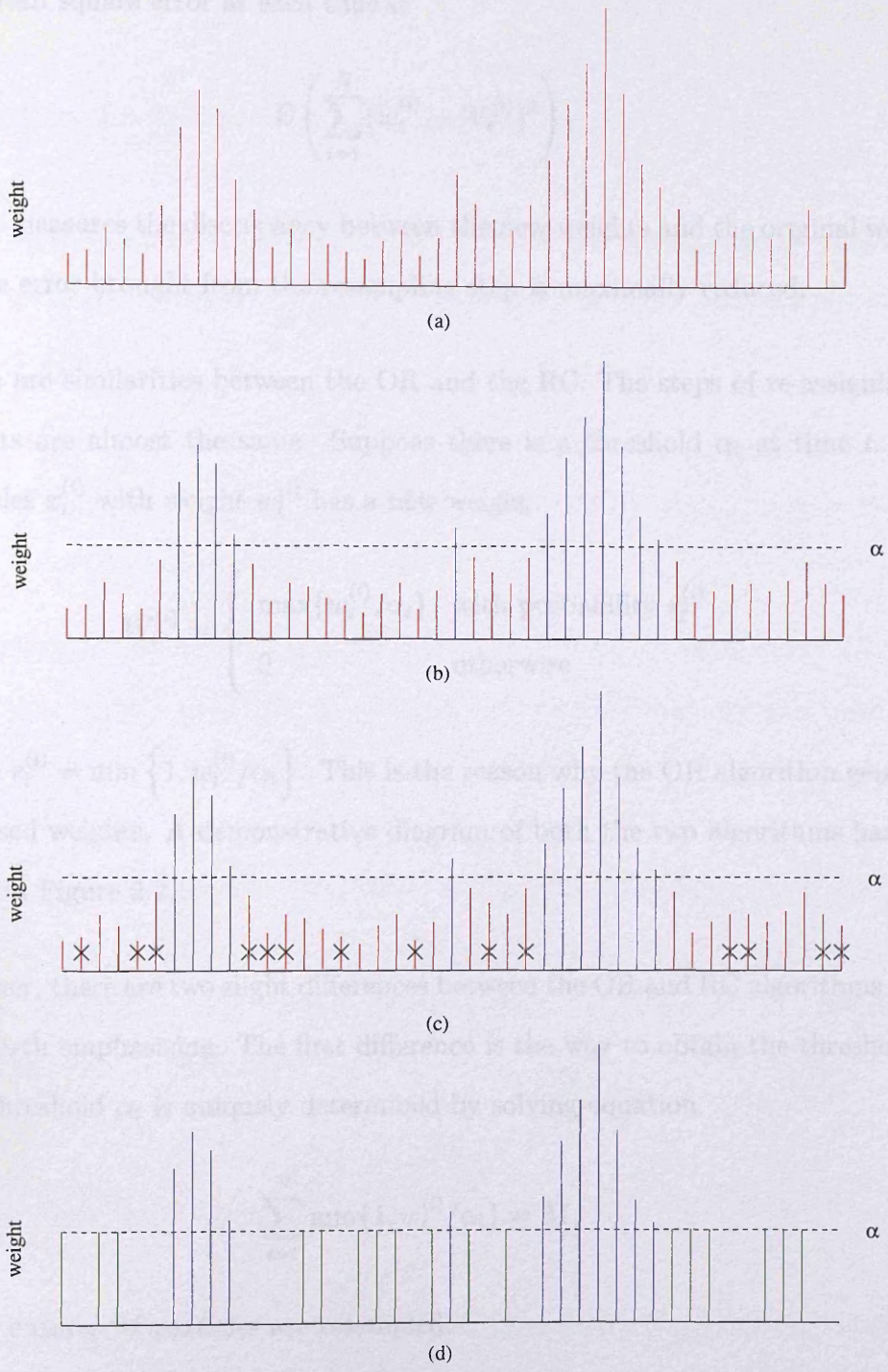


Figure 2.2: Demonstration of the RC and OR algorithm. The original distribution of the weights is given in the plot (a) with the red lines. Then in plot (b), a threshold $\alpha$ (a horizontal dotted line) is decided either arbitrarily or deterministically. The weights greater than $\alpha$ (blue lines) are kept. In plot (c), a resampling is executed, picking some particles randomly, the rest particles are deleted (shown by the cross on the red lines). Plot (d) is a reweighting process, the selected particles are given weights equal to $\alpha$. In the RC algorithm, new particles could then be simulated from time $t = 1$.

the mean square error at each time $t$:

$$\mathrm{E}\left(\sum_{i=1}^{N}(w_t^{(i)} - W_t^{(i)})^2\right), \tag{2.25}$$

which measures the discrepancy between the new weights and the original weights. So the error brought from the resampling step is maximally reduced.

There are similarities between the OR and the RC. The steps of re-assigning the weights are almost the same. Suppose there is a threshold $\alpha_t$ at time $t$. Each particles $x_t^{(i)}$ with weight $w_t^{(i)}$ has a new weight

$$W_t^{(i)} = \begin{cases} \max\{w_t^{(i)}, \alpha_t\} & \text{with probability } r_t^{(i)} \\ 0 & \text{otherwise} \end{cases} \tag{2.26}$$

where $r_t^{(i)} = \min\left\{1, w_t^{(i)}/\alpha_t\right\}$. This is the reason why the OR algorithm generates unbiased weights. A demonstrative diagram of both the two algorithms has been given in Figure 2.2.

However, there are two slight differences between the OR and RC algorithms which are worth emphasizing. The first difference is the way to obtain the threshold $\alpha_t$. The threshold $\alpha_t$ is uniquely determined by solving equation

$$\sum_{i=1}^{N} \min\{1, w_t^{(i)}/\alpha_t\} = M, \tag{2.27}$$

which ensures $M$ particles are resampled.

The second difference is the way to resample the particles. The particles with weights greater than the threshold $\alpha_t$ will be kept. The rest of particles will be resampled by stratified sampling of Carpenter et al. (1999), having new weights equal either $\alpha_t$ or zero.

The specific algorithm sampling $M$ particles from original $N$ particles is listed

below:

**Algorithm 2.6** *Optimal resampling algorithm*

**Step 1** *Consider $N$ particles at time $t$ before the resampling, which have normalised weights $w_t^{(1)}, \ldots, w_t^{(N)}$; compute threshold $\alpha_t$ from the equation (2.27).*

**Step 2** *If $w_t^{(i)} > \alpha_t$ then keep the weights of particles unchanged, i.e. $W_t^{(i)} = w_t^{(i)}$; otherwise, record the particles with weight $w_t^{(i)} \leq \alpha_t$. Suppose there are $L$ such particles;*

**Step 3** *Resample these $L$ particles by the stratified sampling of Carpenter et al. (1999), producing $A = M + L - N$ particles withs weights equal to $\alpha_t$. The rest of particles are given zero weights.*

The *random sampling algorithm* (RSA, Akashi and Kumamoto, 1977) also stores only one copy of the particles, but only one offspring can survive the resampling step no matter how large the associated weight is. Then it leads to a skewness of the weights of the particles.

### 2.3.4 MCMC move

During the calculation of joint posterior distribution $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$, the resampling algorithms mentioned above can only partly solve the sample impoverishment. A more radical approach is to move each particle from its current position to a randomly picked position after resampling. The idea has been first explored by West (1993) through a so-called *adaptive importance sampling*, where the particles are resampled from a kernel density estimate (KDE). A similar method was proposed by Sutherland and Titterington (1994).

An alternative method is the Rao-Blackwellisation (a.k.a marginalisation) of Liu and Chen (1998), in which part of the elements of the states are simulated from the fully conditional distributions, and later Fearnhead (1998) extended it by using a Markov chain to move the particles. Then Gilks and Berzuini (2001) generalised the two methods by embedding a generic MCMC moving step into the resampling stage within a unified particle filter framework. This is called *resample-move* algorithm.

In general, resample-move can only be applied if the particles store the whole path of the state, i.e. $\mathbf{x}_{1:t}^{(i)}$ is a realisation of $\mathbf{x}_{1:t}$. Then at resampling stage, we have:

**Algorithm 2.7** *Resample-move algorithm*

**Step 1** *Given particles* $\mathbf{x}_{1:t}^{(i)}$ *for* $i = 1, \ldots, N$ *obtained at time* $t$*, resample them according to the associated weights* $w_t^{(i)}$.

**Step 2** *Each particle* $\mathbf{x}_{1:t}^{(i)}$ *is moved from the current position to a new random position via one or more iterations of a Markov chain with a transition kernel* $k_t^{(i)}$*:*

$$\mathbf{x}_{1:t}^{new} \sim k_t^{(i)}(\mathbf{x}_{1:t}^{(i)}, \cdot)$$

Any MCMC method such as Gibbs sampling and Metropolis-Hastings can be used to design the kernel $k_t^{(i)}$. Neither burn-in period nor ergodicity in the MCMC move is required for this method because each particle at time $t$ already has approximate marginal distribution $p(x_t|\mathbf{y}_{1:t})$ before the moving.

Generally, the moving step gives an obvious improvement of particle degeneracy at the expense of heavy computation burden though the (fractional move, however, is available, see next chapter for more details).

## 2.4 Smoothing procedure

In the filtering procedure, the conditional distribution of the underlying state $x_t$ is only evaluated given the observations from $y_1$ to $y_t$. In some situations we may wish to calculate $p(x_t|y_{1:l})$, for $l > t$, as this will be more informative about $x_t$. This distribution is generally referred to as a smoothing distribution. There are three smoothing situations:

(i) *Fixed interval smoothing*, where it is $p(x_t|y_{1:n})$ that needs to be calculated for $t = 1, \ldots, n$;

(ii) *Fixed lag smoothing*, where for some fixed value $L > 0$, it is $p(x_t|y_{1:t+L})$ that needs to be calculated for $t = 1, \ldots, n$;

(iii) *Fixed point smoothing*, where for a fixed value $t$, it is $p(x_t|y_{1:t+D})$ that needs to be calculated with $D > t$ increasing.

The focus of the thesis is only on fixed interval smoothing, and we use the term smoothing to refer to the fixed interval smoothing. There are various ways of implementing smoothing with particle filters to be reviewed here.

### 2.4.1 Smoothing by storing particle history

The easiest smoothing method within the particle filters involves storing each particle's history. In the other words, we have to record which particles $x_t^{(i)}$s are produced from the particle $x_{t-1}^{(j)}$ at previous time and so on. $x_{t-1}^{(j)}$ is therefore called as the "ancestor" of the particles $x_t^{(i)}$s (see Figure 2.3). By this way we are actually calculating sequentially the joint posterior distribution of the underlying states $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$.

Hence at time $t+1$, the component of the particle $x_{t+1}^{(i)}$ should be simulated from

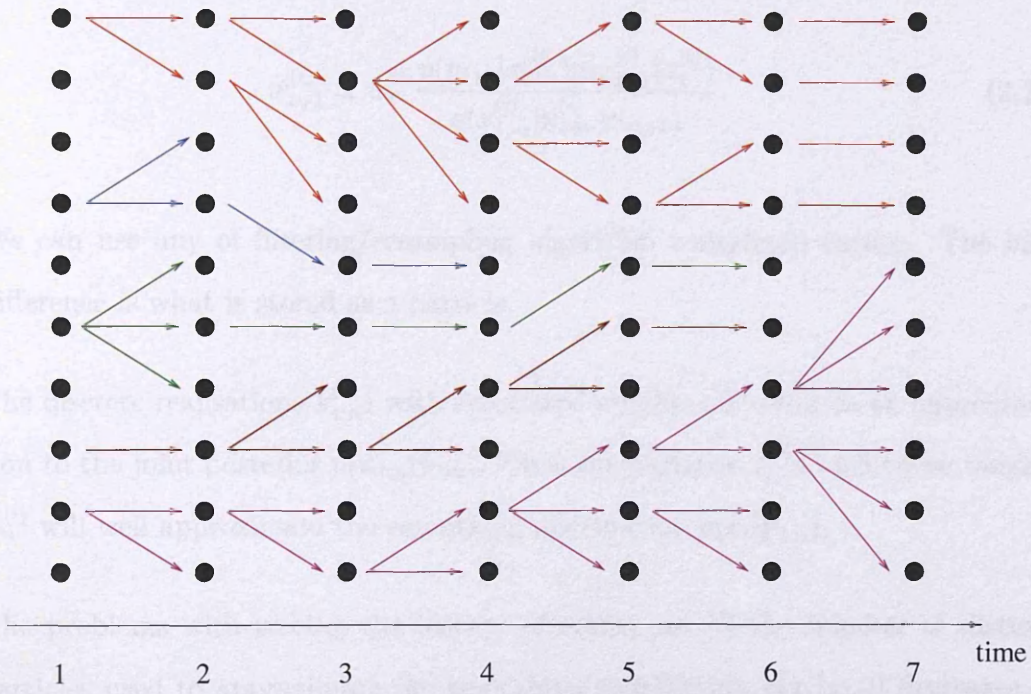


Figure 2.3: A diagram of particle filters running from time 1 to 7. There are 10 dots at each time, which represent 10 particles with different weights. Each particle will give birth to new particles at next time. Only the particles selected by resampling algorithm will survive and are presented at each time. The arrows indicate the relationship between the particle and its off-springs. The same color means the particles have a common ancestor. The history of each particle can be seen very clearly in this way, and we can find that how quickly the number of distinct ancestors of particles reduces as we go back over time. For example at time 7, it is obvious that all the particles are generated by two distinct particles at time 1.

the proposal density function

$$x_{t+1}^{(i)} \sim q(x_{t+1}|\mathbf{x}_{1:t}^{(i)}, \mathbf{y}_{1:t+1}), \tag{2.28}$$

where $\mathbf{x}_{1:t}^{(i)}$ denotes the ancestors of the $x_{t+1}^{(i)}$. The new particle is $\mathbf{x}_{1:t+1}^{(i)} = (x_{t+1}^{(i)}, \mathbf{x}_{1:t}^{(i)})$. So the associated weight is updated as

$$w_{t+1}^{(i)} \propto w_t^{(i)} \frac{p(y_{t+1}|x_{t+1}^{(i)}) p(x_{t+1}^{(i)}|x_t^{(i)})}{q(x_{t+1}^{(i)}|\mathbf{x}_{1:t}^{(i)}, \mathbf{y}_{1:t+1})}. \tag{2.29}$$

We can use any of filtering/resampling algorithm considered earlier. The only difference is what is stored as a particle.

The discrete realisations $\mathbf{x}_{1:n}^{(i)}$s with associated weights $w_n^{(i)}$s will be an approximation to the joint posterior $p(\mathbf{x}_{1:n}|\mathbf{y}_{1:n})$. Thus the particles $x_t^{(i)}$s with those weights $w_n^{(i)}$ will well approximate the smoothing distribution $p(x_t|\mathbf{y}_{1:n})$.

The problems with storing the history of states are (i) the number of distinct particles used to approximate the smoothing distribution $p(x_t|\mathbf{y}_{1:n})$ decreases as time goes (see Figure 2.3). This is a common problem of using resampling in the filtering process of calculating the joint posterior $p(\mathbf{x}_{1:n}|\mathbf{y}_{1:n})$; (ii) storing all the particles' histories will be unrealistic when the data set is huge. For example, approximating $p(x_t|\mathbf{y}_{1:n})$ requires $n$ times as much memory as the SIR filter does. Fix-lag smoothing sometimes can be used to relieve the burden, but is still unable to solve the problem completely.

### 2.4.2 Forward-backward smoothing

A second approach is based on the following relationship (Kitagawa, 1987):

$$\begin{aligned} p(x_t|y_{1:n}) &= \int p(x_t, x_{t+1}|y_{1:n})\mathrm{d}x_{t+1} \\ &= \int p(x_{t+1}|y_{1:n})p(x_t|x_{t+1}, y_{1:n})\mathrm{d}x_{t+1} \\ &= \int p(x_{t+1}|y_{1:n})p(x_t|x_{t+1}, \mathbf{y}_{1:t})\mathrm{d}x_{t+1} \\ &= \int p(x_{t+1}|y_{1:n})\frac{p(x_{t+1}|x_t)p(x_t|\mathbf{y}_{1:t})}{p(x_{t+1}|\mathbf{y}_{1:t})}\mathrm{d}x_{t+1} \\ &= p(x_t|\mathbf{y}_{1:t})\int \frac{p(x_{t+1}|y_{1:n})p(x_{t+1}|x_t)}{p(x_{t+1}|\mathbf{y}_{1:t})}\mathrm{d}x_{t+1} \end{aligned} \tag{2.30}$$

The calculation can be split in two steps. Firstly the filtering density $p(x_t|\mathbf{y}_{1:t})$ can be approximated by the particle filter. The particles will be used to approximate the smoothing density, but the weights assigned to the particles are changed. If we denote the new weights by $w_{t|n}^{(i)}$, then there is a backward recursion that calculates the $w_{t|n}^{(i)}$s from the $w_t^{(i)}$s and $w_{t+1|n}^{(i)}$s. This is the second step of the smoothing method. The particular recursion is the following (Doucet et al., 2000b):

$$w_{t|n}^{(i)} = w_t^{(i)}\left(\sum_{j=1}^{N} w_{t+1|n}^{(j)}\frac{p(x_{t+1}^{(j)}|x_t^{(j)})}{\sum_{l=1}^{N} w_t^{(l)}p(x_{t+1}^{(j)}|x_t^{(l)})}\right) \tag{2.31}$$

where $w_{t|n}^{(i)}$ is the smoothed weight of the $i$th particle at time $t$ conditional on the whole data set. So the smoothing density $p(x_t|\mathbf{y}_{1:n})$ is approximated by:

$$\hat{p}(x_t|\mathbf{y}_{1:n}) = \sum_{i=1}^{N} w_{t|n}^{(i)}\delta(x_t - x_t^{(i)}).$$

The same idea has also been proposed by Hurzeler and Kunsch (1998) by using the rejection sampling method, and Godsill et al. (2004) further extended the algorithm to calculate the joint posterior $p(x_{1:n}|y_{1:n})$.

Similar to the first smoother, the second approach also requires to store all the

history of the states, which costs a great memory storage $O(nN)$, although using fixed lag smoothing can partly overcome the problem. Moreover, to calculate the smoothed weights involves an algorithm with quadratic computational cost $O(nN^2)$, which will be computer prohibitive when data set is very large. However, it can partly overcome the problem of sample impoverishment.

### 2.4.3 Two-filter smoothing

The first two algorithms are completely dependent on the particles of the approximation to the filtering density. If the filtering process gives a poor approximation to that density function, the smoothing density can't be accurately approximated as well. A new smoothing approach is needed.

Motivated by the forward-backward algorithm of Baum et al. (1970) on discrete time HMM, Kitagawa (1996) and Clapp and Godsill (1999) proposed a similar one on the general state space model. It actually runs two independent filter in parallel. One runs in a forward time direction to calculate the filtering distribution $p(x_t|\mathbf{y}_{1:t})$, the other runs in a backward direction to calculate the joint likelihood $p(\mathbf{y}_{t+1:n}|x_t)$. The required posterior distribution $p(x_t|\mathbf{y}_{1:n})$ is a combination of the outputs of the two filters. For this reason, the algorithm is also known as *two filter smoothing* in the signal processing literature. It is derived as follows:

$$\begin{aligned} p(x_t|\mathbf{y}_{1:n}) &= p(x_t|\mathbf{y}_{1:t}, \mathbf{y}_{t+1:n}) \\ &= \frac{p(x_t|\mathbf{y}_{1:t})p(\mathbf{y}_{t+1:n}|\mathbf{y}_{1:t}, x_t)}{p(\mathbf{y}_{t+1:n}|\mathbf{y}_{1:t})} \\ &\propto p(x_t|\mathbf{y}_{1:t})p(\mathbf{y}_{t+1:n}|x_t). \end{aligned} \tag{2.32}$$

The approximation of filtering distribution $p(x_t|\mathbf{y}_{1:t})$ by the particle filter has been well established. So the rest of question is how to calculate the joint likelihood $p(y_{t+1:n}|x_t)$ sequentially in a backward order. Fortunately, this can be done very

easily by factorising the likelihood as follows:

$$\begin{aligned} p(\mathbf{y}_{t:n}|x_t) &= \int p(y_t, \mathbf{y}_{t+1:n}, x_{t+1}|x_t)\mathrm{d}x_{t+1} \\ &= p(y_t|x_t)\int p(\mathbf{y}_{t+1:n}|x_{t+1})p(x_{t+1}|x_t)\mathrm{d}x_{t+1}, \end{aligned} \tag{2.33}$$

and the likelihood therefore can be approximated by a sequential Monte Carlo method with an initial particle approximation to $p(y_n|x_n)$, which is known as *backward filter* (Mayne, 1966).

That is, if at any time $t$, we substitute the particle approximation of the likelihood,

$$\hat{p}(\mathbf{y}_{t+1:n}|x_{t+1}) = \sum_{i=1}^{N} \tilde{w}_{t+1}^{(i)} \delta(x_{t+1} - \tilde{x}_{t+1}^{(i)})$$

into (2.33) then the likelihood can be calculated recursively as

$$\hat{p}(\mathbf{y}_{t:n}|x_t) = \sum_{i=1}^{N} p(y_t|\tilde{x}_t^{(i)})\tilde{w}_{t+1}^{(i)} p(\tilde{x}_{t+1}^{(i)}|\tilde{x}_t^{(i)}), \tag{2.34}$$

Usually, we are able to draw $\tilde{x}_t^{(i)}$ from a proposal density function $q(x_t|\tilde{x}_{t+1}^{(i)}, y_t)$, so the importance weight $\tilde{w}_t^{(i)}$ becomes

$$\tilde{w}_t^{(i)} \propto \tilde{w}_{t+1}^{(i)} \frac{p(y_t|\tilde{x}_t^{(i)})p(\tilde{x}_{t+1}^{(i)}|\tilde{x}_t^{(i)})}{q(\tilde{x}_t^{(i)}|\tilde{x}_{t+1}^{(i)}, y_t)}; \qquad \sum_{i=1}^{N} \tilde{w}_t^{(i)} = 1. \tag{2.35}$$

Proper resampling algorithm is able to be used to reduce the skewness of the distributions of the weights. The following is the whole algorithm:

**Algorithm 2.8** *Backward filter*

**Initialisation** *At time $t = n$, sample $x_n^{(i)}$ from $q(x_n|y_n)$ for $i = 1, \ldots, n$; calculate the weight of each particle as*

$$\tilde{w}_n^{(i)} \propto \frac{p(y_t|\tilde{x}_t^{(i)})}{q(\tilde{x}_t^{(i)}|\tilde{x}_{t+1}^{(i)}, y_t)}; \qquad \sum_{i=1}^{N} \tilde{w}_t^{(i)} = 1.$$

**Filtering** *If* $t \leq n-1$, *sample* $\tilde{x}_t^{(i)}$ *from* $q(x_t|\tilde{x}_{t+1}^{(i)}, y_t)$ *and compute the associated weight* $\tilde{w}_t^{(i)}$ *according to (2.35).*

**Resampling** *Calculate the effective sample size* $N_{ess}$ *by (2.11), if* $N_{ess}$ *is less than a pre-fixed constant, then resample the particles.*

These particles, together with the weighted particles $\left(x_t^{(i)}, w_t^{(i)}\right)$s obtained in the filtering process of calculating $p(x_t|\mathbf{y}_{1:t})$, the smoothing distribution $p(x_t|\mathbf{y}_{1:n})$ is

$$\hat{p}(x_t|\mathbf{y}_{1:n}) = \sum_{j=1}^{N} \tilde{w}_t^{(j)} \sum_{i=1}^{N} w_{t-1}^{(i)} p(\tilde{x}_t^{(j)}|x_{t-1}^{(i)}) \delta(x_t - \tilde{x}_t^{(j)}).$$

The method has again the quadratic computational cost $O(nN^2)$, as the forward-backward smoothing, but has overcome the sample impoverishment. One potential problem is that the integral over $x_t$ might not be finite as the likelihood $p(y_{t:n}|x_t)$ is not a probability density function in terms of $x_t$. If so, (2.33) is unable to be approximated by a swarm of particles (Briers et al., 2004). A solution to the problem is available in the reference.

# Chapter 3

# Changepoints models

## 3.1 Introduction

Changepoints are used to add the flexibility to simple statistical model, as it allows different parts of data set to obey different probability rules. For instance, the data observed before a changepoint follows model $\mathcal{A}$, while the data observed after that changepoint follows model $\mathcal{B}$. Such a model is widely used in the fields of economical, biological and engineering science, for modelling the stock prices, DNA sequences and signals segmentation, amongst many other applications.

A set of specific examples of multiple changepoints models include Gaussian observations with varying mean (Worsley, 1979) or variance (Chen and Gupta, 1997; Johnson et al., 2003); Poisson process with a piece-wise constant rate parameter (Raftery and Akman, 1986; Yang and Kuo, 2001; Ritov et al., 2002); changing linear regression models (Carlin et al., 1992; Lund and Reeves, 2002); and hidden Markov models with time-varying transition matrices (Braun and Muller, 1998).

The incorporation of changepoints makes the model inference much more difficult than a single model. Hinkley (1971) and Smith (1975) proposed a frequentist and

a Baysian method respectively to infer the model. Primarily, this is a difficulty in inferring the number and positions of changepoints - if these were known then inference for the remaining parameters in the model is straightforward.

A relatively simpler model is based on assuming that the number of changepoints $m$ is fixed. However, in most situations, and throughout the thesis, $m$ is treated as a random variable which is unknown. The changepoints problem is therefore generically formulated via a likelihood

$$f(\mathbf{y}_{1:n}|m, \tau^{(m)}, \theta^{(m+1)}), \tag{3.1}$$

where $\tau^{(m)} = (\tau_1, \ldots, \tau_m)$ is the positions of the changepoints occurring on an ordered sequence $x_1 < x_2 < \cdots < x_n$; and $\theta^{(m)} = (\theta_1, \ldots, \theta_m)$ is a vector of parameters associated to each segment. The number of changepoints $m$ dominates the dimension of the model, and can be viewed as a model indicator. The other inferences are given based on it. Deciding the value $m$ involves some model selection techniques.

The thesis focuses on using Bayesian model selection techniques for the multiple changepoints model. The most common Bayesian approaches include: *Bayes factor* (Kass and Raftery, 1995), *Bayesian model averaging* (Hoeting et al., 1999) and *reversible jump Markov chain Monte Carlo* (RJMCMC, Green, 1995) methods. For alternative approach see Shao (1993) for the method for the changepoints model.

The difficulty involved with the Bayesian techniques is how to make the model choice by calculating the posterior distribution of the number of changepoints $p(m|\mathbf{y}_{1:n})$ which is

$$p(m|\mathbf{y}_{1:n}) \propto f(\mathbf{y}_{1:n}|m)p(m), \tag{3.2}$$

where the likelihood function $f(\mathbf{y}_{1:n}|m)$ is an integration over all the parameters and possible values of the changepoints:

$$f(\mathbf{y}_{1:n}|m) = \sum_{\tau^{(m)}} \int_{\theta^{(m)}} f(\mathbf{y}_{1:n}|m, \tau^{(m)}, \theta^{(m)}) p(\theta^{(m)}|m, \tau^{(m)}) \mathrm{d}\theta^{(m)}. \tag{3.3}$$

This quantity is known as *evidence* for the model. However, computing the evidence is essentially infeasible when there are too many combinations of the number and positions of changepoints. Hence, in the thesis, we propose an algorithm to infer the model without the integration of (3.3).

## 3.2 Distribution of changepoints

Suppose there are $n$ observations which are denoted by $y_1, y_2, \ldots, y_n$, and a set of changepoints denoted by $0 < \tau_1 < \tau_2 < \cdots < \tau_m < n$, where the number of changepoints $m$ is unknown. We assume the changepoints only occur at discrete time, $1, \ldots, n-1$.

Then we consider a prior distribution of the changepoints. A natural way is to model the distance between any two successive changepoints, i.e. the length of segment $d = \tau_i - \tau_{i-1}$, for $i = 2, \ldots m$. In the thesis, we adapt the point process of Barry and Hartigan (1993) to model these distances, with a specified probability mass function $g(d)$. Correspondingly, the distribution function of the distance $d$ is defined as $G(d) = \sum_{s=1}^{d} g(s)$. For the first changepoint $\tau_1$, because its position is independent of any other changepoints, we may assign it a different prior $g_0(\tau_1)$.

The point process implies a joint prior distribution of the number and positions of the changepoints:

$$p(\tau^{(m)}) = g_0(\tau_1) \left( \prod_{j=2}^{m} g(\tau_i - \tau_{i-1}) \right) (1 - G(n - \tau_m)). \tag{3.4}$$

A common choice for $g(d)$ is a density function of negative binomial distribution, which is a negative binomial distribution:

$$g(d) = \binom{d-1}{k-1} p^k (1-p)^{d-k}, \qquad d = k, \ldots, n,$$

and $g_0(d)$ is usually of form

$$g_0(d) = \sum_{i=1}^{k} \binom{d-1}{i-1} p^i (1-p)^{d-i}/k, \qquad d = k, \ldots, n,$$

where $p$ is the probability that a changepoint occurs. $p$ is usually pre-fixed through the empirical analysis (see the example of the well log data in Chapter 5). Even if $p$ is unknown, a proper prior can be assigned to it, hence the posterior estimation of it is available (see for example Fearnhead, 2005). This is because the positions of changepoints are uniformly distributed on the interval, independent of $p$, conditional on the number of changepoints (Fearnhead, 2005).

The expectation of $d$ is $k/p$ and the variance is $k(1-p)/p^2$. So the average length of all segments is determined by both $k$ and $p$. Note that

$$\mathrm{var}(d) = \mathrm{E}(d)^2/k - \mathrm{E}(d),$$

which means that given the expectation is fixed, the variability of the lengths of segments are only dependent on the parameter $k$. Then $k$ can be seen as a tuning factor, values of which the number of changepoints can be adjusted. Large values of $k$ can reduce the number of very short segments.

An alternative prior setting is the one of Green (1995). Firstly, they assign a prior $p(m)$ on the number of changepoints $m$, and then given a realisation of $m$, they define a conditional prior on the positions of changepoints, which is $p_m(\tau_i|\tau_{i+1})$, the prior for the position of the $i$th changepoint, given the position of $(i+1)$th changepoint, for $i = 1, \ldots, m-1$. Note that, this setting is basically equivalent to

the point process setting.

## 3.3 Changepoints via state space model

Throughout the thesis, we describe the changepoint problem by a state space model of form (2.1). But the underlying state $X_t$ consists of two components $C_t$ and $B_t$. Normally, $C_t$ is a finite-value process which can be used to describe the changepoints; and $B_t$ is a vector of continuous variables, which are the segment-specified parameters of the likelihood model for observation $Y_t$. Conditional on $C_t$, the parameter and observations satisfy:

$$\begin{aligned} B_t &= \mathbf{f}_t(B_{t-1}, C_t, U_t) \\ Y_t &= \mathbf{g}_t(B_t, V_t). \end{aligned} \tag{3.5}$$

where $U_t$ and $V_t$ are random noises. This is known as *hierarchical state space model* (HSSM). $C_t$ locates in the highest hierarchy in such models, and therefore influences both the transition from $B_{t-1}$ to $B_t$ as well as the observation $Y_t$. Figure 3.1 gives the graphical representation of the dependence structure of a HSSM.

We denote $C_t$ the time of the last changepoint prior to time $t$, then the process of $C_t$ has following point process.

**Theorem 3.3.1** *If the changepoints $\tau_1, \ldots, \tau_m$ follow a point process with probability mass function $g(d)$, where $d = \tau_i - \tau_{i-1}$ for $i = 2, \ldots, m$, then the time of the most recent changepoint prior to time t follows a Markov chain with transition probability* $\Pr(C_{t+1} = i | C_t = j)$ *where*

$$\Pr(C_{t+1} = j | C_t = i) = \begin{cases} \frac{1-G(t-i)}{1-G(t-i-1)} & \text{if } j = i, \\ \frac{G(t-i)-G(t-i-1)}{1-G(t-i-1)} & \text{if } j = t, \\ 0 & \text{otherwise.} \end{cases} \tag{3.6}$$

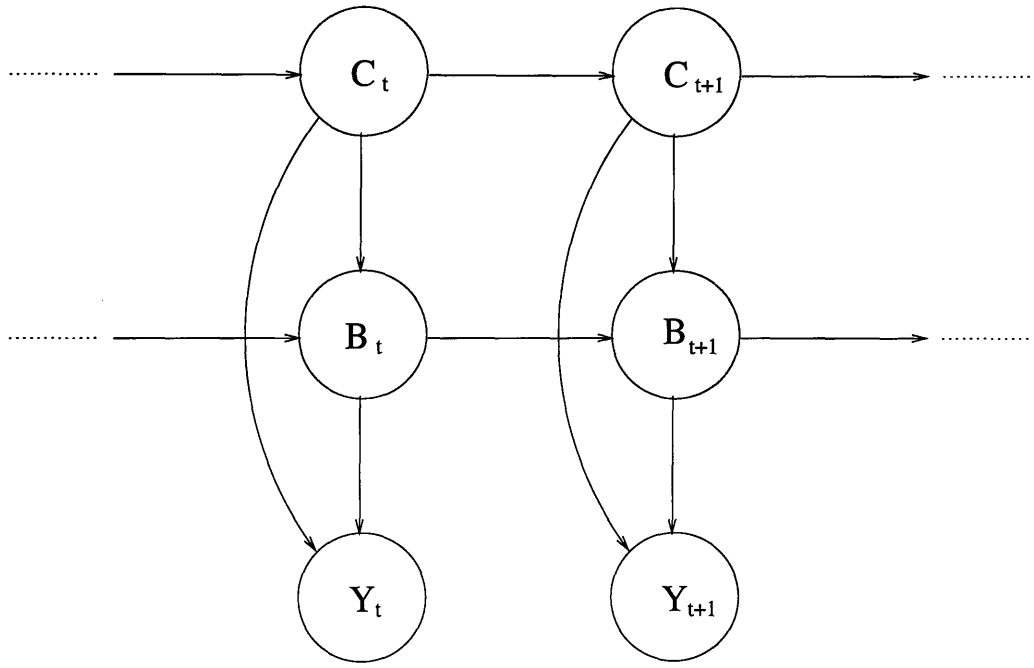


Figure 3.1: A state space model for changepoints problem. The underlying state consists of two components: the changepoint $C_t$ and the parameter $B_t$. $C_t$ stands at the top hierarchy of the model and takes some discrete values in a finite space. So it can be seen as an indicator function. In contrast, $B_t$ and $Y_t$ is modelled can take a value in a continuous space. Conditional on $C_t$, a relationship between $B_t$ and $Y_t$ can be modelled.

**Proof:** Because $C_t$ is the last changepoint prior to time $t$, and can take values up to $t-1$, the probability of $C_t = i$ is $1 - G(t-1-i)$. So when $C_{t+1} = C_t = i$, we have

$$\begin{aligned}\Pr(C_{t+1} = i | C_t = i) &= \frac{\Pr(C_{t+1} = i)}{\Pr(C_t = i)} \\ &= \frac{1 - G(t+1-1-i)}{1 - G(t-1-i)} \\ &= \frac{1 - G(t-i)}{1 - G(t-i-1)}.\end{aligned}$$

Note that conditional on $C_t = i$, $C_{t+1}$ can only take value $i$ and $t$, so we have

$$\begin{aligned}\Pr(C_{t+1} = t | C_t = i) &= 1 - \Pr(C_{t+1} = i | C_t = i) \\ &= 1 - \frac{1 - G(t-i)}{1 - G(t-1-i)} \\ &= \frac{G(t-i) - G(t-i-1)}{1 - G(t-i-1)}.\end{aligned}$$

□

The most simple example of the changepoints model is as follows. If we have a constant parameter per segment such that

$$B_t = \begin{cases} B_{t-1} & \text{if } C_t = C_{t-1}, \\ \mu_t + \sigma_t U_t & \text{otherwise,} \end{cases} \tag{3.7}$$

and the observation satisfy

$$Y_t = B_t + \delta_t V_t, \tag{3.8}$$

where $\mu_t$, $\sigma_t$ and $\delta_t$ are suitably chosen hyper parameters.

$C_t$ is normally a Markov process with transitional probability

$$\Pr(C_t = j | C_{t-1} = i) = \begin{cases} 1-p & \text{if } j = i, \\ p & \text{if } j = t-1. \end{cases} \tag{3.9}$$

This is a special case of choosing $g(d)$ as a geometric distribution, which is equivalently taking $k = 1$ in the negative binomial distribution.

## 3.4 Particle filter approach

With the state space model, it is natural to use particle filters to infer it. However, the standard particle filters will be inefficient in the HSSM because of the high dimensional state space in which a large sample of particles are required to approximate the posterior distribution. A generic method to improve the efficiency of the particle filter is to combine it with the idea of Rao-Blackwellisation (Casella and Robert, 1996). This is known as *Rao-Blackwellised particle filter* (RBPF). The intuitive idea behind it is to reduce the dimension by marginalising out one sub-part of the state space. Please see Doucet (1998); Doucet et al. (2000a); Murphy and Russell (2001) for more examples.

In the changepoints problem, the partition of the state space model is natural, the changepoint $C_t$ and the parameter $B_t$ such that

$$\Pr(C_t, B_t | \mathbf{y}_{1:t}) = \Pr(B_t | C_t, \mathbf{y}_{1:t}) \Pr(C_t | \mathbf{y}_{1:t}). \tag{3.10}$$

We can update $\Pr(C_t|\mathbf{y}_{1:t})$ by particle filtering first, and then $\Pr(B_t|C_t, \mathbf{y}_{1:t})$ conditional on $C_t$. In particular, if $\Pr(B_t|C_t, \mathbf{y}_{1:t})$ has an analytical form, we only need to sample from $\Pr(C_t|\mathbf{y}_{1:t})$, so fewer particles are needed to reach the same accuracy.

### 3.4.1 Mixture Kalman filter

A simplest example of HSSM is a *conditional Gaussian linear state space model* (CGLSSM) in which a :

$$\begin{aligned} B_t &= A(C_t)B_{t-1} + R(C_t)U_t, \\ Y_t &= H(C_t)B_t + S(C_t)V_t, \end{aligned} \tag{3.11}$$

where $A(C_t)$, $H(C_t)$, $R(C_t)$ and $S(C_t)$ are known matrices conditional on $C_t$. $U_t$ and $V_t$ follow a multivariate normal distribution $N(0, I)$. $C_t$ is a Markov random variable with a finite set of values. We consider a general case below, but a special case is the probability structure described in the previous section.

A standard method to inference for the hidden state of CGLSSM is the *mixture Kalman filter* (MKF) proposed by Chen and Liu (2000). They advocated to infer the history of the $C_t$ first, that is $\Pr(C_{1:t}|\mathbf{y}_{1:t})$. It is obvious that conditional on the $C_{1:t}$, the posterior distribution of parameter $B_t$ can be calculated by the Kalman filter. As $C_{1:t}$ take a finite number of values, it is even possible to write down the posterior distribution of $B_t$ as a sum over all possible values of $C_{1:t}$. The problem of this is that the number of terms in the sum will increase exponentially with time. The idea of particle filter is to approximate this sum by one with a fixed number of term; each term correspond to a particle, i.e. a realisation of $C_{1:t}$.

According to (3.10), the joint posterior of $C_{1:t}$ and $B_t$ can be factorised as

$$p(\mathbf{c}_{1:t}, \beta_t|\mathbf{y}_{1:t}) = p(\beta_t|\mathbf{c}_{1:t}, \mathbf{y}_{1:t})p(\mathbf{c}_{1:t}|\mathbf{y}_{1:t}) \tag{3.12}$$

Because given the changepoint $\mathbf{c}_{1:t}$, (3.11) is a Gaussian linear state space model, the $p(\beta_t|\mathbf{c}_{1:t}, \mathbf{y}_{1:t})$ term follows a normal distribution $N(\mu_t(\mathbf{c}_{1:t}), \Sigma_t(\mathbf{c}_{1:t}))$, where $\mu_t(\mathbf{c}_{1:t})$ and $\Sigma_t(\mathbf{c}_{1:t})$ can be recursively calculated by the Kalman filter if $\mathbf{c}_{1:t}$ is

given:

$$\begin{aligned} P_t &= A(c_t^{(i)})\Sigma_{t-1}(c_{t-1}^{(i)})A(c_t^{(i)})^T + R(c_t^{(i)})R(c_t^{(i)})^T, \\ Q_t &= H(c_t^{(i)})P_t H(c_t^{(i)})^T + S(c_t^{(i)})S(c_t^{(i)})^T, \\ \mu_t(\mathbf{c}_{1:t}^{(i)}) &= A(c_t^{(i)})\mu_{t-1}(\mathbf{c}_{1:t-1}^{(i)}) \\ &\quad + P_t H(c_t^{(i)})^T Q_t^{-1}(y_t - H(c_t^{(i)})A(c_t^{(i)})\mu_{t-1}(\mathbf{c}_{1:t-1}^{(i)})), \\ \Sigma_t(\mathbf{c}_{1:t}^{(i)}) &= P_t - P_t H(c_t^{(i)})^T Q_t^{-1} H(c_t^{(i)})P_t; \end{aligned} \tag{3.13}$$

The posterior distribution $p(\mathbf{c}_{1:t}|\mathbf{y}_{1:t})$ is able to be approximated recursively by a standard filtering as:

$$\hat{p}(\mathbf{c}_{1:t}|\mathbf{y}_{1:t-1}) = \sum_{i=1}^{N} p(c_t|c_{t-1}^{(i)})w_{t-1}^{(i)}, \tag{3.14}$$

$$\hat{p}(\mathbf{c}_{1:t}|\mathbf{y}_{1:t}) \propto \sum_{i=1}^{N} p(y_t|c_t^{(i)})p(c_t^{(i)}|c_{t-1}^{(i)})w_{t-1}^{(i)}. \tag{3.15}$$

Then the joint posterior is approximated as

$$\begin{aligned} \hat{p}(\mathbf{c}_{1:t}, \beta_t|\mathbf{y}_{1:t}) &\propto \sum_{i=1}^{N} p(\beta_t^{(i)}|\mathbf{c}_{1:t}^{(i)}, \mathbf{y}_{1:t})p(y_t|c_t^{(i)})p(c_t^{(i)}|c_{t-1}^{(i)})w_{t-1}^{(i)} \\ &= \sum_{i=1}^{N} N\left(\mu_t(\mathbf{c}_{1:t}^{(i)}), \Sigma_t(\mathbf{c}_{1:t}^{(i)})\right) p(y_t|c_t^{(i)})p(c_t^{(i)}|c_{t-1}^{(i)})w_{t-1}^{(i)}; \end{aligned}$$

where the weight of each particle $w_t^{(i)}$ is:

$$w_t^{(i)} \propto p(y_t|c_t^{(i)})p(c_t^{(i)}|c_{t-1}^{(i)})w_{t-1}^{(i)}; \quad \sum_{i=1}^{N} w_t^{(i)} = 1. \tag{3.16}$$

Chen and Liu (2000) adapted the SIS algorithm to obtain these particles, by which each particle $\mathbf{c}_{1:t}^{(i)}$ is simulated from

$$q(c_t|c_{t-1}^{(i)}, y_t) = q(c_t|c_{t-1}^{(i)}, \mu_{t-1}(\mathbf{c}_{1:t-1}^{(i)}), \Sigma_{t-1}(\mathbf{c}_{1:t-1}^{(i)})),$$

for $i = 1, \ldots, N$. Then the importance weight of each particle $w_t^{(i)}$ is sequentially

updated as

$$w_t^{(i)} \propto \frac{p(y_t|c_t^{(i)})p(c_t^{(i)}|c_{t-1}^{(i)})}{q(c_t^{(i)}|c_{t-1}^{(i)}, \mu_{t-1}(\mathbf{c}_{1:t-1}^{(i)}), \Sigma_{t-1}(\mathbf{c}_{1:t-1}^{(i)}))} w_{t-1}^{(i)}; \quad \sum_{i=1}^{N} w_t^{(i)} = 1. \tag{3.17}$$

Finally, a standard resampling step can be used. The detail of the MKF is listed as below:

**Algorithm 3.1** *Mixture Kalman filter*

*At certain time t, suppose we have particles* $\left\{c_{t-1}^{(i)}, \mu_{t-1}(c_{t-1}^{(i)}), \Sigma_{t-1}(c_{t-1}^{(i)})\right\}s$ *with weights* $w_{t-1}^{(i)}s$, *for* $i = 1, \ldots, N$:

**Step 1** *Generate* $\mathbf{c}_{1:t}^{(i)}$ *from* $q(\mathbf{c}_{1:t}|c_{t-1}^{(i)}, \mu_{t-1}(c_{t-1}^{(i)}), \Sigma_{t-1}(c_{t-1}^{(i)}))$;

**Step 2** *Given* $\mathbf{c}_{1:t}^{(i)}$, *update* $\mu_t(\mathbf{c}_{1:t}^{(i)})$ *and* $\Sigma_t(\mathbf{c}_{1:t}^{(i)})$ *by the (3.13);*

**Step 3** *Update the new (normailised) weight as (3.17);*

**Step 4** *Resample* $(\mu_t(\mathbf{c}_{1:t}^{(i)}), \Sigma_t(\mathbf{c}_{1:t}^{(i)}))$ *with the probability proportional to the weights* $w_t^{(i)}$ *if the ESS defined as (2.11) is less than a threshold value.*

An alternative option to MKF is using the optimal resampling method of Fearnhead and Clifford (2003). Every time, the optimal sampling version of the MKF produces $R$ descendants of each previous particle, each for a possible value of $c_t^{(i)}$, so there is no need to use any proposal density function to sample $c_t^{(i)}$. Hence the weights are simply of form (3.16). Consequently, resampling has to be implemented at each time to reduce the number of particles from $RN$ to $N$.

However, the optimal sampling method outperforms the MKF in two aspects: (i) it provides more accurate results in terms of both mean-square and absolute error; and (ii) it is much more efficient when the distribution of the weights are extremely skewed.

### 3.4.2 Rao-Blackwellised particle filter

We consider a general state space model when both system and observation equations are non-linear non-Gaussian. Even the conditional posterior probability $p(\beta_t|c_t, \mathbf{y}_{1:t})$ or $p(\beta_t|\mathbf{c}_{1:t}, \mathbf{y}_{1:t})$ can not be analytically updated. Chopin (2007) then proposed a generic RBPF to infer this kind of model.

As before, the changepoints $C_t$ in the model assumed to follow a Markov process with transitional probability defined as (3.6). The value of parameter $B_t$ is of form (3.7) with constant hyper parameters, i.e. $\mu_i = \mu, \sigma_i = \sigma$ for $i = 1, \ldots, n$. This amounts to simulating a new value of $B_t$ independently from a $N(\mu, \sigma V_t)$ distribution, if there is a new changepoint, denoted by $\xi$; otherwise, the previous value of $B_{t-1}$ will still be used at time $t$.

The observations $y_t$ are driven by these two hidden states together, and a standard particle filter of Gordon et al. (1993) updates the posterior density of $x_t = (c_t, \beta_t)$, $p(x_t|\mathbf{y}_{1:t})$, with a swarm of particles and their associated weights $\left(x_t^{(i)}, w_t^{(i)}\right)$s. Accordingly, the particle $x_t^{(i)}$ is simulated from $p(x_t|x_{t-1}^{(i)})$, which is equivalent to the transition probability of the Markov process of the changepoints, i.e. $p(c_t|c_{t-1})$. The weight simply takes the likelihood: $w_t^{(i)} = p(y_t|x_t^{(i)})$.

However, the algorithm can have an extremely poor performance on this problem because the parameter $\beta_t$ remains constant when there is no changepoint occurring at that time.

#### Rao-Blackwellisation

Similar to the MKF, the standard RBPF gives the following approximation:

$$p(c_t, \beta_t|\mathbf{y}_{1:t}) \propto \sum_{i=1}^{N} p(\beta_t^{(i)}|c_t^{(i)}, \mathbf{y}_{1:t}) p(y_t|c_t^{(i)}) p(c_t^{(i)}|c_{t-1}^{(i)}) w_{t-1}^{(i)},$$

which amounts to drawing a changepoint $c_t^{(i)}$ from a transition density function $p(c_t|c_{t-1}^{(i)})$ first; given a value of $c_t^{(i)}$, the parameter $\beta_t^{(i)}$ is able to be sampled according to (3.7). Finally, assign each particle $\left(c_t^{(i)}, \beta_t^{(i)}\right)$ a weight of form (3.16).

But a more convenient method proposed by Chopin (2007) is not using any proposal strategy. Instead, it produces all possible descendants (which is 2 in this case) of each previous particle with weight as follows:

$$\begin{aligned}
&\left(c_t^{(i)} = t-1, \beta_t^{(i)} = \xi^{(i)}\right), \quad w_t^{(i,1)} \propto w_{t-1}^{(i)} p(y_t|\xi^{(i)}) p(c_t^{(i)}|c_{t-1}^{(i)}); \\
&\left(c_t^{(i)} = c_{t-1}^{(i)}, \beta_t^{(i)} = \beta_{t-1}^{(i)}\right), \quad w_t^{(i,2)} \propto w_{t-1}^{(i)} p(y_t|\beta_{t-1}^{(i)}) p(c_t^{(i)}|c_{t-1}^{(i)}).
\end{aligned}$$

The weights are computed by (3.16). Note that $p(y_t|\beta_t^{(i)}) = p(y_t|c_t^{(i)})$ because the parameter has consistent change with the changepoint. The algorithm produces $2N$ particles at each time, so a resampling scheme has to be implemented every time, not only to reduce the degeneracy effect, but also decrease the number of particles from $2N$ to $N$. Chopin (2007) used the multinomial sampling so that all the particles after the resampling are equally weighted as $1/N$. Resampling in this way is thought to have exactly the same computational cost of using optimal proposal density.

### MCMC move

We can further increase the diversity of the particles by moving them through a MCMC kernel with the idea of Gilks and Berzuini (2001). However, it seems generally difficult to build a MCMC kernel whose equilibrium distribution is $p(x_t|\mathbf{y}_{1:t})$. Even if we are able to do so, the computational cost of moving will be presumably $O(t)$.

Hence, Chopin (2007) proposed a fractional MCMC move on the parameter $\beta_t$

only through a MCMC kernel with equilibrium distribution:

$$\eta_t^{(i)}(\beta_t) = p(\beta_t|c_t^{(i)}, \mathbf{y}_{1:t}) \propto \pi(\beta_t) \prod_{k=c_t^{(i)}+1}^{t} p(y_k|\beta_k), \tag{3.18}$$

where $\pi(\cdot)$ denotes the prior distribution of $\beta_t$s, i.e. $N(\xi^{(i)}, \sigma V_t)$. Note that the observations $y_k$s belong to a same segment, such that all $\beta_k$s are equal to $\beta_t$. There is no need to move $c_t$ due to that the $c_t$ is a discrete variable and takes on a finite number of values. Therefore, a great computational cost is reduced.

The move of the $N$ particles in the algorithm can be computationally intensive with cost that is $O(\sum_{i=1}^{N}(t-c_t^{(i)}))$, so the author suggested to move only a portion of particles, contrary to a complete move by Gilks and Berzuini (2001). The reason for the fractional move, according to the author, is that the increase of the number of observations in a segment tends to decrease the degeneracy of the particles.

Thus a constant $T$ can be pre-determined, and a subset $S$ particles has been chosen such that

$$\sum_{i\in S}(t - c_t^{(i)}) \leq T. \tag{3.19}$$

The $s$ particles are selected randomly from the whole particles without replacement until (3.19) is satisfied. The computational cost of the algorithm is $O(T)$. The whole algorithm is presented as below, and more theoretical details of the algorithm can be found in Chopin (2002).

**Algorithm 3.2** *RBPF with fractional move*

**Sampling** *For each particle* $x_{t-1}^{(i)} = (c_{t-1}^{(i)}, \beta_{t-1}^{(i)})$, *for* $i = 1, \ldots, N$, *produce*

$$\begin{aligned} x_t^{(i,1)} &= (t-1, \xi^{(i)}), \\ x_t^{(i,2)} &= (c_{t-1}^{(i)}, \beta_{t-1}^{(i)}); \end{aligned}$$

**Reweighting** *Assign weights to each particle* $x_t^{(i,1)}$ *and* $x_t^{(i,2)}$*:*

$$\begin{aligned} w_t^{(i,1)} &= p(y_t|\xi^{(i)})p, \\ w_t^{(i,2)} &= p(y_t|\beta_{t-1}^{(i)})(1-p); \end{aligned}$$

**Resampling** *Resample the* $2N$ *particles according to the weights* $w_t^{(i,1)}$ *and* $w_t^{(i,2)}$ *by multinomial sampling to obtain* $N$ *particles.*

**Moving** *Select a subset of* $S$ *resampled particles satisfying (3.19), and move the particles through a Markov chain with equilibrium distribution (3.18). The transition kernel of the Markov chain is*

$$\tilde{\beta}_t^{(i)} \sim k_t^{(i)}(\beta_t^{(i)}, \cdot)$$

**End** $t \leftarrow t+1$. *Go to the sampling step.*

### Choices of the transition kernel

There are several options for the transition kernel $k_t^{(i)}(\beta_t^{(i)}, \cdot)$. Gibbs sampling is the easiest one, but only available when the fully conditional distributions are able to be analytically obtained. Unfortunately, most of time series models do not provide the closed form of the conditional distribution, and Chopin (2007) considered a generic Metropolis-Hasting update instead such that

$$\tilde{\beta}_t^{(i)} = \begin{cases} \xi^{(i)} & \text{with probability } \min\left(1, \frac{q(\beta_t^{(i)}|\xi^{(i)})\eta_t^{(i)}(\xi^{(i)})}{q(\xi^{(i)}|\beta_t^{(i)})\eta_t^{(i)}(\beta_t^{(i)})}\right), \\ \beta_t^{(i)} & \text{otherwise,} \end{cases}$$

where $q(\cdot|\cdot)$ is the proposal distribution, and $\xi^{(i)} \sim q(\xi|\beta_t^{(i)})$. A simple choice of $q(\cdot|\cdot)$ is a Gaussian random walk with mean $\beta_t^{(i)}$ and variance $\gamma^2\hat{\Sigma}_t$. The $\hat{\Sigma}_t$ is an

empirical estimation from the simulated value of $\beta_t$:

$$\hat{\Sigma}_t = \frac{1}{N}\sum_{i=1}^{N}\beta_t^{(i)}(\beta_t^{(i)})^T - (\frac{1}{N}\sum_{i=1}^{N}\beta_t^{(i)})(\frac{1}{N}\sum_{i=1}^{N}\beta_t^{(i)})^T,$$

and $\gamma$ is a tuning parameter to adjust the step of the jump.

Another proposal distribution suggested by Chopin (2007) is the so-called Langevin proposal strategy, in the case of which, the proposal distribution is

$$q(\xi|\beta_t^{(i)}) = N\left(\beta_t^{(i)} + \frac{1}{2}\{H_t^{(i)}\}^{1/2}\nabla \log \eta_t^{(i)}(\beta_t^{(i)}), H_t^{(i)}\right),$$

where $\nabla \log \eta_t^{(i)}$ is the gradient function of $\log \eta_t^{(i)}$, and

$$H_t^{(i)} = -\gamma^2\{\nabla'\nabla \log \eta_t^{(i)}(\beta_t^{(i)})\}^{-1},$$

which is a product of $-\gamma^2$ and the Hessian matrix of $\log \eta_t^{(i)}$ at point $\beta_t^{(i)}$. Both of the two proposals have almost equal performances on the time series model.

### Backward smoothing

We usually assume in the changepoint model that the data across the segments are independent of each other. So the observations after a changepoint $\tau$ don't contain any information of the changepoints prior to that changepoint, i.e.

$$p(\mathbf{y}_{1:n}|\text{changepoint at } \tau) = p(\mathbf{y}_{1:\tau}|\text{changepoint at } \tau)p(\mathbf{y}_{\tau+1:n}|\text{changepoint at } \tau)$$

Conditional on the changepoint at $\tau$:

$$\begin{aligned} p(x_\tau|\mathbf{y}_{1:n}) &\propto p(x_\tau|\mathbf{y}_{1:\tau})p(x_{\tau+1}|x_\tau) \\ &= p(x_\tau|\mathbf{y}_{1:\tau})p(c_{\tau+1} = \tau|c_\tau) \end{aligned} \tag{3.20}$$

The last equality comes because the parameters in segments are independent of each other. Hence, there is no further calculation of the smoothing distribution $p(x_t|y_{1:n})$ for any $t$.

Note that if we take a geometric distribution for the length of segment, then the transitional probability is constant, i.e. $p(\tau_{k+1} = \tau|\tau_k) = p$ for all values of $\tau$, and thus $p(x_\tau|\mathbf{y}_{1:n}) \propto p(x_\tau|\mathbf{y}_{1:\tau})$.

The algorithm starts at time $n$, at which $N$ particles $x_n^{(i)}$ are sampled from $p(x_n|y_{1:n})$ straightforwardly. $x_n^{(i)}$ will give where the most recent changepoint $c_n^{(i)}$ is and what the parameter $\beta_n^{(i)}$ is in the segment. Given $c_n^{(i)}$, the last changepoint prior to it together with the corresponding parameter $\beta_{c_n^{(i)}}$ can be simulated from $p(x_{c_n^{(i)}}|y_{1:c_n^{(i)}})$. The backward recursion will go on until time $t = 0$ is reached. The algorithm has a computational cost of $O(n)$ while the traditional smoothing algorithm (Kitagawa, 1996; Godsill et al., 2004) have a computational cost of $O(n^2)$.

**Algorithm 3.3** *Backward smoothing with geometric distribution of segment length*

*Set* $k = n$;

*While* $k > 0$

*(i) Sample* $x_k^{(i)}$ *from* $p(x_k|y_{1:k})$ *for* $i = 1, \ldots, M$;

*(ii)* $k \leftarrow c_k^{(i)}$ *for* $i = 1, \ldots, M$;

The algorithm easily generates to other distribution of segment length: all that changes is the distribution from which $x_k^{(i)}$ is simulated in step (i).

# Chapter 4

# On-line inference for multiple changepoint problems

## 4.1 Introduction

Changepoint models are commonly used to model heterogeneity of data (usually over time). Given a set of observations collected over time, these models introduce a (potentially random) number of changepoints which split the data into a set of disjoint segments. It is then assumed that the data arise from a single model within each segment, but with different models across the segments. Examples of changepoint problems include Poisson processes with changing intensity (Ritov et al., 2002), changing linear regressions (Lund and Reeves, 2002), Gaussian models with changing variance (Johnson et al., 2003) and Markov models with time-varying transition matrices (Braun and Muller, 1998). These models have been applied to problems in a range of areas, including engineering, physical and biological sciences and finance.

We consider Bayesian inference for changepoint models where the number of changepoints is unknown. The most common approach to such inference for these

models is the use of Markov chain Monte Carlo (MCMC; see for example Chib, 1998; Stephens, 1994), and reversible jump MCMC (Green, 1995). However, for many changepoint models (for example the models in Green, 1995; Punskaya et al., 2002) it is possible to simulate independent realisations directly from the posterior distribution. This idea was used for DNA segmentation by Liu and Lawrence (1999), and has been proposed more generally by Fearnhead (2006) and Fearnhead (2005). The idea for direct simulation is based on exact methods for calculating posterior means (Barry and Hartigan, 1992). The advantages of direct simulation methods over MCMC and reversible jump MCMC are that (i) there is no need to diagnose whether the MCMC algorithm has converged; and (ii) as the draws from the posterior distribution are independent it is straightforward to quantify uncertainty in estimates of features of the posterior distributions based on them. For examples of the potential difficulties with MCMC caused by (i), compare the inferences obtained for the Coal-mining disaster data analysed in Green (1995) with those based on the direct simulation method of Fearnhead (2006).

In this chapter we extend the direct simulation algorithms to on-line problems; where the data is obtained incrementally over time, and new inferences are required each time an observation is made. The use of on-line algorithms has also been suggested for static problems (e.g. Chopin, 2002; Del Moral et al., 2006).

The computational cost of our exact on-line algorithm increases linearly over time, however the on-line version of direct simulation is similar to particle filter algorithms, and we consider using resampling algorithms taken from particle filters to reduce the computational cost of our direct simulation algorithm (at the expense of introducing error). We propose two simple extensions of existing resampling algorithms that are particularly well-suited to changepoint models. One is a stratified extension of the rejection-control approach of Liu et al. (1998), which can limit the maximum amount of error introduced by each resampling step. In simulation studies we find this stratified version can reduce the error of the resulting

algorithm by about one third as compared to the non-stratified version.

The resulting on-line algorithm can be viewed as a Rao-Blackwellised version of the Particle Filter algorithm of Chopin (2007): we have integrated out the parameters associated with each segment. Note that this is an extremely efficient version of Rao-Blackwellisation. For example, consider analysing $n$ data points. Our algorithm with $n$ particles will give exact inference and thus will always outperform the algorithm of Chopin (2007) regardless of the number of particles used in that particle filter. Note however, that the filter of Chopin (2007) can be used more widely, as it does not require that the parameters within each segment can be integrated out.

We apply our on-line algorithm to the problem of segmentation of DNA on the basis of C+G content. This is an example of an application which can involve large data sets, and one of the primary aims of our analysis is to test whether it is practicable to perform Bayesian analysis of such data using the on-line direct simulation algorithm.

The outline of the chapter is as follows. We introduce the class of changepoint models we consider in Section 4.2. In Section 4.3 we introduce the on-line direct simulation algorithm, and approximate versions of it that utilise various resampling ideas. We test these approximate algorithms, and compare different resampling algorithms, on simulated data in Section 4.4 before applying our algorithm to the analysis of the C+G structure of human DNA data (Section 4.5). The chpater concludes with a discussion.

## 4.2 Models and Notations

Assume we have data $\mathbf{y}_{1:n} = (y_1, y_2, \ldots, y_n)$. We consider changepoint models for the data with the following conditional independence property: given the position

of a changepoint, the data before that changepoint is independent of the data after the changepoint. These models can be described in terms of the following hierarchical structure.

Firstly we model the changepoint positions via a Markov process. This Markov process is determined by a set of transition probabilities,

$$p(\text{next changepoint at } t|\text{changepoint at } s). \tag{4.1}$$

For this chapter we make the simplification that these transition probabilities depend only on the distance between the two changepoints. Extending our work to the general case, where the distribution of the length of a segment could depend on the time at which it starts, is straightforward . Specifically we let $g(\cdot)$ be the probability mass function for the distance between two successive changepoints (equivalently the length of segments); so that (4.1) is $g(t-s)$. We further let $G(l) = \sum_{i=1}^{l} g(i)$ be the distribution function of this distance, and assume that $g(\cdot)$ is the probability mass function for the position of the first changepoint.

Note that any such model implies a prior distribution on the number of changepoints. For example if a geometric distribution is used for $g(\cdot)$, then our model implies that there is a fixed probability of a changepoint at any time-point, independent of other changepoints. Hence this model implies a binomial distribution for the number of changepoints.

Now we condition on $m$ changepoints at times $\tau_1, \tau_2, \ldots, \tau_m$. We let $\tau_0 = 0$ and $\tau_{m+1} = n$, so our changepoints define $m+1$ segments, with segment $i$ consisting of observations $\mathbf{y}_{\tau_i+1:\tau_{i+1}}$ for $i = 0, \ldots, m$. We allow a set of $\bar{p}$ possible models for the data from each segment, labeled $\{1, 2, \ldots, \bar{p}\}$, and assume an arbitrary prior distribution for models, common across segments, with the model in a given segment being independent of the models in all other segments.

For a segment $k$ consisting of observations $\mathbf{y}_{s+1:t}$ and model $q_k$ we will have a set of unknown parameters, $\beta_k$ say. We have a prior distribution, $\pi(\beta_k)$ for $\beta_k$ (which may depend on $q_k$), but assume that the parameters for this segment are independent of the parameters in other segments. Finally we define

$$\begin{aligned} P(s,t,q_k) &= p(\mathbf{y}_{s+1:t}|s+1:t \text{ is a segment with model } q) \\ &= \int p(\mathbf{y}_{s+1:t}|\beta_k, \text{model } q_k)\pi(\beta_k)\mathrm{d}\beta_k, \end{aligned} \tag{4.2}$$

and assume that these probabilities can be calculated for all $s < t$ and $q_k$. This requires either conjugate priors for $\beta_k$, or the use of numerical integration. Numerical integration is often computationally feasible in practice if the dimension of the part of $\beta_k$ that cannot be integrated analytically is low (see Fearnhead, 2006, for an example).

For concreteness we describe a specific example of this model which we will use in Section 4.4 (see also Punskaya et al., 2002; Fearnhead, 2005). Here we model the data as piecewise linear regressions. So within a segment we have a linear regression model of unknown order. For a given model order $q_k$, we have

$$\mathbf{y}_{s+1:t} = \mathbf{H}_k\beta_k + \varepsilon \tag{4.3}$$

where $\mathbf{H}_k$ is a $(t-s) \times q_k$ matrix of basis functions, $\beta_k$ is a vector of $q_k$ regression parameters and $\varepsilon$ is a vector of iid Gaussian noise with mean 0 and variance $\sigma_k^2$.

We assume conjugate priors. The variance of the Gaussian noise has an inverse gamma distribution with parameters $\nu/2$ and $\gamma/2$, and the components of the regression vector have independent Gaussian priors. The prior for the $j$th component is Gaussian with mean 0 and variance $\sigma_k^2\delta_j^2$.

The likelihood function of $\mathbf{y}_{s+1:t}$ conditional on a model $q_k$ is obtained by integrat-

ing out the regression parameters $\beta_k$ and variance $\sigma_k^2$ :

$$P(s,t,q_k) = \pi^{-(t-s)/2}\left(\frac{|\mathbf{M}_k|}{|\mathbf{D}_k|}\right)^{\frac{1}{2}} \frac{(\gamma)^{\nu/2}}{(||\mathbf{y}_{s+1:t}||^2_{\mathbf{P}_k}+\gamma)^{(t-s+\nu)/2}} \frac{\Gamma((t-s+\nu)/2)}{\Gamma(\nu/2)} \quad (4.4)$$

where:

$$\begin{aligned} \mathbf{M}_k &= (\mathbf{H}_k^T\mathbf{H}_k + \mathbf{D}_k^{-1})^{-1}, \\ \mathbf{P}_k &= (\mathbf{I}_k - \mathbf{H}_k\mathbf{M}_k\mathbf{H}_k^T), \\ ||\mathbf{y}||^2_{\mathbf{P}_k} &= \mathbf{y}^T\mathbf{P}_k\mathbf{y}, \end{aligned}$$

Moreover, $\mathbf{D}_k = \text{diag}(\delta_1^2, \ldots, \delta_{q_k}^2)$ and $\mathbf{I}_k$ is a $(t-s) \times (t-s)$ identity matrix. See Appendix A for full calculation.

## 4.3 On-line Inference

We consider on-line inference for the multiple changepoint model of Section 4.2. We assume that observations accrue over time, so that $y_t$ is the observation at time $t$. At each time-step, our aim is to calculate the posterior distributions of interest based on all the observations to date. To do this efficiently requires updating our posterior distributions at the previous time-step to take account of the new observation. Note that on-line algorithms can be used to analyse batch data by introducing an artificial time for each observation.

We focus on on-line inference of the position of the changepoints. Under the modeling assumptions of Section 4.2, inference for the parameters conditional on knowing the number and position of the changepoints is straightforward. We first describe an exact on-line algorithm, which is an on-line version of the direct simulation method of Fearnhead (2005). The computational cost of this exact algorithm increases over time, so we then present an approximate on-line algorithm, which

uses resampling ideas from particle filters, and which has constant computational cost over time.

### 4.3.1 Exact on-line Inference

We introduce a state at time $t$, $C_t$, which is defined to be the time of the most recent change-point prior to $t$ (with $C_t = 0$ if there have been no change-points before time $t$). Initially we focus on calculating the posterior distribution for $C_t$ given the observation $\mathbf{y}_{1:t}$. We then describe how, if these distributions are stored for all $t$, it is straightforward to simulate from the joint posterior distribution of the position of all changepoints prior to the current time.

#### Filtering Recursions

The state $C_t$ can take values in $0, 1, \ldots, t-1$, and $C_1, C_2, \ldots, C_t, \ldots$ is a Markov chain. Conditional on $C_t = j$, either $C_{t+1} = j$, which corresponds to no change-point at time $t$, or $C_{t+1} = t$, if there is a changepoint at time $t$. The transition probabilities for this Markov chain can thus be calculated as:

$$p(C_{t+1} = j | C_t = i) = \begin{cases} \frac{1-G(t-i)}{1-G(t-i-1)} & \text{if } j = i, \\ \frac{G(t-i)-G(t-i-1)}{1-G(t-i-1)} & \text{if } j = t, \\ 0 & \text{otherwise,} \end{cases} \tag{4.5}$$

where $G(\cdot)$ is the distribution function of distance between two successive change points.

Now, standard filtering recursions give

$$p(C_{t+1} = j | \mathbf{y}_{1:t+1}) \propto p(y_{t+1} | C_{t+1} = j, \mathbf{y}_{1:t}) p(C_{t+1} = j | \mathbf{y}_{1:t}),$$

and

$$p(C_{t+1} = j|\mathbf{y}_{1:t}) = \sum_{i=0}^{t-1} p(C_{t+1} = j|C_t = i)p(C_t = i|\mathbf{y}_{1:t}).$$

Thus, if we let $w_{t+1}^{(j)} = p(y_{t+1}|C_{t+1} = j, \mathbf{y}_{1:t})$, and substitute in the transition probabilities (4.5), we obtain

$$p(C_{t+1} = j|\mathbf{y}_{1:t+1}) \propto \begin{cases} w_{t+1}^{(j)} \frac{1-G(t-j)}{1-G(t-j-1)} p(C_t = j|\mathbf{y}_{1:t}) & \text{if } j < t, \\ w_{t+1}^{(t)} \sum_{i=0}^{t-1} \left( \frac{G(t-i)-G(t-i-1)}{1-G(t-i-1)} p(C_t = i|\mathbf{y}_{1:t}) \right) & \text{if } j = t. \end{cases}$$

If we define $P(s, t, q_k)$ as in (4.3) then we get for $j < t$

$$\begin{aligned} w_{t+1}^{(j)} &= \frac{p(\mathbf{y}_{(j+1):(t+1)}|C_{t+1} = j)}{p(\mathbf{y}_{(i+1):t}|C_{t+1} = j)} \\ &= \frac{\sum_{q_k=1}^{\bar{p}} P(j, t+1, q_k)p(q_k)}{\sum_{q_k=1}^{\bar{p}} P(j, t, q_k)p(q_k)}, \end{aligned} \tag{4.6}$$

while if $j = t$ then the weight is $\sum_{q_k=1}^{\bar{p}} P(t, t+1, q_k)p(q_k)$.

In most situations, such as for the linear regression models described in Section 2, the incremental weights $w_{t+1}^{(j)}$ can be calculated efficiently, as each $P(j, t, q_k)$ depends on a set of summary statistics of the observations $y_{j+1:t}$, which can be updated recursively. Efficient calculation of the incremental weights $w_{t+1}^{(j)}$ is possible by storing these summary statistics for each set of values for the time of the last changepoint, $j$, and the model for the current segment, $q_k$. These can be updated to give the summary statistics required for each $P(j, t+1, q_k)$. Thus the computational cost of calculating each $w_{t+1}^{(j)}$ is fixed, and does not increase with $t - j$.

### Simulating Changepoints

If we store the filtering densities $p(C_t|\mathbf{y}_{1:t})$ for all $t = 1, \ldots, n$, then it straightforward to simulate from the joint posterior distribution of the position of all changepoints prior to time $n$, using the idea of Chopin (2007). The smoothing distribution, according to Section 3.4.2 is:

$$p(C_t|\mathbf{y}_{1:n}) \propto p(C_t|\mathbf{y}_{1:t})p(\text{next changepoint occurs in } t+1, \ldots, n|C_t),$$

where $\tau$ is the first changepoint after time $t$. To simulate one realisation from this joint density:

(1) Set $t_0 = n$, and $k = 0$

(2) Simulate $t_{k+1}$ from the density $p(C_{t_k}|\mathbf{y}_{1:t_k})p(C_{t_{k+1}} = t_k|C_{t_k})$, and set $k = k + 1$.

(3) If $t_k > 0$ return to (2); otherwise output the set of simulated changepoints, $t_{k-1}, t_{k-2}, \ldots, t_1$.

A simple extension of this algorithm which enables a large sample of realisations of sets of changepoints to be simulated efficiently is described in Fearnhead (2006).

### MAP estimation

We can obtain an on-line Viterbi algorithm for calculating the maximum a posteriori (MAP) estimate of the positions of the changepoints and the model orders for each segment as follows. We define $\mathcal{M}_j$ to be the event that given a changepoint at time $j$, the MAP choice of changepoints and model occurs prior to time $j$. For $t = 1, \ldots, n$, $j = 0, \ldots, t-1$ and $q_k = 1, \ldots, \bar{p}$,

$$P_t(j, q_k) = p(C_t = j, \text{model } q_k, \mathcal{M}_j, \mathbf{y}_{1:t}), \text{ and}$$

$$P_t^{MAP} = p(\text{Changepoint at } t, \mathcal{M}_t, \mathbf{y}_{1:t}).$$

We obtain the following equations

$$P_t(j, q_k) = (1 - G(t - j - 1))P(j, t, q)p(q_k)P_j^{MAP}, \text{ and}$$

$$P_t^{MAP} = \max_{j,q_k}\{P_t(j, q_k)g(t - j)/(1 - G(t - j - 1)\}. \tag{4.7}$$

At time $t$, the MAP estimates of $C_t$ and the current model order are given respectively by the values of $j$ and $q_k$ which maximise $P_t(j, q)$. Given a MAP estimate of $C_t$, $\hat{c}_t$ we can theN calculate the MAP estimates of the changepoint prior to $\hat{c}_t$ and the model order of that segment by the values of $j$ and $q_k$ that maximise the right-hand side of (4.7). This procedure can be repeated to find the MAP estimates of all changepoint positions and model orders.

### 4.3.2 Approximate Inference

The computational and memory costs of the recursions for exact inference presented in Section 4.3.1 both increase with time. The computational cost of both the filtering recursion and MAP recursion at time $t$ is proportional to $t$, the number of possible values of $C_t$. While the memory cost of storing all filtering densities up to time $t$, necessary to simulate from the joint posterior of all changepoints prior to $t$, increases quadratically with $t$. For large data sets, these computational and memory costs may become prohibitive.

A similar problem of increasing computational cost occurs in the analysis of some hidden Markov models – though generally computational cost increases exponentially with time (Chen and Liu, 2000). Particle filters have been successfully applied to these problems (Fearnhead and Clifford, 2003) by using a resampling

step to limit the computational cost at each time-step. Here we show how similar resampling ideas can be applied to the on-line inference of the changepoint models we are considering. We present a variation on the optimal resampling method of Fearnhead and Clifford (2003) which is specifically designed for changepoint models, and show theoretically why this is an optimal resampling algorithm in this case. We also present an extension of the rejection control approach of Liu et al. (1998) which is suitable for the analysis of batch data, and for which it is possible to control the amount of error introduced at each resampling step.

### Controlling Computational Cost

Out first approach is to control the average (and maximum) computational cost for analysing each new observation. At time $t$ our exact algorithm stores the complete posterior distribution of the time of the last changepoint $p(C_t = c_t|\mathbf{y}_{1:t})$, for $c_t = 0, 1, \ldots, t-1$. We can approximate this by a discrete distribution with fewer, $N$, support points. This approximate distribution can be described by the set of support points, $c^{(1)}, \ldots, c^{(N)}$, henceforth called *particles*, and the probability mass associated with each of these particles, $w^{(1)}, \ldots, w^{(N)}$, which we call weights. (The particles and their weights will depend on $t$; we have suppressed this dependence to simplify notation.)

We impose a maximum number of particles to be stored at any one time, $N$, such that whenever we have $N$ particles we immediately perform resampling to reduce the number of particles to $M < N$. The average computational cost per iteration will thus be proportional to $(M+N+1)/2$, and the maximum computational cost per iteration proportional to $N$.

Assume that at the current time point we have $N$ particles; and wish to reduce these to $M$ particles. We propose the following stratified version of the optimal resampling algorithm of Fearnhead and Clifford (2003), which we call Stratified

Optimal Resampling (SOR).

**Initialisation** Assume we currently have a set of ordered particles $c^{(1)} < c^{(2)} < \cdots < c^{(N)}$, with associated weights $w^{(1)}, \ldots, w^{(N)}$, which sum to unity.

**(SOR1)** Calculate $\alpha$ the unique solution to $\sum_{i=1}^{N} \min\{1, w^{(i)}/\alpha\} = M$;

**(SOR2)** For $i = 1, \ldots, N$ if $w^{(i)} \geq \alpha$ then keep particle $c^{(i)}$ with weight $w^{(i)}$. Assume that $A$ particles are kept.

**(SOR3)** Use the stratified resampling algorithm of Carpenter et al. (1999) to resample $M - A$ times from the ordered set of the remaining $N - A$ particles (without shuffling). Each resampled particle is assigned a weight $\alpha$.

The stratified resampling algorithm used in step SOR3 is given in Appendix B. An example of this algorithm is given in Figure 4.1. The use of stratified resampling in step SOR3 means that at most one copy of each particle is kept, as the expected number of times a particle with weight $w$ is resampled is $w/\alpha < 1$ (note there is no advantage in having multiple copies of particles, see Fearnhead and Clifford, 2003). The only difference between this SOR algorithm and the original algorithm of Fearnhead and Clifford (2003) is that particles are ordered before resampling in step SOR3.

As shown in Fearnhead and Clifford (2003), if we denote by $W^{(i)}$ the (random) weight of a particle after resampling (so $W^{(i)} = w^{(i)}$, $\alpha$, or 0 depending on whether the respective particle is kept, resampled or not resampled), then SOR is optimal over all resampling algorithms that satisfy $\mathrm{E}(W^{(i)}) = w^{(i)}$ in terms of minimising the mean square error: $\mathrm{E}(\sum_{i=1}^{N}(W^{(i)} - w^{(i)})^2)$. By ordering the particles in step SOR3 we obtain the further property:

**Theorem 4.3.1** *Consider a set of $N$ particles, $c^{(1)} < c^{(2)} < \ldots < c^{(N)}$ with weights $w^{(1)}, \ldots, w^{(N)}$. Let $W^{(i)}$ be the (random) weight of particle $c^{(i)}$ after re-*

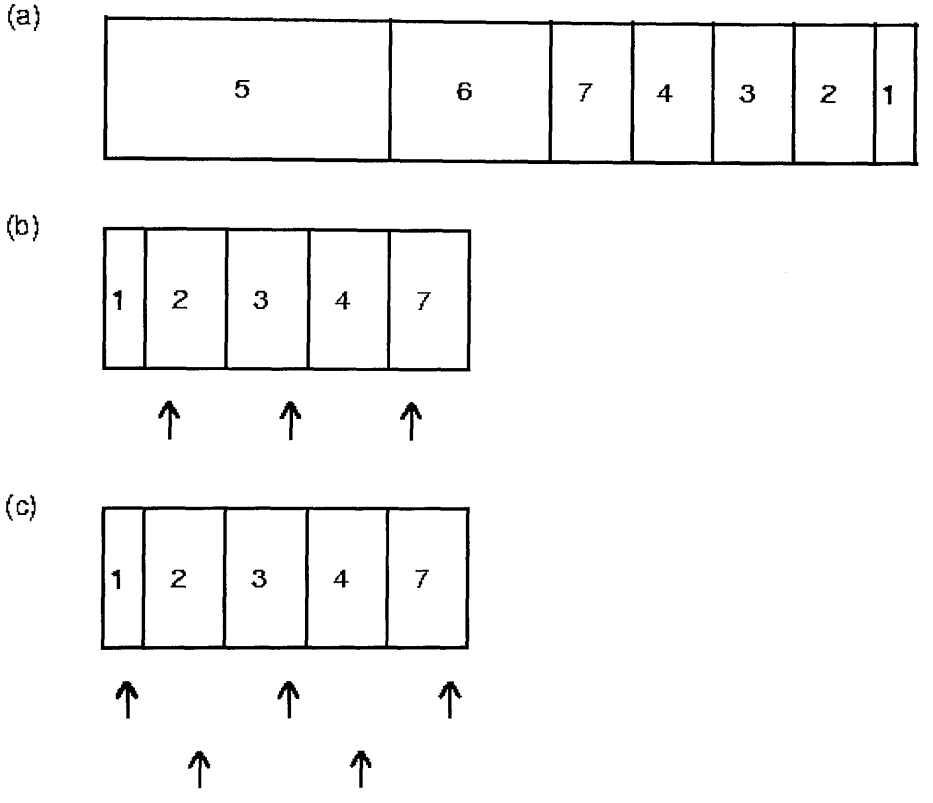


Figure 4.1: Example of the stratified resampling algorithm as used in SOR or SRC. (a) Example set of particles. Each box represents a particle, labelled with its value (the time of the most recent changepoint), and whose width is proportional to its weight. Particle 5 has weight 0.3; particle 6 has weight 0.25; particles 2–4 and 7 each have weight 0.1; and particle 1 has weight 0.05. (b) Stratified resampling within SOR to resample 5 particles. In this case $\alpha = 0.15$ and particles 5 and 6 are kept without resampling. The remaining particles are ordered as shown, with 3 to be resampled. A uniform random variable, $U$, on $[0, \alpha]$ is simulated, and 3 arrows are produced at positions $U$, $U + \alpha$, and $U + 2\alpha$. The particles which are pointed to by the arrows are resampled and are each assigned a weight $\alpha$. (c) Stratified resampling within SOR with $\alpha = 0.2$. Again particles 5 and 6 are kept without resampling, and the remaining particles are ordered. We again simulate $U$, a uniform random variable on $[0, \alpha]$, and place arrows at $U$, $U+\alpha$, $U+2\alpha$ an so on. In this case the number of arrows needed, and hence the number of particles resampled, will depend on $U$. We show two possible set of arrows for this example, the top set produces 3 resampled particles, and the bottom set 2. Each resampled particle is assigned a weight $\alpha$.

*sampling. Define the maximum Kolmogorov Smirnov Distance for a resampling algorithm as*

$$mKSD = \max\left\{\max_i \left|\sum_{j=1}^{i}\left(w^{(i)} - W^{(i)}\right)\right|\right\} \tag{4.8}$$

*where the first maximisation is over realisations of* $W^{(1)}, \ldots, W^{(N)}$ *with positive probability. Then the SOR algorithm above satisfies* $mKSD \leq \alpha$ *(where* $\alpha$ *is defined as in SOR1). Furthermore (i) for a resampling algorithm to have* $mKSD \leq \alpha$ *then all particles with* $w^{(i)} > \alpha$ *must be propagated without resampling;and (ii) the mKSD for the SOR algorithm above is less than or equal to the mKSD of the optimal resampling algorithm of Fearnhead and Clifford (2003), and the rejection control algorithm of Liu et al. (1998).*

**Proof:** See Appendix C. □

Kolmogorov Smirnov distance is a natural metric for the distributions of 1-dimensional random variables. By using (4.8) as a measure of error of a resampling algorithm, we are considering the bound on Kolmogorov Smirnov distance that a resampling algorithm can introduce. The theorem gives a simple interpretation of the $\alpha$ calculated in step SOR1; in terms of an upper bound on the Kolmogorov Smirnov distance between the original and resampled weights.

We define a resampling algorithm to be unbiased if $\mathrm{E}(W^{(i)}) = w^{(i)}$ for all $i$. (This is related to the properly weighted condition of Liu et al., 2001). The optimal resampling algorithm of Fearnhead and Clifford (2003) and rejection control are currently the only other unbiased resampling algorithm which satisfy the condition (i) of Theorem 4.3.1. (Note that rejection control will not produce a fixed number of particles after resampling; though implementing rejection control with a threshold of $\alpha$ will produce on average $N$ resampling particles, and further that while $\mathrm{E}(W^{(i)}) = w^{(i)}$, the resampled weights do not necessarily sum to 1.) So results (i) and (ii) of Theorem 4.3.1 show that our SOR algorithm is optimal over all existing unbiased resampling algorithms in terms of minimising mKSD.

A more natural comparison of resampling algorithms would be in terms of expected Kolmogorov Smirnov distance. The difference between the SOR algorithm and the other two algorithms in (i) is the stratification with the resampling step – this is specifically introduced to minimise the Kolmogorov Smirnov distance between the weighted particles before and after resampling. Intuition suggests, and we conjecture, that SOR performs better in terms of minimising expected Kolmogorov Smirnov distance than either of the two algorithms in (i).

We have presented the SOR algorithm in terms of the general case of resampling $M$ particles from $N$ current particles. For on-line inference, where there is a fixed amount of time to analyse each observation, it is natural to choose $N$ to be the largest number of particles that enable an observation to be analysed in less than this amount of time; and to set $M = N - 1$. In this case there is no difference between SOR and the existing optimal resampling algorithm. In practice, it may be better to choose $N - M > 1$ (see Section 4.4) as this enables the particles to be removed to be jointly chosen in a stratified manner.

### Controlling Resampling Error

An alternative to basing resampling on the average and maximum number of particles to be kept at each time step, is to choose the amount of resampling to control the size of error that is introduced at each time step. Such an approach is most suitable for using on-line algorithms to analyse batch data. For real-time data, the frequency of observations will place an upper bound on the CPU time, and hence the number of particles, that can be used to process each observation. By controlling the resampling error, rather than the number of particles, we cannot ensure that the number of particles always stays below this error.

The idea behind controlling the resampling error is given by the interpretation of $\alpha$ for SOR that comes from Theorem 4.3.1. The value of $\alpha$ defines the maximum error

(as defined by Kolmogorov Smirnov distance) that is introduced by the resampling error. So rather than specifying the number of resampled particles which in turn defines $\alpha$, and hence the amount of error we introduce, we can instead specify $\alpha$ which will then define the number of resampled particles.

Our method for controlling the resampling error is to use a stratified version of rejection control (Liu et al., 1998), rather than adapt the SOR algorithm. For a pre-specified value of $\alpha$, our stratified rejection control (SRC) algorithm is:

**Initialisation** Assume we currently have a set of ordered particles $c^{(1)} < c^{(2)} < \cdots < c^{(N)}$, with associated weights $w^{(1)}, \ldots, w^{(N)}$, which sum to unity.

**(SRC1)** For $i = 1, \ldots, N$ if $w^{(i)} \geq \alpha$ then keep particle $c^{(i)}$ with weight $w^{(i)}$. Assume that $A$ particles are kept.

**(SRC2)** Use the stratified resampling algorithm of Carpenter et al. (1999) to resample from the ordered set of the remaining $N - A$ particles (without shuffling). The expected number of times particle $c^{(i)}$ is resampled is $w^{(i)}/\alpha$. Each resampled particle is assigned a weight $\alpha$.

Again the use of stratified resampling in (SRC2) means that at most one copy of each particle is kept. Note that the sum of the particles' weights after resampling will not necessarily sum to 1 (though they lie between $1-\alpha$ and $1+\alpha$), and should be normalised to produce a probability distribution.

The difference between SRC and rejection control (Liu et al., 1998) is that particles are ordered and stratified resampling is used in step SRC2, as opposed to independent resampling of each particles. (See Figure 4.1 for an example of this step.) The use of stratified resampling means that the maximum error of the unnormalised weights introduced by SRC, as measured by Kolmogorov Smirnov distance, is $\alpha$ (this can be proved in an identical manner to Theorem 4.3.1). Furthermore, the error of the normalised weights can be shown to be bounded above

by $\alpha/(1-\alpha) = \alpha + o(\alpha)$ (see Appendix D).

Note that our implementation of SRC is different to that of standard rejection control. We resample if and only if the smallest weight of a current particle is less than $\alpha$, whereas Liu et al. (1998) recommend monitoring the variance of the weights, and to resample when this variance goes above some threshold. Furthermore, our Theorem 4.3.1 gives a natural interpretation of $\alpha$, the parameter that governs the amount of resampling we do.

## 4.4 Numerical Examples

We tested our algorithm on three simulated examples: the Blocks and Heavisine examples from Donoho and Johnstone (1994) and a piecewise auto-regressive model. Each of the three data-sets are analysed under a piecewise regression model. For the Blocks and Heavisine examples we model the data as realisations from a piecewise polynomial regression with design matrix for a segment consisting of observations at times $s, s+1, \ldots, t$ as

$$\mathbf{H}_k = \begin{pmatrix} 1 & x_s & x_s^2 \\ 1 & x_{s+1} & x_{s+1}^2 \\ \vdots & \vdots & \vdots \\ 1 & x_t & x_t^2 \end{pmatrix}.$$

where $x_s = s/n$ and $n$ is the number of data points. For the piecewise auto-regressive model we have a design matrices of the form

$$\mathbf{H}_k = \begin{pmatrix} y_{s-1} & y_{s-2} & y_{s-3} \\ y_s & y_{s-1} & y_{s-2} \\ \vdots & \vdots & \vdots \\ y_t & y_{t-1} & y_{t-2} \end{pmatrix},$$

In each case we perform model choice within each segment, choosing between model orders 1, 2 and 3. We allow for different variances of the measurement error within each segment, although for the Blocks and Heavisine examples we simulated data with a common error variance across segments. Further details of the model, and calculations required for calculating the $P(s, t, q_k)$s is given in Section 4.2 (See also Punskaya et al., 2002; Fearnhead, 2005).

Our focus here is on the performance of different possible resampling algorithms. The Blocks data set (see Figure 4.2) is a particularly simple data set to analyse, and all reasonable resampling algorithms will give almost identical results. We show the results here to demonstrate how the SRC algorithm naturally adapts the number of particles that are kept. The Blocks data set has a number of obvious changepoints, and when each of these are encountered the number of particles that are needed to be kept is reduced to close to 1.

For the Heavisine example (see Figure 4.2) we compared the accuracy of various resampling algorithms: stratified rejection control (SRC), rejection control (RC), stratified optimal resampling (SOR), and optimal resampling (OR). We considered two values of $\alpha$ for SRC and RC; and for a meaningful comparison, fixed the mean number of particles in OR and SOR to the mean number of particles kept by SRC for each of these two values. If we set the number of resampled particles ($M$) to be one less than the number of particles prior to resampling ($N$), then OR and SOR are identical. We tested both $N = M + 1$ and $N = M + 5$.

Our comparison is based on the Kolmogorov Smirnov distance (KSD) between the true filtering distribution of the most recent changepoint, $p(C_t|y_{1:t})$ (calculated using the on-line algorithm with no resampling), and its approximation based on the various resampling algorithms, for each $t$. Results are given in Table 4.1. The results show that the mean KSD error is reduced by one third by using SRC rather than RC. Both of these methods perform better than the resampling algorithms that use a fixed number of particles (for the same average number of particles);

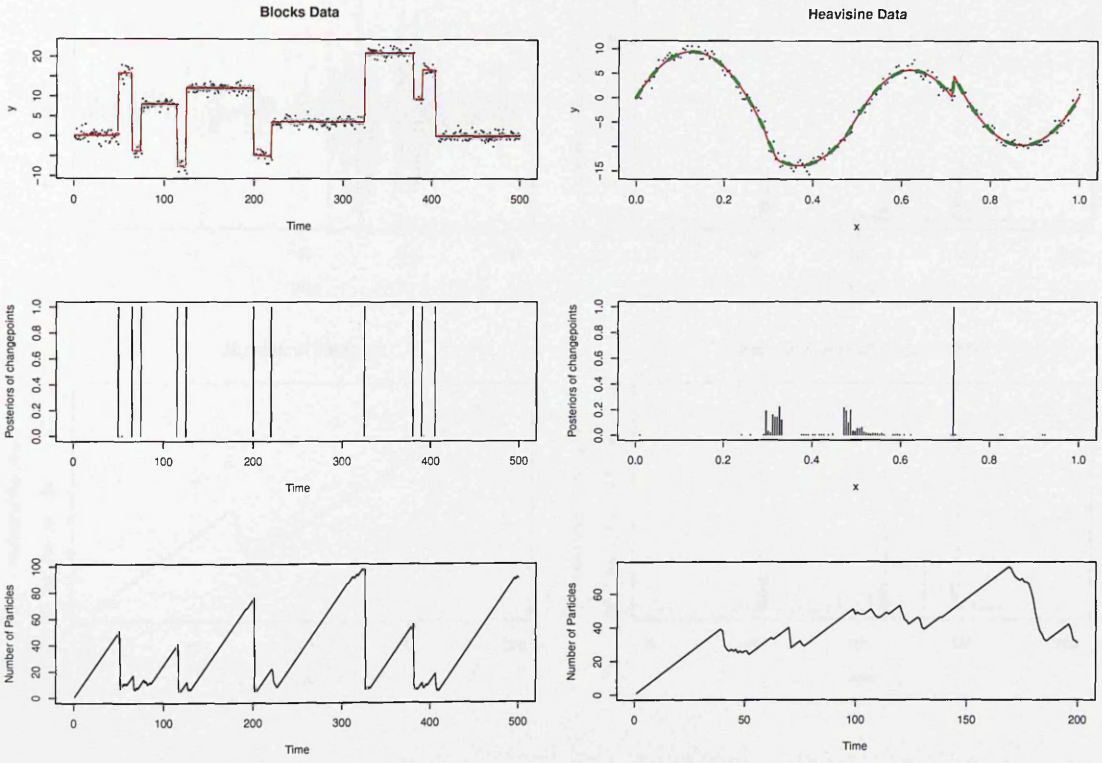


Figure 4.2: The Blocks data set (left-hand column) and Heavisine data set (right-hand column) together with results of analysis by the SRC algorithm with $\alpha = 10^{-6}$: data and inferred signal (top); marginal probability of changepoints (middle); and numbers of particles kept (bottom).

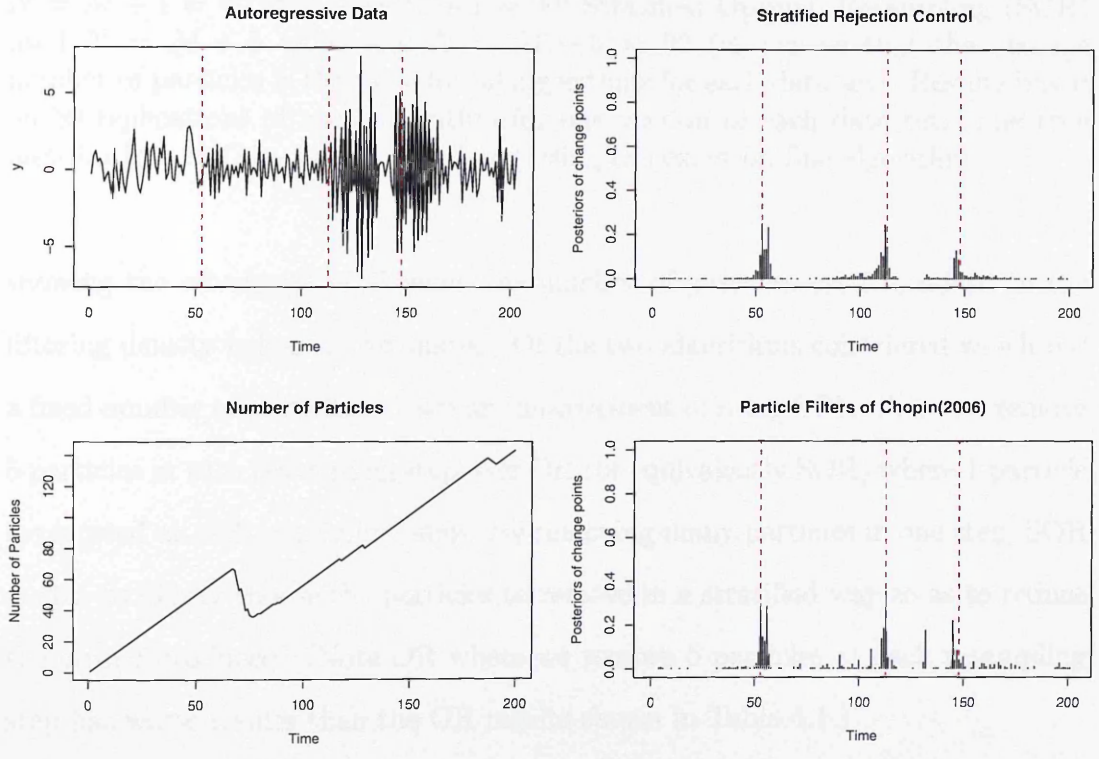


Figure 4.3: Results of analysing the AR data set using SRC with $\alpha = 10^{-6}$, and the particle filter of Chopin (2007) with 50,000 particles: data (top left), marginal probabilities of changepoint for SRC (top right) and particle filter of Chopin (2007) (bottom right), and number of particles kept using SRC (bottom left). The true AR model to the four segments have model orders 1, 1, 2, and 3 respectively. The corresponding parameters are $\beta_k = 0.4$; $\beta_k = -0.6$; $\beta_k = (-1.3, -0.36, 0.25)$ and $\beta_k = (-1.1, -0.24)$ with error variances $1.2^2$, $0.7^2$, $1.3^2$ and $0.9^2$ respectively.

| | SRC | RC | SOR | OR |
|---|---|---|---|---|
| Heavisine | $1.3\times10^{-2}$ | $2.0\times10^{-2}$ | $4.2\times10^{-2}$ | $6.4\times10^{-2}$ |
| AR | $1.3\times10^{-6}$ | $2.2\times10^{-6}$ | $2.2\times10^{-4}$ | $3.5\times10^{-4}$ |

Table 4.1: Mean Kolmogorov Smirnov Distance in $P(C_t|y_{1:t})$ averaged over $t$ for the Heavisine and AR models and the four resampling algorithms. Stratified Rejection Control (SRC) and Rejection Control (RC) were implemented with $\alpha = 10^{-6}$; these algorithms used an average number of 43 and 70 particles for the Heavisine and AR models respectively. Optimal Resampling (OR) was implemented with $N = M + 1 = 49$ and $N = M + 1 = 90$; Stratified Optimal Resampling (SOR) used $N = M + 5 = 51$ and $N = M + 5 = 92$ (chosen so that the average number of particles is the same for all algorithms for each data set). Results based on 50 replications of each algorithm for one version of each data set. The true distribution, $P(C_t|y_{1:t})$, was calculated using the exact on-line algorithm.

showing the advantage of allowing the number of particles used to adapt to the filtering density being approximated. Of the two algorithms considered which use a fixed number of particles, we see an improvement of using SOR where we remove 5 particles at each resampling step over OR (or equivalently SOR) where 1 particle is removed at each resampling step. By removing many particles in one step, SOR is able to jointly choose the particles to remove in a stratified way so as to reduce the error introduced. (Note OR where we remove 5 particles at each resampling step has worse results than the OR results shown in Table 4.1.)

Note that while all resampling algorithms introduce small errors at each resampling step, it is possible for these errors to accumulate. The reason for this, appears to be that the evidence for a changepoint at a given time $t$ can change substantially as more data is collected. If the evidence is small (and hence the filtering probability of a changepoint at $t$ is less than $\alpha$) at a resampling step, this can lead to the corresponding particle being removed. Such a particle can not be "resurrected" as future observations are made, even if they carry strong evidence for a changepoint at $t$. However stratified resampling should ensure that a particle corresponding to a changepoint close to $t$ is kept, and thus the error in estimating the position of the changepoints will still be small.

We repeated this analysis for a piecewise AR model. The results of the SRC analysis with $\alpha = 1 \times 10^{-6}$ given in Figure 4.3, and results of the accuracy of each resampling method given in Table 4.1. We observe similar results to the Heavisine example in terms of the relative performance of the resampling algorithms. In this case SRC again outperforms RC by about a third. The difference in performance between SRC and RC as compared to SOR and OR is quite substantial in this case, because towards the end of the time series it is forced to use too few particles to adequately approximate the filtering densities. This again demonstrates the potential gains to be obtained by allowing the number of particles used to change over time and to adapt to the filtering distribution that is being approximated.

We also ran the particle filter of Chopin (2007). This filter does not integrate out the parameters associated with each segment, so each particle consists of a time for the last changepoint together with a value of the parameters for the current segment. The filter uses MCMC to update the parameters of a subset of particles at each iteration. We ran the filter with 50,000 particles, using a Gibbs sampler update on the parameters of 1/3 of the particles at each iteration. This took over an order of magnitude longer to run than the SRC algorithm, and even is substantially more time-consuming to implement than the exact on-line algorithm.

The results for the estimate of the marginal probabilities of the changepoints is shown in Figure 4.3. The filter of Chopin (2007) suffers from a loss of diversity in the particles – with many positions being assigned zero probability of being a changepoint, when in fact there is a non-negligible probability as can be seen from the output of the SRC filter. To give a quantitative comparison of the two methods we calculated the mean absolute error between the estimates of the marginal probabilities of the changepoints shown in Figure 4.3 with those based on the exact particle filter algorithm. These were 0.010 and 0.002 for the filter of Chopin (2007) and the SRC filter respectively.

## 4.5 DNA Segmentation

In recent years there has been an explosion in the amount of data describing the genetic make-up of different organisms; for example the complete DNA sequence of one human genome is now known as a result of the Human Genome project. There is interest in learning about the genomic features of different organisms, and learning how these features may have evolved and how they correlate with each other and with biological processes.

We consider the problem of understanding the structure of C+G content within the genome. A common model for the C+G content of the human genome is that there are large, of the order of 300 kilobases (kb), regions of roughly homogeneous C+G content, called Isochores (see Bernardi, 2000, for background). Furthermore C+G content is known to correlate with various features of the genome, such as high recombination rates and gene density (Hardison et al., 2003).

Currently, the most common method for segmenting an organism's genome into regions of different C+G content is implemented in the computer program `IsoFinder` (Oliver et al., 2004). This is based on a recursive segmentation procedure, which initially classifies a large genomic region as consisting of a single Isochore (region of common C+G content). It then considers in turn each possible position for adding a changepoint, and splitting the data into two Isochores. For each possible position, a $t$-statistic is calculated for testing whether the mean C+G content is different in the two putative Isochores. For each changepoint, a $p$-value is calculated for its value of the $t$-statistic using a bootstrap procedure, and if the smallest $p$-value is less than some predefined threshold, then the corresponding changepoint is added. This procedure is repeated, with at each step each current Isochore being tested for whether it can be split into two Isochores. See Oliver et al. (2004) for more details, and Oliver et al. (2002) for examples of the use of `IsoFinder`.

We consider a Bayesian approach to segmenting a genomic region into Isochores. The potential advantages of a Bayesian approach include (i) quantifying and averaging over the uncertainty in the number and positions of the Isochores; (ii) jointly estimating all Isochore positions (which Braun et al., 2000, show to be more accurate than segmentation procedures); and (iii) the large amount of data available for each organism makes it straightforward to construct sensible prior distributions. For related examples of the use of Bayesian methods for analysing other aspects of Genomic structure see Salmenkivi et al. (2002); Boys and Henderson (2004).

One of the computational challenges of such an analysis is the large amount of data that needs to be analysed (for example human chromosomes consist of around 100 million bases). We simplify this burden by first summarising our data by the number of DNA sites which are C or G within consecutive windows (each window being of the order of a few kb in width), an approach which also has the advantage of averaging our the very local high variation in C+G content caused for example by CpG islands and Alu elements. We then hope that our on-line changepoint algorithm will be able to efficiently analyse the resulting data, and one of the main aims of the study we present here is to test whether such an approach is computationally practicable for analysing the large amount of genomic data currently available.

The model we use is based on the following simple model for the data $y_{1:n}$, which is similar to the implicit model assumed by `IsoFinder`. Two data sets are shown in Figure 4.4 and 4.5. The $t$th data point, $y_t$, represents the number of DNA bases which are either C or G within the $t$th window. If this window lies within the $i$th Isochore then we assume

$$y_t = \mu_i + \sigma_i \varepsilon_t,$$

where $\mu_i$ is the mean C+G content of each window within the $i$th Isochore, $\sigma_i^2$ is the error variance within the $i$th Isochore, and $\varepsilon_t$ is some independent error. We assume that $\varepsilon_t$ has a Gaussian distribution and we assume standard conjugate

priors (see Section 4.2) for the $\mu_i$s and $\sigma_i$s, with the prior parameters chosen from an initial analysis of C+G data with a moving median filter. For each model we assumed a geometric distribution for the length of each Isochore.

Results of our analysis using SRC with $\alpha = 10^{-6}$ are shown in Figure 4.4 and 4.5. Our main focus is on the computational practicability of a Bayesian analysis of such data, and our method took 1 and 6 seconds respectively on a desktop PC (with AMD Athlon XP processor) to analyse these data sets.

This application does not need to be analysed by an on-line algorithm, such as the one we used. However Fearnhead (2006) showed that the version of our algorithm without resampling can be more efficient for analysing changepoint models than some commonly used MCMC algorithms. Furthermore, by using resampling we have been able to vastly reduce the computational and storage cost of analysing the data. For example our implementation with resampling uses an average of 117 particles at each time step on the data from human chromosome 1; whereas without resampling the algorithm would require an average of over 3,500 particles for each time-step.

Whilst it is difficult to construct a quantitative evaluation of how good the resulting segmentation is, we are encouraged that our method finds the classical H3 and L2 isochores in the MHC regions (at positions 1.9Mb to 2.5Mb and 2.5Mb to 3.0Mb respectively; see Oliver et al., 2001).

## 4.6 Discussions

We have considered a class of changepoint models, which have a specific conditional independence structure (see Section 4.2), and shown how the direct simulation algorithm of Fearnhead (2005) can be implemented on-line. Such an algorithm can be viewed as an exact particle filter, and resampling ideas taken from particle

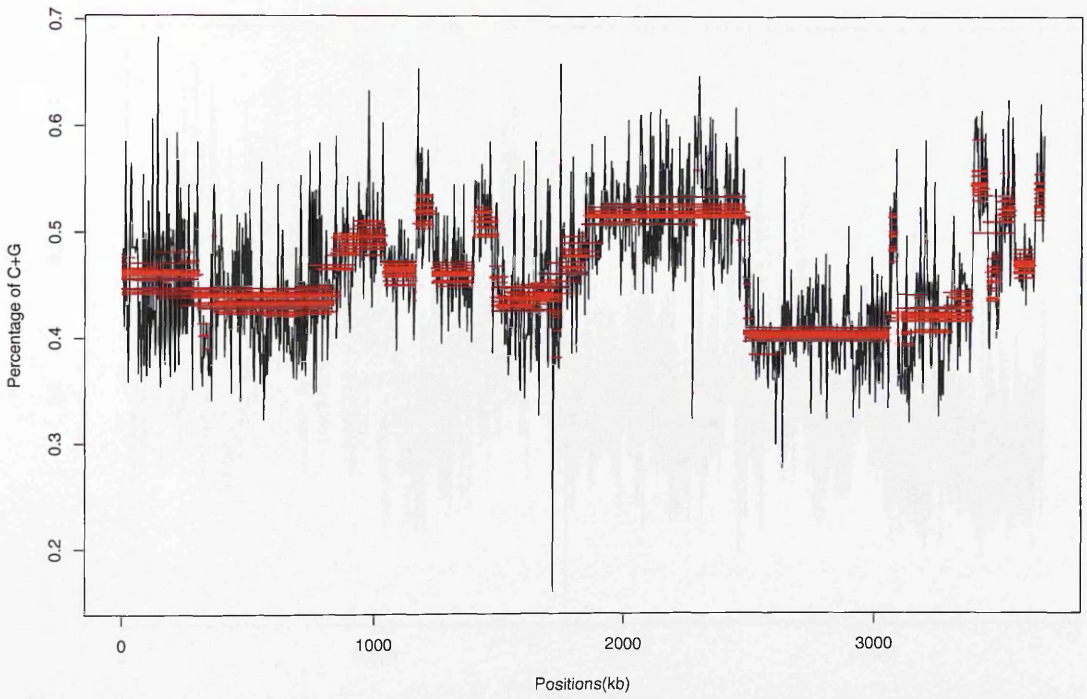


Figure 4.4: Analysis of 3.6 Mb of data from the MHC region. The data consist of number of C+G nucleotides in 3kb windows. We show 20 realisations from the joint posterior distribution of the segmentation.

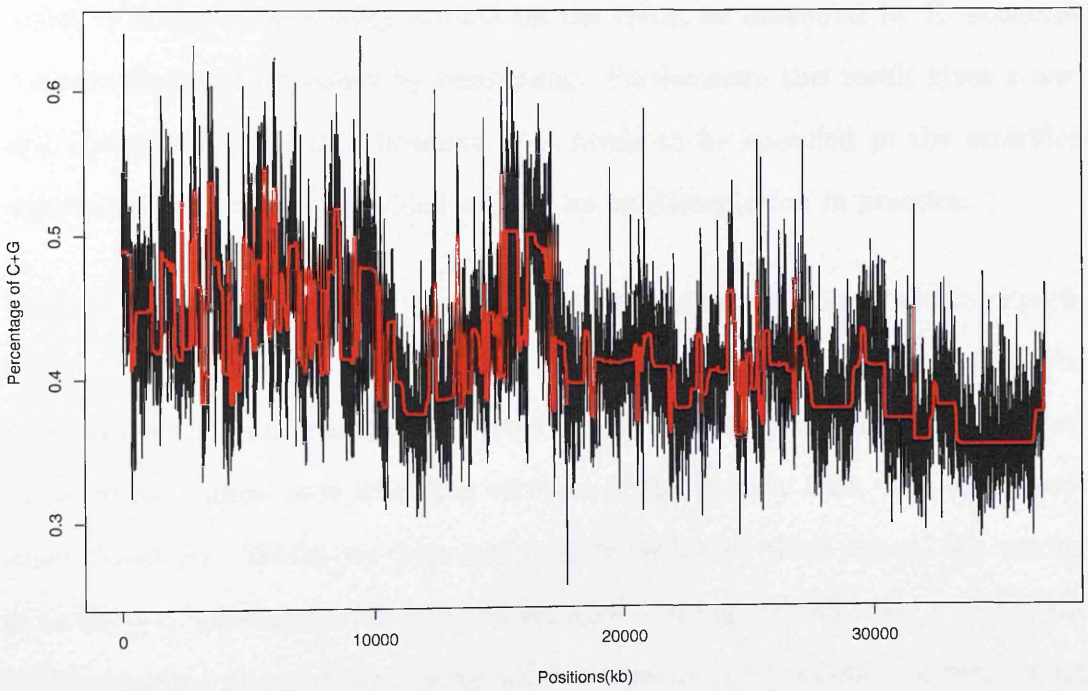


Figure 4.5: Analysis of 35Mb of data from human chromosome 1. The red line is the posterior mean GC content.

filters can be used to reduce the computational complexity of the direct simulation algorithm (at the cost of introducing error). We have presented two simple extensions of existing resampling algorithms, which are particularly well suited to changepoint problems (or any problems where the underlying state of interest is 1-dimensional).

In simulation studies, our new resampling algorithms decreased the error of the resulting particle filter by up to one third, compared to particle filters using the existing resampling approaches. We have shown that the new resampling algorithms satisfy a minimax optimality criteria on the error, as measured by Kolmogorov Smirnov distance, introduce by resampling. Furthermore this result gives a natural interpretation of the threshold that needs to be specified in the stratified rejection control algorithm which will aid its implementation in practice.

There is great flexibility with implementing resampling algorithms within particle filters which we have not explored. For example Liu and Chen (1998) discuss the frequency with which resampling should occur, and Liu et al. (1998) suggest using rejection control only when the variance of the particle filter weights exceeds some threshold. Whilst we have not fully investigated these issues, the results from Section 4.4 suggests that the advantages of using stratification within optimal resampling or rejection control will increase as the frequency of resampling decreases (or equivalently the amount of particles resampled increases at each resampling step). Note that when we allow for different models within each segment, we could resample over the joint space of changepoint positions and model orders. First resampling based on changepoint position, and then on model order for each changepoint position, would enable this to be done whilst still keeping the stratification of particles by changepoint position.

We considered using our on-line algorithm to analyse the structure of C+G content in the human genome. Our main aim was to show that a Bayesian analysis is computationally feasible even for the large data-sets currently available. In prac-

tice we were able to analyse around 35Mb of human data in around 6 seconds of computing time; and this together with the linear computational cost of our algorithm shows that such a Bayesian analysis will scale to analysing data from complete genomes.

The model we implemented was based on the simple implicit model behind existing segmentation procedures. Visual inspection of the data by eye suggests such models are over-simplistic, though we obtained reasonable segmentations. Modelling the variation in C+G content of the human genome is a challenging problem, but we note the flexibility in models allowed within the conditions we require to implement our on-line approach. Various extensions that are possible are allowing the C+G content to vary linearly within each segment; allowing GC content to depend on other features of the Isochore; modelling the autocorrelation in the data (see Fearnhead, 2006); general models for segment lengths; different measurement errors (for example Laplacian errors as used in Ó Ruanaidh and Fitzgerald, 1996, for a related problem); and allowing for a mixture distribution for the segment means (as suggested by Bernardi, 2000).

## 4.7 Appendix

### A: Calculation of likelihood $P(s,t,q_k)$ for $q_k$th order linear model

Given the linear regression model of (4.3) with the conjugate priors as follows:

$$\beta_k \sim NVN(\vec{0}, \sigma_k^2 \mathbf{D}_k), \qquad \sigma_k^2 \sim IG(\nu/2, \gamma/2),$$

where $\vec{0}$ denotes a vector of 0s and $\mathbf{D}_k = diag(\delta_1^2, \ldots, \delta_{q_k}^2)$, we have

$$
\begin{aligned}
& P(s,t,q_k) \\
= & \int_{\sigma_k^2}\int_{\beta_k} p(\mathbf{y}_{s+1:t}, \beta_k, \sigma_k^2, \text{model } q_k)\pi(\beta_k)\pi(\sigma_k^2)\mathrm{d}\beta_k\mathrm{d}\sigma_k^2 \\
= & \int_{\sigma_k^2}\int_{\beta_k} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(\mathbf{y}_{s+1:t}-\mathbf{H}_k\beta_k)^T(\mathbf{y}_{s+1:t}-\mathbf{H}_k\beta_k)^T}{2\sigma_k^2}\right\} \times \\
& \frac{1}{(\sqrt{2\pi}\sigma)^{q_k}|\mathbf{D}_k|^{1/2}} \exp\left\{-\frac{1}{2\sigma_k^2}\beta_k^T\mathbf{D}_k^{-1}\beta_k\right\} \times \\
& \frac{(\gamma/2)^{\nu/2}}{\Gamma(\nu/2)}(\sigma^{-2})^{\frac{\nu}{2}-1}e^{-\frac{\gamma}{2}\sigma^{-2}}\mathrm{d}\beta_k\mathrm{d}\sigma_k^2.
\end{aligned}
$$

If we denote

$$
\begin{aligned}
\mathbf{M}_k &= (\mathbf{H}_k^T\mathbf{H}_k + \mathbf{D}_k^{-1})^{-1}, \\
\mathbf{P}_k &= (\mathbf{I}_k - \mathbf{H}_k\mathbf{M}_k\mathbf{H}_k^T), \\
||\mathbf{y}||_{\mathbf{P}_k}^2 &= \mathbf{y}^T\mathbf{P}_k\mathbf{y},
\end{aligned}
$$

then the integration of $\beta_k$ is then

$$
\begin{aligned}
& \int_{\beta_k} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{(\mathbf{y}_{s+1:t}-\mathbf{H}_k\beta_k)^T(\mathbf{y}_{s+1:t}-\mathbf{H}_k\beta_k)^T}{2\sigma_k^2}\right\} \times \\
& \frac{1}{(\sqrt{2\pi}\sigma)^{q_k}|\mathbf{D}_k|^{1/2}} \exp\left\{-\frac{1}{2\sigma_k^2}\beta_k^T\mathbf{D}_k^{-1}\beta_k\right\} \mathrm{d}\beta_k \\
= & \int_{\beta_k} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{t-s+q_k} \times \\
& \exp\left\{-\frac{1}{2\sigma_k^2}\left((\mathbf{y}_{s+1:t}-\mathbf{H}_k\beta_k)^T(\mathbf{y}_{s+1:t}-\mathbf{H}_k\beta_k)+\beta_k^T\mathbf{D}_k^{-1}\beta_k\right)\right\} \mathrm{d}\beta_k \\
= & \int_{\beta_k} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{t-s+q_k} \times \\
& \exp\left\{-\frac{1}{2\sigma_k^2}(\beta_k^T(\mathbf{H}_k^T\mathbf{H}_k+\mathbf{D}_k^{-1})\beta_k-\mathbf{y}_{s+1:t}^T\mathbf{H}_k\beta_k-\beta_k^T\mathbf{H}_k^T\mathbf{y}_{s+1:t}+\mathbf{y}_{s+1:t}^T\mathbf{y}_{s+1:t}\right\} \mathrm{d}\beta_k \\
= & \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{t-s} \exp\left\{-\frac{1}{2\sigma_k^2}(\mathbf{y}_{s+1:t}^T\mathbf{y}_{s+1:t}-\mathbf{y}_{s+1:t}^T\mathbf{H}_k\mathbf{M}_k\mathbf{H}_k^T\mathbf{y}_{s+1:t})\right\} \times \\
& \int_{\beta_k} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{q_k} \exp\left\{-\frac{1}{2\sigma_k^2}(\beta_k^T-(\mathbf{M}_k\mathbf{H}_k^T\mathbf{y}_{s+1:t})^T)\mathbf{M}_k^{-1}(\beta_k-\mathbf{M}_k\mathbf{H}_k^T\mathbf{y}_{s+1:t})\right\} \mathrm{d}\beta_k \\
= & \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{t-s} |\mathbf{M}_k|^{\frac{1}{2}} \exp\left\{-\frac{1}{2\sigma_k^2}(\mathbf{y}_{s+1:t}^T\mathbf{P}_k\mathbf{y}_{s+1:t})\right\}, \\
= & \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{t-s} |\mathbf{M}_k|^{\frac{1}{2}} \exp\left\{-\frac{||\mathbf{y}_{s+1:t}||^2_{\mathbf{P}_k}}{2\sigma_k^2}\right\},
\end{aligned}
$$

continue to integrate out $\sigma_k^2$, we obtain:

$$
\begin{aligned}
& \int_{\sigma_k^2} \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^{t-s} \exp\left\{-\frac{1}{2\sigma_k^2}||\mathbf{y}_{s+1:t}||^2_{\mathbf{P}_k}\right\} \frac{(\gamma/2)^{\nu/2}}{\Gamma(\nu/2)}(\sigma^{-2})^{\frac{\nu}{2}-1}e^{-\frac{\gamma}{2}\sigma^{-2}}\mathrm{d}\sigma_k^2 \\
= & \left(\frac{1}{\sqrt{2\pi}}\right)^{t-s} \frac{(\gamma/2)^{\nu/2}}{\Gamma(\nu/2)} \int_{\sigma_k^2} \exp\left\{-\frac{||\mathbf{y}_{s+1:t}||^2_{\mathbf{P}_k}+\gamma}{2}\sigma^{-2}\right\}(\sigma^{-2})^{\frac{t-s+\nu}{2}-1}\mathrm{d}\sigma_k^2 \\
= & \left(\frac{1}{\sqrt{2\pi}}\right)^{t-s} \frac{(\gamma/2)^{\nu/2}}{((||\mathbf{y}_{s+1:t}||^2_{\mathbf{P}_k}+\gamma)/2)^{(t-s+\nu)/2}} \frac{\Gamma((t-s+\nu)/2)}{\Gamma(\nu/2)}.
\end{aligned}
$$

Finally, we have

$$
\begin{aligned}
& P(s,t,q_k) \\
= & \; p(\mathbf{y}_{s+1:t}|s+1:t, M_t=0) \\
= & \; \pi^{-(t-s)/2}\left(\frac{|\mathbf{M}_k|}{|\mathbf{D}_k|}\right)^{\frac{1}{2}} \frac{(\gamma)^{\nu/2}}{(||\mathbf{y}_{s+1:t}||^2_{\mathbf{P}_k}+\gamma)^{(t-s+\nu)/2}} \frac{\Gamma((t-s+\nu)/2)}{\Gamma(\nu/2)},
\end{aligned}
$$

Further more, we obtain the posterior distribution of $\beta_k$ and $\sigma_k^2$ which are

$$\begin{aligned} \beta_k &\sim MVN(\mathbf{M}_k\mathbf{H}_k^T\mathbf{y}_{s+1:t}, \sigma_k^2\mathbf{M}_k), \\ \sigma_k^2 &\sim IG\left(\frac{t-s+\nu}{2}, \frac{||\mathbf{y}_{s+1:t}||^2_{\mathbf{P}_k}+\gamma}{2}\right). \end{aligned}$$

## B: Stratified Resampling Algorithm

We describe the stratified resampling algorithm of Carpenter et al. (1999) in terms of the SOR and SRC algorithms. Assume we currently have a set of $N$ ordered particles $c^{(1)} < c^{(2)} < \cdots < c^{(N)}$, with associated weights $w^{(1)}, \ldots, w^{(N)}$, which sum to unity. For the SOR algorithm define $\alpha$ as in step (SOR1); and for SRC we assume that the value of $\alpha$ is given. Resampling of $M$ particles proceeds as follows:

(A) Simulate $u$ a realisation of a uniform random variable on $[0, \alpha]$. Set $i = 1$.

(B) If $w^{(i)} \geq \alpha$ then propagate particle $c^{(i)}$ with weight $w^{(i)}$; else let $u = u - w^{(i)}$; if $u \leq 0$ then resample particle $c^{(i)}$ and assign a weight $\alpha$, and set $u = u + \alpha$.

(C) Let $i = i + 1$; if $i \leq N$ then return to (B).

## C: Proof of Theorem 4.3.1

Theorem 4.3.1 considers the error of a resampling algorithm as measured by:

$$\text{mKSD} = \max\left\{\max_i \left|\sum_{j=1}^{i} w^{(i)} - W^{(i)}\right|\right\}$$

For SOR, if $w^{(i)} \geq \alpha$ then $W^{(i)} = w^{(i)}$ with probability 1. As such we can consider the mKSD solely for the subset of particles which have $w^{(i)} < \alpha$. Assume we have $N'$ such particles, and relabel these particles $c^{(1)} < c^{(2)} < \ldots < c^{(N')}$.

The only randomness in the SOR algorithm is the simulation of $u$ in step (A) of the algorithm detailed in Appendix A. Now for a given value of $u$

$$\sum_{j=1}^{i} W^{(j)} = \alpha \left[ \left( \sum_{j=1}^{i} w^{(j)} + \alpha - u \right) / \alpha \right], \tag{4.9}$$

where $[x]$ is the integer part of $x$. Thus for all $u$ and $i$

$$\left| \sum_{j=1}^{i} w^{(j)} - W^{(j)} \right| \leq \alpha,$$

so mKSD $\leq \alpha$.

For result (i) it suffices to note that if the probability of resampling particle $c^{(i)}$ is strictly less than 1; then mKSD$\geq w^{(i)}$.

For result (ii) it is sufficient to note that both the optimal resampling algorithm of Fearnhead and Clifford (2003) (where particles are shuffled prior to stratified resampling) and rejection control (where each particle with weight less then $\alpha$ is resampled independently of all others) give positive probability to all realisation of weights $W^{(1)}, W^{(2)}, \ldots, W^{(N)}$ that our SOR algorithm does. It trivially follows that the mKSD for these algorithms will be greater than that of our SOR algorithm.

## D: Error bound for SRC

Consider $N$ particles, ordered so that $c^{(1)} < c^{(2)} < \ldots < c^{(N)}$. We denote the weight of these particles prior to resampling by $w^{(i)}$, the unnormalised weights after resampling by $W^{(i)}$, and the normalised weights after resampling by $\bar{W}^{(i)}$. We let $u$ denote the realisation of the Uniform $[0, \alpha]$ random variable used in the stratified resampling algorithm. Finally we let

$$\varepsilon^{(i)} = \sum_{j=1}^{i} \left( w^{(j)} - W^{(j)} \right).$$

The sum of the resampling weights depends on the number of particles resampled in stage SRC2. There exists a constant, $\beta$, satisfying $0 \leq \beta < \alpha$ such that

$$\sum_{i=1}^{N} W^{(i)} = \begin{cases} 1+\alpha-\beta & u \leq \beta, \\ 1-\beta. & u > \beta \end{cases}$$

Fix $u$ and $\beta$. From (4.9) it can be shown that $u - \alpha \leq \varepsilon^{(i)} \leq u$ for all $i$. We consider in turn the situation $u \leq \beta$ and $u > \beta$, corresponding to the two possible values of the sums of the unnormalised weights after resampling.

Firstly, assume $u \leq \beta$. Then we have

$$\begin{aligned} \left| \sum_{j=1}^{i} \left( w^{(j)} - \bar{W}^{(j)} \right) \right| &= \left| \sum_{j=1}^{i} \left( w^{(j)} - W^{(j)}/(1-\beta+\alpha) \right) \right| \\ &= \frac{1}{1+\alpha-\beta} \left| \varepsilon^{(i)} + (\alpha-\beta) \sum_{j=1}^{i} w^{(j)} \right| \\ &\leq \frac{1}{1+\alpha-\beta} \\ &\quad \max \left\{ u + (\alpha-\beta) \sum_{j=1}^{i} w^{(j)}, \alpha - u - (\alpha-\beta) \sum_{j=1}^{i} w^{(j)} \right\}, \end{aligned}$$

where the two terms we are maximising over correspond to the largest positive and negative values of $\varepsilon^{(i)}$. Now, as $u \leq \beta$ and $0 < \sum_{j=1}^{i} w^{(j)} < 1$, both these terms are bounded above by $\alpha$. Thus we have mKSD $< \alpha$ in this case.

Now if $u > \beta$, by a similar argument we obtain

$$\left| \sum_{j=1}^{i} \left( w^{(j)} - \bar{W}^{(j)} \right) \right| \leq \frac{1}{1-\beta} \max \left\{ u - \beta \sum_{j=1}^{i} w^{(j)}, \alpha - u + \beta \sum_{j=1}^{i} w^{(j)} \right\} \leq \frac{\alpha}{(1-\beta)}.$$

The last inequality uses the fact that $u \leq \beta$ and $0 < \sum_{j=1}^{i} w^{(j)} < 1$. Finally as $\beta < \alpha$ we can obtain that mKSD $< \alpha/(1-\alpha)$.

# Chapter 5

# Efficient Bayesian Analysis of Multiple Changepoint Models with Dependence across Segments

## 5.1 Introduction

The most fundamental assumption of the multiple changepoint model we have considered in the early chapter (see also Punskaya et al., 2002) is that the observations across segments are conditional independent of each other given the changepoint position. This limits the scope and applicability of the method we developed. Thus in this chapter, we intend to extend the model by taking into account the dependence across segments, and in particular the Markov dependence across segments. The key idea of our method is that the dependence between neighbouring segments will be through a single set of parameters, $\psi$. If we known the value of $\psi$ for the new segment, then we have independence of the observation in the segment from earlier observations. Thus by approximating the filtering distribution by $\psi$, we can use the ideas from Chapter 4 to obtain an efficient algorithm for analysing

such changepoint models.

The filtering algorithm will calculate the posterior distribution of the most recent changepoint together with the segment model prior to time $t$, for $t = 1, \ldots, n$. The posterior estimation of parameters, conditional on each realisation of changepoints, are also obtained. Given on these outputs, it is possible to calculate the smoothing distribution of changepoints backwards in time.

The computational cost of the exact filtering is $O(n^2)$. But we can introduce the resampling idea of Chapter 4 to reduce it to $O(n)$. The complexity of the smoothing algorithm depends on the number of changepoints. As this increases linearly with the number of observations, the computational cost of the smoothing algorithm is roughly linear with $n$.

We apply our model to a specific curve fitting problem which is among the most widely discussed problems of regression techniques. It usually aims to estimate an assumed functional relationship between a response and some explanatory variables given the noisy measurement, and to predict the response for new values of the co-variates.

Parametric methodologies aim to model the function. The first example of the approach is *polynomial regression* of Anderson (1962) (see also Guttman, 1967; Harger and Antle, 1968; Brooks, 1972; Halpern, 1973, for examples). But this methodology is quite limited due to its global nature, that is, a higher order polynomial is needed to approximate the whole data and the performance is poor in estimating wiggly curves. Moreover, individual observations will affect the distant parts of the curve in unexpected way, resulting a very non-robust estimation.

More modern approaches are the non-parametric methodologies such as smoothing splines (Wahba, 1990; Hastie and Tibshirani, 1990; Green and Silverman, 1994) and kernel smoother (Muller and Stadtmuller, 1987; Fan and Gijbel, 1995). These

methodologies involve a selection of the number and positions of the changepoints (a.k.a knots in non-parametric literature) that determine the segments. Selecting these changepoints has to be very careful because the small number of changepoints reduces the degree of freedom of the fitted curve and the large number of changepoints produces over-fitting. For more detailed discussion see Hansen and Kooperberg (2002). Adaptive techniques can be embedded to enable an automatic selection of changepoints. Typical frequentist examples include the work of Wahba (1975); Smith (1982); Friedman and Silverman (1989); Friedman (1991); Stone et al. (1997) and Zhou and Shen (2001).

Bayesian selection techniques are also available. The most common ideas use RJMCMC. Smith and Kohn (1996) firstly use the methodology to select the number of changepoints for an additive model. A more general methodology is to calculate the posterior distribution of the number and positions of changepoints by the RJMCMC (e.g. Denison et al., 1998; DiMatteo et al., 2001). Then we will obtain a rich class of positions of the changepoints, from which a single collection of changepoints can be picked according to its posterior probability and can be used for the spline smoothing. Other RJMCMC would calculate a higher dimensional posterior including both the changepoints and parameters, at the expense of introducing a serious additional computational burden (Mallick, 1998; Punskaya et al., 2002).

Our approach is to model the function by a *piecewise polynomial*, which consists of a sequence of low order polynomials defined within different segments. The novelty of our model is that we have defined two types of changepoints which related to whether the underlying signals is (i) continuous or (ii) discontinuous at each changepoint. This generalises the earlier models. For example, the model of Denison et al. (1998) is such a model with all changepoints continuous while the model of Punskaya et al. (2002) is such a model with all changepoints discontinuous.

The chapter is constructed as follows. In Section 5.2, we introduce a generic hierarchical model. The changepoints model we considered is a specific case of it.

In Section 5.3, the online algorithm used to calculate the posterior distribution of the positions of the changepoints is given. The posterior estimation of local parameters are also calculated. Then smoothing distribution of the changepoints is obtained in Section 5.4, given the realisations of changepoints and parameters. Two smoothing algorithms are used. A block sampling for the parameters has been introduced in Section 5.5. The resulting algorithm has been evaluated in several aspects in Section 5.6, and has been tested on a few simulated data in Section 5.7. Finally, we implement the algorithm a real data set in Section 5.8. This is a well log data with 4050 observations and around 30 changepoints. Both piecewise constant and piecewise quadratic models are used. The chapter ends up with a discussion.

## 5.2 Changepoint model

We consider the following hierarchical model for observations $y_{1:n} = (y_1, \ldots, y_n)$.

Firstly, we model the changepoints within a framework we have discussed in Chapter 3, i.e. a point process with a propability mass function $g(d)$ is considered, where $d$ is the distance between two successive changepoints. Note that this setting is exactly the same as we used in Chapter 4. In the specific applications considered in this chapter, we take $g(d)$ as a probability mass function of a geometric distribution for the changepoints, i.e. $g(d) = p(1-p)^{d-1}$. Though alternative distributions have been suggested by (Fearnhead, 2006).

The changepoints split the data into $m+1$ segments. For each segment $k$ consisting of observations $\mathbf{y}_{s+1:t}$, we associate a model $\mathcal{M}_k$ and a vector of parameters $\theta_k$. The model is drawn from a finite set of possible models labeled $1, 2, \ldots, \bar{p}$, and we assume a prior distribution for the model of a segment and that there is independence of the choice of model across different segments.

We split the parameter into two components $\theta_k = (\psi_k, \phi_k)$, with the $\psi_k$ component being allowed to depend on features of the previous segment and the $\phi_k$ component just depending on the $\psi_k$ value and model of the current segment. Thus for segment $k = 1, \ldots, m$, we will have that the conditional probability of the model and parameters for the segment can be factorised as

$$p(\mathcal{M}_k)p_{\mathcal{M}_k}(\psi_k|\psi_{k-1}, \phi_{k-1}, \tau_k, \tau_{k-1})p_{\mathcal{M}_k}(\phi_k|\psi_k).$$

For the first segment we assume a prior for $\phi_0$. In particular we will consider a family of prior distributions $p(\phi_k|\zeta_{s\mathcal{M}_k})$,for $k = 1, \ldots, m$ with $\phi_0$ being drawn from $p(\phi_0|\zeta_{0\mathcal{M}_1})$ for some known value $\zeta_{0\mathcal{M}_1}$. Note that the $\zeta_{s\mathcal{M}_k}$s denoting hyper-parameters in the interval beginning at $s$ and associated with model $\mathcal{M}_k$.

Finally we have a likelihood model for the observations within a segment. Given the $k$th segment, and with model $\mathcal{M}_k$ and parameter $(\phi_k, \psi_k)$, we have a likelihood model

$$p_{\mathcal{M}_k}(\mathbf{y}_{s+1:t}|\psi_k, \phi_k). \tag{5.1}$$

We assume that conditional on the changepoints, segment models and parameters, that the observations within each segment are independent of each other.

We also make some assumptions about the priors for $\phi_k$ and $\psi_k$. In particular we assume that for all values of $\zeta_{s\mathcal{M}_k}$ and $\mathcal{M}_k$ that the joint prior $p_{\mathcal{M}_k}(\psi_k|\zeta_{s\mathcal{M}_k})p_{\mathcal{M}_k}(\phi_k|\psi_k)$ is conjugate for the likelihood of the observations within a segment. Thus, we define the marginal likelihood function similar to that in Chapter 4:

$$\begin{aligned} & P(s, t, \mathcal{M}_k, \zeta_{s\mathcal{M}_k}) \\ = \; & P(\mathbf{y}_{s+1:t}|\text{model } \mathcal{M}_k, \text{ parameter } \zeta_{s\mathcal{M}_k}) \\ = \; & \int\int p_{\mathcal{M}_k}(\mathbf{y}_{s+1:t}|\psi_k, \phi_k)p_{\mathcal{M}}(\psi_k|\zeta_{s\mathcal{M}_k})p_{\mathcal{M}_k}(\phi_k|\psi_k)\mathrm{d}\phi_k\mathrm{d}\psi_k. \end{aligned} \tag{5.2}$$

Under our assumption of conjugate priors, we have that these marginal likelihoods can be calculated analytically for all values of $\zeta_{s\mathcal{M}_k}$.

The way we modelled the data can be only approximately inferred. The key idea is to update the prior distribution of $\psi_k$ segment by segment, from $k = 0$ to $m$. That is we use the posterior distribution of $\psi_{k-1}$ as the prior distribution of $\psi_k$ if there is a dependence across these two segments.

**Illustration**

Figure 5.1 and 5.2 give two specific cases that we have to consider the dependence structure in the multiple changepoints model. If the observations across the two segments are independent of each other, a curve (e.g. the red lines in the plots) in the second segment can be fitted based only on the observations in the segment. This is the examples in Chapter 4. However, if we require a connection at the changepoint between the curves in each segment, we have to take the dependence between the observations into account.

We fit the curve per segment, say we have $y = a_k + b_k x$ for $k = 1, 2$ in Figure 5.1. $b_k$ in each segment, which is the slope of the curve is independent of each other, so we have $\phi_k = b_k$, and we assume $b_k$s are i.i.d under a prior distribution $F_b$. If we use a similar prior distribution for $a_k$s, it is likely to generate a red curve in Figure 5.1. To make sure the curves in each segment are connected at the changepoint, we assign an informative prior to $a_2$ so that it contains the information from the first segment. To achieve this, the posterior distribution of the hyper-parameters of $a_1$ can be used as the prior distribution of the hyper-parameters of $a_2$. Hence we have $\psi_2 = a_2$.

We now give some quadratic regression examples of our generic dependent changepoints models. The last of these models will form the basis of the application of

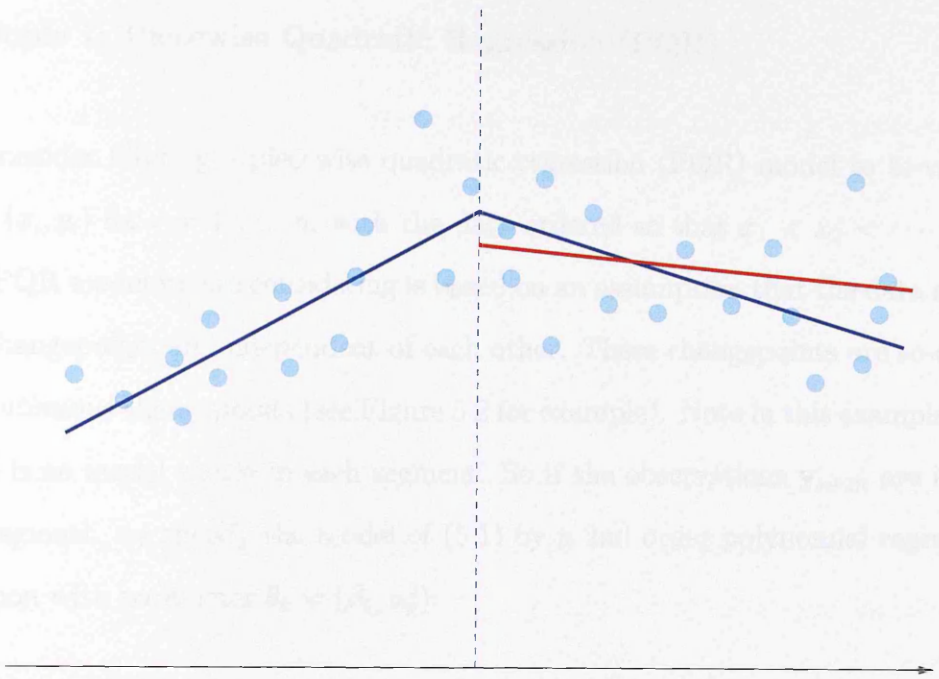

Figure 5.1: An example of a dependent changepoint and an independent changepoint. The vertical dotted line indicates the position of changepoint.

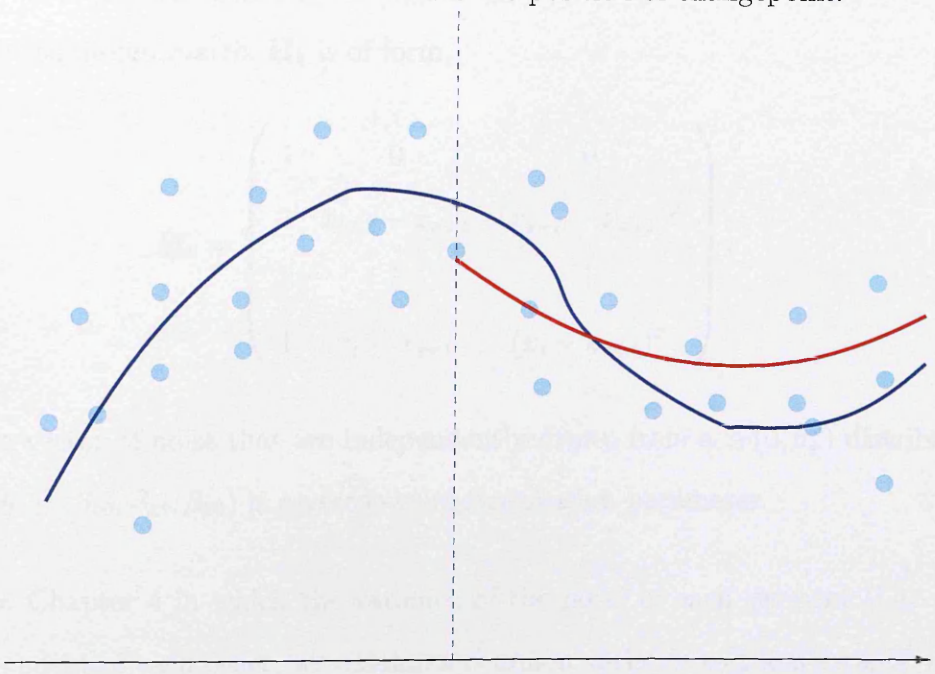

Figure 5.2: An example of a continuous changepoint and a discontinuous changepoint. The vertical dotted line indicates the position of changepoint

our method in Sections 5.7.

### Example 1: Piecewise Quadratic Regression (PQR)

We consider filtering a piecewise quadratic regression (PQR) model to bi-variate data $(x_i, y_i)$ for $i = 1, \ldots, n$, with the data ordered so that $x_1 < x_2 < \cdots < x_n$. The PQR model we are considering is based on an assumption that the data across the changepoints are independent of each other. These changepoints are so-called *discontinuous* changepoints (see Figure 5.2 for example). Note in this example that there is no model choice in each segment. So if the observations $\mathbf{y}_{s+1:t}$ are in the $k$th segment, we specify the model of (5.1) by a 2nd order polynomial regression function with parameter $\theta_k = (\beta_k, \sigma_k^2)$:

$$\mathbf{y}_{s+1:t} = \mathbf{H}_k \beta_k + \varepsilon_k, \tag{5.3}$$

where the design matrix $\mathbf{H}_k$ is of form

$$\mathbf{H}_k = \begin{pmatrix} 1 & 0 & 0 \\ 1 & x_{s+2} - x_{s+1} & (x_{s+2} - x_{s+1})^2 \\ \vdots & \vdots & \vdots \\ 1 & x_t - x_{s+1} & (x_t - x_{s+1})^2 \end{pmatrix},$$

$\varepsilon_k$ is a vector of noise that are independently drawn from a $N(0, \sigma_k^2)$ distribution, and $\beta_k = (\beta_{k0}, \beta_{k1}, \beta_{k2})$ is a vector-valued regression parameter.

Unlike Chapter 4 in which the variance of the noise in each segment (i.e. $\sigma_k^2$) is independent of each other, we assume a common variance to the noises, $\sigma^2$ across all the segments. However, to hit the above framework, we introduce $\sigma_k^2$ to denote its value in the $k$th segment, hence

$$\sigma_k^2 = \sigma^2, \text{ for } k = 1, \ldots, m. \tag{5.4}$$

Note hat the $\sigma_k^2$ in the $k$th segment is dependent on $\sigma_{k-1}^2$ in the previous segment (since they are actually equal); and $\beta_k$ is dependent only on the value of $\sigma_k^2$. Hence, using the notation above, we have $\psi_k = \sigma_k^2$, $\phi_k = \beta_k$.

We use the following priors for the variance $\sigma_k^2$ and the regression parameter $\beta_k$, for the sake of conjugacy:

$$\begin{aligned} \psi_k &\sim IG(\nu_k/2, \gamma_k/2), \\ \phi_k|\psi_k &\sim MVN(\vec{0}, \psi_k \mathbf{D}_k), \end{aligned}$$

where $IG$ denotes the inverse Gamma distribution and $MVN$ denotes the multivariate normal distribution. $\mathbf{D}_k = diag(\delta_{0k}^2, \delta_{1k}^2, \delta_{2k}^2)$ is a diagonal matrix. With the notations above, we have $\zeta_{s\mathcal{M}_k} = (\nu_k, \gamma_k)$, the value of which should be identical in each segment.

Note that the regression model we considered in Chapter 4 is basically equivalent to PQR, except for allowing a choice on the model order in each segment. Hence we can drop $\mathcal{M}_k$ in the marginal likelihood of (5.2) and calculate it analytically with the above prior settings by (4.4). See Appendix A in Chapter 4 for details.

**Example 2: Continuous PQR**

We can also allow for the dependence across segments, by assuming continuity of the underlying function across the changepoints, (see Denison et al., 1998; DiMatteo et al., 2001, for examples). These changepoints are called *continuous* changepoints. See Figure 5.2 for an example.

As the data are dependent across segments, we have to model the data in a whole batch perspective such that

$$\mathbf{y}_{1:n} = \mathbf{H}\beta + \varepsilon, \tag{5.5}$$

where $\mathbf{H}$ is an $n \times (2m+3)$ design matrix. If we define $k \in \{0, \ldots, m\}$ and $l \in \{1, 2\}$ then the $i$th row and $j$th column element of the matrix is defined as

$$\mathbf{H}_{ij} = \begin{cases} 1 & \text{if } j = 1; \\ (x_i - x_{\tau_k+1})^l & \text{if } j = 2k+l+1, \text{ and } \tau_k < i \leq \tau_{k+1} \\ (x_{\tau_{k+1}} - x_{\tau_k+1})^l & \text{if } j = 2k+l+1, \text{ and } \tau_{k+1} > i \\ 0 & \text{otherwise .} \end{cases} \tag{5.6}$$

Note that the value of $k$ denotes the segment corresponding to column $j$, and $l$ denotes whether the column suits to the linear or quadratic component. The entry then dependent on whether the $i$th observations lies in the $k$th segment, is after the $k$th segment or is before the $i$th segment.

The regression parameter is

$$\beta = (\beta_{00}, \beta_{01}, \beta_{02}, \beta_{11}, \beta_{12}, \ldots, \beta_{m1}, \beta_{m2})^T. \tag{5.7}$$

For segment $k$, we let $\sigma_k^2$ and $\beta_{k0}$ denote the variance of $\varepsilon$ in (5.5) and the value of the regression function at the start of the segment, respectively. $\beta_{k0}$ is determined by $\beta_{(k-1)0}, \ldots, \beta_{(k-1)2}$ via a linear predictor:

$$\beta_{k0} = \beta_{(k-1)0} + \beta_{(k-1)1}(x_{\tau_k+1} - x_{\tau_{k-1}+1}) + \beta_{(k-1)2}(x_{\tau_k+1} - x_{\tau_{k-1}+1})^2.$$

The idea is that the dependence between neighbouring segments is only through these two parameters. Thus we have $\psi_k = (\beta_{k0}, \sigma_k^2)$. Finally our conjugate priors are

$$\begin{aligned} \psi_k &= \begin{cases} \sigma_k^2 & \sim IG(\frac{\nu_k}{2}, \frac{\gamma_k}{2}), \\ \beta_{k0}|\sigma_k^2 & \sim N(\mu_k, \sigma_k^2\eta_k^2), \end{cases} \\ \phi_k|\psi_k &\sim MVN(\vec{0}, \sigma_k^2 \mathbf{D}_k') \end{aligned}$$

where $\mathbf{D}'_k = diag(\delta^2_{1k}, \delta^2_{2k})$. Similarly, we denote $\zeta_{s\mathcal{M}_k} = (\mu_k, \eta^2_k, \nu_k, \gamma_k)$, and this is the posterior estimation from $(k-1)$th segment.

Again, we do not have model choices in each segment. Thus the analytical form of (5.2) is:

$$\begin{aligned} & P(s,t,\zeta) \\ = & \int_{\sigma^2_k}\int_{\tilde{\beta}_k} p(\tilde{\mathbf{y}}_{s+1:t}|\tilde{\beta}_k,\sigma^2_k)p(\tilde{\beta}_k)p(\sigma^2_k)\mathrm{d}\tilde{\beta}_k\mathrm{d}\sigma^2_k \\ = & \pi^{-(t-s)/2}\left(\frac{|\mathbf{M}_k|}{|\mathbf{D}'_k|}\right)^{\frac{1}{2}} \frac{\gamma^{\nu_k/2}}{(||\tilde{\mathbf{y}}_{s+1:t}||^2_{\mathbf{P}_k}+\gamma_k)^{(t-s+\nu_k)/2}} \times \\ & \frac{\gamma((t-s+\nu_k)/2)}{\Gamma(\nu_k/2)}, \end{aligned} \tag{5.8}$$

where

$$\begin{aligned} \tilde{y}_j &= y_j - \mu_k, \qquad j = s+1,\ldots,t, \\ \tilde{\beta}_{k0} &\sim N(0,\sigma^2_k\eta^2_k), \end{aligned}$$

and the forms of $\mathbf{M}_k$, $\mathbf{P}_k$ and $\mathbf{D}_k$ change correspondingly. See Appendix A for the justification of reparameterisation.

### Example 3: Continuous/Discontinuous PQR

For the final example, we consider a model combining Example 1 and 2. We do this by introducing two models for each segment. These models correspond to whether the changepoint at the beginning of the segment is continuous or not. The $\psi_k$ and $\phi_k$ and their priors are defined as for Example 2.

## 5.3 Forward filtering

We now introduce an online algorithm for the hierarchical model of Section 5.2. It borrows the idea of particle filters to approximate the posterior distributions of interest, such as the position of changepoints $\tau_k$, the corresponding model $\mathcal{M}_k$, and the associated function parameter $\theta_k$.

### 5.3.1 Filtering recursion

Firstly, we have to introduce an artificial time $t$ to enable the online inference on the model. So at each time $t$, we have a new data $y_t$ observed. Then we denote $C_t$ the most recent changepoint prior to time $t$ and $M_t$ the model being used in the segment between in $C_t$ and $t$. The corresponding regression parameters at time $t$ are denoted as $\Theta_t = (\Psi_t, \Phi_t)$.

Given the process of the changepoints we discussed in Section 5.2, the $C_t$ follow a Markov process with the transitional probability of (4.5). Our aim is to recursively calculate the filtering distributions $p(C_t, M_t, \mathbf{y}_{1:t})$.

#### When there is no new changepoint

If $C_{t+1} = C_t$, that means there is no new changepoint and the model still applies to observation $y_{t+1}$. Suppose the last changepoint before time $t$ is $\tau_k = s$ $(s < t)$, so we have

$$\begin{aligned} C_{s+1} &= \cdots = C_{t+1} = s, \\ M_{s+1} &= \cdots = M_{t+1} = \mathcal{M}, \\ \Psi_{s+1} &= \cdots = \Psi_{t+1} = \psi, \end{aligned}$$

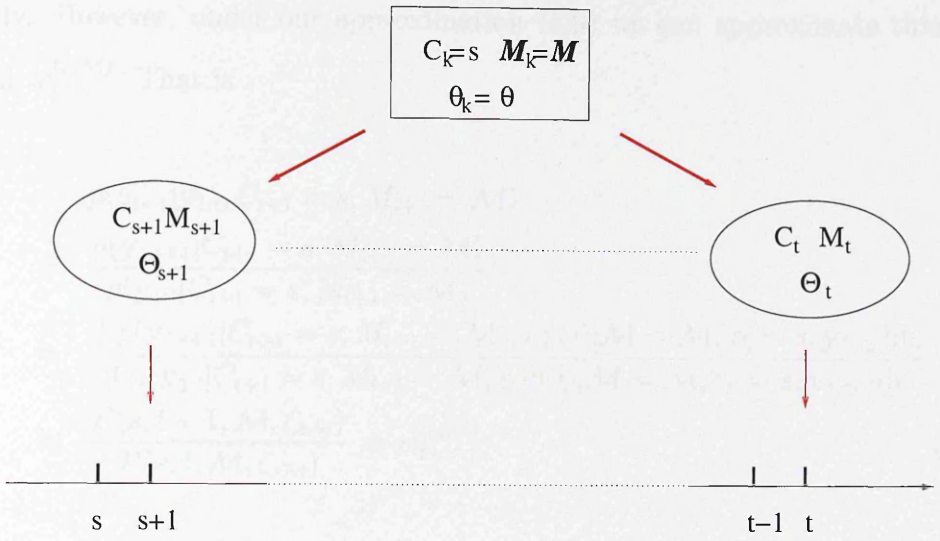


Figure 5.3: A demonstration of $C_t$, $M_t$ and $\Theta_t$ and their values (shown in the rectange)

The key idea to our approach is that $p(\psi|\mathbf{y}_{1:\tau_k}, \tau_k = s, \mathcal{M}_k = \mathcal{M})$ can be approximated by a member of our conjugate prior family for $\psi$ that is

$$\pi(\psi|\zeta_{s\mathcal{M}}) \qquad \text{for some } \zeta_{s\mathcal{M}}.$$

Particularly, we define $\zeta_{s\mathcal{M}}$ to be the value of hyper-parameter so that we have the approximation:

$$p(\psi|\mathbf{y}_{1:\tau_k}, \tau_k = s, \mathcal{M}_k = \mathcal{M}) \approx \pi(\psi|\zeta_{s\mathcal{M}}). \tag{5.9}$$

Now we can use a standard recursive formula for the filtering distribution at time $t+1$:

$$p(C_{t+1} = s, M_{t+1} = \mathcal{M}|\mathbf{y}_{1:t+1}) \propto$$
$$p(y_{t+1}|\mathbf{y}_{1:t}, C_{t+1}, M_{t+1})p(C_{t+1}|C_t = s)p(M_{t+1})p(C_t = s, M_t = \mathcal{M}|\mathbf{y}_{1:t}). \tag{5.10}$$

See Figure 5.3 for a illustration. For our model, we can not calculate $p(y_{t+1}|\mathbf{y}_{1:t}, C_{t+1}, M_{t+1})$

exactly. However, under our approximation (5.9) we can approximate this by a weight $w_{t+1}^{(s,\mathcal{M})}$. That is

$$\begin{aligned} & p(y_{t+1}|\mathbf{y}_{1:t}, C_{t+1} = s, M_{t+1} = \mathcal{M}) \\ = & \frac{p(\mathbf{y}_{1:t+1}|C_{t+1} = s, M_{t+1} = \mathcal{M})}{p(\mathbf{y}_{1:t}|C_{t+1} = s, M_{t+1} = \mathcal{M})} \\ = & \frac{\int p(\mathbf{y}_{1:t+1}|C_{t+1} = s, M_{t+1} = \mathcal{M}, \psi)p(\psi|\mathcal{M} = \mathcal{M}, \tau_k = s, \mathbf{y}_{1:\tau_k})\mathrm{d}\psi}{\int p(\mathbf{y}_{1:t}|C_{t+1} = s, M_{t+1} = \mathcal{M}, \psi)p(\psi|\mathcal{M} = \mathcal{M}, \tau_k = s, \mathbf{y}_{1:\tau_k})\mathrm{d}\psi} \\ \approx & \frac{P(s, t+1, \mathcal{M}, \zeta_{s\mathcal{M}})}{P(s, t, \mathcal{M}, \zeta_{s\mathcal{M}})} := w_{t+1}^{(s,\mathcal{M})}, \end{aligned} \tag{5.11}$$

As with the calculation of $w_{t+1}^{(s,\mathcal{M})}$, it has a computational complexity of $O(t-s)$. However, it is possible to calculate the $w_{t+1}^{(s,\mathcal{M})}$ recursively. This recursive computing only has a computational cost of $O(1)$ with sequential updated $\zeta$. Appendix B gives details how the calculation is implemented on our specific curve fitting examples.

### When there is a new changepoint

We now consider calculating $p(C_{t+1} = t, M_{t+1} = \mathcal{M}|\mathbf{y}_{1:t+1})$ which comes to a new changepoint at time $t$. We denote the parameter in the new segment $\theta = (\psi, \phi)$ and the parameter in the old segment $\theta' = (\psi', \phi')$. However, $\psi$ is determined by the model as well as the information in the previous segment: the previous parameter $\theta'$, the previous changepoint $C_t = s'$ and model $M_t = \mathcal{M}'$. Thus we obtain

$$\begin{aligned} & p_{\mathcal{M}}(\psi|C_{t+1} = t, M_{t+1} = \mathcal{M}, \mathbf{y}_{1:t}) \\ \approx & \sum_{s'} \sum_{\mathcal{M}'} p_{\mathcal{M}}(\psi|\zeta_{s'\mathcal{M}'}, s', \mathcal{M}', \mathbf{y}_{s'+1:t})p(s', \mathcal{M}'|\tau_k = t, \mathbf{y}_{1:t}), \end{aligned} \tag{5.12}$$

where

$$\begin{aligned} & p_{\mathcal{M}}(\psi|\zeta_{s'\mathcal{M}'}, s', \mathcal{M}', \mathbf{y}_{s'+1:t}) \\ \propto & \int_{\phi'}\int_{\psi'} p_{\mathcal{M}}(\psi|\psi', \phi', s', \mathcal{M}')\pi(\psi'|\zeta_{s'm'}) \\ & \times\pi(\phi'|\psi')p(\mathbf{y}_{s'+1:t}|\psi')\mathrm{d}\psi'\mathrm{d}\phi', \end{aligned} \tag{5.13}$$

and

$$p(s', \mathcal{M}'|\tau_k = t, \mathbf{y}_{1:t}) \propto p(C_t = s', M_t = \mathcal{M}'|\mathbf{y}_{1:t})p(C_{t+1} = t|C_t = s'). \tag{5.14}$$

We approximate (5.12) by $\pi(\psi|\zeta_{s\mathcal{M}})$, for some $\zeta_{s\mathcal{M}}$. (There are many possibilities for this approximation; our approach is to calculate $\zeta_{s\mathcal{M}}$ via a method of moments procedure - see below).

Using our approximation, we set

$$\begin{aligned} & p(C_{t+1} = t, M_{t+1} = \mathcal{M}|\mathbf{y}_{1:t+1}) \\ \propto & \; p(y_{t+1}|C_{t+1} = t, M_{t+1} = \mathcal{M})p(M_{t+1} = \mathcal{M}) \\ & \times\left(\sum_{s'}\sum_{m'} p(C_{t+1} = t|C_t = s')p(C_t = s', M_t = \mathcal{M}'|\mathbf{y}_{1:t})\right) \\ \approx & \; w_{t+1}^{(t,\mathcal{M})}p(M_{t+1} = \mathcal{M}) \\ & \times\left(\sum_{s'}\sum_{m'} p(C_{t+1} = t|C_t = s')p(C_t = s', M_t = \mathcal{M}'|\mathbf{y}_{1:t})\right) \end{aligned} \tag{5.15}$$

where

$$w_{t+1}^{(t,\mathcal{M})} = P(t, t+1, \mathcal{M}, \zeta_{t\mathcal{M}}). \tag{5.16}$$

We now give details for calculating $\zeta_{s\mathcal{M}}$ for our 3 examples.

**Example 1: PQR**

As $\psi = \psi'$, (5.12) simplifies to

$$\sum_{s'} p(\psi|\zeta_{s'}, s', \mathbf{y}_{s'+1:t}) p(s'|\tau_k = t, \mathbf{y}_{1:t}),$$

We choose $\zeta_{s\mathcal{M}'}$ to match the first and second moments of $\psi^{-1} = \sigma^{-2}$ (i.e $\mathrm{E}(\psi^{-1})$ and $\mathrm{E}(\psi^{-2})$) between (5.12) and $\pi(\psi|\zeta_{s'})$. In this particular case, the $\zeta_{s\mathcal{M}'}$ can be calculated explicitly since it is the parameters of an inverse Gamma distribution (such that $\mathrm{E}(\psi^{-1}) = \nu/\gamma$).

**Example 2: Continuous PQR**

Here $\psi = (\sigma_k^2, \beta_{k0})$ and $\zeta_{s\mathcal{M}_k} = (\mu_k, \eta_k, \nu_k, \gamma_k)$ where $\mu_k$, $\eta_k$, $\nu_k$ and $\gamma_k$ are hyper-parameters for $\beta_{k0}$ and $\sigma_k^2$. The update of $\nu$ and $\gamma$ are as in Example 1. So we concentrate on the update of $\mu$ and $\eta^2$ for the intercept $\beta_{k0}$. According to (5.12), we have the following equation:

$$\begin{aligned} & p(\beta_{k0}|\mathbf{y}_{1:t}, C_{t+1} = t) \\ = & \sum_{s'} p(\beta_{k0}|\zeta_{s'}, C_t = s', \mathbf{y}_{s'+1:t}) p(C_t = s'|\tau_k = t, \mathbf{y}_{1:t}), \end{aligned}$$

where the $p(\beta_{k0}|\zeta_{s'}, C_t = s', \mathbf{y}_{s'+1:t})$ follows a normal distribution with mean and variance calculated from a distribution $p(\beta_k|\zeta_{s'}, C_t = s', \mathbf{y}_{s'+1:t})$ that is a multivariate normal distribution of $\beta_{k-1} = (\beta_{(k-1)0}, \beta_{(k-1)1}, \beta_{(k-1)2})$. Using $\beta_{k0} = \mathbf{h}_{s'}\beta_{k-1}$ where $\mathbf{h}_{s'} = (1, (x_t - x_{s'+1}), (x_t - x_{s'+1})^2)$, we then have

$$\begin{aligned} \mu_k &= \mathrm{E}(\beta_{k0}) = \sum_{s'} \mathbf{h}_{s'} \mathrm{E}(\beta_{k-1}|s') p(s'|\tau_k = t, \mathbf{y}_{1:t}) \\ \eta_k^2 &= \mathrm{var}(\beta_{k0})/E(\sigma^2) = \sum_{s'} \left(\mathbf{h}_{s'} \mathrm{var}(\beta_{k-1}) \mathbf{h}_{s'}^T\right) p(s'|\tau_k = t, \mathbf{y}_{1:t})/\mathrm{E}(\sigma^2) \end{aligned}$$

where $\mathrm{E}(\sigma^{-2}) = \nu_s/\gamma_s$.

**Example 3: Continuous/discontinuous PQR**

For this example, we only need to adapt the different calculations of $\zeta_{s\mathcal{M}}$ for different types of changepoints. At the continuous changepoint, $M_t = 1$, we have the same calculation as Example 2. For discontinuous changepoints, $M_t = 2$, we calculate the $\nu$ and $\gamma$ as for Example 1. The value of $\mu$ and $\eta^2$ are only dependent on these two parameters.

### 5.3.2 Filtering with resampling

Storing $p(C_t, M_t|\mathbf{y}_{1:t})$ at any time $t$ actually involves a storage of $\bar{p}t$ different values of $(C_t, M_t)$ which we call particles, with associated probability. On top of the linear increase of the memory storage, the computing cost increase quadratically. To reduce this, the resampling idea introduce in Chapter 4 can be used so that we approximate $p(C_t, M_t|\mathbf{y}_{1:t})$ by a set of $K \ll \bar{p}t$ particles and their associated probabilities. This can reduce the computing cost to be $O(t)$. In the examples that will be tested in Section 5.7, we make an extension of the SRC method developed in Chapter 4. The specific algorithm is following:

**Algorithm 5.1 Initialisation** *At a certain time $t$, we have $n$ particles $(C_t = s, M_t = \mathcal{M})$ for $i = 1, \ldots, n$ which are indexed in an ascending order of positions of change points. For those who have same positions we index them in an ascending order of $m$ (so first discontinuous then continuous). The particles are associated with weights $w_t^{(s,\mathcal{M})} = p(C_t = s, M_t = \mathcal{M}|\mathbf{y}_{1:t})$. Choosing a threshold, $\alpha = 10^{-6}$ say, we have:*

**(SRC1)** *For $i = 1, \ldots, n$ if $w_t^{(s,\mathcal{M})} \geq \alpha$ then keep particle $(C_t = s, M_t = \mathcal{M})$ with weight $w_t^{(s,\mathcal{M})}$. Assume that $A$ particles are kept.*

**(SRC2)** *Use the stratified resampling algorithm of Carpenter et al. (1999) to resample from the ordered set of the remaining $N - A$ particles (without shuf-*

*fling). The marginal probability of resampling particle* $(C_t = s, M_t = \mathcal{M})$ *is* $w_t^{(s,\mathcal{M})}/\alpha$. *Each resampled particle is assigned a weight* $\alpha$.

## 5.4 Backward smoothing

The filtering procedure produces the distribution of $p(C_t, M_t|\mathbf{y}_{1:t})$. Often our interest is in the joint distribution of all changepoints and models. Simulating from this joint distribution is possible by constructing a recursive backward procedure to simulate the marginal distribution of $p(C_t, M_t|\psi, C_{t+1} = t, M_{t+1} = \mathcal{M}', \mathbf{y}_{1:n})$ from $t = n-1$ to 1, and $\mathcal{M}' = 1, \ldots, \tilde{p}$.

We conditional on $C_{t+1}$ and $M_{t+1}$ due to the dependence structure of the observations, and the reason we have to conditional on $\psi$ is because it is common to all segments such that

$$p(C_t, M_t|\psi, C_{t+1} = t, M_{t+1} = \mathcal{M}', \mathbf{y}_{1:n}) \propto p(C_t, M_t|\psi, C_{t+1} = t, M_{t+1} = \mathcal{M}', \mathbf{y}_{1:t}),$$

given a fixed value of $\psi$. We can simulate $\psi$ according to the changepoints and segment model, as well as the previous value of $\psi$ in the segment. Thus for the marginal distribution, we have

$$\begin{aligned} & p(C_t = s, M_t = \mathcal{M}|\psi, t, \mathcal{M}', \mathbf{y}_{1:n}) \\ \propto\ & p(C_t = s, M_t = \mathcal{M}, \psi|t, \mathcal{M}', \mathbf{y}_{1:t}) \\ =\ & p(C_t = s, M_t = \mathcal{M}|\mathbf{y}_{1:t}) p(C_{t+1} = t|C_t = s) p(M_{t+1} = \mathcal{M}') \\ & \times p(\psi|\mathbf{y}_{1:t}, C_t = s, M_t = \mathcal{M}). \end{aligned} \tag{5.17}$$

The first part of the product has already been calculated and stored during the filtering procedure, the $p(C_{t+1} = t|C_t = s)$ and $p(M_{t+1} = \mathcal{M}')$ are already known, so we only need to calculate the posterior distribution of $\psi$, which is equivalently

to measure how good the simulated value of $\psi$ is under the current model. The smoothing algorithm is detailed as the following:

**Algorithm 5.2** *Simulating from smoothing distribution backwardly*

*Initialise:* $t \leftarrow n$;

*Simulate* $C_n$ *and* $M_n$ *from* $p(C_n, M_n|\mathbf{y}_{1:n})$;

*Simulate* $\psi$ *from*

$$\hat{p}(\psi|\mathbf{y}_{1:n}, C_n = s, M_n = \mathcal{M}) \propto \pi(\psi|\zeta_{s\mathcal{M}})p(\mathbf{y}_{s+1:n}|\psi);$$

*Let* $\psi' \leftarrow \psi$;

*Let* $t \leftarrow C_n$;

*While* $t > 0$

- *calculate* $\hat{p}(\psi'|\mathbf{y}_{1:t}, C_t, M_t) \propto \pi(\psi'|\zeta_{tM_t})$
- *simulate* $C_t$ *and* $M_t$ *from the distribution proportional to*

$$p(C_t, M_t|\mathbf{y}_{1:t})p(C_{t+1}|C_t)p(\psi'|C_t, M_t, \mathbf{y}_{1:t});$$

- *simulate* $\psi$ *from*

$$p(\psi|\psi', C_t, M_t, \mathbf{y}_{1:t}) \propto \pi(\psi|\zeta_{C_tM_t})p(\mathbf{y}_{C_t+1:t}|\psi)p(\psi'|\psi, \mathbf{y}_{C_t+1:t});$$

- *Let* $\psi' \leftarrow \psi$;

**Example 1: PQR**

For the discontinuous PQR in which $\psi = \sigma^2$, the smoothing procedure is fairly simple. Let $\zeta_s = (\nu_s, \gamma_s)$, we only need to simulate $\psi$ once at time $n$, and calculate

the density of $\psi$ at each time $t$, which is

$$\begin{aligned} p(\psi'|C_t = s, \mathbf{y}_{1:t}) &= p(\psi'|\zeta_s, \mathbf{y}_{s+1:t}) \\ &\sim IG\left(\frac{\nu'_s}{2}, \frac{\gamma'_s}{2}\right), \end{aligned} \tag{5.18}$$

where

$$\begin{aligned} \nu'_s &= \nu_s + t - s, \\ \gamma'_s &= \gamma_s + ||\mathbf{y}_{s+1:t}||^2. \end{aligned}$$

**Example 2: Continuous PQR**

The smoothing becomes much more complicated in the continuous PQR case, as we have $\psi = (\sigma^2, (\beta_{k0})_{k=0,\ldots,m-1})$. The $(\beta_{k0})_{k=0,\ldots,m-1}$ are intercepts in each segment except for the last one, and when simulating backwardly, each of them is dependent on the value of previous simulation. The last intercept $\beta_{m0}$ can be simulated independently.

Specifically speaking, the $\sigma^2$ can still be simulated once at time $n$, as it is a common parameter to all segments. The calculation of the density is the same as Example 1. Also, $\beta_{m0}$ can be simulated directly according to the posterior mean and variance calculated during the filtering procedure. That is

$$p(\beta_{m0}|\sigma^2) \sim N\left((\mathbf{M}'\mathbf{h}^T_{\tau_{m-1}}(y_{\tau_m} - \mu_s))_{11} + \mu_s, \sigma^2\eta_s^2\right),$$

where

$$\begin{aligned} \mathbf{h}_{\tau_{m-1}} &= (1, (x_{\tau_m} - x_{\tau_{m-1}+1}), (x_{\tau_m} - x_{\tau_{m-1}+1})^2), \\ \mathbf{M}' &= (\mathbf{h}^T_{\tau_{m-1}}\mathbf{h}_{\tau_{m-1}} + \mathbf{D}^{-1})^{-1}, \\ \mathbf{D} &= diag(\eta^2, \delta_1^2, \delta_2^2), \end{aligned}$$

and $(\mathbf{M})_{11}$ denotes the first row and the first column of the matrix $\mathbf{M}$.

Then we can simulate $\beta_{k0}$ conditional on the value of $\beta_{(k+1)0}$ which has already been drawn from its segment. However, to simulate $(\beta_{k0})_{k=0,\ldots,m-1}$, we need a conditional sampling methods suggested by Rue and Held (2005). This is actually calculating the conditional mean and variance of $\beta_k$ subjecting to $\mathbf{H}_k\beta_k = \beta_{(k+1)0}$ by following formula,

$$\begin{aligned}\vec{\mu}^* &= \vec{\mu} - Q\mathbf{H}_k^T(\mathbf{H}_kQ\mathbf{H}_k^T)^{-1}(\mathbf{H}_k\vec{\mu} - \beta_{(k+1)0}),\\ Q^* &= Q - Q\mathbf{H}_k^T(\mathbf{H}_kQ\mathbf{H}_k^T)^{-1}\mathbf{H}_kQ,\end{aligned}$$

where the $\vec{\mu}$ and $Q$ are the original mean and variance of $\beta_k$, which are

$$\begin{aligned}\vec{\mu} &= \mathbf{M}_k\mathbf{H}_k^T\mathbf{y}_{\tau_k+1:\tau_{k+1}},\\ Q &= \sigma^2\mathbf{M}_k,\end{aligned}$$

respectively. Based on the $\vec{\mu}^*$ and $Q^*$, the $\beta'_k$ can be simulated, and we simply take the first element of it as the estimated intercept. With the simulated $\beta'_{k0}$, the density of $\beta_{k0}$ can be calculated that is

$$p(\beta_{k0}|\beta'_{k0}, \zeta, \mathbf{y}_{\tau_k+1:\tau_{k+1}}) \sim N(\vec{\mu}^*, Q^*).$$

### Continuous/discontinuous PQR

Again, we have to simulate both the positions $(C_t)$ and types $(M_t)$ of changepoints, and the smoothing algorithm developed in Example 1 and 2 should be properly used according to the value of $M_t$. The most distinctive part of the algorithm is that the simulation of intercept $(\beta_{k0})_{k=0,\ldots,m-1}$ under each model is different. If the changepoint is discontinuous, the intercept is drawn from normal distributing with mean and variance already calculated during the filtering procedure. If

the changepoint is continuous, the intercept is drawn from a conditional normal distribution as discussed in Example 2.

## 5.5 Parameter estimation

All the parameters have already been estimated concurrently with the filtering procedure, as we have discussed in Section 5.3. A better approach is to estimate these parameters in a batch perspective given all the positions and types of changepoints are simulated. With appropriate conjugate priors such as those we specified in our examples, an analytical solution to the posterior estimation is available, but can be quite computing expensive, particularly in Example 2 and 3. Alternatively, we can approximate the estimation by some numerical methods.

## 5.6 Evaluation of methodology

Our methodology only provides approximate inference of the model due to the following two reasons: (i) the assumption of conditional independence in approximating the marginal likelihood $p(y_{t+1}|\mathbf{y}_{1:t}, C_{t+1} = s, M_{t+1} = \mathcal{M})$ and; (ii) the use of resampling scheme in approximating the posterior distribution $p(C_t, M_t|\mathbf{y}_{1:t})$. We focus on the model in Example 3.

Hence we diagnose the methodology in the two aspects. Firstly, we measure the discrepancy between the exact and the approximated model by root mean square errors (RMSE). We hope the value is negligible. Secondly, we evaluate the overall performance of our methodology by looking at the importance weights, which is the ratio of the marginal likelihoods based on the true model and the approximated model.

### 5.6.1 Accuracy

We want to measure the difference between the exact model and our approximated model by calculating the respective posterior distributions. However, it is impossible to obtain the distribution of the exact multiple change point model within the particle filter framwork, due to the exponentially large number of combination of changepoints and segment models. Thus we just consider a single change point model with $n$ observations, where the exact algorithm is to calculate the posterior distribution simply by Bayes theorem, i.e

$$\begin{aligned} p(C|\mathbf{y}_{1:n}) &\propto p(\mathbf{y}_{1:n}|C=j)p(C=j) \\ &\propto p(\mathbf{y}_{1:n}|C=j) \quad \text{since } p(C=j)=\frac{1}{n} \\ &= \sum_{M} p(\mathbf{y}_{1:n}|C=j,M)p(M), \end{aligned} \tag{5.19}$$

$M$ is the indicator variable which specifies the type of change point. For the whole batch of signals, a linear regression model like (5.3) also exists. In the case of the change point is continuous type ($M=0$), $\mathbf{H}$ is a $n$ by 5 matrix like

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_1 & 0 \\ \mathbf{H}_2 & \mathbf{H}_3 \end{pmatrix}, \tag{5.20}$$

where $\mathbf{H}_1, \mathbf{H}_2$ and $\mathbf{H}_3$ are of forms

$$\mathbf{H_1} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & x_2 - x_1 & (x_2 - x_1)^2 \\ \vdots & \vdots & \vdots \\ 1 & x_\tau - x_1 & (x_\tau - x_1)^2 \end{pmatrix},$$

$$\mathbf{H_2} = \begin{pmatrix} 1 & x_{\tau+1} - x_1 & (x_{\tau+1} - x_1)^2 \\ \vdots & \vdots & \vdots \\ 1 & x_{\tau+1} - x_1 & (x_{\tau+1} - x_1)^2 \end{pmatrix},$$

and

$$\mathbf{H_3} = \begin{pmatrix} 0 & 0 \\ x_{\tau+2} - x_{\tau+1} & (x_{\tau+2} - x_{\tau+1})^2 \\ \vdots & \vdots \\ x_n - x_{\tau+1} & (x_n - x_{\tau+1})^2 \end{pmatrix}.$$

In the case of that the change point is discontinuous type (i.e. $M = 1$), $\mathbf{H}$ is a $n$ by 6 matrix of the form

$$\mathbf{H} = \begin{pmatrix} \mathbf{H}_1 & 0 \\ 0 & \mathbf{H}_2 \end{pmatrix},$$

where $\mathbf{H}_1$ and $\mathbf{H_2}$ are of forms

$$\mathbf{H}_1 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & x_2 - x_1 & (x_2 - x_1)^2 \\ \vdots & \vdots & \vdots \\ 1 & x_\tau - x_1 & (x_\tau - x_1)^2 \end{pmatrix},$$

$$\mathbf{H_2} = \begin{pmatrix} 1 & 0 & 0 \\ 1 & x_{\tau+2} - x_{\tau+1} & (x_{\tau+2} - x_{\tau+1})^2 \\ \vdots & \vdots & \vdots \\ 1 & x_n - x_{\tau+1} & (x_n - x_{\tau+1})^2 \end{pmatrix}.$$

Since there is only one change point, the weight defined by (5.11) can be calculated

exactly by (4.4) with $\mathbf{H}$ defined here.

We can adapt the output of our methodology to the case of a single changepoint by calculating $p(C_n = s|C_s = 0, \mathbf{y}_{1:n})$ instead of $p(C_n|\mathbf{y}_{1:n})$. This conditional probability is

$$p(C_n = s|C_s = 0, \mathbf{y}_{1:n}) \propto p(C_n = s|\mathbf{y}_{1:n})p(C_s = 0|\mathbf{y}_{1:n}),$$

where both terms on the right hand side are calculated by our filtering algorithm as $p(C_s = 0|\mathbf{y}_{1:n}) = p(C_s = 0|\mathbf{y}_{1:s})$ under our geometric distribution.

We simulate 200 signals from the Heavisine function (see Donoho and Johnstone (1994) and the Appendix C) plus normal noises. We split the data into three parts so that each part has a single change point. The lengths of the data set are 100 (top),70 (middle) and 100 (bottom) respectively. There are overlaps for these data sets. Figure 5.5 demonstrates the re-constructed curves from the 3 different data sets (right panels) as well as the difference between them (left panels). To give a comprehensive comparison, we also put the true curve and the differences between the true and fitted curves in this figure.

Figure 5.4 gives the marginal posterior distribution of change points (left panels) and its joint distribution of being discontinuous(right panels),respectively. Each plot has two outputs produced by the two algorithms. Corresponding to Figure 5.5, the two plots at the bottom give almost identical distributions which due to the positions and types of changepoints being obvious from the data. However, the results generated on the basis of the first two data sets are worth more attention. The two distributions in each plot are at first sight quite different. But the results are still satisfactory. Firstly, they have shown where the true change points are with relative probabilities. Secondly, they have given the types of change points correctly with very small probabilities of being continuous change points. Thirdly, the rough positions and the types of change points found by both exact

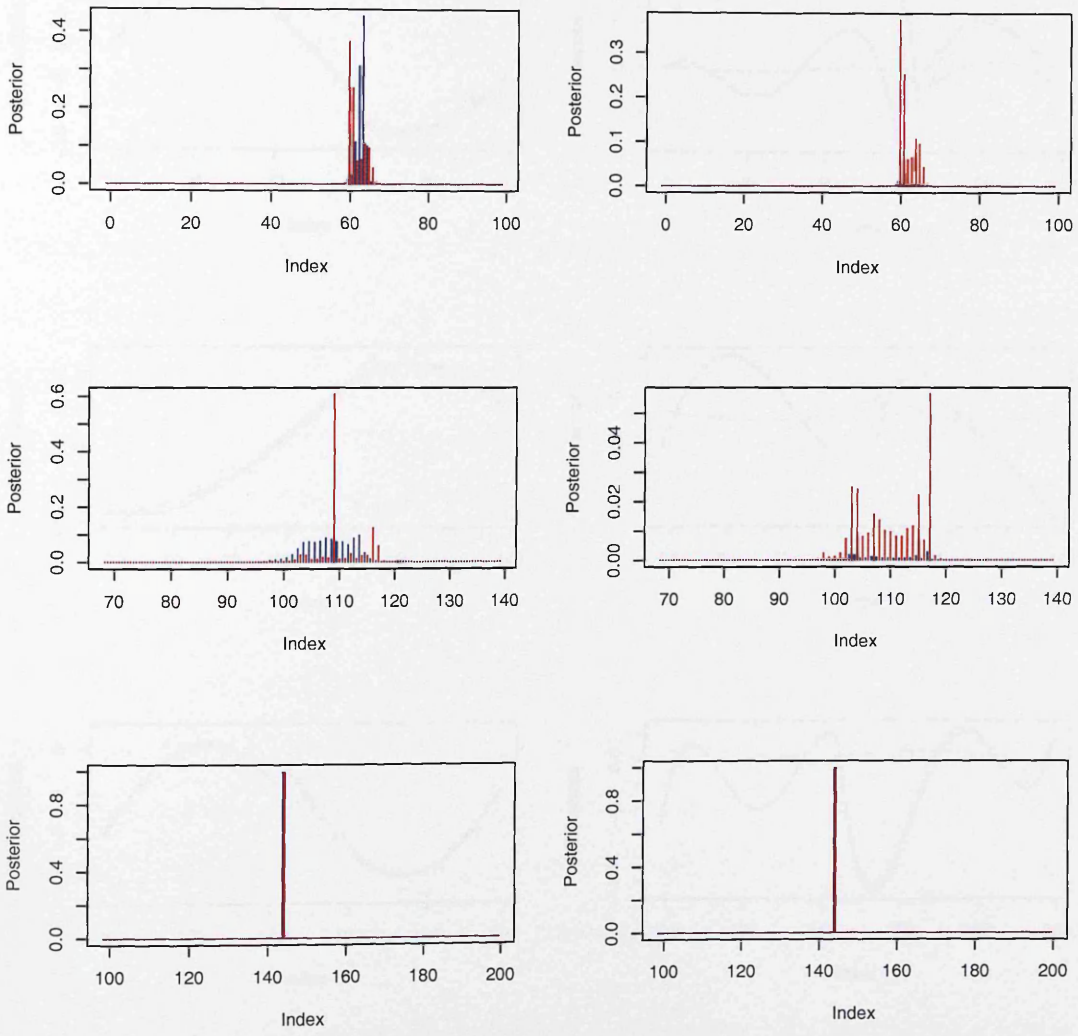


Figure 5.4: The plots on the left show the two different marginal posterior distribution of the position of change point,i.e. $p(C|\mathbf{y}_{1:n})$ (The approximated distribution is actually $p(C_n|C_s = 0, \mathbf{y}_{1:n})$). The plots on the right show the two different joint posterior distribution, i.e. $p(C, M|\mathbf{y}_{1:n})$. (Red line: Approximated distribution; Blue line: Exact distribution)

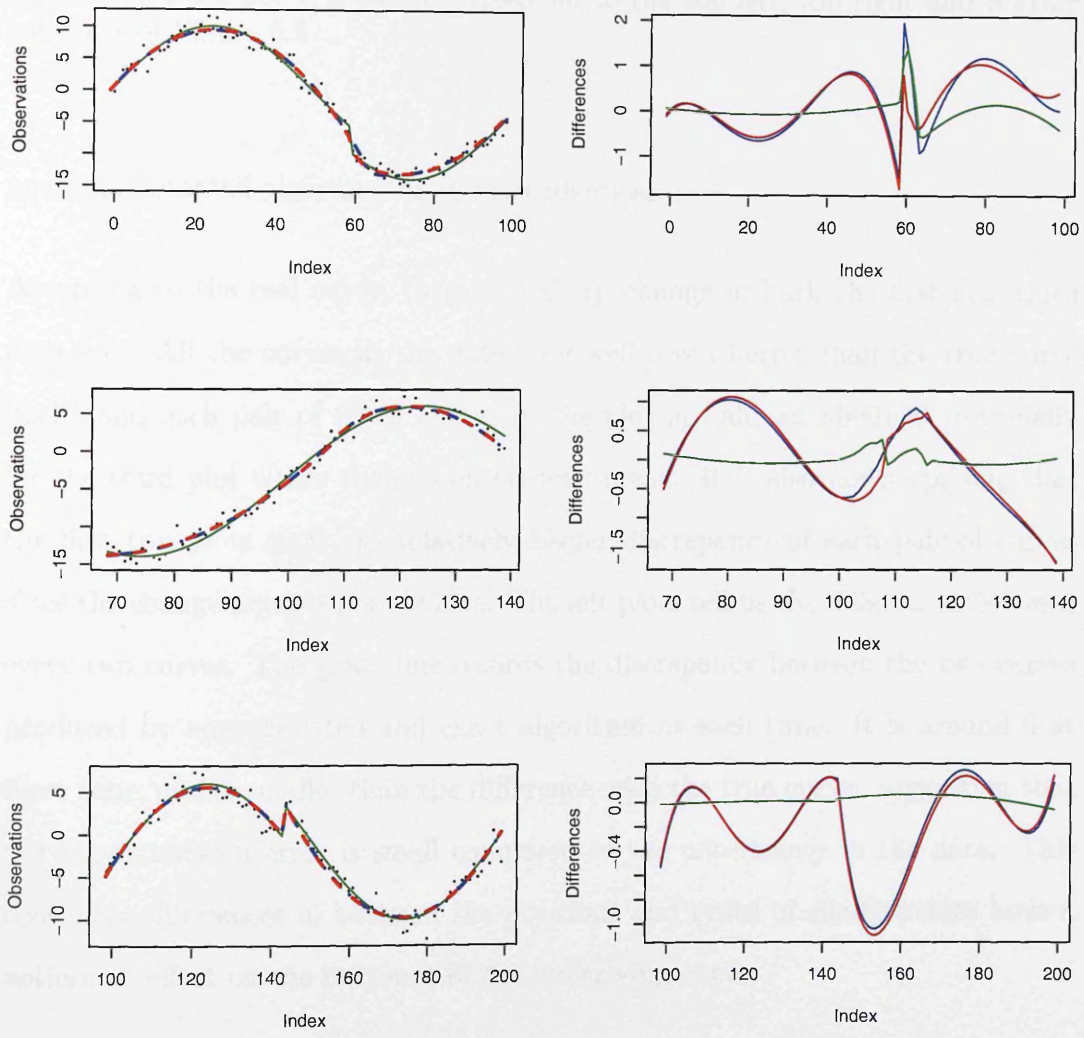


Figure 5.5: Left panels: The three lines in each plot are true curve (green solid line), fitted curves by approximated algorithm (red dash line) and by exact algorithm (blue dash line), respectively. All the curves are produced by averaging across 100 independent realisations. Right panels: The three lines in each plot are differences between every two curves over time. The red line is that between approximately fitted and true curves. The blue line is that between exactly fitted and true curves. The green line is that between approximately fitted and exactly fitted curves.

| RMSE | Data Set 1 | Data Set 2 | Data Set 3 |
|---|---|---|---|
| True & Appr | 0.5841401 | 0.7313837 | 0.4033361 |
| True & Exact | 0.6741613 | 0.7322288 | 0.4074075 |
| Appr & Exact | 0.2790919 | 0.0878510 | 0.0313473 |

Table 5.1: The Root Mean Square Error between every two lines out of the true line, the line produced by approximate algorithm and the line produced by exact algorithm. Data Set 1, 2 and 3 correspond to the top left, top right and bottom left plots of Figure 5.5

and approximated algorithm are almost identical.

According to the real curve, there is a sharp change in both the first and third data sets. All the curves fit the data very well (even better than the true curve itself) and each pair of fitted curves in the plot are almost identical, especially for the third plot where there is an obvious break. It is also not surprising that the first two plots show the relatively bigger discrepancy of each pair of curves since the change point is not obvious. The left plots tell us the differences between every two curves. The green line records the discrepancy between the two curves produced by approximated and exact algorithm at each time. It is around 0 at most time, and is smaller than the difference with the true curve: suggesting that the approximation error is small compared to the uncertainty in the data. This shows the differences in between the positions and types of changepoints have a noticeable effect on the influence of the underlying curve.

We can quantify the difference between the performances of algorithms by calculating the root mean square errors (RMSE) between curves. The RMSEs between the exact line and the true line, according to Table 5.1, are almost the same as those between the true line and approximated line. Furthermore, there is no evidence that the exact algorithm does better at fitting the curves: in fact RMSE is lower for the approximate algorithm in all cases.

### 5.6.2 Importance weights

We measure the discrepancy between the approximated and the true model with respect to the importance weights, to see the effect brought from the approximation on the whole data set, not only limited to the single changepoint model.

The importance weight is a ratio between the true joint probability and the approximated one, which is of form:

$$w_{import} = \frac{p^{true}(\mathbf{C}, \mathbf{M}|\mathbf{y}_{1:n})}{p^{appr}(\mathbf{C}, \mathbf{M}|\mathbf{y}_{1:n})}, \tag{5.21}$$

where $\mathbf{C}$ denotes all the changepoints positions and $\mathbf{M}$ denotes the corresponding model types. The idea is that we can simulate sets of $(\mathbf{C}, \mathbf{M})$ (which is a natural output of our filtering and smoothing algorithm) from the $p^{appr}(\mathbf{C}, \mathbf{M}|\mathbf{y}_{1:n})$ and evaluate this proposal probability directly. Further, we can calculate the true joint probability for a set of realisation of $(\mathbf{C}, \mathbf{M})$. Thus $p^{true}(\mathbf{C}, \mathbf{M}|\mathbf{y}_{1:n})$ can be calculated in a batch way as:

$$p^{true}(\mathbf{C}, \mathbf{M}|\mathbf{y}_{1:n}) \propto p(\mathbf{y}_{1:n}|\mathbf{C}, \mathbf{M})p(\mathbf{C})p(\mathbf{M}),$$

where $p(\mathbf{y}_{1:n}|\mathbf{C}, \mathbf{M})$ is tractable as we have a linear model (5.5) and conjugate priors.

Obviously, the calculation is involved with a huge matrix if the data is quite large, and this is the reason we can only check the importance weight on a small amount of data, 500 say. Evidence that $p^{appr}(\mathbf{C}, \mathbf{M}|\mathbf{y}_{1:n}) \approx p^{true}(\mathbf{C}, \mathbf{M}|\mathbf{y}_{1:n})$ would be that the result importance weight will be approximately constant. This would further suggest that if the ratio between the two models is required then our algorithm would give a good approximate distribution.

Figure 5.6 and 5.7 show importance weights for 1000 simulation of (5.21) under for

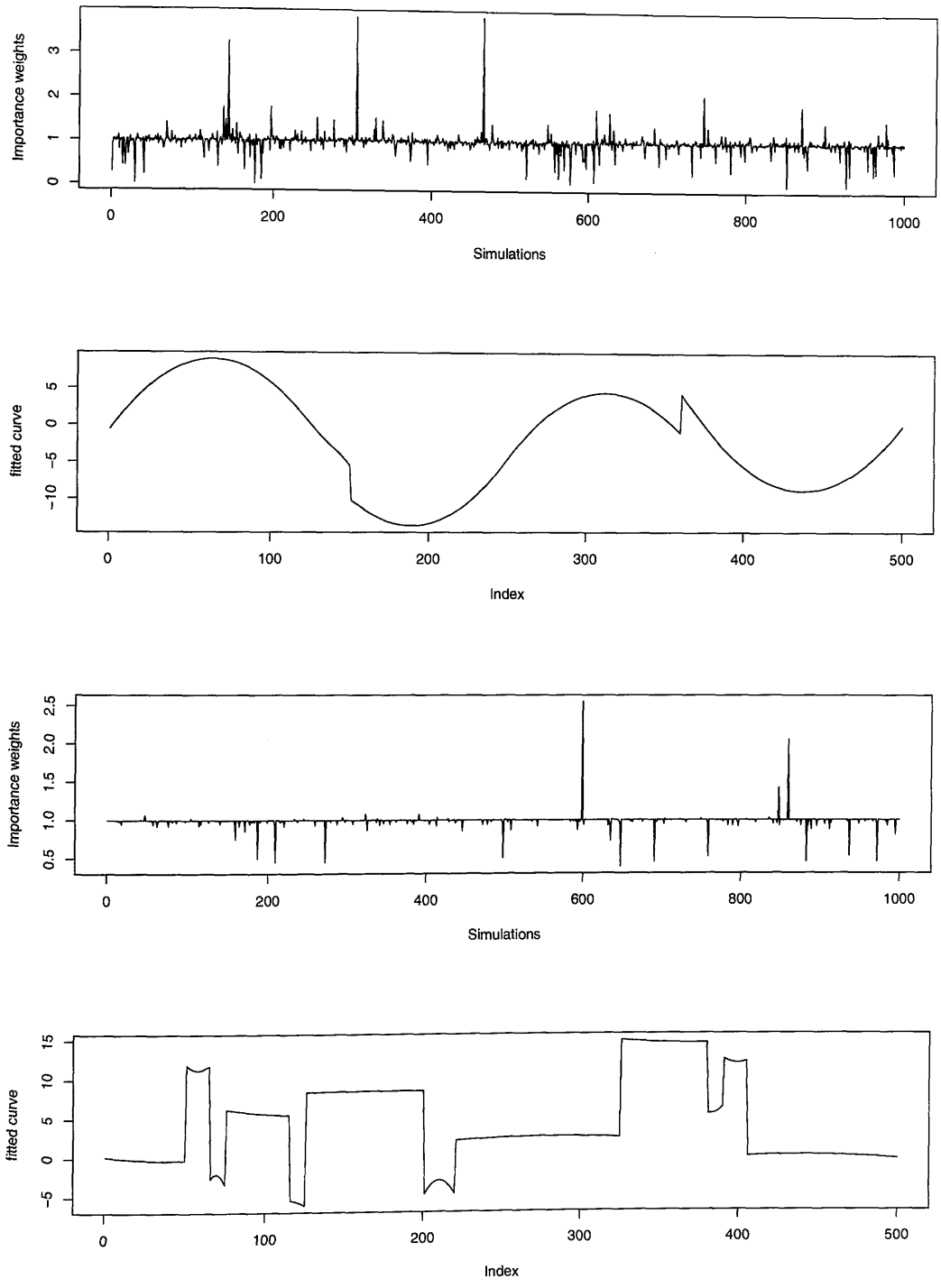


Figure 5.6: The importance weights from 1000 replicates of simulations and the fitted curves, each of which is an average of 1000 realisations, chosen from the simulation results by the importance weights we calculated. Upper: The importance weights and fitted curve of the Heavisine data. Bottom: The importance weights and fitted curve of the Blocks data

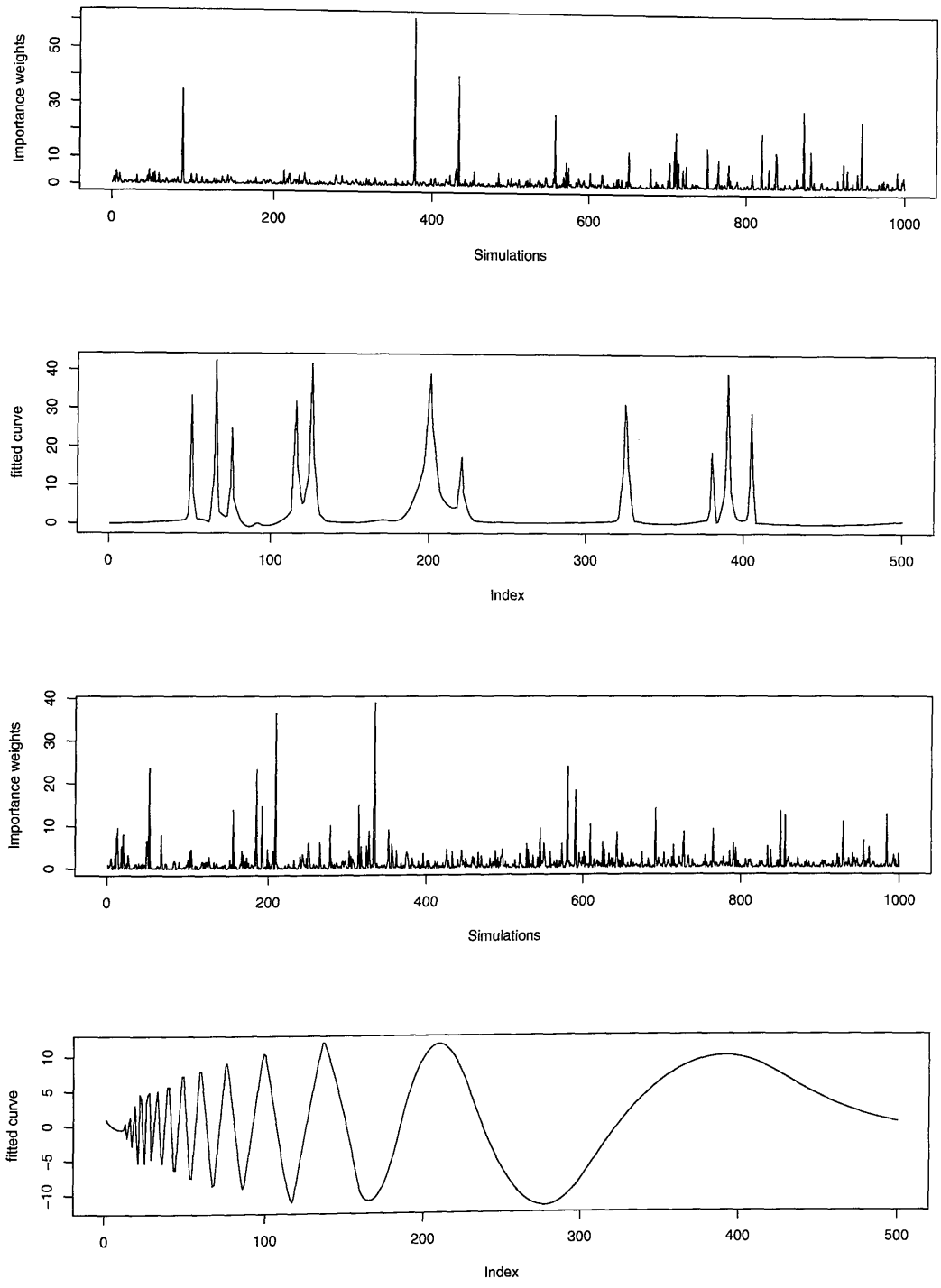


Figure 5.7: The importance weights from 1000 replicates of simulations and the fitted curves, each of which is an average of 1000 realisations, chosen from the simulation results by the importance weights we calculated. Upper right: The importance weights and fitted curve of the Bumps data. Bottom: The importance weights and fitted curve of the Doppler data.

| Example | Heavisine | Blocks | Bumps | Doppler |
|---|---|---|---|---|
| ESS | 633 | 621 | 387 | 372 |

Table 5.2: The effective sample size (ESS) of each data set based on the 1000 simulation in Figure 5.6 and 5.7.

each of the distribution as well as an average of the fitted curves that are selected from the 1000 realisations according to the importance weights. Note that the performance varies with data sets. This can be easily seen from the ESS of the importance sampling given in Table 5.2. With the data sets which have more changepoints, and more uncertainty about changepoints (e.g. Bumps and Doppler data), the importance sampling has a smaller ESS value ( which are 387 and 372, respectively). Note that the ESS used on those data sets for 1000 realisations is still impressive for what is a very high dimension problem.

## 5.7 Simulation studies

We tested our filtering and smoothing algorithm, with our model of Example 3, on a variety of simulated data sets. These data sets include both smooth and unsmooth functions, by which the key feature of our algorithm will be fully explored. Most of the focus here is on the Bayesian analysis with the model of Example 3.

For simplicity, we specify the same hyper-parameters for all the data we will test, although more sensible choices should be the empirical estimations. These hyper parameters are listed below:

(i) The segments follow a geometric distribution with parameter $p = 0.004$;

(ii) The segment model has a Bernoulli distribution with parameter equal to 0.5;

(iii) The $\sigma^2$ follows an inverse Gamma prior with $\nu = 10^{-3}$ and $\gamma = 10^{-3}$;

(iv) Given the $\sigma^2$, the regression parameter $\beta_k$ follow a multivariate normal prior with mean $\vec{0}$ and $\mathbf{D}_k = diag(10^4, 10^6, 10^8)$.

Note that the types of changepoints are influenced by the prior distribution of $\beta_{k0}$s. So we can not use improper priors for $\beta_{k0}$.

With these prior settings, we simulate for each data set 1000 replicates of the changepoints and segment models, and hence the regression parameters, based on which, a curve can be fitted. The final curve is an average of the 1000 simulations. All the results are obtained in a few seconds on a desktop PC with an AMD XP 1700Hz CPU. This is much faster than some iterative methodology such as MCMC. The efficiency comes from two aspects: (i) the SRC resampling scheme and; (ii) the sequential computing process for the weights $w_t^{(s,\mathcal{M})}$ (see Appendix B for the calculation).

To monitor our results, we look at the mean square errors (MSE) of the models generated by our algorithm, given by

$$\text{MSE} = \frac{1}{n}\sum_{t=1}^{n}(y_t - f(x_t))^2, \tag{5.22}$$

where $f(\cdot)$ is a piecewise quadratic regression model. With these values, we undertake comparisons of our methodology (denoted as CPF) with some others, the full list being:

A1 The partial Bayesian approach of Denison et al. (1998) (DMS) implemented with a hybrid sampler. By partial, we mean that changepoints are estimated with a RJMCMC methodology while the regression parameters are estimated by a least square estimation;

A2 The fully Bayesian approach of DiMatteo et al. (2001), which is known as *Bayesian adaptive regression splines* (BARS). They use a particular conjugate prior on the regression parameter called the unit-information prior

(Kass and Wasserman, 1995) so that the RJMCMC can be used to infer the changepoints as well as parameters.

A3 The online algorithm of Chapter 4 (denoted as DPF), in which all changepoints are thought to be discontinuous.

Note that for A1 and A2, we use a Poisson distribution with mean equal to 5 to model the number of changepoints, and use the same prior as ours for $\sigma^2$, that is $\nu = 10^{-3}$ and $\gamma = 10^{-3}$. However, for A3 that is the online algorithm of Chapter 4, we have to use an very informative prior for the $\sigma^2$ to make sure the curve can be fitted.

### 5.7.1 Smooth curves

We begin with two smooth functions which only contain continuous changepoints:

$$\begin{aligned} &(a) \quad h(x) = x + 2\exp(-16x^2) && x \in [-2, 2], \\ &(b) \quad h(x) = \sin(2x) + 2\exp(-16x^2) && x \in [-2, 2]. \end{aligned}$$

The data has been analysed by Denison et al. (1998), and to ease the comparison with the their results , we only simulate 200 data for each function in $[0, 1]$, as they did. We take the values $x_t$ on a uniform grid with $n = 200$ points, i.e. $x_t = (t-1)/(n-1)$. However, $x_t$ can be also randomly located on the interval $[0, 1]$. The larger data set can be used without any difficulty though. The zero-mean distributed noises $\varepsilon$ have variances equal to $\sigma^2 = 0.4^2$ and $\sigma^2 = 0.3^2$ respectively so that the signal to noise ratio is 3.

Although the tested curves are continuous everywhere, we allow for a model choice in our algorithm. The prior probability of being each type of changepoints is 0.5. The algorithm will choose a proper type according to the data itself. Figure 5.8 shows the posterior distribution of both the types and the positions of the

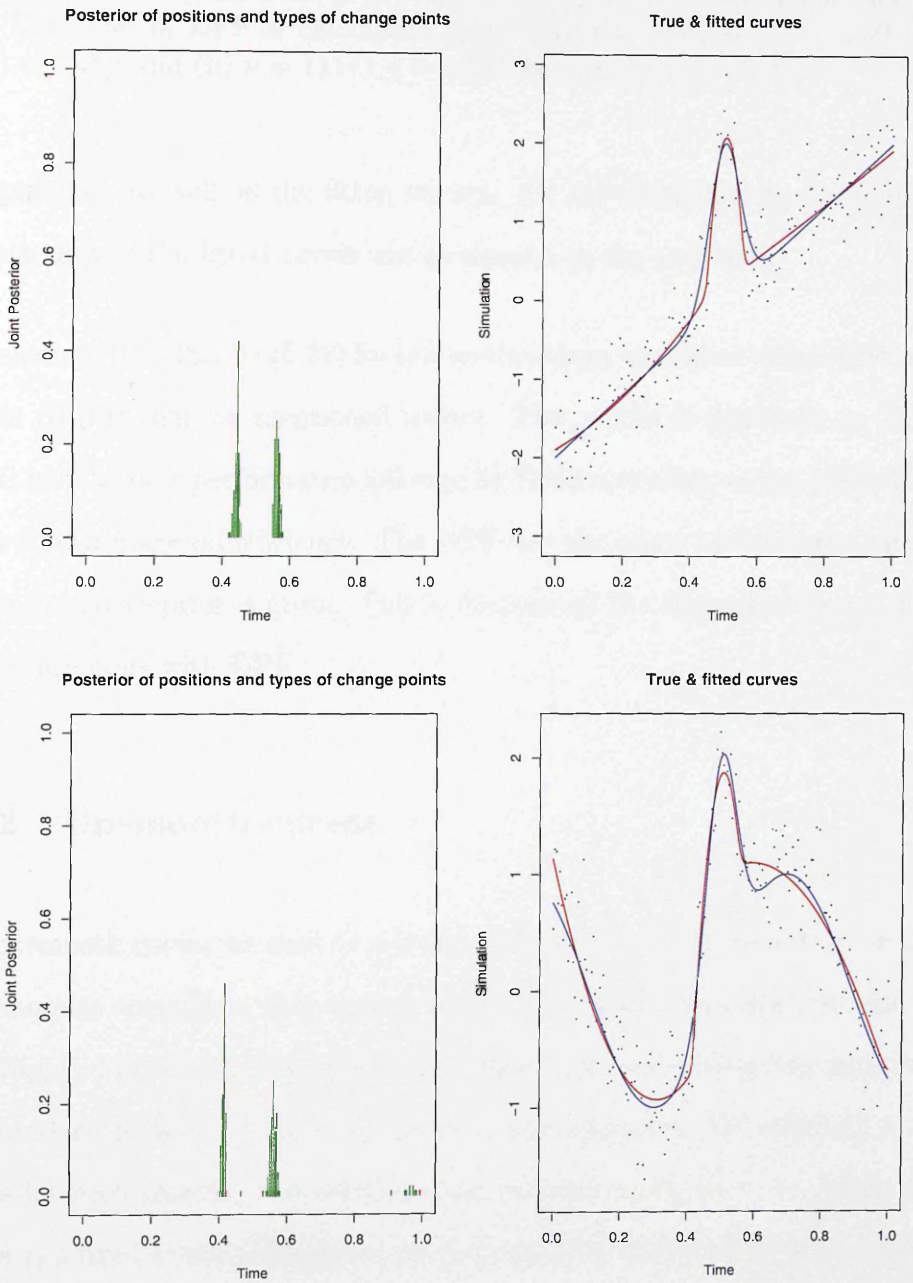


Figure 5.8: Left plots: The marginal posterior distribution of positions and types of changepoints $p(C_t, M_t|\mathbf{y}_{1:t})$, red lines represent the discontinuous changepoints and green lines represent the continuous changepoints. Right plots: The true curves (blue lines) and the fitted curves (red lines)

| Example | DMS | BARS | DPF | CPF |
|---|---|---|---|---|
| (a) | 0.0097 | 0.0071 | 0.0150 | 0.0121 |
| (b) | 0.0087 | 0.0039 | 0.0131 | 0.0090 |

Table 5.3: Mean square error of each methodology on the smooth curves in Figure 5.8. The MSE of DPF is calculated based on (a)$\nu = 6250$, $\gamma = 1000$ (so that $E(\sigma^2) = 0.4^2$); and (b) $\nu = 11111$,$\gamma = 1000$ (so that $E(\sigma^2) = 0.3^2$)

changepoints, as well as the fitted curves. All the changepoints are found to be continuous and the fitted curves are as smooth as the true ones.

We calculate the MSE by (5.22) for our methodology as well as three other methodologies (i)-(iii) that we mentioned before. The results is displayed in Table 5.3. BARS has the best performance followed by DMS according to the MSE. However, we have advantage on efficiency. The DPF has the worst performance although a very informative prior is given. This is because all the changepoints are modelled as discontinuous with DPF.

### 5.7.2 Unsmooth curves

The unsmooth curves we used to test the performance of our algorithm on the wiggly examples come from four typical simulated signals (*Heavisine*, *Blocks*, *Bumps* and *Doppler*) in Donoho and Johnstone (1994). The underlying functions (without the standard noise $\varepsilon$) of them are given in the appendix. We simulate $n = 2048$ points for each sample, and set the noise variance to be $\sigma^2 = 1$. Again we take values $x_t$ a fixed uniform design on the interval [0,1] such that $x_t = (t-1)/(n-1)$.

We first focus on the Heavisine data set only because of its typicality. The performances of our algorithm on all the other data sets will be presented afterward.

According to Donoho and Johnstone (1994), the Heavisine plus white noise is of form consists of three Sine curves with different intercepts. The three curves are connected at two real change points, 0.3 and 0.72 in the interval (0,1). But there

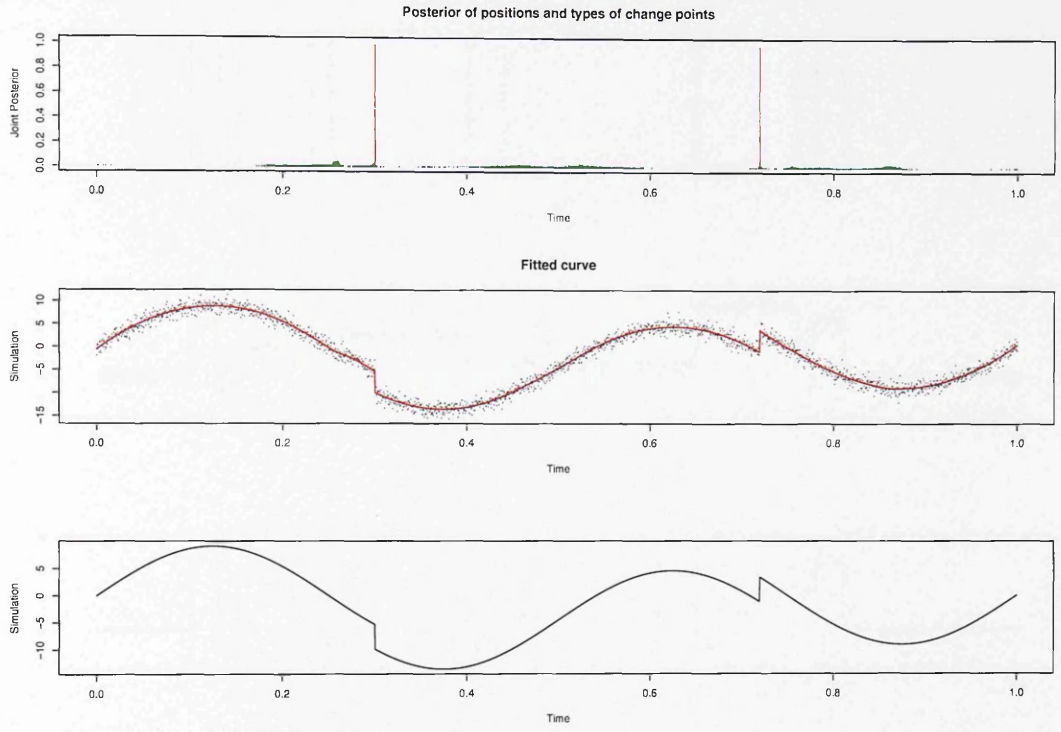


Figure 5.9: Heavisine curve. Upper: The marginal posterior distribution of positions and types of changepoints, $p(C_t, M_t|\mathbf{y}_{1:n})$. Red lines represent the discontinuous changepoints while green lines represent continuous ones. Middle: the simulated data and the fitted curve. Bottom: the true curve.

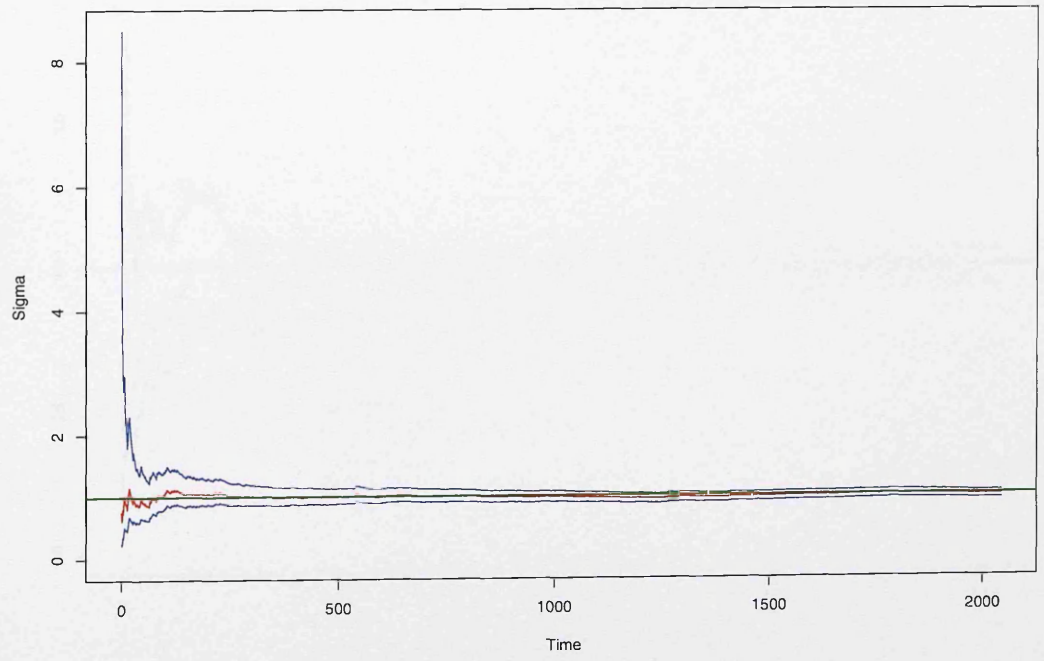


Figure 5.10: The variance of noise, $\sigma^2$ (red line), and its 95% confidence interval (blue lines) over time.

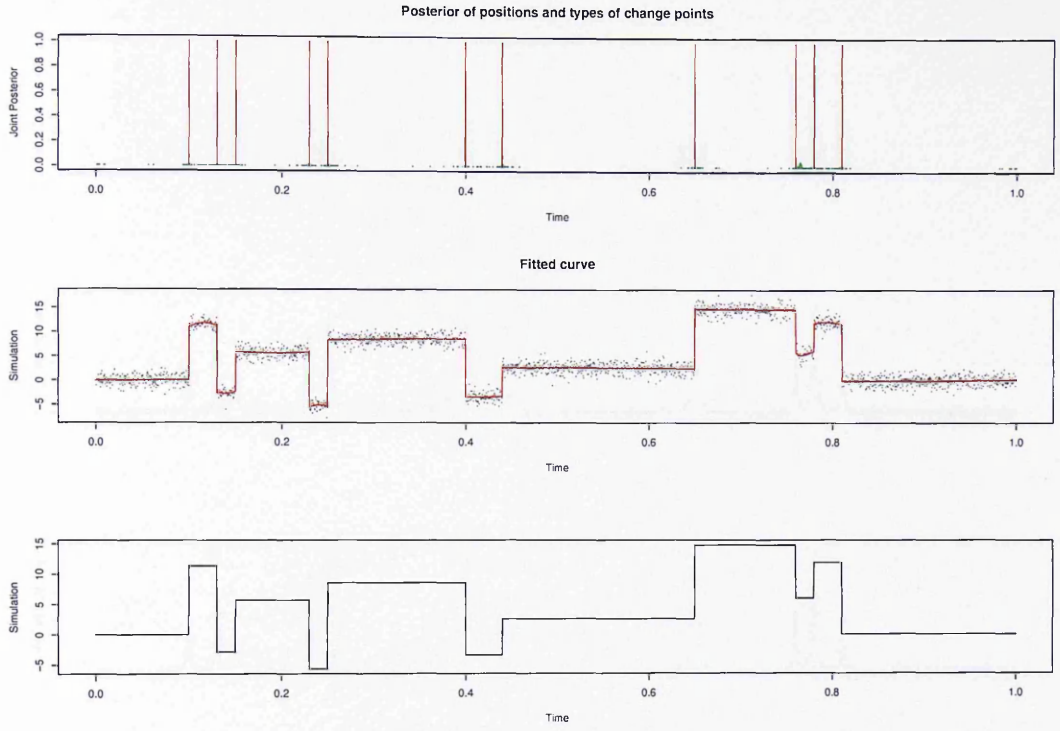


Figure 5.11: Blocks curve: Upper: The marginal posterior distribution of positions and types of changepoints, $p(C_t, M_t|\mathbf{y}_{1:n})$. Red lines represent the discontinuous changepoints while green lines represent continuous ones. Middle: the simulated data and the fitted curve. The bottom plot is the true curve.

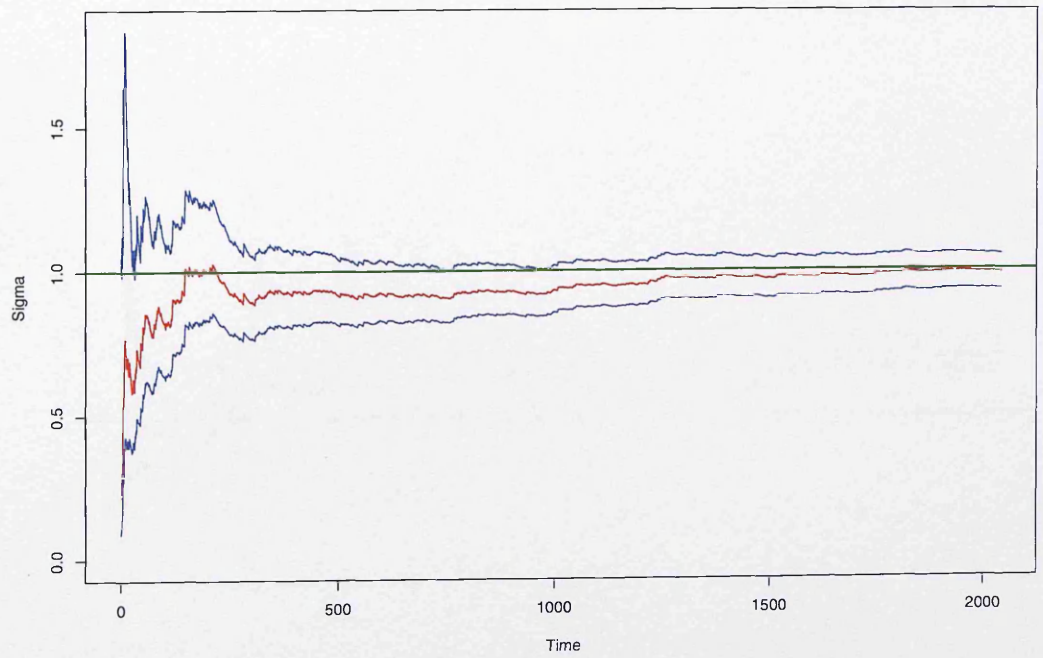


Figure 5.12: The variance of noise, $\sigma^2$ (red line), and its 95% confidence interval (blue lines) over time.

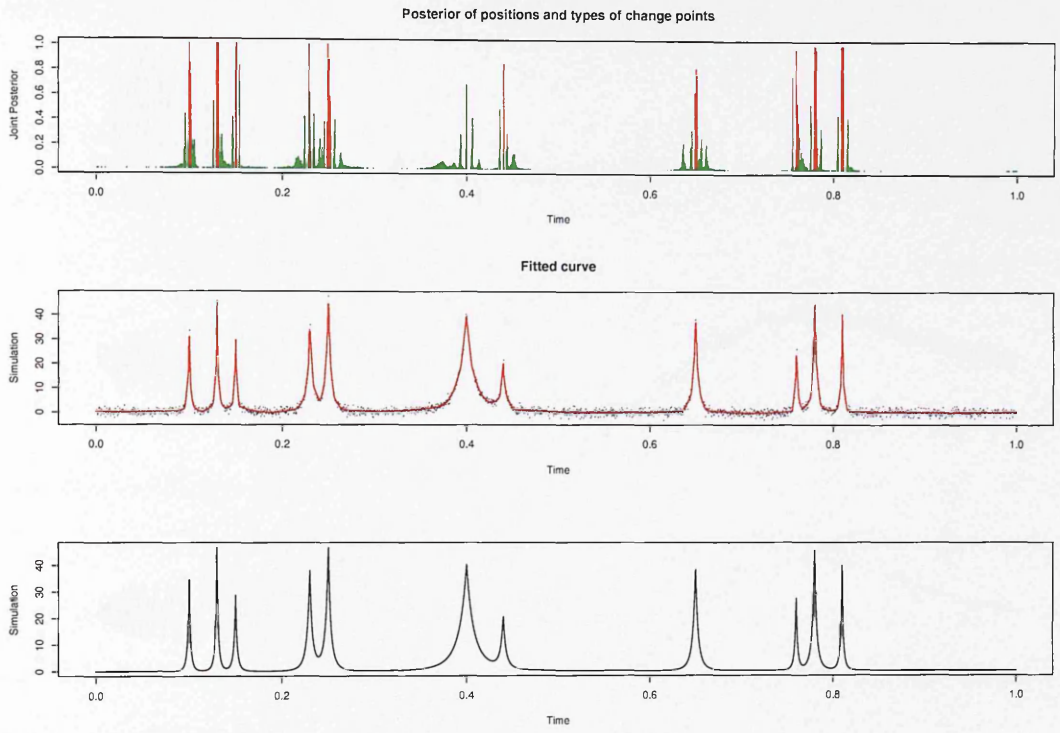


Figure 5.13: Bumps curve: Upper: The marginal posterior distribution of positions and types of changepoints, $p(C_t, M_t|\mathbf{y}_{1:n})$. Red lines represent the discontinuous changepoints while green lines represent continuous ones. Middle: the simulated data and the fitted curve. Bottom: the true curve.

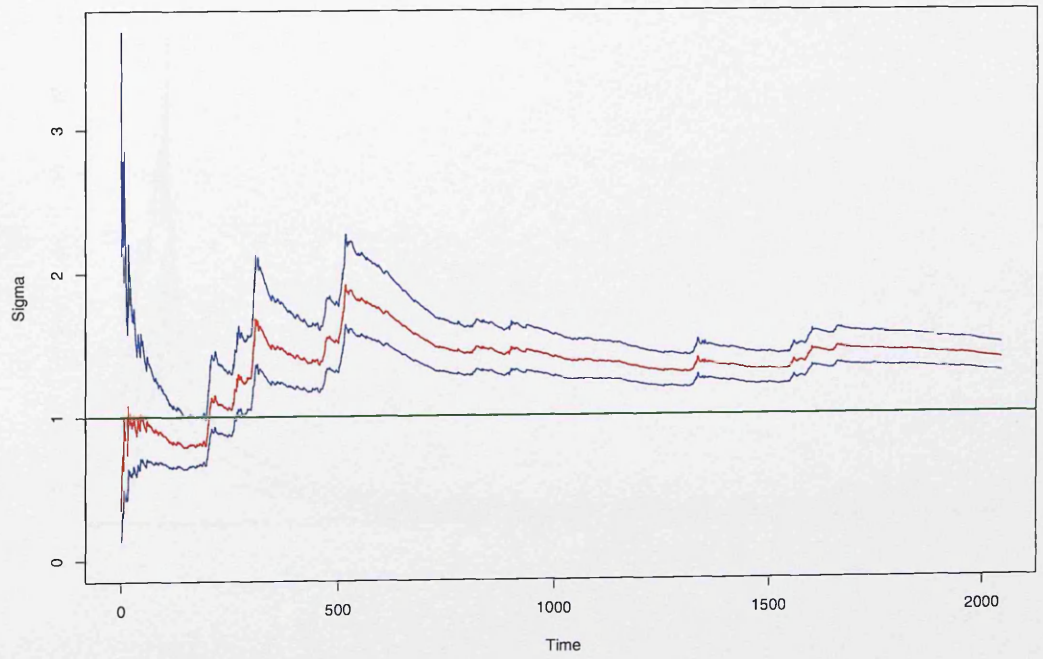


Figure 5.14: The variance of noise, $\sigma^2$ (red line) , and its 95% confidence interval (blue lines) over time.

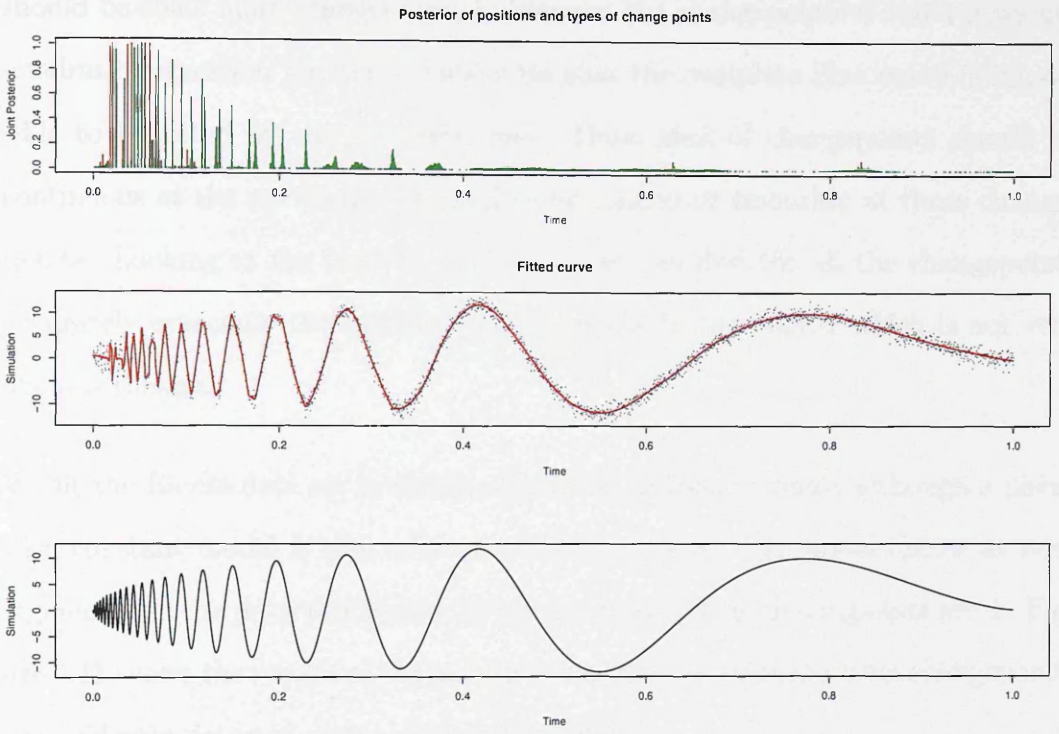


Figure 5.15: Doppler curve: Upper: The marginal posterior distribution of positions and types of changepoints, $p(C_t, M_t|\mathbf{y}_{1:n})$. Red lines represent the discontinuous changepoints while green lines represent continuous ones. Middle: the simulated data and the fitted curve. Bottom: the true curve.

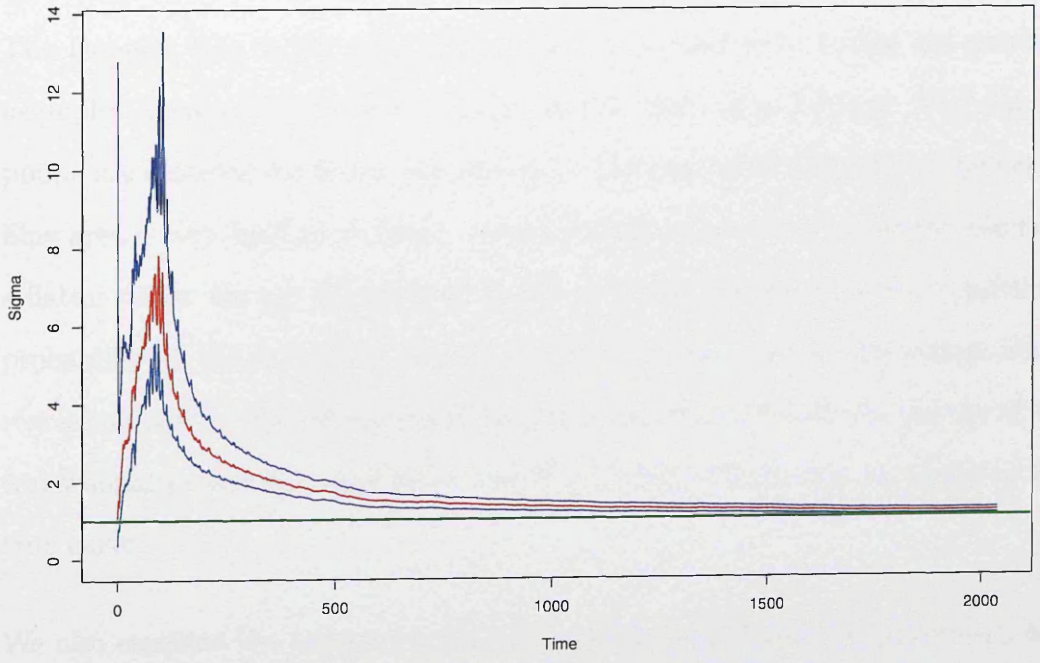


Figure 5.16: The variance of noise, $\sigma^2$ (red line), and its 95% confidence interval (blue lines) over time.

should be some more changepoints in between the changepoints 0 and 1 if we use quadratic regression function. This is because the complete Sine curve is impossible to be fitted by one quadratic line. These kind of changepoints should be continuous as the curves are required to be connected smoothly at those changepoints. Looking at the plots in Figure 5.9, we can find the all the changepoints accurately especially the continuous changepoint in the middle which is not very obvious though.

We fit the Blocks data set by using a piecewise quadratic model although a piecewise constant model is also available in which there is no model choice at each segment, i.e. the prior probability of being discontinuous changepoints are 1. Figure 5.11 shows the results of the quadratic model. The discontinuous changepoints are evidently detected with a probability being 1.

The Bumps data set in Figure 5.13 is a bit harder, as they are involved with much more changepoints. However, the performance of our algorithm is still surprisingly good.

The Doppler data is the most difficult one to be analysed. Unlike the previous examples, there are no evident changepoints in terms of its formula. The changepoints are detected for fitting purpose only. The data points located in the beginning area is very hard to be fitted. An informative prior for $\sigma^2$ generally results in a flatter curve. On top of changing this prior, we can also increase the transitional probability at the expense of increasing computational cost for the estimation of regression parameters. Many more changepoints are found with the change of the transitional probability, and hence the fitted curve will become as wiggly as the true curve.

We also examine the variance of the noise, $\sigma^2$, by calculating the mean and 95% confidence interval of $\sigma^2$. Figures 5.10-5.16 present the evolving process of $\sigma_t^2$ for $t = 1, \ldots, 2048$. All but the Bumps data set have a good performance that the

$\sigma_t^2$ converges to the true value of $\sigma^2$, which is 1. The Bumps data returns a value approximately equal to 1.5, because it does fit the peaks in the data set well.

The MSE of these four results are also listed in Table 5.4, together with other three methodologies. In general, the BARS outperforms the other three methodologies on relatively smooth function (e.g. the Heavisine and the Doppler) in terms of MSE. However, BARS is the most time-consuming methodology followed by the DMS, and BARS is not good at analysing the data with too many discontinuous changepoints (e.g. the Blocks and the Bumps). DPF and CPF are almost the same on both the efficiency and accuracy. The advantage of the CPF is that no informative prior is needed.

For the Heavisine and Blocks data sets, the CPF has a big improvement on MSE, compared to the DMS. However, for the Bumps and the Doppler data sets, the continuous function assumed by DMS model is a better choice. This might because that (i) in Bumps example, the estimation $\sigma^2$ is incorrect; and (ii) in Doppler example, the transitional probability is not properly chosen. Without these two problems, the result of our methodology is surprisingly better, as that can be seen from the Heavisine and Blocks example.

Further we check the performance of our algorithm on Example 1 by choosing the prior probability of being discontinuous changepoints as 1. The MSEs calculated on these four data sets are 0.025, 0.018, 0.305 and 0.194 respectively. The performances are only slightly worse than the DPF case except for the Blocks data where the advantage of DPF is its ability of fit a piecewise constant model. Otherwise the results suggest our method does well at inferring $\sigma^2$.

A more detailed comparison will be presented in the next section.

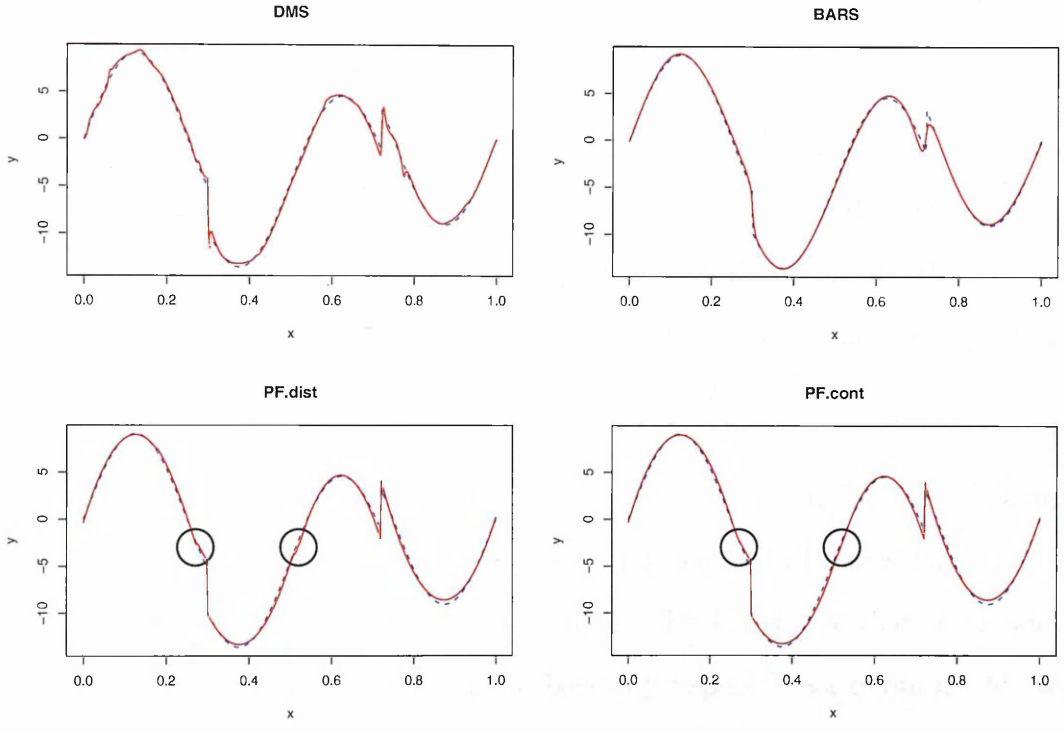


Figure 5.17: We have compared 4 methodologies on the Heavisine data set. Top left: the methodology of Denison et al. (1998); Top right: methodology of DiMatteo et al. (2001); Bottom left: Methodology of Chapter 4; Bottom right: Methodology of our algorithm. The blue dash line is the true curve and the red line is the fitted curve. The circles in the bottom plots indicate the slight difference of those two curves.

| Example | DMS | BARS | DPF | CPF |
|---|---|---|---|---|
| Heavisine | 0.033 | 0.015 | 0.022 | 0.017 |
| Blocks | 0.170 | 0.285 | 0.010 | 0.019 |
| Bumps | 0.167 | 0.251 | 0.294 | 0.286 |
| Doppler | 0.135 | 0.081 | 0.188 | 0.194 |

Table 5.4: Mean square error of each methodology on the unsmooth functions in Figure 5.9–5.15. The MSE of DPF is calculated base on $\nu = 1000$ and $\gamma = 1000$ (so that $\mathrm{E}(\sigma^2) = 1$

### 5.7.3 Further comparison

Examining our algorithm only the MSE is not complete. As the selling-point of our algorithm is its ability to adapt both the sharp jumps and gradual change in the curves, we compare the result of our algorithm with the other algorithm (i)-(iii) on the Heavisine data by plotting these fitted curves. The result of comparison is presented in Figure 5.17.

From these plots, it can be seen very clearly that the methodologies of Denison et al. (1998) and DiMatteo et al. (2001) are unable to deal with the sharp changes. Both two fitted curves are smooth everywhere. This is because that the model of piecewise polynomials they were using is basically a special case of our model, with all changepoints continuous. Another disadvantage of such kind of models is that the dependence across the segments are likely to produce much more changepoints. For example, Denison et al. (1998) found on average 17 changepoints while our methodology only found 7 changepoints on average, which suggests an over-fitting with their methodology.

In contrast, the methodology developed in the previous chapter is unable to keep the smoothness of the curve. As it supposes all the observation across segments are independent of each other, the model they were using is basically another extreme case of our model, i.e. the model with all changepoints discontinuous. This is also the case of Punskaya et al. (2002). Obviously, our model reconciles these two kinds of models by adding a model choice step in each segment.

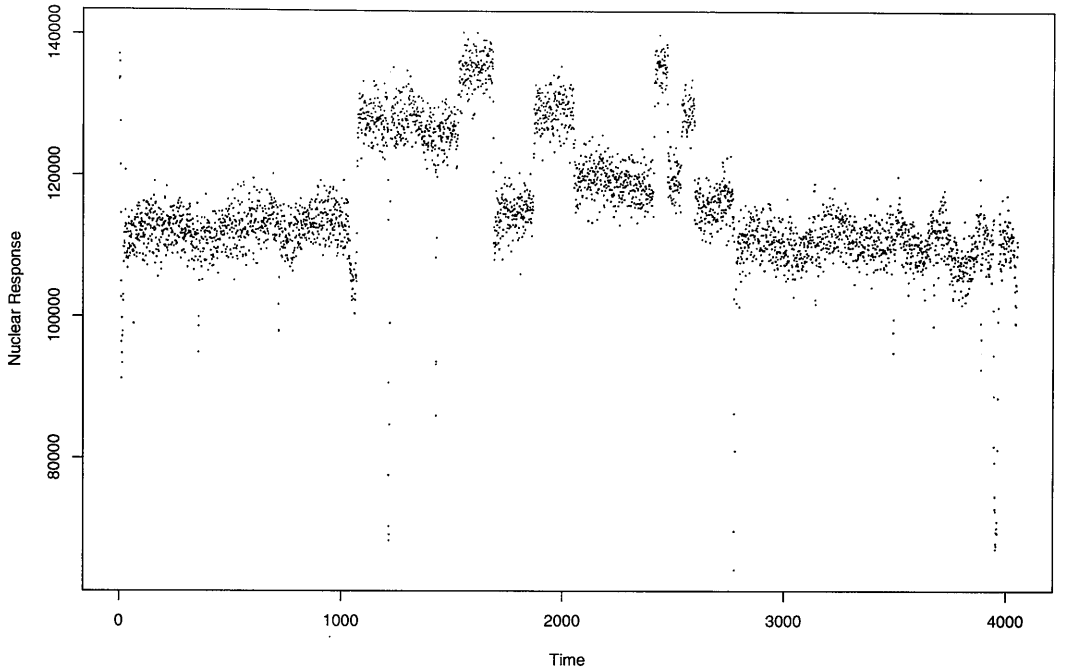


Figure 5.18: Plot of well-log data.

## 5.8 Well-log data

A typical example of changepoints detection problem is the well-log data considered in Chapter 5 of the book by Ó Ruanaidh and Fitzgerald (1996). This is a geophysical data containing measurement of nuclear magnetic response at 4050 time points. The data, which is plotted in the Figure 5.18, is collected from around the drill head when drilling for oil, and contains the information about the rock structure that is being drilled through. In particular it contains the information about the boundaries between rock strata. When drilling for oil, it is important to be able to detect these boundaries, in order to adjust the drilling pressure for the new rock type. These readjustment could prevent blow-outs: sudden uncontrolled flows of drilling fluid, oil or water, up the borehole, which are caused when the pressure exerted by fluids in the rocks exceeds the pressure in the borehole.

### 5.8.1 Historical methodologies

A number of techniques, based on a step function that is used to model the underlying states, have been well established. The changepoints in the step function relate to the rock strata boundaries, and are assumed to follow a Markov process with a transitional probability $p$. Ó Ruanaidh and Fitzgerald (1996) firstly analysed the data by using a Gibbs sampler, based upon the following conditions: (i) the outlier points are removed by hand before the data were analysed; (ii) the number of changepoints is fixed to 13 in the data; and (iii) the distribution of observations is modelled as Laplacian.

Fearnhead (1998) pointed out that a normal distribution is more appropriated to the observations. With such a model, Fearnhead and Clifford (2003) developed an on-line algorithm to the data through the Mixture Kalman filters (see also Section 3.4.1), in which the decision about both the outliers and the number of changepoints can be dealt with automatically. The computational cost of the algorithm is roughly linear to the data. An MCMC version of the on-line algorithm is the Gibbs sampler in Chapter 6 of Cappe et al. (2005).

Starting from a similar model (where there is no function for outliers), the forward-backward algorithm of Fearnhead (2005) can produce an exact inference for the model. The outliers are removed by hand before the data analysis (Ó Ruanaidh and Fitzgerald, 1996). But the single step function model of Fearnhead and Clifford (2003) is inappropriate here as it requires too many changepoints to fit the data.

### 5.8.2 Model and prior

The step function only gives a very rough estimation to the underlying states, and it is not accurate sometimes, as the rock style usually changes in a smooth way. Thus, adjusting the drilling pressure very suddenly at some time is dangerous,

which can easily cause the blow-outs. By contrast, we consider the piecewise quadratic regression model of Example 3 so that a much more detailed underlying curve can be provided: if a discontinuous changepoint is detected, it means a change of the rock structure; if a continuous changepoint is detected, a smooth adjustment of the pressure should be made within the structure.

The major outlier points are removed from the data before they were analysed. Then the rest of 3952 data points are assumed to follow an informative prior with $\mu = 115000$ and $\upsilon = 10000$. For the sake of computational simplicity, we suggest to rescale the data to follow a standard normal distribution such that

$$X^{new} = \frac{X - 115000}{10000}.$$

The time between two changepoints is assumed to follow a geometric distribution with expectation $1/p$, so the $X_t$ follows a Markov process with transitional probability $p$. Visual inspection shows that there are roughly 16 changes in a sequence of about 4000 observations, so the average of the incremental distribution is about 250, suggesting that a suitable value of the transitional probability $P(X_{t+1}|X_t)$ is $p = 1/250$.

Fearnhead and Clifford (2003) suggested $\varepsilon$ follows a $N(0, \sigma^2)$ distribution, where $\sigma$ is set to 0.25 $(= 2500/10000)$.

However, we assign an inverse Gamma distribution to $\sigma^2$ with $\nu = 2$ and $\gamma = 2$; and a multivariate normal distribution to $\beta$ with mean vector equal to zero and variance matrix of form $\sigma^2 \times diag(4^2, 100^2, 1000^2)$, as their prior distributions respectively. The two types of the changepoint have an equal prior probability, i.e. $p(M_1) = p(M_2) = 0.5$.

All priors but that of $\beta_{k0}$ are uninformative, because the $\beta_{k0}$ affects the decision of the types of changepoints. According to the settings in the previous methodologies,

$\beta_{k0} \sim N(0, 0.25^2)$, we choose $\delta_0 = 1/0.25 = 4$.

### 5.8.3 Results

We simulate 1000 realisations of the positions of changepoints and the underlying states for each model. The filtering and smoothing algorithm is realised by a *C++* code with an *R* interface. The block sampling part for the regression parameters, and hence the calculation of the fitted curve are done in a complete *R* environment. It only took 7 minutes for the filtering and smoothing algorithm to run over the data.

The result obtained from a piecewise quadratic model is given in Figure 5.19. The marginal posterior distribution of the positions of the changepoints is calculated empirically, and the underlying signal indicated by the red curve is an average of the 1000 realisations of the fitted curves. The 17 major clusters are evidently segmented by the 16 discontinuous changepoints. The continuous changepoints are scattered in between these discontinuous changepoints, to reflect the gradual change of the rock style. Note that the number of changepionts also depends on the prior of $\beta_{k0}$.

The common parameter $\sigma^2$ is updated by the Kalman filter at each time. The observations are long enough to give a very narrow range of $\sigma$. The posterior distribution is shown in Figure 5.20. The mode of $\sigma$ is 0.232 which is close to the empirical estimation of Fearnhead and Clifford (2003).

## 5.9 Discussions

We have extended the model considered in Chapter 4, allowing for the dependence of the observations across segments. As a specific application , we have considered

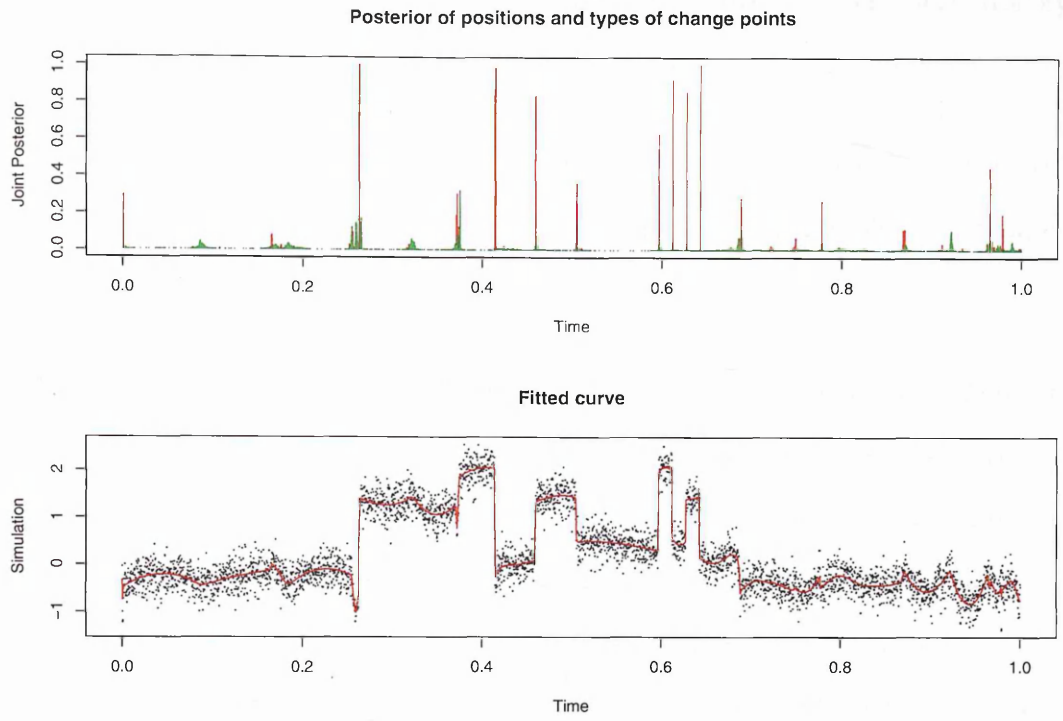


Figure 5.19: Upper plot: The marginal smoothing distribution of positions and types of changepoints $p(C_t, M_t|\mathbf{y}_{1:n})$. Green lines indicate the continuous changepoints and blue lines indicate discontinuous changepoints. Bottom plot: The underlying line for well log data. The data is properly scaled and a piecewise quadratic model is used.

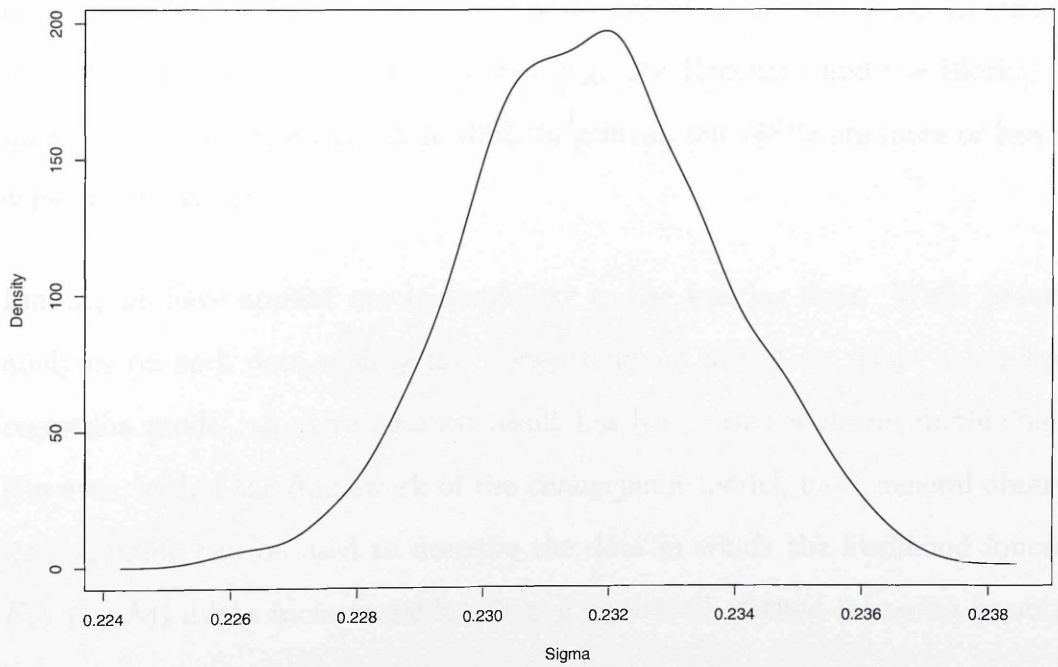


Figure 5.20: The posterior distribution of $\sigma$ in piecewise quadratic model.

a piecewise polynomial model for curve fitting. The novelty of our methodology is its ability to handle both continuous and discontinuous changepoints.

The parameter estimation could be concurrent with the filtering procedure. However, a better estimation based on the block sampling is available at the expense of greater computational cost.

Unlike that in Chapter 4, this changepoint model can be only inferred approximately even without the resampling steps, due to the dependence structure of the observations. Hence we have focused on the error brought by the approximation. Firstly we have difference between the approximated model and true model on a single changepoint case. Then for the multiple changepoints case, we have drawn the plots of importance weights over simulations. The results from this two aspects have shown that our approximation is sensible. We have not checked the effect of the resampling algorithm, as it has been fully explored in Chapter 4. The theoretical result developed there will be similar in the two dimensional case.

We have tested our methodology on both the smooth and unsmooth curves, and compared the results with three other well established methodologies in terms of mean square error. On some data sets (e.g. the Heavisine and the Blocks), we have obtained surprisingly small MSE. In general, our MSEs are more or less the same as the others.

Finally, we have applied our methodology to the well log data. While previous analyses on such data mainly use a step function model we adapt a quadratic regression model. A more detailed result has been given with our methodology. However, within the framework of the changepoint model, more general observation equation can be used to describe the data in which the likelihood function $P(s, t, \zeta, \mathcal{M})$ might be intractable. Thus a numerical method is needed to obtain it.

Compared to the RJMCMC, our methodology does have an advantage on computing time due to the innovative resampling algorithm. All examples mentioned in the chapter can be analysed in a few minutes. Compared to the previous online algorithm, we have a more powerful smoothing procedure, with which we can start from a very diffuse prior while the previous ones need an informative one. However, the performance of our method is partly dependent on the data size. For example, if the data is small, we may not have enough observations to gaurantee the convergence of $\sigma^2$.

## 5.10 Appendix

### A: Reparamterisation when calculating the $P(s,t,\zeta_{s\mathcal{M}})$ in Example 2

In Example 2, the marginal likelihood $P(s,t,\zeta_{s\mathcal{M}})$ can be easily calculated analytically by reparameterising $\beta_k$ as

$$\begin{aligned}\tilde{\beta}_{k0} &= \beta_{k0} - \mu_k, \\ \tilde{\beta}_{kj} &= \beta_{kj}, \qquad \text{for } j = 1, \ldots, q-1.\end{aligned}$$

Thus

$$\tilde{\beta}_k \sim MVN(\vec{0}, \sigma^2 \mathbf{D}_k),$$

where $\mathbf{D}_k = diag(\eta^2, \delta_1^2, \ldots, \delta_{q-1}^2)$. Consequently, we define $\tilde{y}_t$ such that

$$\tilde{\mathbf{y}}_{s+1:t} = \mathbf{y}_{s+1:t} - \vec{\mu}_k = \mathbf{H}_k \tilde{\beta}_k + \varepsilon.$$

With the reparamterisations, the marginal likelihood $P(s,t,\zeta_{a\mathcal{M}})$ can be calculated in the same way as it is calculated in Example 1.

## B: Sequential calculation of $w_{t+1}^{(i)}$

We rewrite $w_{t+1}^{(i)}$ as

$$\begin{aligned} w_{t+1}^{(i)} &= \frac{p(\mathbf{y}_{s+1:t+1}|C_{t+1})}{p(\mathbf{y}_{s+1:t}|C_{t+1})} \\ &= \frac{\int_{\sigma^2}\int_{\beta_k} f(\mathbf{y}_{s+1:t+1}|\beta_k,\sigma^2)\pi(\beta_k)\pi(\sigma^2)\mathrm{d}\beta_k\mathrm{d}\sigma^2}{\int_{\sigma^2}\int_{\beta_k} f(\mathbf{y}_{s+1:t}|\beta_k,\sigma^2)\pi(\beta_k)\pi(\sigma^2)\mathrm{d}\beta_k\mathrm{d}\sigma^2} \\ &= \frac{\int_{\sigma^2}\int_{\beta_k} f(y_{t+1}|\beta_k,\sigma^2)f(\mathbf{y}_{s+1:t}|\beta_k,\sigma^2)\pi(\beta_k)\pi(\sigma^2)\mathrm{d}\beta_k\mathrm{d}\sigma^2}{\int_{\sigma^2}\int_{\beta_k} f(\mathbf{y}_{s+1:t}|\beta_k,\sigma^2)\pi(\beta_k)\pi(\sigma^2)\mathrm{d}\beta_k\mathrm{d}\sigma^2} \\ &= \int_{\sigma^2}\int_{\beta_k} f(y_{t+1}|\beta_k,\sigma^2)\pi(\beta_k|\mathbf{y}_{s+1:t})\pi(\sigma^2|\mathbf{y}_{s+1:t})\mathrm{d}\beta_k\mathrm{d}\sigma^2, \end{aligned}$$

For the sake of simplicity, we ignore the subscript $t$ and $t+1$ here. Integrating out $\beta_k$ first, we have

$$\begin{aligned} &\int_{\beta_k} \frac{1}{\sqrt{2\pi\sigma}} \times \exp\left\{-\frac{(y_{t+1}-\mathbf{H}\beta_k)^2}{2\sigma^2}\right\} \times \\ &\frac{1}{\sqrt{(2\pi\sigma)^p|\Sigma|^{1/2}}} \exp\left\{-\frac{1}{2\sigma^2}(\beta_k-\vec{\mu})^T\Sigma^{-1}(\beta_k-\vec{\mu})\right\}\mathrm{d}\beta_k \\ = &\int_{\beta_k}\left(\frac{1}{\sqrt{2\pi\sigma}}\right)^{p+1}|\Sigma|^{-1/2}\times \\ &\exp\left\{-\frac{1}{2\sigma^2}[(y_{t+1}-\mathbf{H}\beta_k)^T(y_{t+1}-\mathbf{H}\beta_k)+(\beta_k-\vec{\mu})^T\Sigma^{-1}(\beta_k-\vec{\mu})]\right\}\mathrm{d}\beta_k \\ = &\int_{\beta_k}\left(\frac{1}{\sqrt{2\pi\sigma}}\right)^{p+1}|\Sigma|^{-1/2}\times \\ &\exp\left\{-\frac{1}{2\sigma^2}\left[\beta_k^T(\mathbf{H}^T\mathbf{H}+\Sigma^{-1})\beta_k - 2\beta_k^T(\mathbf{H}^T y_{t+1}+\Sigma^{-1}\vec{\mu})+y_{t+1}^2-\vec{\mu}^T\Sigma^{-1}\vec{\mu}\right]\right\}\mathrm{d}\beta_k \end{aligned}$$

Denoting

$$\begin{aligned} \mathbf{M} &= (\mathbf{H}^T\mathbf{H}+\Sigma^{-1})^{-1}, \\ \mathbf{N} &= (\mathbf{H}^T y_{t+1}+\Sigma^{-1}\vec{\mu}), \\ ||y_{t+1}||^2 &= y_{t+1}^2+\vec{\mu}^T\Sigma^{-1}\vec{\mu}-\mathbf{N}^T\mathbf{M}\mathbf{N}, \end{aligned}$$

the integration can be reduced to

$$\begin{aligned}
&\frac{1}{\sqrt{2\pi}\sigma}\frac{|\mathbf{M}|^{1/2}}{|\Sigma|^{1/2}}\exp\left\{-\frac{||y_{t+1}||^2}{2\sigma^2}\right\}\times \\
&\int_{\beta_k}\left(\frac{1}{\sqrt{2\pi}\sigma}\right)^p|\mathbf{M}|^{-1/2}\exp\left\{-\frac{1}{2\sigma^2}[\beta_k-\mathbf{MN}]^T M^{-1}[\beta_k-\mathbf{MN}]\right\}\mathrm{d}\beta_k \\
= &\frac{1}{\sqrt{2\pi}\sigma}\frac{|\mathbf{M}|^{1/2}}{|\Sigma|^{1/2}}\exp\left\{-\frac{||y_{t+1}||^2}{2\sigma^2}\right\}, \qquad (5.23)
\end{aligned}$$

then the posterior mean and variance of $\beta_k$ are $\mu = MN$ and $\sigma^2\eta^2 = \sigma^2 M$ respectively. Combined with (5.23), we continue to integrate out $\sigma^2$ as below:

$$\begin{aligned}
&\int_{\sigma^2}\frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{||y_{t+1}||^2}{2\sigma^2}\right\}\frac{(\gamma/2)^{\nu/2}}{\Gamma(\nu/2)}\left(\frac{1}{\sigma^2}\right)^{\nu/2-1}e^{-\frac{\gamma}{2\sigma^2}}\mathrm{d}\sigma^2 \\
= &\frac{1}{\sqrt{2\pi}}\frac{(\gamma/2)^{\nu/2}}{\Gamma(\nu/2)}\int_{\sigma^2}(\sigma^{-2})^{\frac{\nu+1}{2}-1}\exp\left\{-\frac{||y_{t+1}||^2+\gamma}{2}\sigma^{-2}\right\}\mathrm{d}\sigma^2 \\
= &\frac{1}{\sqrt{2\pi}}\frac{(\gamma/2)^{\nu/2}}{((||y_{t+1}||^2+\gamma)/2)^{(\nu+1)/2}}\frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)}, \qquad (5.24)
\end{aligned}$$

again the posterior parameters are $\tilde{\nu} = \nu + 1$ and $\gamma = \gamma + ||y_{t+1}||^2$ respectively. Finally, we have

$$w^{(i)} = \frac{1}{\sqrt{\pi}}\frac{(\gamma)^{\nu/2}}{(||y_{t+1}||^2+\gamma)^{(\nu+1)/2}}\frac{\Gamma((\nu+1)/2)}{\Gamma(\nu/2)}\frac{|\mathbf{M}|^{1/2}}{|\Sigma|^{1/2}}. \qquad (5.25)$$

Correspondingly, the posterior parameters of beta and sigma can be updated as

$$\begin{aligned}
\mathbf{M}_t &= (\mathbf{H}_t^T\mathbf{H}_t + \Sigma_{t-1}^{-1})^{-1}, \\
\mathbf{N}_t &= (\mathbf{H}_t^T y_t + \Sigma_t^{-1}\vec{\mu}_t), \\
||y_t||^2 &= y_t^2 + \vec{\mu}_t^T\Sigma_t^{-1}\vec{\mu}_t - \mathbf{N}_t^T\mathbf{M}_t\mathbf{N}_t, \\
\vec{\mu}_{t+1} &= \mathbf{M}_t\mathbf{N}_t, \\
\Sigma_{t+1} &= \mathbf{M}_t, \\
\nu_{t+1} &= \nu_t + 1, \\
\gamma_{t+1} &= \gamma_t + ||y_t||^2.
\end{aligned}$$

## C: Formulas for the smooth functions

Heavisine:

$$h(x) = 4\sin(4\pi x) - \text{sgn}(x - 0.3) - \text{sgn}(0.72 - x);$$

Blocks:

$$h(x) = \sum h_k K(x - x_k);$$

where

$$\begin{aligned} K(t) &= (1 + \text{sgn}(t))/2; \\ x_k &= (0.1, 0.13, 0.15, 0.23, 0.25, 0.4, 0.44, 0.65, 0.76, 0.78, 0.81); \\ h_k &= (4, -5, 3, -4, 5, -4.2, 2.1, 4.3, -3.1, 2.1, -4.2); \end{aligned}$$

Bumps:

$$h(x) = \sum h_k K((x - x_k)/w_k);$$

where

$$\begin{aligned} K(t) &= (1 + |t|^4)^{-1}; \\ x_k &= (0.1, 0.13, 0.15, 0.23, 0.25, 0.4, 0.44, 0.65, 0.76, 0.78, 0.81); \\ h_k &= (4, 5, 3, 4, 5, 4.2, 2.1, 4.3, 3.1, 2.1, 4.2); \\ w_k &= (0.005, 0.005, 0.006, 0.01, 0.01, 0.03, 0.01, 0.01, 0.005, 0.008, 0.005); \end{aligned}$$

Doppler:

$$h(x) = (x(1-x))^{0.5}\sin(2\pi(1+\epsilon)/(x+\epsilon)), \qquad \epsilon = 0.05.$$

# Chapter 6

# Conclusion

The thesis has developed direct simulation methods for the multiple changepoint problems. The methods are based on the idea of particle filters, and our main contributions are in the following four areas.

Firstly, we have proposed an exact on-line algorithm to infer the changepoint model. That is to construct an exact filtering recursion for the changepoints over the time. The computational cost is quadratic in the number of observations. Once the filtering recursions have been solved, the direct simulations from the joint distribution of the changepoints is straightforward;

Secondly, we have introduced resampling ideas from particle filter to reduce the computational cost. We have extended two resampling methods on the basis of optimal resampling of Fearnhead and Clifford (2003) and rejection control of Liu et al. (1998), to involve stratified sampling. The importance of the resampling is that it provides an algorithm that scales linearly with the amount of data, at the expense of introducing small errors. The computing saving is important for application to large data sets such as the problem of inferring human GC content that we considered. Our new stratified rejection control algorithm allows the particle filter to automatically choose the number of particles required at each

time-step, and this can lead to large reductions in error over methods with a fixed number of particles at each step. Both the two new resampling algorithms substantially outperform standard resampling algorithms on examples we consider. The approximation error brought by the two resampling algorithms can also be bounded in terms of Kolmogorov-Smirnov distance.

Thirdly, we have extended the on-line algorithm to make it applicable for the changepoint models with Markov dependence structure. However, unlike the earlier case, we can not obtain the exact filtering distribution due to the dependence, so an approximation method is used. We further have shown empirically that the approximation error is small by checking a single changepoint case and the importance weights based on a piecewise polynomial model.

Finally, we have applied our model to the specific curve fitting problem in which the underlying curve is normally subjected to the continuity at some changepoints. A piecewise polynomial model is considered. Compared with alternative models, the main advantage of our model is that it allows for both discontinuous and continuous changepoints. Our algorithm also has the advantage in terms of the computing speed as there is a magnitude of saving on the computational cost compared to the traditional RJMCMC method.

# Bibliography


Akashi, H. and Kumamoto, H. (1977). Random sampling approach to state estimation in switching environments. *Automatica*, 13:429–434.

Anderson, B. D. O. and Moore, J. B. (1979). *Optimal Filtering.* Englewood Cliffs, New Jersey:Prentice Hall inc.

Anderson, T. W. (1962). The choice of the degree of polynomial regression as a multiple decision problem. *Ann. Math. Statist*, 33:255–265.

Arulampalam, S., Maskell, S., Gordon, N., and Clapp, T. (2002). A tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50.

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

Barry, D. and Hartigan, J. A. (1993). A Bayesian analysis for change point problems. *Journal of the American Statistical Association*, 88:309–319.

Baum, L. E., Petrie, T. P., Soules, G., and Weiss, N. (1970). A maximisation technique occuring in the statistical analysis of probabilistic functions of Markov chains. *Ann. Math. Statist.*, 41:164–171.

Bernardi, G. (2000). Isochores and evolutionary genomics of vertebrates. *Gene*, 241:3–17.

Berzuini, C., Best, N. G., Gilks, W. R., and Larizza, C. (1997). Dynamic conditional independence and Markov chain Monte Carlo methods. *Journal of the American Statistical Association.*, 92:1403–1412.

Boys, R. J. and Henderson, D. A. (2004). A Bayesian approach to DNA sequence segmentation. *Biometrics*, 60:573–588.

Braun, J. V., Braun, R. K., and Muller, H. G. (2000). Multiple changepoint fitting via quasilikelihood, with application to DNA sequence segmentation. *Biometrika*, 87:301–314.

Braun, J. V. and Muller, H. G. (1998). Statistical methods for DNA sequence segmentation. *Statistical Science*, 13:142–162.

Briers, M., Doucet, A., and Maskell, M. (2004). Smoothing algorithms for state-space models. *Technical Report*, Department of Engineering, University of Cambridge.

Brooks, R. J. (1972). A decision theory approach to optimal regression designs. *Biometrika*, 59:563–571.

Brooks, S. P., Giudici, P., and Roberts, G. O. (2003). Efficient contruction of reversible jump Markov chain Monte Carlo proposal distributions. *Journal of the Royal Statistical Society, Ser. B*, 65:3–39.

Bucy, R. S. and Senne, K. D. (1971). Digital synthesis of Nonlinear Filters. *Automatica*, 7:287–298.

Cappe, O., Moulines, E., and Ryden, T. (2005). *Inference in Hidden Markov Models.* Springer, New York.

Carlin, B. P., Gelfand, A. E., and Smith, A. F. M. (1992). Hierarchical Bayesian analysis of changepoint problems. *Applied Statistics*, 41:389–405.

Carpenter, J., Clifford, P., and Fearnhead, P. (1999). Improved particle filter for nonlinear problems. *IEE proceedings of radar and sonar navigation*, 146:2–7.

Casella, G. and Robert, C. P. (1996). Rao-Blackwellization of sampling shcemes. *Biometrika*, 83:81–94.

Chen, J. and Gupta, A. K. (1997). Testing and locating changepoints with application to stock prices. *Journal of the American Statistical Association*, 92:739–747.

Chen, R. and Liu, J. (2000). Mixture Kalman filters. *Journal of the Royal Statistical Society, Ser. B*, 62:493–508.

Chib, S. (1996). Calculating posterior distribution and modal estimates in Markov mixture model. *Journal of Econometrics*, 75:79–98.

Chib, S. (1998). Estimation and comparison of multiple change-point models. *Journal of Econometrics*, 86:221–241.

Chopin, N. (2002). A sequential particle filter method for static models. *Biometrika*, 89:539–551.

Chopin, N. (2007). Dynamic detection of change points in long time series. *Annals of the Institute of Statistical Mathematics*, 59:349–366.

Clapp, T. C. and Godsill, S. J. (1999). Fixed-lag smoothing using sequential importance sampling. In Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M., editors, *Bayesian Statistics 6*. Oxford University Press, Oxford.

Cochran, W. G. (1963). *Sampling Techniques, 3rd edition*. London, Wiley.

Crisan, D. and Lyons, T. (1997). A particle approximation of the solution of the Kushner-Stratnovitch equation. *Technique Report*, Imperial College, 180 Queen's Gate, London, SW7 2BZ, Department of Mathematics.

Del Moral, P., Doucet, A., and Jasra, A. (2006). Sequential Monte Carlo samplers. *Journal of the Royal Statistical Society, Series B*, 68:411–436.

Denison, D. G. T., Mallick, B. K., and Smith, A. F. M. (1998). Automatic bayesian curve fitting. *Journal of the Royal Statitical Society, series B*, 60:333–350.

DiMatteo, I., Genovese, C. R., and Kass, R. (2001). Bayesian curve-fiiting with free-knot splines. *Biometrika*, 88:1055–1071.

Donoho, D. L. and Johnstone, I. M. (1994). Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81:425–455.

Doucet, A. (1998). On sequential simulation based methods for Bayesian filtering. *Technique Report*, CUED/F-INFENG/TR.310:University of Cambridge, Signal Processing Group, Department of Engineering.

Doucet, A., de Freitas, J. F., Murphy, K., and Russell, S. (2000a). Rao-Blackwellised particle filtering for dynamic Bayesian networks. *Uncertainty in Artificial Intelligence.*

Doucet, A., de Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practise.* Springer–Verlag; New York.

Doucet, A., Godsill, S., and Andrieu, C. (2000b). On sequentail Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, 10:197–208.

Fan, J. and Gijbel, I. (1995). Data-driven bandwidth selection in local polynomial fitting: variable bandwidth and spatial adaption. *Journal of the Royal Statistical Society, Ser. B*, 57:371–394.

Fearnhead, P. (1998). *Sequential Monte Carlo Methods in Filter Theory.* Thesis of DPhil., Oxford University.

Fearnhead, P. (2005). Exact Bayesian curve fitting and signal segmentation. *IEEE Transactions on Signal Processsing*, 53:2160–2166.

Fearnhead, P. (2006). Exact and efficient Bayesian inference for multiple change-point problems. *Statistics and Computing*, 16:203–213.

Fearnhead, P. and Clifford, P. (2003). Online inference for hidden Markov models. *Journal of the Royal Statistical Society, Series B*, 65:887–899.

Fearnhead, P. and Liu, Z. (2007). On-line inference for multiple changepoints problems. *Journal of the Royal Statistical Society, Series B*, 69:589–605.

Fishman, G. S. (1996). *Monte Carlo - Concepts, Algorithms and Applications.* New York: Springer.

Friedman, J. H. (1991). Multivariate adaptive regression splines. *Ann. Statist.*, 11:1–141.

Friedman, J. H. and Silverman, B. W. (1989). Flexible parsimonious smoothing and additve modelling (with discussion). *Technometrics*, 31:3–39.

Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo integration. *Econometrica*, 24:1317–1399.

Gilks, W. R. and Berzuini, C. (2001). Following a moving target - Monte Carlo inference for dynamic Bayesian models. *Journal of the Royal Statistical Society, Series B*, 63:127–146.

Gilks, W. R., Richardson, S., and Spiegelhalter, D. J. e. (1996). *Markov chain Monte Carlo in Practise.* Chapman & Hall, London.

Godsill, S., Doucet, A., and West, M. (2004). Monte Carlo smoothing for nonlinear time series. *Journal of the American Statistical Association*, 99:156–168.

Godsill, S. J. and Clapp, T. C. (2001). Fixed-lag smoothing using sequential importance sampling. In Doucet, A., de Freitas, N., and gordon, N., editors, *Sequential Monte Carlo Methods in Practice*, pages 137–158. pringer–Verlag; New York.

Gordon, N., Salmond, D., and Smith, A. F. M. (1993). Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F*, 140:107–113.

Green, P. (1995). Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732.

Green, P. J. and Silverman, B. W. (1994). *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach.* Chapman & Hall/CRC.

Guttman, I. (1967). The use of concept of a future observation in goodness-of-fit problems. *Journal of the Royal Statistical Society, Ser. B*, 29:83–100.

Halpern, E. F. (1973). Polynomial regession from a bayesian approach. *Journal of the American Statistical Association*, 68:137–143.

Hammersley, J. M. and Handscomb, D. C. (1964). *Monte Carlo Methods.* London:Methuen.

Handschin, J. E. and Mayne, D. Q. (1969). Monte carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International Journal of Control*, 9:547–559.

Handschin, J. E. and Mayne, D. Q. (1970). Monte Carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, 6:555–563.

Hansen, M. H. and Kooperberg, C. (2002). Spline adaption in extended linear models. *Stastical Science*, 17(1):2–51.

Hardison, R. C., Roskin, K. M., Yanf, S., Diekhans, M., Kent, W. J., Weber, R., Elnitski, L., and Li et al., J. (2003). Covariation in frequencies of substitution, deletion, transposition, and recombination during eutherian evolution. *Genome Research*, 13:13–26.

Harger, H. and Antle, C. (1968). The choice of the degree of polynomial model. *Journal of the Royal Statistical Society, Ser. B*, 30:469–471.

Hastie, M. H. and Tibshirani, R. (1990). *Generalised Additive Models.* Chapman & Hall, London.

Hasting, W. K. (1970). Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57:97–109.

Hinkley, D. V. (1971). Inference about the change-point from cumulative sum tests. *Biometrika*, 58:509–523.

Hoeting, J. A., Madigan, D., Raftery, A. E., and Volinsky, C. T. (1999). Bayesian model averaging: a tutorial. *Statistical Science*, 14(4):382–417.

Hurzeler, H. and Kunsch, H. R. (1998). Monte Carlo approximation for general state space model. *Journal of Computational and Graphical Statistics*, 7(2):175–193.

Isard, M. and Blake, A. (1996). Contour tracking by stochastic propagation of conditional density. In *Proc. European Conf. Computer Vision*. Cambridge, UK.

Jazwinski, A. H. (1973). *Stochastic Process and Filtering Theory*. Academic Press.

Johnson, T. D., Elashoff, R. M., and Harkema, S. J. (2003). A Bayesian change-point analysis of electromyographic data: detecting muscle activation patterns and associated applications. *Biostatistics*, 4:143–164.

Kalman, R. and Bucy, R. (1961). New results in linear filtering and prediction theory. *Journal of Basic Engineering, Transactions ASME Series, D*, 83:95–108.

Kass, R. E. and Raftery, A. E. (1995). Bayes factor and model uncertainty. *Journal of the American Statistical Association*, 90:773–795.

Kass, R. E. and Wasserman, L. (1995). A reference Bayesian test for nested hypotheses and its relationship to the Schwarz criterion. *Journal of the American Statistical Association*, 90:928–934.

Kitagawa, G. (1987). Monte Carlo filter and smoother for non-Gaussian non-linear state space models. *Journal of the American Statistical Association*, 82 (400):1032–1063.

Kitagawa, G. (1996). Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of the Computational and Graphical Statistics*, 5:1–25.

Kong, A., Liu, J. S., and Wong, W. H. (1994). Sequential imputations and Bayesian missing data problem. *Journal of the American Statistical Association*, 89:278–288.

Liu, J. S. (1996). Metropolised independent sampling with comparsion to rejection sampling and importance sampling. *Statistics and Computing*, 6:113–119.

Liu, J. S. (2001). *Monte Carlo Strategies in Scienctific Computing*. Springer–Verlag; New York.

Liu, J. S. and Chen, R. (1995). Blind deconvolution via sequential imputations. *Journal of the American Statistical Association.*, 90:567–576.

Liu, J. S. and Chen, R. (1998). Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93:1032–1044.

Liu, J. S., Chen, R., and Logvinenko, T. (2001). A theoretical framework for sequential importance sampling with resampling. In Doucet, A., de Freitas, N., and gordon, N., editors, *Sequential Monte Carlo Methods in Practice*, pages 225–246. Springer–Verlag; New York.

Liu, J. S., Chen, R., and Wong, W. H. (1998). Rejection control and sequential importance sampling. *Journal of the American Statistical Association*, 93:1022–1031.

Liu, J. S. and Lawrence, C. E. (1999). Bayesian inference on biopolymer models. *Bioinformatics*, 15:38–52.

Lund, R. and Reeves, J. (2002). Detection of undocumented changepoints: A revision of the two-phase regression model. *Journal of Climate*, 15:2547–2554.

Mallick, B. K. (1998). Bayesian curve fitting by polynomial of random order. *J. Statist. Plann. Inference*, 70:91–109.

Mayne, D. Q. (1966). A solution of the smoothing problem for linear dynamic systems. *Automatica*, 4:73–92.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. (1953). Equations of state calculations by fast computing machines. *J. Chem. Phys*, 21:1087–1091.

Muller, H. G. and Stadtmuller, U. (1987). Variable bandwidth kernel estimators of regression curves. *Annal of Statistics*, 15:182–201.

Murphy, K. and Russell, S. (2001). Rao-Blackwellised particle filtering for dynamic Bayesian networks. In Doucet, A., de Freitas, N., and gordon, N., editors, *Sequential Monte Carlo Methods in Practice*, pages 499–515. Springer–Verlag; New York.

Neal, R. M. (1998). Annealed importance sampling. *Technique Report*, 9805:University of Toronto, Toronto, Ontario, Canada, Department of Statistics and Department of Computer Science.

Oliver, J. L., Bernaola-Galvan, P., Carpena, P., and Roman-Roldan, R. (2001). Isochore chromosome maps of eukaryotic genomes. *Gene*, 276:47–56.

Oliver, J. L., Carpena, P., Hackenberg, M., and Bernaola-Galvan, P. (2004). IsoFinder: computational prediction of isochores in genome sequences. *Nuceleic Acids Research*, 32:W287–W292. Web Server Issue.

Oliver, J. L., Carpena, P., Roman-Roldan, R., Mata-Balaguer, T., Mejias-Romero, A., Hackenberg, M., and Bernaola-Galvan, P. (2002). Isochore chromosome maps of the human genome. *Gene*, 300:117–127.

Ó Ruanaidh, J. J. K. and Fitzgerald, W. J. (1996). *Numerical Bayesion Methods Applied to Signal Processing.* New York: Springer.

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: auxiliary particle filters. *Journal of the American Statistical Association.*, 94:590–599.

Punskaya, E., Andrieu, C., Doucet, A., and Fitzgerald, W. J. (2002). Bayesian curve fitting using MCMC with applications to signal segmentation. *IEEE transactions on signal processing*, 50:744–758.

Raftery, A. E. and Akman, V. E. (1986). Bayesian analysis of a Poisson process with a change-point. *Biometrika*, 73:85–89.

Ritov, Y., Raz, A., and Bergman, H. (2002). Detection of onset of neuronal activity by allowing for heterogeneity in the change points. *Journal of Neuroscience Methods*, 122:25–42.

Robert, C. P. and Casella, G. (1999). *Monte Carlo Statistical Methods.* Springer–Verlag.

Rue, H. and Held, L. (2005). *Gaussian Markov Random Fields: Theory and Applications*, volume 104 of *Monographs on Statistics and Applied Probability.* Chapman & Hall, London.

Salmenkivi, M., Kere, J., and Mannila, H. (2002). Genome segmentation using piecewise constant intensity models and reversible jump MCMC. *Bioinformatics*, 18:S211–S218.

Shao, J. (1993). Linear model selection by cross validation. *Journal of the American Statistical Association*, 88:486–494.

Smith, A. F. M. (1975). A Bayesian approach to inference about a change-point in a sequence of random variables. *Biometrika*, 62:407–416.

Smith, M. and Kohn, R. (1996). Nonparameteric regression using Bayesian variable selection. *J. Econometrics*, 75:317–343.

Smith, P. L. (1982). Curve fitting and modelling with splines using statistical variable selection techniques. *NASA report 166034*, pages Langely Research Centre, Hampton, VA.

Sorenson, H. W. and Alspach, D. L. (1971). Recursive Bayesian estimations using Gaussian sums. *Automatica*, 7:465–479.

Stephens, D. A. (1994). Bayesian retrospective multiple-changepoint identification. *Applied Statistics*, 43:159–178.

Stone, C. J., Hansen, M., Kooperberg, C., and Truong, Y. K. (1997). Polynomial splines and their tensor products in extended linear modelling. *Ann. Statist.*, 25:1371–1470.

Sutherland, A. I. and Titterington, D. M. (1994). Bayesian analysis of image sequences. *Technique Report*, 94-3:Department of Statistics, University of Glasgow, UK.

Tierney, L. (1994). Markov chain for exploring posterior distributions. *Annal of Statistics*, 22:1701–1762.

Wahba, G. (1975). Smoothing noisy data with spline functions. *Number Math*, 24:383–393.

Wahba, G. (1990). *Spline Models for Observational Data.* Society for Industrial and Applied Mathematics, Philadelphia.

West, M. (1992). Modelling with Mixtures. In Bernardo, J. M., Berger, J. O., Dawid, A. P., and Smith, A. F. M., editors, *Bayesian Statistics 4.* Clarenton Press, London.

West, M. (1993). Mixture models, Monte Carlo, Bayesian updating and dynamic models. In *Computing Science and Statistics, 24*, pages 325–333. Interface Foundation of North America, Fairfax Station, VA.

West, M. and Harrison, J. (1997). *Bayesian forcasting and dynamic models.* New York: Springer.

Worsley, K. J. (1979). On the likelihood ratio test for a shift in location of normal populations. *Journal of the American Statistical Association*, 74:363–367.

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

Zhou, S. and Shen, X. (2001). Spatially adaptive regression splines and accurate knot selection schemes. *Journal of the American Statistical Association*, 96:247–259.