Detecting Change Points in Time Series Using the Bayesian approach with Perfect Simulation

Andrew Stephen Richens

A thesis presented for the degree of

Master of Science

in

Statistics

at the

University of Canterbury,

Christchurch, New Zealand.

8 February 2008

Acknowledgements

I would like to thank everyone involved in the process of this thesis. In particular, I wish to acknowledge Dr. Dominic Lee, my primary supervisor, for his assistance and guidance throughout. I also want to thank my assistant supervisors: Marco Reale for answering questions I had, and Glynn Russell at Christchurch Women's hospital for the data used in chapter 6.

I would also like to thank Bill Rea, Marina Zahari and Xin Zhao in particular, for their help on answering any statistical questions that I had, and also my office mate Michael Langton for much help on computers in general, and Matlab in particular.

Many other people have made this time very enjoyable. In particular, I wish to thank the other postgraduate students in the department: Ron Begg, Hannes Diener, Scott Graybill, Klaas Hartman, Johnny Humphries, Maarten Jordens, Michael Langton and Daniel Lond.

The University of Canterbury, and in particular, the Mathematics and Statistics Department has supported me throughout my entire university career. I would like to thank in particular those staff who have taught and encouraged me over the last seven years.

Finally I wish to thank my Family. My daughter Rivers who has been lots of fun growing, and to my sons Star and Canyon, who both came along during the course of this research. I also owe much gratitude to my wife Kathleen, without whose support this could never have started, and without whose assistance this would never have finished.

Table of Contents

Abstract

Many regression problems can be modelled as independent linear regressions on disjoint segments. The problem of interest is to find the number and location of the changepoints where the segments end, and to find the model order inside each segment. A new approach presented in Fearnhead (2005, 2006) is considered. This is a Bayesian approach using perfect simulation from the posterior distribution of the model. Some improvements to this algorithm are suggested: a method for selecting parameters for the prior distribution used, and an algorithm that eliminates a source of error found in Fearnhead (2005, 2006). This method is analysed by testing it on several simulations, with different model types and order. Three real datasets are then investigated; these are geological data, road safety data and medical data of preterm babies. Despite errors in certain situations, the algorithm is shown to be successful in many of the investigated cases, and an easy and efficient way of finding changepoints.

Chapter 1

Introduction

1.1 The problem

Changepoints models are often required when performing regression on time series. Changepoints occur when there is a change in the parameters of the underlying model of the time series. The data for each segment between changepoints arise from a single model, with different models for each segment, where the number and positions of the changepoints are assumed to be unknown. The changepoint problem is then to find these changepoints, and the model order in each segment. Examples of changepoint problems are Gaussian models with changing variance (Chen and Gupta, 1997; Johnson et al., 2003), Markov models with time-varying matrices (Braun and Muller 1998), and linear regression with varying regression parameters. Such models arise in areas as diverse as finance, biological sciences, engineering and medicine.

1.2 Research Objective

The objective in this research is to study the effectiveness of a new method of changepoint detection, suggested recently by Fearnhead (2005, 2006), and to make some improvements to this method. This method makes use of the Bayesian approach together with a perfect simulation technique, and is an offline procedure. As such, it is suitable for the retrospective analysis and understanding of time series data. The effectiveness of the method will be analysed through several Monte Carlo simulation studies and three applications to real data: geological, transportation and medical data sets. The results will give an indication of the strengths and weaknesses of the method, and to the additions made.

1.3 Review of Relevant Research

Most of the recent research in change-point detection adopts the Bayesian approach (see, for example, Yang and Kuo (2001) and Barry and Hartigan (1993)) which allows direct probability statements about unknown quantities, as opposed to weaker confidence statements about them. The Bayesian approach takes inference with probability models to its natural mathematical conclusion without the need for other ad hoc tools. Moreover, the Bayesian approach allows prior information and constraints to be easily incorporated into the inferential process. Bayesian solutions are often unavailable analytically and have to be computed numerically using Monte Carlo procedures. There are two main ways to do this: Markov chain Monte Carlo (MCMC), and perfect simulation. In both cases, a sample is generated from the posterior distribution, and used for inference. In

MCMC, a Markov chain is designed whose stationary distribution is the desired posterior distribution. By running the Markov chain until it is stationary, the required sample points are given by the states of the stationary chain. Previously, MCMC has been the favoured method for the changepoint problem (Chib 1998, Stephens 1994). In practice, however, it is extremely difficult to determine when the Markov chain has converged to its stationary distribution. Therefore, whenever it is possible to simulate directly and independently from the posterior distribution this is preferred, thereby avoiding the convergence problem faced with MCMC.

The method studied in this research represents the state of the art in change point detection (Fearnhead, 2005, 2006). The perfect simulation method that is used is a forward-backward algorithm based on product partition models (Barry and Hartigan, 1992).

1.4 Outline

This thesis is focused on a recent method for finding changepoints proposed by Fearnhead (2005, 2006); using perfect simulation to sample from the posterior distribution of changepoints. This method is described in detail, some adaptations are suggested, and it is tested on simulated data. The method is then used to investigate some real data.

In chapter two the method from Fearnhead (2005, 2006) is explained. The model, justification of choice of prior distributions, method of simulation and implementation is

covered in detail. A new method for decreasing the subjectivity of the algorithm using a particular choice of hyperparameters is detailed. Another extension to the method of Fearnhead (2005, 2006) is described; where errors are reduced by grouping of the posterior probabilities.

In chapter three the method is applied to many simulated data sets, to test its strengths and weaknesses. Different data sets with polynomial and autoregressive models of various model orders and signal to noise ratios are investigated, to gauge the performance of the algorithm in different situations.

Chapters four, five and six are investigations of specific real data sets. In chapter four data from a geological application is assayed; investigating the change in rock types in a bore hole. Chapter 5 investigates the effect that seat belt legislation had in the number of road deaths and injuries in the UK, while in chapter 6 the algorithm is used to investigate the applicability of a particular model for monitoring the health of premature babies.

Chapter 2

Algorithm

2.1 Introduction

In this chapter we look at the algorithm (Fearnhead 2005, 2006) that we use to find changepoints in time series. We describe in detail the model and the required prior distributions, and how these are used to perfectly sample from the posterior probabilities of given points being changepoints. We then discuss practical considerations of using this algorithm, and look at some adaptations to improve ease of use and performance. We discuss ways of decreasing the subjectivity of the choice of hyperparameters by developing methods to automate them, and finally describe a way to reduce some error in the results.

2.2 Notation and Model

We assume we have a time series of length n. We denote the data $y = y_{1:n} = (y_1, y_2, \ldots, y_n)$, and a subset of the data by $y_{i:j} = (y_i, y_{i+1}, \ldots, y_j)$. We have m segments, where we define a segment to be the data points between two changepoints; a

changepoint is taken to be the last point of a segment. We define the first changepoint $\tau_0 = 0$, and the rest as $\tau_1, \tau_2, \ldots, \tau_m$, with $\tau_m = n$. Thus the *i*th segment is written $y_{(\tau_{i-1}+1):\tau_i}$.

We can then model this segment with a linear regression of order p_i, and denote the vector of these p_i regression coefficients β_i. We then define G_i to be the matrix of basis functions. Throughout this thesis we will be mainly looking at polynomial and AR models, although it is possible to use this method to find changepoints for any data that can be modelled by a linear regression. Thus the model for the *i*th segment is

$$y_{(\tau_{i-1}+1):\tau_i} = G_i \beta_i + \varepsilon_{(\tau_{i-1}+1):\tau_i} \qquad 2.1$$

where the ε term is a vector of independent and identically distributed (iid) normal random variables, with mean 0 and variance σ_i^2. We assume the number and position of changepoints, the number and value of the regression parameters and the variance of the error term to be unknown.

We perform a Bayesian inference on the number and position of changepoints and the model order of the segments, allowing us to make probability statements about these unknown quantities. The posterior distribution for the changepoints is directly and independently simulated from, avoiding the (often difficult) problem of deciding whether an approximate method has converged to the required distribution. Once an inference on

the location of changepoints is complete, the distribution of model order in each segment can be calculated.

For the rest of this chapter, we take $y_{s:t}$ to be a segment, and so we stop using subscripts (*i* previously) to indicate which segment we refer to. We assume all prior distributions to be independent across segments.

Following Fearnhead (2005, 2006) we use an inverse gamma prior distribution with shape parameter $\frac{\nu}{2}$ and scale parameter $\frac{\gamma}{2}$ for σ², and for the *j*th regression coefficient we use a normal prior with mean zero and variance $\sigma^2\delta_j^2$. These distributions are chosen because they are conjugate priors, which helps to simplify the following equations.

Thus if $y_{s:t}$ is a segment with model order *q*, then the relevant prior distributions would be

$$\begin{aligned} \sigma^2 &\sim IG\left(\frac{\nu}{2},\frac{\gamma}{2}\right) \\ \beta_j \mid \sigma^2 &\sim N\left(0,\sigma^2{\delta_j}^2\right), \end{aligned} \qquad 2.2$$

and, from equation 2.1:

$$y_{s:t} \mid \beta,\sigma^2 \sim N\left(G\beta,\sigma^2 I\right). \qquad 2.3$$

The density functions for these distributions are

$$f\left(\sigma^2\right)=\frac{\left(\frac{\gamma}{2}\right)^{\frac{\nu}{2}}}{\Gamma\left(\frac{\nu}{2}\right)}\left(\sigma^2\right)^{\frac{-\nu-2}{2}}e^{\frac{-\gamma}{2\sigma^2}} \qquad 2.4$$

$$f\left(\beta\mid\sigma^2\right)=\prod_{i=1}^{q}f\left(\beta_i\mid\sigma^2\right)=\frac{1}{\left(2\pi\sigma^2\right)^{\frac{q}{2}}\prod_{i=1}^{q}\delta_i}e^{\frac{-1}{2\sigma^2}\beta^T\Delta^{-1}\beta} \qquad 2.5$$

(since the prior distributions are independent between segments), and

$$f\left(y_{s:t}\mid\beta,\sigma^2\right)=\frac{1}{\left(2\pi\sigma^2\right)^{\frac{t-s+1}{2}}}e^{\frac{-1}{2\sigma^2}\left(y_{s:t}-G\beta\right)^T\left(y_{s:t}-G\beta\right)} \qquad 2.6$$

where $\Delta = diag(\delta_1,\delta_2,\ldots,\delta_q)$.

Other unknowns for which prior distributions are needed are the number and position of changepoints, and the regression order *q*, which we fix to be less than a maximum model order $\overline{p}$. The priors we use for these are geometric and uniform, respectively. That is,

$$\begin{aligned} f\left(m,\tau_1,\tau_2,\ldots,\tau_m\right)&=\lambda^m\left(1-\lambda\right)^{n-m}\\ f\left(q\right)&=\frac{1}{\overline{p}} \end{aligned} \qquad 2.7$$

where λ is a hyperparameter for the prior geometric distribution. We use a uniform prior for q, since we have no prior information about the model order.

2.3 Generation from the Posterior

In this section we look at how to obtain a perfect simulation from the posterior distribution of the number and position of changepoints.

2.3.1 Setup

We look at the density function of a particular section of the data, $y_{s:t}$ given that $s:t$ is a segment, and the model order is q.

Therefore G is a $(t-s+1)\times q$ matrix of basis functions for a model of order q.

First we note that

$$f(y_{s:t},\beta,\sigma^2)=f(y_{s:t}\mid\beta,\sigma^2)\cdot f(\beta\mid\sigma^2)\cdot f(\sigma^2) \qquad 2.8$$

and so the marginal density of $y_{s:t}$ is

$$f(y_{s:t})=\iint f(y_{s:t},\beta,\sigma^2)d\beta d\sigma^2. \qquad 2.9$$

Now, from equations 2.4-2.6,

$$f(y_{s:t}) = \int\int \frac{1}{(2\pi\sigma^2)^{\frac{t-s+1}{2}}} e^{\frac{-1}{2\sigma^2}(y_{s:t}-G\beta)^T(y_{s:t}-G\beta)} \frac{1}{(2\pi\sigma^2)^{\frac{q}{2}} \prod_{i=q}^{q} \delta_i} e^{\frac{-1}{2\sigma^2}\beta^T\Delta^{-1}\beta} \times \frac{\left(\frac{\gamma}{2}\right)^{\frac{\nu}{2}}}{\Gamma\left(\frac{\nu}{2}\right)} (\sigma^2)^{\frac{-\nu-2}{2}} e^{\frac{-\gamma}{2\sigma^2}} d\beta d\sigma^2. \quad 2.10$$

Simplifying:

$$f(y_{s:t}) = \frac{\left(\frac{\gamma}{2}\right)^{\frac{\nu}{2}} (2\pi)^{\frac{s-t-q-1}{2}}}{\prod_{i=1}^{q} \delta_i \Gamma\left(\frac{\nu}{2}\right)} \int (\sigma^2)^{\frac{s-t-q-\nu-3}{2}} e^{\frac{-\gamma}{2\sigma^2}} \int e^{\frac{-1}{2\sigma^2}\left((y_{s:t}-G\beta)^T(y_{s:t}-G\beta)+\beta^T\Delta^{-1}\beta\right)} d\beta d\sigma^2. \quad 2.11$$

Now we note that

$$(y_{s:t}-G\beta)^T(y_{s:t}-G\beta)+\beta^T\Delta^{-1}\beta = (\beta-\Sigma G^T y_{s:t})^T \Sigma^{-1}(\beta-\Sigma G^T y_{s:t}) + \|y\|_P^2 \quad 2.12$$

where

$$\begin{aligned} \Sigma &= \left(G^T G + \Delta^{-1}\right)^{-1} \\ P &= \left(I - G\Sigma G^T\right) \\ \|y\|_P^2 &= y_{s:t}^T P y_{s:t} \end{aligned} \quad 2.13$$

where *I* here is the $(t-s+1)\times(t-s+1)$ identity matrix; consequently

$$\begin{aligned}\int e^{\frac{-1}{2\sigma^2}\left((y_{s:t}-G\beta)^T(y_{s:t}-G\beta)+\beta^T\Delta^{-1}\beta\right)}d\beta &= e^{\frac{-\|y\|_P^2}{2\sigma^2}}\int e^{\frac{-1}{2}\left(\beta-\Sigma G^T y_{s:t}\right)^T\left(\sigma^2\Sigma\right)^{-1}\left(\beta-\Sigma G^T y_{s:t}\right)}d\beta \\ &= \left(2\pi\sigma^2\right)^{\frac{q}{2}}\left|\Sigma\right|^{\frac{1}{2}}e^{\frac{-\|y\|_P^2}{2\sigma^2}},\end{aligned} \tag{2.14}$$

since we note that the integrand follows a multivariate normal distribution, up to a normalising constant. Hence

$$f\left(y_{s:t}\right)=\frac{\left(2\pi\right)^{\frac{s-t-1}{2}}\left|\Sigma\right|^{\frac{1}{2}}\prod_{i=1}^{q}\delta_i^{-1}\left(\frac{\gamma}{2}\right)^{\frac{\nu}{2}}}{\Gamma\left(\frac{\nu}{2}\right)}\int\left(\sigma^2\right)^{\frac{s-t-\nu-3}{2}}e^{\frac{-1}{2\sigma^2}\left(\gamma+\|y\|_P^2\right)}d\sigma^2, \tag{2.15}$$

where we again notice that the integrand is from a known density function, this time an inverse gamma distribution with shape $\frac{t-s+\nu+1}{2}$ and scale $\frac{1}{2}\left(\gamma+\|y\|_P^2\right)$, and so the normalising constant is

$$\int\left(\sigma^2\right)^{\frac{s-t-\nu-3}{2}}e^{\frac{-1}{2\sigma^2}\left(\gamma+\|y\|_P^2\right)}d\sigma^2=\Gamma\left(\frac{t-s+\nu+1}{2}\right)\left(\frac{\gamma+\|y\|_P^2}{2}\right)^{\frac{s-t-\nu-1}{2}}. \tag{2.16}$$

Consequently

$$P(s,t,q) = f(y_{s:t}) = \frac{\pi^{\frac{s-t-1}{2}} |\Sigma|^{\frac{1}{2}} \prod_{i=1}^{q} \delta_i^{-1} \gamma^{\frac{\nu}{2}} \left(\gamma + \|y\|_P^2\right)^{\frac{s-t-\nu-1}{2}} \Gamma\left(\frac{t-s+\nu+1}{2}\right)}{\Gamma\left(\frac{\nu}{2}\right)}. \qquad 2.17$$

Now we define

$$Q(s) = P(y_{s:n} \mid \text{change point at } s-1) \qquad 2.18$$

Thus we have

$$\begin{aligned} Q(s) = \sum_{t=s}^{n-1} & \Pr(y_{s:n}, \text{ next changepoint at } t \mid \text{change point at } s\text{-}1) \\ & + \Pr(y_{s:n}, \text{ no further changepoints} \mid \text{change point at } s\text{-}1) \end{aligned} \qquad 2.19$$

and so

$$Q(s) = \sum_{t=s}^{n-1} \sum_{q=1}^{\bar{p}} \frac{1}{\bar{p}} P(s,t,q) Q(t+1) \lambda (1-\lambda)^{t-s} + \sum_{q=1}^{\bar{p}} \frac{1}{\bar{p}} P(s,n,q) (1-\lambda)^{n-s} \qquad 2.20$$

2.3.2 Changepoints and Model Order

In this section we see how we use equations 2.17 and 2.20 to simulate from the posterior distribution of changepoints.

First we initialise the process by setting $\tau_0 = 0$. Then we calculate the probability of each point being the first changepoint, and simulating from this distribution. To do this we first check to see if there are no more changepoints, and if there are more, find the location of the first. Having found the first changepoint, we repeat the process to find the next. We iterate this procedure until we find that there are no more changepoints.

That is, after having found changepoint j-1 at time s-1, we calculate

$$\Pr\left(\tau_j = t \mid \tau_{j-1} = s-1, y_{1:n}\right) \propto \sum_{q=1}^{\bar{p}} \frac{1}{\bar{p}} P(s,t,q) Q(t+1) \lambda (1-\lambda)^{t-s} \qquad 2.21$$

$$\Pr\left(\tau_j = n \mid \tau_{j-1} = s-1, y_{1:n}\right) \propto \sum_{q=1}^{\bar{p}} \frac{1}{\bar{p}} P(s,n,q) (1-\lambda)^{n-s} \qquad 2.22$$

and scale these values so that their sum is one, and hence a probability distribution on $s, s+1, \ldots, n$. Then we draw a random number from a uniform distribution on (0,1), and check to see if this is less than the probability that there are no more changepoints. If it is, then we say that $\tau_j = n$, and therefore $y_{s:n}$ is the final segment. If not, then we find the first value of t for which our uniform random number is less than $\Pr\left(\tau_j = t \mid \tau_{j-1} = s-1, y_{1:n}\right)$, and set τ_j to be this value of t. We then repeat this process until $\tau_j = n$ for some value of j, and we say that $m = j$, i.e. there are j changepoints. Having completed this process, we now have a set of points that correspond to

changepoints in the time series, perfectly simulated from the posterior distribution of changepoints.

Having found these probabilities, we can now look at the posterior distribution of model order for each segment:

$$p(q \mid \tau_{i-1} = s-1, \tau_i = t, y_{1:n}) \propto \frac{1}{\overline{p}} P(s,t,q), \qquad 2.23$$

again scaling to ensure a distribution. Equation 2.23 depends on the location of the changepoints, so we need to make a decision for each point as to whether or not it is a changepoint. To do this we use the posterior probability of each point being a changepoint, and set a cutoff probability p_c, so that if our posterior probability is greater than p_c then we accept that point as a changepoint. For most purposes, $p_c = 0.5$ will suffice, so we accept those points that are more likely than not to be changepoints. We note, however, that in some cases the value for p_c that we have chosen will depend on the application.

2.4 Implementation

In this section we look at the practical considerations of applying the algorithm, as outlined above. First we take note of the quantities that we need to input into the algorithm. These inputs are $y_{1:n}, \overline{p}, \Delta, \nu, \gamma, \lambda, k, p_c$ and the type of basis function. $y_{1:n}$ is

the time series in which we wish to find the changepoints, and $\overline{p}$ is the maximum model order that we wish to consider. This latter is usually specified before our analysis.

The values Δ, ν, γ and λ are all hyperparameters for the various prior distributions as explained above. We discuss choosing values for these in the next section. k is the number of sample points we take from the posterior distribution of changepoints; for most cases we use 10,000. We set p_c as the cut-off probability for accepting a point as a changepoint, so that if the proportion of simulations that finds a time point to be a changepoint is greater than (or equal to) p_c, then we take that point to be a changepoint. This is relevant to the distribution of model order in each segment, as this is conditional on the position of the changepoints. The type of underlying functions with which we model each segment is required to define what type of basis vectors we use for computing G.

We can compute all of the relevant values of equations 2.17 and 2.20 before we start our simulations. For the value of $P(s,t,q)$ from equation 2.17, we create $\overline{p}$ separate $n \times n$ matrices, one for each different value of q. For a specific one of these matrices, the *(i,j)*-th entry is defined $P(i,j,q)$, noting that, since we always have $t > s$ in equation 2.17, this matrix will be upper triangular. Having calculated these, we can then use them to calculate the values in the vector Q. To find $Q(n)$, we substitute $s = n$ into equation 2.20, resulting in

$$Q(n) = \sum_{q=1}^{\bar{p}} \frac{1}{\bar{p}} P(n, n, q), \tag{2.24}$$

and then using the recursive definition of equation 2.20 we find the previous values of Q. We note here that since $Q(s)$ for a given time point depends on future values of $Q(s)$, that we require the entire data set to complete our inference on the location of the changepoints, and thus our algorithm is offline.

Equation 2.20 suffers from numerical instability (Fearnhead 2005), making implementation a problem. We can avoid this issue by using the following identities:

$$\log Q(s) = \log Q(s+1) + \log\left(\frac{Q(s)}{Q(s+1)}\right) \tag{2.25}$$

$$\begin{aligned}\frac{Q(s)}{Q(s+1)} = &\sum_{q=1}^{\bar{p}} \frac{1}{\bar{p}} \sum_{t=s}^{n-1} \lambda \\ &\times \exp\{\log P(s,t,q) + \log Q(t+1) \\ &\qquad - \log Q(s+1) + (t-s)\log(1-\lambda)\} \\ &+ \sum_{q=1}^{\bar{p}} \frac{1}{\bar{p}} \exp\left(\log P(s,n,q) - \log Q(s+1) + (n-s)\log(1-\lambda)\right)\end{aligned} \tag{2.26}$$

Thus in our implementation of the algorithm, we calculate the log of $Q(s)$ instead of $Q(s)$ itself, and when we come to find the values in equation 2.21, we then take the exponential of these values.

Since equation 2.26 only uses the log of the $P(s,t,q)$ values from equation 2.16, we only need calculate these log values:

$$\begin{aligned}\log P(s,t,q) = &\left(\frac{s-t-1}{2}\right)\log\pi + \frac{1}{2}\log|\Sigma| - \sum_{i=1}^{q}\log\delta_i + \frac{\nu}{2}\log\gamma \\ &+\left(\frac{s-t-\nu-1}{2}\right)\log\left(\gamma + \|y\|_P^2\right) + \log\Gamma\left(\frac{t-s+\nu+1}{2}\right) - \log\Gamma\left(\frac{\nu}{2}\right)\end{aligned} \qquad 2.27$$

again noting that we exponentiate these values for equations 2.21 and 2.22.

Now we note that equation 2.17 depends on P and Σ, both of which depend on *G*, our $(t-s+1)\times q$ matrix of basis functions (where we assume $y_{s:t}$ is a segment). Thus the *(i,j)*-th entry of *G* is the explanatory variable associated with the *j*th regression coefficient for the *i*th time point in our segment. The convention we use here is that a model of order *q* is one that has *q* regression coefficients, so, for example, a quadratic model would be of order 3; consequently, for a polynomial regression of order *q*, our matrix *G* would be

$$\begin{bmatrix} 1 & x_s & x_s^2 & \cdots & x_s^{q-1} \\ 1 & x_{s+1} & x_{s+1}^2 & \cdots & x_{s+1}^{q-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_t & x_t^2 & \ldots & x_t^{q-1} \end{bmatrix}, \qquad 2.28$$

where $x(s,s+1,\ldots,t)$ is our vector of explanatory variables. Since we are looking at time series, this vector ***x*** is made up of time values, and since they are a basis, the particular values we use are irrelevant, requiring only that $x_{a+1} = x_a + 1$, for all $a \in (s,s+1,\ldots,t)$.

For convenience we take $(s, s+1, \ldots, t)$ to be $(1,2,\ldots,t-s+1)$. Thus we can rewrite 2.28 as

$$\begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 2 & 4 & \cdots & 2^{q-1} \\ 1 & 3 & 9 & \cdots & 3^{q-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & t-s+1 & (t-s+1)^2 & \cdots & (t-s+1)^{q-1} \end{bmatrix}. \qquad 2.29$$

The other relevant type of basis function is autoregressive. In line with our notation, an autoregressive model of order *q*, written as AR(*q*), refers to

$$y_t = \beta_0 + \beta_1 y_{t-1} + \ldots + \beta_q y_{t-q} + \varepsilon_t \qquad 2.30$$

In this case our explanatory variables are the previous values of the time series, and an increase in order corresponds to the lag of these variables. So we have

$$\begin{bmatrix} 1 & y_{s-1} & y_{s-2} & \cdots & y_{s-q} \\ 1 & y_s & y_{s-1} & \cdots & y_{s-q} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & y_{t-1} & y_{t-2} & \cdots & y_{t-q} \end{bmatrix} \qquad 2.31$$

as our matrix *G*, with the exception that if any subscripted value in 2.31 is less than one (for example, if we need to find y_{-3}), then we use zero for that value.

2.5 Choice of Hyperparameters

In the next two sections we look at some adaptations to improve the above algorithm.

One problem that we face is that we have a number of hyperparameters to decide, before we can run the algorithm, and the algorithm is quite sensitive to some of these. In this section we look at some methods for deciding on what these values should be.
The vector of the hyperparameters we require is

$$\Theta = \left(\lambda, \nu, \gamma, \delta_1^2, \delta_2^2, \ldots, \delta_{\bar{p}}^2\right) \qquad 2.32$$

The hyperparameters ν and γ, when halved, are parameters from an inverse gamma distribution, which is a prior for σ², the variance for our error term in equation 2.1.

To calculate these we first need an estimate of ε, which we obtain by calculating a local linear approximation of the data. We subtract this approximation from the data, leaving us with the residuals, from which we subtract their mean, to ensure that the residuals have zero mean. We then bootstrap by taking a large number of samples with replacement (usually 1,000) from these residuals, each of length n. We calculate the variance of each of these samples, and this gives us 1,000 data points, which we treat as samples from the distribution of σ^2. We then use the mean and variance of this sample as the mean and variance of our inverse gamma prior, and hence find v and γ.

Given that our prior for β_i is normal, with mean zero and variance $\sigma^2\delta_i^2$, we find an estimate for δ_i. We note that β_i is unlikely to be more than $3\sigma\delta_i$. Thus we wish to choose δ_i such that $3\sigma\delta_i$ is an upper bound for all possible values of β_i that we wish to consider, and use this upper bound to find δ_i. Clearly this method requires using a specific value for σ, so for this we use the mode of the inverse gamma distribution as found in the previous paragraph.

This method of finding δ_i merely changes the problem from specifying a value of δ_i to that of specifying an upper bound for allowable values of β_i, but this seems a simpler method of using whatever prior information we have before we analyse the data. Alternatively, if we have none, then it is usually possible to find appropriate values for these upper bounds from a visual inspection of the data.

For example, when we look at AR models of order one, we can be sure that the AR regression coefficient β would be less than one, and so we could use this as our default upper bound if we have no other prior information.

The hyperparameter λ is the parameter of a geometric prior on the number and position of the changepoints. As such, λ can be treated as an estimate of the probability of each point being a changepoint. Clearly, before our analysis we don't know the number of changepoints, and so can't estimate this probability. For this parameter, we usually require some a priori information regarding the number or likelihood of changepoints, such as a previous analysis, or some expert opinion on the data. Failing this, it is possible to introduce a hyperprior distribution on λ, thus removing the need to guess a value. For

example, if we choose a uniform prior on λ, then we require no input value for our geometric prior. The problem with this type of solution is that it creates a dependence between the prior distributions for different segments, which violates an assumption we made in section 2.2.

It should be noted that if any specific knowledge is known about the source of the data *a priori*, this information should be used preferentially to the methods above.

2.6 Peak finding

One problem we note with the method is that on different iterations the algorithm may find the same changepoint in different locations, which can result in a range of values, each of which has a small posterior probability of being a changepoint, when in fact there is only one true changepoint. We call this situation leakage, due to its similarity to spectral leakage.

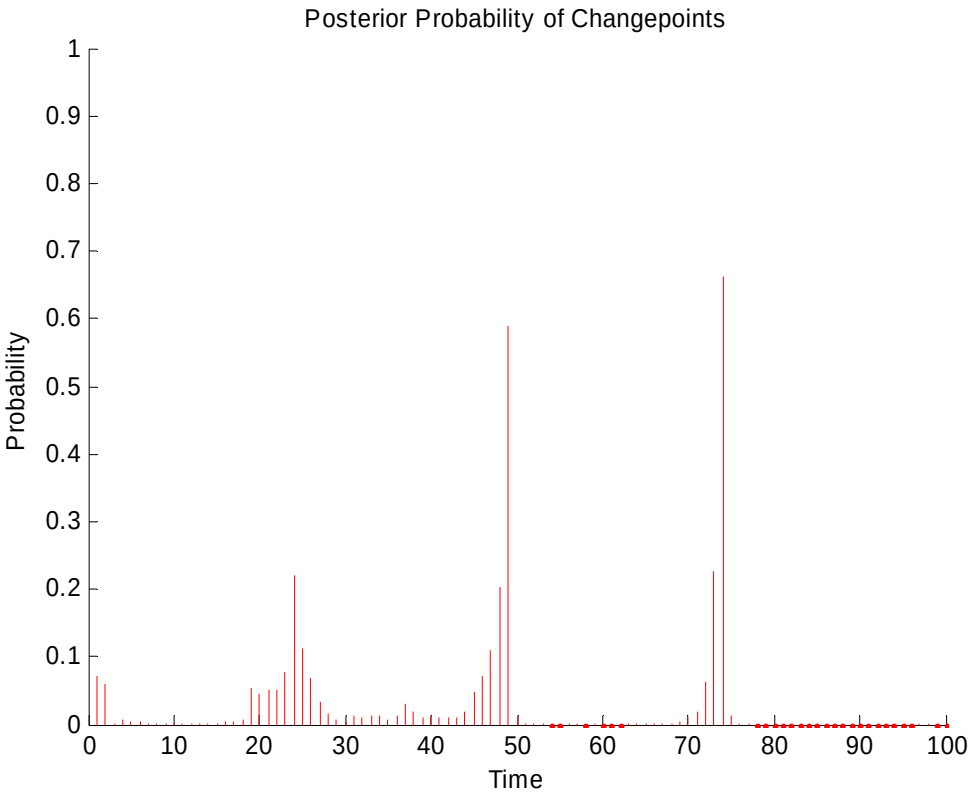

Figure 2.1 Posterior probabilities of changepoints

In figure 2.1 we have the results of Fearnhead's algorithm on a simulated data set with true changepoints at 25, 50 and 75. We see that many of the points around the true changepoints have small probability values, resulting in the method finding changepoints near the true ones in different iterations, and this makes the probability values at the true changepoints less that it would otherwise be.

To avoid this problem we need to combine all of the nearby changepoints into one. To find all of the values we wish to consider as changepoints, and not merely artifacts of being near a changepoint, we look at all of the local maxima of the vector of posterior probabilities. That is, if a time point has a higher posterior probability of being a

changepoint than the time points immediately before and after it, we consider it as a changepoint, as these points are the ones we find most likely to be changepoints.

Having found the allowable changepoints, we then need to accumulate the probabilities of the nearby points into the correct locations. We define 'nearby' by finding the midpoints between two local maxima, or peaks, and then group all time points from one midpoint to the next as associated with the peak in that group. (For the first peak, we associate all of the values less than the first midpoint with it as the first group, and do similarly with the last group.)

An alternate method of doing this grouping would be to create a window of a certain length around each peak, and group all time points in this window with the relevant peak. Then we would leave all values not in any group as they were. This may be useful for certain applications, particularly where it is important to know the exact position of changepoints accurately.

Having grouped all of the time points with exactly one peak in each group, we can now combine the probabilities in each group to its associated peak. We note that we cannot simply sum all of the probabilities in the group; if any iteration found more than one changepoint in a given group, then summing may give us a result of more than one. To eliminate this problem, we then look at each iteration and count the number of simulated changepoints in each group. For each iteration, if we find at least one changepoint in a given group, then we increment the count of simulated changepoints at that group's peak

by one, irrespective of how many changepoints found in that group on that iteration. All time points that are not peaks have a zero count of changepoints. We call this our peak finding algorithm.

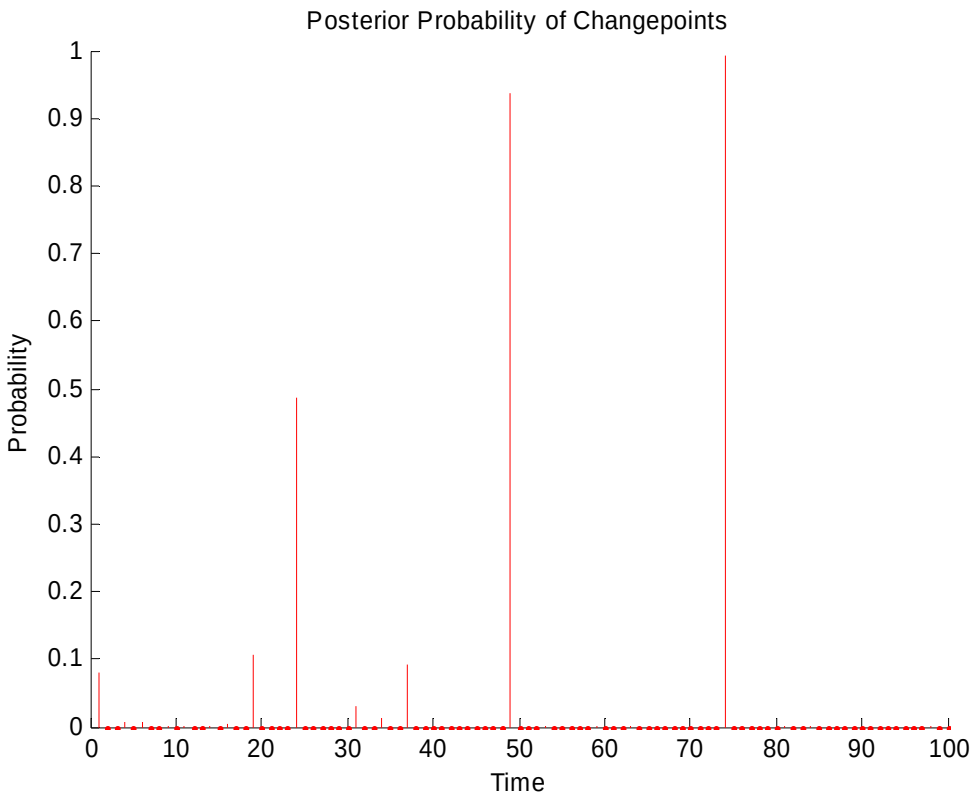

Figure 2.2 Posterior probabilities of changepoints, with peak finding.

Figure 2.2 shows the same results as those in figure 2.1, having had our peak finding method run on them, showing the changepoints we would expect with much higher posterior probabilities.

We can also see in figure 2.2 that there are some points that have a low probability of being changepoint, where we know that there are no true changepoints. This is an artifact

of our peak finding algorithm; if the original analysis has a large amount of leakage, then we may have points nearby that are themselves local maxima of posterior probabilities, and thus will be a peak. If this occurs, we would expect that some non-changepoints will accumulate significant probabilities in our peak finding algorithm. In most cases however, these will have lower probabilities than the true changepoints.

This peak finding algorithm allows us to make more accurate probability statements about the existence of changepoints.

2.7 Summary

This chapter details an algorithm for finding the changepoints in a time series. The algorithm uses a Bayesian approach to the changepoint problem, using perfect simulation to sample directly from the posterior distribution of changepoints and, having found the changepoints, to calculate the distribution of model order in each segment.

Two adaptations to the algorithm are made; the first of these is a way to use the data to find accurate values for most of the hyperparameters, which is important as the algorithm can be sensitive to these values. The second is the peak finding algorithm, which fixes the leakage problem by grouping nearby changepoints into one place, which we found greatly improved the probability of the true changepoints being found as such.

Chapter 3

Simulations

3.1 Introduction

In this chapter we will use the algorithm on some specifically simulated data, as a means of assessing the quality of the method. To test the algorithm we generate data with known change point positions, with various signal to noise ratios, model types and model orders to see how effectively the algorithm finds the correct segments and model orders.

3.2 Setup

We look at simulated data with changepoints at known locations, and run our algorithm on this data, to see how well it finds the known changepoints under different circumstances, such as different model types, model orders and signal to noise ratios.

We look at data that is polynomial of different orders (constant, linear and quadratic), and AR data with different levels of signal to noise ratios, to test how well our algorithm finds the changepoints under these various circumstances.

For each simulation we take 10,000 perfect simulations from the posterior distribution of changepoints. We use the method of finding hyperparameters discussed in section 2.5, which gives all of the values required except for λ, for which we can simply use the true proportion of changepoints.

3.2.1 Receiver Operating Characteristic

Receiver operating characteristics (ROCs) are used as a measure of how well our algorithm performs, and to make comparisons between different analyses on the same data. An ROC is a plot of the probability of a type one error against one minus the probability of a type two error. Here we treat the algorithm as a test to see if each point is a changepoint, so our null hypothesis for each point is that it is not a changepoint, and our alternate is that it is. Thus

$$\alpha = \Pr(\text{TI error}) = \Pr(\text{We find a change point} \mid \text{No true change point}) \quad 3.1$$

$$\beta = \Pr(\text{TII error}) = \Pr(\text{We find no change point} \mid \text{A true change point exists}) \quad 3.2$$

$$\gamma = 1 - \beta = \Pr(\text{We find a change point} \mid \text{A true change point exists}), \quad 3.3$$

so our ROC curve is a plot of α on the x axis against γ on the y axis. We note that a perfect test has unit area under the ROC curve, and a test that involves random guessing

has an area of 0.5. Thus we can use the ROC to see how powerful our test is, by seeing how close the area under it is to one, and to compare two methods on the same data, by comparing the area under the ROC curve of the two methods.

The values used in the ROC are calculated by running the algorithm a large number of times (k), where the data used in each of these runs has the same model and noise distribution, but the actual noise values are resampled from their common distribution. Each of these k iterations gives us a vector of probabilities for each point being a changepoint, which ensures a large sample of vectors of posterior probabilities of changepoints with which to create our ROC. For each iteration we then look at a large number (m) of cutoff probabilities, equally spaced from zero to one. We find the time points in our current vector of posterior probabilities that are larger than each of these cutoff probabilities, and take these to be the changepoints corresponding to the particular iteration and cutoff probability currently being looked at. We can then compute two $k \times m$ matrices, one whose entries are the values of α, which we call T_1, and the other whose entries are values of γ, which we call T_2. That is, for the ith vector of posterior probabilities of changepoints, and the jth cutoff probability

$$T_1(i,j) = \frac{\text{The number of false change points we find}}{\text{The number of points that are not true change points}} \qquad 3.4$$

and

$$T_2(i,j) = \frac{\text{The number of true change points we find}}{\text{The number of true change points}}. \qquad 3.5$$

Having computed these, each column is then averaged, thus for each value of cutoff probability, there are two probabilities (one from each matrix) that uses the information from all of the iterations of our algorithm that we ran. So we now have two vectors of length *m* with averaged probabilities, which we call U_1 and U_2 respectively, and can then plot *m* points on a scatter plot, with the x axis value coming from U_1 and the y axis value being the corresponding value from U_2. Typically we use $k = 200$ and $m = 1000$.

A problem we have with using the ROC as described for our simulations is that the following cases only have 2-3 changepoints, and so there is a severe imbalance between the number of changepoints and non-changepoints. Since the denominator of equation 3.4 is so large in comparison to that of equation 3.5, most of the points in our ROC have comparatively low values of α, and so are mostly on the left of the curve. This tends to bias the area under the ROC curve towards one, and so is not a very useful measure of performance. To counter this, we use a modification of the ROC: we use the vector U_3 instead of U_1, where

$$U_3 = U_1^{\frac{\log(\text{number of change points})}{\log(\text{number of non-change points})}}. \qquad 3.6$$

The vector U_3 in equation 3.6 is shifted towards what would be expected if there were an equal number of changepoints and non-changepoints. That is, if there are more non-

changepoints than changepoints (as is usually the case), then U_3 will be larger than U_1, and if there are the same number then equation 3.6 does not change U_1.

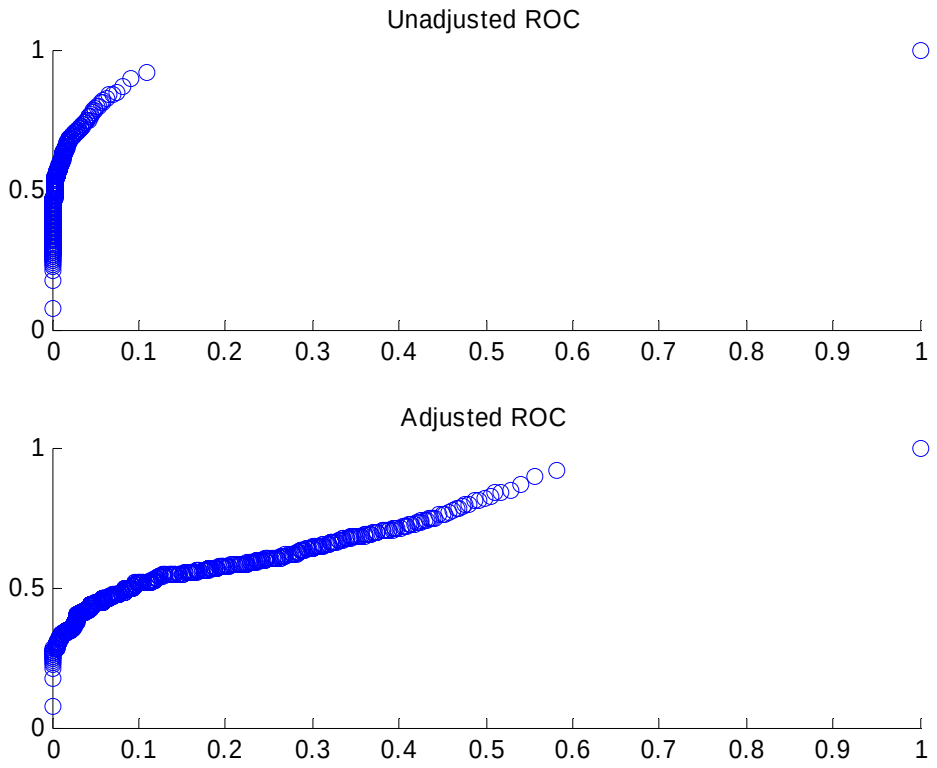

Figure 3.1 Comparison of adjusted and unadjusted ROCs.

Figure 3.1 shows examples of both the unadjusted (top) and adjusted (bottom) ROC curves for a dataset with two true changepoints on which the algorithm performed poorly. The unadjusted ROC curve seems to imply an accurate assay, which in this case is unjustified, where the adjusted ROC curve shows the errors more accurately.

The area under the ROC curve is estimated using a trapezoidal approximation.

Another issue is that sometimes our algorithm finds a changepoint at a location very near, but not exactly at, the location of a true changepoint. If this occurs then it can drastically effect the ROC, since it can mean finding a false changepoint with high probability (when in fact the changepoint found is very near the true changepoint), and not finding the true changepoint at all. The peak finding algorithm described in section 2.6 can greatly amplify this problem. To counter this effect, we introduce a tolerance on the position of true changepoints, so that any 'nearby' points that are found to be changepoints are counted as a true changepoint that we have found, as in the numerator for equation 3.5. A tolerance of two usually works well.

3.3 Polynomials

The first class of examples we will look at is that of polynomials. In this scenario a change point is a point where at least one of the coefficients of the underlying polynomial model changes. We will look at the accuracy of the algorithm for data sets where the model is constant, linear and quadratic.

3.3.1 Constant

The first case we look at is a polynomial of order one, i.e. a constant function (using the convention mentioned in section 2.4). In this case, a change point represents a point where the value of the constant changes. In our first simulation, we have a function with values (-1, 1, 3, 10), with a noise term that comes from a N(0,0.25) distribution (where we take $N(\mu,\sigma^2)$ to refer to a normal distribution with mean μ and variance σ^2).

$$y_i = N(0,0.25) + \begin{cases} -1 & 1 \le i \le 25 \\ 1 & 26 \le i \le 50 \\ 3 & 51 \le i \le 75 \\ 10 & 76 \le i \le 100 \end{cases}. \qquad 3.7$$

This gives us quite a strong signal to noise ratio.

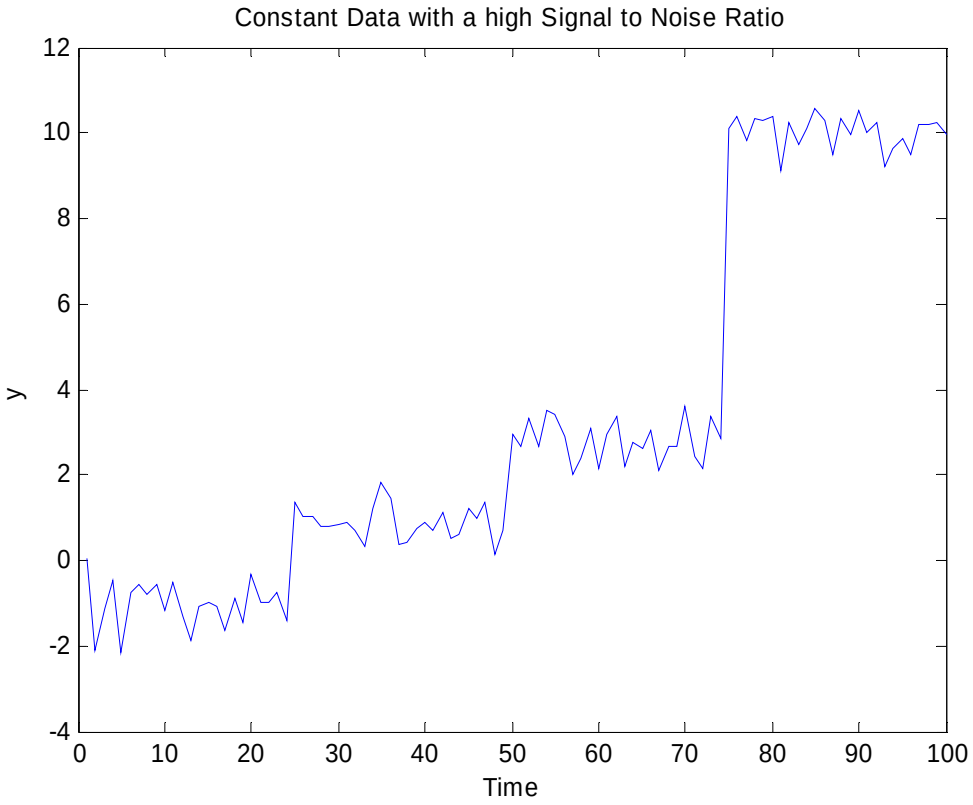

Figure 3.2 Data with constant segments

In figure 3.2 it is very obvious from a visual inspection to see where the changepoints occur, and we expect our algorithm to find them with little error.

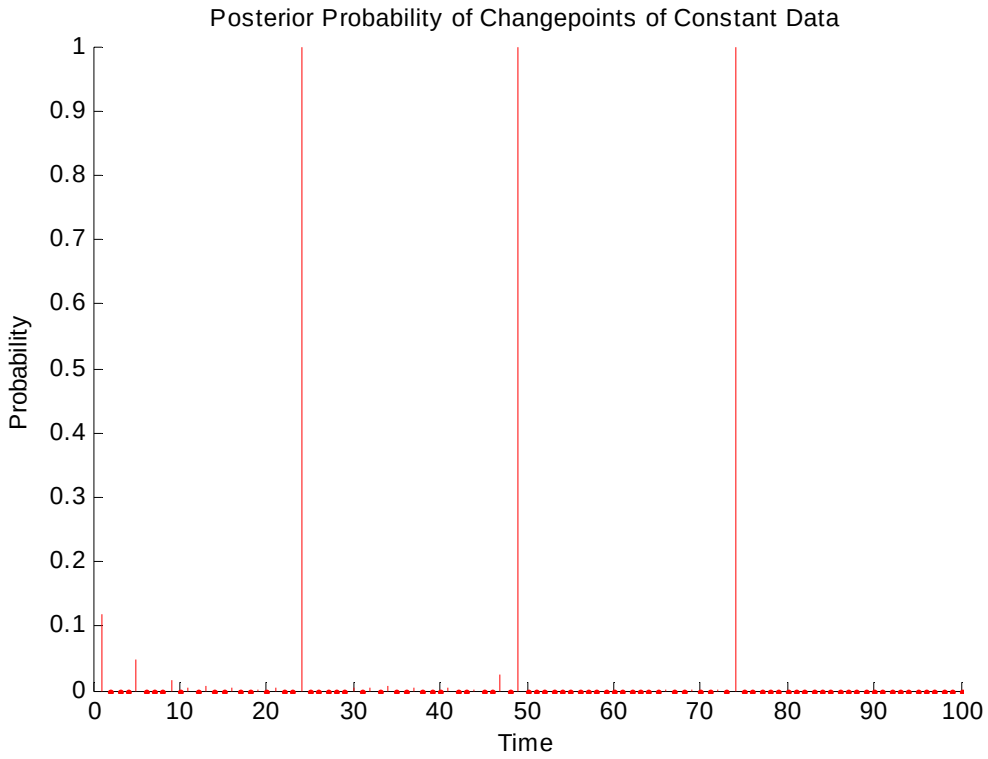

Figure 3.3 Posterior probability of changepoints for the data in Figure 3.2

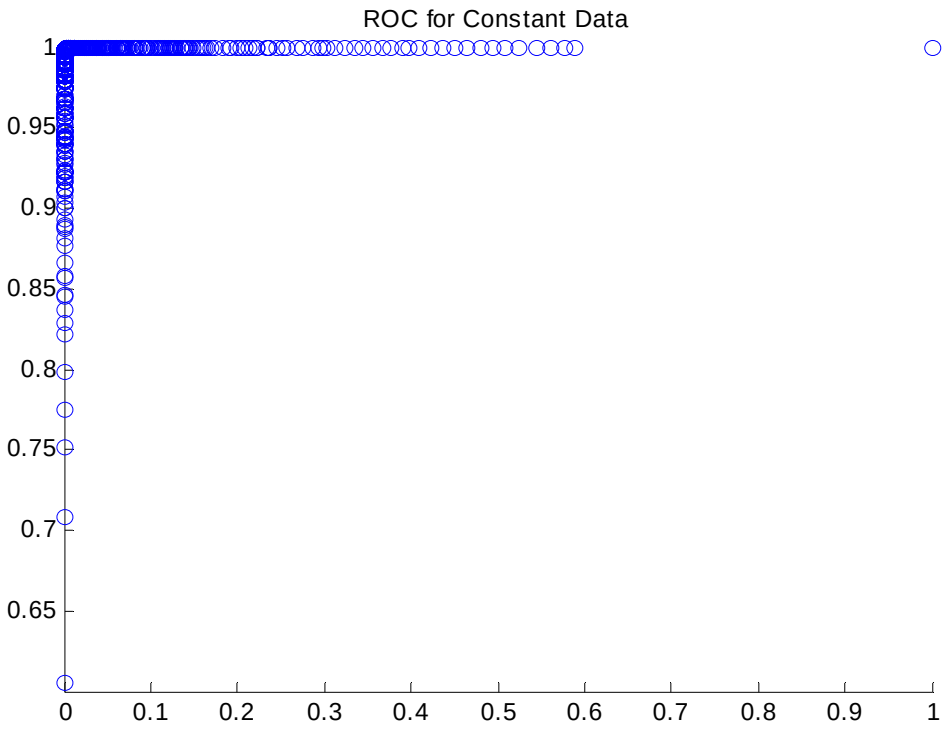

Figure 3.4 ROC for constant data. Area=1

As expected, Figures 3.3 and 3.4 show that the algorithm finds no error in identifying the correct changepoints.

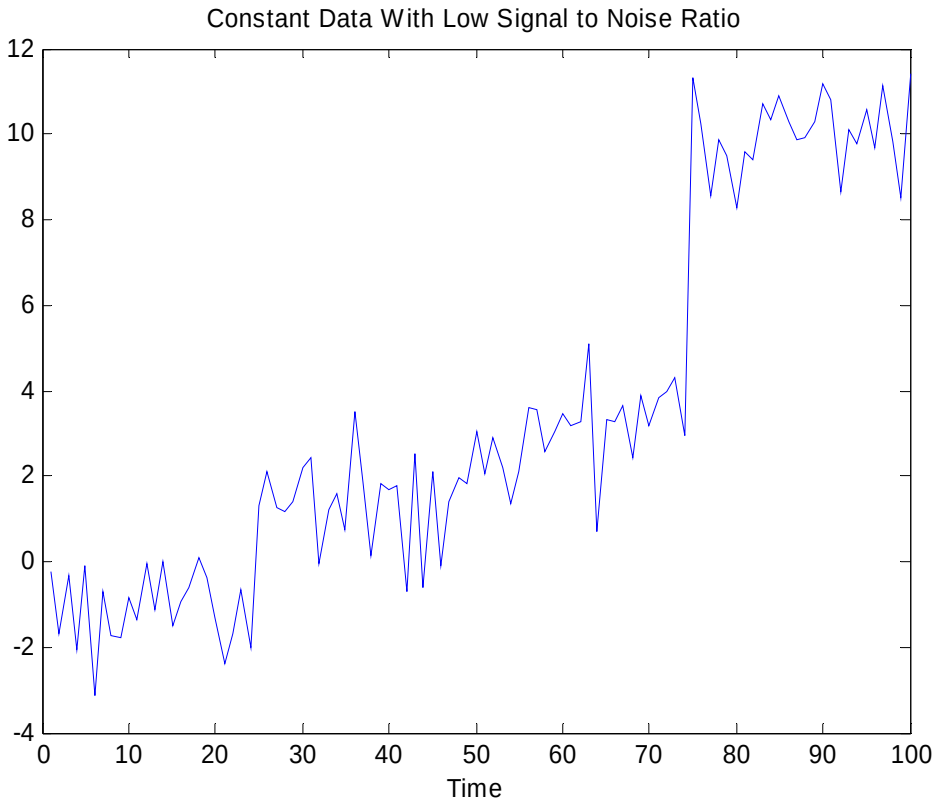

Figure 3.5 Data with constant segments, with a low signal to noise ratio

In Figure 3.5 we use the same data as in Figure 3.3, but have increased the noise variance to 1, reducing the signal to noise ratio.

$$y_i = N(0,1) + \begin{cases} -1 & 1 \le i \le 25 \\ 1 & 26 \le i \le 50 \\ 3 & 51 \le i \le 75 \\ 10 & 76 \le i \le 100 \end{cases}. \qquad 3.8$$

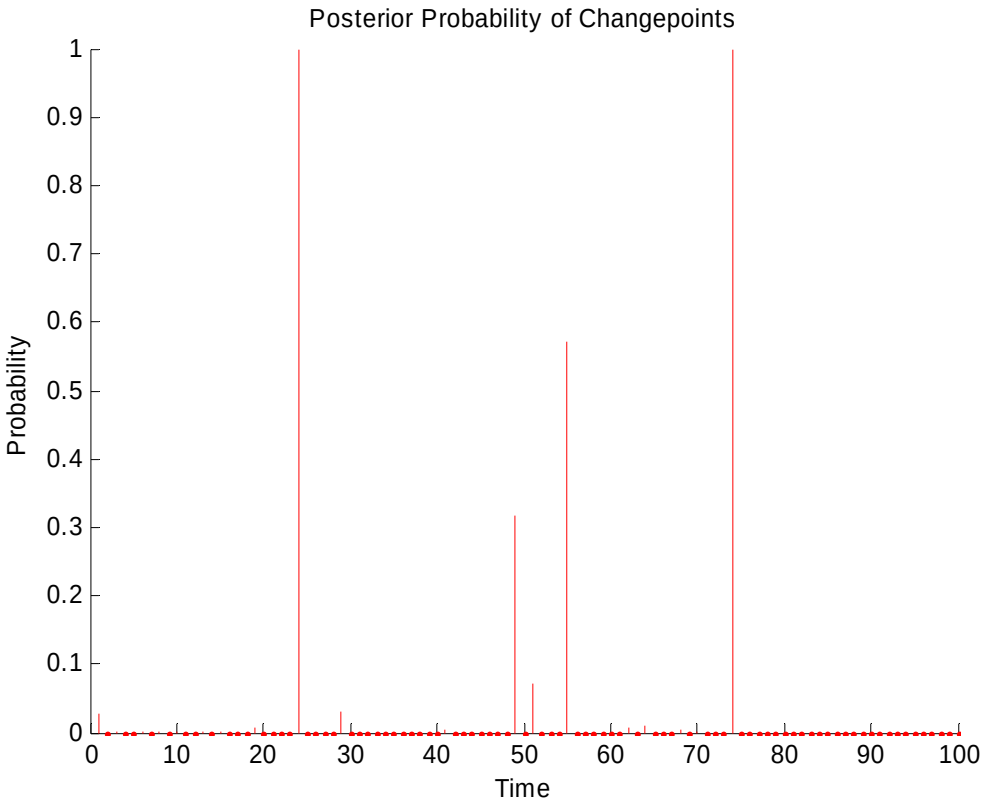

Figure 3.6 Posterior probability of changepoints for the data from Figure 3.5

Figure 3.6 shows that the last changepoint is found with unit probability, as expected, since the step size is so great that the additional noise has no effect. The posterior probability for the second changepoint is quite small, with some leakage to nearby points, (despite the peak finding algorithm), which we expect due to the small signal to noise ratio. Interestingly, we find the first changepoint certainly, despite the step size being the same as that of the second changepoint. This is the result of random variability; for another similarly generated data set we may find the probabilities for the first and second changepoint reverse, as in figure 3.7.

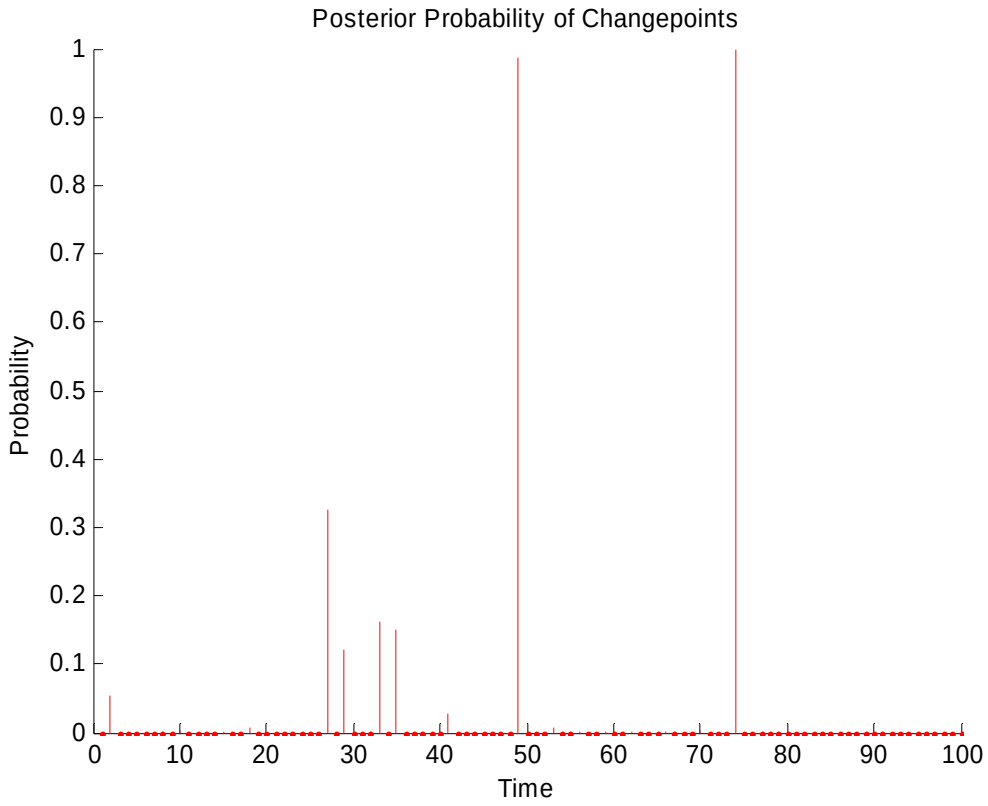

Figure 3.7 Similar result to Figure 3.6, for data with a different noise sequence

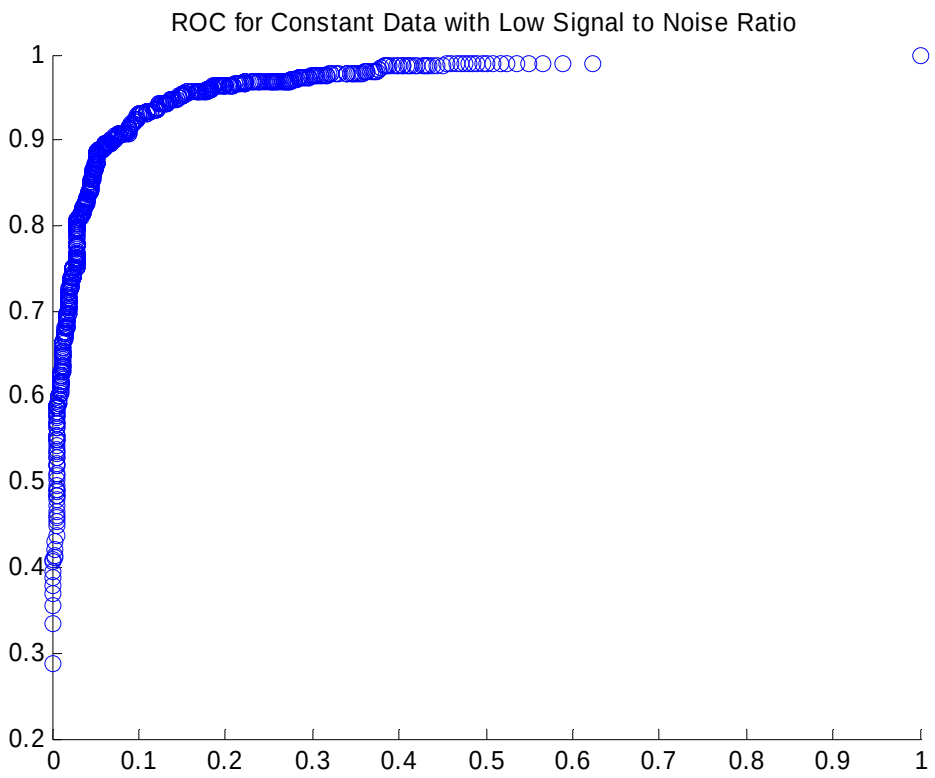

Figure 3.8 ROC for constant data with low signal to noise ratio. Area=0.9664

The ROC in figure 3.8 has a lower area than the previous ROC in Figure 3.4; a consequence of the increased noise causing the algorithm to miss some changepoints as the decreased signal to noise ratio makes analysis more erroneous.

3.3.2 Linear

Data sets with linear segments are investigated in this section. The first time series we look at has three segments, with linear coefficients that are positive, zero and negative respectively with a high signal to noise ratio.

$$y_i = N(0,0.0025) + \begin{cases} \dfrac{i}{25} & 1 \le i \le 25 \\ 1 & 26 \le i \le 65 \\ \dfrac{165-2i}{35} & 66 \le i \le 100 \end{cases}. \qquad 3.9$$

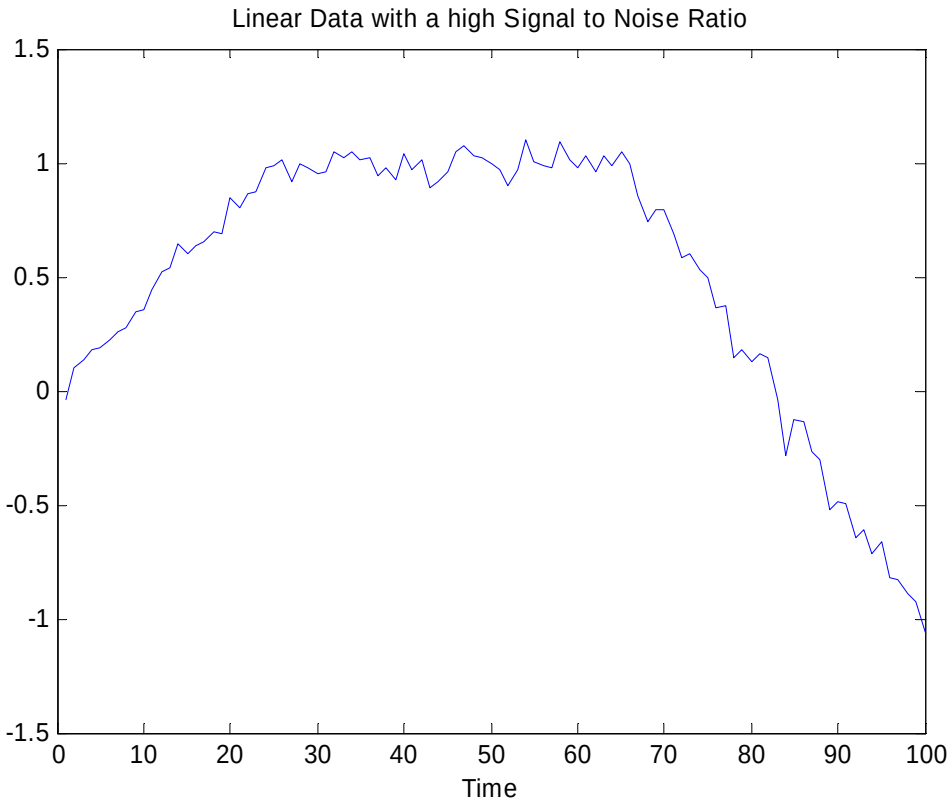

Figure 3.9 Data with linear segments

Linear data with a high signal to noise ratio is illustrated in Figure 3.9, where changepoints exist at times 25 and 70. The low noise level and high signal produced make the changepoint locations visually clear.

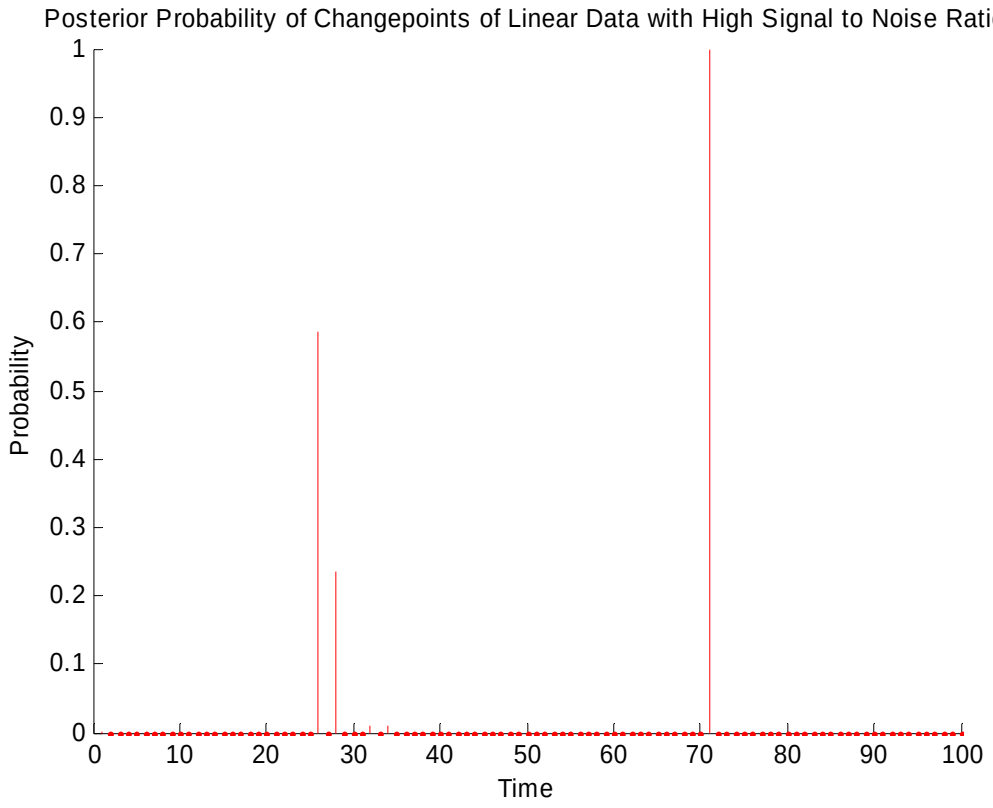

Figure 3.10 Posterior probability of changepoints for the data from Figure 3.9

The results in Figure 3.10 show what we would expect: we find the second changepoint with higher probability, since the change in slope is greater at that changepoint, yet still find the first with a high probability. We also notice some leakage into a nearby point, which is an artifact of our peak finding algorithm, as discussed in section 2.6.

The results in Figure 3.10 show what we had anticipated, the second changepoint found with high probability, since the change in slope is greater at that changepoint, yet still we find the first changepoint with a high probability. We also notice some leakage into a nearby point in the first changepoint.

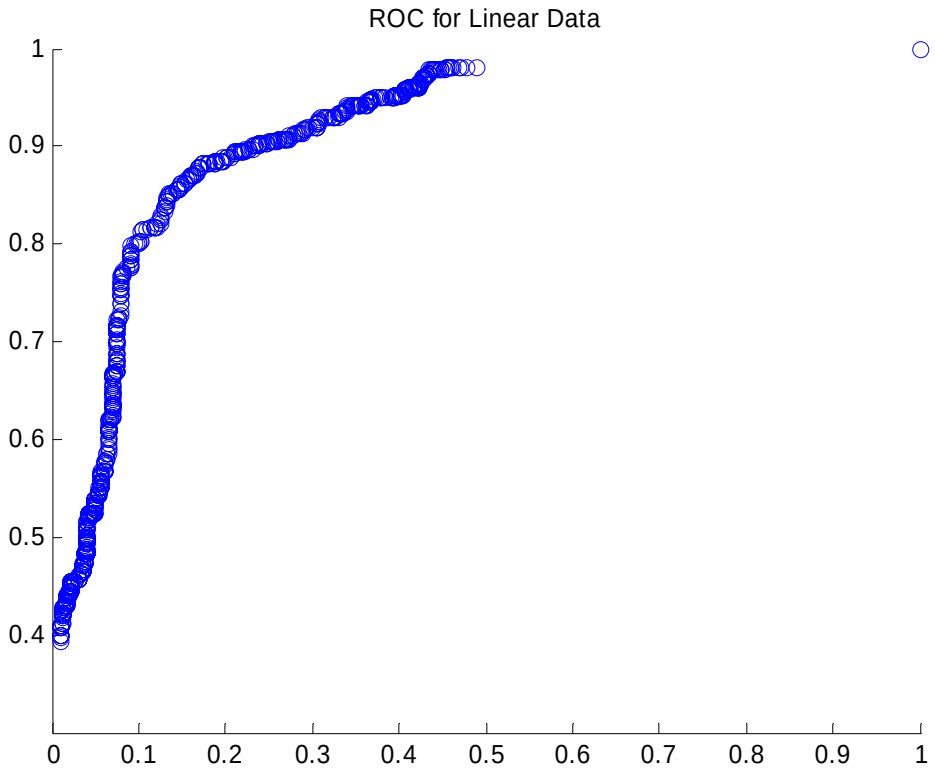

Figure 3.11 ROC for linear data. Area=0.9151

The ROC in Figure 3.11 shows a decline in accuracy of the algorithm from the constant models. This seems to indicate that the algorithm performs worse with higher order models.

Since we are now looking at data whose model has more than one parameter, we can use equation 2.23 to find the posterior distribution for model order in each segment. As mentioned in section 2.3.2, this requires setting a cutoff probability. Unless otherwise stated, we use the value p_c=0.5 throughout.

Segment	Constant	Linear
1	0	1
2	0.989	0.011
3	0	1

Table 3.1 Posterior distributions for model order

Table 3.1 shows the posterior distribution of model order for each segment; there is almost no error in these results.

In the next case, similar data is examined, with a lower signal to noise ratio.

$$y_i = N(0,1) + \begin{cases} i & 1 \le i \le 25 \\ \dfrac{7i+75}{10} & 26 \le i \le 50 \\ 42.5 & 51 \le i \le 75 \\ \dfrac{1700-17i}{10} & 76 \le i \le 100 \end{cases}. \qquad 3.10$$

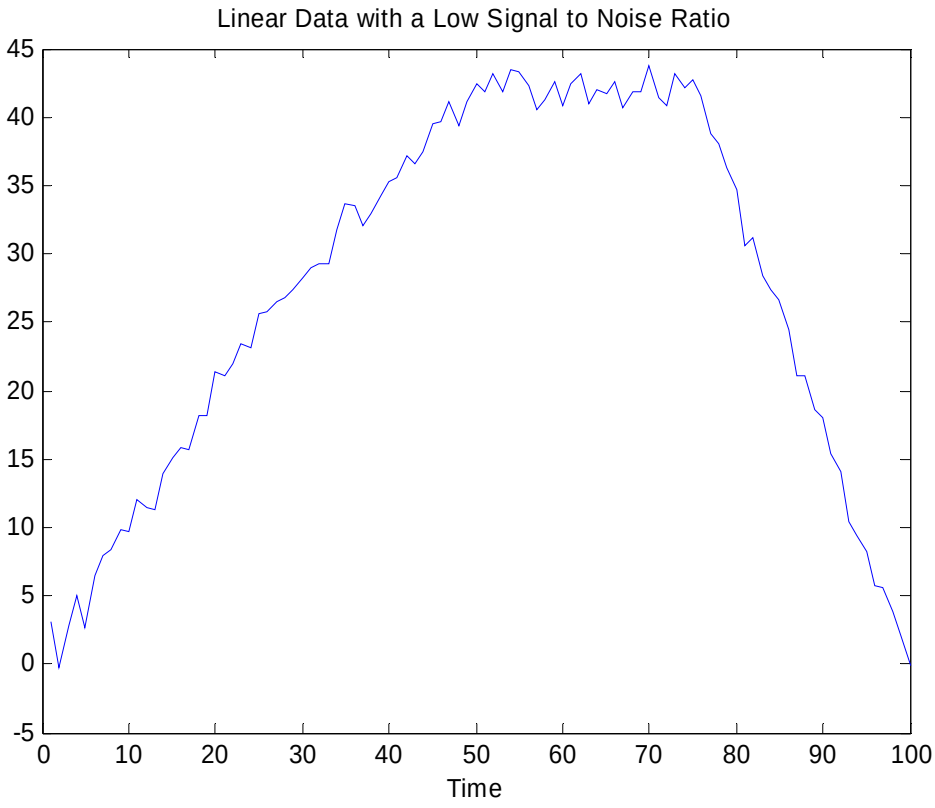

Figure 3.12 Linear data with a low signal to noise ratio

From looking at Figure 3.12, we can clearly see the changepoints at times 50 and 75, however the changepoint at time 25 is not quite so obvious, due to the slight change in slope (from 1 to 0.7), and the relatively large noise term.

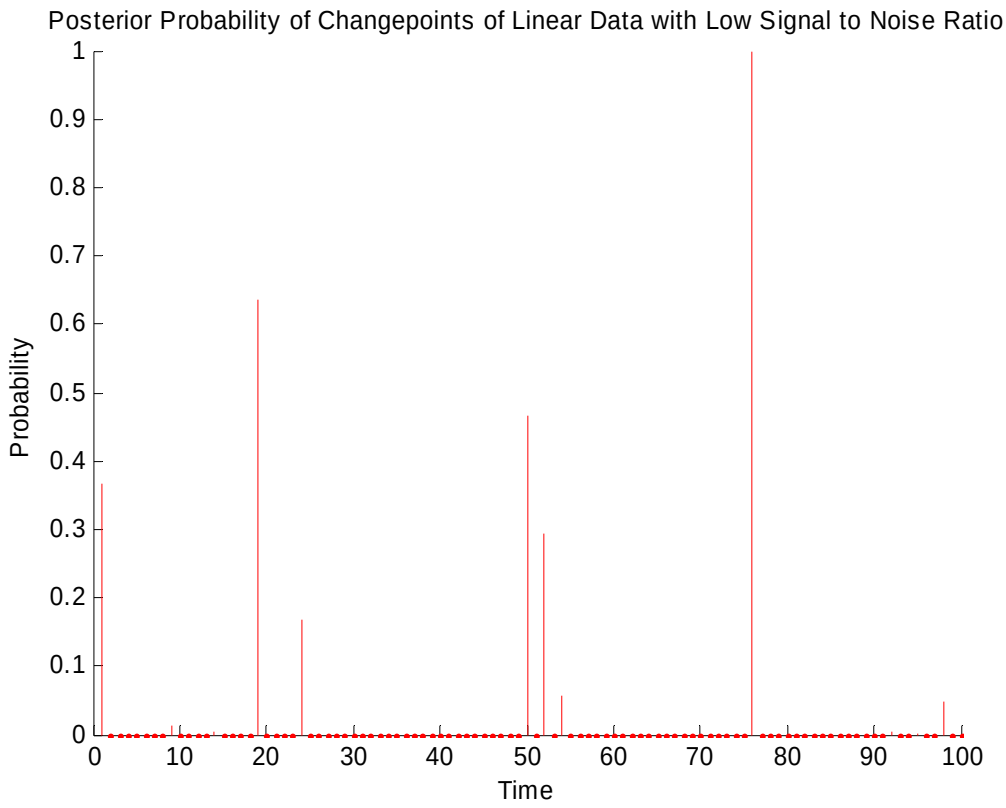

Figure 3.13 Posterior probability of changepoints for the data from Figure 3.12

In the results of our algorithm, however, we see that it finds this first changepoint with quite a high likelihood, as well as successfully identifying the last one, as shown in Figure 3.13. The middle changepoint, however, doesn't seem reach the same probability values as the other two, as it suffers from quite severe leakage.

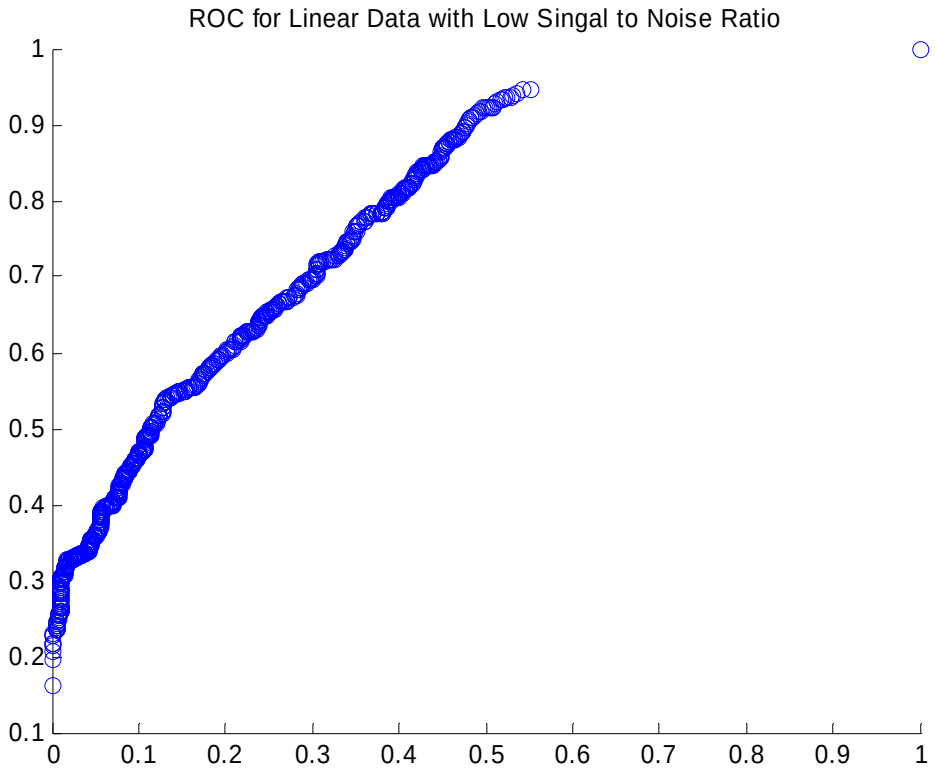

Figure 3.14 ROC for linear data with low signal to noise ratio. Area=0.8024

Figure 3.14 shows a significant decrease in area from Figure 3.11, as the larger noise causes more leakage, causing more false changepoints to be found while missing real ones.

To correctly calculate the posterior distribution for model order, the cutoff value p_c needs to be lowered from 0.5 to include the middle changepoint in this case.

Segment	Constant	Linear
1	0	1
2	0	1
3	0.9729	0.0271
4	0	1

Table 3.2 Posterior distributions for model order

Table 3.2 displays the distribution of model order for each segment; interestingly, the only segment with any error is the lower order one, as in Table 3.1.

A property of the algorithm that we wish to measure the effect of is the peak finding algorithm. To do this, we compute an ROC for series with the model shown in Figure 3.12.

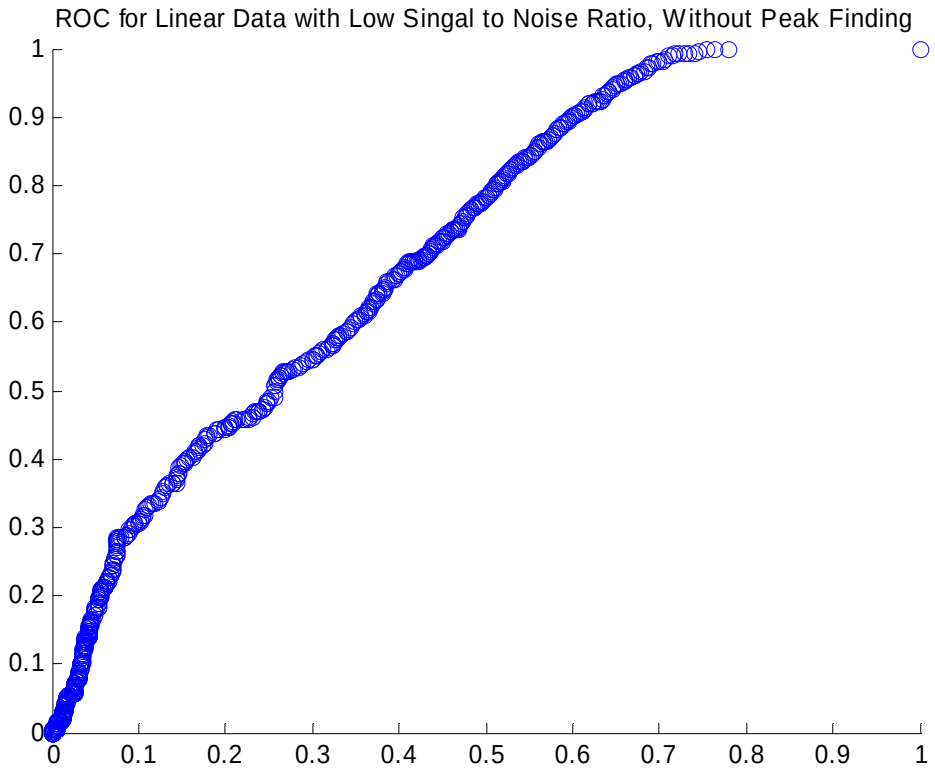

Figure 3.15 ROC for linear data without peak finding. Area=0.7150

Figure 3.15 indicates the significant difference that the peak finding method makes. Without this method, the posterior probabilities for a true changepoint are leaked into many points near the true changepoint, causing false changepoints to be found to have higher probabilities; similarly, true changepoints are found less frequently.

3.3.3 Quadratic

In this section the maximum order of the polynomial model is increased to include quadratic segments.

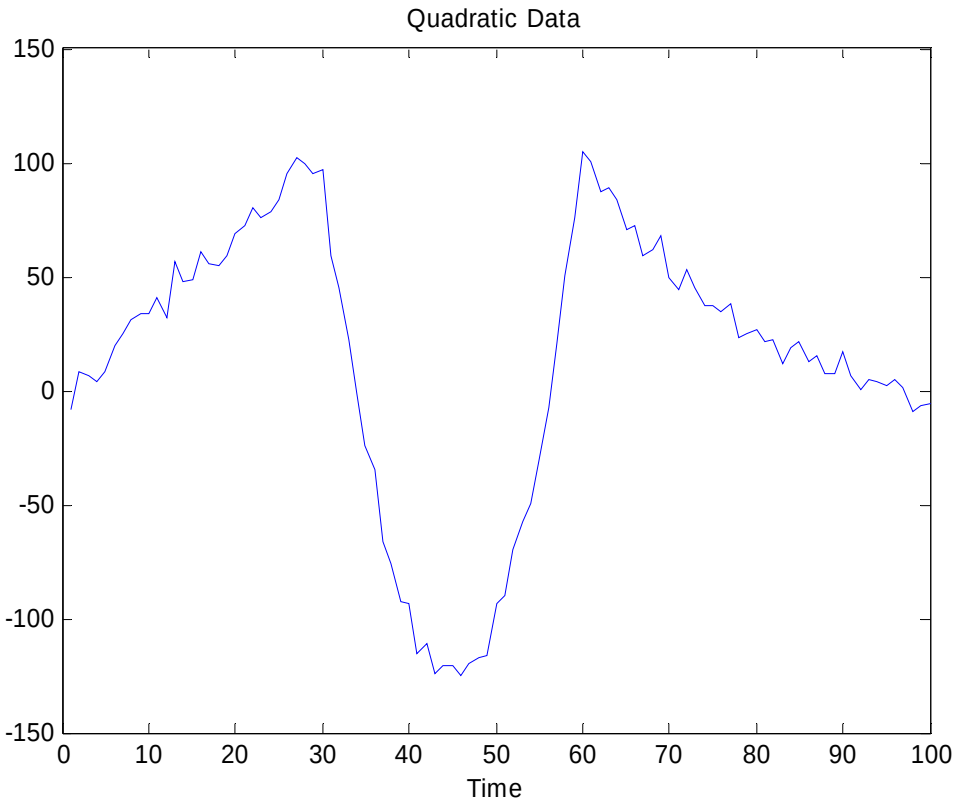

Figure 3.16 Data with quadratic segments

$$y_i = N(0,25) + \begin{cases} \dfrac{10i}{3} & 1 \le i \le 30 \\ i^2 - 90i + 1900 & 31 \le i \le 60 \\ \dfrac{i^2 - 200i + 10000}{16} & 61 \le i \le 100 \end{cases}. \qquad 3.11$$

Figure 3.16 shows the first series assayed in this section: quadratic data with a high signal to noise ratio. The first segment is linear, the other two are quadratic. The results of the analysis are presented in Figures 3.17 and 3.18.

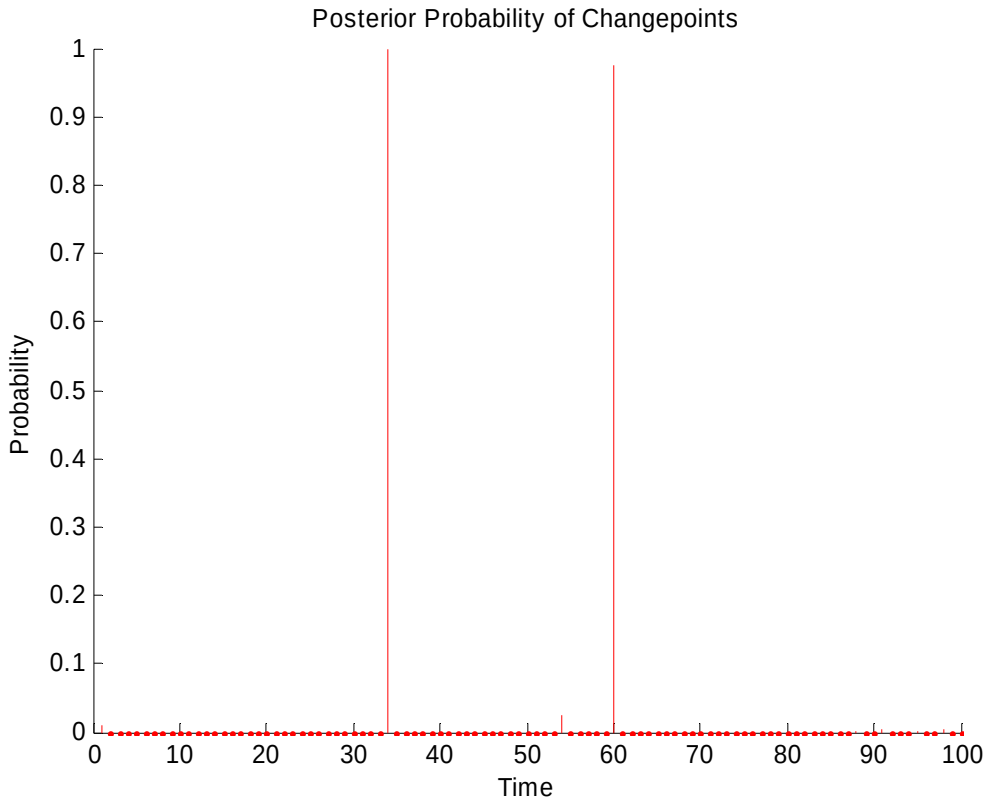

Figure 3.17 Posterior probability of changepoints for the data from Figure 3.16

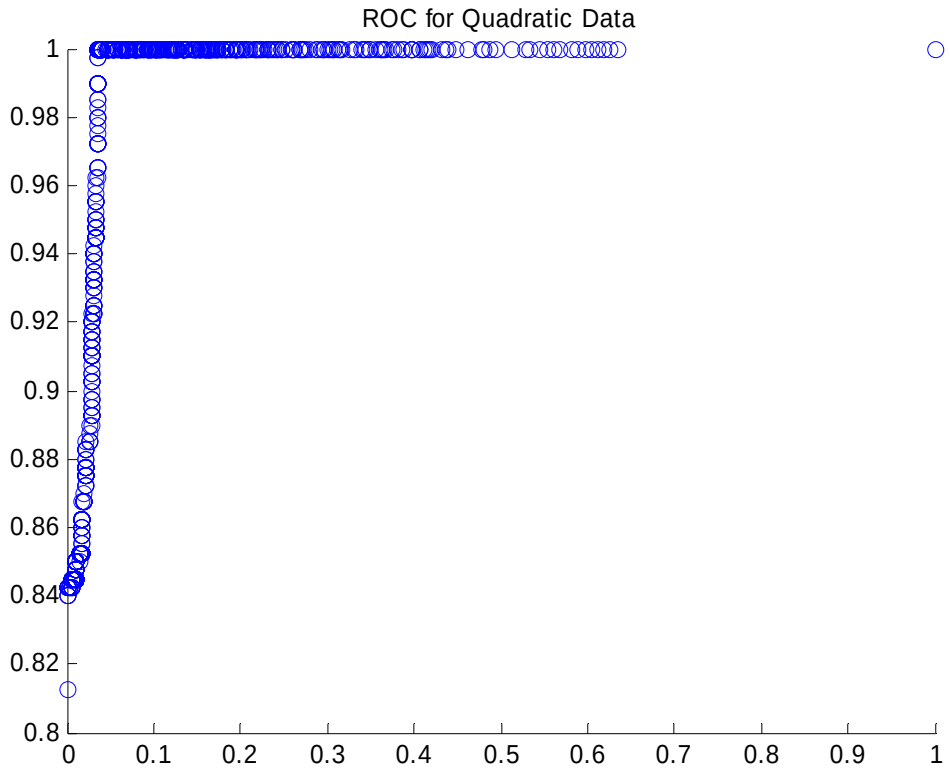

Figure 3.18 ROC for quadratic data. Area=0.9998

Figure 3.17 shows that the changepoints are found with very high probability, however the first changepoint is found at time 34, instead of 30 where it should be. Figure 3.18, however, shows very accurate results, indicating that in most of the 200 similarly generated data sets the algorithm was run on the true changepoints were found in the correct positions.

Segment	Constant	Linear	Quadratic
1	0	0.9939	0.0061
2	0	0	1
3	0.1559	0.8412	0.0029

Table 3.3 Posterior distributions for model order

Table 3.3 displays the posterior distributions of model order for each segment .An interesting point here is that the algorithm finds the third segment, which is known to be quadratic, is most likely linear or constant. This illustrates that the method for finding the distribution of model order is not very robust; segments with low quadratic terms may have very low probability of being quadratic according to the algorithm.

Next we look at quadratic data where the changepoints occur at points that are differentiable.

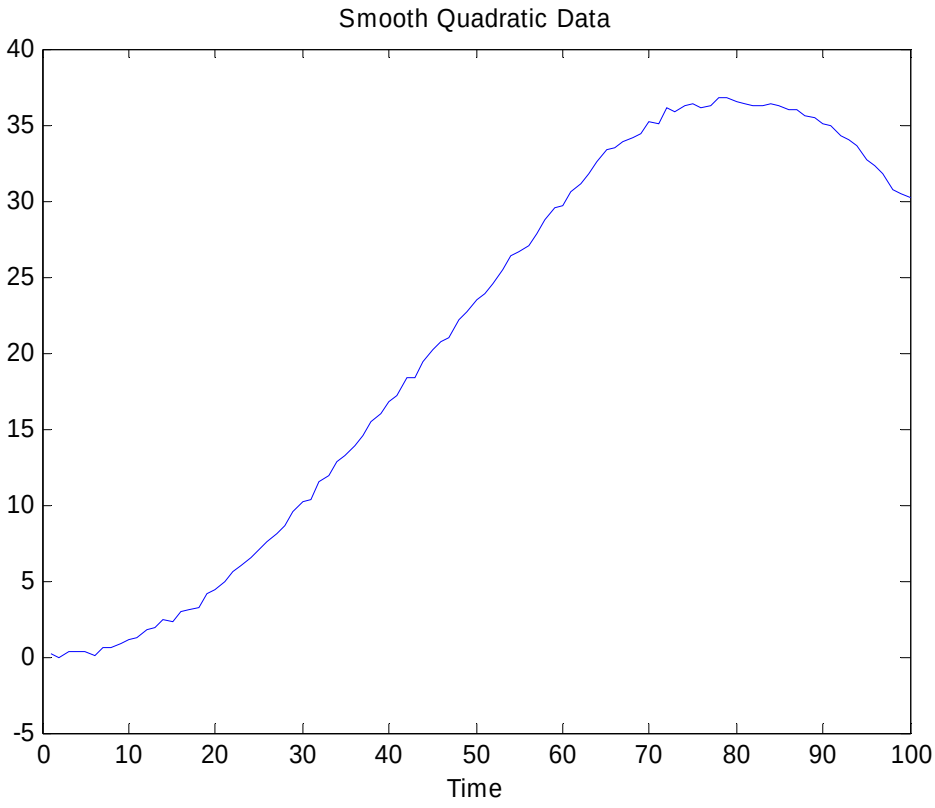

Figure 3.19 Quadratic data with smooth changes

$$y_i = N(0,0.0625) + \begin{cases} \dfrac{i^2}{90} & 1 \le i \le 30 \\ \dfrac{2i-30}{3} & 31 \le i \le 60 \\ \dfrac{-i^2+160i-4200}{60} & 61 \le i \le 100 \end{cases}. \qquad 3.12$$

In figure 3.19 the data is smooth; there are no obvious changepoints. The segment from 30 to 60, however, is linear, while the other two are quadratic. We use a very small noise here, as we are more interested in how the algorithm performs on this smooth data, and it has been shown that noisier data decreases performance.

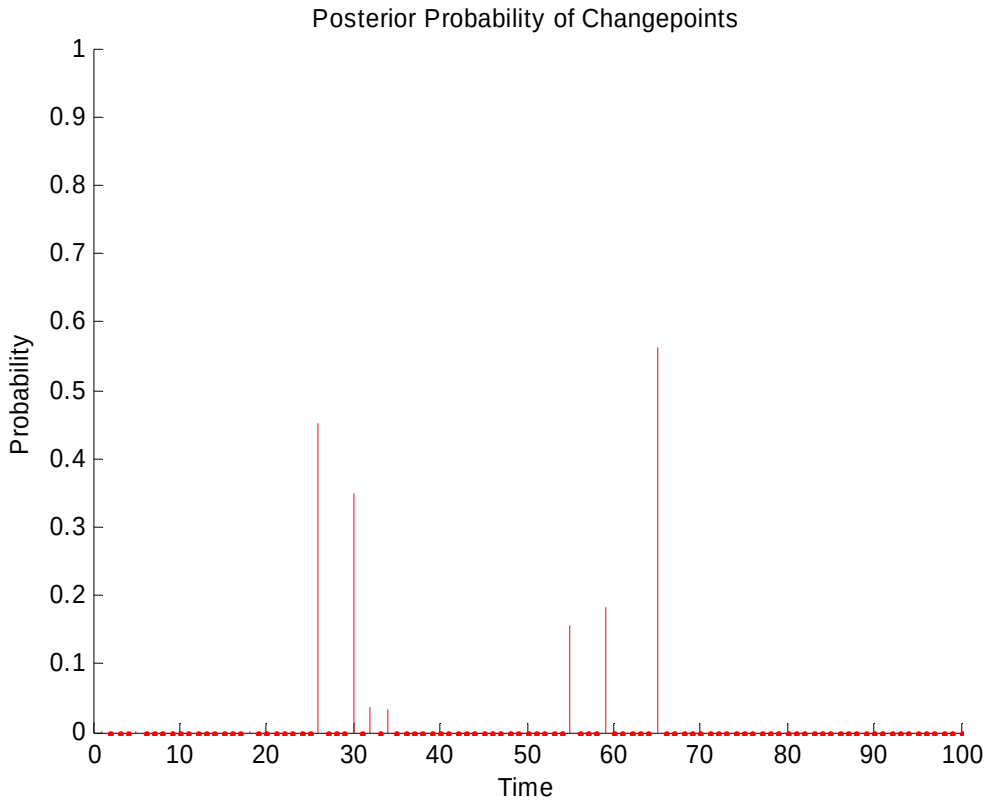

Figure 3.20 Posterior probability of changepoints for smooth data

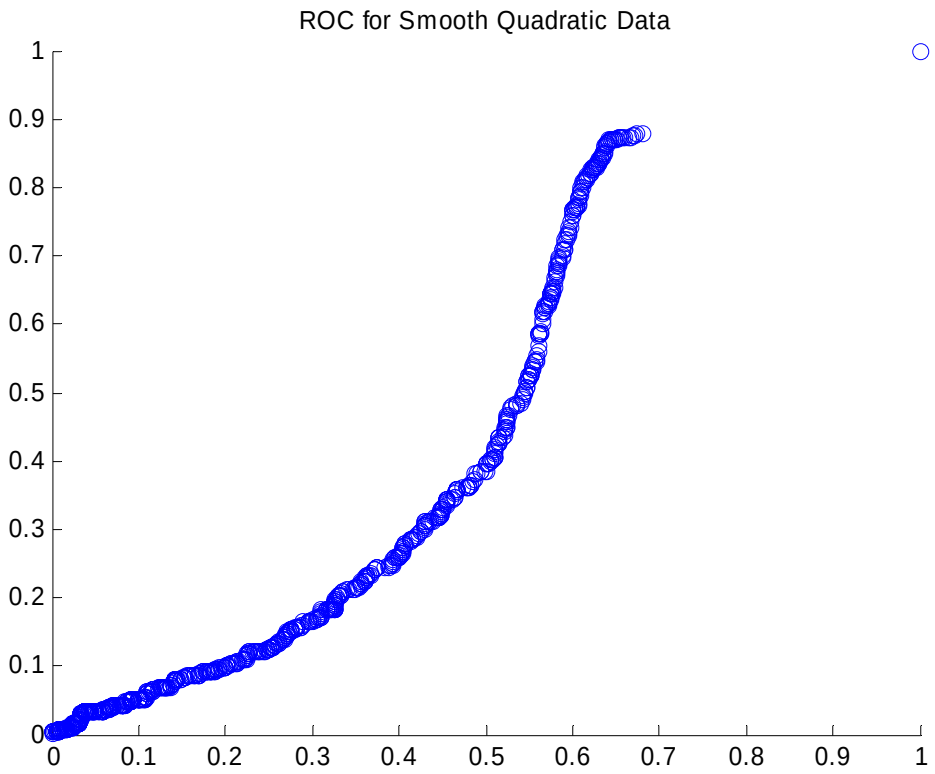

Figure 3.21 ROC for smooth quadratic data. Area=0.5014

Figure 3.20 shows that the changepoints are in locations near, but not exactly in the correct positions. Also there are low posterior probabilities, seemingly due to quite severe leakage. The ROC curve in figure 3.21 also indicates the poor performance of the algorithm on this data. From this example, and from Figure 3.12, it can be inferred that the algorithm performs better on data where the changes between the segments are sharper.

Lowering the value of p_c to include the two higher value near the true changepoints, we can find the distributions of model order for each segment, reported in Table 3.4

Segment	Constant	Linear	Quadratic
1	0	0	1
2	0	0.9975	0.0025
3	0	0	1

Table 3.4 Posterior distributions for model order

Despite the lack of accuracy in finding the changepoints at the correct position, Table 3.4 shows the correct model orders are found with certainty.

3.4 Autoregressive

The other type of data that we simulate is autoregressive, that is, data with segments that follow equation 2.30.

We generate our first data set for this section using the following equation:

$$y_t = N(0,1) + \begin{cases} 0.9y_{t-1} & 2 \le t \le 512 \\ 1.69y_{t-1} - 0.81y_{t-2} & 513 \le t \le 768 \\ 1.32y_{t-1} - 0.81y_{t-2} & 769 \le t \le 1024 \end{cases} \qquad 3.13$$

This equation comes from Davis et al. (2004) and using this data allows us to compare our results, and hence method, with those of Davis et al. (2004). A typical realisation of this data is shown in Figure 3.22, which is referred to as the Davis data.

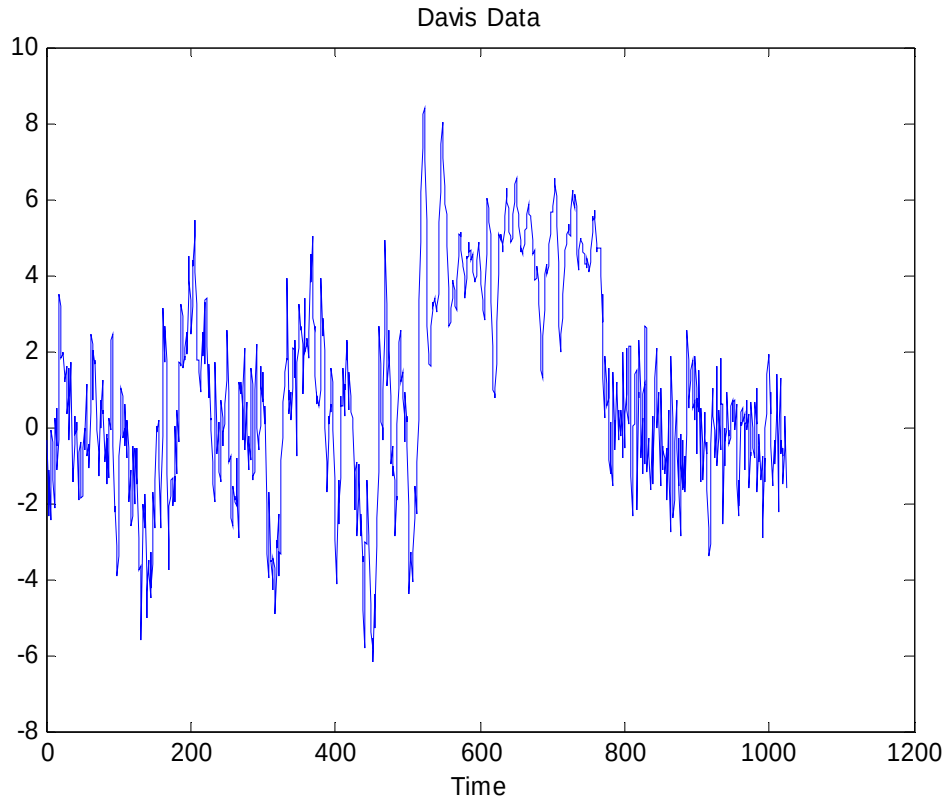

Figure 3.22 Davis data

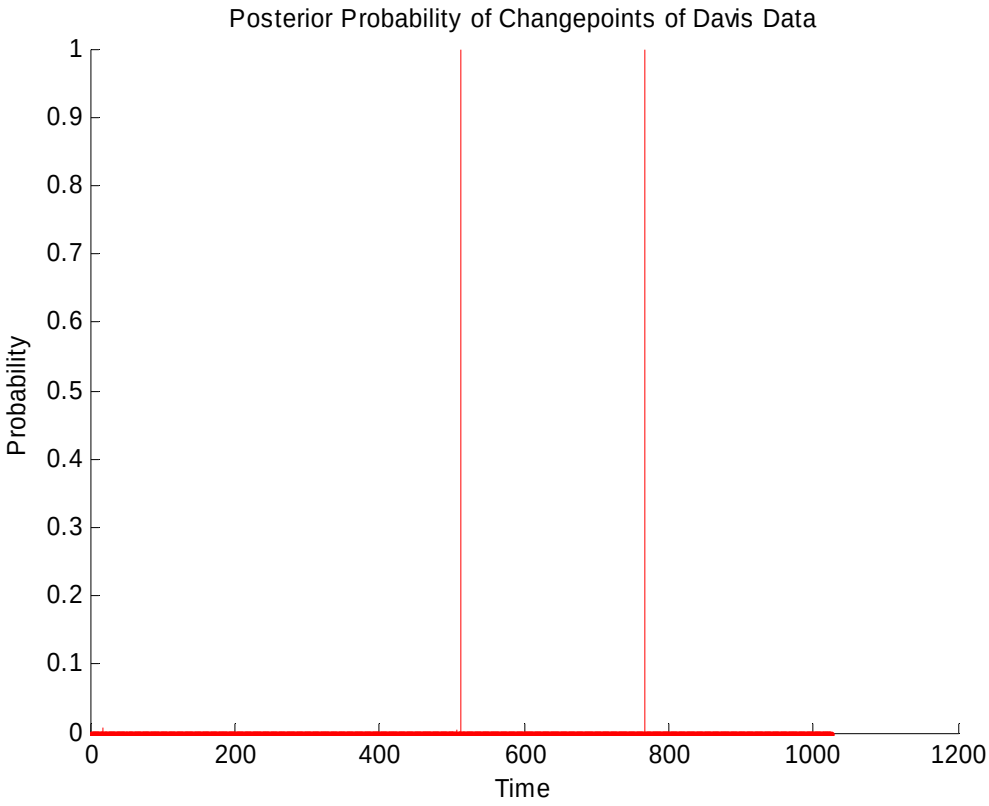

Figure 3.23 Posterior probabilities of changepoints for the Davis data

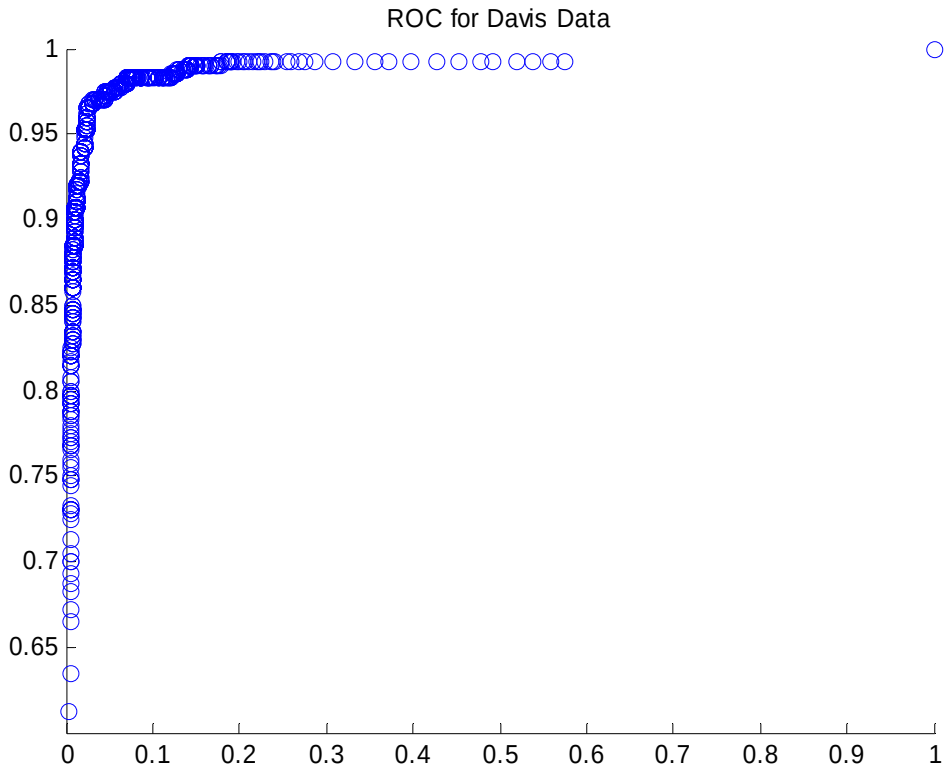

Figure 3.24 ROC curve for the Davis data. Area=0.9875

Figures 3.23 and 3.24 display the results obtained for the data in Figure 3.22. Figure 3.23 shows a flawless assessment of the data, finding both probabilities with certainty, and finding no false changepoints. The ROC curve is figure 3.24, with an area of 0.9875, also indicates a high level of accuracy of the algorithm on data from equation 3.13.

Segment	Constant	AR(1)	AR(2)
1	0.093	0.8996	0.0074
2	0	0	1
3	0	0	1

Table 3.5 Posterior distributions for model order

Table 3.5 shows that the distribution of model order for the segments found by the algorithm matches the true orders from equation 3.13, again noting that the only uncertainty is in the segment with lower model order, as in tables 3.2 and 3.4.

The second data set we investigate in this section also comes from Davis et al. (2004), and is defined by

$$y_t = a_t y_{t-1} - 0.81 y_{t-2} + \varepsilon_t \quad 3 \le t \le 1024, \tag{3.14}$$

where $a_t = 0.8\left[1 - 0.5\cos\left(\frac{\pi t}{1024}\right)\right]$, and $y_1, y_2, \varepsilon_1, \ldots, \varepsilon_{1024}$ are all distributed from a standard normal. Figure 3.25 shows a plot of this data.

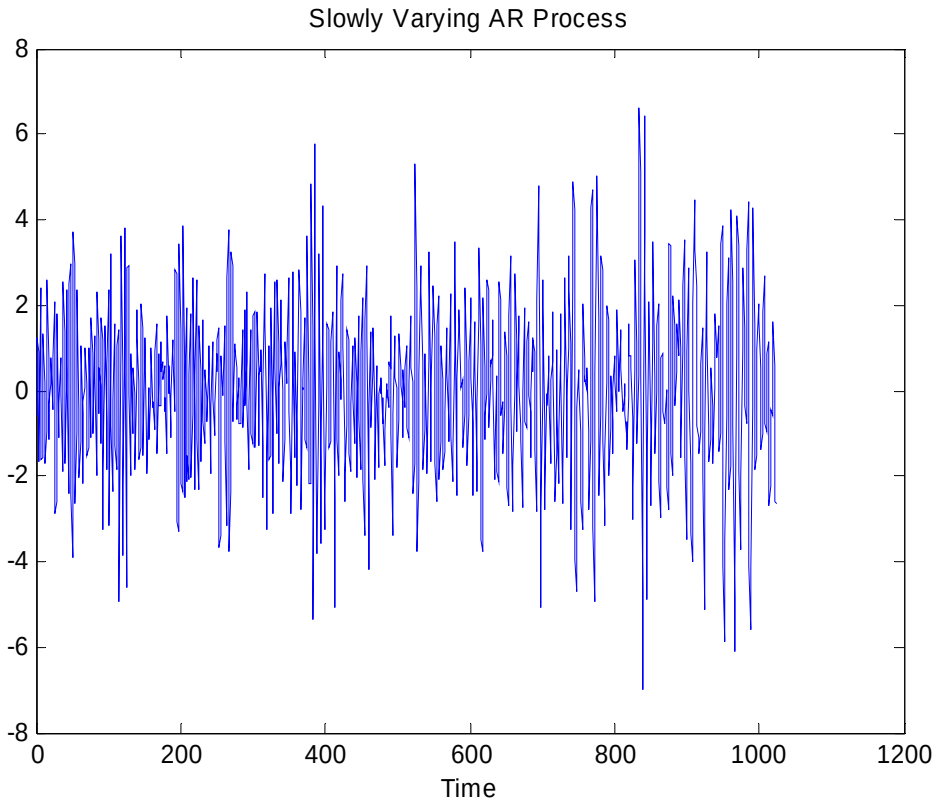

Figure 3.25 Data generated from equation 3.14

The parameter of the lag 1 term is constantly changing, and so every point in the series is a changepoint; however, these changes are so small at each time point that it is impossible to see the change in Figure 3.25, unlike Figure 3.22. Figure 3.26 below shows the time varying parameter a_t.

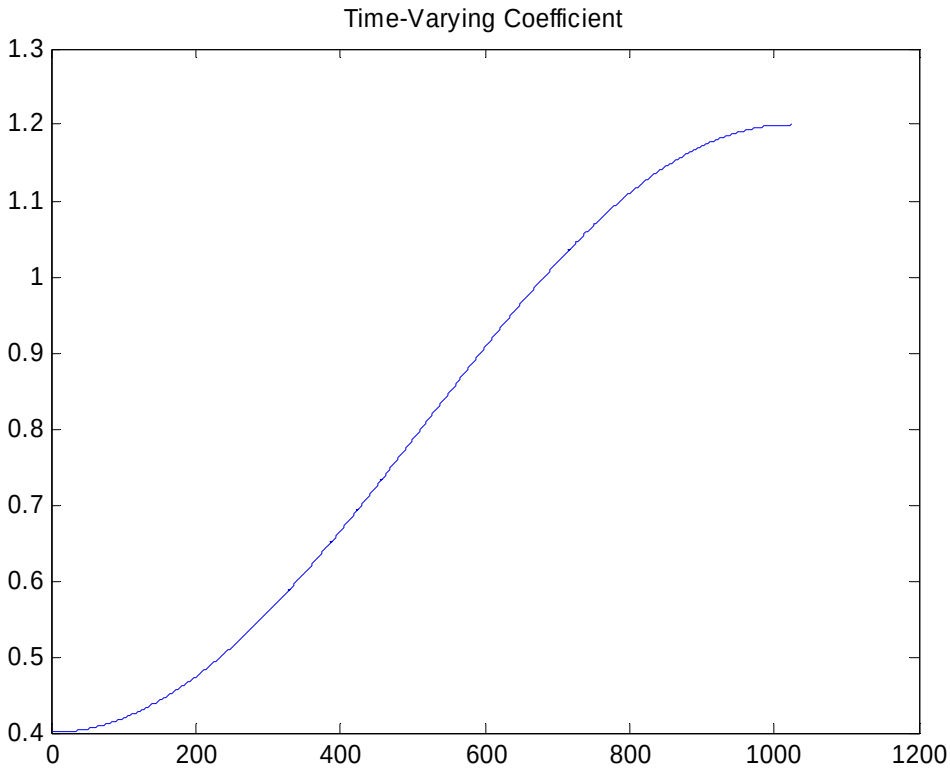

Figure 3.26 The coefficient a_t in equation 3.14

Davis et al. (2004) found changepoints at time 318 and 614. The algorithm we employ finds no changepoints on every iteration, i.e. the posterior probability of each point being a changepoint in zero for every point.

The reason no changepoints are found is due to an extremely low signal to noise ratio. The change in coefficient is never more that 0.002, and the noise variance is 1 throughout, completely obscuring the slight changes. It was shown in section 3.3.1 that a low signal to noise ratio has the effect of decreasing the likelihood of finding true changepoints, and this example merely takes this situation to an extreme.

3.5 Summary

In this chapter we have tested the performance of our algorithm by running it on various simulated data sets. We found that in most cases it accurately finds the location of changepoints with high probability, and ignores the non-changepoints. The situations that caused more erroneous results were found to be (i) larger noise variance added to the underlying model, and (ii) smoother transitions from one segment into the next. These results are unsurprising; larger nose variances obscure the changes between segments by reducing the signal to noise ratio, and smoother transitions make the exact location of the change more difficult to determine.

The distribution of model order for each segment was also investigated, with mixed results. In most cases, the distribution gave unit probabilities to the true model order in each segment, and those values that were less than one, were very high. Interestingly, it was found that these lower certainties tended to correspond to the segments that had lower than maximum model order, as in Tables 3.2, 3.4 and 3.5. Most significantly, however, is that in Table 3.3 a segment that is known to be quadratic was found to be most likely linear, or possibly constant. This indicates that the distributions of model order found by the algorithm are not always accurate, and so may need to be checked by an independent method.

Chapter 4

Well Log Data

4.1 Introduction

In this chapter we analyse a well-log data set. The data set looked at, shown in Figure 4.1, is measurements of the nuclear-magnetic response of underground rocks (Ó Ruanaidh and Fitzgerald,1996). Measurements were made at regular time points as a probe was lowered into a bore hole (Ó Ruanaidh and Fitzgerald, 1996). It is assumed the underlying model is piecewise constant, where a changepoint indicates a new level of nuclear-magnetic response, and therefore a new type of rock is being encountered by the probe. Finding changepoints for data of this sort has important consequences for oil-drilling, where changes in rock type need to be known in order to adjust the pressure in the borehole when encountering a new type of rock, to avoid blowouts (Fearnhead and Clifford 2003).

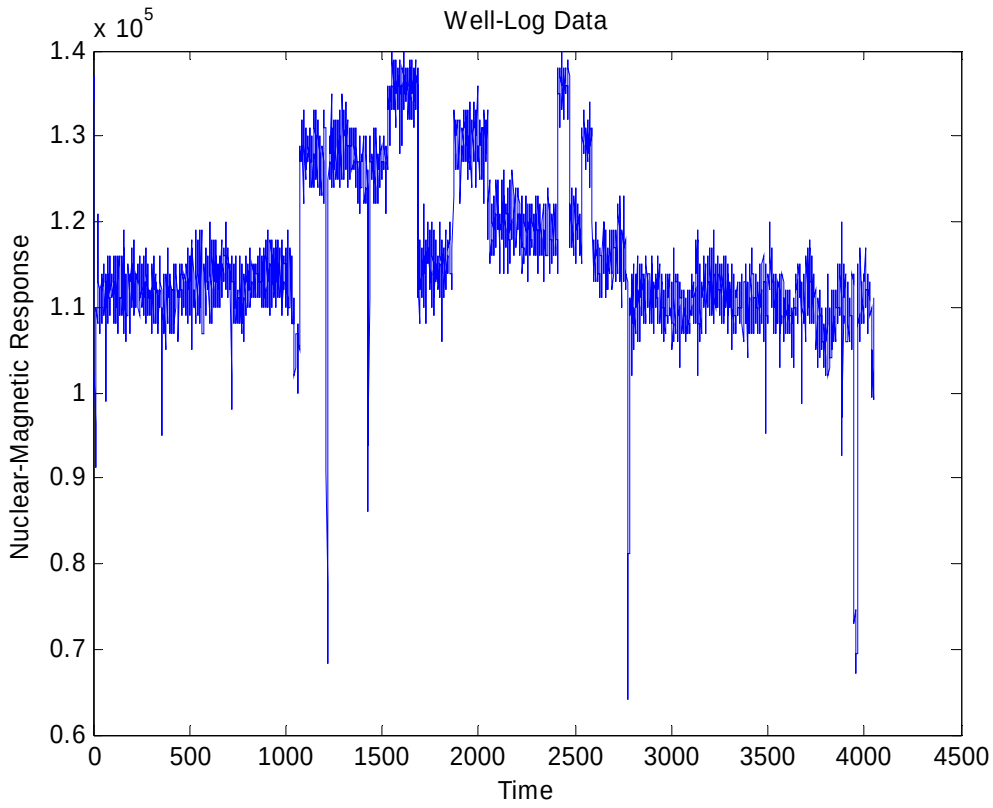

Figure 4.1 Well log data

4.2 Analysis of the data

Our algorithm is used to do an in depth analysis of the data. To reduce numerical instabilities in our analysis the data is scaled into the range (0,1). Doing this enables us to easily make more accurate hyperparameter selections, without affecting the number and position of the changepoints.

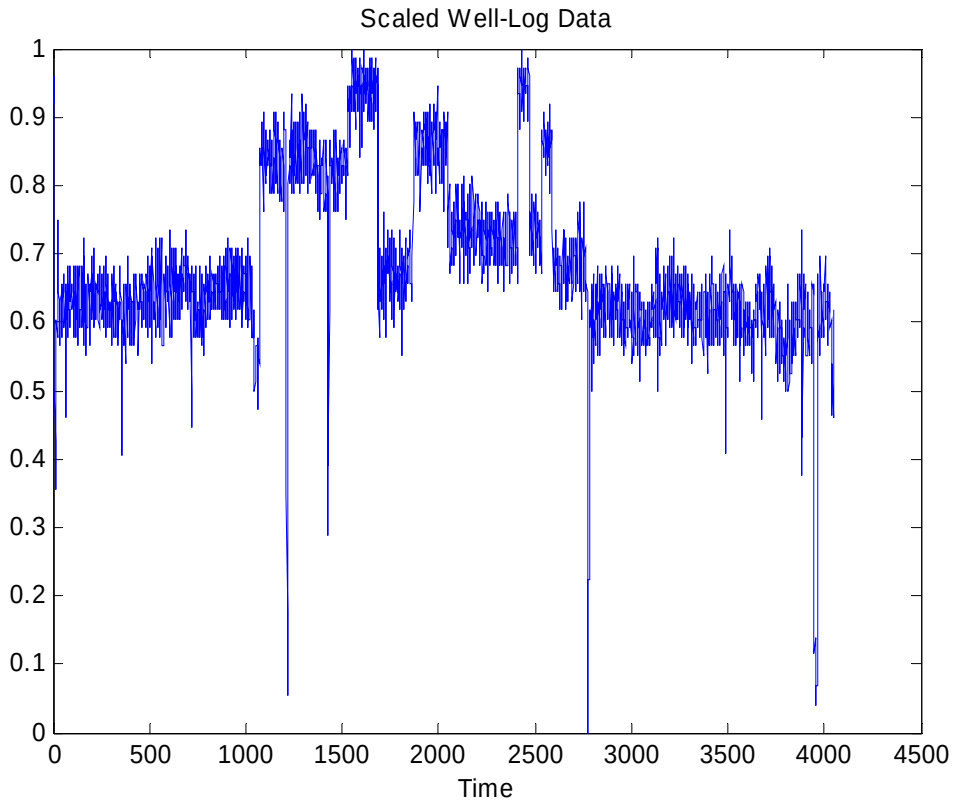

Figure 4.2 Scaled well log data

4.2.1 Hyperparameter selection

Using our bootstrapping algorithm, we find our hyperparameter values for the inverse gamma prior distribution for σ^2 to be

$$\begin{aligned} \nu &= 645.75 \\ \gamma &= 1.868 \end{aligned} \tag{4.1}$$

Since the model in this case is order 0, there is only one regression parameter β. Since this cannot be more than one, we set the value of δ such that the prior variance $\delta^2\sigma^2$ of β

makes it very likely that β is less than one, noting that β has prior mean of zero, and using the mode of our inverse gamma prior for σ^2 to calculate δ. Thus we have

$$\delta = 1.274 \times 10^7 \qquad 4.2$$

We use the same value of λ used in Fearnhead (2006):

$$\lambda = 0.004 \qquad 4.3$$

4.2.2 Results

In this section the algorithm is run on the well log data set, using the priors as discussed in section 4.2.1. Figure 4.3 is a plot of the posterior probabilities of each point being a changepoint. Each probability is calculated as the proportion of the 10,000 perfect simulations from the posterior distribution of changepoints that gave a changepoint at that location.

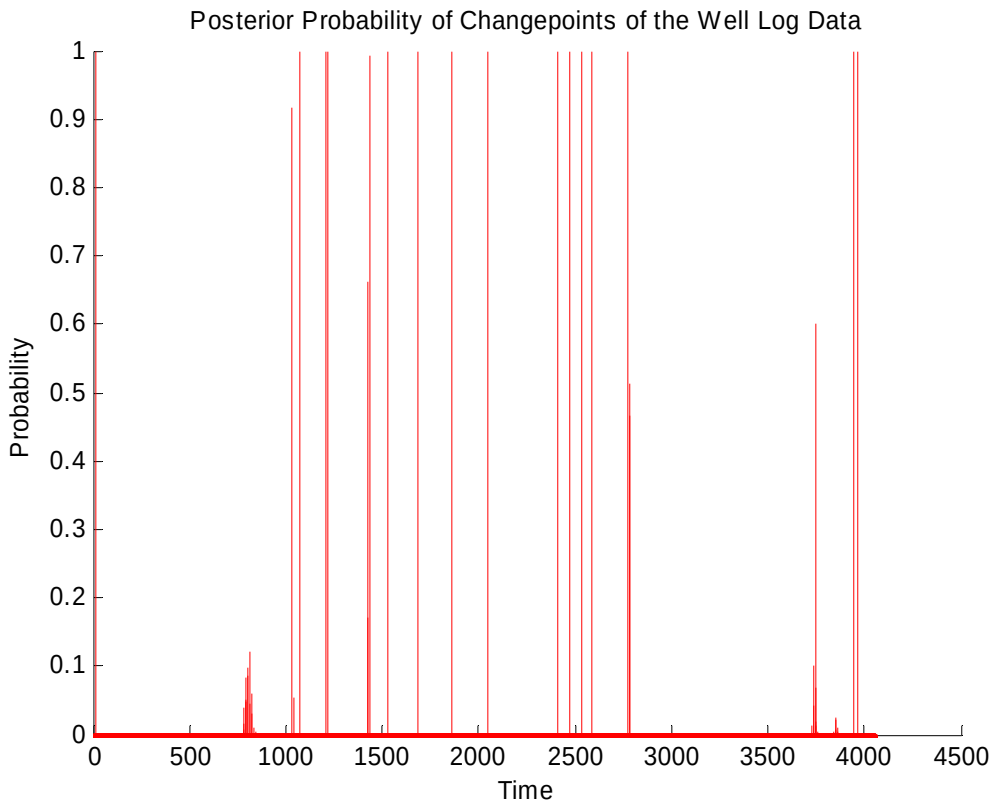

Figure 4.3 Posterior probability of changepoints of the well log data

Firstly, we observe the majority of nonzero probabilities are one, indicating a robust result; if the algorithm picks one on these points in one iteration, it seems to pick it in all of them. This unambiguity illustrates that there is a clear distinction in the rock types, as the algorithm is very explicit on where they occur.

Between times 780 and 840 we find a number of points with probabilities no greater than 0.15; a questionable changepoint of probability 0.6 at time 3744, and a very probable changepoint at time 1034 with probability .916. These points would need to be investigated further before further inference could be made

Given that this data has an underlying constant model, it is possible to make observations regarding changepoints merely by looking at the data and seeing where the level appears to change. Thus we can plot the changepoints over the data, and see how well they match.

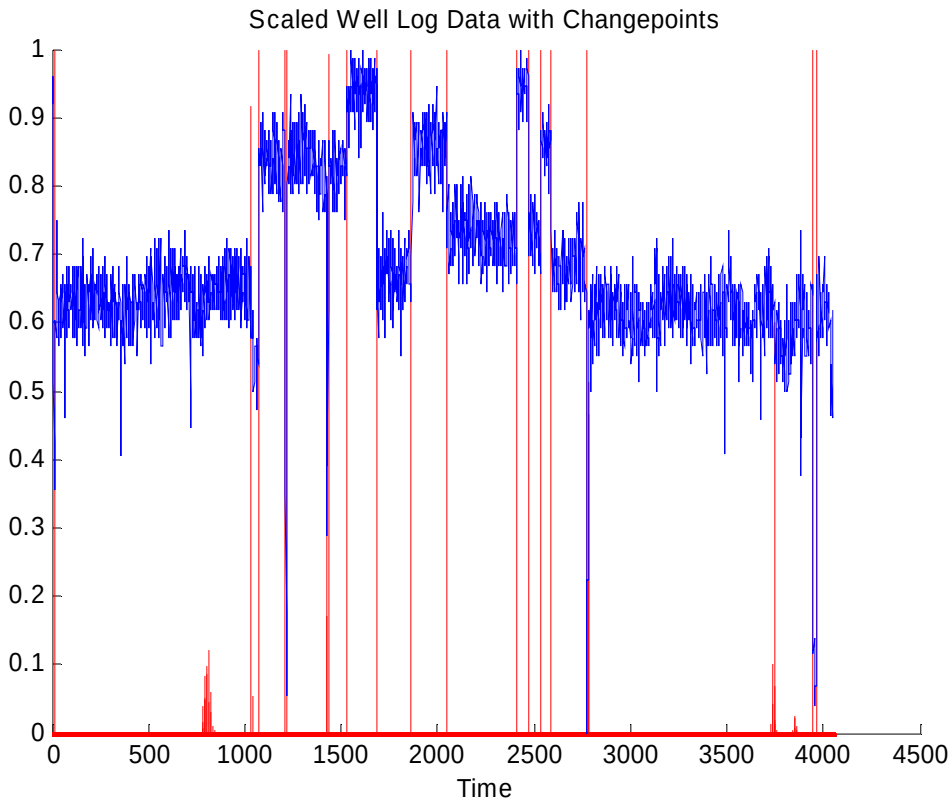

Figure 4.4 Scaled well log data with posterior probability of changepoints

Figure 4.4 shows the data overlaid with the posterior probability of the changepoints. This clearly shows that all of the changepoints with unit probability occur exactly where we would expect them to. Looking at the three exceptions discussed in the previous paragraph, we see that the small probabilities over times 780 to 840 correspond to an area that seems to increase slightly over the 60 time points, suggesting a single changepoint that has been picked up at different places on different iterations, or a change to a rock

with a very similar value of nuclear-magnetic response. The 0.916 probability at time 1034 seems quite clearly to be a drop in the level, albeit for a short time. The 0.6 probability at time 3744 is quite difficult to discern visually, justifying the ambiguous probability at that point.

To see the difference that our peak finding algorithm makes, we can look at the results of our algorithm if we do not use it.

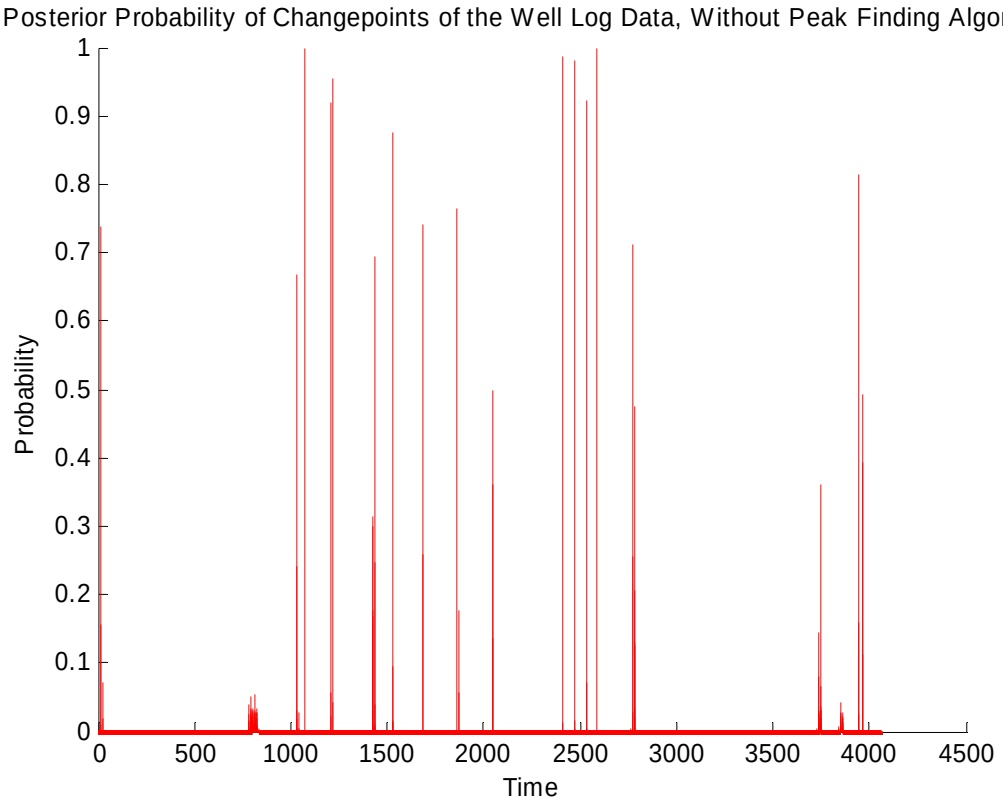

Figure 4.5 Posterior probability of changepoints of the well log data, without the peak finding algorithm

Figure 4.5 shows the results of our algorithm when not using the peak finding algorithm. We notice that the probabilities of changepoints decrease significantly. This occurs because on different iterations, the algorithm is locating the same changepoint in slightly different positions, which disguises the likelihood of many of the changepoints.

4.3 Comparison

In this section, we compare our results to those of Fearnhead (2006), and Fearnhead and Clifford (2003)

4.3.1 Fearnhead

In this section, we compare our results to those of Fearnhead (2006). To accurately compare, we use two values of λ, as Fearnhead did, namely $\lambda=0.004$ (see section 4.1) and $\lambda=0.013$. The second value is the mode of the posterior distribution for λ, using a uniform prior (with a geometric likelihood function), from Fearnhead (2006).

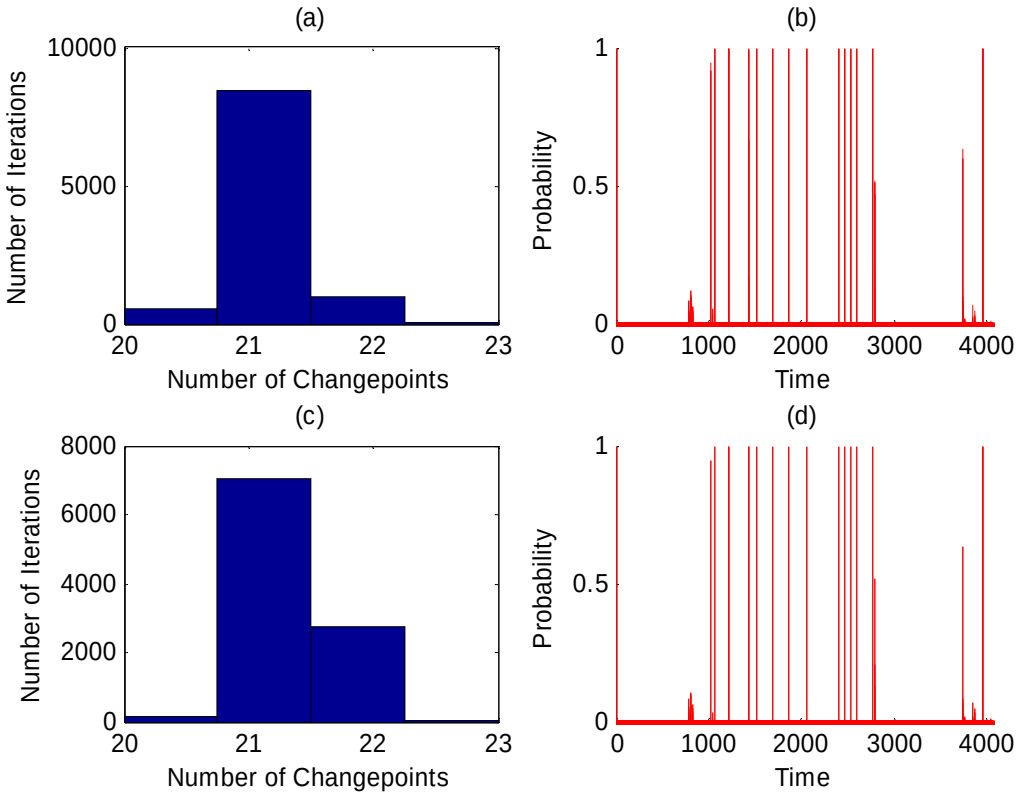

Figure 4.6 Results of analysis of the well log data

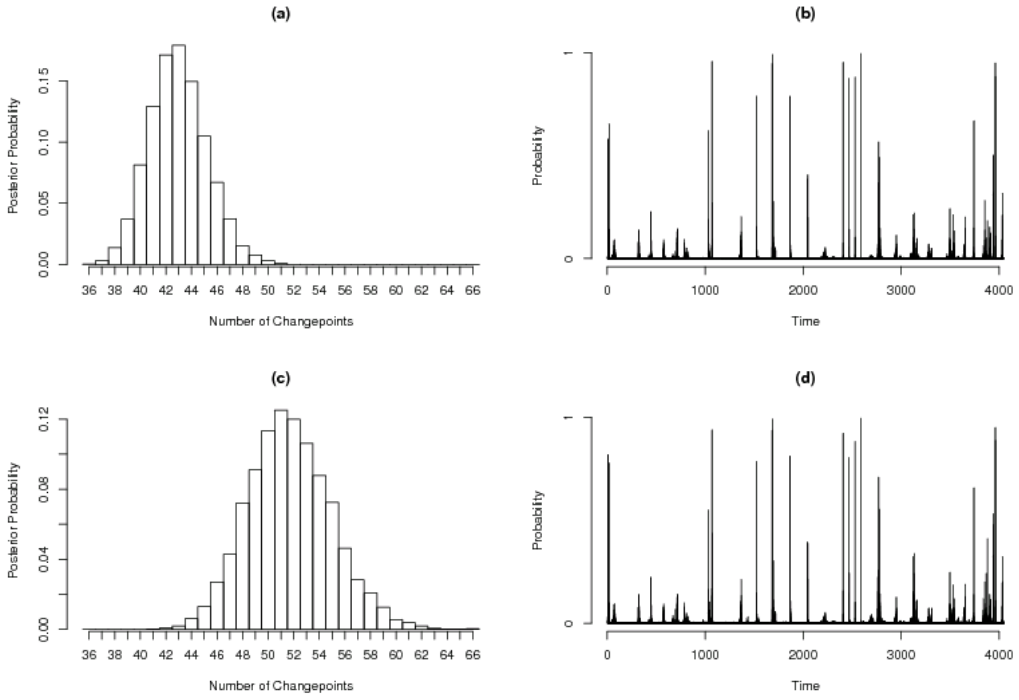

Figure 4.7 Results of analysis of the well log data from Fearnhead 2006

Figures 4.6 and 4.7 show similar analyses of the data, from our investigation, and from Fearnhead 2006, respectively. In both cases, (a) and (b) are for λ=0.004, and (c) and (d) for λ=0.013. The left hand plots give the distribution of the number of changepoints, while the right hand plots give the posterior distribution of the position of the changepoints. In both cases the results are based on 10,000 perfect simulations from the posterior distribution. In both cases the mean of this distribution increases when we increase λ, as expected. In our investigation, however, this shift of mean doesn't change the mode of the distribution at all, which remains at 21, due to the much lower variance.

Comparing the histograms, we see that Fearnhead has a much higher variance in his distribution of number of changepoints, and also has many more changepoints. The

reason for the disparity is the peak finding algorithm. The peak finding algorithm decreases the number of changepoints in each iteration by combining a group of changepoints that are 'close' into one changepoint, as explained in chapter 2. This has the effect of reducing the number of times the same actual changepoint is picked up more than once in a single iteration by the algorithm. A consequence of this is a much smaller variance in the posterior distribution of the number of changepoints.

Looking at the plots of posterior distribution of the changepoints, the first difference noticed is that in our analysis most of the nonzero probabilities are one, compared to Fearnhead, which finds many with probabilities in the range 0.4 to 1, with only 3 or 4 getting quite close to one. The reason for this difference is again the peak finding algorithm, as this eliminates the effect of having one true changepoint giving nonzero probabilities of being in contiguous positions.

Other than this effect we note that all of the changepoints with high probability occur mostly in the same position in both analyses.

The different values of λ seem to make very little difference in the posterior probability plots, particularly in Figure 4.6, whereas in Figure 4.7 we can see in some cases the values in (d) are slightly higher than those in (b), as we would expect since (d) has a larger prior probability on each time being a changepoint.

4.3.2 Fearnhead and Clifford

Fearnhead (2006) makes comparisons with Fearnhead and Clifford (2003). This latter paper takes a different approach to the problem. The main differences are that this method is not perfect simulation, as it uses MCMC; it uses a particle filtering algorithm, and it is online, i.e. it makes inference as to whether a point is a changepoint based only on the time values up to that point, compared to Fearnhead (2005, 2006) and our method, which use the whole data set.

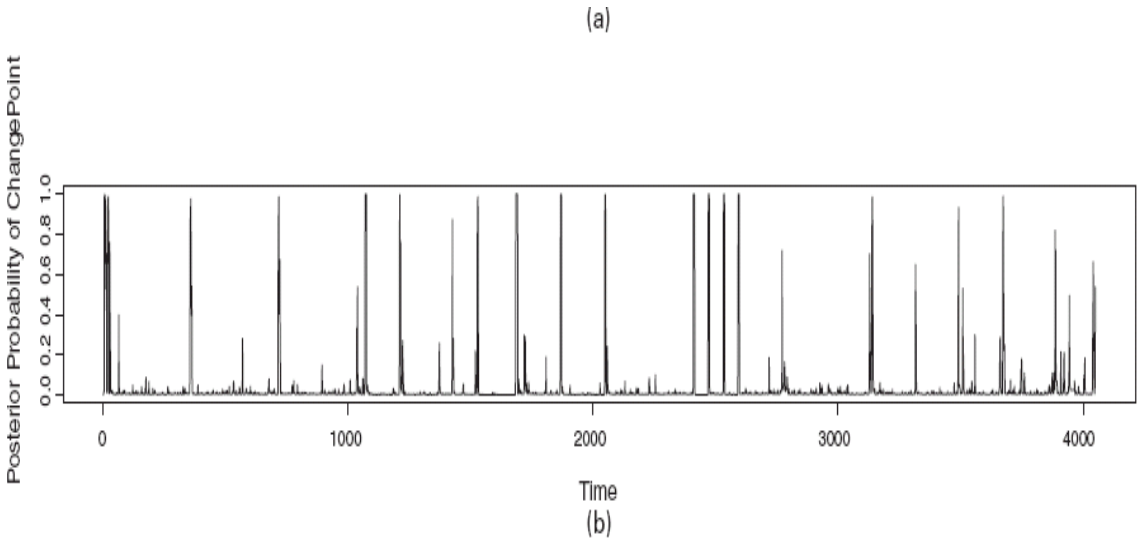

Figure 4.8 Analysis of Well-Log Data from Fearnhead and Clifford (2003)

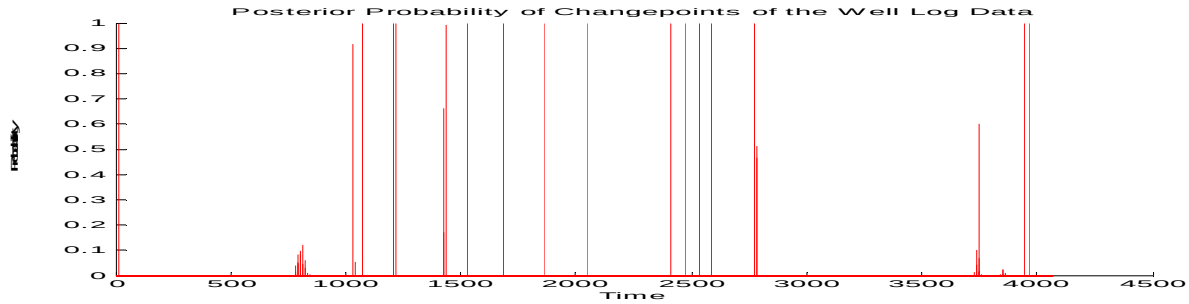

Figure 4.9 Probability of changepoints of the well log data

Figure 4.8 is an analysis of the well log data from Fearnhead and Clifford (2003), and figure 4.9 is our analysis, repeated here for comparison. Most of the changepoints we find are also found by Fearnhead and Clifford, although their method seems to find changepoints that we didn't, particularly in the ranges 0-1000, and 3000-3500.

As noted in Fearnhead (2006), Fearnhead and Clifford observed 16 changepoints (much closer to our estimate of 21, than to Fearnhead's of 40-60). The smaller number of changepoints is attributed to it being an online inference, using only data before a given point to make any inference on it, and to their only inferring a changepoint when the posterior probability of a changepoint within the last 10 points was greater than 0.9.

4.4 Summary

In this chapter we have looked at the well log data, using our algorithm to investigate the changepoints. We compared our results to those of Fearnhead, and found that the differences between our results and theirs was that our method with the peak finding algorithm reduced the posterior variance of the distribution of the number of changepoints, finding less changepoints, but finding those that it does with greater certainty. We found that the changepoints we discovered are almost exactly what we would expect from a visual inspection of the data.

Chapter 5

Seat Belt Data

5.1 Introduction

Our algorithm is used to find changepoints in a data set relating to the introduction of the seatbelt legislation in the UK. The data is the monthly number of deaths and serious injuries on UK roads from January 1975 to December 1984 (Brockwell and Davis, 2002). The seatbelt legislation was introduced in 1983 with the intention to decrease the number of deaths and serious injuries of those traveling in motor vehicles. The data set, from Brockwell and Davis (2002), is investigated with our algorithm and compared to Davis et al's. (2004) results.

5.2 Motivation

In February 1983 new seat belt legislation was introduced; we wish to see if this created a changepoint in the data from Davis et al (2002) shown in Figure 5.1. This would indicate that the legislation made a difference in the number of serious injuries or deaths.

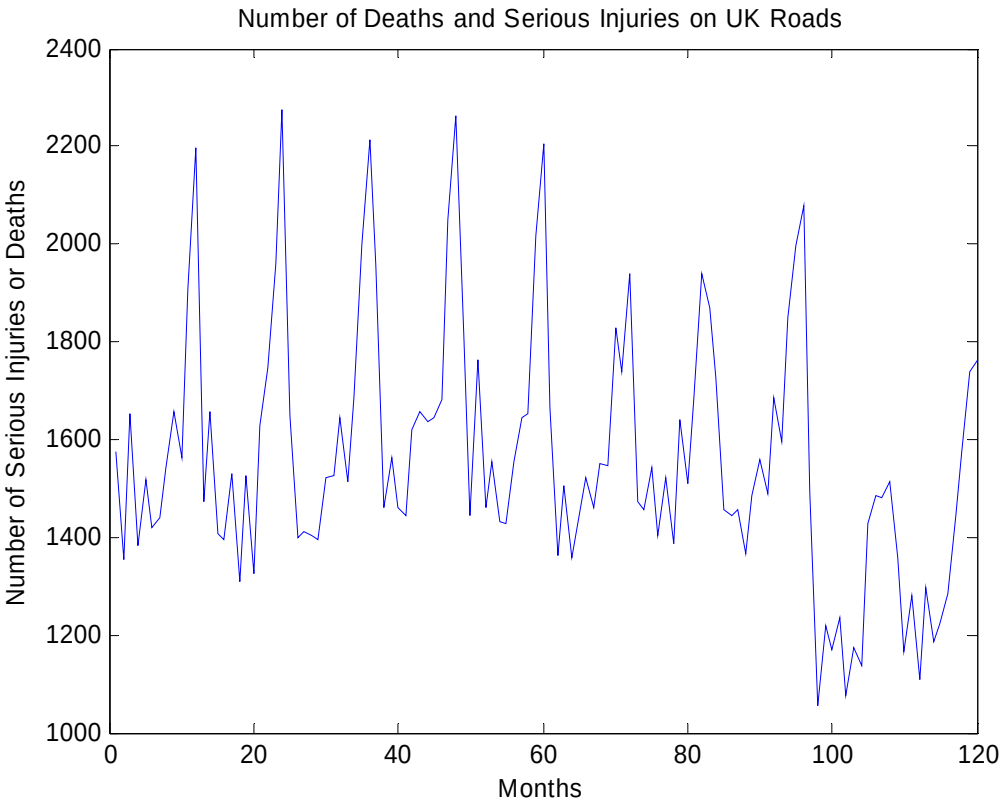

Figure 5.1 Number of deaths and serious injuries on UK roads from January 1975 to December 1984

We also compare our results to those from Davis et al. (2004), which uses a minimum description length (MDL) principle to find changepoints. The idea behind MDL is that the best model is the one that allows for maximum compression of the data. Davis et al. (2004) used a genetic algorithm to find the best model derived by the MDL. This algorithm is called Auto-PARM.

5.3 Setup

5.3.1 Notation

We call the time series y, and note that there are 120 time points. To perform our analysis we remove the seasonal component by looking at the 12 step differenced data, which we call x. Thus

$$x_{1:108} = y_{13:120} - y_{1:108} \qquad 5.1$$

is the series that we analyse, as in Davis et al (2004) and Brockwell and Davis (2002). We plot x in figure 5.2.

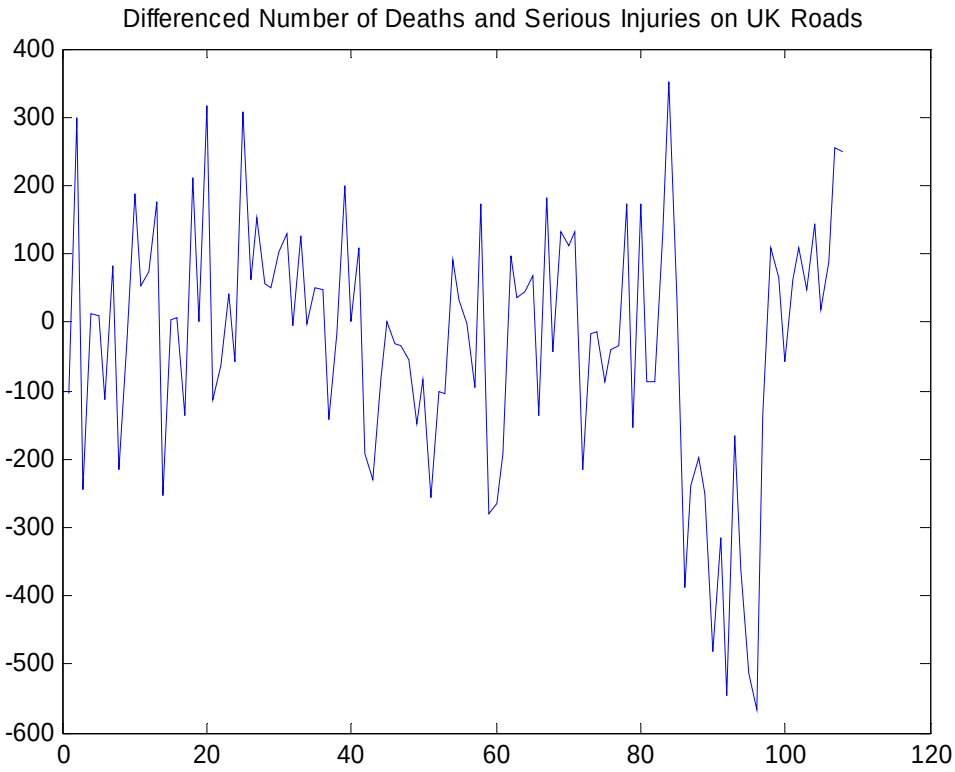

Figure 5.2 The 12-step differenced of the number of serious injuries and deaths on UK roads

Davis et al. (2004) found two changepoints in this series, one at February 1983 (time 86), and another at February 1984 (time 98), displayed in Figure 5.3.

The results of Auto-PARM on the seat belt data is displayed in Figure 5.3

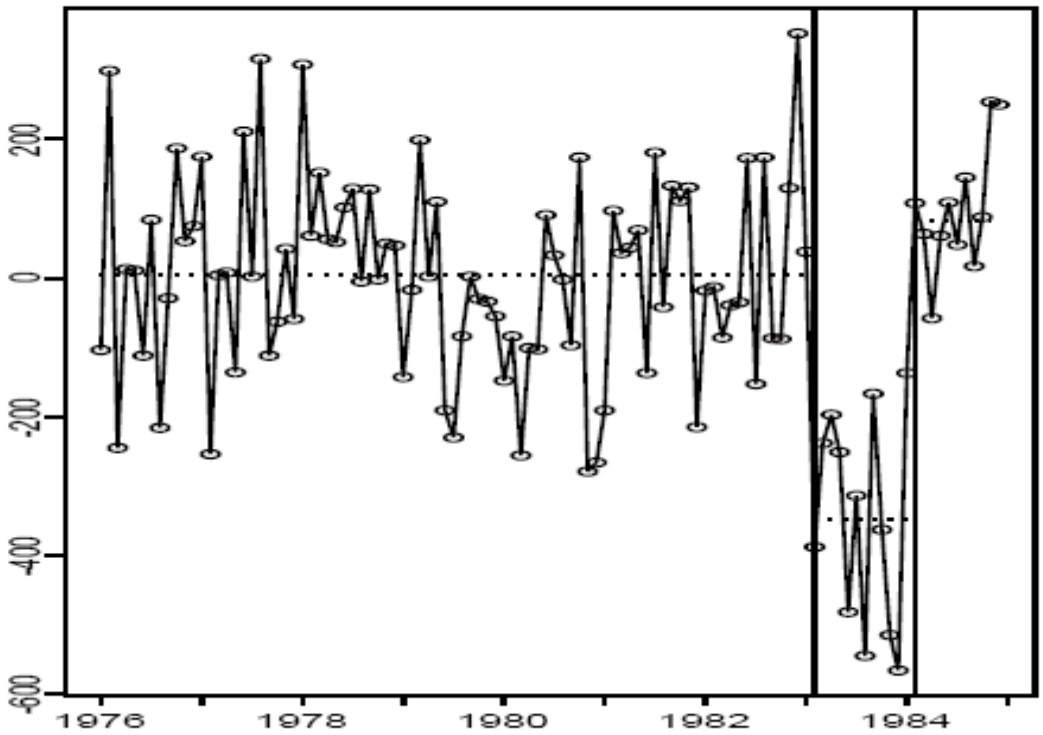

Figure 5.3 Results of Auto-Parm on the seat belt data (Davis et al., 2004)

5.3.2 Hyperparameter Selection

For an AR(1) model (see definition in equation 2.30) to be stationary, we require that

$$|\beta| < 1 \qquad 5.2$$

where β is the lag 1 coefficient (Chatfield 2003).

All of the segments found by Davis et al. (2004) were found to be either AR(1) or to have no AR component. For comparison, we run our algorithm assuming an AR model with maximum order 1, i.e. constant or AR(1).

Using the methods discussed in section 2.5, we find that the parameters for our inverse gamma prior distribution are

$$\begin{aligned} \nu &= 1.593 \times 10^4 \\ \gamma &= 5.031 \times 10^6 \end{aligned} \tag{5.3}$$

For the values of δ_1^2 and δ_2^2 we note the maximum values for the level and the AR(1) coefficient are 600 and one, respectively, and so we find

$$\begin{aligned} \delta_1^2 &= 0.7931 \\ \delta_2^2 &= 1.5601 \end{aligned}. \tag{5.4}$$

For this data set we have some previous analysis on which to base our choice of λ on: since Davis et al. (2004) finds two changepoints, we take

$$\lambda = \frac{2}{108}. \tag{5.5}$$

5.4 Analysis

Using the values found in section 5.3.2, we run our algorithm.

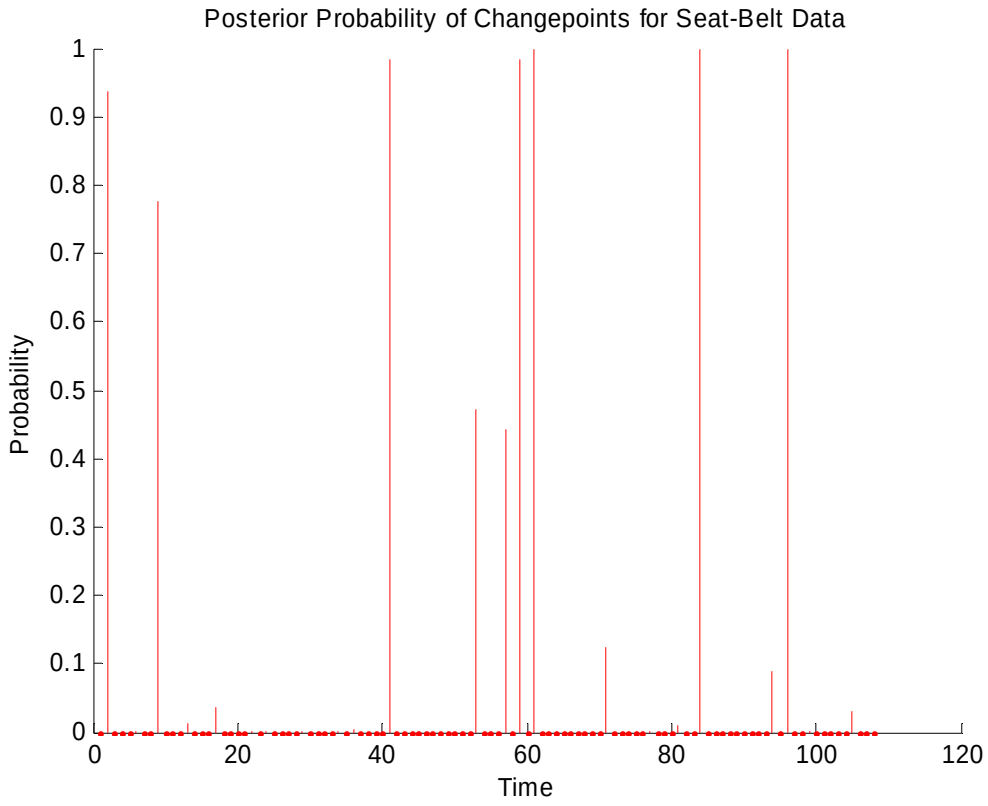

Figure 5.4 Posterior probability of changepoints for

As mentioned above, the motivation for investigating this data is to see whether the seat-belt legislation had an effect on the series. This legislation was introduced in February 1983, which corresponds to time point 86 in our differenced data. Figure 5.4 shows a changepoint at time 84 with certainty, indicating that there is indeed a change in the process at this point in time.

The other changepoint found by Davis et al. (2004) occurs at time 98, where we see a changepoint at time 96 with probability one, which strongly suggests that this

corresponds to the same phenomena. Other changepoints were also found earlier in the series, with various associated probabilities.

Davis et al. (2004) noted that in their analysis that the first two sections are iid, and the last is an AR(1) model. This means that they found $x_{1:86}$ and $x_{87:98}$ to have no AR component, and therefore essentially constant models, that change level at time 86. The segment $x_{98:108}$ follows an AR(1) model, which indicates that β_1 from equation 2.30 is nonzero, unlike the previous segments.

In our analysis, we can compute the posterior probability of the model order for each segment, conditioning on the location of the changepoints. In this case we set the cutoff probability to be 0.5, and thus find 8 segments.

When looking at the probability distribution for model order in each segment, we find that in all 8 segments have the probability of being either AR(1) or having no AR component is approximately equal. This means that for every segment, we have no way to decide whether that segment has an AR component or not.

A solution to this problem is to use the Schwarz information criterion (SIC) to decide what the best model for each segment is (we ignore the two segments of length 2). We present these results in table 5.1, noting that a smaller value for SIC indicates a preferable model.

Segment	SIC for constant	SIC for AR(1)	Choice
2-9	9.768	9.071	Ar(1)
10-41	9.726	9.770	Constant
42-59	9.644	9.810	Constant
62-84	9.919	10.013	Constant
85-96	10.069	10.286	Constant
97-108	9.180	9.330	Constant

Table 5.1 Model order choice for each segment

Davis found 1:86 and 87:98 as constant segments, and 99:108 as AR(1). We find many more changepoints, and have shown them to all be constant, except for the first one. We can now verify that there are indeed changepoints where we say there are.

Clearly the changepoint at time 9 must be, as the model order changes, i.e. the AR coefficient changes from 0 to something non-zero.

Looking at the changepoints from one constant model to another, we test to see that these are indeed different levels. We perform a two tailed t-test on the pairwise contiguous constant segments, under the null hypothesis that the segments have equal means. The p-values for this test are presented in table 5.2

Segments	Mean	p-value
10-41	0.6729	
42-59	0.5316	1.62×10^{-4}
62-84	0.9644	2.81×10^{-3}
85-96	0.2542	6.00×10^{-8}
97-108	0.7019	6.17×10^{-7}

Table 5.2 Tests of difference in mean.

Table 5.2 shows the consecutive constant segment and their means. The third column consists of p-values for testing a difference in the means of the segment in that row and the row above it. So, for example, the p-value for testing that the segments 10-41 and 42-59 have different means is 1.62×10^{-4}.

We compare this to Davis et al (2004), which found that the times from 1 to 86 and 87 to 98 to be two separate segments, each with no AR component, and 99 to 108 to have an AR(1) model. Our analysis shows that we find many more changepoints than this, which we have verified using an independent statistical method. Moreover we find that the last segment has no AR component, and that some of the first segment found in Davis et al. (2004) is actually from an AR(1) model, not constant.

5.5 Summary

In this analysis of the number of deaths and serious injuries on UK roads, many changepoints were found. Most importantly, a changepoint was found to occur around the time that new seat belt legislation was introduced. This change was shown to be a very significant decrease in the mean number of deaths and serious injuries.

An independent method for finding the model order in each segment was used, which verified the changepoints found above, as opposed to those found by Davis et al. (2004).

Chapter 6

Baby Data

6.1 Introduction

Our algorithm is used to determine the changepoints in the variability of physiological signals from preterm babies. The premature body of a preterm baby experiences cardiorespiratory instabilities due to underdeveloped internal systems. Some pediatric illnesses also affect cardiorespiratory functions in similar ways, so it can be difficult to determine if a preterm baby is ill or not. Clinicians think information about the state of health of preterm babies is contained in the variability of their physiological measurements. One way to quantify this variability is through use of a stochastic volatility model (Lee et al. 2005).

6.2 Motivation

Finding the changepoints using our algorithm will help to improve the modeling of variability in physiological signals. The series we investigate is 4050 points of blood oxygen concentration taken every two seconds (Zhao et al. 2007). We wish to see

whether using a stochastic volatility model is an appropriate method to model the variability in this data.

In a stochastic volatility model, the data is modeled by

$$z_t = e^{\frac{v_t}{2}} \varepsilon_t \qquad 6.1$$

where ε_t comes from a standard normal distribution. We see that equation 6.1 is a heteroskedatic normal model, whose time varying variance is governed by v_t, referred to as volatility in the finance literature. The volatility is assumed to evolve according to

$$v_t = a v_{t-1} + b \eta_t + c \qquad 6.2$$

(with η_t also distributed according to a standard normal), which we can see follows an AR(1) model. Thus once we find this series, we can run our algorithm to see if a single AR(1) model is appropriate for the data.

The data we have is measurements of blood oxygen concentration, and we call this series *x*. This data is discrete, as the machine that measures it rounds to the nearest percent. They are preprocessed by adding a uniform (-0.5,0.5) jitter to convert them to real values. This is displayed in Figure 6.1.

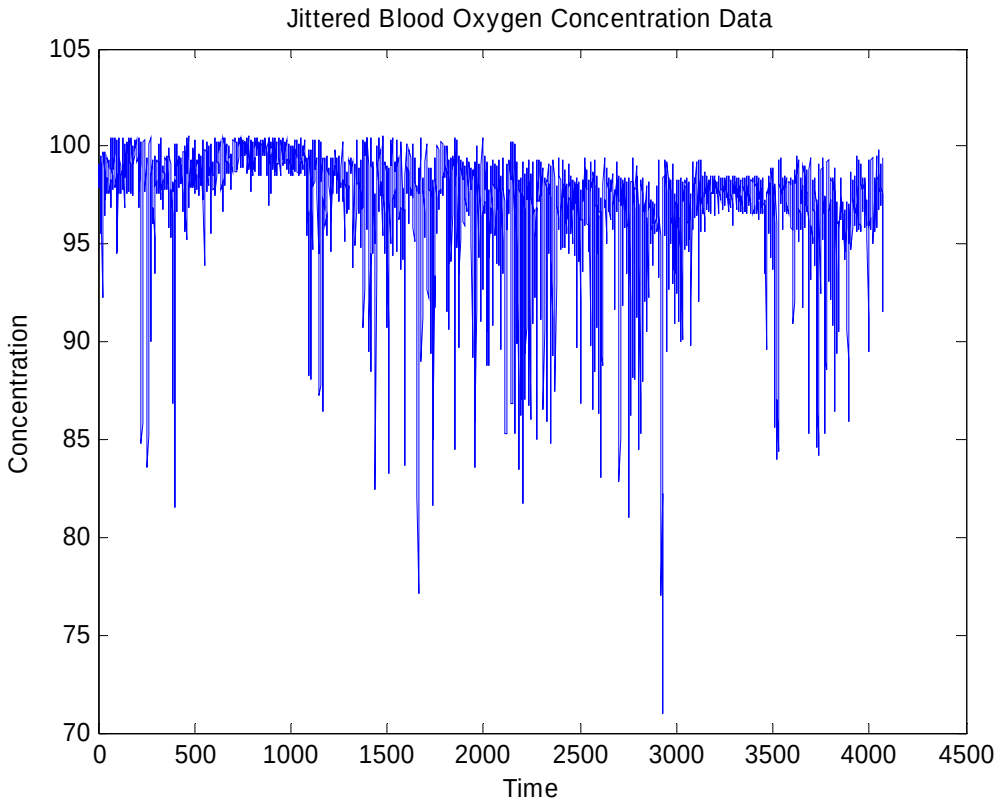

Figure 6.1 Jittered blood oxygen concentration

According to equation 6.1, the data should be a zero mean normal process with time varying variance. Clearly, the x series does not conform to this and has to be transformed by taking

$$z_t = \log\left(\frac{x_t}{x_{t-1}}\right), \qquad 6.3$$

displayed in Figure 6.2.

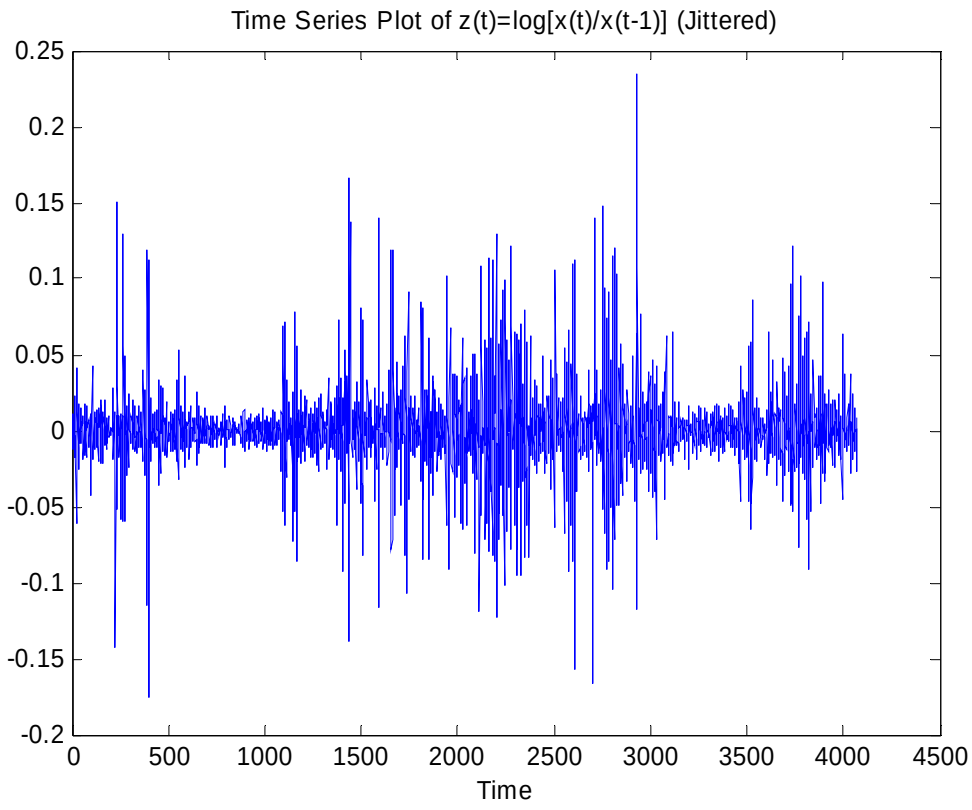

Figure 6.2 Jittered values of z_t

Now, e^{v_t} is the variance at each point of z_t. To calculate these values, we take windows of certain length of the transformed data around a specific point, and take the variance of the window as the associated value for that point. If this window should go below time 1 or above time 4070, then the current window simply has less values in it. The length of the window was decided by taking many different values of window length, and choosing the one that has the lowest mean squared error when compared to estimates of the time varying variance obtained from the stochastic volatility model (Zhao et al. 2007). It was found that the best window length to use is 7.

Having calculated estimates of these values of e^{v_t}, the estimates for v_t are found by taking logarithms, shown in Figure 6.3.

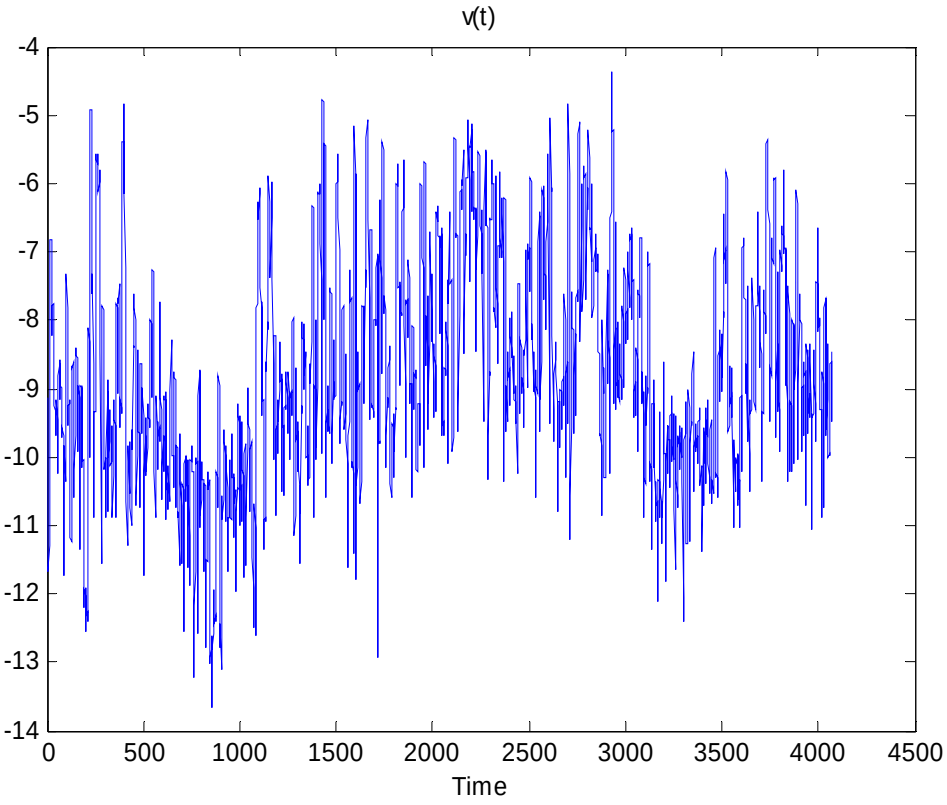

Figure 6.3 Plot of v_t for transformed oxygen measurements.

We can now apply our algorithm to see if there are any changepoints, assuming an AR model with maximum order one. If changepoints exist then we conclude that v_t cannot be modeled by a single AR(1) model. If this is the case then we cannot use a stochastic volatility model with fixed coefficients for z_t.

6.3 Prior Selection

For ease of prior selection, we run our algorithm on the data shifted so as to have mean zero, which reduces the error in the choice of hyperparameters.

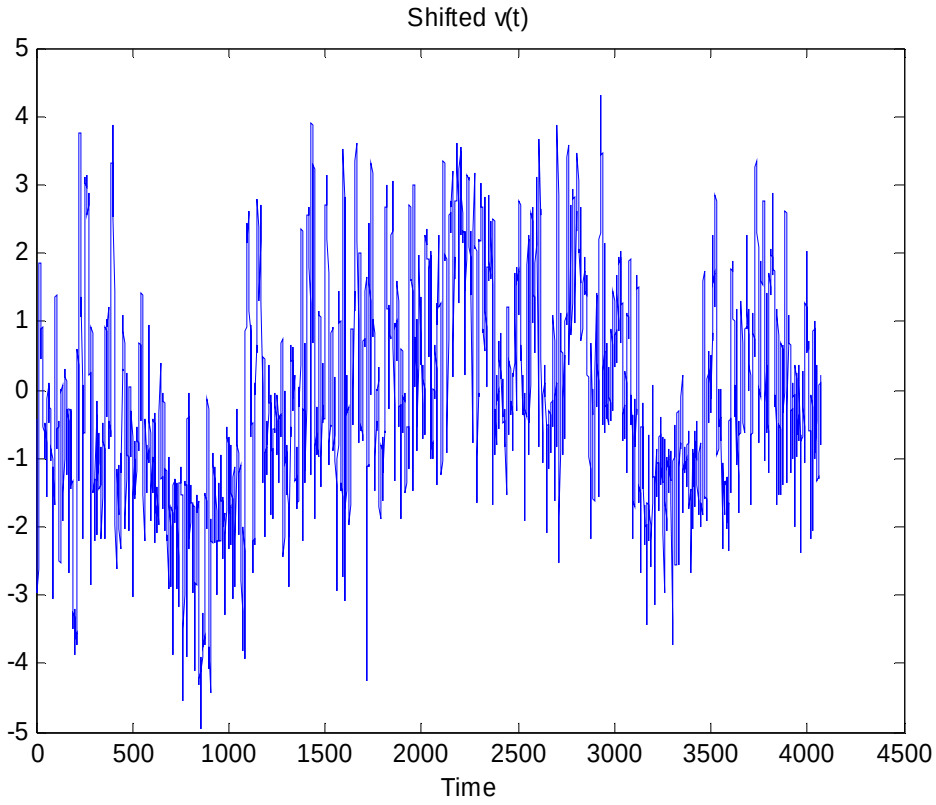

Figure 6.4 Shifted data

We again use the methods outlined in section 2.5 to find the required values ν, γ, δ_1 and δ_2. The upper bound values we use are 3 for the level, and 1 for the lag 1 coefficient. So we find that

$$\begin{aligned} \nu &= 1.4569 \\ \gamma &= 1.2174 \times 10^{-3} \\ \delta_1^2 &= 0.6876 \\ \delta_2^2 &= 0.0764 \end{aligned} \qquad 6.4$$

Given that we have no previous analysis for this data, we have no information on which to base our choice of λ. We choose the value $\lambda = \frac{1}{4070}$, as this is the lowest sensible value of λ for this data. (A choice of λ=0 always returns no changepoints, and we wish the denominator to be 4070, the length of the series.) We wish to choose the lowest value, as this insures that we do not overestimate the number of changepoints, i.e. if we do find any changepoints, we can be more certain that the data cannot be fitted with a single AR(1) model.

6.4 Analysis

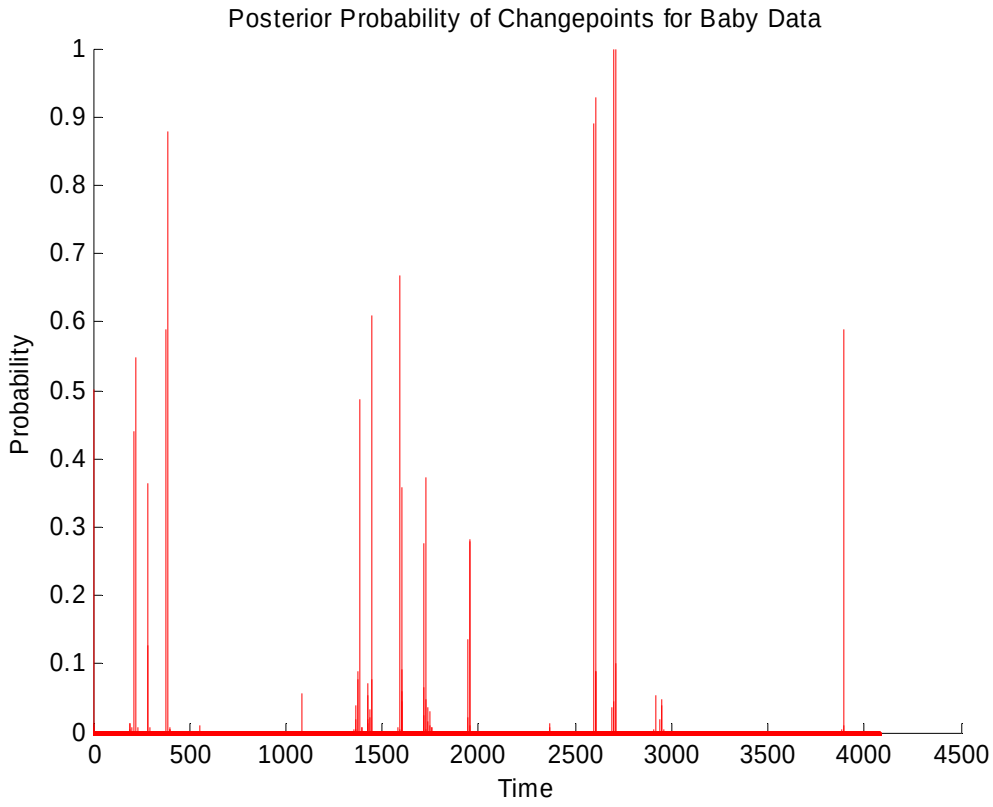

Figure 6.5 Posterior probability of changepoints for the data in Figure 6.3

Figure 6.5 illustrates that we do indeed find some changepoints, including some with near certain probability. This indicates that we cannot model the series *v* with a single AR(1) model, and hence cannot apply the stochastic volatility model with fixed coefficients to the series *z*.

6.5 Summary

In this chapter we looked at a data set of physiological measurements from a preterm baby, to investigate how to model it. In particular, we wished to see if it is possible to model the variability of blood oxygen concentration data using a stochastic volatility model with fixed coefficients. We found that this may not be the best model, as we showed that the data may have more than one AR(1) segment.

Chapter 7

Conclusions

7.1 Research Conclusions

Changepoint detection is required for many time series and it can reflect important implications for the subject the data was sourced from. A changepoint in a time series often indicates occurrences of important events; hence it is important and of interest in many applications to detect where these changepoints occur. In this research a method of changepoint detection, an algorithm from Fearnhead (2005, 2006) was improved and analysed. Our algorithm was run on multiple simulations to test its performance and accuracy, and then used to analyse three real data sets.

The algorithm described in this research has many advantages over other similar methods. It was found in the simulation studies that the algorithm performed with high levels of sensitivity and selectivity, as measured by the ROCs. The method can be adapted to find changepoints in any data with segments that can be modelled with equation 2.1, that is, any type of linear regression.

Extensions to the original algorithm from Fearnhead (2005, 2006) that are presented in this research increase the effectiveness and simplicity of the method. A major difficulty in using the original algorithm is the problem of finding good values of the hyperparameters of the prior distributions to input. A poor choice of these values can greatly reduce the accuracy of the method. The method suggested in section 2.5 explains a method of using the data to inform these parameters, which proved to give accurate results in chapter 3.

Another problem of the original algorithm is that due to leakage of probabilities, it can become difficult to see where the true changepoints are and, more significantly, what the true posterior probability of a point being a changepoint is. Section 2.6 describes the peak finding algorithm, which solves this problem by shifting the probabilities of nearby points to the points whose probabilities are local maxima. This allows a more accurate inference on the probability of a point being a changepoint, and reduces the problem of having many consecutive points with small probabilities of being changepoints. A drawback to this method, however, is that in situations where the leakage is severe, the algorithm can pick up multiple maxima around a single changepoint, and so even with the peak finding algorithm many changepoints may be found.

Another weakness of the algorithm was found in the calculation of the posterior distribution of model order for each segment. While most cases the simulation studies showed success in identifying the true model order with high probability, sometimes, as

in Table 3.3, the distribution indicated the wrong model order entirely. Similarly, in chapter 5 the distribution of model order gave ambiguous results for every segment, indicating they were all equally likely constant and AR(1).

7.2 Further Extensions

An extension to the algorithm from Fearnhead (2005, 2006) discussed in this paper is presented in Fearnhead and Liu (2007). This new method is an online algorithm, which means that the inference made on whether a point is a changepoint is based only on the data before that point; the rest of the series is not used. The advantage of this type of method is that decisions about the existence and location of changepoints can be made while the data is being sampled. This can be important, for example, in medical data, where inferences on the data can be required instantly.

Another potential direction for extension is investigating situations where changes occur continuously over a period of time, instead of at a single point. This would solve the problem found in analysing the second data set in section 3.4, where the series was changing continuously. With such an extension to the algorithm, a series with data like that in Figure 3.25 in the middle of it would find this section of data to be a single change that varies over time.

Appendix A

Map of Matlab Functions

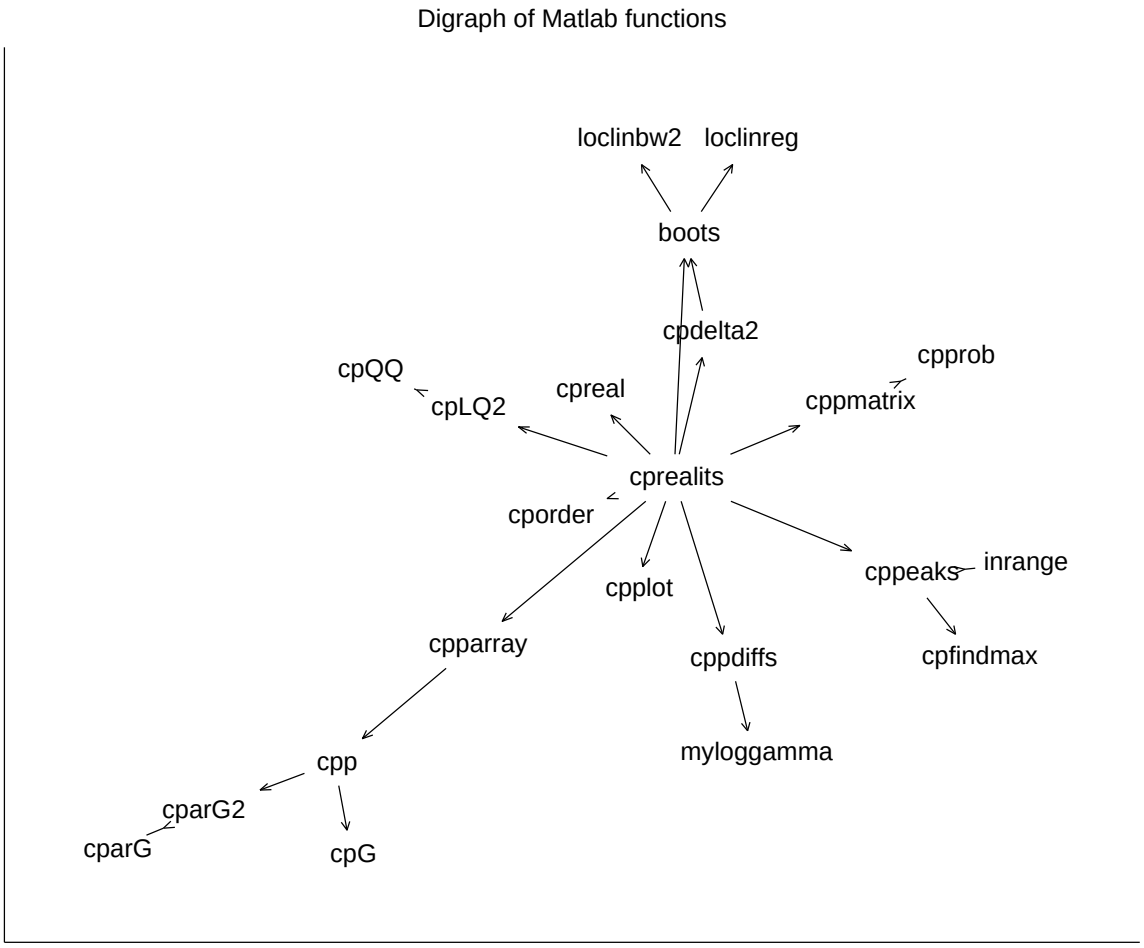

Figure A.1 Map of Matlab functions

Appendix B

List of Matlab Functions

boots

```
function [nu,Gamma]=boots(x,m)
% [nu,Gamma]=boots(x,m)
%
% Performs a bootstrapping method to find the mode (nu) and variance
% (Gamma) of an inverse gamma distribution. We fit a local linear
% approximation to x, and subtract this from x to obtain the residuals.
We
% bootstrap these m times, and use moment matching to find nu and
Gamma.

% Andrew Richens
n=length(x);
mn=m*n;
t=(1:n)';
bw=loclinbw2(t,x,1,1);
llr=loclinreg(t,x,t,bw,1);
res=x-llr;
res=res-mean(res);
y=randsample(res,mn,true);
y=reshape(y,n,m);
yvar=var(y,0,1);
mu=mean(yvar);
Gamma=var(yvar);
nu=mu*(mu^2+Gamma)/(mu^2+3*Gamma);
```

cparG

```
function G=cparG(y,a,b,q)
% G=cparG(y,a,b,q)
%
% Fills in values for a basis matrix for an AR(q) model, from a to b. y
is
% the time series.

% Andrew Richens
s=b-a+1;
G=zeros(s,q);
G(2:s,1)=y(a:b-1);
if a~=1
    G(1,1)=y(a-1);
end
```

```
for i=2:q
    G(i:s,i)=G(i-1:s-1,i-1);
end
for i=2:q
    for j=1:s
        if a-i+j-1>0
            G(j,i)=y(a-i+j-1);
        end
    end
end
```

cparg2

```
function G=cparg2(y,a,b,q)
% G=cparg2(y,a,b,q)
%
% Uses cparG to create a basis matrix for an AR(q) model, from a to b.
This function
% is used when a constant parameter is allowed in the AR model. y is
the
% data.

% Andrew Richens
F=cparG(y,a,b,q-1);
s=size(F);s=s(1);
q=ones(s,1);
G=[q F];
```

cpdelta2

```
function del=cpdelta2(k,y,m)
% del=cpdelta2(k,y,m)
%
% Finds best value of delta. k is upper limit (3*sd=k), y is the data,
m is
% number of bootstraps, usually 1000.

% Andrew Richens
nu=boots(y,m);% nu is mode of IG, Gamma is var, mu is mean;
del=k^2/(9*nu);
```

cpfindmax

```
function v=cpfindmax(x)
% v=cpfindmax(x)
%
% Used in peak finding algorithm. Creates a vector of positions of
local
% maxima for the series x.

% Andrew Richens
n=length(x);
i=1;
v=zeros(1,n);
```

```
if x(1)>x(2)
    v(1)=1;
    i=i+1;
end
for j=2:n-1
    if x(j)>x(j-1)&&x(j)>x(j+1)
        v(i)=j;
        i=i+1;
    end
end
if x(n)>x(n-1)
    v(i)=n;
    v=v(1:i);
else v=v(1:i-1);
end
```

cpG

```
function [G]=cpG(a,b,q)
% [G]=cpG(a,b,q)
%
% Creates the matrix of polynomial basis function of order q (design
% matrix) from a to b.

% Andrew Richens
c=b-a+1;
G=zeros(c,q);
G(:,1)=1;
for i=2:q
    G(:,i)=(1:c).^(i-1);
end
```

cpLQ2

```
function LQ=cpLQ2(y,q,A,lambda)
% LQ=cpLQ2(y,q,A,lambda)
%
% Creates the vector of log(Q) values for data y. A is the q-cell array
% of matrices with P(s,t), one for each value of q, the maximum allowed
% order. lambda is the parameter of the geometric prior on the position
and
% number of changepoints.

% Andrew Richens
n=length(y);
LQ=zeros(1,n);
for j=1:q
    a(j)=A{j}(n,n);
end
LQ(n)=log(sum(a));      %initialization of Q(n)
for i=n-1:-1:1
    LQ(i)=LQ(i+1)+cpQQ(y,i,q,LQ,A,lambda);
end
```

cporder

```
function cq=cporder(q,A,C,n)
% cq=cporder(q,A,C,n)
%
% Calculates the posterior distribution of model order in each segment.
q
% is the maximum model order, A is a cell array of P(s,t) values, C is
a
% vector of changepoints, and n is the length of the time series.

% Andrew Richens
lc=length(C);
b=[0 C];
cq=zeros(q,lc+1);
for i=1:length(C)
    for j=1:q
        cq(j,i)=exp(A{j}(b(i)+1,b(i+1)));
    end
    cq(:,i)=cq(:,i)/sum(cq(:,i));
end
if lc>0
    for j=1:q
        cq(j,lc+1)=exp(A{j}(b(lc)+1,n));
    end
    cq(:,lc+1)=cq(:,lc+1)/sum(cq(:,lc+1));
end
```

cpp

```
function p=cpp(y,D,a,b,q,Delta,Gamma,dec)
% p=cpp(y,D,a,b,q,Delta,Gamma,dec)
%
% Finds the p(s,t,q) values from Fearnhead 2005. y is the data, a and b
are
% the current values s and t, q is the model order, Delta is a diagonal
% matrix of variances (delta^2), Gamma is the scale parameter on the
% inverse gamma prior on the variance. dec is the model choice.

% Andrew Richens
n=length(y);
if strcmp(dec,'Pn')
    G=cpG(a,b,q);
elseif strcmp(dec,'Ar')
    G=cparG2(y,a,b,q);
elseif strcmp(dec,'0')
    G=zeros(b-a+1,q);
else error('Must Choose Ar, Pn or 0');
end
M=inv(G'*G+inv(Delta));
yab=y(a:b);
p1=1/2*log(det(M));
p21=D{2}(b-a+1);
p22=log(Gamma-(yab'*G)*M*(G'*yab)+yab'*yab);
p2=p21*p22;
```

```
p3=D{1}(b-a+1);
p4=D{4};
p5=D{5}; %Delta is a matrix of Delta_i^2
p6=D{6};
p7=D{3}(b-a+1);
p=p1+p2+p3+p4+p5+p6+p7;
```

cpparray

```
function A=cpparray(y,D,q,Delta,Gamma,dec)
% A=cpparray(y,D,q,Delta,Gamma,dec)
%
% Uses cpp to find all values of P(s,t,q). This function puts these
values
% in an upper triangular matrix A.

% Andrew Richens
n=length(y);
A=zeros(n,n);
for i=1:n
    for j=i:n
        A(i,j)=cpp(y,D,i,j,q,Delta(1:q,1:q),Gamma,dec);
    end
end
```

cppdiffs

```
function D=cppdiffs(y,q,nu,Delta,Gamma)
% D=cppdiffs(y,q,nu,Delta,Gamma)
%
% Creates a length 6 cell array with important values required in cpp.

% Andrew Richens
n=length(y);
D{1}(1:n)=myloggamma(((1:n)+nu)/2);
D{2}(1:n)=-1/2*(nu+(1:n));
D{3}(1:n)=-((1:n))/2*log(pi);
D{4}=-myloggamma(nu/2);
D{5}=-.5*log(det(Delta(1:q,1:q)));
D{6}=nu/2*log(Gamma);
```

cppeaks

```
function [out,range]=cppeaks(bigm,T,n)
% [out,range]=cppeaks(bigm,T,n)
%
% Performs the peak finding method. bigm is a cell array of vectors of
% positions of changepoints. The length of this array is the number of
% iterations performed. T is the vector of number of number of
iterations
% each point is found as a changepoint (this is the sum of bigm). n is
the
% length of the series.
```

```
% out is new values of T, after the centering of peaks. range is a
vector
% of midpoints between the peaks.

% Andrew Richens
l=length(bigm);
a=cpfindmax(T); % vector of local maxima of T
la=length(a);
range=zeros(1,la+1);
for i=2:la
    range(i)=(a(i-1)+a(i))/2;
end
range(la+1)=n;%A vector of midpoints of the peaks
out=cell(1,length(bigm));
for j=1:l              %each element of bigm (single iteration of
cpreal)
    count=1;
    for i=1:la           %each peak
        c=inrange(bigm{j},ceil(range(i)),floor(range(i+1))); %vector of
positions of bigm{j} that are in current range
        lc=length(c);
        if lc>0
            out{j}(count)=a(i);%Puts a value that is a relevant peak
into out{j}
            count=count+1;
        end
    end
end
```

cpplot

```
function C=cpplot(y,T,thresh)
% C=cpplot(y,T,thresh)
%
% Fids C, the vector of points with probability of being a changepoint
% higher than thresh

% Andrew Richens
n=length(y);
C=zeros(1,n);
k=1;
for i=1:n
    if T(i)>=thresh
        C(k)=i;
        k=k+1;
    end
end
C=C(1:k-1);
for i=1:length(C)
end
```

cppmatrix

```
function P=cppmatrix(y,A,LQ,q,lambda)
% P=cppmatrix(y,A,LQ,q,lambda)
```

```
%
% Creates a matrix where each row is the output of cpprob. The ith row
% assumes a changepoint at time i.

%Andrew Richens
n=length(y);
P=zeros(n);
for i=1:n-1
    [P(i,:)]=cpprob(y,q,i,A,LQ,lambda);
end
P(n,n) = 1;
```

cpprob

```
function Pr=cpprob(q,tau,A,Q,lambda)
% Pr=cpprob(q,tau,A,Q,lambda)
%
% Conditioning on a changepoint at time tau, finds the a vector of
% probabilities for every point grater than tau being a changepoint.
% Finds probabilities of changepoints for times t to n given a change
point

% Andrew Richens
B=zeros(q,n);
for i=1:q
    B(i,tau)=A{i}(tau,n)+(n-tau)*log(1-lambda)-Q(tau);
    B(i,tau+1:n)=log(lambda)-Q(tau)+A{i}(tau,tau:n-1)+Q(tau+1:n)+(0:n-
tau-1).*log(1-lambda);
end
B=exp(B);
B=sum(B,1);
Pr=sum(B,1);
Pr(tau+1:n)=Pr(tau+1:n)/sum(Pr(tau+1:n));
```

cpQQ

```
function QQ=cpQQ(y,t,q,LQ,A,lambda)
% QQ=cpQQ(y,t,q,LQ,A,lambda)
%
% A sub function of cpLQ2, this calculates specific values for that
% function.

% Andrew Richens
n=length(y);
for i=1:q
    B=[zeros(1,t-1) lambda*exp(A{i}(t,t:n-1)+LQ((t+1):n)-LQ(t+1)+log(1-
lambda).*(0:(n-1-t))) 0];
    b(i)=1/q*sum(B);
    c(i)=1/q*exp(A{i}(t,n)-LQ(t+1)+(n-t)*log(1-lambda));
end
QQ=log(sum(b)+sum(c));
```

cpreal

```
function R=cpreal(y,P)
% R=cpreal(y,P)
%
% Draws one sample from the posterior distribution of changepoints. R
is a
% vector of locations of changepoints.

% Andrew Richens
n=length(y);
tau=0;
k=0;
R=zeros(1,n-1);
while rand>P(tau+1,tau+1)
    cpr = cumsum(P(tau+1,tau+2:n));
    tau=sum(cpr<rand)+1+tau;
    if tau>=n-1
        break
    end
    k=k+1;
    R(k)=tau;
end
if ~k, R = [];
else R=R(1:k);
end
```

cprealits

```
function [T,cq]=cprealits(y,q,dk,lambda,m,thresh,dec,peaks)
%[T,cq]=cprealits(y,q,dk,lambda,m,thresh,dec,peaks)
% This function finds the location of changepoints in a time series y.
%
%   Inputs:
%   q is the maximum model order allowed
%
%   dk is a q-vector of the maximum possible value considered of the
%   regression coefficients, starting with the lowest order.
%
%   lambda is the parameter for the prior distribution on the number of
%   changepoints
%
%   m is the number of iterations, or samples from the posterior
%   distribution of changepoints.
%
%   thresh is the cutoff probability for a point to be considered a
%   changepoint
%
%   dec is the type of model to find changepoints in. Use ;Pn; for
%   polynomial, or 'Ar' for autoregressive
%
%   peaks turns the peak finding method on. Use 'On' or 'Off' The
default
%   is on.
%
%   Outputs:
%   T is a vector wit the posterior probability of a changepoint for
each
```

```
%   point in the series.
%
%   cq is a matrix where the ith column is the posterior distribution
for
%   the model order. There is one column for each segment found. The
first
%   values correspond to lower model order.

% Andrew Richens
warning off MATLAB:log:logOfZero
if nargin<8
    peaks='On';
end
Delta=zeros(q,q);
for i=1:q
    Delta(i,i)=cpdelta2(dk(i),y,1000);
end
[nu,Gamma]=boots(y,1000);
D=cppdiffs(y,q,nu,Delta,Gamma);
A=cell(1,q);
for i=1:q
    A{i}=cpparray(y,D,i,Delta,Gamma,dec);
end
n=length(y);
ncp=zeros(1,m);
LQ=cpLQ2(y,q,A,lambda);
P=cppmatrix(y,A,LQ,q,lambda);
T=zeros(1,length(y));
bigm=cell(1,m);
for i=1:m
    [R]=cpreal(y,P);
    bigm{i}=R;
    ncp(i)=length(R);
    for j=1:length(R)
        T(R(j))=T(R(j))+1;
    end
end
if  strcmp(peaks,'On')
    out=cppeaks(bigm,T,n);
    T=zeros(1,n);
    for i=1:m
        for j=1:length(out{i})
            T(out{i}(j))=T(out{i}(j))+1;
        end
    end
end
T=T/m;
C=cpplot(y,T,thresh);
cq=cporder(q,A,C,n);
```

inrange

```
function c=inrange(x,a,b)
% finds out the location in x of all values between a and b inclusive

% Andrew Richens
```

```
c=zeros(1,length(x));
j=0;
for i=1:length(x);
    if x(i)>a&&x(i)<b
        j=j+1;
        c(j)=i;
    end
end
c=c(1:j);
```

loclinbw2

```
function bw = loclinbw2(x,y,kertype,bwmethod)
% Bandwidth selection for local linear kernel regression.
% Estimates the optimal asymptotic mean integrated weighted squared
error.
%
% Assumptions: Homoscesdastic, bounded predictor support.
%
% Inputs: x = column vector of predictors.
%         y = column vector of responses.
%         kertype = kernel type:
%                   1 = Gaussian,
%                   2 = Epanechnikov.
%         bwmethod = bandwidth selection method:
%                    1 = rule-of-thumb,
%                    2 = direct plug-in.
%
% Output: bw = bandwidth.
%
% Reference: Ruppert, D., Sheather, S. J. and Wand, M. P. (1995),
% “An effective bandwidth selector for local least squares regression”,
% Journal of the American Statistical Association 90, 1257-1270.

% Dominic Lee
n = length(x);
a = min(x); b = max(x);
ba = b - a;
[xsort,xindex] = sort(x);
x = x(xindex); y = y(xindex);
switch kertype
    case 1, ck = 1 / (2 * sqrt(pi));
    case 2, ck = 15;
    otherwise, disp('Invalid or unsupported kernel type.'), return
end
Nstar = 5;
Nmax = max(min(floor(n/20),Nstar),1);
[N,beta,blklen,rss] = mallows(x,y,Nmax);
cumblklen = cumsum(blklen);
v = rss / (n - 5 * N);
switch bwmethod
    case 1
        m2 = zeros(n,1);
        for blk = 1:N
            if blk == 1, iblk = 1:cumblklen(blk);
            else        iblk = cumblklen(blk-1)+1:cumblklen(blk); end
```

```
            X = [ones(blklen(blk),1) x(iblk) x(iblk).^2];
            m2(iblk) = X * (beta(3:5,blk) .* [2;6;12]);
        end
        cm = mean(m2 .* m2);
    case 2
        m2 = zeros(n,1); m4 = zeros(n,1);
        for blk = 1:N
            if blk == 1, iblk = 1:cumblklen(blk);
            else         iblk = cumblklen(blk-1)+1:cumblklen(blk); end
            Iblk = ones(blklen(blk),1);
            X = [Iblk x(iblk) x(iblk).^2];
            m2(iblk) = X * (beta(3:5,blk) .* [2;6;12]);
            m4(iblk) = 24 * beta(5,blk) * Iblk;
        end
        cm24 = mean(m2 .* m4);
        if cm24 < 0
            switch kertype
                case 1, ck2 = 3 / (8 * sqrt(pi));
                case 2, ck2 = 315;
            end
        else
            switch kertype
                case 1, ck2 = 15 / (16 * sqrt(pi));
                case 2, ck2 = 787.5;
            end
        end
        vba = v * ba;
        g = ((ck2 * vba) / (abs(cm24) * n))^(1/7);
        alpha = .05;
        anew = a + alpha * ba; bnew = b - alpha * ba;
        m2g = locpolyreg(x,y,x,3,2,g,1,0);
        cm = mean(m2g .* m2g .* (x > anew & x < bnew));
        ck3 = 4 * (.5 + 2 * sqrt(2) - 4 * sqrt(3) / 3) / sqrt(2 * pi);
        h = ((ck3 * v * vba) / (cm * cm * n * n))^(1/9);
        [mh,dof] = locpolyreg(x,y,x,1,0,h,1,1);
        res = y - mh;
        v = sum(res .* res) / dof;
    otherwise
        disp('Invalid or unsupported bandwidth selection method.'),
return
end
bw = ((ck * v * ba) / (cm * n))^(.2);
```

loclinreg

```
function yfit = loclinreg(x,y,s,bw,kertype)
% Local linear kernel regression.

% Dominic Lee
n = length(x);
ns = length(s);
yfit = zeros(ns,1);
for i = 1:ns
    switch kertype
        case 1, K = normpdf(x,s(i),bw); % normal kernel
```

```
        case 2, K = .75 * (1 - ((x - s(i)) / bw).^2) / bw; %
Epanechnikov kernel
    end
    xdev = x - s(i);
    xdevK = xdev .* K;
    s0 = sum(K);
    s1 = sum(xdevK);
    s2 = sum(xdev .* xdevK);
    w = ((s2 - s1 * xdev) .* K) / (s0 * s2 - s1 * s1);
    yfit(i) = w' * y;
end
```

myloggamma

```
function lg=myloggamma(x)
% lg=myloggamma(x)
% Finds the log of the gamma function applied to x. Using log(gamma(x))
is
% limited to values up to 171, anything higher returns inf. This
function
% circumvents this error.

% Andrew Richens
l=length(x);
a=x-floor(x);
lg=zeros(length(x),1);
for i=1:length(x)
    a=x(i)-floor(x(i));
    n = ceil(x(i));
    if a, y = a:n-1+a; y(n) = gamma(a);
    else  y =1:n; y(n) = 1;
    end
    lg(i)=sum(log(y));
end
```

References

Barry, D. and Hartigan, J.A. (1992). Product partition models for change point problems. *The Annals of Statistics*, **20**, 260-269.

Barry, D. and Hartigan, J.A. (1993). A Bayesian analysis of change point problems. *Journal of the American Statistical Association*, **88**, 309-319.

Braun, J. V. and Muller, H. G. (1998). Statistical models for DNA sequence segmentation. *Statistical Science*, **13**, 142-162.

Brockwell, P.J. and Davis, R.A. (2002). *Introduction to Time Series and Forecasting* (2nd ed.). New York: Springer-Verlag.

Chatfield, C. (2003). *The Analysis of Time Series: An Introduction* (6th ed.) Chapman & Hall/CRC.

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.

Chib, S. (1998). Estimation and comparison of multiple change-point models. *Journal of Econometrics*, **86**, 221-241.

Davis, R. A., Lee, T. C. M., and Rodriguez-Yam, G. A. (2004). Structural breaks estimation for non-stationary time series. Available at http://www.cireq.umontreal.ca/activites/050520/papers/Davis.pdf

Fearnhead, P. (2005). Exact Bayesian curve fitting and signal segmentation. *IEEE Transactions on Signal Processing*, **53**, 2160-2166.

Fearnhead, P. (2006). Exact and efficient Bayesian inference for multiple changepoint problems. *Statistics and Computing*, **16**, 203-213.

Fearnhead, P. and Clifford, P. (2003). Online inference for well-log data. *Journal of the Royal Statistical Society, Series B.* **65**, 887-899.

Fearnhead, P. and Liu, Z. (2007). On-line inference for multiple changepoint problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **69**, 589-605.

Johnson, T.D., Elashoff, R.M. and Harkema, S.J. (2003). A Bayesian changepoint analysis of electromyographic data: detecting muscle activation patterns and associated applications. *Biostatistics*, **4**, 143-164.

Lee, D.S., Russell, G., Reale, M., Tunnicliffe-Wilson, G. and Roscoe, J. (2005). Analysing physiological signals from preterm babies. *2nd Workshop on Hidden Markov Models and Complex Systems*, 5-8 December 2005, Wellington, New Zealand.

Ó Ruanaidh, J.J.K. and Fitzgerald, W.J. (1996). *Numerical Bayesian Methods Applied To Signal Processing.* New York: Springer.

Stephens, D. A. (1994). Bayesian retrospective multiple-changepoint identification. *Applied Statistics*, 43, 159-178.

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.

Zhao, X., Hou, Q., Lee, D., Reale, M., Scarrott, C., Russell, G., MacDonald, A. and Zahari, M. (2007). A comparison between alternative volatility estimations: Application on blood oxygen concentration of preterm infants. *International Congress on Modelling and Simulation*, 10-13 December 2007, Christchurch, New Zealand.