Reversible-jump MCMC methods in Bayesian statistics

Jialin Ai
200662838

Supervised by Jochen Voss

Submitted in accordance with the requirements for the
module MATH5871M: Dissertation in Statistics
as part of the degree of

Master of Science in Statistics with Applications to Finance

The University of Leeds, School of Mathematics

August 2012

The candidate confirms that the work submitted is his/her own and that appropriate credit has been given where reference has been made to the work of others.

Abstract

This dissertation is about the reversible-jump Markov chain Monte Carlo (RJMCMC) method in Bayesian statistics and its application to solving the change-points problem. A change-points problem is to determine the parameters of a step model with flexible number of steps. This model is widely used in different fields. Thus many application, such as change-points in genes determining and flexible number's variables selection in regression, benefit from RJMCMC.

Firstly I introduce Markov chain Monte Carlo (MCMC) and the Metropolis-Hastings (M-H) Algorithm. I explain the concept "reversibility", give the relationship between "reversible" and "stationary", and then prove that the Markov chain generated by M-H is reversible and stationary.

Secondly, I introduce Bayesian estimation and realise it with M-H algorithm. I gave a simple and clear example and implement it in R with Bayesian estimates algorithm.

Next, I introduced the subspaces assumption of reversible-jump Markov chain, and the basic form of the acceptance probability jumping between subspaces of differing dimensionality based on Green (1995).

Finally, I try to solve a change-points problem of coal-mining disasters data. I calculate the acceptance probabilities of different move types, combine the results and the Metropolis-Hastings algorithm into an R program. And I also analyse the reversible-jump Markov chain generated by the program.

Contents

List of Figures

Declaration

No portion of the work referred to in this dissertation has been submitted in support of an application for another degree or qualification of this or any other university or institution of learning.

In the attached submission I have not presented anyone else's work as my own. Where I have taken advantage of the work of others, I have given full acknowledgement. I have read and understood the University's published rules on plagiarism and also any more detailed rules specified at School or module level. I know that if I commit plagiarism I can be expelled from the University and that it is my responsibility to be aware of the University's regulations on plagiarism and their importance.

I re-confirm my consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

I confirm that I have declared all mitigating circumstances that may be relevant to the assessment of this piece of work and that I wish to have taken into account. I am aware of the School's policy on mitigation and procedures for the submission of statements and evidence of mitigation.

Signed: _____

Chapter 1

Introduction

Definition 1. [1] *A stochastic process $X = (X_n)_{n\in\mathbb{N}_0}$ with values in a set S is called a **Markov chain**, if*

$$P(X_n \in A_n | X_{n-1} \in A_{n-1}, X_{n-2} \in A_{n-2}, \ldots, X_0 \in A_0) = P(X_n \in A_n | X_{n-1} \in A_{n-1}) \tag{1.1}$$

*for all $A_0, A_1, \ldots, A_n \subseteq S$ and all $n \subseteq \mathbb{N}$, i.e. if the distribution of X_n depends only through X_{n-1}. The set S is called the **state space** of X. The distribution of X_0 is called the **initial distribution** of X. Often X_0 is deterministic, i.e. $P(X_0 = x) = 1$ for some $x \in S$; in this case x is called the **initial value** or **starting point** of X. The index n is typically interpreted as **time**.*

Many statistical problems, such as inference about curves, surfaces and imagines, involve a discrete choice between a set of models. The dimension of the models is not known. And one will not be convinced of the assumption of models with parameter vectors in changeless dimension, unless there is an objective theory or common sense to support it. In these cases, a model with fixed dimension is not sufficient to get the best answer.

Facing the varying dimension, we may assume a group of models with the same number of parameters first. And then estimate the values of parameters under this assumption. A plausible model in this model group can be detected by Markov chain Monte Carlo method. We can repeat this process and get a good model under each assumption. At last we have a sequence of good models with different number of parameters.

But the problem is which model is the "best" among this results? It is difficult to compare the models with different number of parameters. Which one should be rejected and which one should retained are beyond the estimates processes described above. We need a new method to jump from one dimensionality to another dimensionality. That is why we need reversible jump Markov chain Monte Carlo method.

Green (1995) has added weight to the assertion that "a Bayesian approach is attractive for such a problem". He introduces a novel method capable of jumping between subspaces of differing dimension. This extends the scope of the Metropolis-Hastings method.

[1] Voss (2011), pp.44

In this article, I introduce the Metropolis-Hastings algorithm and Bayesian Estimates method. According to the reversible jump Markov chain Monte Carlo algorithm in Green (1995), I realise it in R to solve "the change-point problem". It is a problem concerned with the identification of the point in a time series at which some change occurs.

"R is a system for statistical computation and graphics. The core of R is an interpreted computer language which allows branching and looping as well as modular programming using functions. The R distribution contains functionality for a large number of statistical procedures. There is also a large set of functions which provide a flexible graphical environment for creating various kinds of data presentations."[2]

And all the R code for this project can be downloaded from
`http://www1.maths.leeds.ac.uk/~voss/projects/2011-RJMCMC/`.

In Chapter 2, I introduce the Metropolis-Hastings algorithm and prove the reversibility of Markov chain generated by Metropolis-Hastings method.

In Chapter 3, the Bayesian parameter estimates method is shown.

In Chapter 4, a reversible jump Markov chain Monte Carlo method in Green (1995) is introduced.

In Chapter 5, I explain change-point problem. After calculating the acceptance probability of each move type, I realise the algorithms in R and finally analyse the results.

[2]R-project website (1998-)

Chapter 2

Markov Chain Monte Carlo Methods

In this chapter, the Metropolis-Hastings (M-H) method, a popular algorithm for Markov chain Monte Carlo methods, will be introduced.

The Metropolis-Hastings method first appears in the form of the simulated annealing algorithm introduced by Metropolis et al.(1953). Readers can go for more details about the Metropolis-Hastings in Robert & Casella (2004).

In the first section, the algorithm for the Metropolis-Hastings method will be shown and the "reversibility" of the M-H method is proved. In the second section, we talk about the definition of reversibility, the relation between the reversibility of a Markov chain and the stationarity of probability density for this Markov chain, and the proof that the M-H method is a proper algorithm generating a stationary Markov chain.

2.1 The Metropolis-Hastings method

Definition 2. *A **Markov Chain Monte Carlo** (MCMC) method for the simulation of a distribution f is any method producing an ergodic Markov chain (X_t) whose stationary distribution is f.*[1]

The Metropolis-Hastings Method is a popular method to generate a new Markov chain with the target stationary distribution which can be applied in many situation without much constraint.

Metropolis-Hastings Algorithm:

Input:
a target density π
a transition density $q(x, y)$
$x_0 \in \mathbb{R}$ with $\pi(x_0) > 0$

[1] Robert & Casella (2004), pp.268

Output:
a Markov chain with stationary density π

for $n = 1, 2, 3 \ldots$
generate $y_n \sim q(x_{n-1}, \cdot)$
generate $U_n \sim U[0, 1]$
if $U_n \leqslant \alpha(x_{n-1}, y_n)$ $\quad \Big(\text{where } \alpha(x, y) = \min\Big(\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}, 1\Big)\Big)$
then $x_n \longleftarrow y_n$ (accepted)
else $x_n \longleftarrow x_{n-1}$ (rejected)

First we show that the transition probabilities for the Markov chain generated by Metropolis-Hastings algorithm satisfy $p(A, B) = p(B, A)$ for all $A, B \subseteq U$, where $p(A, B) = P(x_{n-1} \in A, x_n \in B)$.
For all $B \subseteq U$ and $x \in U$ with $x \notin B$, we have

$$\begin{aligned} q(x, B) =& P(x_n \in B | x_{n-1} = x) \\ =& P(y_n \in B | x_{n-1} = x) \cdot P(\text{accepted}) \\ =& \int_B q(x, y) \cdot \alpha(x, y)\, dy \\ =& \int_B q(x, y) \cdot \min\Big(\frac{\pi(y)q(y,x)}{\pi(x)q(x,y)}, 1\Big)\, dy \\ =& \int_B \min\Big(\frac{\pi(y)q(y,x)}{\pi(x)}, q(x, y)\Big)\, dy \end{aligned}$$

Thus, for all $A, B \subseteq U$ with $A \cap B = \emptyset$, we get

$$\begin{aligned} p(A, B) =& \int_A \pi(x) \cdot q(x, B)\, dx \\ =& \int_A \int_B \pi(x) \cdot \min\Big(\frac{\pi(y)q(y,x)}{\pi(x)}, q(x, y)\Big)\, dy\, dx \\ =& \int_A \int_B \min(\pi(y)q(y, x), \pi(x)q(x, y))\, dy\, dx && \text{(Fubini's Theorem)} \\ =& \int_B \int_A \min(\pi(y)q(y, x), \pi(x)q(x, y))\, dx\, dy && \text{(This is symmetric in } x,y) \\ =& \int_B \int_A \min(\pi(x)q(x, y), \pi(y)q(y, x))\, dy\, dx \\ =& p(B, A) && (*) \end{aligned}$$

In the third step, Fubini's Theorem[2] is used: "Let $(X, \mathcal{A}, \mu)$ and $(Y, \mathcal{B}, \nu)$ be σ-finite measure spaces, and let $(X \times Y, \mathcal{A} \times \mathcal{B}, \mu \times \nu)$ be the product measure space. If f is a nonnegative

[2] Knapp (2005), pp.271

measurable function on $X \times Y$, then $\int_Y f(x,y)\,d\nu(y)$ and $\int_X f(x,y)\,d\mu(x)$ are measurable, and

$$\begin{aligned}\int_{X\times Y} f\,d(\mu\times\nu) &= \int_X \Big[\int_Y f(x,y)\,d\nu(y)\Big]\,d\mu(x)\\ &= \int_Y \Big[\int_X f(x,y)\,d\mu(x)\Big]\,d\nu(y).\text{"}\end{aligned}$$

For general $A, B \subseteq U$, we get

$$\begin{aligned}p(A,B) =& p(A\setminus B, B) + p(A\cap B, B)\\ =& p(A\setminus B, B) + p(A\cap B, A\cap B) + p(A\cap B, B\setminus A)\\ =& p(B, A\setminus B) + p(A\cap B, A\cap B) + p(B\setminus A, A\cap B) \quad \text{(According to property (*))}\\ =& p(B, A\setminus B) + p(B, A\cap B)\\ =& p(B, A)\end{aligned}$$

If a transition density $q(x,y)$ and an invariant density π exits, then $p(A,B) = p(B,A), \forall A, B$ i.e. $\int_A \int_B \pi(x)q(x,y)\,dydx = \int_B \int_A \pi(y)q(y,x)\,dxdy, \forall A, B$. We will see in the next section, that this property, called reversibility, is enough so that the M-H algorithm can generate a stationary distribution.

2.2 Reversibility

To explain the meaning of reversibility, we give some formal definitions of different concepts. But to avoid technical complication, we do not explain all the technical details here.

Definition 3. [3] *A **transition kernel** is a function K defined on $\mathcal{X} \times \mathcal{B}(\mathcal{X})$ (where $\mathcal{B}(\mathcal{X})$ is the Borel σ-algebra of $\mathcal{X}$, i.e. the second argument of K is a subset of $\mathcal{X}$) such that*

(i) $\forall x \in \mathcal{X}$, $K(x,\cdot)$ *is a probability measure;*

(ii) $\forall A \in \mathcal{B}(\mathcal{X})$, $K(\cdot, A)$ *is measurable.*

When $\mathcal{X}$ is discrete, the transition kernel simply is a (transition) matrix K with elements $P_{xy} = P(X_n = y | X_{n-1} = x), x, y \in \mathcal{X}$.

In the continuous case, the kernel also denotes the conditional density $K(x,x')$ of the transition $K(x,\cdot)$; that is, $P(X \in A|x) = \int_A K(x,x')dx'$.

For example, in section 2.1, the transition density q is the transition kernel. The meaning of $p(x,y)$ and that of $q(x,y)$ are different: $p(x,y) = P(x_{n-1} = x, x_n = y)$ is a joint probability,

[3] Robert & Casella (2004), pp.208

while $q(x,y) = P(x_n = y|x_{n-1} = x)$ is a conditional probability. And we have $p(x,y) = \pi(x)q(x,y)$ (Bayes' Rule).

Definition 4. [4] *A Markov chain with transition kernel K satisfies the **detailed balance** condition if there exists a function f satisfying*

$$f(x)K(x,y) = f(y)K(y,x) \tag{2.1}$$

for every (x,y).

"The balance detailed condition (2.2) express an equilibrium in the flow of the Markov chain, namely that the probability of being in x and moving to y is the same as the probability of being in y and moving back to x."[5]

Definition 4a. *The alternative form of the detailed balance condition is:*

$$\int_A \int_B f(x)K(x,y)\,dydx = \int_B \int_A f(y)K(y,x)\,dxdy \tag{2.2}$$

When f is a density π, the detailed balance condition implies that the chain is reversible.

Definition 5. *Suppose that a Markov chain with transition density q satisfies the detailed balance condition with a probability density function π, i.e. $q(y,x)\pi(y) = q(x,y)\pi(x)$. Then the chain is π-**reversible**.*

With the existence of q and invariant density π, the detailed balance and reversibility are the same property.

In section 2.1, we have proved that the Monte Carlo generated by M-H algorithm is reversible.

Definition 6. [6] *A probability density $\pi : \mathbb{R} \to [0,\infty)$ is a **stationary** density for this Markov chain, if it satisfies $\int_S \pi(x)q(x,y)dx = \pi(y)$ for all $y \in S$.*

Theorem 1. *If X is a π-reversible Markov chain, then π is a stationary distribution of X.*

proof. If X satisfies the detailed balance condition for a probability vector π, x and y are the states in different time, then we have

$$\int_U \pi(x)q(x,y)\,dx = \int_U \pi(y)q(y,x)\,dx = \pi(y)\int_U q(y,x)\,dx = \pi(y)$$

We now have solved the problem of constructing an MCMC method. The Markov chain generated by M-H algorithm is stationary.

[4] Robert & Casella (2004), pp.230
[5] Robert & Casella (2004), pp.230
[6] Voss (2011), pp.49

But note that the detailed balanced condition "is not necessary for f to be a stationary measure associated with transition kernel K", it just provides "a sufficient condition that is often easy to check and that can be used for most MCMC algorithms."[7]

[7] Robert & Casella (2004), pp.230

Chapter 3

Bayesian Parameter Estimates

3.1 Bayesian method

"In statistics, Bayesian inference is a method of inference in which Bayes' rule is used to update the probability estimate for a hypothesis as additional evidence is learned."[1] In a Bayesian model, we can make a more accurate estimation of a distribution by studying the information from data.

A model can be classified into two categories: parametric and non-parametric. Only the estimation of parametric models will be discussed here. Apart from a basic assumption of a statistical model, the estimation of the parameter θ of the model (in some cases, there is more than one parameter) is sufficient to estimate the distribution.

Assuming the parameter θ of the statistical model is random itself, it was treated as a random variable of which the probability can be estimated and adjusted. Before adding information from more data, the distribution of θ is called the prior distribution which can described by a density $p(\theta)$. The conditional distribution of θ, given the data X is called the posterior distribution which can be described by a density $p(\theta|x)$.

Let $p(x|\theta)$ be a density of the distribution of the data X depending on θ in the model. By Bayes' rule, the density of the posterior distribution can be calculated from

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{\int p(x|\tilde{\theta})p(\tilde{\theta})\, d\tilde{\theta}}. \tag{3.1}$$

3.2 Example of Bayesian estimates

An example is given now to explain the Bayesian parameter estimates. I analysed and realised it in a computer program by myself. The parameter to estimate in the example is the mean μ of the bivariate normal distribution. We have the values of data from X_1 to X_m, their density given μ and the prior distribution of μ. We are going to estimate μ by calculating a sample of

[1] Wikipedia, Bayesian inference (2012)

size n from the posterior distribution of μ.

Example:

Parameter: $\mu \in \mathbb{R}^2$
Prior distribution: μ is uniformly distributed on $R = [-10, 10] \times [-10, 10]$.
$p(\mu) = \frac{1}{|R|} \cdot \mathbb{1}_R(\mu)$ i.e. $p(\mu) = 0.0025$ when $\mu \in R$ and $p(\mu) = 0$ elsewhere
Data: $X_1, X_2, \ldots, X_m \sim \mathcal{N}_2(\mu, I_2)$ $i.i.d.$ I_2 is the identity matrix in $\mathbb{R}^2$.
$p(x_1, \ldots, x_m|\mu) = \prod_{i=1}^{m} \frac{1}{2\pi} e^{-\frac{(x_i-\mu)^T(x_i-\mu)}{2}}$

We now realise Bayesian estimation with Metropolis-Hastings Algorithm. The posterior distribution of the parameter is treated as the target density. For each transition step, the prior distribution of parameter is exactly the posterior distribution gotten from the last transition step. We have proved in Chapter 2 that M-H algorithm can generate a stationary Markov chain. Thus after enough estimate steps, we can get a sequence of estimated values of the parameter under the stationary posterior distribution.

Bayesian parameter estimates are obtained using the M-H Algorithm:

Input:
target density $\pi(\mu) = \frac{1}{Z} p(x_1, \ldots, x_m|\mu)p(\mu)$
where Z is the constant value of $\int p(x_1, \ldots, x_m|\tilde{\mu})p(\tilde{\mu})\, d\tilde{\mu}$
transition density $q(\mu, \tilde{\mu}) = \frac{1}{2\pi\sigma} e^{-\frac{1}{2\sigma^2}(\mu-\tilde{\mu})^T(\mu-\tilde{\mu})}$
$\mu_0 = (0, 0)^T \in \mathbb{R}^2$ with $\pi(\mu_0) = 0.0025 > 0$

Output:
a Markov chain with stationary density $\pi(\mu)$

for $n = 1, 2, 3 \ldots$
generate $\tilde{\mu}_n = \mu_{n-1} + \varepsilon_n$ where $\varepsilon_n \sim \mathcal{N}((0, 0)^T, \sigma^2 I)$
generate $U_n \sim U[0, 1]$
if $U_n \leqslant \alpha(\mu_{n-1}, \tilde{\mu}_n)$ where

$$\alpha(\mu, \tilde{\mu}) = \min\left(\frac{p(x_1, \ldots, x_m|\tilde{\mu})p(\tilde{\mu})}{p(x_1, \ldots, x_m|\mu)p(\mu)}, 1\right) \tag{3.2}$$

then $\mu_n \longleftarrow \tilde{\mu}_n$ (accepted)
else $\mu_n \longleftarrow \mu_{n-1}$ (rejected)

Here we explain how to get the acceptance probability (3.2).

$$\begin{aligned}
\alpha(\mu, \tilde{\mu}) &= \min\Big(\frac{\pi(\tilde{\mu})q(\tilde{\mu}, \mu)}{\pi(\mu)q(\mu, \tilde{\mu})}, 1\Big) \\
&= \min\Big(\frac{\frac{1}{Z}p(x_1, \ldots, x_m|\tilde{\mu})p(\tilde{\mu}) \cdot \frac{1}{2\pi\sigma}e^{-\frac{1}{2\sigma^2}(\tilde{\mu}-\mu)^T_{1\times 2}(\tilde{\mu}-\mu)_{2\times 1}}}{\frac{1}{Z}p(x_1, \ldots, x_m|\mu)p(\mu) \cdot \frac{1}{2\pi\sigma}e^{-\frac{1}{2\sigma^2}(\mu-\tilde{\mu})^T_{1\times 2}(\mu-\tilde{\mu})_{2\times 1}}}, 1\Big) \\
&= \min\Big(\frac{p(x_1, \ldots, x_m|\tilde{\mu})p(\tilde{\mu})}{p(x_1, \ldots, x_m|\mu)p(\mu)}, 1\Big)
\end{aligned}$$

R rounding errors can be caused by machine calculation. When a denominator and a numerator are close to each other, the number of the quotient is insensitive to the small changes of them. Since the result of the division by machine is usually an approximate value which is called a truncated value or a rounding value of the original real quotient, the small changes of the quotient, i.e. the digital tail of the quotient is easily truncated. The information reserved in the truncated tail can be neglected. In contrast, this missing information of ratio will survive in the difference of log values of denominators and numerators. By taking the exponent of the difference we can make the different obvious again. And the result will be more close to the real value.

So to decrease the error, we realise it in a log form:

$$\begin{aligned}
\alpha(\mu_{n-1}, \tilde{\mu}_n) &= \min\Big(\frac{p(x_1, \ldots, x_m|\tilde{\mu}_n)p(\tilde{\mu}_n)}{p(x_1, \ldots, x_m|\mu_{n-1})p(\mu_{n-1})}, 1\Big) \\
&= \min\Big(\frac{\prod_{i=1}^{m}\frac{1}{2\pi}e^{-\frac{(x_i-\tilde{\mu}_n)^2}{2}} \cdot \frac{1}{|R|} \cdot \mathbb{1}_R(\tilde{\mu}_n)}{\prod_{i=1}^{m}\frac{1}{2\pi}e^{-\frac{(x_i-\mu_{n-1})^2}{2}} \cdot \frac{1}{|R|} \cdot \mathbb{1}_R(\mu_{n-1})}, 1\Big) \\
&= \min\Big(\frac{\prod_{i=1}^{m}\frac{1}{2\pi}e^{-\frac{(x_i-\tilde{\mu}_n)^2}{2}} \cdot \mathbb{1}_R(\tilde{\mu}_n)}{\prod_{i=1}^{m}\frac{1}{2\pi}e^{-\frac{(x_i-\mu_{n-1})^2}{2}} \cdot \mathbb{1}_R(\mu_{n-1})}, 1\Big) \\
&\quad (\mathbb{1}_R(\tilde{\mu}_n) \neq 0 \text{ otherwise it will be rejected \& } \mathbb{1}_R(\mu_{n-1}) \neq 0 \text{ because it was accepted.}) \\
&= \exp(\min(\sum_{i=1}^{m}\log(\frac{1}{2\pi}) - \sum_{i=1}^{m}\frac{(x_i-\tilde{\mu}_n)^2}{2} - (\sum_{i=1}^{m}\log(\frac{1}{2\pi}) - \sum_{i=1}^{m}\frac{(x_i-\mu_{n-1})^2}{2}), 0)) \\
&= \exp(\min(-\sum_{i=1}^{m}\frac{(x_i-\tilde{\mu}_n)^2}{2} + \sum_{i=1}^{m}\frac{(x_i-\mu_{n-1})^2}{2}, 0)).
\end{aligned}$$

This is all theoretical calculation required in the algorithm. We will implement in R.

1. Before estimation, we create a synthetic dataset including (i) the "real" value of the μ, (ii) the sample data generating from the distribution with the certain μ and (iii) the mean of samples i.e. the unbiased statistic of μ.

   Firstly, we generate a certain μ randomly in R. In order to show as much as information, a

μ with X-coordinate value near zero and Y-coordinate value far away from zero is chosen:

```
# value of mu
mu<-runif(2,min=-10,max=10)

> mu
[1] -0.363120  7.674977
```

Take m samples $X_1, X_2, \ldots, X_m \sim \mathcal{N}_2(\mu, I_2)$:

```
# value from x[1] to x[m]
m=100
x_axis <- rnorm(m,mean=mu[1],sd=1)
y_axis <- rnorm(m,mean=mu[2],sd=1)
x <- matrix (c(x_axis,y_axis),nrow=m,ncol=2,byrow=FALSE)
```

The coordinates of sample mean are

```
> mean(x[,1])
[1] -0.3420167
> mean(x[,2])
[1] 7.510462
```

2. Now we start to estimate μ:

   Set log value of the conditional distribution $\ln(p(x_1, \ldots, x_m|\mu)) = \sum_{i=1}^{m} \ln(\frac{1}{2\pi} e^{-\frac{(x_i-\mu)^T(x_i-\mu)}{2}})$:

```
# value of log(p(x|mu))
log_p_condition <- function(x,mu){
    sum=0
    for (i in 1:m){
        sum=sum+log(1/(2*pi))-0.5*t(x[i,]-mu)%*%(x[i,]-mu)
    }
    return(sum)
}
```

   Set log value of the prior distribution $\ln(p(\mu)) = \ln(\frac{1}{|R|} \cdot \mathbb{1}_R(\mu))$:

```
#value of log(p(mu))
log_p_prior <- function(mu){
    if(mu[1]>=-10 && mu[1]<=10 && mu[2]>=-10 && mu[2]<=10)
      {return (log(0.0025))}
    else return (-Inf)
}
```

   Determine the α acceptance probability:

```
# the log value of non.normalised target density
log_pi.Z <- function(mu){
    return(log_p_condition(x,mu)+log_p_prior(mu))
```

```
}

# the acceptance probability
alpha <- function(mu,mu_tilde){
  return(exp(log_pi.Z(mu_tilde)-log_pi.Z(mu)))
}
```

A trick has been used in R code to avoid calculating α. Instead, we only calculate the "alpha" where $\alpha = \min(\texttt{alpha}, 1)$. Generating U from $U[0,1]$, $P(U < 1) = 1$.

$$\begin{aligned} P(U < \alpha) &= P((U < \texttt{alpha}) \wedge (U < 1)) \\ &= P(U < \texttt{alpha}) \cdot P(U < 1) \qquad \text{(For alpha is independent of 1)} \\ &= P(U < \texttt{alpha}) \end{aligned}$$

Thus, we can compare U to alpha directly and has the same result.

The function of the μ estimating process :

```
#generate paths from the MH process
MH <- function(n,mu,sigma){
    path <- matrix(0,nrow=n,ncol=2)
    for (i in 1:n){
        mu_tilde <- rnorm (2,mu,sigma)
        U <- runif (1)
        if (U < alpha(mu,mu_tilde)){
           mu <- mu_tilde
        }
        path[i,] <- mu
    }
    return(path)
}
```

We took point $(0, 0)$ as a initial value. We took $\sigma = 1$, $\sigma = 0.1$, $\sigma = 0.01$ respectively and estimated μ for 3000 times. The sequence of points shows the estimating process.

The result is shown in figure 3.1. The left column is the estimates for X-axis of μ, and the right column is for Y-axis. The two pictures on top illustrate estimates with $\sigma = 1$. The moves are very fast and the stable estimate processes are attained soonest. The two pictures at the bottom show estimates with $\sigma = 0.01$. The amplitudes for every jump are the smallest on average among three rows. They take the longest time to attain the stable estimate processes, and the paths are the smoothest.

Comparing the different σ estimation, the bigger σ make the jump much farther but rough. The smaller σ make the estimation process slower but more accurate.

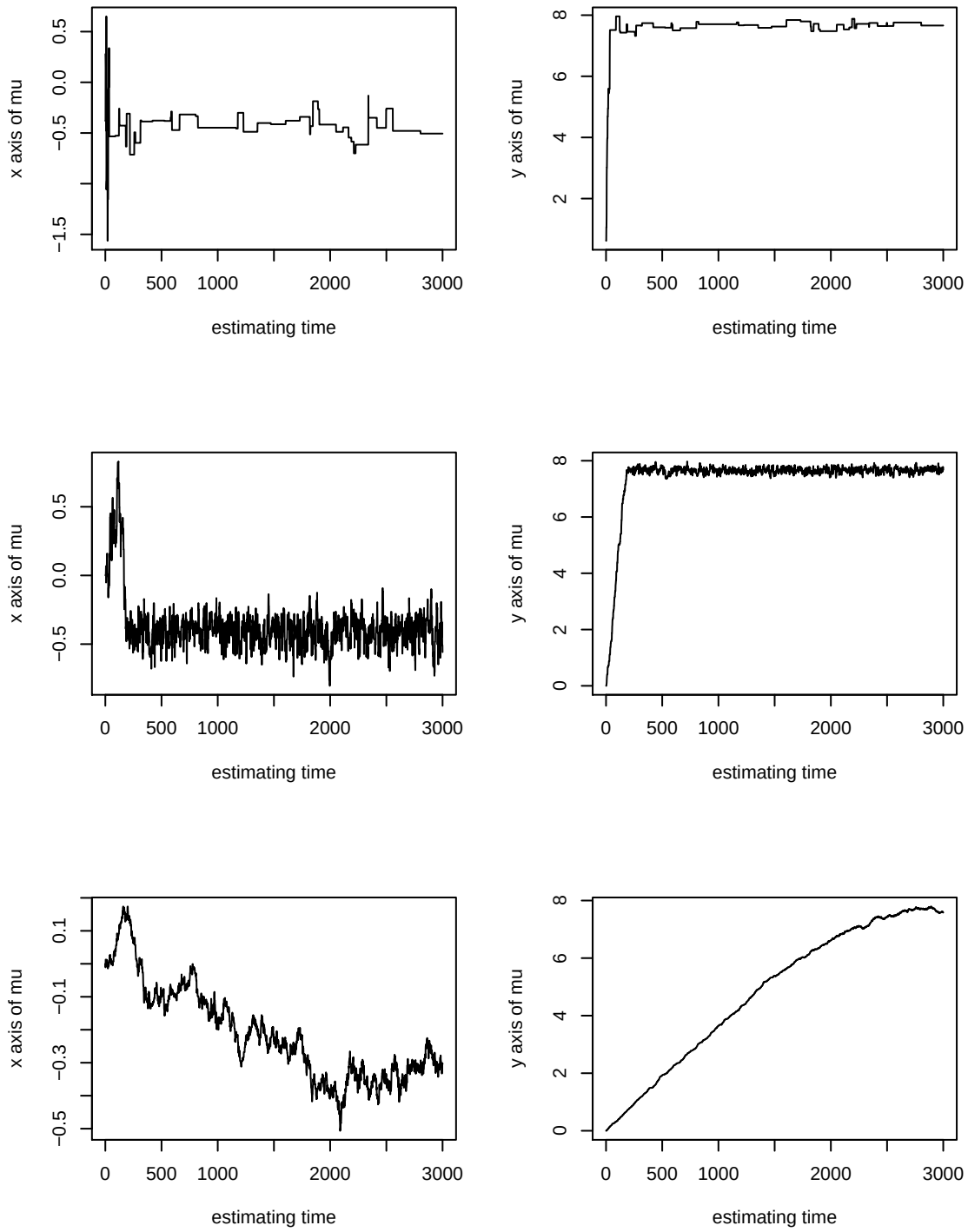


Figure 3.1: M-H Markov chain with burn-in period. Left: μ_1*(real value is* -0.363120*); Right:* μ_2*(real value is* 7.674977*). Top: proposal deviation is* $\sigma = 1$*; Center:* $\sigma = 0.1$*; Bottom:* $\sigma = 0.01$*. Starting point is* $(0, 0)$.

3. "Burn-in" is a colloquial term that describes the practice of throwing away some iterations at the beginning of an MCMC run.[2] From the figures above, the beginning unstable segment of each MCMC estimation process is the "burn-in" period.

   Different transition density will make this process shorter or longer. Besides, we can find some initial estimating value are better than the others. Proper starting point will shorten the burn-in time. For example on the case above, the sample mean is a good choice as starting point, since it is an unbiased statistic of the normal distribution mean parameter μ. The stable estimate will be directly attained at the very beginning. The figure 3.2 display the more important parts in estimate processes without burn-in periods.

   Besides choosing proper starting value, there are many other ways to let the estimates result get rid of the influence of burn-in period or to make the burn-in period shorter. One of the most important ways is to have longer the estimating time. The longer time it takes, the variance of the estimating values become smaller. And the Markov chain is getting closer to being stationary.

   In figure 3.3, the 2-dimensional paths of μ are demonstrated. The left column's pictures corresponding figure 3.1 include the burn-in period while the right column's exclude it corresponding figure 3.2.

A more complex example is given in section 5.4 where we can get deeper understanding about the Bayesian estimates and burn-in period.

[2] Geyer (2010), pp.19

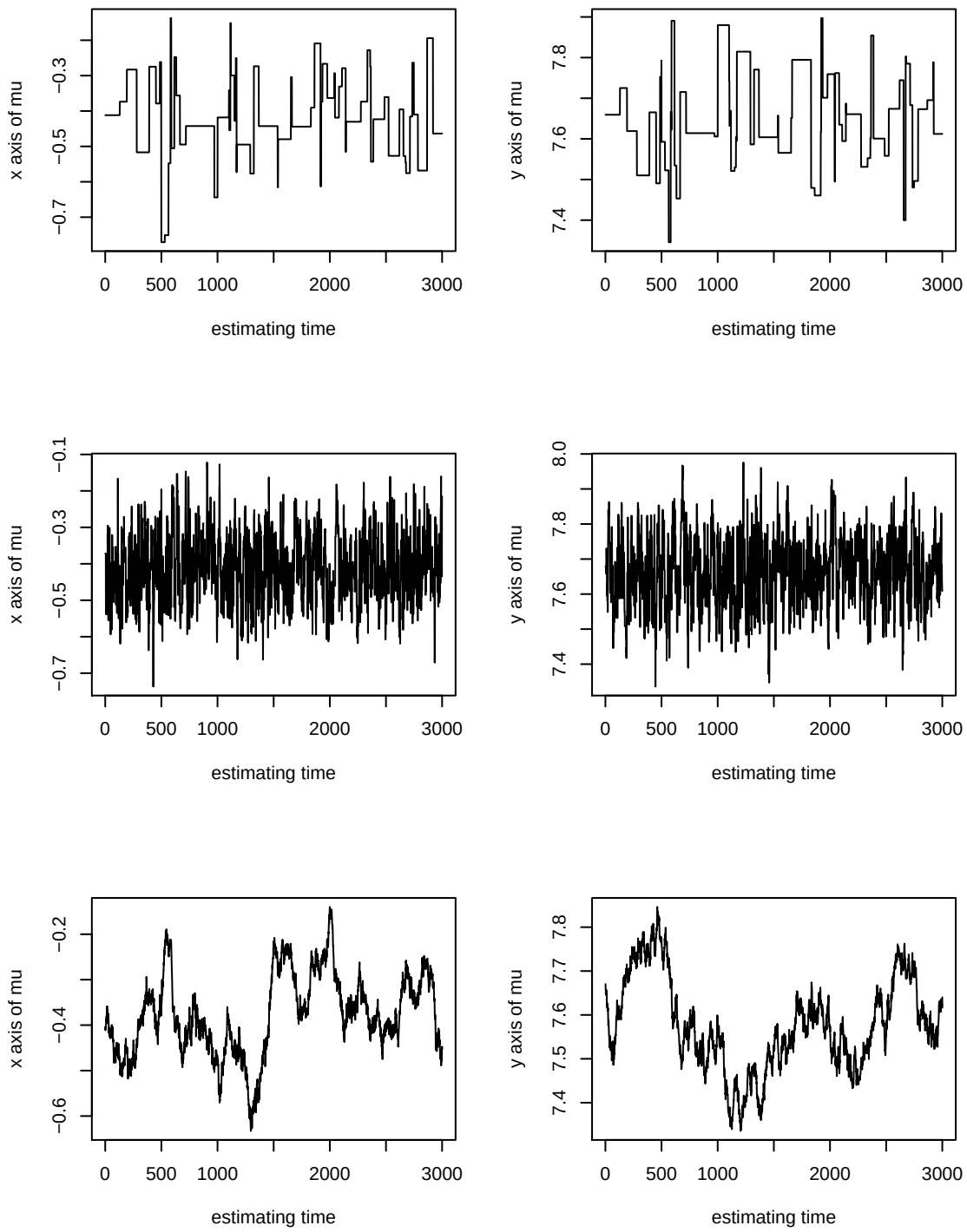


Figure 3.2: M-H Markov chain without burn-in period. Left: μ_1*(real value is* -0.363120*); Right:* μ_2*(real value is* 7.674977*). Top: proposal deviation is* $\sigma = 1$*; Center:* $\sigma = 0.1$*; Bottom:* $\sigma = 0.01$*. Starting point is* $(-0.3420167, 7.510462)$.

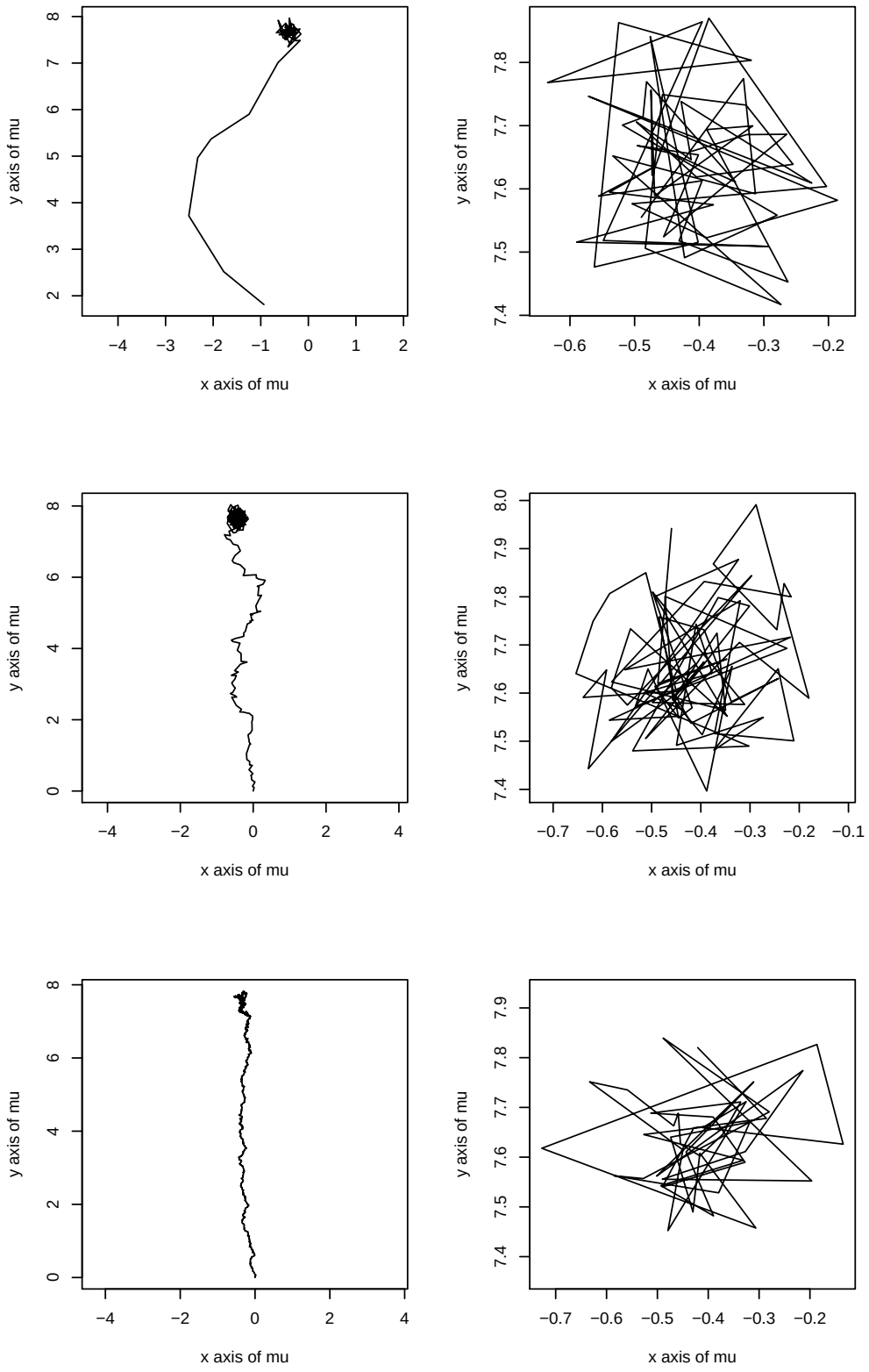


Figure 3.3: The paths of the M-H Markov chains. Left: with burn-in period; Right: without burn-in period. Top: proposal deviation is $\sigma = 1$*; Center:* $\sigma = 0.1$*; Bottom:* $\sigma = 0.01$.

Chapter 4

Reversible jump Markov Chain Bayesian model determination

We introduced the Bayesian model used in Metropolis-Hastings Markov chain Monte Carlo computation in last chapter. Green (1995) applies it to a varying-dimension problem and creates a novel method capable of reversible jumping between subspaces of differing dimensionality.

It solves the difficulty of comparing models with differing dimensionality. We can choose the "plausible" dimensionality and then get a optimal model by applying it in MCMC computation. This method not only adds weight to the role of Bayesian approach in model selecting, but also extends to the scope of Metropolis-Hastings method.

In the first section of this chapter, we will introduce the basic assumptions of the models and Green's general idea in getting the acceptance probabilities jumping between subspaces of differing dimensionality. In the second section, we will describe the important example of the acceptance probability calculation of switching between one-dimension models and two-dimension models.

4.1 General form of the acceptance probability

Suppose that there is a countable collection of candidate models $\{\mathscr{M}_k, k \in \mathscr{K}\}$. Model $\mathscr{M}_k$ has a vector $\theta^{(k)}$ of unknown parameters lying in $\mathbb{R}^{n_k}$ where the number of parameters n_k varies between different models. We treat the pair $x = (k, \theta^{(k)})$ as a single point x in the Markov chain Monte Carlo computation. For a given k, x lies in $\mathscr{C}_k = \{k\} \times \mathbb{R}^{n_k}$; generally, x moves in $\mathscr{C} = \bigcup_{k \in \mathscr{K}} \mathscr{C}_k$.

For example, in this article, a "change-point problem" in which there is an unknown number of steps in a piecewise constant regression function on a period of time, will be analysed by selecting the "good" model from $\{\mathscr{M}_k, k \in \mathscr{K}\}$. Figure 5.1 in Chapter 5 shows approximate appearances of these models. For $k \in \mathscr{K} = \{1, 2, \dots\}$, model $\mathscr{M}_k$ implies there are exactly k

steps in the piecewise function. This is different from Green's convention (1995)[1]. We need to specify $k-1$ positions of change-points between each pair of neighbour steps and k "heights" of the steps. Thus $\theta^{(k)}$ is a vector of length $n_k = 2k-1$.

One reason why I treat k as number of steps instead of number of change-points is that setting k heights is more clear and direct in programming. For example, assuming `s` is change-point positions vector in R, `s[0]` cannot be evaluated. We can only set `s[1]`$\sim$`s[2]` to be the first step. If k is set to be the number of change-points, $k=0$ when there is only one step. `s[1]`$\sim$`s[2]` is denoted by `s[k+1]`$\sim$`s[k+2]`. However, $k=1$ if k is set to be the number of steps. In this situation, `s[1]`$\sim$`s[2]` is denoted by `s[k]`$\sim$`s[k+1]`. The latter is clearer and simpler than the former. Another reason is also mentioned on page 27 (Section 5.1). Green ever claimed "there are k steps"[2], which may confuse the readers.

In order to allow x to transverse freely across the combined parameter space $\mathscr{C}$, different types of move m can be derived between the subspaces $\{\mathscr{C}_k\}$ and a proposal $\tilde{x}$ is chosen between available moves at each transition. We restrict our attention to Markov chains in which the transition kernel $K(x,\tilde{x})$[3] is aperiodic, irreducible, and satisfies detailed balance[4]:

$$\int_A \int_B \pi(x)K(x,\tilde{x})\,d\tilde{x}dx = \int_B \int_A \pi(\tilde{x})K(\tilde{x},x)\,dxd\tilde{x}. \tag{4.1}$$

We have talked about "reversibility" in Chapter 2. And we know that this detailed balance relationship requires the equilibrium probability of moving from A to B to equal that from B to A, i.e. $p(A,B) = p(B,A)$, for all Borel sets A, B in $\mathscr{C}$.

Green (1995) proves that, for this to hold, it is sufficient to have

$$\int_A \int_B \pi(x)q_m(x,\tilde{x})\alpha_m(x,\tilde{x})\,dxd\tilde{x} = \int_B \int_A \pi(\tilde{x})q_m(\tilde{x},x)\alpha_m(\tilde{x},x)\,d\tilde{x}dx \tag{4.2}$$

for all moves types m.

Green (1995) also gets

$$\alpha_m(x,\tilde{x})f_m(x,\tilde{x}) = \alpha_m(\tilde{x},x)f_m(\tilde{x},x), \tag{4.3}$$

supposing that $\pi(x)q_m(x,\tilde{x})$ has a finite density $f_m(x,\tilde{x})$ with respect to a symmetric measure ξ_m on $\mathscr{C}\times\mathscr{C}$.

Thus we take

$$\alpha_m(x,\tilde{x}) = \min\{1, \frac{f_m(\tilde{x},x)}{f_m(x,\tilde{x})}\} \tag{4.4}$$

[1] Green said "For $k \in \mathscr{K} = \{0,1,2,\dots\}$, model $\mathscr{M}_k$ says that there are exactly k change-points."

[2] Green (1995), pp.718

[3] Transition kernel $K(x,B)$ is the conditional probability of moving to set B, conditioned on starting in x. Refer to Definition 3in Chapter 2.

[4] Refer to (2.2) in Chapter 2

4.2 Switching between two subspaces

When it comes to a move between two dimension-different subspaces $\mathscr{C}_k$ and $\mathscr{C}_{\tilde{k}}$ (where $\mathscr{C}_k = \{k\} \times \mathscr{R}^{n_k}$ and $\mathscr{C}_{\tilde{k}} = \{\tilde{k}\} \times \mathscr{R}^{n_{\tilde{k}}}$), the specific acceptance probability forms of them are often unapparent due to the complicated corresponding relation of parameters between $\mathscr{M}_k$ and $\mathscr{M}_{\tilde{k}}$. However, the general forms of these acceptance probability are kind of similar on the aspect of representation mode which will be introduced below and the aspect of the basic method to derive them.

The simplest cases are the moves between subspace $\mathscr{C}_1$ and $\mathscr{C}_2$.

For the move from a point $x = (2, \theta_1, \theta_2) \in \mathscr{C}_2$ to, for example, the point $\tilde{x} = (1, \frac{1}{2}(\theta_1 + \theta_2)) \in \mathscr{C}_1$, the equilibrium joint proposal probability

$$P(B, A) = \int_B \pi(x) \int_A q_2(x, \tilde{x})\, d\tilde{x} dx$$

where $A \subset \mathscr{C}_1$,$B \subset \mathscr{C}_2$, have a density with respect to a singular measure on $\mathbb{R} \times \mathbb{R}^2$ i.e. the density of space $\{((\theta_1, \theta_2), \theta) : \theta = \frac{1}{2}(\theta_1 + \theta_2)\}$. Keeping the detailed balanced attainable, the reversible move from A to B should be defined via a proposal distribution q_1 with all its probability on $\{((1, \theta), (2, \theta_1, \theta_2)) : \theta_1 = \theta + u, \theta_2 = \theta - u\}$ and u is a random variable drawn from some distribution, independently of the current θ.

To implement the dimension-match, we need to bring in extra randomly generated parameters, such as the u introduces in last paragraph.

$$\varphi : \theta^{(1)}, u^{(1)} \longmapsto \theta^{(2)}, u^{(2)} \qquad \psi : \theta^{(2)}, u^{(2)} \longmapsto \theta^{(1)}, u^{(1)}$$
$$\varphi : \mathbb{R}^{n_1+m_1} \longrightarrow \mathbb{R}^{n_2+m_2} \qquad \psi : \mathbb{R}^{n_2+m_2} \longrightarrow \mathbb{R}^{n_1+m_1}$$

"consider just one move type, which always switches subspaces, so that $q(x, \mathscr{C}) = 0$ for $x \in \mathscr{C}_1$, and $q(x, \mathscr{C}_2) = 0$ for $x \in \mathscr{C}_2$; the subscript m is being suppressed. The probability of choosing this move will be denoted by $j(x)$. A typical way of accomplishing a transition from $\mathscr{C}_1$ to $\mathscr{C}_2$ will be by generating a vector of continuous random variables $u^{(1)}$ of length m_1, independently of $\theta^{(1)}$, and then setting $\theta^{(2)}$ to be some deterministic function of $\theta^{(1)}$ and $u^{(1)}$. Similarly, to switch back, $u^{(2)}$ of length m_2 will be generated and $\theta^{(1)}$ set to some function of $\theta^{(2)}$ and $u^{(2)}$. For dimension-matching, there must be a bijection between $(\theta^{(1)}, u^{(1)})$ and $(\theta^{(2)}, u^{(2)})$. In particular, the length of $u^{(1)}$ and $u^{(2)}$ must satisfy $n_1 + m_1 = n_2 + m_2$."[5]

$$\varphi : \theta^{(1)}, u^{(1)} \longmapsto \theta^{(2)}(\theta^{(1)}, u^{(1)}), u^{(2)}$$
$$n_1 + m_1 = \quad n_2 \quad + \quad m_2$$

[5] Green (1995), pp.716

$$\psi : \theta^{(2)}, u^{(2)} \longmapsto \theta^{(1)}(\theta^{(2)}, u^{(2)}), u^{(1)}$$
$$n_2 + m_2 = \quad n_1 \quad + \quad m_1$$

Green (1995) shows that for $x = (1, \theta^{(1)}) \in \mathscr{C}_1$ and $\tilde{x} = (2, \theta^{(2)}) \in \mathscr{C}_2$, if set

$$f(x, \tilde{x}) = p(1, \theta^{(1)}) j(1, \theta^{(1)}) q_1(u^{(1)})$$
$$f(\tilde{x}, x) = p(2, \theta^{(2)}) j(2, \theta^{(2)}) j(2, \theta^{(2)}) q_2(u^{(2)}) \Big| \frac{\partial(\theta^{(2)}, u^{(2)})}{\partial(\theta^{(1)}, u^{(1)})} \Big|$$

and otherwise $f(x, \tilde{x}) = 0$. Besides, for $A \subset \mathscr{C}_1$ and $B \subset \mathscr{C}_2$, set ξ such that

$$\xi(A \times B) = \xi(B \times A) = \lambda\{(\theta^{(1)}, u^{(1)}) : \theta^{(1)} \in A,\ \theta^{(2)}(\theta^{(1)}, u^{(1)}) \in B\}$$

where λ denotes $(n_1 + m_1)$-dimensional Lebesgue measure. Then for all $x, \tilde{x} \in \mathscr{C}$, $f(x, \tilde{x})$ is the density with respect to ξ of the equilibrium joint proposal distribution $\pi(x)q(x, \tilde{x})$.

According to (4.4), the acceptance probability for the proposed transition from $(1, \theta^{(1)})$ to $(2, \theta^{(2)})$ is

$$\alpha((1, \theta^{(1)}), (2, \theta^{(2)})) = \min\Big\{1, \frac{p(2, \theta^{(2)}|y) j(2, \theta^{(2)}) q_2(u^{(2)})}{p(1, \theta^{(1)}|y) j(1, \theta^{(1)}) q_1(u^{(1)})} \Big| \frac{\partial(\theta^{(2)}, u^{(2)})}{\partial(\theta^{(1)}, u^{(1)})} \Big| \Big\} \tag{4.5}$$

In practise, either $n_1 > n_2$ or $n_2 > n_1$ is possible. The jump are often set up so that $n_1 + m_1 = n_2$ or $n_2 + m_2 = n_1$. And we don't need to generate $u^{(2)}$ or $u^{(1)}$ correspondingly. The expression of the acceptance probability simplifies. For example, if $m_2 = 0$, it becomes

$$\alpha((1, \theta^{(1)}), (2, \theta^{(2)})) = \min\Big\{1, \frac{p(2, \theta^{(2)}|y) j(2, \theta^{(2)})}{p(1, \theta^{(1)}|y) j(1, \theta^{(1)}) q_1(u^{(1)})} \Big| \frac{\partial(\theta^{(2)})}{\partial(\theta^{(1)}, u^{(1)})} \Big| \Big\}. \tag{4.6}$$

where $j(x)$ means the probability of choosing move x, x is denoted by the starting point; $\Big| \frac{\partial(\theta^{(2)})}{\partial(\theta^{(1)}, u^{(1)})} \Big|$ is the Jacobian of the parameters vectors.

4.3 Moving among more than two subspaces

We have determined the acceptance probability of two subspaces. When it comes to moving among more than two subspaces, the case seems to be a little bit more complicated.

If we calculate all the specific acceptance probabilities of any two subspaces, the amount of calculation will be too large. It is because every different pairs of subspaces can generate varying forms of acceptance probabilities. And it is not easy to teach a machine the rule of analysing the specific form of (4.5) for a particular pair of subspaces.

In practice, we don't need to calculate all possible acceptance probabilities' forms. We can "decompose" the jump from subspace $\mathscr{C}_k$ to subspace $\mathscr{C}_{\tilde{k}}$ into a sequence of small jumps from one subspace to its "neighbour" subspace. For example, jumping from $\mathscr{C}_3$ to $\mathscr{C}_5$ is equal to jumping from $\mathscr{C}_3$ to $\mathscr{C}_4$ and then to $\mathscr{C}_5$. The acceptance probability of the big jump from 3-dimensions to 5-dimensions is just the product of the acceptance probabilities of jumps from 3D to 4D and from 4D to 5D. If any small jump is rejected, the big jump is rejected.

In the experiment of Chapter 5, the forms of acceptance probabilities from one subspace to its "left" neighbour subspace (death) are the same. And so are the probabilities to its "right" neighbour (birth). Here death means that the number of steps decrease from $k+1$ to k; birth means the number of steps increase from k to $k+1$.

As it is said in Green (1995), there are at most four available move types at each transition: (Height,Position,Birth,Death). These have probabilities η_k for Height, π_k for Position, b_{k-1} for Birth, and d_{k-1} for Death (depending only on the current number of steps k), satisfying $\eta_k + \pi_k + b_{k-1} + d_{k-1} = 1$ ($d_0 = \pi_1 = 0$ and $b_{k_{\max}-1} = 0$). "Apart from these constrains, these probabilities were chosen so that

$$b_k = c\min\{1, p(k+1)/p(k)\}, \qquad d_{k+1} = c\min\{1, p(k)/p(k+1)\},$$

with the constant c as large as possible subject to $b_k + d_k \leqslant 0.9$ for all $k = 0, 1, \ldots, k_{\max}$. This choice ensures that $b_k p(k) = d_{k+1} p(k+1)$, which is the condition on b_k and d_k that would guarantee certain acceptance in the corresponding, but much simpler, Hastings sampler for the number of steps alone." [6]Finally for $k > 1$, we take $\eta_k = \pi_k$.

These probabilities of choosing the move type will be applied in the experiment in section 5.6. j(x) in formula (4.6) exactly refers to these probabilities of choosing move.

[6]Green (1995), pp.719

Chapter 5

Application to Multiple Change-point Problems for a Poisson Process

A classic set for illustrating method for change-point analysis is the point process of date of serious coal-mining disaster between 1851 and 1962, given by Raftery and Akman (1986).

Green (1995) develops a Bayesian multiple change-point analysis of the data above by assuming the rate function to be a step function.

In this chapter, I re-analyse the disasters data, and set up a program to select a proper step model for this data with the Metropolis-Hastings algorithm, especially with Green's method of jump between the different dimensionality.

I have supplemented some contents of Green (1995) on this paper. Green directed gives the likelihood function. I explain the details of the model likelihood. I add the calculation processes of the acceptance probabilities form of all move types, while Green (1995) presents the result formulas directly. And I also write an R program to analyse the plausible models of data, while Green (1995) only gives the analysing results.

The problem, the basic model frame and the all kind of its parameters will be introduced in section 5.1. And In the section 5.2, 5.3, 5.5, I apply the content of chapters 2,3,4 to calculating the acceptance probabilities for each type of move and designing the specific algorithms. In section 5.4 and 5.6, I implement all algorithms in R and analyse the result of the model selection.

We introduce a concept "Poisson Process" here.

Definition 7. [1] *A point process N is called a **Poisson process** with intensity function λ, if*

i $N(t)$ *has independent increments;*

ii $N(b) - N(a)$ *is* $Poisson(\int_a^b \lambda(t)\,dt)$*-distributed.*

The point process of data of the coal-mining is treated as a Poisson process in our analysis. And details are followed.

[1] Refer to Grandell (1997), pp.53. λ is treated as a constant in the original text, but it is treated as a function in the modified definition here.

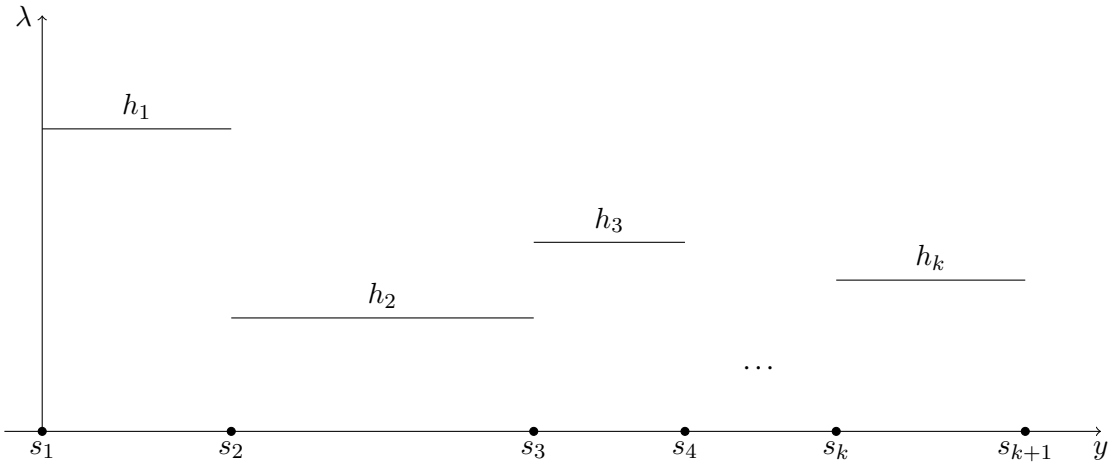


Figure 5.1: The model for intensity function λ.

5.1 Description of the problem

In Green (1995)'s paper, 192 disasters in 112 years is shown by data points $\{y_i, i = 1, 2, \ldots, n\} \in [0, L]$. But now the data (Dates of Coal-mining Disasters) is given in "coal" command in one R package called "boot". "This data frame gives the dates of 191 explosions in coal mines which resulted in 10 or more fatalities. The time span of the data is from March 15, 1851 until March 22, 1962." The data format is date. "The integer part of date gives the year. The day is represented as the fraction of the year that had elapsed on that day."[2] We use a command like this to invoke the data:

```
library("boot")
```

For convenience in this article, we assume the rate function to be a step function $\lambda : \mathbb{R} \to \mathbb{R}^+$ on $[s_1, s_{k+1}]$ where $s_1, s_2, \ldots, s_{k+1}$ is ordered by year instead of by day. We fix s_1 to the beginning of 1851 and s_{k+1} to the end of 1962. Moreover, h_j, the jth piece of $\lambda(.)$, is a constant function on $[s_j, s_{j+1})$.

In our model, λ is the intensity function of a Poisson process $\{y_i, i = 1, 2, \ldots, n\}$. The probability of m disasters occurring in any interval $[a, b] \subseteq [s_1, s_{k+1}]$ is

$$P(\text{ the number of } y_i \in [a, b] = m) = e^{-\Lambda_{a,b}} \frac{\Lambda_{a,b}^m}{m!} \quad \text{where} \quad \Lambda_{a,b} = \int_a^b \lambda(s) ds.$$

Conditioned on m points being present in the interval $[a, b]$, the individual points are independently distributed with density $\lambda/\Lambda_{a,b}$. Thus we can compute the likelihood of the data given λ, as

$$p(y_i \in [a, b] | \lambda) = m! \cdot e^{-\Lambda_{a,b}} \frac{\Lambda_{a,b}^m}{m!} \cdot \prod_{i=1}^{m} \frac{\lambda(y_i)}{\Lambda_{a,b}}$$

[2]This imformation is given by "boot" package information source (2012)

The factor $m!$ is the number of arrangements when m points are ordered.

In the interval $[s_1, s,_{k+1}]$, there are n disasters in total. Thus we have

$$\begin{aligned}
p(y_1, y_2, \ldots, y_n|\lambda) =& n! \cdot e^{-\Lambda} \frac{\Lambda^n}{n!} \cdot \prod_{i=1}^{n} \frac{\lambda(y_i)}{\int_{s_1}^{s_{k+1}} \lambda(s)ds} \qquad (\text{where} \Lambda = \textstyle\int_{s_1}^{s_{k+1}} \lambda(s)ds) \\
=& e^{-\Lambda} \cdot \Lambda^n \cdot \frac{\prod_{i=1}^{n} \lambda(y_i)}{\Lambda^n} \\
=& e^{-\Lambda} \cdot \prod_{i=1}^{n} \lambda(y_i) \\
=& \exp(-\Lambda + \log \prod_{i=1}^{n} \lambda(y_i)) \\
=& \exp(-\Lambda + \sum_{i=1}^{n} \log \lambda(y_i)) \\
=& \exp(-\Lambda + \sum_{j=1}^{k} (\text{number of } y_i \in [s_j, s_{j-1})) \cdot \log \lambda(y_i))
\end{aligned}$$

In our model, $s_1 < s_2 < \cdots < s_{k+1}$, and $\lambda(y_i) = h_j$ when $y_i \in [s_j, s_{j+1})$.
Thus $\Lambda = \int_{s_1}^{s_{k+1}} \lambda(s)ds = \sum_{j=1}^{k} h_j(s_{j+1} - s_j)$. And

$$p(y_1, y_2, \ldots, y_n|\lambda) = \exp(-\sum_{j=1}^{k} h_j(s_{j+1} - s_j) + \sum_{j=1}^{k} (\text{number of } y_i \in [s_j, s_{j-1})) \cdot \log h_j) \quad (5.1)$$

is what we called the likelihood.

In R code, a "count.disasters" function is set to count the number of disasters happen during a selected date interval.

```
# disaster number function m(.)
count.disasters <- function(s1, s2) {
        return(sum(coal >= s1 & coal < s2))
}
```

There are three kinds of random variables in this model (5.1):

1. the number of steps k

   Green (1995) takes k as the number of change-points, but sometimes he claim his k is the number of steps, which may confuse the readers. Here I assume k to be the number of steps. So correspondingly the change-points' number is $k - 1$. Differing from Green's k, which can be 0 in the extreme situation that only one step $[0, L]$ exists, our k starts from 1 in the extreme situation. Based on Poisson distribution's definition, we supposes our k

drawn from Poisson distribution. The prior probabilities are

$$p(k) = e^{-\lambda}\frac{\lambda^{k-1}}{(k-1)!}$$

and conditioned on $k \leqslant k_{max}$.

2. $\{h_j : j = 1, 2, \cdots, k\}$
   The heights $h_1, h_2, \cdots, h_k$ are independently distributed as $\Gamma(\alpha, \beta)$. According to Green, " it is not appropriate to select an improper gamma distribution $\Gamma(0, 0)$ for heights, because that cause in surmountable difficulties with normalisation across differing numbers of step; all the probability in the posterior would be assigned to the simplest model".

   In the R code, we give proper values to "a" and "b" which represent the parameter α and β of Gamma distribution $\Gamma(\alpha, \beta)$ respectively.

   ```
   # value of Gamma(a,b)
   a=1
   b=200/365.24
   ```

3. $\{s_j : j = 2, 3, \cdots, k\}$ (But note that s_1 and s_{k+1} are fixed points.)
   Given k, the step position $s_2, s_3, \ldots, s_k$ are distributed as the even-numbered order statistics from $2k-1$ points $\{t_i, i = 1, 2, 3, \ldots, 2k-1\}$ uniformly distributed on (s_1, s_{k+1}). That is $s_2 = t_2, s_3 = t_4, \cdots, s_j = t_{2j-2}, \cdots, s_k = t_{2k-2}$. This model introduced by Green can avoid too many "short" steps with $s_{j+1} - s_j$ small. Since there may be no data in the interval (s_j, s_{j+1}), such short intervals are barely penalised by the likelihood and so survive in the posterior.

A reversible jump Monte Carlo sampler is a good way to approach λ within the given information of real data. There are only four types of transitions from the prior distribution to the posterior distribution: (i) a change to the height of a randomly chosen step,(ii) a change to the position of a randomly chosen step, (iii) "birth" of a new step at a randomly chosen location in $[s_1, s_{k+1}]$, and (iv) "death" of a randomly chosen step, which is according to Green.

We now analyse λ in more detail and try to calculate it in R.

5.2 The first kind of moves: Height

Parameter: $h_j \in (0, +\infty)$ where $j = 1, \ldots, k$
Prior distribution: the heights $h_1, \ldots, h_k$ are independently drawn from the $\Gamma(\alpha, \beta)$.

$$p(h_j) = \frac{\beta^\alpha h_j^{\alpha-1} e^{-\beta h_j}}{\Gamma(\alpha)}$$

Data: $\{y_i, i = 1, 2, \ldots, n\} \in [s_1, s_{k+1}]$ is a Poisson process. h_j, the jth piece of $\lambda(.)$, is a constant function on $[s_j, s_{j+1})$. m_j is the number of disasters which happened during $[s_j, s_{j+1})$.

$$p(y_1, y_2, \ldots, y_n|h_j) = \exp(-\sum_{j=1}^{k} h_j(s_{j+1} - s_j) + \sum_{j=1}^{k} m_j \log h_j)$$

Green chooses h_j at random. Then proposes a change to $\tilde{h_j}$ such that $\log(\tilde{h_j}/h_j)$ is uniformly distributed on the interval $[-\frac{1}{2}, \frac{1}{2}]$. According this assumption, I derive the transition density $q(h_j, \tilde{h_j})$ of this proposing:

$$\log \frac{\tilde{h}}{h} = u \text{ (where } u \sim U[-\frac{1}{2}, \frac{1}{2}]) \Longrightarrow \tilde{h} = e^u h \text{ (where } \tilde{h} \in [\frac{h}{\sqrt{e}}, \sqrt{e}h])$$

$$F(\nu) = P(\tilde{h} \leqslant \nu) = P(e^u h \leqslant \nu) = P(u \leqslant \log(\frac{\nu}{h})) = \frac{\log(\frac{\nu}{h}) - (-\frac{1}{2})}{\frac{1}{2} - (-\frac{1}{2})} = \log(\frac{\nu}{h}) + \frac{1}{2}$$

$$f(\nu) = F'(\nu) = \frac{h}{\nu} \cdot \frac{1}{h} = \frac{1}{\nu} \quad \text{Thus } q(h, \tilde{h}) = \frac{1}{\tilde{h}}$$

Bayesian parameter estimates M-H Algorithm:

Input:
target density $\pi(h_j) = \frac{1}{Z} p(y_1, y_2, \ldots, y_n|h_j)p(h_j)$
where Z is the constant value of $\int p(y_1, y_2, \ldots, y_n|\tilde{h_j})p(\tilde{h_j})\, d\tilde{h_j}$
transition density

$$q(h_j, \tilde{h_j}) = \frac{1}{\tilde{h_j}} \text{ (where } \tilde{h} \in [\frac{h}{\sqrt{e}}, \sqrt{e}h])$$

$h_j^0 = 1$ with $\pi(h_j^0) = \dfrac{\beta^\alpha h_j^{0(\alpha-1)} e^{-\beta h_j^0}}{\Gamma(\alpha)} > 0$

Output:
a Markov chain with stationary density $\pi(h_j)$

for $n = 1, 2, 3 \ldots$
generate $\tilde{h_j^n}$ by $\ln \tilde{h_j^n} = \ln h_j^{n-1} + u_n$ where $u_n \sim U[-\frac{1}{2}, \frac{1}{2}]$
generate $U_n \sim U[0, 1]$
if $U_n \leqslant \alpha(h_j^{n-1}, \tilde{h_j^n})$ where

$$\alpha(h_j, \tilde{h_j}) = \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{h_j})p(\tilde{h_j})h_j}{p(y_1, y_2, \ldots, y_n|h_j)p(h_j)\tilde{h_j}}, 1) \tag{5.2}$$

then $h_j^n \longleftarrow \tilde{h_j^n}$ (accepted)
else $h_j^n \longleftarrow h_j^{n-1}$ (rejected)

Here we explain how to get the acceptance probability (5.2).

$$\begin{aligned}
\alpha(h_j, \tilde{h_j}) &= \min(\frac{\pi(\tilde{h_j})q(\tilde{h_j}, h_j)}{\pi(h_j)q(h_j, \tilde{h_j})}, 1) \\
&= \min(\frac{\frac{1}{Z}p(y_1, y_2, \ldots, y_n|\tilde{h_j})p(\tilde{h_j}) \cdot \frac{1}{\tilde{h_j}}}{\frac{1}{Z}p(y_1, y_2, \ldots, y_n|h_j)p(h_j) \cdot \frac{1}{h_j}}, 1) \\
&= \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{h_j})p(\tilde{h_j}) \cdot h_j}{p(y_1, y_2, \ldots, y_n|h_j)p(h_j) \cdot \tilde{h_j}}, 1)
\end{aligned}$$

To decrease the error caused by machine calculation, we realise it in a log form:

$$\begin{aligned}
&\alpha(h_j, \tilde{h_j}) \\
&= \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{h_j})p(\tilde{h_j})h_j}{p(y_1, y_2, \ldots, y_n|h_j)p(h_j)\tilde{h_j}}, 1) \\
&= \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{h_j})\frac{\beta^\alpha \tilde{h_j}^{\alpha-1}e^{-\beta\tilde{h_j}}}{\Gamma(\alpha)} \cdot h_j}{p(y_1, y_2, \ldots, y_n|h_j)\frac{\beta^\alpha h_j^{\alpha-1}e^{-\beta h_j}}{\Gamma(\alpha)} \cdot \tilde{h_j}}, 1) \\
&= \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{h_j})}{p(y_1, y_2, \ldots, y_n|h_j)} \cdot (\frac{\tilde{h_j}}{h_j})^\alpha \cdot e^{-\beta(\tilde{h_j}-h_j)}, 1) \\
&= \min((\frac{\exp(-(\sum_{i=1}^{j-1} + \sum_{i=j+1}^{k})h_i(s_{i+1}-s_i) + (\sum_{i=1}^{j-1} + \sum_{i=j+1}^{k})m_i \log h_i}{\exp(-\sum_{j=1}^{k} h_j(s_{j+1}-s_j) + \sum_{j=1}^{k} m_j \log h_j)} \\
&\quad + \frac{-\tilde{h_j}(s_{j+1}-s_j) + m_j \log \tilde{h_j}}{\exp(-\sum_{j=1}^{k} h_j(s_{j+1}-s_j) + \sum_{j=1}^{k} m_j \log h_j)}) \cdot (\frac{\tilde{h_j}}{h_j})^\alpha \frac{e^{-\beta\tilde{h_j}}}{e^{-\beta h_j}}, 1) \\
&= \exp(\min((h_j - \tilde{h_j})(s_{j+1}-s_j) + m_j(\log \tilde{h_j} - \log h_j) + \alpha \log \tilde{h_j} - \alpha \log h_j - \beta(\tilde{h_j} - h_j), 0)) \\
&= \exp(\min((h_j - \tilde{h_j})(s_{j+1}-s_j+\beta) + (m_j+\alpha)(\log \tilde{h_j} - \log h_j), 0))
\end{aligned}$$

The R code corresponding to the formula of α above is:

```
alpha.hmove <- function(s,h,h_tilde,j) {
        m <- count.disasters(s[j],s[j+1])
```

```
        log_likelihood_ratio <- (
           (h[j] - h_tilde) * (s[j+1] - s[j])
           + m * (log(h_tilde) - log(h[j]))
        )
        log_prior_ratio <- (
           a * (log(h_tilde) - log(h[j])) - b * (h_tilde - h[j])
        )
        log_pi_ratio <- log_likelihood_ratio + log_prior_ratio
        return(exp(log_pi_ratio))
}
```

The following program implements M-H algorithm for height moves only.

```
Move <- function(n, h, s, k) {
    H <- vector(length=n)
    for (i in 1:n){
        j <- sample(1:k,size=1)
        u <- runif(1,-0.5,0.5)
        h_tilde <- h[j] * exp(u)
        U <- runif (1)
        if (U < alpha.hmove(s,h,h_tilde,j)) {
            h[j] <- h_tilde
        }
        H[i] <- h
    }
        return(H)
}
```

We can calculation the average height given $k = 1$ by the R code above. In the program of next section, the mean value of the result given $k = 1$ can be used as the initial value of height in each segment of function λ in order to shorten the burn-in period.

To calculate height given $k = 1$, we should set k to be 1 and the initial s sequence s to be $\{s[1], s[2] : s[1] = 1851, s[2] = 1963\}$. Take a random value, like 1, to be an initial h. And then run the main program, saving the result estimates in a matrix X. Figure out the mean.

```
k0 <- 1
s0 <- c(1851,1963)
h0 <- c(1)
X <- Move(10000, h0, s0, k0)
mean(X)
```

One simulation result is:

```
> mean(X)
[1] 1.707852

> var(X)
[1] 0.01399577
```

After many times estimates and with a large sample size, the results of program show that the average height in $[1851, 1963]$ is around 1.7, the standard error is around 0.015.

5.3 The second kind of moves: Position

Parameter: $s_j \in (s_{j-1}, s_{j+1})$ where $j = 2, \ldots, k$

Prior distribution: Given k, the step position $s_2, s_3, \ldots, s_k$ are distributed as the even-numbered order statistics from $2k-1$ points $\{t_i, i = 1, 2, 3, \ldots, 2k-1\}$ uniformly distributed on (s_1, s_{k+1}). That is $s_2 = t_2, s_3 = t_4, \cdots, s_j = t_{2j-2}, \cdots, s_k = t_{2k-2}$. ($s_1$ and s_{k+1} are fixed for convenience).

$$\pi(s_2, \cdots, s_k) = \frac{(2k-1)!}{(s_{k+1} - s_1)^{2k-1}} \mathbb{1}_{\{s_1 < s_2 < \cdots < s_k < s_{k+1}\}} \prod_{j=1}^{k} (s_{j+1} - s_j) \tag{5.3}$$

Data: $\{y_i, i = 1, 2, \ldots, n\} \in [0, L]$ is a Poisson process. h_j, the jth piece of $x(.)$, is a constant function on $[s_j, s_{j+1})$. m_j is the number of disasters which happened during $[s_j, s_{j+1})$.

$$p(y_1, y_2, \ldots, y_n | s_j) = \exp(-\sum_{j=1}^{k} h_j (s_{j+1} - s_j) + \sum_{j=1}^{k} m_j \log h_j)$$

Green (1995) doesn't give the formula (5.3). Now I give calculation process and explain how to get (5.3), which will be used in the calculation of acceptance probability, according to the distribution of s_i.

$\forall s_i \in [s_1, s_{k+1}], i = 2, 3, \ldots, k$, s_i are random variables.

Each $s_i \sim U[s_1, s_{k+1}]$ has density $p(s_i) = \dfrac{1}{s_{k+1} - s_1}$. Thus the joint distribution is

$$\begin{aligned}
&\pi(s_2, \ldots, s_j, \ldots, s_k) \\
=&(2k-1)! \int\int \ldots \int\int \frac{1}{(s_{k+1} - s_1)^{2k-1}} \mathbb{1}_{\{s_1 < t_1 < s_2 < t_3 < s_3 < \cdots < s_k < t_{2k-1} < s_{k+1}\}} \, dt_1 \, dt_3 \ldots dt_{2k-3} \, dt_{2k-1} \\
&\qquad ((2k-1)! \text{ means the number of all permutations of } 2k-1 \text{ points } \{t_i, i = 1, 2, \ldots, 2k-1\}) \\
&\qquad\qquad (\mathbb{1}_{\{s_1 < t_1 < s_2 < t_3 < s_3 < \cdots < s_k < t_{2k-1} < s_{k+1}\}} \text{ means the range of integrals.}) \\
=&\frac{(2k-1)!}{(s_{k+1} - s_1)^{2k-1}} \int\int \cdots \int\int \mathbb{1}_{\{s_1 < s_2 < t_3 < s_3 < \cdots < t_{2k-3} < s_k < t_{2k-1} < s_{k+1}\}} (s_2 - s_1) \, dt_3 \, dt_5 \ldots dt_{2k-3} \, dt_{2k-1} \\
=&\frac{(2k-1)!}{(s_{k+1} - s_1)^{2k-1}} \int\int \cdots \int \mathbb{1}_{\{s_1 < s_2 < s_3 < \cdots < t_{2k-3} < s_k < t_{2k-1} < s_{k+1}\}} (s_2 - s_1)(s_3 - s_2) \, dt_5 \, dt_7 \ldots dt_{2k-3} \, dt_{2k-1} \\
=&\ldots
\end{aligned}$$

$$=\frac{(2k-1)!}{(s_{k+1}-s_1)^{2k-1}}\mathbb{1}_{\{s_1<s_2<\cdots<s_k<s_{k+1}\}}(s_2-s_1)(s_3-s_2)\dots(s_j-s_{j-1})(s_{j+1}-s_j)\dots(s_{k+1}-s_k)$$
$$=\frac{(2k-1)!}{(s_{k+1}-s_1)^{2k-1}}\mathbb{1}_{\{s_1<s_2<\cdots<s_k<s_{k+1}\}}\prod_{j=1}^{k}(s_{j+1}-s_j)$$

And if $s_1,\dots,s_{j-1},s_{j+1},\dots,s_{k+1}$ are determined and only s_j is variable,

$$\begin{aligned}\pi(s_j|s_2,\cdots,s_{j-1},s_{j+1},\cdots,s_k)&=\int\int\frac{1!}{s_{j+1}-s_{j-1}}\mathbb{1}_{\{s_{j-1}<t_{2j-3}<s_j<t_{2j-1}<s_{j+1}\}}\,dt_{2j-3}\,dt_{2j-1}\\&=\int\frac{1}{s_{j+1}-s_{j-1}}\mathbb{1}_{\{s_{j-1}<s_j<t_{2j-1}<s_{j+1}\}}(s_j-s_{j-1})\,dt_{2j-1}\\&=\frac{1}{s_{j+1}-s_{j-1}}\mathbb{1}_{\{s_{j-1}<s_j<s_{j+1}\}}(s_j-s_{j-1})(s_{j+1}-s_j)\end{aligned}$$

Bayesian parameter estimates M-H Algorithm:

Input:
target density
$\pi(s_j|s_2,\cdots,s_{j-1},s_{j+1},\cdots,s_k)=\frac{1}{Z}p(y_1,y_2,\dots,y_n|s_j)p(s_j|s_2,\cdots,s_{j-1},s_{j+1},\cdots,s_k)$
where Z is the constant value of
$\int p(y_1,y_2,\dots,y_n|\tilde{s_j})p(\tilde{s_j}|s_2,\cdots,s_{j-1},s_{j+1},\cdots,s_k)\,d\tilde{s_j}$
transition density $q(s_j,\tilde{s_j})=\dfrac{1}{s_{j+1}-s_{j-1}}$
$s_j^0\in(s_{j-1},s_{j+1})$ with $\pi(s_j^0)>0$

Output:
a Markov chain with stationary density $\pi(s_j)$

for $n=1,2,3\dots$
generate $\tilde{s_j}^n\sim U(s_{j-1},s_{j+1})$
generate $U_n\sim U[0,1]$
if $U_n\leqslant\alpha(s_j^{n-1},\tilde{s_j}^n)$ where

$$\alpha(s_j,\tilde{s_j})=\min\left(\frac{p(y_1,y_2,\dots,y_n|\tilde{s_j})p(\tilde{s_j}|s_2,\cdots,s_{j-1},s_{j+1},\cdots,s_k)}{p(y_1,y_2,\dots,y_n|s_j)p(s_j|s_2,\cdots,s_{j-1},s_{j+1},\cdots,s_k)},1\right)\tag{5.4}$$

then $s_j^n\longleftarrow\tilde{s_j}^n$ (accepted)
else $s_j^n\longleftarrow s_j^{n-1}$ (rejected)

Here we explain how to get the acceptance probability (5.4).

$$\begin{aligned}
\alpha(s_j, \tilde{s_j}) &= \min(\frac{\pi(s_2, \cdots, \tilde{s_j}, \cdots, s_k)q(\tilde{s_j}, s_j)}{\pi(s_2, \cdots, s_j, \cdots, s_k)q(s_j, \tilde{s_j})}, 1) \\
&= \min(\frac{\pi(\tilde{s_j}|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)\pi(s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)q(\tilde{s_j}, s_j)}{\pi(s_j|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)\pi(s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)q(s_j, \tilde{s_j})}, 1) \\
&= \min(\frac{\pi(\tilde{s_j}|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)q(\tilde{s_j}, s_j)}{\pi(s_j|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)q(s_j, \tilde{s_j})}, 1) \\
&= \min(\frac{\frac{1}{Z}p(y_1, y_2, \ldots, y_n|\tilde{s_j})\pi(\tilde{s_j}|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)\frac{1}{s_{j+1} - s_{j-1}}}{\frac{1}{Z}p(y_1, y_2, \ldots, y_n|s_j)\pi(s_j|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)\frac{1}{s_{j+1} - s_{j-1}}}, 1) \\
&= \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{s_j})\pi(\tilde{s_j}|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)}{p(y_1, y_2, \ldots, y_n|s_j)\pi(s_j|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)}, 1)
\end{aligned}$$

To decrease the error caused by machine calculation, we realise it in a log form.

$$\begin{aligned}
&\alpha(s_j, \tilde{s_j}) \\
&= \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{s_j})\pi(\tilde{s_j}|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)}{p(y_1, y_2, \ldots, y_n|s_j)\pi(s_j|s_2, \cdots, s_{j-1}, s_{j+1}, \cdots, s_k)}, 1) \\
&= \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{s_j})\frac{1}{s_{j+1}-s_{j-1}}(\tilde{s_j} - s_{j-1})(s_{j+1} - \tilde{s_j})}{p(y_1, y_2, \ldots, y_n|s_j)\frac{1}{s_{j+1}-s_{j-1}}(s_j - s_{j-1})(s_{j+1} - s_j)}, 1) \\
&= \min((\frac{\exp(-(\sum_{i=1}^{j-2} + \sum_{i=j+1}^{k})h_i(s_{i+1} - s_i) - h_{j-1}(\tilde{s_j} - s_{j-1}) - h_j(s_{j+1} - \tilde{s_j}))}{\exp(-\sum_{j=1}^{k} h_j(s_{j+1} - s_j) + \sum_{j=1}^{k} m_j \log h_j)} \\
&\quad \times \exp((\sum_{i=1}^{j-2} + \sum_{i=j+1}^{k})m_i \log h_i + \tilde{m}_{j-1} \log h_{j-1} + \tilde{m}_j \log h_j) \cdot \frac{(\tilde{s_j} - s_{j-1})(s_{j+1} - \tilde{s_j})}{(s_j - s_{j-1})(s_{j+1} - s_j)}, 1) \\
&= \exp(\min(-h_{j-1}(\tilde{s_j} - s_{j-1}) - h_j(s_{j+1} - \tilde{s_j}) + \tilde{m}_{j-1} \log h_{j-1} + \tilde{m}_j \log h_j - (-h_{j-1}(s_j - s_{j-1}) \\
&\quad - h_j(s_{j+1} - s_j) + m_{j-1} \log h_{j-1} + m_j \log h_j) \cdot \log \frac{(\tilde{s_j} - s_{j-1})(s_{j+1} - \tilde{s_j})}{(s_j - s_{j-1})(s_{j+1} - s_j)}, 0)) \\
&= \exp(\min((h_j - h_{j-1})(\tilde{s_j} - s_j) + (\tilde{m}_{j-1} - m_{j-1}) \log h_{j-1} + (\tilde{m}_j - m_j) \log h_j \\
&\quad + \log(\tilde{s_j} - s_{j-1}) + \log(s_{j+1} - \tilde{s_j}) - \log(s_j - s_{j-1}) - \log(s_{j+1} - s_j), 0))
\end{aligned}$$

The R codes corresponding to the formula of α above is:

```
alpha.smove <- function(s,h,s_tilde,j) {
               m1 <- count.disasters(s[j-1],s[j])
        m1_tilde <- count.disasters(s[j-1],s_tilde)
               m2 <- count.disasters(s[j],s[j+1])
        m2_tilde <- count.disasters(s_tilde,s[j+1])
```

```
        log_likelihood_ratio <- (
           (h[j] - h[j-1]) * (s_tilde - s[j]) + (m1_tilde - m1) * log(h[j-1])
           + (m2_tilde - m2) * log(h[j])
        )
        log_prior_ratio <- (
           log(s_tilde - s[j-1]) + log(s[j+1] - s_tilde)
           - log(s[j] - s[j-1]) - log(s[j+1] - s[j])
        )
        log_pi_ratio <- log_likelihood_ratio + log_prior_ratio
        return(exp(log_pi_ratio))
}
```

5.4 Experiment with fixed number of steps

The "`count.disasters`" function and the "`alpha.hmove`" function is in section 5.1, 5.2 and the "`alpha.smove`" function is ready in section 5.3.

With the height and position acceptance algorithms, we can now determine a model given a fixed k steps. The probabilities of choosing which height to change and those of choosing which position of change-point to vary are set to be equal.

Here is the M-H algorithm main program:

```
Move <- function(n,h_initial,s_initial){
    H <-matrix(NA,nrow=n,ncol=k)
    S <- matrix(NA,nrow=n,ncol=k+1)
    h <- h_initial
    s <- s_initial

    for (i in 1:n){
        j <- sample(1:(2*k-1),size=1,replace=TRUE)

        if (j <= k) {              #from 1 to k is for Height
            u <- runif(1,-0.5,0.5)
            h_tilde <- h[j] * exp(u)
            U <- runif (1)
            if (U < alpha.hmove(s,h,h_tilde,j)){
                h[j] <- h_tilde
            }
        }

        if (j > k) {               #from k+1 to 2k-1 is for Position
            j = j - k + 1
            s_tilde <- runif(1,s[j-1],s[j+1])
             U <- runif (1)
             if (U < alpha.smove(s,h,s_tilde,j)){
```

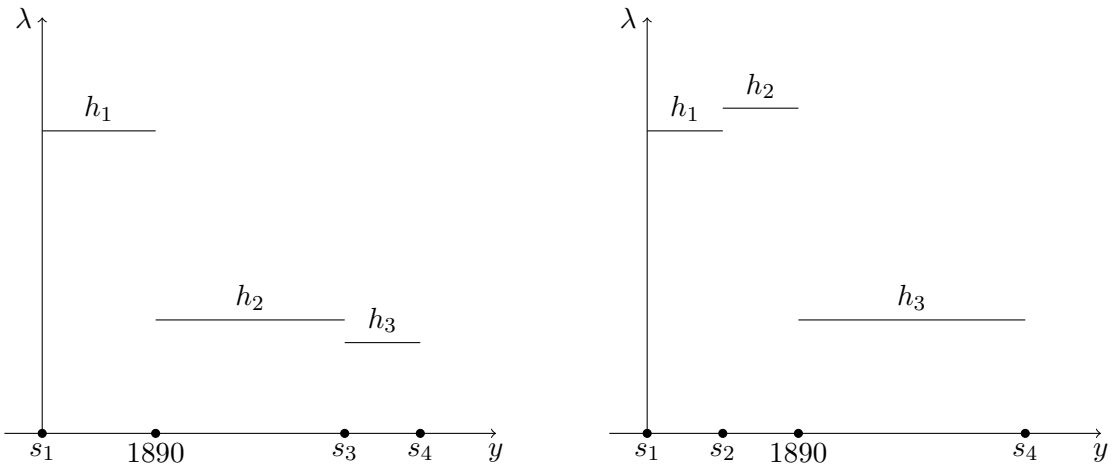


Figure 5.2: The change-point around 1890.

```
                s[j] <- s_tilde
              }
            }

          H[i,] <- h
          S[i,] <- s
          }
          return(cbind(H,S))
    }
```

We estimate for $1,000$, $10,000$, and $100,000$ times respectively given $k = 3$:

```
k <- 3
s0 <- c(1851, 1890.5, 1927, 1963)
h0 <- c(1.7,1.7,1.7)
X<-Move(1000,h0,s0)
X<-Move(10000,h0,s0)
X<-Move(100000,h0,s0)
```

In the path and distribution pictures of heights, the red, blue and green lines represent h_1, h_2, h_3 respectively. In those pictures of positions, the red and blue lines represent s_2, s_3 respectively.

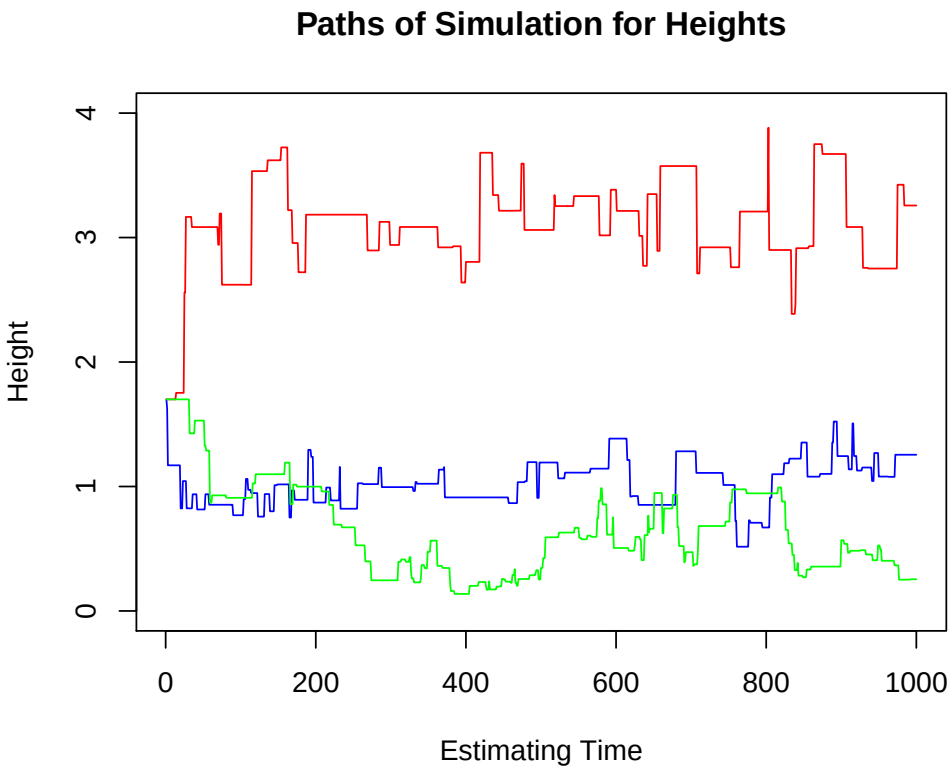
1. We can see from the figure 5.3, 5.5, 5.7, as the estimating time n (i.e. the steps of Markov chain) getting longer, the shapes of heights' distributions get smoother. When $n = 1000$ or $n = 10,000$, the height paths are rough, ande the peaks of distributions are blunt and uncertain, which show that sample size $n = 1000$ and $n = 10,000$ are too small to determine a good model.

2. Figure 5.6 and figure 5.8 show that position $s \approx 1890$ is an important change-point. For most of time, s_1(red line) sits at approximate 1890 and s_2(blue line) sits at approximate

1950. But sometimes, s_2 will jump close to 1890 instead of other positions. This jump may be companied by the special height's changes, which is illustrated remarkably in figure 5.5(Top). We can see from 5.5, h_2(blue) jumps close to h_1(red) at time $t \approx 4000$.

I have put the common case and this special jump case together in figure 5.2. Note that position approximate 1890 can either be the 1th change-point or the 2nd change-point. The special jump is corresponding to the right picture of 5.2.

3. We have analysed the Markov chains generated by the model given $k = 3$. Similarly, we can estimate λ for $100,000$ times given $k = 2$ (figure 5.9, 5.10) and $k = 4$ (figure 5.11, 5.12). We can deduce that the model given $k = 2$ is also a simple plausible model. But the model given $k = 4$ seems to be not so good, because s_2 and s_3 are very close to each other and it is reasonable to treat them as a single change-point.



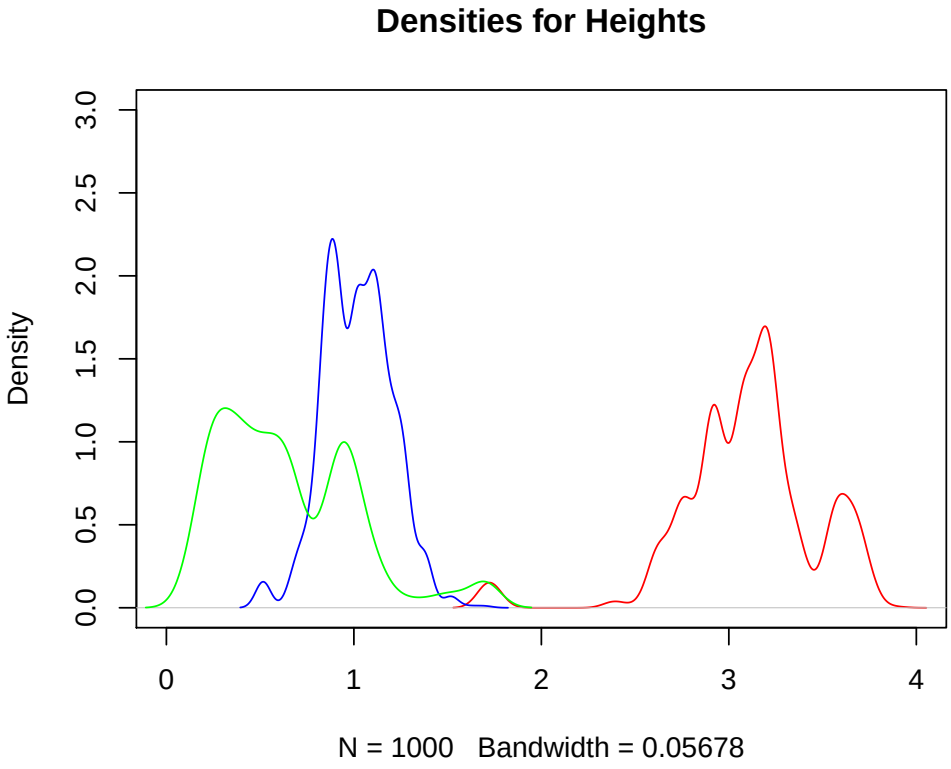
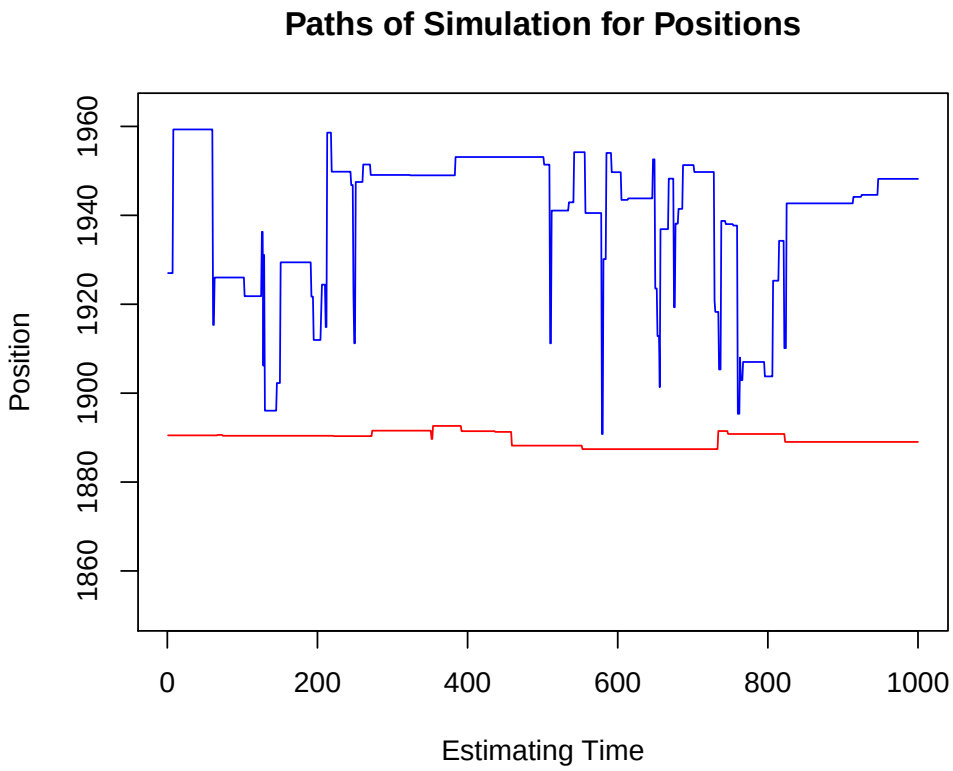


Figure 5.3: The paths and distribution of heights for 1,000 steps of the Markov chain given $k = 3$. *Red:*h_1 *Bule:*h_2 *Green:*h_3.



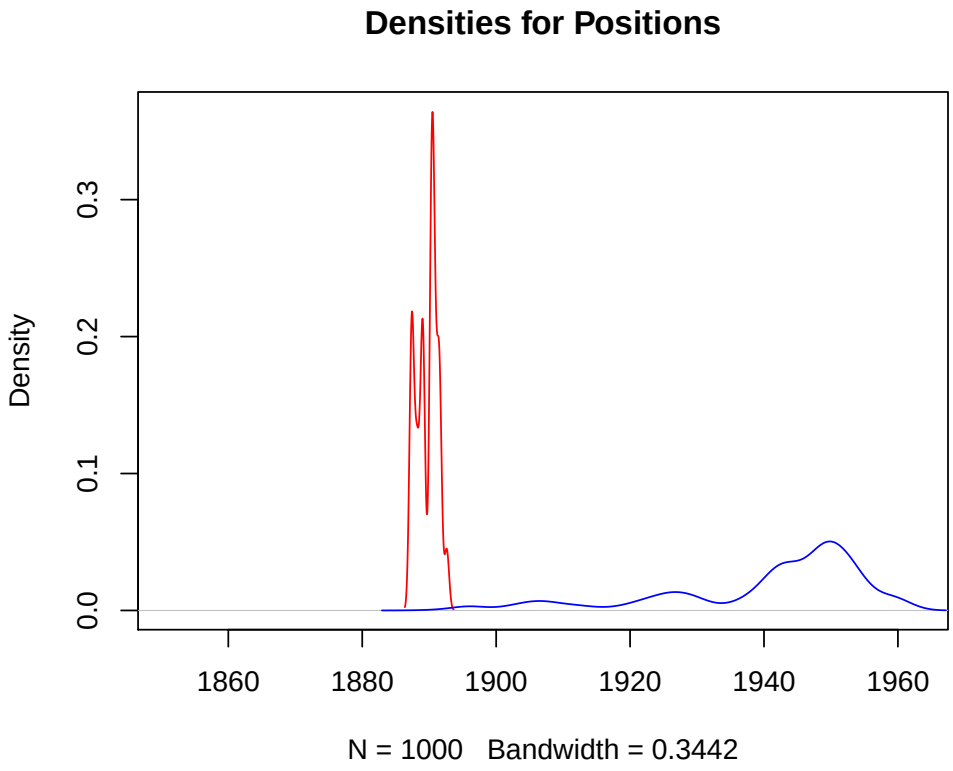
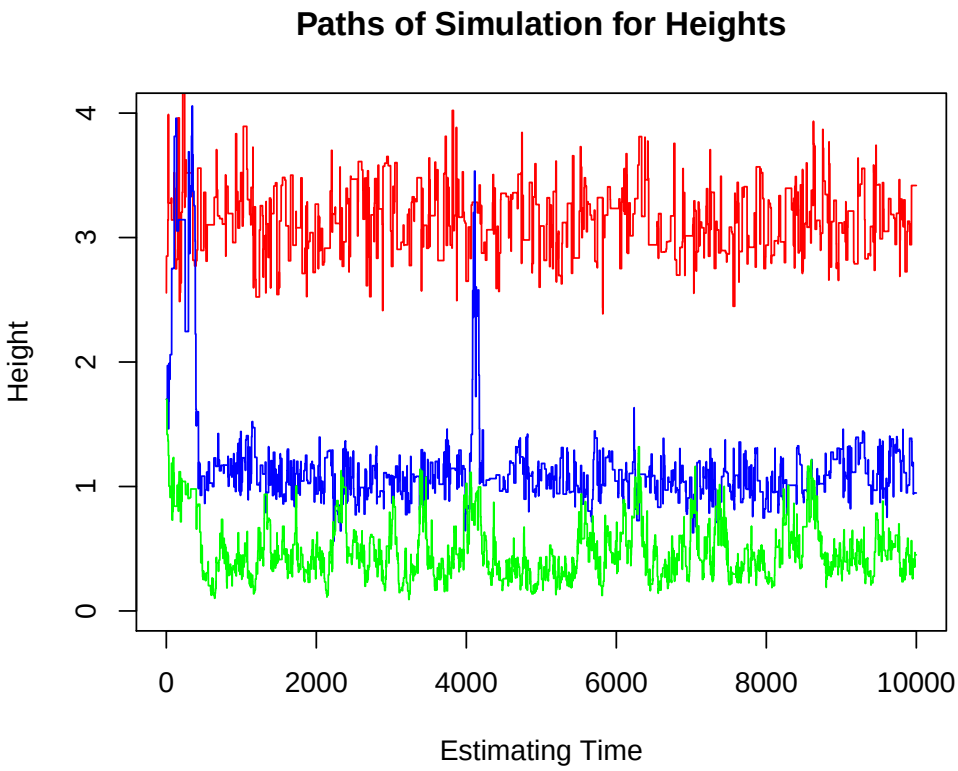


Figure 5.4: The paths and distribution of positions for 1,000 steps of the Markov chain given $k = 3$. *Red:*s_2 *Bule:*s_3.



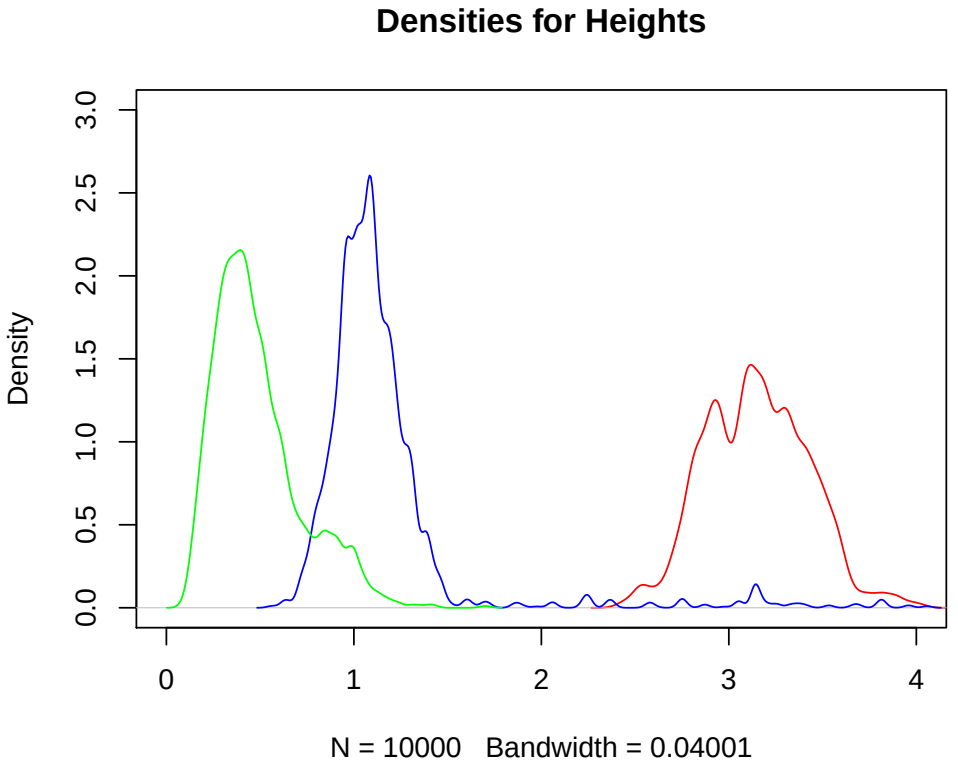
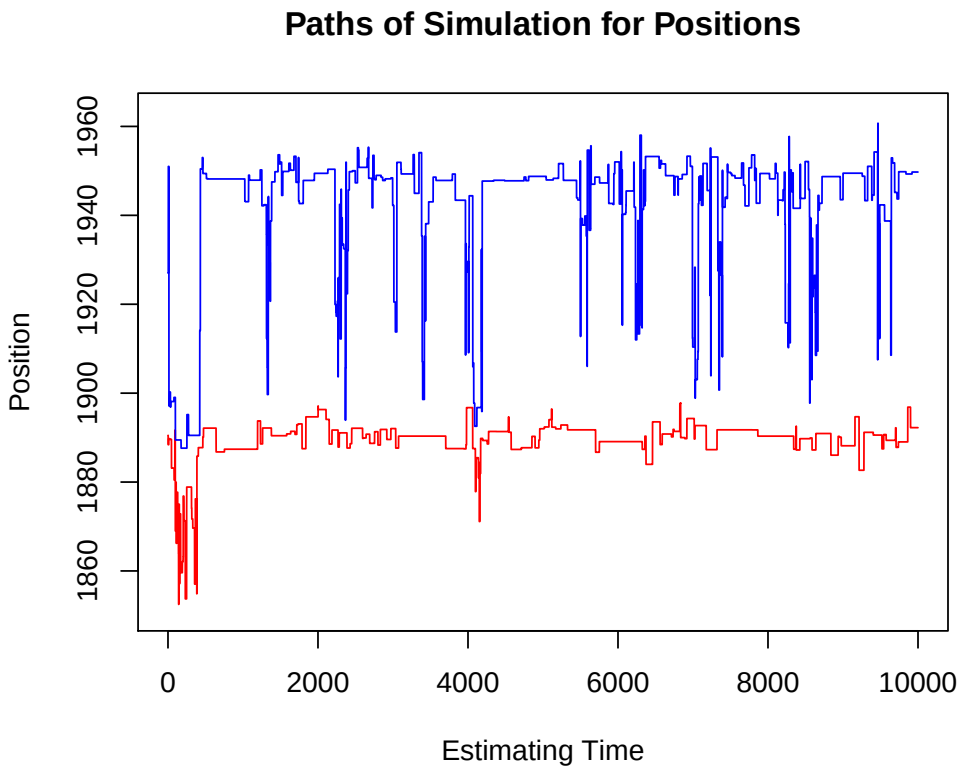


Figure 5.5: The paths and distribution of heights for 10,000 steps of the Markov chain given $k = 3$. *Red:*h_1 *Bule:*h_2 *Green:*h_3.



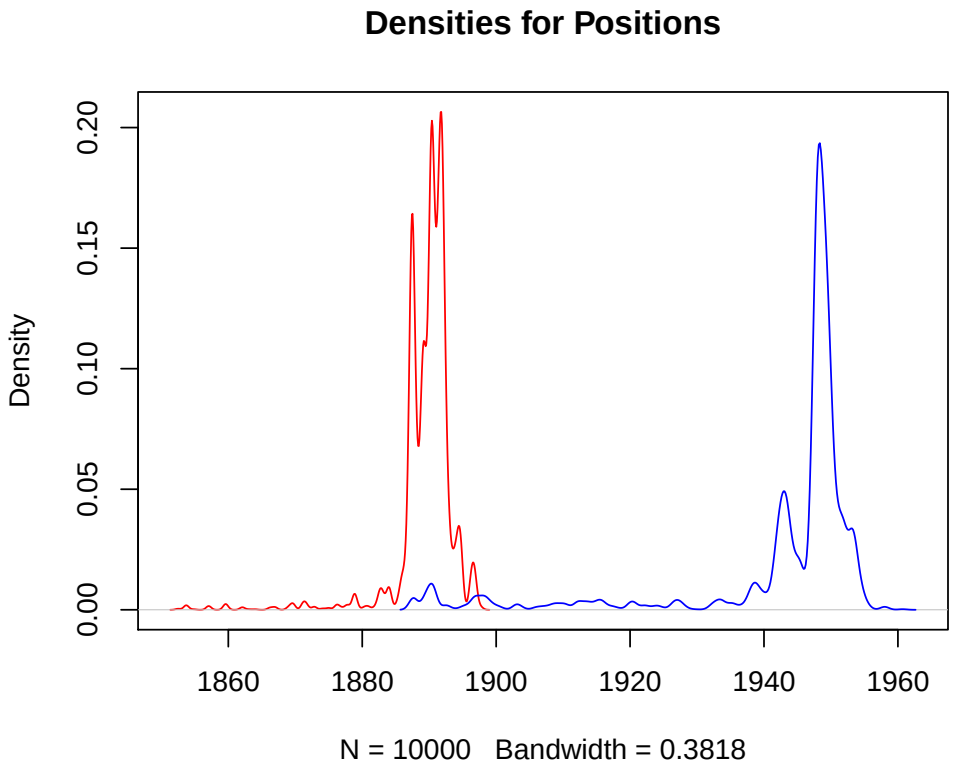
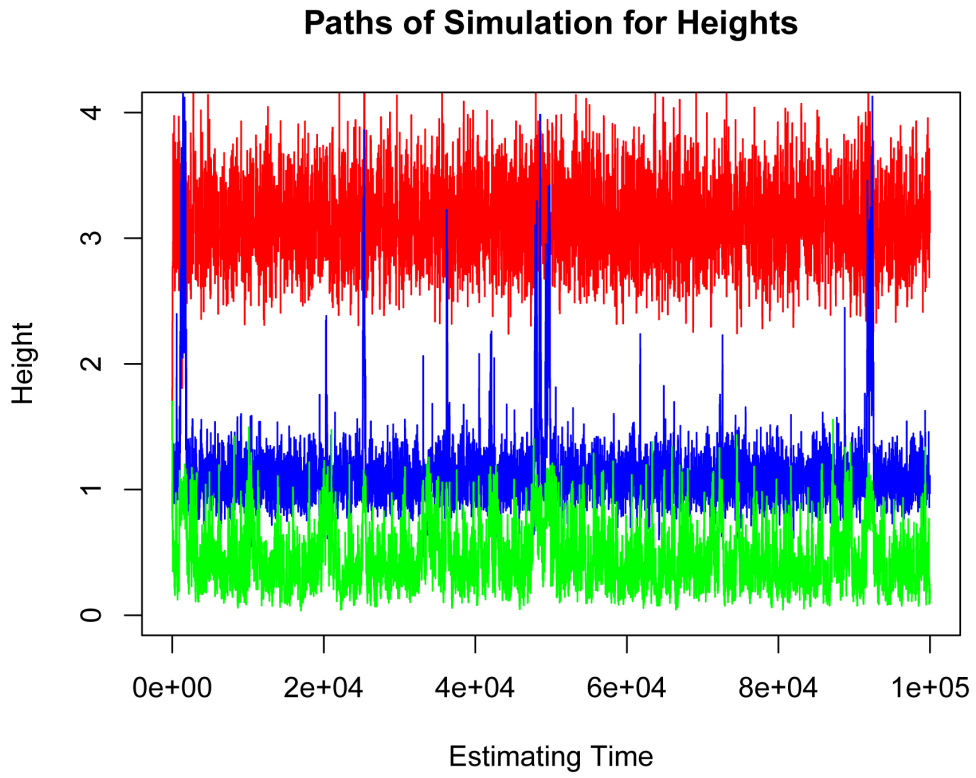


Figure 5.6: The paths and distribution of positions for 10,000 steps of the Markov chain given $k = 3$. *Red:*s_2 *Bule:*s_3.



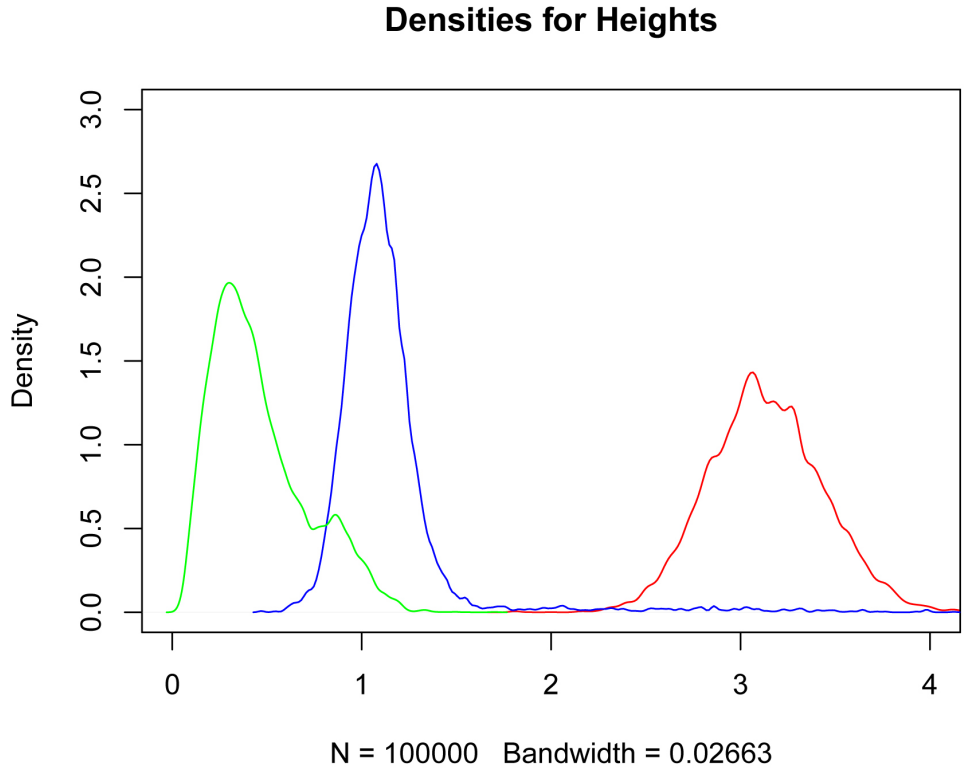


Figure 5.7: The paths and distribution of heights for 100,000 steps of the Markov chain given $k = 3$. *Red:*h_1 *Bule:*h_2 *Green:*h_3.

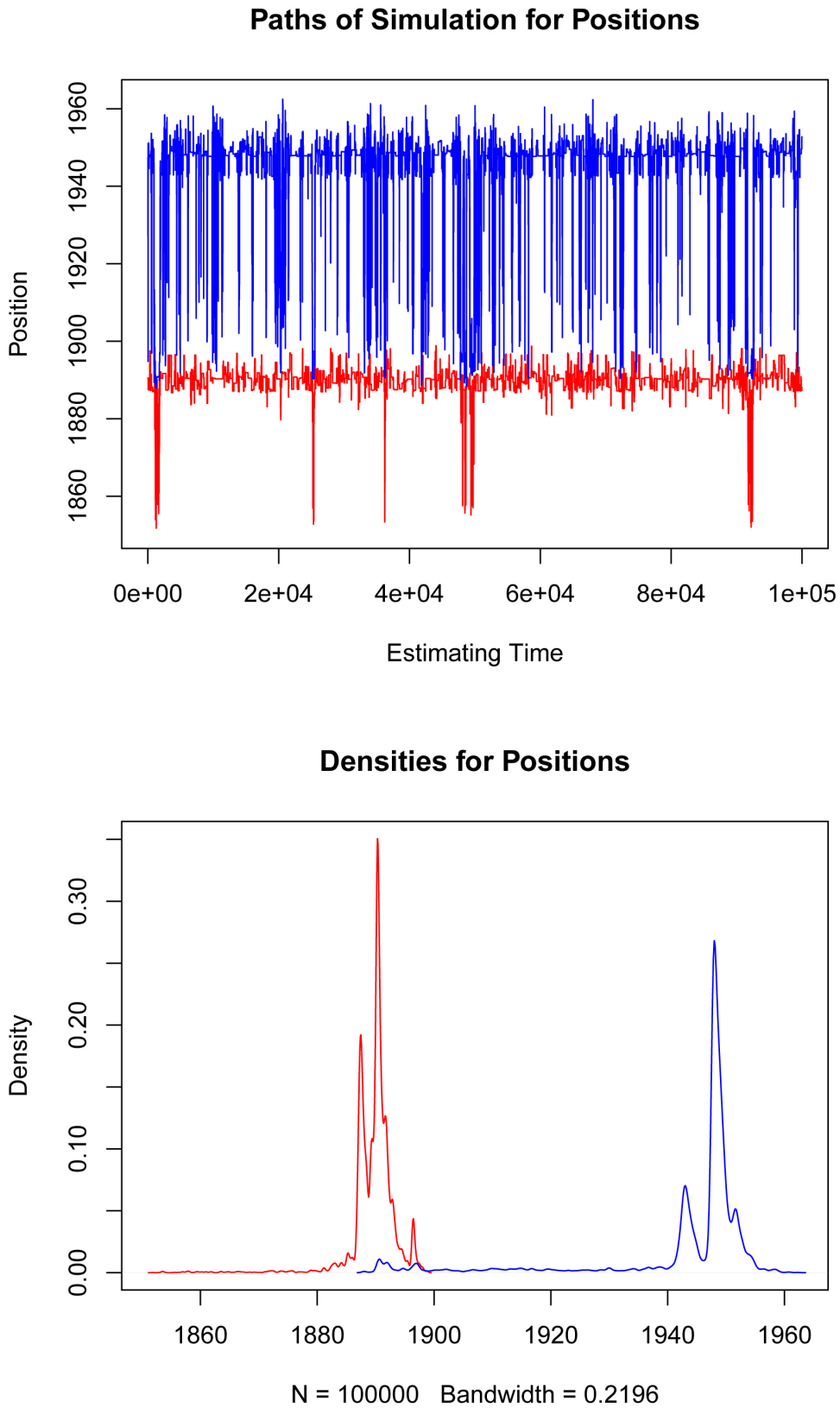
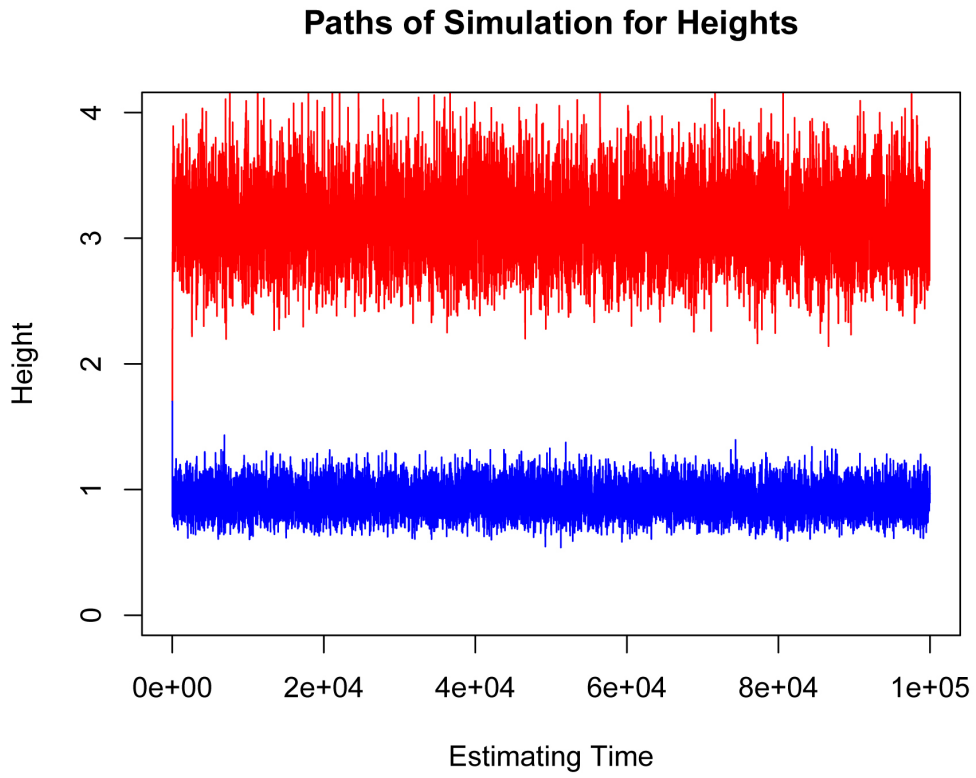


Figure 5.8: The paths and distribution of positions for 100,000 steps of the Markov chain given $k = 3$. *Red:*s_2 *Bule:*s_3.



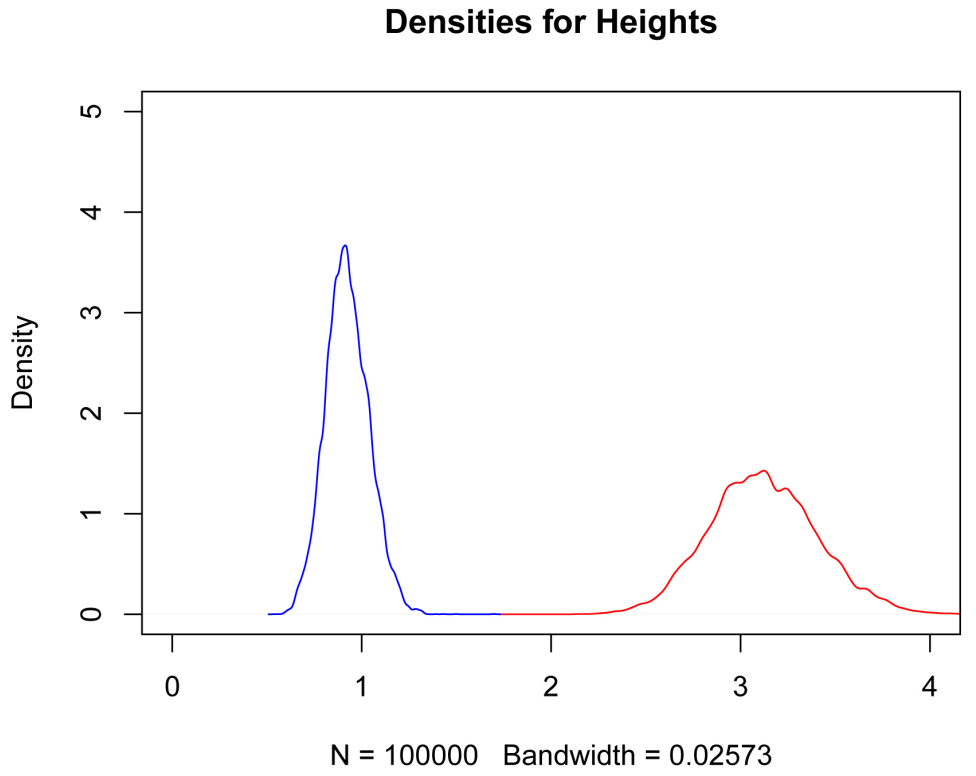
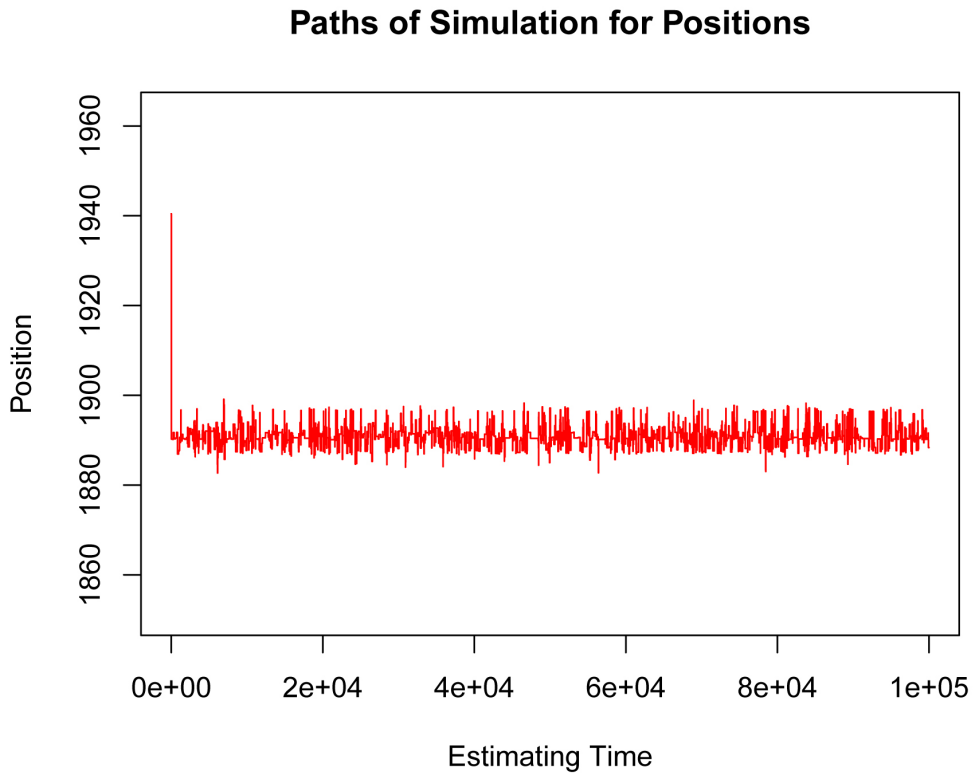


Figure 5.9: The paths and distribution of heights for 100,000 steps of the Markov chain given $k = 2$. *Red:*h_1 *Bule:*h_2.



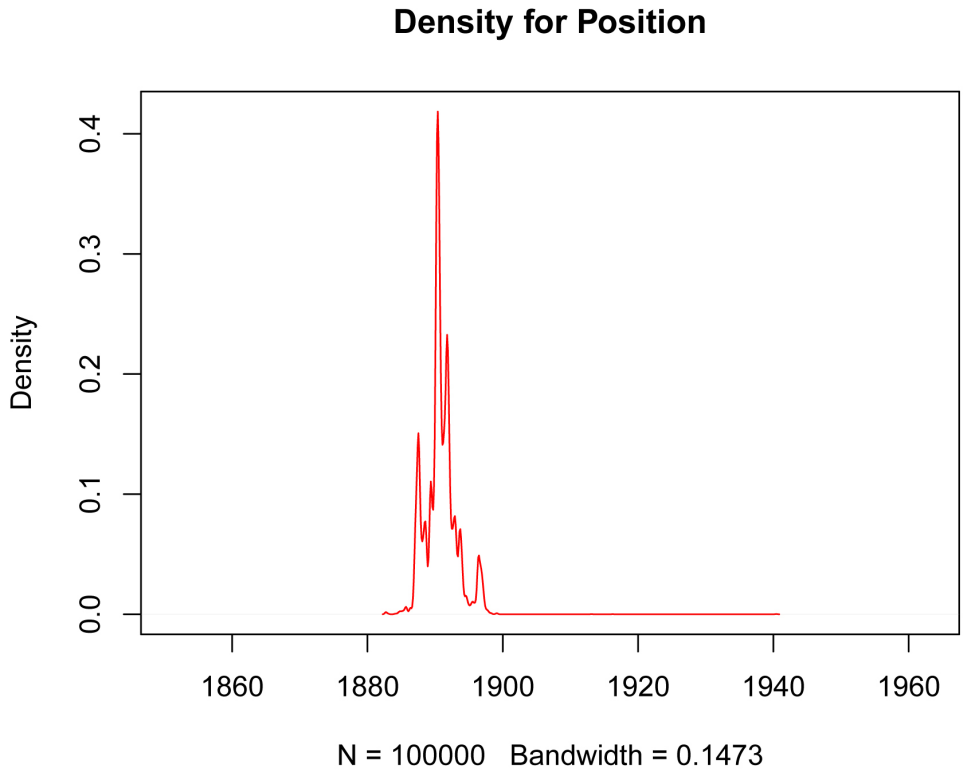
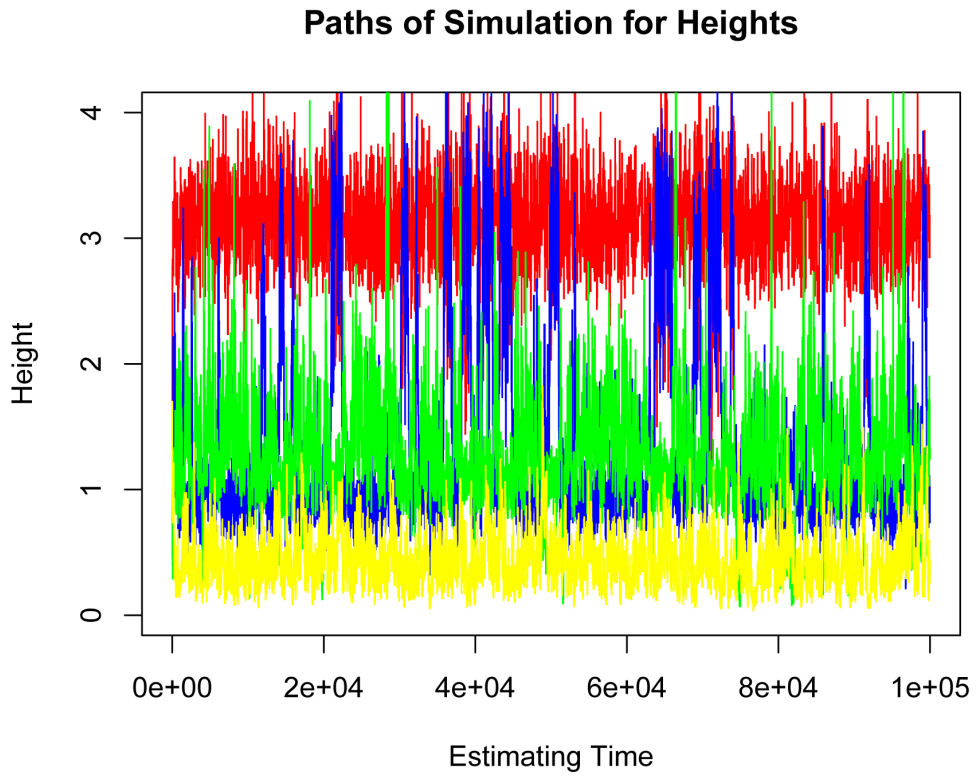


Figure 5.10: The paths and distribution of positions for 100,000 steps of the Markov chain given $k = 2$. *Red:*s_2.



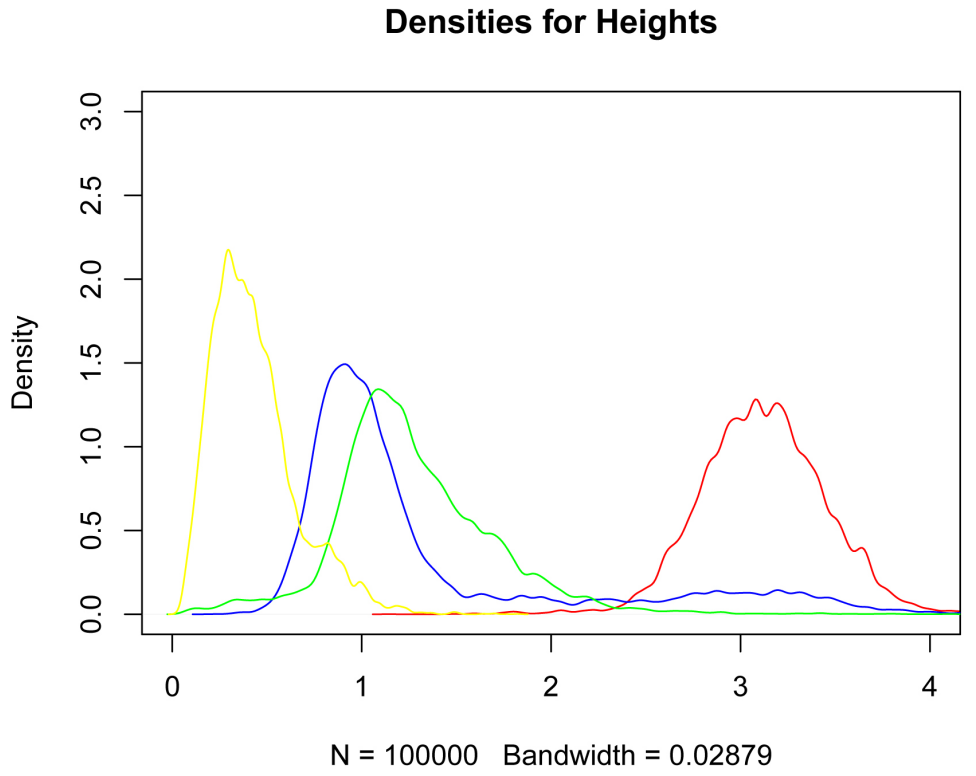
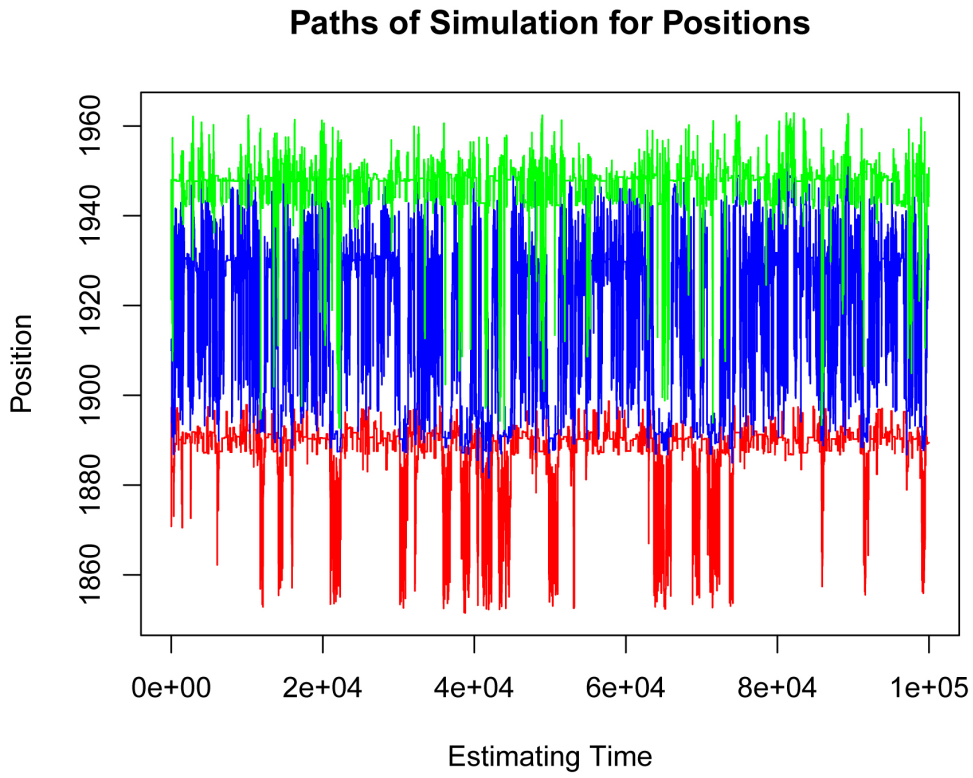


Figure 5.11: The paths and distribution of heights for 100,000 steps of the Markov chain given $k = 4$. *Red:*h_1 *Bule:*h_2 *Green:*h_3 *Yellow:*h_4.



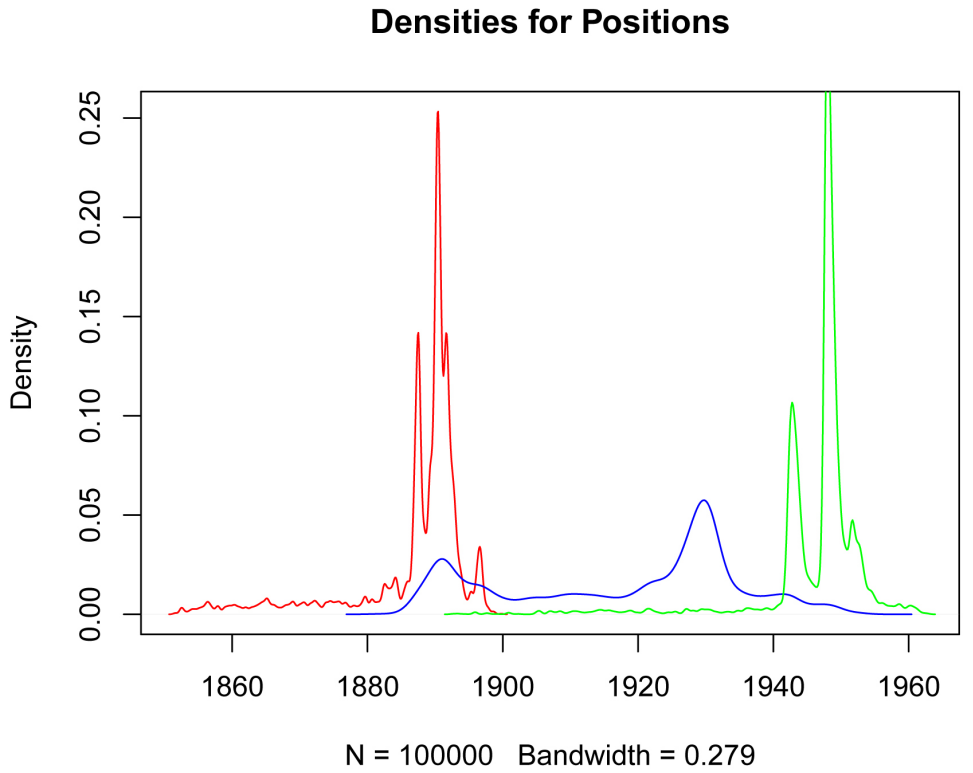


Figure 5.12: The paths and distribution of positions for 100,000 steps of the Markov chain given $k = 4$. *Red:*s_2 *Bule:*s_3 *Green:*s_4.

5.5 The third kind of moves: Number of Steps

After fixing k, we now make k flexible. Green (1995) found a way for Markov chain to jump among different model subspaces with different dimensionality.

Assumed subspaces $\mathscr{C}_k = \{k\} \times \mathscr{R}^{2k+1}$ where $k \in \mathbb{N}^+$ and $1 \leqslant k \leqslant k_{max}$. k is drawn from the Poisson distribution conditioned on $k \leqslant k_{max}$.

$$p(k) = e^{-\lambda}\frac{\lambda^{k-1}}{(k-1)!}$$

Parameter: $x \in \mathscr{C}_k$ where $x = (k, h_1, \ldots, h_j, \ldots, h_k, s_2, \ldots, s_j, s_{j+1}, \ldots, s_k)$. Assume $\theta^{(k)} = (h_1, \ldots, h_j, \ldots, h_k, s_2, \ldots, s_j, s_{j+1}, \ldots, s_k)$, thus $x = (k, \theta^{(k)})$

Prior distribution: $p(x) = p(k) \cdot \frac{(2k-1)!}{(s_{k+1}-s_1)^{2k-1}}(s_2 - s_1)\ldots(s_{j+1}-s_j)\ldots(s_{k+1}-s_k) \cdot \prod_{j=1}^{k} \frac{\beta^\alpha h_j^{\alpha-1}e^{-\beta h_j}}{\Gamma(\alpha)}$

Data: $\{y_i, i = 1, 2, \ldots, n\} \in [s_1, s_{k+1}]$ is a Poisson process. h_j, the jth piece of $\lambda(.)$, is a constant function on $[s_j, s_{j+1})$. m_j is the number of disasters which happened during

$$p(y_1, y_2, \ldots, y_n|x) = \exp(-\sum_{j=1}^{k} h_j(s_{j+1} - s_j) + \sum_{j=1}^{k} m_j \log h_j)$$

5.5.1 Birth Steps

For the birth of steps, $\tilde{k} = k + 1$ and $x \in \mathscr{C}^{k-1}$, $\tilde{x} \in \mathscr{C}^k$.

Green first choose a position s^* uniformly distributed on $[0, L]$. We hypothesize s^* lies within an existing interval (s_j, s_{j+1}). If accepted, s'_{j+1} will be set to be s^*, and $s_{j+1}, s_{j+2}, \cdots, s_k$ will be rebelled as $s'_{j+2}, s'_{j+3}, \cdots, s'_{k+1}$, with corresponding changes to the labelling of step heights.

h_j is a compromise between h'_j and h'_{j+1}. Green uses a weighted geometric mean for this compromise, so that

$$(s^* - s_j)\log h'_j + (s_{j+1} - s^*)\log h'_{j+1} = (s_{j+1} - s_j)\log h_j$$

and defines the perturbation to be such that

$$h'_{j+1} = \frac{1-u}{u}h'_j$$

with u drawn uniformly from $[0, 1]$.

The probability of choosing move x is denoted by $j(x)$.

Bayesian parameter estimates M-H Algorithm:

Input:

target density

$\pi(k) = \frac{1}{Z} p(y_1, y_2, \ldots, y_n|x)p(x)$

where Z is the constant value of $\int p(y_1, y_2, \ldots, y_n|\tilde{x})p(\tilde{x})\, d\tilde{x}$

transition density $q(x, \tilde{x}) = q(u^{(k)}) = q(u) = 1$

$x^0 \in \mathscr{C}_k$ with $\pi(x^0) > 0$

Output:

a Markov chain with stationary density $\pi(x)$

for $n = 1, 2, 3 \ldots$

generate $s^* \sim U(s_j, s_{j+1})$

generate $u \sim U[0, 1]$

calculate h'_j, h'_{j+1} from $h'_j = h_j(\frac{u}{1-u})^{\frac{s_{j+1}-s^*}{s_{j+1}-s_j}}$, $h'_{j+1} = h_j(\frac{1-u}{u})^{\frac{s^*-s_j}{s_{j+1}-s_j}}$

generate $U_n \sim U[0, 1]$

if $U_n \leqslant \alpha(x, \tilde{x})$ where

$$\alpha(x, \tilde{x}) = \min(\frac{p(y_1, y_2, \ldots, y_n|\tilde{x})}{p(y_1, y_2, \ldots, y_n|x)} \cdot \frac{p(\tilde{x})}{p(x)} \cdot \frac{j(\tilde{x})}{j(x)q(u^{(k)})} \cdot \Big|\frac{d(\theta^{(\tilde{k})})}{d(\theta^{(k)}, u^{(k)})}\Big|, 1) \tag{5.5}$$

then $x^n \longleftarrow \tilde{x}^n$ (accepted)

else $x^n \longleftarrow x^{n-1}$ (rejected)

Here we explain how to get the acceptance probability (5.5).

1. We first compute the ratio of the likelihood.

$$\begin{aligned}
&\frac{p(y_1, y_2, \ldots, y_n|\tilde{x})}{p(y_1, y_2, \ldots, y_n|x)} \\
=&\frac{\exp(-(\sum_{i=1}^{j-1} + \sum_{i=j+1}^{k})h_i(s_{i+1} - s_i) - h'_j(s^* - s_j) - h'_{j+1}(s_{j+1} - s^*))}{\exp(-\sum_{j=1}^{k} h_j(s_{j+1} - s_j) + \sum_{j=1}^{k} m_j \log h_j)} \\
&\times \exp((\sum_{i=1}^{j-1} + \sum_{i=j+1}^{k})m_i \log h_i + m'_j \log h'_j + m'_{j+1} \log h'_{j+1}) \\
=&\exp(-h'_j(s^* - s_j) - h'_{j+1}(s_{j+1} - s^*) + m'_j \log h'_j + m'_{j+1} \log h'_{j+1}
\end{aligned}$$

$$
\begin{aligned}
&- (-h_j(s_{j+1} - s_j) + m_j \log h_j)) \\
=&\exp(-h'_j(s^* - s_j) - h'_{j+1}(s_{j+1} - s^*) + m'_j \log h'_j + m'_{j+1} \log h'_{j+1} \\
&+ h_j(s_{j+1} - s_j) - m_j \log h_j)
\end{aligned}
$$

2. Next, we compute the ratio of prior distribution.

$$
\begin{aligned}
\frac{p(\tilde{x})}{p(x)} =& \frac{p(k+1) \cdot \frac{(2(k+1)-1)!}{(s_{k+1}-s_1)^{2(k+1)-1}}(s_2 - s_1)\dots(s^* - s_j)(s_{j+1} - s^*)\dots(s_{k+1} - s_k)}{p(k) \cdot \frac{(2k-1)!}{(s_{k+1}-s_1)^{2k-1}}(s_2 - s_1)\dots(s_{j+1} - s_j)\dots(s_{k+1} - s_k)} \\
&\times \frac{(\prod_{i=1}^{j-1} \times \prod_{i=j+1}^{k}) \frac{\beta^\alpha h_i^{\alpha-1} e^{-\beta h_i}}{\Gamma(\alpha)} \times \frac{\beta^\alpha {h'_j}^{\alpha-1} e^{-\beta h'_j}}{\Gamma(\alpha)} \times \frac{\beta^\alpha {h'_{j+1}}^{\alpha-1} e^{-\beta h'_{j+1}}}{\Gamma(\alpha)}}{\prod_{j=1}^{k} \frac{\beta^\alpha h_j^{\alpha-1} e^{-\beta h_j}}{\Gamma(\alpha)}} \\
=& \frac{e^{-\lambda}\frac{\lambda^k}{k!}}{e^{-\lambda}\frac{\lambda^{k-1}}{(k-1)!}} \frac{2k(2k+1)}{(s_{k+1}-s_1)^2} \frac{(s^* - s_j)(s_{j+1} - s^*)}{(s_{j+1} - s_j)} \times \frac{\beta^\alpha}{\Gamma(\alpha)} \big(\frac{h'_j h'_{j+1}}{h_j}\big)^{\alpha-1} \exp\{-\beta(h'_j + h'_{j+1} - h_j)\} \\
=& \frac{2\lambda(2k+1)}{(s_{k+1}-s_1)^2} \frac{(s^* - s_j)(s_{j+1} - s^*)}{(s_{j+1} - s_j)} \times \frac{\beta^\alpha}{\Gamma(\alpha)} \big(\frac{h'_j h'_{j+1}}{h_j}\big)^{\alpha-1} \exp\{-\beta(h'_j + h'_{j+1} - h_j)\}
\end{aligned}
$$

3. And then compute the proposal ratio.

$$
\frac{j(\tilde{x})}{j(x)q(u^{(k)})} = \frac{d_k \cdot p(s_{j+1})}{b_{k-1} \cdot p(s^*) \cdot q(u)} = \frac{d_k \cdot \frac{1}{k}}{b_{k-1} \cdot \frac{1}{s_{k+1} - s_1} \cdot \frac{1}{1-0}} = \frac{d_k(s_{k+1} - s_1)}{b_{k-1}k}
$$

b_{k-1} means the probability of choosing the move from $\mathscr{C}_k$ to $\mathscr{C}_{k+1}$;
d_k means the probability of choosing the move from $\mathscr{C}_{k+1}$ to $\mathscr{C}_k$.
The subscript of probability means the number of change-points in the starting subspace.

4. Last is the Jacobian.

$$
\begin{aligned}
&\Big|\frac{d(\theta^{(\tilde{k})})}{d(\theta^{(k)}, u^{(k)})}\Big| \\
=&\Big|\frac{d(s'_2, s'_3, \dots, s'_j, s'_{j+1}, s'_{j+2}, \dots, s'_{k+1}, h'_1, h'_2, \dots, h'_j, h'_{j+1}, h'_{j+2}, \dots, h'_{k+1})}{d(s_2, s_3, \dots, s_j, s^*, s_{j+1} \dots, s_k, h_1, h_2, \dots, h_j, u, h_{j+1} \dots, h_k)}\Big| \\
=&\begin{vmatrix} \frac{\partial S'}{\partial S} & \frac{\partial S'}{\partial H} \\ \frac{\partial H'}{\partial S} & \frac{\partial H'}{\partial H} \end{vmatrix}
\end{aligned}
$$

$$
= \left|\begin{array}{ccccccccccccccc}
1 & 0 & \cdots & & & & \cdots & 0 & 0 & 0 & \cdots & & & \cdots & 0 \\
0 & 1 & \cdots & & & & \cdots & 0 & 0 & & & & & & 0 \\
0 & 0 & 1 & & & & \cdots & \vdots & \vdots & & & & & & \vdots \\
\vdots & & & \ddots & & & & & & & & & & & \\
 & & & & \ddots & & & & & & & & & & \\
 & & & & & \ddots & & \vdots & \vdots & & & & & & \vdots \\
\vdots & & & & & & 1 & 0 & 0 & & & & & & 0 \\
0 & \cdots & & & & & \cdots & 1 & 0 & 0 & \cdots & & & \cdots & 0 \\
0 & 0 & \cdots & & & & \cdots & 0 & 1 & 0 & \cdots & & & \cdots & 0 \\
0 & & & & & & & 0 & 0 & 1 & & & & & \\
\vdots & & & & & & & 0 & 0 & & \ddots & & & & \\
\vdots & & & \frac{\partial h'_j}{\partial s_j} & \frac{\partial h'_j}{\partial s^*} & \frac{\partial h'_j}{\partial s_{j+1}} & & \vdots & \vdots & & & \frac{\partial h'_j}{\partial h_j} & \frac{\partial h'_j}{\partial u} & & \\
\vdots & & & \frac{\partial h'_{j+1}}{\partial s_j} & \frac{\partial h'_{j+1}}{\partial s^*} & \frac{\partial h'_{j+1}}{\partial s_{j+1}} & & \vdots & \vdots & & & \frac{\partial h'_{j+1}}{\partial h_j} & \frac{\partial h'_{j+1}}{\partial u} & & \\
0 & & & & & & & 0 & 0 & & & & & \ddots & 0 \\
0 & 0 & \cdots & & & & \cdots & 0 & 0 & 0 & & & & 0 & 1
\end{array}\right|
$$

$$
=1 \times 1 \times \cdots \times \begin{vmatrix} \frac{\partial h'_j}{\partial h_j} & \frac{\partial h'_j}{\partial u} \\ \frac{\partial h'_{j+1}}{\partial h_j} & \frac{\partial h'_{j+1}}{\partial u} \end{vmatrix} \times \cdots \times 1
$$

(the property of lower triangular matrix (**))

$$
= \begin{vmatrix} \frac{\partial h'_j}{\partial h_j} & \frac{\partial h'_j}{\partial u} \\ \frac{\partial h'_{j+1}}{\partial h_j} & \frac{\partial h'_{j+1}}{\partial u} \end{vmatrix}
$$

Here properties (**) is given by theorems:

Theorem 2. [3] *The determinant of an upper triangular or a lower triangular matrix is the product of the entries on its main diagonal.*

[3] Roman (2005), pp.4

Theorem 3. [4] *If a square matrix M has the block diagonal form*

$$M = \begin{pmatrix} B_1 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & B_n \end{pmatrix}$$

then $\det(M) = \prod \det(B_i)$.

Thus,

$$\begin{aligned}
\Big|\frac{d(\theta^{(\tilde{k})})}{d(\theta^{(k)}, u^{(k)})}\Big| &= \begin{vmatrix} (\frac{u}{1-u})^r & h_j \cdot r(\frac{u}{1-u})^{r-1}\frac{1}{(1-u)^2} \\ (\frac{1-u}{u})^{1-r} & h_j(1-r)(\frac{1-u}{u})^{-r}(-\frac{1}{u^2}) \end{vmatrix} \\
&= |-h_j\{(1-r)\frac{u^{2r-2}}{(1-u)^{2r}} + r\cdot\frac{u^{2r-2}}{(1-u)^{2r}}\}| \\
&= h_j\frac{u^{2r-2}}{(1-u)^{2r}} \qquad \text{(where } r = \frac{s_{j+1}-s^*}{s_{j+1}-s_j} \text{ and } 1-r = \frac{s^*-s_j}{s_{j+1}-s_j}\text{)} \\
&= h_j\frac{[u^{r-1}\cdot u + u^{r-1}(1-u)]^2}{(1-u)^{2r}} \\
&= \frac{\big[h_j\big(\frac{u}{1-u}\big)^r + h_j\big(\frac{1-u}{u}\big)^{1-r}\big]^2}{h_j} \\
&= \frac{(h'_j + h'_{j+1})^2}{h_j}
\end{aligned}$$

Combining these formulas together, we get the acceptance probability.

$$\begin{aligned}
\alpha(x,\tilde{x}) &= \min\Big(\frac{p(\tilde{x}|y_1,y_2,\ldots,y_n)}{p(x|y_1,y_2,\ldots,y_n)}\cdot\frac{j(\tilde{x})}{j(x)q(u^{(k)})}\cdot\Big|\frac{d(\theta^{(\tilde{k})})}{d(\theta^{(k)},u^{(k)})}\Big|, 1\Big) \\
&\qquad\qquad \text{(note that } p(x|y) = p(y|x)p(x)/p(y)\text{)} \\
&= \min\Big(\frac{p(y_1,y_2,\ldots,y_n|\tilde{x})}{p(y_1,y_2,\ldots,y_n|x)}\cdot\frac{p(\tilde{x})}{p(x)}\cdot\frac{j(\tilde{x})}{j(x)q(u^{(k)})}\cdot\Big|\frac{d(\theta^{(\tilde{k})})}{d(\theta^{(k)},u^{(k)})}\Big|, 1\Big) \\
&= \min(\exp(-h'_j(s^*-s_j) - h'_{j+1}(s_{j+1}-s^*) + m'_j\log h'_j + m'_{j+1}\log h'_{j+1} + h_j(s_{j+1}-s_j) - m_j\log h_j) \\
&\quad\times\frac{2\lambda(2k+1)}{(s_{k+1}-s_1)^2}\frac{(s^*-s_j)(s_{j+1}-s^*)}{(s_{j+1}-s_j)}\times\frac{\beta^\alpha}{\Gamma(\alpha)}\Big(\frac{h'_jh'_{j+1}}{h_j}\Big)^{\alpha-1}\exp\{-\beta(h'_j+h'_{j+1}-h_j)\} \\
&\quad\times\frac{d_k(s_{k+1}-s_1)}{b_{k-1}k}\times\frac{(h'_j+h'_{j+1})^2}{h_j}, 1)
\end{aligned}$$

[4] Roman (2005), pp.4

$$
\begin{aligned}
&= \min(\exp\{-h'_j(s^* - s_j) - h'_{j+1}(s_{j+1} - s^*) + m'_j \log h'_j + m'_{j+1} \log h'_{j+1} + h_j(s_{j+1} - s_j) - m_j \log h_j \\
&\quad + \log 2\lambda(2k+1) \underline{-2\log(s_{k+1} - s_1)} + \log(s^* - s_j) + \log(s_{j+1} - s^*) - \log(s_{j+1} - s_j) \\
&\quad + \alpha \log \beta - \log \Gamma(\alpha) + (\alpha - 1) \log h'_j + (\alpha - 1) \log h'_{j+1} \underline{-(\alpha - 1) \log h_j} - \beta(h'_j + h'_{j+1} - h_j) \\
&\quad + \log d_k \underline{+ \log(s_{k+1} - s_1)} - \log b_{k-1} - \log k + 2\log(h'_j + h'_{j+1}) \underline{- \log h_j}\}, 1) \\
&\qquad\qquad \text{(parts of underlined terms balance out)} \\
&= \min(\exp\{-h'_j(s^* - s_j) - h'_{j+1}(s_{j+1} - s^*) + m'_j \log h'_j + m'_{j+1} \log h'_{j+1} + h_j(s_{j+1} - s_j) - m_j \log h_j \\
&\quad + \log 2 + \log \lambda + \log(2k+1) \underline{- \log(s_{k+1} - s_1)} + \log(s^* - s_j) + \log(s_{j+1} - s^*) - \log(s_{j+1} - s_j) \\
&\quad + \alpha \log \beta - \log \Gamma(\alpha) + (\alpha - 1)(\log h'_j + \log h'_{j+1}) \underline{- \alpha \log h_j} - \beta(h'_j + h'_{j+1} - h_j) \\
&\quad + \log d_k - \log b_{k-1} - \log k + 2\log(h'_j + h'_{j+1})\}, 1)
\end{aligned}
$$

Applying all of these formulas to an R program, we get

```
alpha.birth <- function(s,h,s_star,h1_prime,h2_prime,j,k) {
        m1_prime <- count.disasters(s[j],s_star)
        m2_prime <- count.disasters(s_star,s[j+1])
        mj <- m1_prime + m2_prime
        log_likelihood_ratio <- (
           - h1_prime * (s_star - s[j]) - h2_prime * (s[j+1] - s_star)
           + m1_prime * log(h1_prime) + m2_prime * log(h2_prime)
           + h[j] * (s[j+1] - s[j]) - mj * log(h[j])
        )
        log_prior_ratio <- (
           log(2) + log(lambda) + log(2*k+1) - log(s[k+1] - s[1])
           + log(s_star - s[j]) + log (s[j+1] - s_star)
           - log(s[j+1] - s[j]) + a * log(b) - log(gamma(a))
           + (a-1) * (log(h1_prime) + log(h2_prime)) - a * log(h[j])
           - b * (h1_prime + h2_prime - h[j])
        )
        log_proposal_ratio <- log(death(k)) - log(birth(k-1)) - log(k)
        log_Jacobian <- 2 * log(h1_prime + h2_prime)
        log_pi_ratio <- log_likelihood_ratio + log_prior_ratio
                        + log_proposal_ratio + log_Jacobian
        return(exp(log_pi_ratio))
}
```

5.5.2 Death Steps

For the death of steps, $\tilde{k} = k - 1$ and $x \in \mathscr{C}^k$, $\tilde{x} \in \mathscr{C}^{k-1}$.

If s_{j+1} is removed, the new height over the interval $(s'_j, s'_{j+1}) = (s_j, s_{j+2})$ is h'_j, the

weighted geometric mean satisfying

$$(s_{j+1} - s_j)\log h_j + (s_{j+2} - s_{j+1})\log h_{j+1} = (s'_{j+1} - s'_j)\log h'_j$$

And s_{j+1} that is proposed for removal is drawn at random from $s_2, s_3, \cdots, s_k$.

The acceptance probability for the corresponding death step has the same form with the appropriate change of labelling of the variables, and the ratio terms inverted.

1. likelihood ratio

$$\begin{aligned}&\frac{p(y_1, y_2, \ldots, y_n|\tilde{x})}{p(y_1, y_2, \ldots, y_n|x)}\\ =&\exp(-h'_j(s_{j+2} - s_j) + m'_j \log h'_j + h_j(s_{j+1} - s_j) + h_{j+1}(s_{j+2} - s_{j+1})\\ &- m_j \log h_j - m_{j+1}\log h_{j+1})\end{aligned}$$

2. prior distribution ratio

$$\frac{p(\tilde{x})}{p(x)} = \frac{(s_{k+1} - s_1)^2}{2\lambda(2k+1)}\frac{(s_{j+2} - s_j)}{(s_{j+1} - s_j)(s_{j+2} - s_{j+1})} \times \frac{\Gamma(\alpha)}{\beta^\alpha}\Big(\frac{h'_j}{h_j h_{j+1}}\Big)^{\alpha-1}\exp\{\beta(h_j + h_{j+1} - h'_j)\}$$

3. proposal ratio

$$u = \frac{h_j}{h_j + h_{j+1}}$$

$$\frac{j(\tilde{x})q(u^{(\tilde{k})})}{j(x)} = \frac{b_{k-2}\cdot p(s^*)\cdot q(u)}{d_{k-1}\cdot p(s_{j+1})} = \frac{b_{k-2}\cdot \dfrac{1}{s_{k+1} - s_1}\cdot\dfrac{1}{1-0}}{d_{k-1}\cdot\dfrac{1}{k}} = \frac{b_{k-1}k}{d_k(s_{k+1} - s_1)}$$

d_{k-1} means the probability of choosing the move from $\mathscr{C}_k$ to $\mathscr{C}_{k-1}$,

b_{k-2} means the probability of choosing the move from $\mathscr{C}_{k-1}$ to $\mathscr{C}_k$.

The subscript of probability means the number of change-points in the starting subspace.

4. Jacobian

$$\Big|\frac{d(\theta^{(\tilde{k})}, u^{(\tilde{k})})}{d(\theta^{(k)})}\Big| = \frac{h'_j}{(h_j + h_{j+1})^2}$$

Thus,we have the acceptance probability.

$$\alpha(x,\tilde{x}) = \min\Big(\frac{p(\tilde{x}|y_1, y_2, \ldots, y_n)}{p(x|y_1, y_2, \ldots, y_n)}\cdot\frac{j(\tilde{x})q(u^{(\tilde{k})})}{j(x)}\cdot\Big|\frac{d(\theta^{(\tilde{k})}, u^{(\tilde{k})})}{d(\theta^{(k)})}\Big|, 1\Big)$$

(note that $p(x|y) = p(y|x)p(x)/p(y)$)

$$\begin{aligned}&= \min\Big(\frac{p(y_1, y_2, \ldots, y_n|\tilde{x})}{p(y_1, y_2, \ldots, y_n|x)}\cdot\frac{p(\tilde{x})}{p(x)}\cdot\frac{j(\tilde{x})q(u^{(\tilde{k})})}{j(x)}\cdot\Big|\frac{d(\theta^{(\tilde{k})}, u^{(\tilde{k})})}{d(\theta^{(k)})}\Big|, 1\Big)\\ &= \min(\exp\{-h'_j(s_{j+1} - s_j) + m'_j\log h_j + h_j(s_{j+1} - s_j) + h_{j+1}(s_{j+2} - s_{j+1}) - m_j\log h_j - m_{j+1}\log h_{j+1}\end{aligned}$$

$$
\begin{aligned}
&- \log 2 - \log \lambda - \log(2k+1) + \log(s_{k+1} - s_1) - \log(s_{j+1} - s_j) - \log(s_{j+2} - s_{j+1}) + \log(s_{j+2} - s_j) \\
&- \alpha \log \beta + \log \Gamma(\alpha) - (\alpha - 1)(\log h_j + \log h_{j+1}) + \alpha \log h'_j + \beta(h_j + h_{j+1} - h'_j) \\
&- \log d_{k-1} + \log b_{k-2} + \log k - 2 \log(h_j + h_{j+1})\} \, , 1)
\end{aligned}
$$

Combining all of these formulas into an R program, we get

```
alpha.death <- function(s,h,hj_prime,j,k) {
        m1 <- count.disasters(s[j],s[j+1])
        m2 <- count.disasters(s[j+1],s[j+2])
        mj_prime <- m1 + m2
        log_likelihood_ratio <- (
           - hj_prime * (s[j+2] - s[j]) + mj_prime * log(hj_prime)
           + h[j] * (s[j+1] - s[j]) + h[j+1] * (s[j+2] - s[j+1])
           - m1 * log(h[j]) - m2 * log(h[j+1])
        )
        log_prior_ratio <- (
           - log(2) - log (lambda) - log(2*k-1) + log(s[k+1] - s[1])
           - log(s[j+1] - s[j]) - log (s[j+2] - s[j+1])
           + log(s[j+2] - s[j]) - a * log(b) + log(gamma(a))
           - (a-1) * (log(h[j]) + log(h[j+1])) + a * log(hj_prime)
           + b * (h[j] + h[j+1] - hj_prime)
        )
        log_proposal_ratio <- log(birth(k-2)) + log(k-1) -log(death(k-1))
        log_Jacobian <- - 2 * log(h[j] + h[j+1])
        log_pi_ratio <- log_likelihood_ratio
           + log_prior_ratio + log_proposal_ratio + log_Jacobian
        return(exp(log_pi_ratio))
}
```

Bayesian parameter estimates M-H Algorithm:

Input:
target density
$\pi(k) = \frac{1}{Z} p(y_1, y_2, \ldots, y_n|x)p(x)$
where Z is the constant value of $\int p(y_1, y_2, \ldots, y_n|\tilde{x})p(\tilde{x})\, d\tilde{x}$
transition density $q(x, \tilde{x}) = q(u^{(k)}) = q(u) = 1$
$x^0 \in \mathscr{C}_k$ with $\pi(x^0) > 0$

Output:
a Markov chain with stationary density $\pi(x)$

for $n = 1, 2, 3 \ldots$
calculate h'_j from $h'_j = h_j^{\frac{s_{j+1}-s_j}{s_{j+2}-s_j}} \times h_{j+1}^{\frac{s_{j+2}-s_{j+1}}{s_{j+2}-s_j}}$

generate $U_n \sim U[0,1]$
if $U_n \leqslant \alpha(x, \tilde{x})$ where

$$\alpha(x,\tilde{x}) = \min\left(\frac{p(y_1,y_2,\ldots,y_n|\tilde{x})}{p(y_1,y_2,\ldots,y_n|x)} \cdot \frac{p(\tilde{x})}{p(x)} \cdot \frac{j(\tilde{x})q(u^{(\tilde{k})})}{j(x)} \cdot \left|\frac{d(\theta^{(\tilde{k})}, u^{(\tilde{k})})}{d(\theta^{(k)})}\right|, 1\right) \tag{5.6}$$

then $x^n \longleftarrow \tilde{x}^n$ (accepted)
else $x^n \longleftarrow x^{n-1}$ (rejected)

5.6 Experiment with varying number of steps

The "`count.disasters`" function and the "`alpha.hmove`" function is in section 5.1, 5.2. The "`alpha.smove`" function is ready in section 5.3. The"`alpha.birth`" function is in section 5.5.1 and the"`alpha.death`" function is in section 5.5.2.

With the height, position, birth and death acceptance algorithms, we can now determine a model with flexible number of steps. The parameter λ of prior distribution of k is assumed to be 3. And the probabilities of choosing the move type in each step is discussed in section 4.3.

Here is the M-H algorithm main program:

```
lambda = 3

birth <- function(change_points){
         return(3.6/7 * min(1,lambda/(change_points+1)))
         }
death <- function(change_points){
         return(3.6/7 * min(1,change_points/lambda))
         }

Move <- function(n, h, s, k) {
    H <- list()
    S <- list()
    K <- vector(length=n)
    K[1] <- k
    H[[1]] <- h
    S[[1]] <- s

    for (i in 2:n){
        position_prob <-
           ifelse(k<=1, 0, 0.5 * (1 - birth(k-1) - death(k-1)))

        height_prob <- 1 - birth(k-1) - death(k-1) - position_prob
```

```
type <- runif(1)

# j from 1 to k is for Height h[1] to h[k]
if (type>1-height_prob) {
        j <- sample(1:k,size=1)
        u <- runif(1,-0.5,0.5)
    h_tilde <- h[j] * exp(u)
    U <- runif (1)
    if (U < alpha.hmove(s,h,h_tilde,j)) {
        h[j] <- h_tilde
    }
}


# j from 2 to k is for Position s[2] to s[k]
if (type<=1-height_prob && type>1-height_prob-position_prob){
    j <- sample(1:(k-1), size=1) + 1
    s_tilde <- runif(1,s[j-1],s[j+1])
    U <- runif (1)
    if (U < alpha.smove(s,h,s_tilde,j)) {
        s[j] <- s_tilde
    }
}


# j from 1 to k-1 is for death of steps d[2] to d[k]
if (type>=birth(k-1) && type<=birth(k-1)+death(k-1)) {
    j <- sample(1:(k-1), size=1)
    r <- (s[j+2] - s[j+1]) / (s[j+2] - s[j])
    hj_prime <- h[j]^(1-r) * h[j+1]^r
    U <- runif(1)
    if (U < alpha.death(s,h,hj_prime,j,k)){
        k <- k - 1
        h[j] <- hj_prime
        h <- h[-(j+1)]
        s <- s[-(j+1)]
    }
}


# j from 1 to k is for birth of steps b[1] to b[k]
if (type<=birth(k-1)){
    j <- sample(1:k,size=1)
    s_star <- runif(1,s[j],s[j+1])
```

```
            u <- runif(1)
            r <- exp(log(s[j+1] - s_star) - log(s[j+1] - s[j]))
            h1_prime = h[j] * exp(r * (log(u) - log(1-u)))
            h2_prime = h[j] * exp((1-r) * (log(1-u) - log(u)))
            U <- runif(1)
            if (U < alpha.birth(s,h,s_star,h1_prime,h2_prime,j,k)){
                s <- c(s[1:j], s_star, s[(j+1):(k+1)])
                if (j > 1) {
                    left <- h[1:(j-1)]
                } else {
                    left <- c()
                }
                if (j < k) {
                    right <- h[(j+1):k]
                } else {
                    right <- c()
                }
                h <- c(left, h1_prime, h2_prime, right)
                k <- k + 1
            }
        }

        K[i] <- k
        H[[i]] <- h
        S[[i]] <- s
    }

    return(list(h=H,k=K,s=S))
}
```

We estimates for $1,000$, $10,000$ and $100,000$ times respectively and draw the corresponding histograms of k.

```
k0 <- 1
s0 <- c(1851,1963)
h0 <- c(1.7)
X <- Move(100000, h0, s0, k0)
H <- X$h
K <- X$k
S <- X$s
hist(K,breaks=seq(0.5, max(K)+0.5, by=0.5)+0.20)
```

We can see from figure 5.13,5.14,5.15 that as the estimating step n getting larger, the range of k values tends to be wider. But the proportion of the extreme values like 1 and numbers over 8 becomes smaller and smaller. When n is not large enough, the estimates still stay in burn-

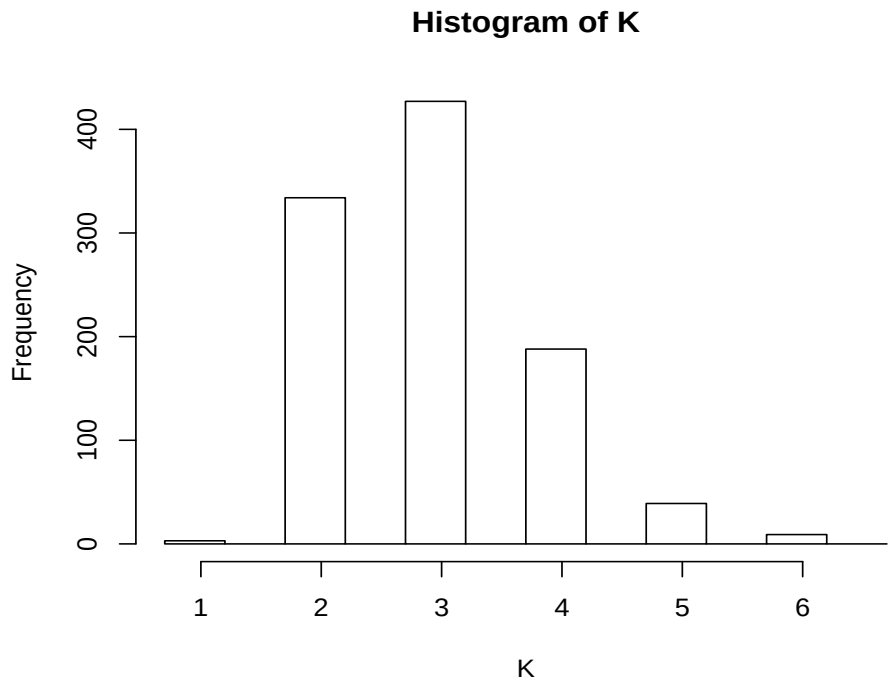


Figure 5.13: Posterior distribution of k, the number of steps, for $1,000$ steps of Markov chain.

in period, like figure 5.13, $k = 2$ still keep high proportion in the histogram. But after burn-in period, the basic distribution of k does not change much as estimating time getting longer. $k = 3$ and $k = 4$ always keep their dominance.

I put the estimate results of the models with the same dimensionality into one matrix. So that I separate several data groups with $k = 2$, $k = 3$, $k = 4$ etc. The commands is put in the appendix.

Comparing figure 5.16,5.17,5.18,5.19,5.20,5.21 correspondingly to figure 5.7,5.8,5.9,5.10,5.11,5.12 in section 5.4, we found that the distributions of heights and positions of original Markov chain and those of jump Markov chain are similar; but their paths of certain heights and positions can be quite different.

1. For Heights

   Comparing figure 5.7(Top) to figure 5.16(Top), the latter's h_2 (blue) jumps close to h_1 much more often than the former's does.

   Comparing figure 5.11(Top) to figure 5.20(Top), the latter's h_2 (blue) jumps close to h_1 much more often than the former's does.

2. For Positions

   Comparing figure 5.8(Top) to figure 5.17(Top), the latter's s_2 (red) jumps forward to approximate 1851 much more often than the former's does.

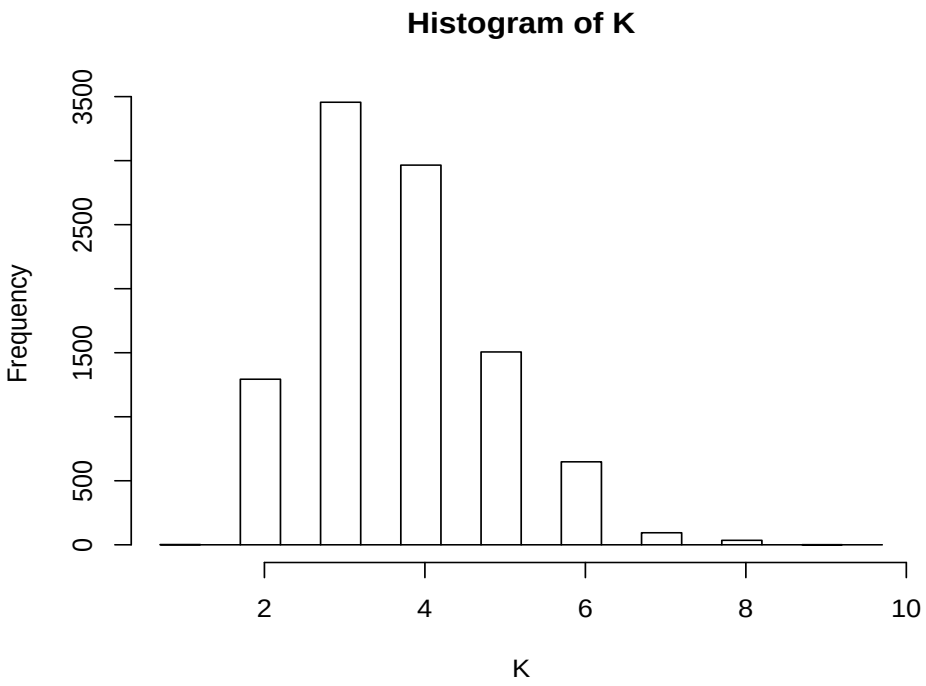


Figure 5.14: Posterior distribution of k, the number of steps, for $10,000$ steps of Markov chain.

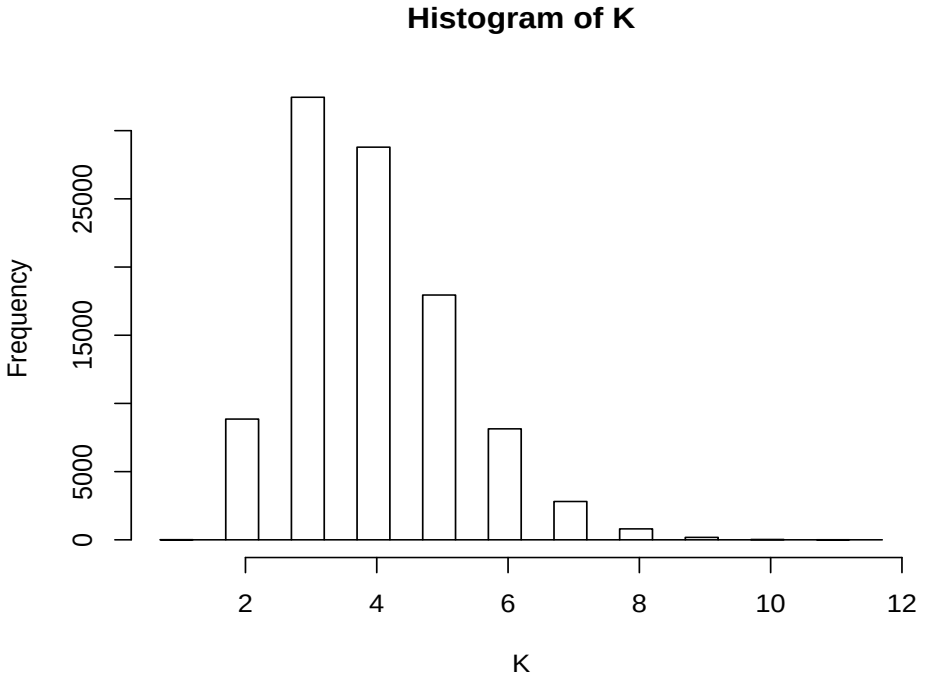
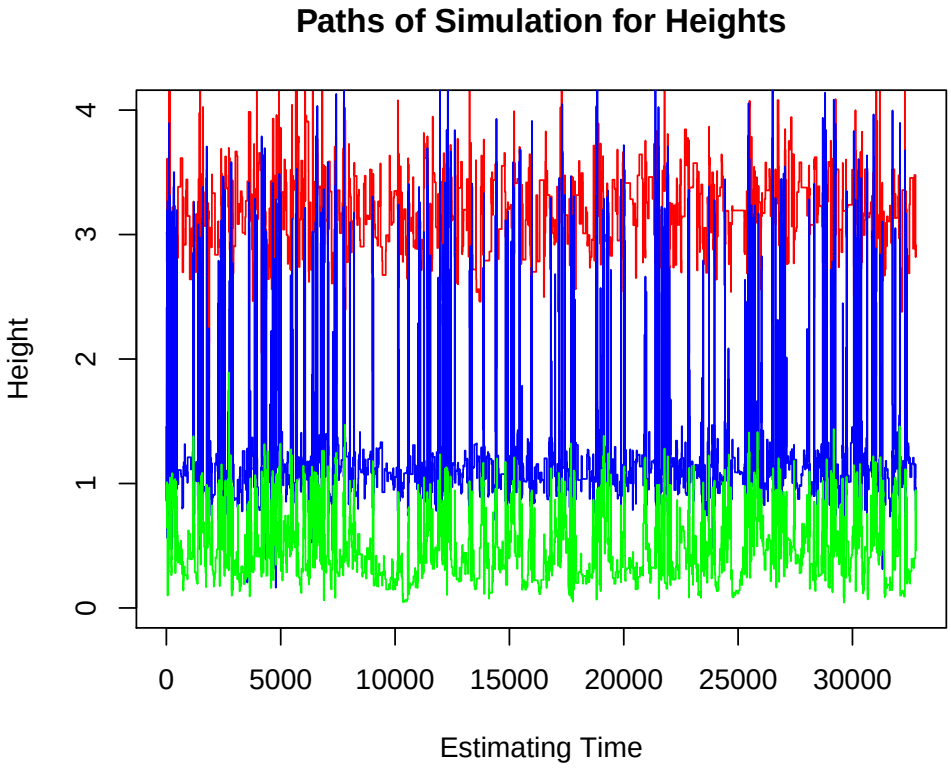


Figure 5.15: Posterior distribution of k, the number of steps, for $100,000$ steps of Markov chain.

Comparing figure 5.12(Top) to figure 5.21(Top), the latter's s_2 (red) jumps forward to approximate 1851 much more often than the former's does.

Although far-jumping of h_2 and s_2 happen more often in jump Markov chain, the distributions of the h_2 and s_2 in jump Markov chain are nearly the same with those in original Markov chain. This result illuminates that heights and positions change more actively in jump Markov chain.

Because of this activity of jump Markov chain, special outcome can happen. Let's have a look at figure 5.22. This is about the distributions of positions in a jump Markov chain conditioned on $k = 3$. Note that there are two peaks in density of s_2 (red), differing from figure 5.8.



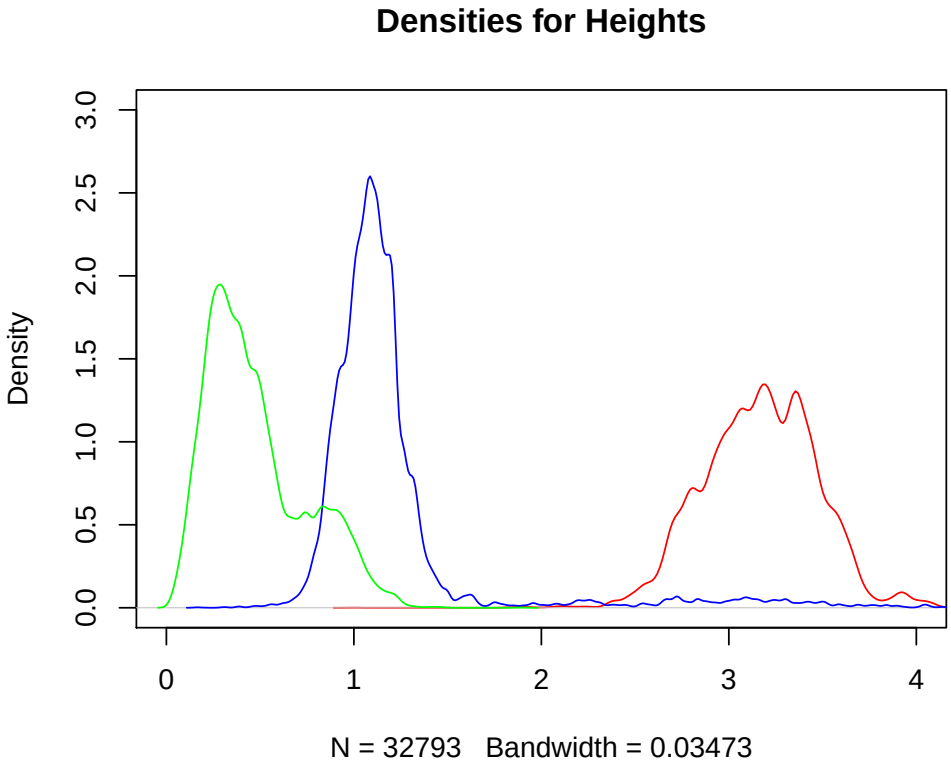


Figure 5.16: The paths and distribution of heights for $100,000$ *steps of Markov chain conditioned on* $k = 3$*. Red:*h_1 *Bule:*h_2 *Green:*h_3.

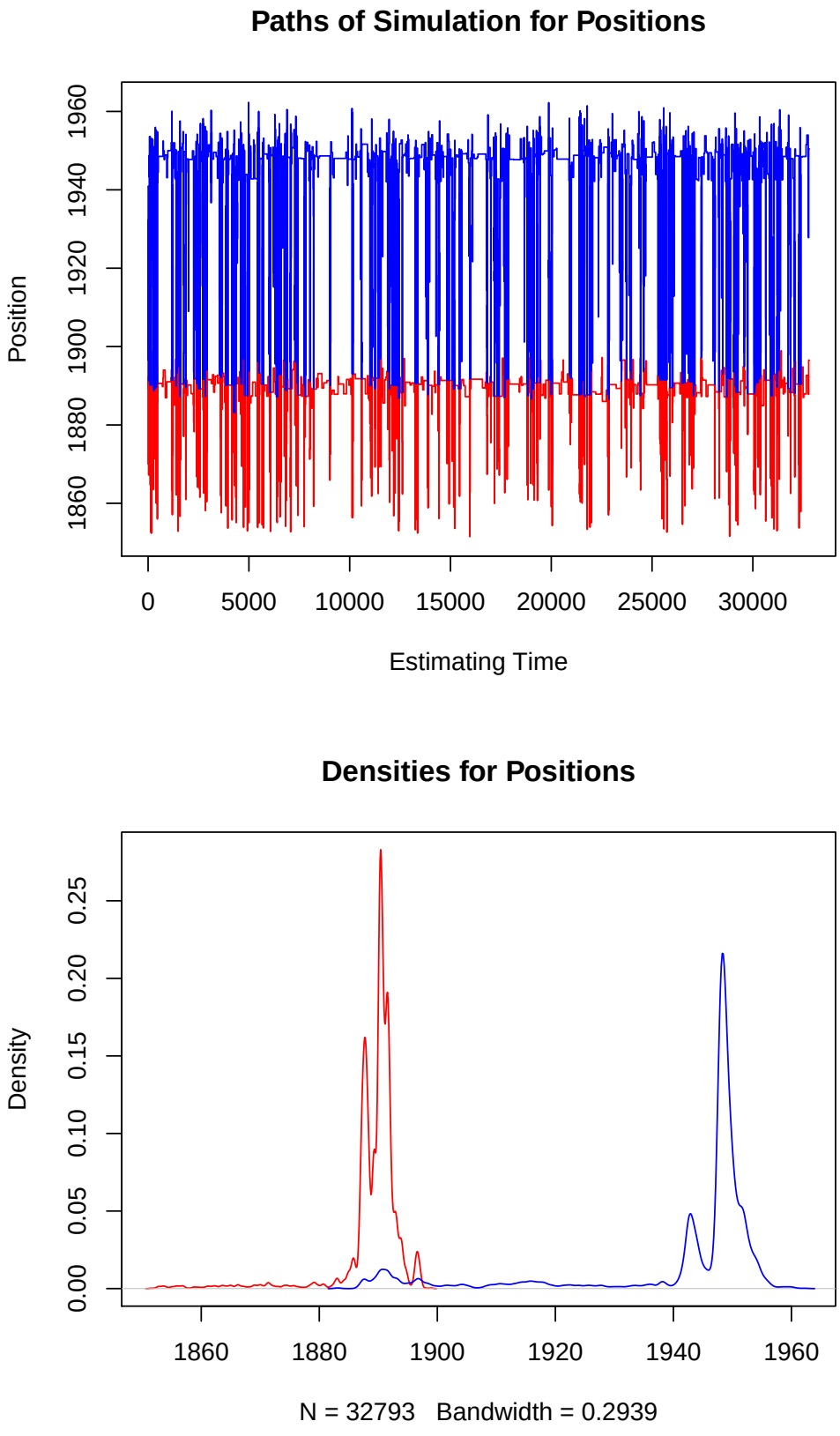
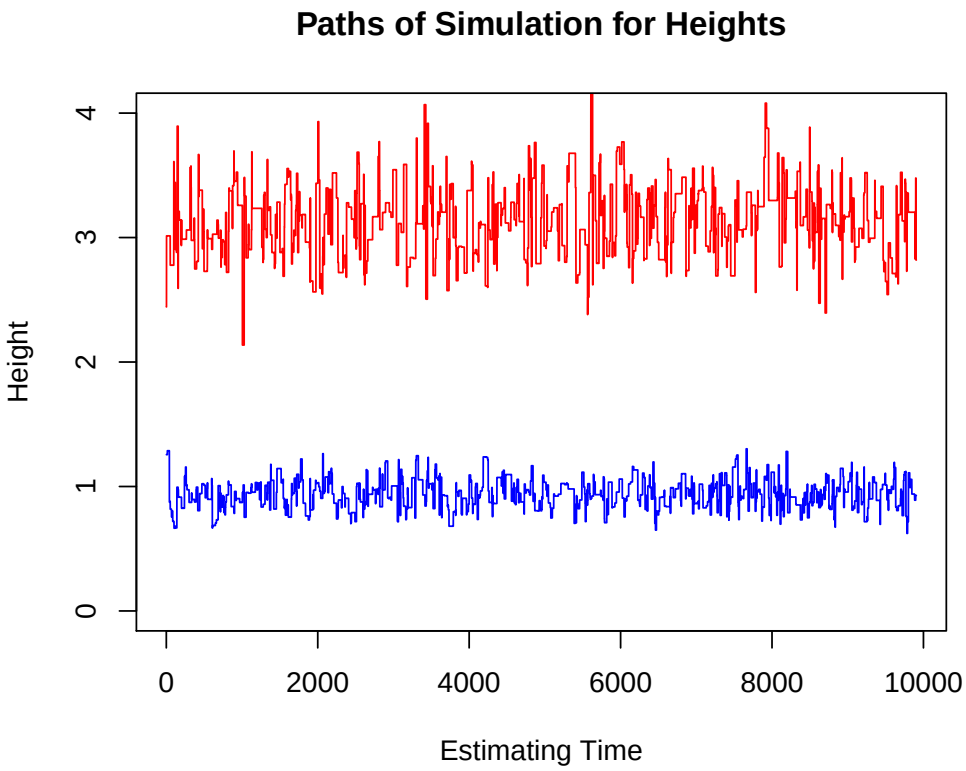


Figure 5.17: The paths and distribution of positions for $100,000$ *steps of Markov chain conditioned on* $k = 3$. *Red:*s_2 *Bule:*s_3.



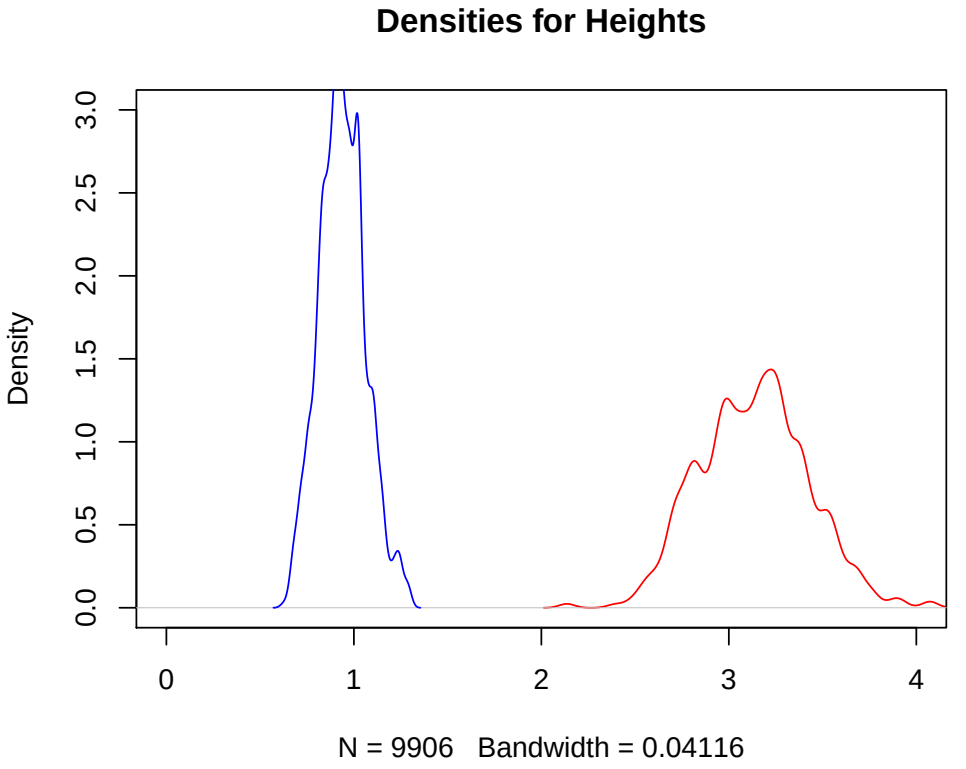
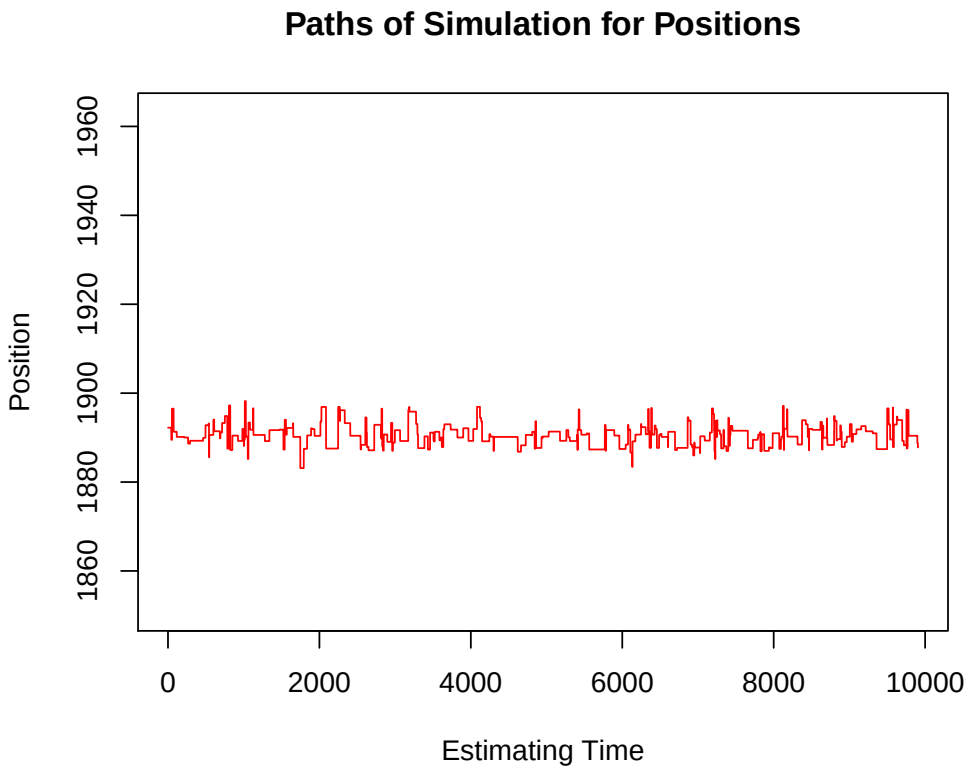


Figure 5.18: The paths and distribution of heights for $100,000$ *steps of Markov chain conditioned on* $k = 2$*. Red:*h_1 *Bule:*h_2*.*



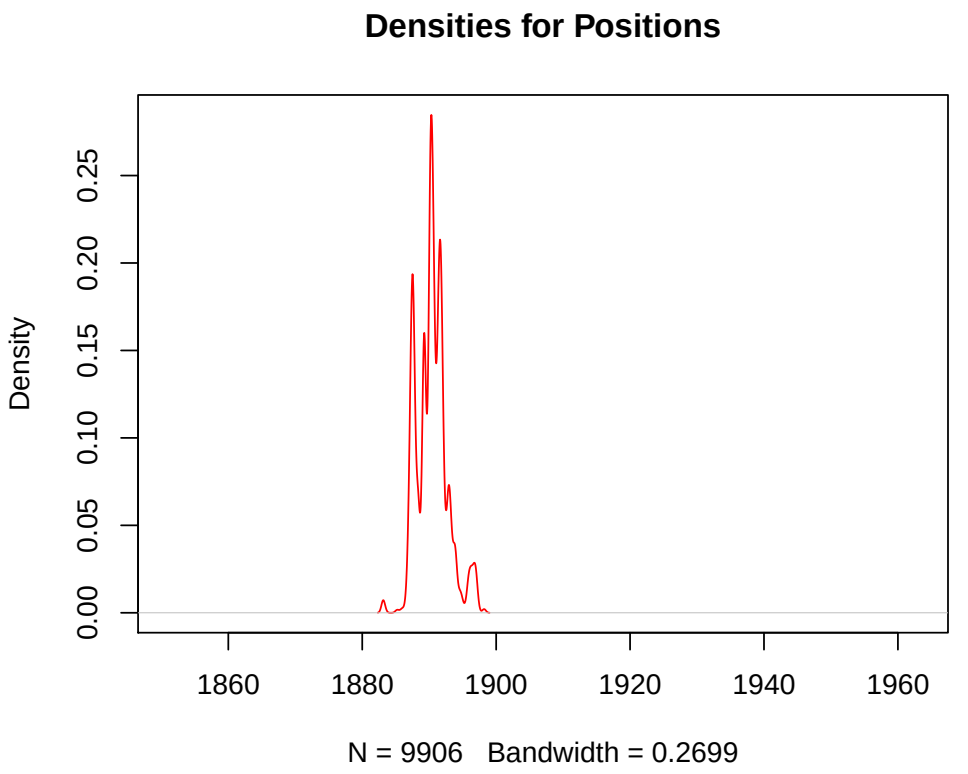
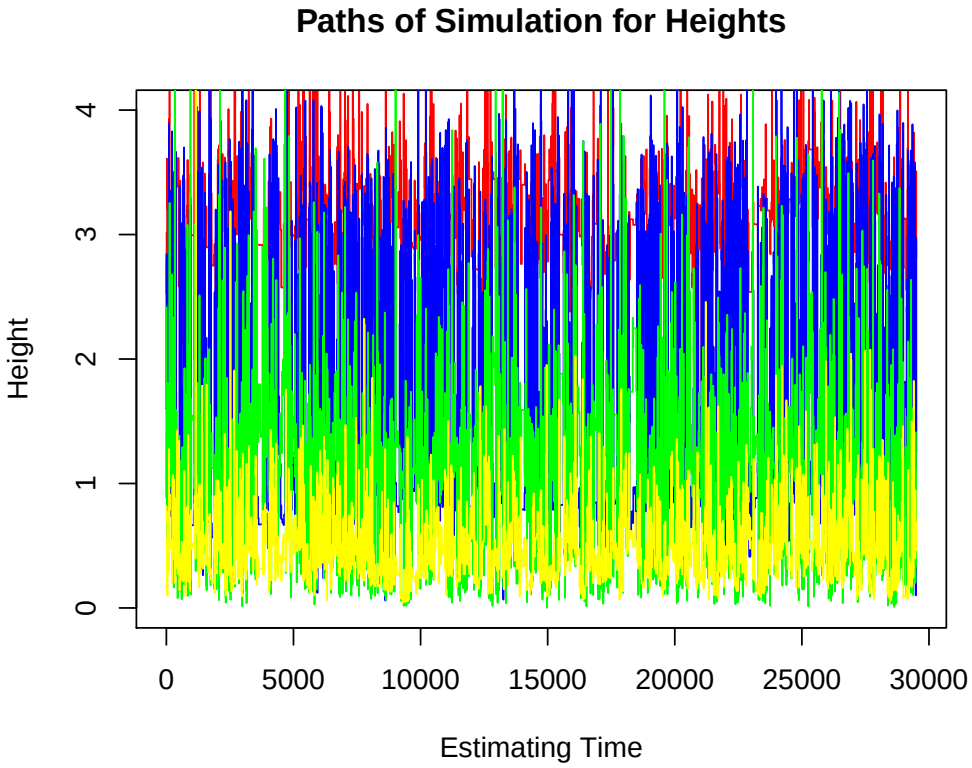


Figure 5.19: The paths and distribution of positions for $100,000$ *steps of Markov chain conditioned on* $k = 2$*. Red:*s_2.



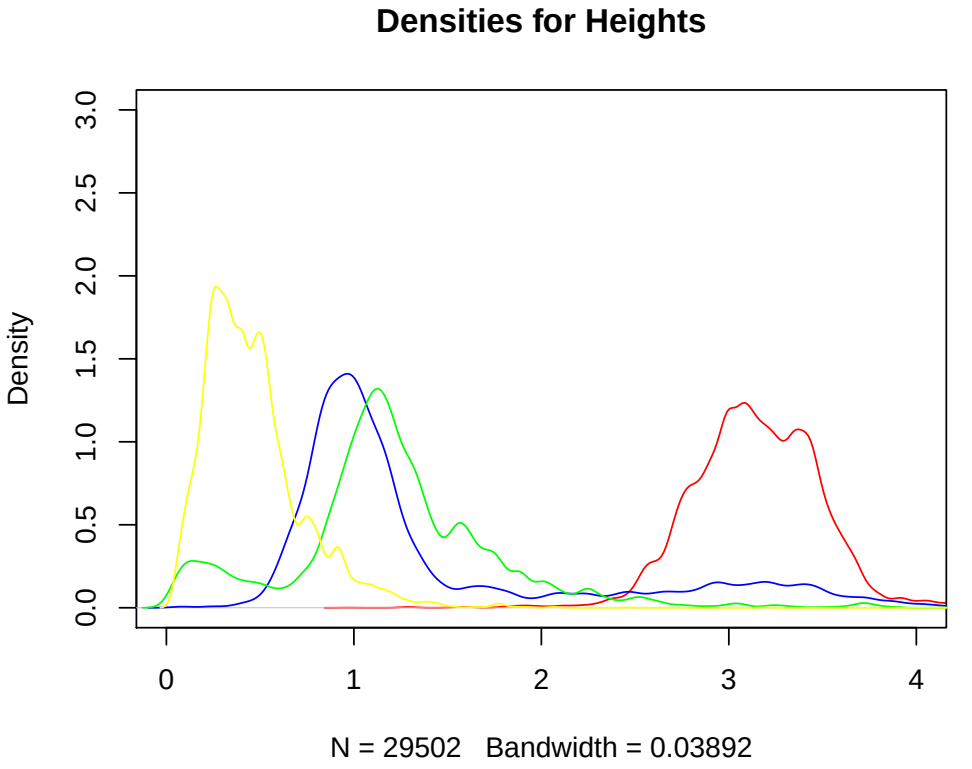


Figure 5.20: The paths and distribution of heights for $100,000$ *steps of Markov chain conditioned on* $k = 4$*. Red:*h_1 *Bule:*h_2 *Green:*h_3 *Yellow:*h_4*.*

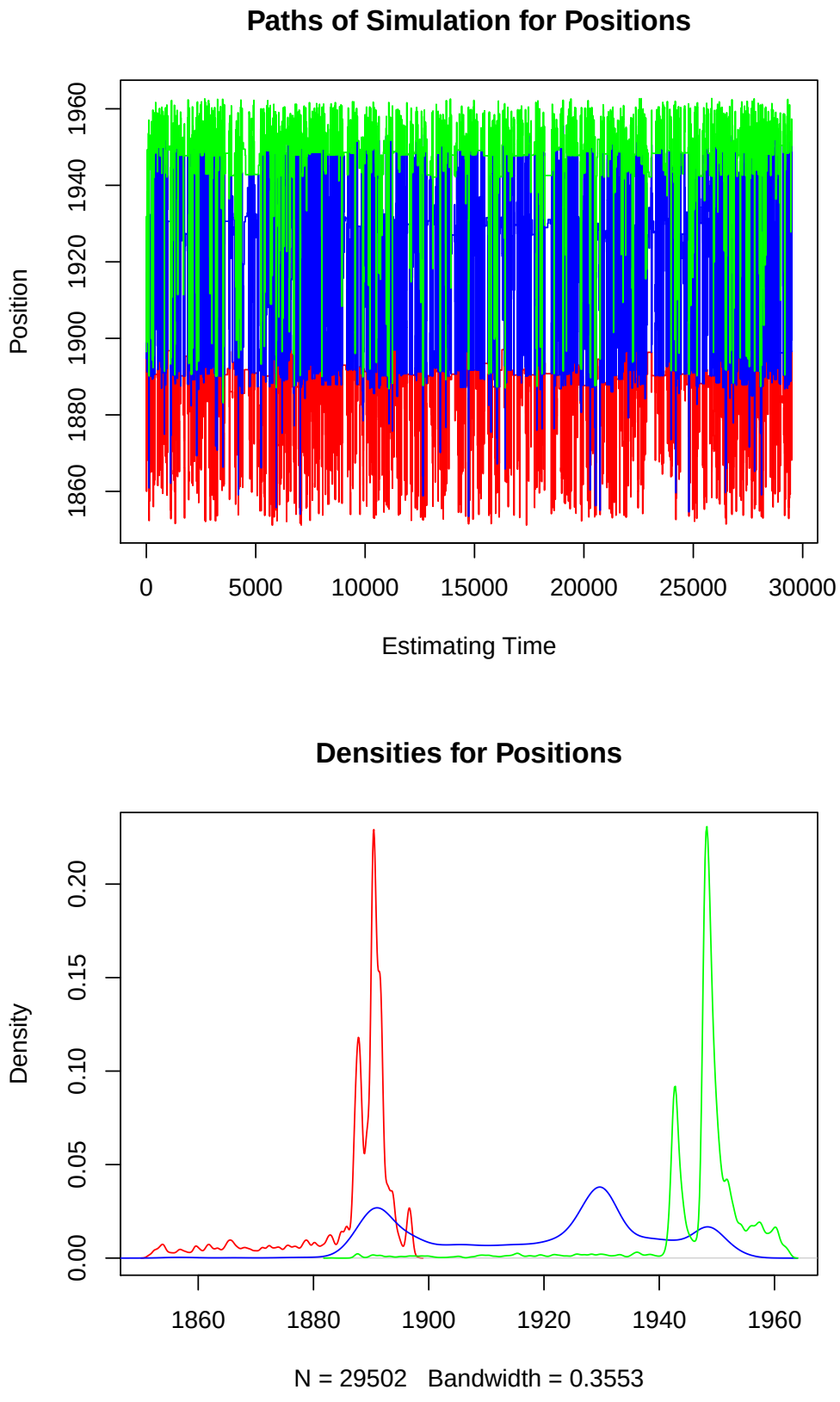
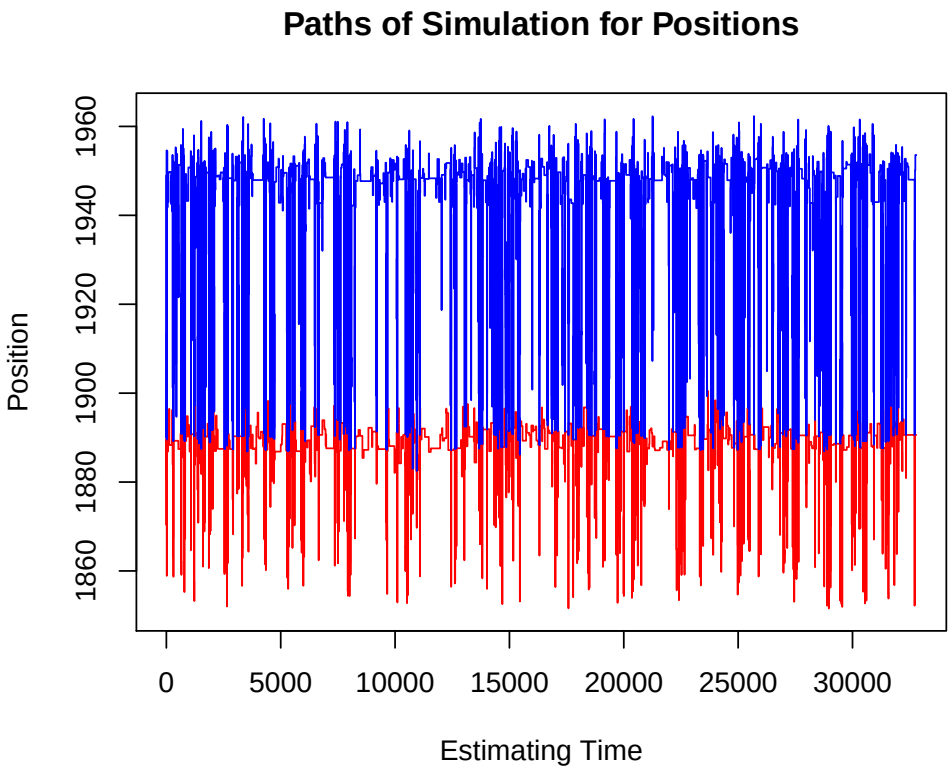


Figure 5.21: The paths and distribution of positions for $100,000$ *steps of Markov chain conditioned on* $k = 4$*. Red:*s_2 *Bule:*s_3 *Green:*s_4*.*



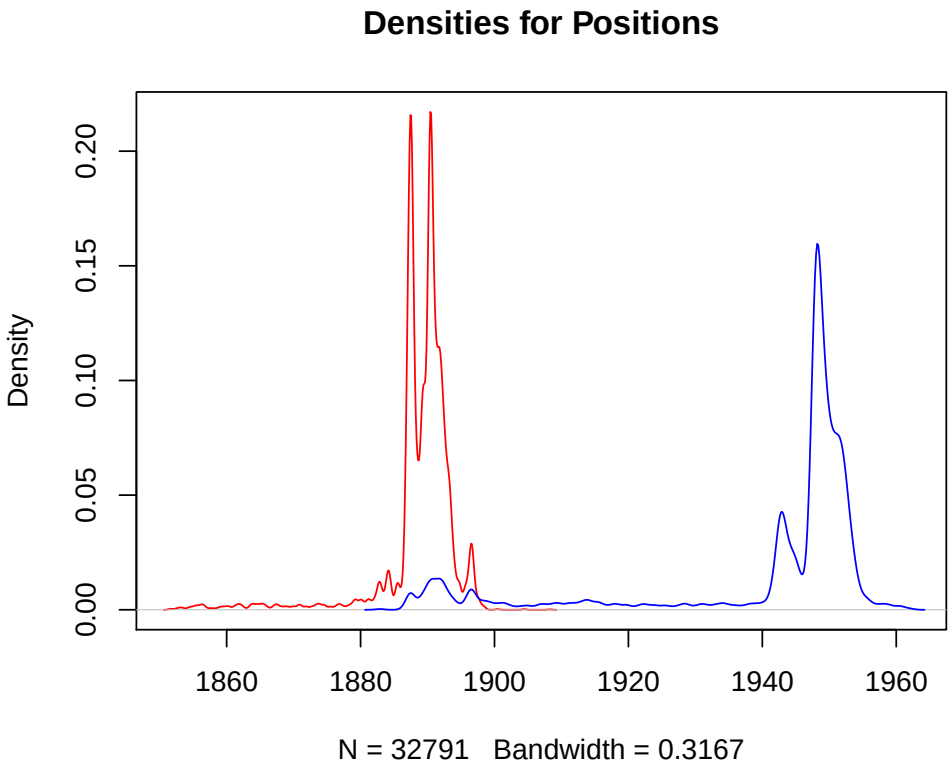


Figure 5.22: The paths and distribution of positions for $100,000$ *steps of Markov chain conditioned on* $k = 3$*. Red:*s_2 *Bule:*s_3*.*

Chapter 6

Conclusion

Reversible-jump Markov chain Monte Carlo extends the range of models selection. Original MCMC can only be used in fixed dimension model estimates. But reversible-jump MCMC make Markov chain jump among different model subspaces with different number of parameters.

With this advantage, reversible-jump MCMC can "compare" the models in different dimensionality. A plausible model will arise from a reversible-jump Markov chain generated by the Metropolis-Hastings algorithm Bayesian estimates.

As we discussed in this article, reversible-jump MCMC can be applied to solve the change-points problem and determine a model with flexible number of parameters.

Many practical applications benefit from this method, such as DNA copy number changes on the genome, change-points determining in genes, number of variables selection in regression etc. Thus, reversible-jump Markov chain Monte Carlo method is really an important topic deserve to be digged for..

Bibliography

"boot" package information source: website
http://astrostatistics.psu.edu/su07/R/html/boot/html/coal.html,
accessed 24 August 2012.

Charlie J. Geyer, (2010), Introduction to Markov Chain Monte Carlo, *Handbook of Markov Chain Monte Carlo: Methods and Applications*, Taylor and Francis Group, LLC, pp.3-48, https://www.psych.umn.edu/faculty/waller/classes/Bayes/Readings/HandbookChapter1.pdf, accessed 18 August 2012.

Christian P. Robert & George Casella, (2004), *Monte Carlo Statistical Methods, 2nd edition*, New York: Springer.

Jan Grandell, (1997), *Mixed Poisson Process*, London: Chapman & Hall.

Peter Green, (1995), Reversible jump Markov chain Monte Carlo computation and Bayesian model determination, *Biometrika* **82**, 4, pp. 711-732.

Anthony W. Knapp, (2005), *Basic Real Analysis*, Birkhäuser.

Home Website of R software ©1998-2012 by Kurt Hornik
http://www.r-project.org/.

Steven Roman, (2005), *Advanced Linear Algebra, 2nd edition*, California: Springer.

Jochen Voss, (2011), MATH5835-Statistical Computation Lecture Notes, University of Leeds.

Wikipedia, Bayesian inference, http://en.wikipedia.org/wiki/Bayesian_inference, accessed 19 August 2012.

APPENDIX

I put the estimate results of the models with the same dimensionality into one matrix. So that I separate several data groups with $k = 2$, $k = 3$, $k = 4$ etc. using the command followed.

```
T2<-list()
T3<-list()
T4<-list()
R2<-list()
R3<-list()
R4<-list()
j2<-0
j3<-0
j4<-0
for (i in 1:length(K)){
if (K[i]==3){
    j3=j3+1
    T3[j3]<-X$h[i]
        R3[j3]<-X$s[i]
        }
else if (K[i]==4){
    j4=j4+1
    T4[j4]<-X$h[i]
        R4[j4]<-X$s[i]
        }
else if (K[i]==2){
    j2=j2+1
    T2[j2]<-X$h[i]
        R2[j2]<-X$s[i]
        }
}
H2 <- matrix(unlist(T2), ncol=2, byrow=TRUE)
H3 <- matrix(unlist(T3), ncol=3, byrow=TRUE)
H4 <- matrix(unlist(T4), ncol=4, byrow=TRUE)
S2 <- matrix(unlist(R2), ncol=3, byrow=TRUE)
S3 <- matrix(unlist(R3), ncol=4, byrow=TRUE)
S4 <- matrix(unlist(R4), ncol=5, byrow=TRUE)
```