EXPLORING COMPLEX LOSS FUNCTIONS FOR POINT ESTIMATION

Kuntalee Chaisee
A thesis submitted for the degree of Doctor of Philosophy

University of Bath
Department of Mathematical Sciences

August 2015

COPYRIGHT

Attention is drawn to the fact that copyright of this thesis rests with its author. This copy of the thesis has been supplied on condition that anyone who consults it is understood to recognise that its copyright rests with its author and no information derived from it may be published without the prior written consent of the author.

This thesis may be made available for consultation within the University library and may be photocopied or lent to other libraries for the purposes of consultation.

Signature of Author...........................................................................................................

Kuntalee Chaisee
Summary

This thesis presents several aspects of simulation-based point estimation in the context of Bayesian decision theory. The first part of the thesis (Chapters 4 - 5) concerns the estimation-then-minimisation (ETM) method as an efficient computational approach to compute simulation-based Bayes estimates. We are interested in applying the ETM method to compute Bayes estimates under some non-standard loss functions. However, for some loss functions, the ETM method cannot be implemented straightforwardly. We examine the ETM method via Taylor approximations and cubic spline interpolations for Bayes estimates in one dimension. In two dimensions, we implement the ETM method via bicubic interpolation.

The second part of the thesis (Chapter 6) concentrates on the analysis of a mixture posterior distribution with a known number of components using the Markov chain Monte Carlo (MCMC) output. We aim for Bayesian point estimation related to a label invariant loss function which allows us to estimate the parameters in the mixture posterior distribution without dealing with label switching. We also investigate uncertainty of the point estimates which is presented by the uncertainty bound and the crude uncertainty bound of the expected loss evaluated at the point estimates based on MCMC samples. The crude uncertainty bound is relatively cheap, but it seems to be unreliable. On the other hand, the uncertainty bound which is approximated a 95% confidence interval seems to be reliable, but are very computationally expensive.

The third part of the thesis (Chapter 7), we propose a possible alternative way to present the uncertainty for Bayesian point estimates. We adopt the idea of leaving out observations from the jackknife method to compute jackknife-Bayes estimates. We then use the jackknife-Bayes estimates to visualise the uncertainty of Bayes estimates. Further investigation is required to improve the method and some suggestions are made to maximise the efficiency of this approach.
Acknowledgements

First and foremost, I would like to thank my supervisor, Dr Merrilee Hurn for giving me invaluable advice and providing support and encouragement throughout my PhD years. It has been a great experience to work under her supervision.

Further, I would like to thank the Department of Mathematical Sciences, University of Bath for their great support and the Royal Thai Government for their financial support.

Finally, I would like to say a big thank you to my beloved parents and sister for their encouragement and great advice whenever I have difficult time and for believing in me.
Contents

List of Figures ix

List of Tables xiv

1 Thesis Overview 1

1.1 Introduction ................................................................. 1

1.1.1 Bayesian statistics ...................................................... 2

1.1.2 The Bayes estimate .................................................... 2

1.1.3 Simulation methods .................................................... 3

1.2 The aim of the thesis ...................................................... 4

1.3 The thesis organisation ................................................... 6

2 Background 7

2.1 Introduction ................................................................. 7

2.2 Bayesian point estimation ................................................ 8

2.3 Monte Carlo methods .................................................... 16
2.3.1 Classical Monte Carlo integration .......................... 16
2.3.2 Importance sampling ........................................ 17

2.4 Markov chains ................................................... 20
2.4.1 Discrete state spaces ....................................... 21
2.4.2 Continuous state spaces ................................... 24

2.5 Markov chain Monte Carlo (MCMC) ......................... 24
2.5.1 The Metropolis-Hastings algorithm ....................... 25
2.5.2 Gibbs sampling .............................................. 26
2.5.3 How the Metropolis-Hastings algorithm works .......... 28
2.5.4 Gibbs sampling is a special case of Metropolis-Hastings sampling 30
2.5.5 Remark on algorithms .................................... 32

2.6 Dealing with MCMC samples ................................. 32
2.6.1 Convergence of the MCMC output ....................... 33
2.6.2 Burn-in period .............................................. 33
2.6.3 Effective Sample Size ..................................... 34

3 Optimisation Methods ............................................ 37

3.1 Introduction .................................................... 37
3.2 Deterministic optimisation methods ......................... 38
3.2.1 The Nelder-Mead method ................................ 38
3.2.2 Newton’s method ........................................... 39
4.3.4 The ETM method ........................................ 88
4.3.5 Cubic spline approximation for the linex loss function .... 92
4.3.6 Cubic spline approximation for the quotient loss function ... 95
4.3.7 Discussion and conclusion ............................. 98

5 Bicubic Interpolation for Bayes Estimates in Two Dimensions 101

5.1 Introduction ............................................. 101
5.2 Bicubic interpolation .................................... 103
5.3 Bicubic interpolation to approximate loss functions .......... 108
5.4 How to find a suitable set of grid points ................... 110
5.5 The ETM method via bicubic interpolation .................. 115
5.6 The implementation of the ETM method .................... 117
5.7 Bicubic interpolation for the linex loss function ............ 120
5.8 Bicubic interpolation for the quotient loss function ......... 124
5.9 Discussion and conclusion ................................ 126

6 Point Estimation in Bayesian Mixture Modelling 129

6.1 Introduction ............................................. 129
6.2 Mixture posterior distributions ............................ 131
6.2.1 Conditional posterior distributions and MCMC algorithms ... 133
6.3 The study of prior sensitivity and simulation results of the MCMC samples 139
6.4 Inference for the mixture posterior distribution ............. 153
List of Figures

2.1 Plot of the quadratic loss function ........................................... 11
2.2 Plot of the absolute loss function ........................................... 12
2.3 Plot of the $0 - 1$ loss function ........................................... 15

3.1 Plots of the function $f$ using $\sigma^2 = 0.1$ (top left), $\sigma^2 = 0.25$ (top right), $\sigma^2 = 0.5$ (bottom left) and $\sigma^2 = 1$ (bottom right) ................................................... 49

4.1 Plot of the linex loss function ........................................... 65
4.2 Plot of the linex loss function and Taylor series approximations, $T_2 = T_{2,\theta}(\theta), T_4 = T_{4,\theta}(\theta), T_7 = T_{7,\theta}(\theta) \text{ and } T_9 = T_{9,\theta}(\theta)$. ........................................... 66
4.3 Plot of the Taylor series approximations, $T_7 = T_{7,\theta}(\theta), T_8 = T_{8,\theta}(\theta), T_{10} = T_{10,\theta}(\theta)$, and $T_{14} = T_{14,\theta}(\theta)$ (for the model N1). ........................................... 70
4.4 Plot of the Taylor series approximations, $T_7 = T_{7,\theta}(\theta), T_8 = T_{8,\theta}(\theta), T_{10} = T_{10,\theta}(\theta)$ and $T_{14} = T_{14,\theta}(\theta)$ (for the model G). ........................................... 73
4.5 Plot of the quotient loss function ........................................... 75
4.6 Plot of the quotient loss function and the Taylor series approximations, $T_2 = T_{2,\theta}(\theta), T_4 = T_{4,\theta}(\theta), T_7 = T_{7,\theta}(\theta)$ and $T_9 = T_{9,\theta}(\theta)$. ........................................... 77
4.7 Plot of the linex loss function and cubic spline approximation using 6 knots where $s_1 = s_{1,\theta}(\theta), \ldots, s_5 = s_{5,\theta}(\theta)$. ........................................... 85
4.8 Plot of the linear loss function and cubic spline approximation using 8 knots where $s_1 = s_{1,\theta}(\theta), \ldots, s_7 = s_{7,\theta}(\theta)$. 

4.9 Plot of the quotient loss function and cubic spline approximation using 5 knots where $s_1 = s_{1,\theta}(\theta), \ldots, s_4 = s_{4,\theta}(\theta)$.

4.10 Plot of the quotient loss function and cubic spline approximation using 9 knots where $s_1 = s_{1,\theta}(\theta), \ldots, s_8 = s_{8,\theta}(\theta)$.

5.1 The grid square from grid points $\{\theta_1, 1, \theta_1, 2, \theta_1, 3\}$ on $\theta_1$-axis and $\{\theta_2, 1, \theta_2, 2, \theta_2, 3\}$ on $\theta_2$-axis.

5.2 A point $(\theta_1, \theta_2)$ in a rectangle $[\theta_1, \theta_1 + 1] \times [\theta_2, \theta_2 + 1]$ and a point $(u, v)$ in the unit square $[0, 1] \times [0, 1]$.

5.3 Consider the rectangle $[\theta_1, 2, \theta_1, 3] \times [\theta_2, 2, \theta_2, 3]$.

5.4 Add the midpoint on $\theta_1$-axis and use four corner points on the green rectangle for bicubic interpolation.

5.5 Add the midpoint on $\theta_2$-axis and use four corner points on the green rectangle for bicubic interpolation.

5.6 Repeat adding the midpoint on $\theta_1$-axis and use the green rectangle for bicubic interpolation.

5.7 Repeat adding the midpoint on $\theta_2$-axis after adding the midpoint on $\theta_1$-axis and use the green rectangle for bicubic interpolation.

5.8 Obtain the smallest rectangle which has dimension $d_1 \times d_2$ related to the midpoints on $[\theta_1, 2, \theta_1, 3]$ and $[\theta_2, 2, \theta_2, 3]$.

5.9 Add all the points on the $\theta_1$-axis and $\theta_2$-axis of the rectangle $[\theta_1, 1, \theta_1, 3] \times [\theta_2, 1, \theta_2, 3]$.

5.10 Consider the other rectangles: $[\theta_1, 1, \theta_1 + 1] \times [\theta_2, 1, \theta_2 + 1]$ for $i = 1, 3$ and 4 to obtain all grid points.

5.11 The quantile grid points and the generated samples $(\theta_1^{(n)}, \theta_2^{(n)})$ (red crosses).
5.12 The empty rectangles after adding grid points between the quantile grid points.
5.13 The dummy values to represent the generated sample in the empty rectangles.
6.1 Histogram of the galaxy data.
6.2 Trace plots of 100,000 iterations of the MCMC samples using the non-informative priors.
6.3 The probability density functions of the Inverse-Gamma distributions with different values of the shape and scale parameters.
6.4 Histograms of the three MCMC samples of $\omega$ (in 3 columns) from using the priors in (6.17) with $\sigma^2_j \sim \text{InvGam}(1,1)$ (first row), $\sigma^2_j \sim \text{InvGam}(3,1)$ (second row), $\sigma^2_j \sim \text{InvGam}(4,0.5)$ (third row) and $\sigma^2_j \sim \text{InvGam}(5,0.1)$ (fourth row).
6.5 Histograms of the three MCMC samples of $\mu$ (in 3 columns) from using the priors in (6.17) with $\sigma^2_j \sim \text{InvGam}(1,1)$ (first row), $\sigma^2_j \sim \text{InvGam}(3,1)$ (second row), $\sigma^2_j \sim \text{InvGam}(4,0.5)$ (third row) and $\sigma^2_j \sim \text{InvGam}(5,0.1)$ (fourth row).
6.6 Histograms of the three MCMC samples of $\sigma^2$ (in 3 columns) from using the priors in (6.17) with $\sigma^2_j \sim \text{InvGam}(1,1)$ (first row), $\sigma^2_j \sim \text{InvGam}(3,1)$ (second row), $\sigma^2_j \sim \text{InvGam}(4,0.5)$ (third row) and $\sigma^2_j \sim \text{InvGam}(5,0.1)$ (fourth row).
6.7 Histograms of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma^2_j$ for $j = 1, \ldots, 3$.
6.8 Trace plots of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma^2_j$ for $j = 1, \ldots, 3$.
6.9 Histograms of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma^2_j$ for $j = 1, \ldots, 4$.
6.10 Trace plots of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma^2_j$ for $j = 1, \ldots, 4$. 
6.11 Plot of the geometric annealing (cooling) schedule. .......................... 155

6.12 The approximated mixture densities with the component $k = 3$ (top) and $k = 4$ (bottom) evaluated at the point estimate $\hat{\xi}$. .......................... 158

6.13 The expected loss values based on $N = 50,000$ MCMC samples evaluated at each iteration from the SA algorithm of the mixture model with the components $k = 3$ (top) and $k = 4$ (bottom). .......................... 162

6.14 The uncertainty bounds and the crude uncertainty bounds of the expected loss function based on $N = 50,000$ MCMC samples, evaluated at the $j$th iteration of the SA algorithm for the mixture model with the component $k = 3$. .......................... 168

6.15 The uncertainty bounds and the crude uncertainty bounds of the expected loss function based on $N = 50,000$ MCMC samples, evaluated at the $j$th iteration of the SA algorithm for the mixture model with the component $k = 4$. .......................... 169

7.1 Histogram of the simulated data .......................... 181

7.2 The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x(i)$ for $i = 1, \ldots, 82$ of the parameter $\mu$ where the vertical red lines are the expected values of the parameter: three components (above) and component 1 (bottom). .......................... 183

7.3 The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x(i)$ for $i = 1, \ldots, 82$ of the parameter $\mu$ where the vertical red lines are the expected values of the parameter: component 2 (above) and component 3 (bottom). 184

7.4 The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x(i)$ for $i = 1, \ldots, 82$ of the parameter $\sigma^2$ where the vertical red lines are the expected values of the parameter: three components (above) and component 1 (bottom). .......................... 185
7.5 The Bayes estimates (solid green dots and solid blue dots represent component 2 and component 3, respectively) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values \( x_{(i)} \) for \( i = 1, \ldots, 82 \) of the parameter \( \sigma^2 \) where the vertical red lines are the expected values of the parameter.

7.6 The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values \( x_{(i)} \) for \( i = 1, \ldots, 82 \) of the parameter \( \omega \) where the vertical red lines are the expected values of the parameter: three components (above) and component 1 (bottom).

7.7 The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values \( x_{(i)} \) for \( i = 1, \ldots, 82 \) of the parameter \( \omega \) where the vertical red lines are the expected values of the parameter: component 2 (above) and component 3 (bottom).
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The comparison of the used time of 100 iterations of the SA method to other deterministic methods while searching for maximum values of the function $f$ with $\sigma^2 = 0.25$.</td>
<td>50</td>
</tr>
<tr>
<td>3.2</td>
<td>Numerical results of the maximisation problem using $\sigma^2 = 0.1$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n+2)$, $i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.</td>
<td>53</td>
</tr>
<tr>
<td>3.3</td>
<td>Numerical results of the maximisation problem using $\sigma^2 = 0.25$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n+2)$, $i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.</td>
<td>54</td>
</tr>
<tr>
<td>3.4</td>
<td>Numerical results of the maximisation problem using $\sigma^2 = 0.5$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n+2)$, $i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.</td>
<td>55</td>
</tr>
<tr>
<td>3.5</td>
<td>Numerical results of the maximisation problem using $\sigma^2 = 1$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n+2)$, $i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.</td>
<td>56</td>
</tr>
<tr>
<td>4.1</td>
<td>The comparison of the linex Bayes estimates $\hat{\theta}^{LN}$ and $\hat{\theta}^{LT_n}$ with respect to Model N1 with the sample size $N = 100,000$.</td>
<td>69</td>
</tr>
<tr>
<td>4.2</td>
<td>The comparison of the linex Bayes estimates $\hat{\theta}^{LN}$ and $\hat{\theta}^{LT_n}$ with respect to Model G with the sample size $N = 100,000$.</td>
<td>72</td>
</tr>
<tr>
<td>4.3</td>
<td>The comparison of the used time (seconds) to calculate the linex Bayes estimates $\hat{\theta}^{LN}$ and $\hat{\theta}^{LT_n}$.</td>
<td>74</td>
</tr>
</tbody>
</table>
4.4 The comparison of the linex Bayes estimates $\hat{\theta}_{LN}$ and $\hat{\theta}_{LT}$, and the times based on a generated sample of $\theta$ from Model N1 with the sample size $N$. ................................. 74

4.5 The comparison of the quotient Bayes estimates $\hat{\theta}_{QN}$ and $\hat{\theta}_{QT}$ with respect to Model N1 with the sample size $N = 100,000$. ................................. 78

4.6 The comparison of the quotient Bayes estimates $\hat{\theta}_{LN}$ and $\hat{\theta}_{LT}$ with respect to Model G with the sample size $N = 100,000$. ................................. 79

4.7 The comparison of the linex Bayes estimates $\hat{\theta}_{LN}$ and $\hat{\theta}_{LS_n}$ with respect to Model N1 with the sample size $N = 100,000$. ................................. 93

4.8 The comparison of the linex Bayes estimates $\hat{\theta}_{LN}$ and $\hat{\theta}_{LS_n}$ with respect to Model G with the sample size $N = 100,000$. ................................. 93

4.9 The comparison of the used time (seconds) to calculate the linex Bayes estimates $\hat{\theta}_{LN}$ and $\hat{\theta}_{LS_n}$ with respect to Model N1. ................................. 93

4.10 The comparison of linex Bayes estimates and the used time based on a generated sample of $\theta$ from Model N1 with the sample size $N$. ................................. 94

4.11 The comparison of the quotient Bayes estimates $\hat{\theta}_{QN}$ and $\hat{\theta}_{QS_n}$ with respect to Model N1 with the sample size $N = 100,000$. ................................. 97

4.12 The comparison of the quotient Bayes estimates $\hat{\theta}_{QN}$ and $\hat{\theta}_{QS_n}$ with respect to Model G with the sample size $N = 100,000$. ................................. 97

4.13 The comparison of the used time (seconds) to calculate the quotient Bayes estimates $\hat{\theta}_{QN}$ and $\hat{\theta}_{QS_n}$ with respect to Model N1. ................................. 97

4.14 The comparison of the quotient Bayes estimates and the used time based on a generated sample of $\theta$ from Model N1 with the sample size $N$. ................................. 98

5.1 The linex Bayes estimates $(\hat{\theta}_1, \hat{\theta}_2)^{LN}$ and $(\hat{\theta}_1, \hat{\theta}_2)^{LP}$ with respect to Model N2 with the sample size $N = 100,000$. ................................. 122

5.2 The comparison of the used time (seconds) to calculate the linex Bayes estimates $(\hat{\theta}_1, \hat{\theta}_2)^{LN}$ and $(\hat{\theta}_1, \hat{\theta}_2)^{LP}$ with respect to Model N2. ................................. 123
5.3 The comparison of linex Bayes estimates \((\hat{\theta}_1, \hat{\theta}_2)^{LP}\) which is obtained when \((n_1, n_2) = (13, 9)\) is used and the use time based on a generated sample of \((\theta_1, \theta_2)\) from Model N2 with the sample size \(N\). .............. 124

5.4 The quotient Bayes estimates \((\hat{\theta}_1, \hat{\theta}_2)^{QN}\) and \((\hat{\theta}_1, \hat{\theta}_2)^{QP}\) with respect to Model N2 with the sample size \(N = 100,000\). .............. 125

5.5 The comparison of the used time (seconds) to calculate the quotient Bayes estimates \((\hat{\theta}_1, \hat{\theta}_2)^{QN}\) and \((\hat{\theta}_1, \hat{\theta}_2)^{QP}\) with respect to Model N2. ... 125

5.6 The number of empty rectangles in bicubic interpolation for the linex loss function and the quotient loss function with the sample size \(N = 100,000\).126

5.7 The comparison of linex Bayes estimates \((\hat{\theta}_1, \hat{\theta}_2)^{QP}\) which is obtained when \((n_1, n_2) = (9, 6)\) is used and the use time based on a generated sample of \((\theta_1, \theta_2)\) from Model N2 with the sample size \(N\). .............. 126

6.1 Prior distributions for \(\sigma^2_j\) for \(j = 1, \ldots, k\) and the values of means and variances. .............. 143

6.2 The point estimates of the parameters of the Normal mixture with the components \(k = 3\) based on \(N\) MCMC samples. .............. 157

6.3 The point estimates of the parameters of the Normal mixture with the components \(k = 4\) based on \(N\) MCMC samples. .............. 157

6.4 The sample means and the credible intervals constructed by using the 2.5% and 97.5% quantiles of the simulated samples with \(N = 10,000\) for \(k = 3\). .............. 160

6.5 The approximated 95% confidence interval and the crude bound evaluated at \(\hat{\xi}^*\) based on MCMC with \(N = 10,000\) for \(k = 3\). .............. 166

6.6 The widths of the credible intervals and the two uncertainty bounds; the approximated 95% confidence interval and the crude bound evaluated at \(\hat{\xi}^*\) based on MCMC with \(N = 10,000\) for \(k = 3\). .............. 167
7.1 The first quantiles (Q1), medians, means and the third quartiles (Q3) of the Bayes estimates obtained from 500 simulations using 500 simulated datasets.
Chapter 1

Thesis Overview

1.1 Introduction

This thesis on “Exploring complex loss functions for point estimation” is motivated by the study of a label invariant loss function which was suggested in Celeux et al. (2000) for point estimation of parameters in a mixture posterior distribution without dealing with label switching. Normally, computing point estimates related to unusual loss functions such as the label invariant loss function cannot be done analytically and so simulation methods are implemented. Computing simulation-based point estimates could be very computationally expensive because it basically involves simulation methods such as Markov chain Monte Carlo (MCMC) methods to generate a sample for estimation of the expected loss function, and optimisation methods to search for the minimum (e.g. simulated annealing). We have found that by implementing an “estimation-then-minimisation (ETM)” approach, we could significantly reduce the computational cost. Therefore, we want to study the use of this method for some other non-standard loss functions where the ETM approach could not be implemented straightforwardly.

There will be three main topics of interest in this thesis. Firstly, we concentrate on the ETM method for point estimation under some non-standard loss functions. To implement the ETM method, the loss function must be expressed as a decomposed form. In this thesis, we express the loss function in the decomposed form by using the Taylor series approximation and the interpolation method. We want to show efficiency of the ETM method by showing that simulation-based point estimates from using the ETM method with (loss) function approximations are as good as ones from using a method with true loss functions, but that the ETM method is faster.
Secondly, we focus on point estimation of parameters and uncertainty assessment in a Bayesian mixture model using a label invariant loss function suggested in Celeux et al. (2000). We propose two uncertainty bounds to assess uncertainty of point estimates without dealing label switching. Thirdly, we propose the jackknife method to assess the uncertainty of point estimates. Overall, the thesis concerns Bayesian point estimation and the uncertainty of point estimates based on MCMC simulation. In next three subsections, we briefly describe some fundamental issues which are used in this thesis.

### 1.1.1 Bayesian statistics

In Bayesian statistics, the posterior distribution or simply the posterior for parameter $\theta$ combines two sources of information about $\theta$; the subjective prior belief about $\theta$ which is represented by the prior distribution or simply the prior, and information about $\theta$ contained in the data given a particular model which is represented by the likelihood function or simply the likelihood. The posterior distribution, $\pi(\theta | x)$ for parameter $\theta$ given observed data $x = (x_1, x_2, \ldots, x_n)$ is then formulated by using Bayes’ theorem

$$\pi(\theta | x) = \frac{l(x | \theta)p(\theta)}{\int_\Theta l(x | \theta)p(\theta)d\theta},$$

(1.1)

where $l(x | \theta)$ is the likelihood and $p(\theta)$ is the prior. The denominator can be thought of as a function of $x$ denoted by $m(x) = \int_\Theta l(x | \theta)p(\theta)d\theta$ which is also known as the normalising constant. Therefore, the posterior distribution is often written in the memorable form as the product of the likelihood and the prior,

$$\pi(\theta | x) \propto l(x | \theta) \times p(\theta).$$

(1.2)

### 1.1.2 The Bayes estimate

One of the common questions often asked in statistics is what is the best point estimate of the unknown parameter $\theta$ (or the parameter vector)? To answer this in a Bayesian framework, we need to specify a loss function, $L(\hat{\theta}, \theta)$ which represents the loss incurred by estimating $\theta$ with the estimate $\hat{\theta}$ in the parameter space $\Theta$. The best point estimate would be $\hat{\theta} = \theta$, however the true value of $\theta$ is unknown. Therefore, the best point estimate will be to choose the value of $\hat{\theta}$ which minimises the expected loss function with respect to the posterior distribution, $\pi(\theta | x)$, and is known as the Bayes estimate. The Bayes estimate can be derived analytically under some standard loss functions. For example, under the quadratic loss function, the Bayes estimate is the posterior mean (see Chapter 2, Example 2.1). Nevertheless, some standard loss functions for which
analytical Bayes estimates are available might not be suitable in every problem. In practice, the loss function is specified by the decision maker and is usually complex so it is difficult or even impossible to derive the Bayes estimate related to the loss function analytically. This necessitates an approximation of the Bayes estimate by numerical methods or by simulation.

1.1.3 Simulation methods

The thesis concerns Bayes estimates obtained by using simulation methods. We use a simulation technique to estimate expectation and then a minimisation method to deal with the minimisation problem to obtain Bayes estimates. The simulation method which will be used for this problem is the Monte Carlo method and more specifically, Markov chain Monte Carlo (MCMC) methods. The Monte Carlo method can be loosely described as a statistical simulation-based method where we monitor a quantity of interest of a distribution from simulated samples from the distribution. Credit for inventing the Monte Carlo method often goes to Stanislaw Ulam who worked for John von Neumann on the United States Manhattan Project during World War II. He claimed to be stimulated by playing poker and his uncle once borrowed money from him to go gambling in Monte Carlo. That is why the method is called Monte Carlo to refer to the city of Monte Carlo where lots of gambling goes on, Johansen et al. (2007), Richey (2010). John von Neumann and Stanislaw Ulam suggested the simulation method to investigate properties of neutron travel through radiation shielding. They, along with others, used simulation for many other nuclear weapon problems and established most of the fundamental methods of Monte Carlo simulation, Harrison (2010). The Monte Carlo method is often known as Monte Carlo integration, that is using independent samples of a parameter \( \theta \) generated from the distribution (the posterior distribution, \( \pi(\theta|x) \) in equation (1.1)) to approximate integrals according to the Strong Law of Large Numbers.

However, generating a sample from a posterior distribution might be difficult, especially for higher-dimensional distributions of \( \theta \). The reason is often the complicated integration normalising constant \( m(x) \). Markov chain Monte Carlo (MCMC) methods can overcome this difficulty. MCMC simulation, introduced in the statistics literature by Hastings (1970), is a class of algorithms for sampling that constructs a Markov chain which (hopefully) converges to the distribution of interest according to the Markov chain principle. MCMC algorithms enable us to generate an approximate sample from any desired distribution, provided we can compute the value of a function which is proportional to the density of the desired distribution. Therefore, without knowing the normalising constant \( m(x) \), we can generate a sample of the parameter \( \theta \) from the
The posterior distribution $\pi(\theta|x)$ expressed in the proportional form (1.2) by using MCMC algorithms. The samples generated using the MCMC methods or simply the MCMC samples are not independent so we cannot rely on the Strong Law of Large Number applied in Monte Carlo integration to approximate integrals. Fortunately, the *Ergodic Theorem* makes approximation of integrals based on the MCMC sample feasible, Robert and Casella (2004), Section 6.7.1.

### 1.2 The aim of the thesis

In this thesis, we want to find an efficient and effective approach to compute Bayes estimates under some non-standard loss function. Typically, there are two computational tasks to calculate Bayes estimates: estimation and minimisation. The estimation part refers to estimation of the (posterior) expected loss function which often involves generating a (MCMC) sample of the parameter $\theta$ and evaluating integrals. The minimisation part refers to the search for $\hat{\theta}$ which yields the minimum of the expected loss function. The simulated annealing algorithm is often implemented to search for the Bayes estimates for non-standard loss function such as the invariant loss function used for a mixture posterior distribution. It is relatively expensive to perform. As a result, computing Bayes estimates is often a computationally expensive process. The computational cost can be reduced by implementing the approach called the ETM method. Why is the ETM method cheap (fast)? We illustrate how the ETM approach works by considering the quadratic loss function as follows.

Suppose we want to compute the Bayes estimate related to the quadratic loss function given by

$$L(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2.$$  

The expected loss function with respect to the posterior distribution is

$$E_\pi[L(\hat{\theta}, \theta)] = E_\pi[(\hat{\theta} - \theta)^2].$$  

(1.3)

The quadratic loss function can be written as a decomposed form

$$L(\hat{\theta}, \theta) = \hat{\theta}^2 - 2\hat{\theta}\theta + \theta^2,$$

hence

$$E_\pi[L(\hat{\theta}, \theta)] = \hat{\theta}^2 - 2\hat{\theta}E_\pi[\theta] + E_\pi[\theta^2].$$  

(1.4)
We then apply a minimisation method to the expected loss function with respect to $\hat{\theta}$. The two expressions (1.3) and (1.4) of the expected loss function make a difference in the minimisation stage. The expression (1.3) requires the estimation of $E_{\pi}[(\hat{\theta} - \theta)^2]$ for every candidate value of $\hat{\theta}$. The expression (1.4), on the contrary, requires the estimation of $E_{\pi}[\theta]$ and $E_{\pi}[\theta^2]$ once for any candidate value of $\hat{\theta}$. We have to do estimation and minimisation simultaneously if we follow expression (1.3) but only do estimation once and then do minimisation if we follow expression (1.4). Certainly, doing estimation once then doing minimisation is a cheaper implementation than doing estimation and minimisation simultaneously many times. The implementation following the decomposed form (1.4) is denoted as the ETM method.

Unlike with the quadratic loss function, the ETM method cannot always be implemented directly. We want to study further how to implement the ETM method to compute Bayes estimates under some non-standard loss functions. For those loss functions, we need to transform them into a suitable form that allows us to separate the estimation parts and the minimisation parts from each other. To do so, we could approximate loss functions by using polynomial approximations. In this thesis, we want to investigate the ETM method via two function approximations; the Taylor series approximation and the cubic spline interpolation. Furthermore, we extend the use of the ETM method to compute Bayes estimates for loss functions defined in two dimensions. In two dimensions, we want to examine the ETM method via bicubic interpolation.

We also implement the ETM method in a label invariant loss function which is used for point estimation in Bayesian mixture modelling. In Bayesian mixture modelling, we might not be able to use posterior means to estimate parameters of interest because of the label switching problem in MCMC samples. In this thesis, we not only use the ETM method to compute simulation-based Bayes estimates related to a label invariant loss function for mixture modelling but also assess the uncertainty of these point estimates. Furthermore, we could use the procedure from the ETM method to assess the uncertainty of the point estimates using the crude uncertainty bound of the expected loss evaluated at the point estimates based on MCMC samples. Finally, we propose the idea of the jackknife method to assess the uncertainty without dealing with label switching. The jackknife method is usually used for estimating the bias and variance of a statistic of interest, Miller (1974), however in this thesis, we adopt the idea of deleting observations to compute the “jackknife-Bayes estimates” and want to use them to visualise the uncertainty of the Bayes estimates for the mixture posterior distribution. We also reduce the computational cost of computing the jackknife-Bayes estimates by implementing importance sampling. This jackknife approach is more generally applicable to any estimate generated as a computational approximation to a Bayes estimate.
1.3 The thesis organisation

The thesis will be organised with the following chapters. Chapter 2 is a review of Bayesian statistics for point estimation and simulation techniques. In this chapter, we provide key concepts of some simulation techniques which could be used to compute Bayes estimates. In Section 2.3, we give the basic description of Monte Carlo methods which could be used to estimate the expected loss function according to the Strong Law of Large Numbers. In Section 2.4, we highlight some properties of a Markov chain which are useful for understanding the Markov chain Monte Carlo (MCMC) methods provided in Section 2.5. Chapter 3 contains the notion of the simulated annealing method. In Section 3.4, we examine the advantage of simulated annealing against some deterministic optimisation methods. Chapter 4 covers the implementation of the ETM method for some non-standard loss functions in one dimensional parameter space. We show the ETM method via the Taylor series approximation in Section 4.2 and via the cubic spline interpolation in Section 4.3. Chapter 5 is an extension of the ETM method on the two dimensional parameter space via bicubic interpolation. Chapter 6 contains the study of Bayes estimates of a mixture posterior distribution. We show how to generate MCMC samples for a mixture posterior in Section 6.2 and discuss simulation results of Bayes estimates under the label invariant loss function in Section 6.4. We also describe how to approximate uncertainty bounds to assess the uncertainty of the Bayes estimates for a mixture model in Section 6.5. Chapter 7 presents the use of the jackknife method to assess uncertainty of the estimated parameters. Basic concepts of the jackknife method are described in Section 7.2. We then present an example of the uncertainty provided by using the idea of jackknifing in Section 7.4. In Section 7.5, we show how the method works using simulation study. Finally, Chapter 8 contains thesis conclusions and future work.
Chapter 2

Background

2.1 Introduction

This chapter covers fundamental knowledge of statistical methods to be used throughout the thesis. The main tool is simulation-based point estimation from a Bayesian perspective. We begin with Bayesian point estimation based on a decision theoretical approach using loss functions. There are some loss functions which are widely used because the point (Bayes) estimates related to them are analytically feasible and convenient to use. However, it might be impossible to derive the Bayes estimates under some complex loss functions analytically. The Bayes estimates under some non-standard loss functions can be computed by using simulation methods. Basically, simulation methods are based on random variables that should be independent and identically distributed (i.i.d) according to a desired distribution. We present the basic notion of Monte Carlo methods that use such generated random variables to approximate integrals. However, generating i.i.d. samples from some distributions, for example many posterior distributions, is sometimes very difficult. Markov chain Monte Carlo (MCMC) can provide a convenient way to draw samples from the posterior distribution. The term Markov chain refers to a stochastic process in which future states are independent of past states given the present state while the term Monte Carlo refers to simulation techniques used to approximate integrals. We provide enough foundation for understanding MCMC methods along with their application for point estimation related to a loss function. Before introducing MCMC methods, we briefly state some fundamental notions of Markov chains which are essential to establishing the convergence of MCMC algorithms. We also detail the two well known MCMC algorithms; the Metropolis-Hastings algorithm and the Gibbs sampler.
This chapter is organised as follows. In Section 2.2, we describe parameter estimation from a Bayesian perspective. In this section, we also provide some well known loss functions and their corresponding Bayes estimates. In Section 2.3, we introduce Monte Carlo methods which are broadly used for estimating expectation. Section 2.4 gives an overview of basic notions of Markov chains for understanding the Markov chain Monte Carlo (MCMC) methods described in Section 2.5. Finally, Section 2.6 contains some important characteristics of MCMC output for practical purposes.

2.2 Bayesian point estimation

In this section, we describe point estimation of parameter $\theta$ in the Bayesian framework related to statistical decision theory; see more in Berger (1985). From a decision-theoretic point of view, there are three spaces involved: $\mathcal{X}$, observation space (sample space), $\Theta$, parameter space, and $\mathcal{D}$, decision space (or action space) where the standard estimation setting is $\mathcal{D} = \Theta$. Then we take a decision $d \in \mathcal{D}$ related to the parameter $\theta \in \Theta$ based on the observed data $x \in \mathcal{X}$ where $x$ and $\theta$ are connected by the posterior distribution $\pi(\theta|x)$. No decision can be made without potential losses (errors), so statisticians or decision makers specify a loss function to represent the payoff consequent on the decision $d \in \mathcal{D}$. Therefore, a loss function is a function of a decision $d$ and the parameter, $\theta$ denoted by $L(d, \theta)$. It gives a measurement of how good or bad each particular decision is. Mathematically speaking, a loss function $L(d, \theta) = 0$ if and only if $d = \theta$. When $\theta$ is known, we can induce the best zero loss. This means that the best decision would be the true value of $\theta$ or $d = \theta$. For any decision $d \neq \theta$, the loss should be greater than or equal to 0, whether $d$ is over or under estimating $\theta$. Moreover, if $d_1$ is further away from $\theta$ than $d_2$ is, then it is more sensible to have loss functions such that $L(d_1, \theta)$ is greater than or equal to $L(d_2, \theta)$. It is generally impossible to minimise (in $d$) the loss function $L(d, \theta)$ when $\theta$ is unknown. According to the Bayes rule, the best decision will be defined to be the value of $d$ which minimises the expected loss, where expectation is taken across the distribution of $\theta$ after observing data $x$. In other words, the best decision is the value which minimises the expected loss function with respect to the posterior distribution, $\pi(\theta|x)$, Robert (2007), Section 4.2, and such a decision is then called the Bayes decision.

In this thesis, we focus on a single point estimate of the parameter $\theta$, so the decision will be made is $d = \hat{\theta}$ and a loss function is denoted by $L(\hat{\theta}, \theta)$. The point estimate $\hat{\theta}$ which minimises the expected loss function is then called the Bayes point estimate or simply the Bayes estimate of the parameter $\theta$. Therefore, the Bayes estimate is a
solution of
\[
\arg\min_{\hat{\theta}} \mathbb{E}_{\pi}[L(\hat{\theta}, \theta)],
\]
where
\[
\mathbb{E}_{\pi}[L(\hat{\theta}, \theta)] = \int_{\Theta} L(\hat{\theta}, \theta) \pi(\theta|x) d\theta.
\]
Depending on the complexity of the loss \( L \) and the posterior distribution \( \pi(\theta|x) \), the value of \( \hat{\theta} \) may be determined analytically or numerically. In general, it is difficult to determine the value of \( \hat{\theta} \) analytically because of either the complicated posterior distribution or the complex loss functions. Nevertheless, there are some loss functions for which the analytical Bayes estimates are feasible.

**Standard loss functions and Bayes estimates**

The following theorems and definitions will be used in the proofs of the Bayes estimates under standard loss functions.

**Theorem 2.1.** The first fundamental theorem of calculus, Spivak (2006).

Let \( f \) be integrable on \([a, b]\), and define \( F \) on \([a, b]\) by

\[
F(x) = \int_{a}^{x} f(t) dt.
\]

If \( f \) is continuous at \( c \) in \([a, b]\), then \( F \) is differentiable at \( c \), and

\[
F'(c) = f(c).
\]

**Theorem 2.2.** The second fundamental theorem of calculus, Spivak (2006).

If \( f \) is integrable on \([a, b]\) and \( f = g' \) for some function \( g \), then

\[
\int_{a}^{b} f(x) dx = g(b) - g(a).
\]

**Improper integrals**

1. If \( \int_{a}^{t} f(x) dx \) exists for every \( t > a \), then

\[
\int_{a}^{\infty} f(x) dx = \lim_{t \to \infty} \int_{a}^{t} f(x) dx
\]
provided the limit exists and is finite.

2. If \( \int_t^b f(x)dx \) exists for every \( t < b \), then

\[
\int_{-\infty}^b f(x)dx = \lim_{t \to -\infty} \int_t^b f(x)dx
\]

provided the limit exists and is finite.

3. If \( \int_c^t f(x)dx \) and \( \int_t^c f(x)dx \) are both convergent, then

\[
\int_{-\infty}^\infty f(x)dx = \int_c^c f(x)dx + \int_c^\infty f(x)dx,
\]

where \( c \) is any number.

The fundamental theorems and the improper integrals imply that

\[
\frac{d}{dy} \int_{-\infty}^y f(x)dx = f(y) \tag{2.2}
\]

\[
\frac{d}{dy} \int_y^\infty f(x)dx = -f(y) \tag{2.3}
\]

**Definition 2.1.** The **median** of the distribution of a random variable \( X \) is any number \( m \) such that

\[
\Pr\{X \geq m\} \geq \frac{1}{2} \quad \text{and} \quad \Pr\{X \leq m\} \geq \frac{1}{2}
\]

or, equivalently, the inequalities

\[
\int_{-\infty}^m dF(x) \geq \frac{1}{2} \quad \text{and} \quad \int_m^{\infty} dF(x) \geq \frac{1}{2}
\]

for any probability distribution on the real line \( \mathbb{R} \) with cumulative distribution function \( F \). If \( X \) is continuous, \( m \) satisfies

\[
\int_{-\infty}^m f(x)dx = \int_m^{\infty} f(x)dx = \frac{1}{2}
\]

Casella and Berger (1990).

**Definition 2.2.** The **Taylor series** of a real or complex-valued function \( f(x) \) that is infinitely differentiable at a real or complex number \( a \) is the power series

\[
f(a) + f'(a)(x - a) + \frac{f''(a)}{2!}(x - a)^2 + \frac{f'''(a)}{3!}(x - a)^3 + \ldots
\]
which can be written as
\[ \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!} (x - a)^n, \]

where \( f^{(n)}(a) \) denotes the \( n \)th derivative of \( f \) evaluated at the point \( a \). When \( a = 0 \), the series is called a Maclaurin series, Spivak (2006).

Example 2.1. The quadratic loss function
\[ L(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2 \]

The quadratic loss function

![Figure 2.1: Plot of the quadratic loss function](image)

The quadratic loss function penalises error more severely as the error increases that positive and negative differences are symmetric. Under the quadratic loss function, the Bayes estimate for \( \theta \) is the posterior mean.
Proof.

\[ \mathbb{E}_\pi[L(\hat{\theta}, \theta)] = \mathbb{E}_\pi[(\hat{\theta} - \theta)^2] = \mathbb{E}_\pi[\hat{\theta}^2 - 2\hat{\theta}\theta + \theta^2] = \hat{\theta}^2 - 2\hat{\theta}\mathbb{E}_\pi[\theta] + \mathbb{E}_\pi[\theta^2]. \]

Now, consider

\[ \text{Var}_\pi(\theta) = \mathbb{E}_\pi[\theta^2] - (\mathbb{E}_\pi[\theta])^2 \]

\[ \mathbb{E}_\pi[\theta^2] = (\mathbb{E}_\pi[\theta])^2 + \text{Var}_\pi(\theta). \]

Then

\[ \mathbb{E}_\pi[L(\hat{\theta}, \theta)] = \hat{\theta}^2 - 2\hat{\theta}\mathbb{E}_\pi[\theta] + (\mathbb{E}_\pi[\theta])^2 + \text{Var}_\pi(\theta) = (\hat{\theta} - \mathbb{E}_\pi[\theta])^2 + \text{Var}_\pi(\theta) \]

which is minimised when at \( \hat{\theta} = \mathbb{E}_\pi[\theta] \). \( \square \)

**Example 2.2.** The absolute loss function

\[ L(\hat{\theta}, \theta) = |\hat{\theta} - \theta| \]

**The absolute loss function**

![The absolute loss function](image)

Figure 2.2: Plot of the absolute loss function
The absolute loss function penalises error in a similar way to the quadratic loss function but the error does not increase as rapidly. Under the absolute loss function, the Bayes estimate for $\theta$ is the posterior median. To prove this, we restrict our proof to when the parameter space $\Theta$ is the real line, $\mathbb{R}$.

Proof.

\[
E_{\pi}[L(\hat{\theta}, \theta)] = E_{\pi}[|\hat{\theta} - \theta|]
\]
\[
= \int_{-\infty}^{\infty} |\hat{\theta} - \theta| \pi(\theta|x) d\theta
\]
\[
= \int_{-\infty}^{\hat{\theta}} (\hat{\theta} - \theta) \pi(\theta|x) d\theta + \int_{\hat{\theta}}^{\infty} (\theta - \hat{\theta}) \pi(\theta|x) d\theta
\]
\[
= \hat{\theta} \int_{-\infty}^{\hat{\theta}} \pi(\theta|x) d\theta - \int_{-\infty}^{\hat{\theta}} \theta \pi(\theta|x) d\theta
\]
\[
+ \int_{\hat{\theta}}^{\infty} \theta \pi(\theta|x) d\theta - \hat{\theta} \int_{\hat{\theta}}^{\infty} \pi(\theta|x) d\theta.
\]

Differentiate the expected loss function with respect to $\hat{\theta}$,

\[
\frac{d}{d\hat{\theta}} E_{\pi}[L(\hat{\theta}, \theta)] = \frac{d}{d\hat{\theta}} \left[ \hat{\theta} \int_{-\infty}^{\hat{\theta}} \pi(\theta|x) d\theta \right] - \frac{d}{d\hat{\theta}} \left[ \int_{-\infty}^{\hat{\theta}} \theta \pi(\theta|x) d\theta \right]
\]
\[
+ \frac{d}{d\hat{\theta}} \left[ \int_{\hat{\theta}}^{\infty} \theta \pi(\theta|x) d\theta \right] - \frac{d}{d\hat{\theta}} \left[ \hat{\theta} \int_{\hat{\theta}}^{\infty} \pi(\theta|x) d\theta \right]
\]
\[
= \hat{\theta} \pi(\hat{\theta}|x) + \int_{-\infty}^{\hat{\theta}} \pi(\theta|x) d\theta - \hat{\theta} \pi(\hat{\theta}|x) - \hat{\theta} \pi(\hat{\theta}|x)
\]
\[
+ \hat{\theta} \pi(\hat{\theta}|x) - \int_{\hat{\theta}}^{\infty} \pi(\theta|x) d\theta \quad \text{(by Equations (2.2) and (2.3))}
\]
\[
= \int_{-\infty}^{\hat{\theta}} \pi(\theta|x) d\theta - \int_{\hat{\theta}}^{\infty} \pi(\theta|x) d\theta.
\]

By using the usual relationship between the probability density functions and the cumulative distribution function $F$,

\[
F(\hat{\theta}|x) = \int_{-\infty}^{\hat{\theta}} \pi(\theta|x) d\theta,
\]

so we have

\[
\int_{\hat{\theta}}^{\infty} \pi(\theta|x) d\theta = \int_{-\infty}^{\infty} \pi(\theta|x) d\theta - \int_{-\infty}^{\hat{\theta}} \pi(\theta|x) d\theta = 1 - F(\hat{\theta}|x).
\]
It follows that

\[
\frac{d}{d\hat{\theta}} \mathbb{E}_\pi[L(\hat{\theta}, \theta)] = \int_{-\infty}^{\hat{\theta}} \pi(\theta|x) d\theta - \int_{\hat{\theta}}^{\infty} \pi(\theta|x) d\theta \\
= F(\hat{\theta}|x) - (1 - F(\hat{\theta}|x)) \\
= 2F(\hat{\theta}|x) - 1. \tag{2.4}
\]

Equating the derivative of the expected loss function in Equation (2.4) to zero and solving for \(\hat{\theta}\) to obtain either minimum or maximum,

\[
\frac{d}{d\hat{\theta}} \mathbb{E}_\pi[L(\hat{\theta}, \theta)] = 2F(\hat{\theta}|x) - 1 = 0 \\
F(\hat{\theta}|x) = \frac{1}{2}.
\]

From Definition 2.1, the value \(\hat{\theta}\) with half of the posterior probability less than it, and half greater than it, is the posterior median, \(m\). We verify that we have minimised the expected loss function by showing the second derivative of the expected loss function at the posterior median is greater than zero. We have

\[
\frac{d^2}{d\hat{\theta}^2} \mathbb{E}_\pi[L(\hat{\theta}, \theta)] = \frac{d}{d\hat{\theta}} [2F(\hat{\theta}|x) - 1] \\
= 2\pi(\hat{\theta}|x) \geq 0.
\]

Since \(\pi(\theta|x)\) is a probability density, it is non-negative everywhere including its median. Therefore, the best point estimate for \(\theta\) under the absolute loss function is the posterior median. \(\square\)

**Example 2.3.** The \(0 - 1\) loss function

Given \(\alpha > 0\),

\[
L(\hat{\theta}, \theta) = \begin{cases} 
1 & \text{if } |\hat{\theta} - \theta| > \alpha, \\
0 & \text{otherwise}.
\end{cases}
\]
The previous loss functions give \( L(\theta, \hat{\theta}) = 0 \) but for the 0−1 loss function continues to be zero for \( \hat{\theta} \) within \( \alpha \) of \( \theta \). It penalises all errors greater than \( \alpha \) equally with magnitude 1. Under the 0−1 loss function, the Bayes estimate for \( \theta \) is the posterior mode.

**Proof.** Consider the inequality \(|\hat{\theta} - \theta| > \alpha\). It can be written as two inequalities for \( \theta \) of the form \( \theta > \hat{\theta} + \alpha \) or \( \theta < \hat{\theta} - \alpha \). The expected loss function is then

\[
E_\pi[L(\hat{\theta}, \theta)] = \int_{-\infty}^{\hat{\theta} - \alpha} \pi(\theta|x)d\theta + \int_{\hat{\theta} + \alpha}^{\infty} \pi(\theta|x)d\theta = F(\hat{\theta} - \alpha|x) + 1 - F(\hat{\theta} + \alpha|x).
\]

From the Taylor series in Definition 2.2, the first two terms in a Taylor series approximation of \( f(y) \) at \( a \) is \( f(a) + f'(a)(y - a) \). Setting \( y = \hat{\theta} + \alpha, a = \hat{\theta} \) and \( f(y) = F(y|x) \).

\[
F(\hat{\theta} + \alpha|x) \approx F(\hat{\theta}|x) + \alpha F'(\hat{\theta}|x) = F(\hat{\theta}|x) + \alpha \pi(\hat{\theta}|x).
\]  

(2.5)

Similarly, setting \( y = \hat{\theta} - \alpha, a = \hat{\theta} \),

\[
F(\hat{\theta} - \alpha|x) \approx F(\hat{\theta}|x) - \alpha F'(\hat{\theta}|x) = F(\hat{\theta}|x) - \alpha \pi(\hat{\theta}|x).
\]  

(2.6)
Therefore, by Equations (2.5) and (2.6), we have

\[
E_\pi[L(\hat{\theta}, \theta)] = 1 + F(\hat{\theta} - \alpha|x) - F(\hat{\theta} + \alpha|x) \\
\approx 1 + (F(\hat{\theta}|x) - \alpha\pi(\hat{\theta}|x)) - (F(\hat{\theta}|x) + \alpha\pi(\hat{\theta}|x)) \\
= 1 - 2\alpha\pi(\hat{\theta}|x).
\] (2.7)

To minimise the expected loss function in the form of Equation (2.7), we maximise \(2\alpha\pi(\hat{\theta}|x)\) which is equivalent to maximising \(\pi(\hat{\theta}|x)\). Therefore, the maximiser of the density \(\pi(\hat{\theta}|x)\) is the posterior mode.

2.3 Monte Carlo methods

Monte Carlo methods refer to a class of computational algorithms that rely on repeated random sampling to obtain numerical approximations. They are sometimes referred to as stochastic simulation, and are often used for evaluating integrals, and are based on random samples generated from a density related to a parameter of interest denoted by \(\pi(\theta)\).

2.3.1 Classical Monte Carlo integration

Suppose we want to evaluate the integral

\[
I := E_\pi[h(\theta)] = \int_\Theta h(\theta)\pi(\theta)d\theta.
\] (2.8)

Note that the density \(\pi\) described in this section could represent the posterior distribution \(\pi(\theta|x)\) or any other distribution. The concept of the Monte Carlo method is using an independent sample \(\theta_1, \ldots, \theta_N\) generated from the density \(\pi\) to approximate (2.8) by the empirical average

\[
\hat{I} = \frac{1}{N} \sum_{j=1}^{N} h(\theta_j),
\] (2.9)

since \(\hat{I}\) converges almost surely (a.s.) to \(E_\pi[h(\theta)]\) by the Strong Law of Large Numbers, Robert and Casella (2004), Section 3.2.
Bias and variance of $\bar{I}$

Consider

$$\mathbb{E}_\pi[\bar{I}] = \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}_\pi[h(\theta_j)]$$

$$= \frac{1}{N} \sum_{j=1}^{N} \mathbb{E}_\pi[h(\theta)]$$

$$= \mathbb{E}_\pi[h(\theta)] = I.$$

Therefore, the estimator $\bar{I}$ is an unbiased estimator of $I = \mathbb{E}_\pi[h(\theta)]$. If $h^2$ has a finite expectation with respect to $\pi$, then

$$\text{Var}_\pi(\bar{I}) = \frac{1}{N^2} \sum_{j=1}^{N} \text{Var}_\pi h(\theta_j)$$

$$= \frac{1}{N} \text{Var}_\pi(h(\theta))$$

$$= \frac{1}{N} \int_{\Theta} (h(\theta) - \mathbb{E}_\pi[h(\theta)])^2 \pi(\theta) d\theta. \quad (2.10)$$

The variance (2.10) can be estimated from the generated sample $\theta_1, \ldots, \theta_n$ through the following formula

$$v = \frac{1}{N^2} \sum_{j=1}^{N} (h(\theta_j) - \bar{I})^2.$$

Furthermore, if the variance $\text{Var}_\pi(h(\theta))$ is finite, then, by the Central Limit Theorem, the Monte Carlo error converges in distribution to the Normal distribution,

$$\sqrt{N} (\bar{I} - \mathbb{E}_\pi[h(\theta)]) \xrightarrow{d} N(0, \text{Var}_\pi(h(\theta))) \quad \text{as} \quad N \to \infty. \quad (2.11)$$

This may be used to construct confidence intervals on the approximation of $\mathbb{E}_\pi[h(\theta)]$, Johansen et al. (2007).

2.3.2 Importance sampling

It is sometimes too difficult to sample $\theta$ from the target distribution $\pi$. Importance sampling provides a way to evaluate the integral (2.8) by using a sample from an other distribution called the instrumental distribution, $g(\theta)$, instead of using a sample from
the target distribution $\pi(\theta)$. It is based on the identity

\[
\int_{\Theta} \pi(\theta) d\theta = \int_{\Theta} g(\theta) \frac{\pi(\theta)}{g(\theta)} d\theta \\
= \int_{\Theta} g(\theta) w(\theta) d\theta
\]

for all $g(\cdot)$, such that $g(\theta) > 0$ for (almost) all $\theta$ with $\pi(\theta) > 0$. Then we can generalise this identity (the importance sampling identity) by considering the expectation (2.8),

\[
E_{\pi}[h(\theta)] = \int_{\Theta} \pi(\theta) h(\theta) d\theta \\
= \int_{\Theta} g(\theta) \frac{\pi(\theta)}{g(\theta)} h(\theta) d\theta \\
= \int_{\Theta} g(\theta) w(\theta) h(\theta) d\theta \\
= E_{g}[w(\theta) h(\theta)], \tag{2.12}
\]

if $g(\theta) > 0$ for (almost) all $\theta$ with $\pi(\theta) h(\theta) \neq 0$. Importance sampling is sometimes known as weighted sampling because it uses the weight $w(\theta)$ to correct for the fact that we sample from the instrumental distribution $g(\theta)$.

Assume that we have an independent sample $\theta_1, \ldots, \theta_N$ from a given distribution $g$. Then, provided $E_{g}[w(\theta) h(\theta)]$ exists, by the Strong Law of Large Numbers, we have

\[
\frac{1}{N} \sum_{j=1}^{N} w(\theta_j) h(\theta_j) \overset{a.s.}{\to} E_{g}[w(\theta) h(\theta)],
\]

and hence by (2.12), we have

\[
\frac{1}{N} \sum_{j=1}^{N} w(\theta_j) h(\theta_j) \overset{a.s.}{\to} E_{\pi}[h(\theta)].
\]

As a result, we can estimate $I := E_{\pi}[h(\theta)]$ by

\[
\hat{I} = \frac{1}{N} \sum_{j=1}^{N} w(\theta_j) h(\theta_j). \tag{2.13}
\]

Note that

\[
E_{g}[w(\theta)] = \int_{\Theta} w(\theta) g(\theta) d\theta = \int_{\Theta} \frac{\pi(\theta)}{g(\theta)} g(\theta) d\theta = 1,
\]

so $(1/N) \sum_{j=1}^{N} w(\theta_j)$ converges to 1 as $N \to \infty$ by the Strong Law of Large Numbers.
Bias and variance of the importance sampling estimator

Consider

$$E_g[\hat{I}] = \frac{1}{N} \sum_{j=1}^{N} E_g[w(\theta_j)h(\theta_j)]$$

$$= \frac{1}{N} \sum_{j=1}^{N} E_g[w(\theta)h(\theta)]$$

$$= E_\pi[h(\theta)] = I.$$

Therefore, $\hat{I}$ is an unbiased estimator of $I = E_\pi[h(\theta)]$. The variance of $\hat{I}$ is

$$\text{Var}_g(\hat{I}) = \frac{1}{N^2} \sum_{j=1}^{N} \text{Var}_g(w(\theta_j)h(\theta_j))$$

$$= \frac{1}{N} \text{Var}_g(w(\theta)h(\theta)).$$

Although importance sampling works for every choice of $g$, there are obviously some choices that are better than others, Robert and Casella (2004). While the estimator $\hat{I}$ in Equation (2.13) does converge almost surely to expectation (2.8), its variance is finite only when

$$E_g\left[h^2(\theta) \frac{\pi^2(\theta)}{g^2(\theta)}\right] = E_\pi\left[h^2(\theta) \frac{\pi(\theta)}{g(\theta)}\right] = \int_{\Theta} h^2(\theta) \frac{\pi(\theta)}{g(\theta)} d\theta < \infty.$$

Generally, choices of instrumental distributions that lead to finite variance estimators are preferable. The following two conditions are each sufficient (and quite restrictive) for a finite variance of $\hat{I}$:

- $\pi(\theta) < Mg(\theta)$ $\forall \theta \in \Theta$ and $\text{Var}_\pi(h(\theta)) < \infty$;
- $\Theta$ is compact, $\pi$ is bounded above on $\Theta$, and $g$ is bounded below on $\Theta$.

Among the instrumental distributions $g$ leading to finite variances for the estimator $\hat{I}$, the optimal choice of the instrumental distribution $g$ is the one which gives the minimal variance of $\hat{I}$ which can be determined by the following theorem.

**Theorem 2.3.** The choice of $g$ that minimises the variance of the estimator $\hat{I}$ is

$$g^*(\theta) = \frac{|h(\theta)|\pi(\theta)}{\int_{\Theta} |h(t)|\pi(t)dt},$$

see proof in Robert and Casella (2004), Theorem 3.2.
The practically important result of Theorem 2.3 is that we should choose whenever possible an instrumental distribution \( g \) which satisfies

\[
g(\theta) \propto \pi(\theta)|h(\theta)|.\]

### 2.4 Markov chains

In this section, we present basic notions of Markov chains that are essential to understanding the convergence of Markov chain Monte Carlo (MCMC) methods which are described in the next section. For further details of Markov chains, we refer the reader to Meyn and Tweedie (2009) and Norris (1998). We describe Markov chains by mainly following Chapter 4 from Markov chain Monte Carlo: stochastic simulation for Bayesian inference, Gamerman and Lopes (2006). Therefore, for more details of Markov chains, we refer to this book. Furthermore, we suggest Robert and Casella (2004) for more notions of Markov chains, especially for use in Monte Carlo simulation.

A Markov chain is a special type of stochastic process (random process), which deals with characterisation of sequences of random variables. A collection of random quantities \( \{\theta_t : t \in T\} \) for some index set \( T \) is said to be a stochastic process with state space, \( S \) and index set \( T \). The notation \( \theta_t \) indicates the value of the process at time \( t \). The index set \( T \) is assumed to be the set of natural numbers \( \mathbb{N} \) for the case of discrete time stochastic processes. We turn our attention to results of Markov chains that concern ergodicity and the central limit theorem. These two properties are particularly relevant to MCMC simulation.

We initially describe Markov chains in the discrete state space case which is somewhat easier to understand than the general state space case. The study of general state space Markov chains where continuous state spaces are included is usually more complicated. However, we explain how the concepts described in the context of discrete state spaces might be extended to continuous domains via the use of probability densities that we can relate to MCMC simulation later.
2.4.1 Discrete state spaces

Definition 2.3. A discrete state space Markov chain is a sequence of random quantities with the property that

\[ P(\theta_t|\theta_0, \theta_1, \ldots, \theta_{t-1}) = P(\theta_t|\theta_{t-1}) \quad (2.14) \]

for all \( \theta_0, \theta_1, \ldots, \theta_{t-1}, \theta_t \in S \) and \( t \geq 1 \).

According to property (2.14), the Markov chain is usually characterised by the memorylessness property or the Markov property that the future \( \theta_t \) depends only on the current \( \theta_{t-1} \) in the same state space \( S \).

Definition 2.4. A transition kernel is a function \( P \) defined as:

(i) \( \forall \theta \in S, P(\theta, \cdot) \) is a probability distribution over \( S \);
(ii) \( \forall A \subset S, \) the function \( \theta \mapsto P(\cdot, A) \) can be evaluated.

For discrete state spaces, this transition kernel function is called a transition probability and satisfies:

- \( P(\theta, \phi) \geq 0, \forall \theta, \phi \in S \);
- \( \sum_{\phi \in S} P(\theta, \phi) = 1, \forall \theta \in S \).

The transition probability is represented by the transition probability matrix, \( P_{\theta\phi} := P(\theta, \phi) \) with \( \theta, \phi \) elements which represent the probability of moving from state \( \theta \) to state \( \phi \) and is given by

\[ P_{\theta\phi} = P(\phi|\theta), \quad \theta, \phi \in S. \]

Note that \( P_{\theta\phi}^m \) is the transition probability given by the probability of a chain moving from state \( \theta \) to state \( \phi \) in exactly \( m \) steps.

Definition 2.5. (Recurrent/Positive Recurrent). Let \( T_\phi \) be a hitting (or return) time of \( \phi \). The probability of the chain starting from state \( \theta \) hitting state \( \phi \), at any later step is

\[ \rho_{\theta\phi} = P(T_\phi < \infty). \]

A state \( \phi \) is said to be recurrent if the Markov chain, starting in \( \phi \), eventually returns to \( \phi \) (\( \rho_{\phi\phi} = 1 \)). If a Markov chain starts at a recurrent state \( \phi \), \( T_\phi \) is a finite random
quantity whose mean $E[T_\phi]$ can be evaluated. If this mean is finite, the state $\phi$ is said to be positive recurrent and otherwise the state is called null recurrent.

**Definition 2.6. (Irreducible).** A Markov chain is said to be irreducible if for every $\theta, \phi \in S$, there exists a $k$ such that

$$P(\theta_{t+k} = \phi | \theta_t = \theta) > 0.$$ 

In other words, a chain is irreducible if it is possible to eventually get from any state $\theta$ to any other state $\phi$ in finite number of steps $k$.

**Definition 2.7. (Periodic/Aperiodic).** A state $\theta$ in a discrete state space Markov chain is said to be periodic with period $d(\theta)$ if starting from state $\theta$ the chain returns to it within a fixed number of steps $d(\theta)$ defined as:

$$d(\theta) = \gcd\{t : P(\theta_t = \theta | \theta_0 = \theta) > 0\},$$

where $\gcd$ denotes the greatest common divisor. A chain possessing such a state is said to have a cycle of length $d$. If $d(\theta) = 1$, then the state $\theta$ is said to be aperiodic.

An irreducible Markov chain is positive recurrent if some (and hence all) states $\theta$ are positive recurrent.

**Definition 2.8. (Ergodic).** A state is said to be ergodic if it is aperiodic and positive recurrent. A Markov chain is ergodic if all of its states are ergodic.

A study of Markov chain for simulation is usually a study of the asymptotic behaviour of the chain as the number of steps or iterations $t \to \infty$. A key aspect of the study is a stationary distribution $\pi$.

**Definition 2.9. (Stationary).** A distribution $\pi$ is said to be a stationary distribution of a Markov chain with transition probabilities $P_{\theta\phi}$, if

$$\sum_{\theta \in S} \pi(\theta)P(\theta, \phi) = \pi(\phi) \quad \forall \phi \in S. \quad (2.15)$$

Equation (2.15) can be written in matrix notation as $\pi = \pi P$.

Once the chain reaches a stage where $\pi$ is the distribution of the chain, the chain will retain this distribution for all subsequent stages. This distribution is also known as the invariant or equilibrium distribution.

**Theorem 2.4. (Existence and Uniqueness).** Each irreducible and aperiodic Markov chain has a unique stationary distribution $\pi$. 

22
Theorem 2.5. (Convergence). Let $\theta_t$ be an irreducible and aperiodic Markov chain with stationary distribution $\pi$ and arbitrary initial value $\theta_0$. Then

$$P(\theta_t = \theta_0) \to \pi(\theta), \text{ as } t \to \infty.$$  

Theorem 2.6. (Ergodic Theorem). Given $\theta_1, \theta_2, \ldots, \theta_N$ of an ergodic Markov chain with limiting distribution $\pi$, if $E[h(\theta)|\theta \sim \pi(\theta)] < \infty$, then the sample mean converges to the expectation of $h(\theta)$ under $\pi$ following the Strong Law of Large Numbers,

$$\frac{1}{N} \sum_{i=1}^{N} h(\theta_i) \overset{a.s.}{\to} E[h(\theta)|\theta \sim \pi(\theta)].$$

Assume that we want to obtain the sequence of states $\theta_t, \theta_{t-1}, \ldots$ in reversed order. It can be shown that this sequence satisfies

$$P(\theta_t|\theta_{t+1}, \theta_{t+2}, \ldots) = P(\theta_t|\theta_{t+1})$$

and therefore defines a Markov chain.

Definition 2.10. (Reversible). A Markov chain is said to be reversible if

$$\pi(\theta)P(\theta, \phi) = \pi(\phi)P(\phi, \theta) \text{ for all } \theta, \phi \in S. \quad (2.16)$$

It can be interpreted as saying that the rate at which the system moves from $\theta$ to $\phi$ is the same as the rate at which it moves from $\phi$ to $\theta$. For this reason, Equation (2.16) is sometimes referred to as the detailed balance equation.

Proposition 2.1. If a Markov kernel satisfies the detailed balance condition in Equation (2.16) for some distribution $\pi$ then:

1. $\pi$ is the invariant distribution of the chain.
2. The chain is reversible with respect to $\pi$.

A reversible chain is useful because if there is a distribution $\pi$ satisfying (2.16) for an irreducible chain, then the chain is positive recurrent reversible with stationary distribution $\pi$.  

23
2.4.2 Continuous state spaces

For a continuous state space, there are a few changes from the discrete case but the main results are still valid. Unlike the discrete case, transition matrices presenting transition probabilities cannot be constructed for the continuous case. For the continuous case, if

$$\mathbb{P}(\theta_t \in A | \theta_{t-1} = \phi) = \int_A p(\theta, \phi) d\theta,$$

where $A \subset S$, then the transition kernel is $p(\theta, \phi)$. The stationary or invariant distribution $\pi$ of a chain with transition kernel $p(\theta, \phi)$ must satisfy

$$\int_{-\infty}^{\infty} \pi(\theta)p(\theta, \phi)d\theta = \pi(\phi). \quad (2.17)$$

Equation (2.17) is the continuous version of Equation (2.15). The concepts of invariant distribution, reversibility and detailed balance which are important properties for establishment of limiting results are essentially unchanged from the discrete setting. However, they are defined in different ways. The important condition of reversibility of a chain is given by

$$\pi(\theta)p(\theta, \phi) = \pi(\phi)p(\phi, \theta), \quad \text{for all } \theta, \phi \in S, \quad (2.18)$$

in direct analogy with the discrete case.

2.5 Markov chain Monte Carlo (MCMC)

We have shown that, using importance sampling, we can approximate an expectation $E_{\pi}[h(\theta)]$ without having to sample directly from $\pi$. However, finding an instrumental distribution $g$ that provides efficient estimation of $E_{\pi}[h(\theta)]$ can be difficult, especially in large dimensions. In this section, we describe Markov Chain Monte Carlo (MCMC) methods that allow one to obtain an approximate sample from $\pi$ without having to sample from $\pi$ directly. These methods are based on a Markov chain whose stationary distribution is the distribution of interest $\pi$. Mathematically speaking, the goal of MCMC methods is opposite from a Markov chain. While the concept of studying a Markov chain is to determine conditions under which there exists a stationary distribution and to establish this stationary distribution, the concept of MCMC methods is to sample $\theta$ from the desired (known) stationary distribution, $\pi$. As a result, the stationary distribution is sometimes called the target distribution in MCMC methods. More precisely, MCMC methods construct an ergodic Markov chain which has the target distribution $\pi$ as the stationary distribution. If we run the Markov chain for long
enough, we have an approximate sample from the target distribution that the ergodic average converges to the expectation under $\pi$. In a Bayesian framework, the target distribution $\pi$ is the posterior distribution $\pi(\theta|x)$.

**Definition 2.11.** *A Markov chain Monte Carlo (MCMC) method for the simulation of a distribution $\pi$ is any method producing an ergodic Markov chain $\{\theta_t, t \in T\}$ whose stationary distribution is $\pi$.*

Two of the most popular MCMC methods are the *Metropolis-Hastings* algorithm and *Gibbs sampling*. The name of the Metropolis-Hastings algorithm stems from the work of Metropolis et al. (1953) and Hastings (1970). The Metropolis-Hastings algorithm is based on proposing candidate values sampled from a proposal distribution, which are then either accepted or rejected according to a probability rule. The probability rule reflects how likely the state is to be sampled from the target distribution $\pi$. Gibbs sampling was first proposed by Geman and Geman (1984) and further developed by Gelfand and Smith (1990). Here the transition kernel is formed by the full conditional distributions; it aims to sample from a high-dimensional distribution by sampling from a collection of more tractable lower dimensional distributions. Gibbs sampling is in fact a special case of Metropolis-Hastings sampling in which the candidate value is always accepted as we will see in more detail in Subsection 2.5.4.

### 2.5.1 The Metropolis-Hastings algorithm

The fundamental idea of a Metropolis-Hastings algorithm is to generate a candidate value $\theta^*$ from the arbitrary distribution $q$, and either accept or reject $\theta^*$ as a value from the target distribution $\pi(\theta)$. We initially provide the Metropolis-Hastings algorithm 2.1 then detail how it works in Subsection 2.5.3. Note that we use a superscript $t$ to denote a sample at iteration $t$. 

25
Algorithm 2.1. The Metropolis-Hastings algorithm

1. Choose an arbitrary initial point $\theta^0$ for which $\pi(\theta^0) > 0$.

2. At time $t$
   - Sample a candidate value $\theta^*$ for $\theta^{t+1}$ from $q(\theta^*, \theta^t)$
   - Calculate
     \[
     \alpha(\theta^t, \theta^*) = \min \left( \frac{\pi(\theta^*) q(\theta^*, \theta^t)}{\pi(\theta^t) q(\theta^t, \theta^*)}, 1 \right) \tag{2.19}
     \]

3. If $\alpha(\theta^t, \theta^*) = 1$, then accept the candidate value $\theta^*$ and set $\theta^{t+1} = \theta^*$. If $\alpha(\theta^t, \theta^*) < 1$, then
   - generate $u \sim U(0, 1)$
   - if $u \leq \alpha(\theta^t, \theta^*)$ then accept candidate value $\theta^*$ and set $\theta^{t+1} = \theta^*$, otherwise reject $\theta^*$ and set $\theta^{t+1} = \theta^t$.

4. Repeat step 2-3 until a full sample $(\theta^1, \theta^2, \ldots, \theta^N)$ has been obtained.

The distribution $q$ is called the proposal distribution and the probability $\alpha(\theta^*, \theta^t)$ in Equation (2.19) is the Metropolis-Hastings acceptance probability. If the proposal distribution is symmetric, $q(\theta^t, \theta^*) = q(\theta^*, \theta^t)$, then the acceptance probability (2.19) is given by

\[
\alpha(\theta^*, \theta^t) = \min \left( \frac{\pi(\theta^*)}{\pi(\theta^t)}, 1 \right)
\]

in which case the algorithm reverts to being called just the Metropolis algorithm.

2.5.2 Gibbs sampling

Assume that the target distribution is $\pi(\theta)$ where $\theta = (\theta_1, \ldots, \theta_d)$. Note that the component $\theta_i$ can be a scalar, a vector or even a matrix. For simplicity, we regard each of them as a scalar.

Gibbs sampling is a method for generating samples from a multivariate target distribution and its margins. It simulates the multivariate distribution by using the univariate
full conditional densities as
\[
\pi(\theta_t^i | \theta_1^t, \theta_2^t, \ldots, \theta_{i-1}^t, \theta_{i+1}^t, \ldots, \theta_d^t),
\]
where \( \theta_1^t, \theta_2^t, \ldots, \theta_{i-1}^t, \theta_{i+1}^t, \ldots, \theta_d^t \) are the most recent sampled values for the rest of the variables. The key to Gibbs sampling is that the density of full conditional densities \( \pi(\theta_i | \theta_1, \theta_2, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_d) \) can be found by treating \( \theta_1, \theta_2, \ldots, \theta_{i-1}, \theta_{i+1}, \ldots, \theta_d \) as constants in the joint density \( \pi(\theta_1, \theta_2, \ldots, \theta_d) \).

Let \( \theta_{-i} \) denote the set \( \theta \setminus \theta_i \) and suppose that the full conditional distribution \( \pi(\theta_i | \theta_{-i}) \) for \( i = 1, \ldots, d \) are available and can be sampled from. Gibbs sampling can be implemented by the Algorithm 2.2.

**Algorithm 2.2.** The Gibbs sampling algorithm

1. Choose an arbitrary initial point
   \[
   \theta^{(0)} = (\theta_1^0, \ldots, \theta_d^0) \quad \text{for which} \quad \pi(\theta^0) > 0.
   \]

2. Generate \( \theta_i^t \) for \( i = 1, \ldots, d \) from the conditional distributions as follows
   \[
   \begin{align*}
   \theta_1^{(t)} & \sim \pi(\theta_1 | \theta_2^{t-1}, \theta_3^{t-1}, \ldots, \theta_d^{t-1}) \\
   \theta_2^{(t)} & \sim \pi(\theta_2 | \theta_1^{t}, \theta_3^{t-1}, \ldots, \theta_d^{t-1}) \\
   & \vdots \\
   \theta_k^{(t)} & \sim \pi(\theta_k | \theta_1^{t}, \ldots, \theta_{k-1}^{t}, \theta_{k+1}^{t-1}, \ldots, \theta_d^{t-1}) \\
   & \vdots \\
   \theta_d^{(t)} & \sim \pi(\theta_d | \theta_1^{t}, \ldots, \theta_{d-1}^{t})
   \end{align*}
   \]

When convergence is reached, the resulting value \( \theta^t \) is an approximate sample drawn from the distribution \( \pi(\theta) \). Gibbs sampling is considered as a special case of the Metropolis-Hastings algorithm in which the candidate is always accepted. However, Gibbs sampling can provide very correlated samples that take a long time to move around the state space.
2.5.3 How the Metropolis-Hastings algorithm works

The following notion of the Metropolis-Hastings algorithm is based on the work in Chib and Greenberg (1995). Consider a discrete Markov chain $\theta_t$ with transition probabilities

$$P(\theta, \phi) = \mathbb{P}(\theta_t = \phi | \theta_{t-1} = \theta) \quad \text{for all } \theta, \phi \in S,$$

where $S$ is the state space of a chain. For the continuous case, transition probabilities can be represented as

$$P(\theta, \phi) = p(\theta, \phi).$$

Note that $p(\theta, \phi)$ indicates the transition from $\theta$ to $\phi$.

Suppose that the chain has a stationary distribution $\pi$. Irreducibility and aperiodicity are sufficient conditions in order that $\pi$ be the stationary distribution (by Theorem 2.5). This means the stationary distribution $\pi = \{ \pi(\theta) : \theta \in S \}$ satisfies

$$\pi(\phi) = \sum_{\theta \in S} \pi(\theta) P(\theta, \phi),$$

in the discrete case. For the continuous case as in Equation (2.17), we have

$$\pi(\phi) = \int_{\theta} \pi(\theta) p(\theta, \phi) d\theta. \quad (2.20)$$

The kernel transition $p(\theta, \phi)$ denotes probability when the chain moves (from $\theta$ to $\phi$), hence we denote $p(\theta, \theta)$ as the probability that the chain remains in the same state. Therefore, the kernel transition $p(\theta, \phi)$ can be written in the general form

$$p(\theta, \phi) = p^*(\theta, \phi) + \mathbb{I}(\theta, \phi)p(\theta, \theta),$$

where

$$\mathbb{I}(\theta, \phi) = \begin{cases} 1 & \text{if } \phi = \theta \\ 0 & \text{otherwise.} \end{cases}$$

We can see that the chain must move somewhere as

$$\int p(\theta, \phi) d\phi = \int (p^*(\theta, \phi) + \mathbb{I}(\theta, \phi)p(\theta, \theta)) d\phi = 1. \quad (2.21)$$

If $p^*(\theta, \phi)$ satisfies the reversible condition or detailed balance, then

$$\pi(\theta)p^*(\theta, \phi) = \pi(\phi)p^*(\phi, \theta), \quad \text{for all } \theta, \phi \quad (2.22)$$
(see Definition 2.10, Proposition 2.1 and Equation (2.18)). Therefore, \( p(\theta, \phi) \) is the transition kernel of a Markov chain with stationary distribution \( \pi \). To verify this we consider the right-hand side of Equation (2.20):

\[
\int \pi(\theta)p(\theta, \phi)d\theta = \int (\pi(\theta)p(\theta, \phi) + \pi(\phi)p(\phi, \phi))d\theta
\]

\[
= \int (\pi(\theta)p(\theta, \phi) + \pi(\phi)(\phi, \theta)p(\phi, \phi))d\theta
\]

\[
= \int (\pi(\phi)p(\phi, \theta) + \pi(\phi)(\phi, \theta)p(\phi, \phi))d\theta
\]

\[
= \pi(\phi) \left[ \int (p(\phi, \theta) + \pi(\phi)p(\phi, \phi))d\theta \right]
\]

\[
= \pi(\phi). \quad \text{(by Equation (2.22))}
\]

We can relate the notion described above to the Metropolis-Hastings algorithm as follows: Let \( q(\theta, \phi) \) be the proposal density. If the proposal density satisfies the detailed balance, then

\[
\pi(\theta)q(\theta, \phi) = \pi(\phi)q(\phi, \theta), \quad \text{for all } \theta, \phi \in S. \quad (2.23)
\]

We can use \( q(\theta, \phi) \) as the transition kernel of a Markov chain with stationary distribution \( \pi \). However, it is unlikely that the proposal density \( q(\theta, \phi) \) satisfies detailed balance. We might find, for some \( \theta, \phi \) that

\[
\pi(\theta)q(\theta, \phi) > \pi(\phi)q(\phi, \theta). \quad (2.24)
\]

In this case, the process moves from \( \theta \) to \( \phi \) too often and from \( \phi \) to \( \theta \) too rarely. This can be corrected by reducing the number of moves from \( \theta \) to \( \phi \) with an acceptance probability, \( \alpha(\theta, \phi) < 1 \) such that

\[
\pi(\theta)q(\theta, \phi)\alpha(\theta, \phi) = \pi(\phi)q(\phi, \theta).
\]

Therefore,

\[
\alpha(\theta, \phi) = \frac{\pi(\phi)q(\phi, \theta)}{\pi(\theta)q(\theta, \phi)}
\]

Note that we do not reduce the number of moves from \( \phi \) to \( \theta \), so we take \( \alpha(\theta, \phi) = 1 \). On the other hand, if we reverse the position of \( \theta \) and \( \phi \) in inequality (2.24), it gives the general formula for the acceptance probability,

\[
\alpha(\theta, \phi) = \min \left( \frac{\pi(\phi)q(\phi, \theta)}{\pi(\theta)q(\theta, \phi)}, 1 \right).
\]
This means that we can write the transition kernel of the Markov chain with stationary distribution as function of two elements: an arbitrary transition kernel \( q(\theta, \phi) \) known as a proposal distribution and a probability \( \alpha(\theta, \phi) \) such that
\[
p^*(\theta, \phi) = q(\theta, \phi)\alpha(\theta, \phi), \quad \text{if } \theta \neq \phi
\]
Consequently, there is a positive probability left for the chain to remain at \( \theta \) given by
\[
p(\theta, \phi) = 1 - \int q(\theta, \phi)\alpha(\theta, \phi)d\phi.
\]
We can make the link to the acceptance probability (2.19) by taking \( \theta = \theta^t \) and \( \phi = \theta^* \) in Algorithm 2.1.

2.5.4 Gibbs sampling is a special case of Metropolis-Hastings sampling

In this subsection, we will show that Gibbs sampling is a special case of Metropolis-Hastings sampling where the candidate value is always accepted.

Recall the Metropolis-Hastings sampling with the target density \( \pi(\theta) \); for each \( t = 2, 3, \ldots, N \), the proposal density \( q(\theta^{t+1}|\theta^t) \) is used to simulate a candidate value \( \theta^* \) for \( \theta^{t+1} \) which is accepted with probability
\[
\alpha(\theta^t, \theta^*) = \min \left( \frac{\pi(\theta^*)q(\theta^*, \theta^t)}{\pi(\theta^t)q(\theta^t, \theta^*)}, 1 \right).
\]
In Gibbs sampling, univariate conditional distributions are used to generate the single element \( \theta_i \), \( i = 1, 2, \ldots, d \). Therefore, the single element \( \theta_i \) is only sampled once each \( d \) iterations. We consider the single move from \( \theta^t \) to \( \theta^{t+1} \) to show that Gibbs sampling is a special case of Metropolis-Hastings sampling as follows.

Let \( \theta^* \) be the candidate value for \( \theta^{t+1} \) so that
\[
\theta^* = (\theta_1^t, \theta_2^t, \ldots, \theta_{i-1}^t, \theta_i^*, \theta_{i+1}^{t-1}, \ldots, \theta_d^{t-1})
\]
and
\[
\theta^t = (\theta_1^t, \theta_2^t, \ldots, \theta_{i-1}^t, \theta_{i+1}^{t-1}, \ldots, \theta_d^{t-1}).
\]
In Gibbs sampling, generating \( \theta^* \) is equivalent to generating \( \theta_i^* \) for \( \theta_i \) from its full
conditional density. Therefore, we can use

\[ q(\theta^t, \theta^*) = \pi(\theta^* | \theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^t_{i+1}, \ldots, \theta^t_d) \]

and

\[ q(\theta^*, \theta^t) = \pi(\theta^t_{i-1} | \theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^t_{i+1}, \ldots, \theta^t_d) \]

as the proposal densities. Therefore, the acceptance probability for accepting \( \theta^* \) as \( \theta^{t+1} \) in the Metropolis-Hastings algorithm is given by

\[ \alpha(\theta^t, \theta^*) = \min \left( \frac{\pi(\theta^*) q(\theta^* , \theta^t)}{\pi(\theta^t) q(\theta^t, \theta^*)} , 1 \right). \]

In order to show that Gibbs sampling is a special case of Metropolis-Hastings sampling where the acceptance probability is always 1, we will show

\[ \frac{\pi(\theta^*) q(\theta^* , \theta^t)}{\pi(\theta^t) q(\theta^t, \theta^*)} = 1. \]

Consider two random vectors \( z_1 \) and \( z_2 \), so by the joint distribution for conditionally dependent variables;

\[ \pi(z_2) = \frac{\pi(z_1, z_2)}{\pi(z_1 | z_2)}, \quad (2.25) \]

Let \( z_2 = (\theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^t_{i+1}, \ldots, \theta^t_d) \). So when \( z_1 = \theta^*_i \), \( \theta^* = (z_1, z_2) \). Therefore,

\[ \frac{\pi(\theta^*)}{q(\theta^t, \theta^*)} = \frac{\pi(\theta^*)}{\pi(\theta^* | \theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^t_{i+1}, \ldots, \theta^t_d)} = \frac{\pi(\theta^*)}{\pi(z_1, z_2)} = \frac{\pi(z_2)}{\pi(z_1 | z_2)} = \pi(z_2) \quad \text{By equation (2.25)} \]

\[ = \pi(\theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^t_{i+1}, \ldots, \theta^t_d). \quad (2.26) \]

Similarly,

\[ \frac{\pi(\theta^t)}{q(\theta^*, \theta^t)} = \frac{\pi(\theta^t)}{\pi(\theta^t_{i-1} | \theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^t_{i+1}, \ldots, \theta^t_d)} = \frac{\pi(z_1, z_2)}{\pi(z_1 | z_2)} = \pi(z_2) \quad \text{By equation (2.25)} \]

\[ = \pi(\theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^t_{i+1}, \ldots, \theta^t_d). \quad (2.27) \]
Equations (2.26) and (2.27) result in

\[
\frac{\pi(\theta^*) q(\theta^*, \theta^t)}{\pi(\theta^t) q(\theta^t, \theta^*)} = \frac{\pi(\theta^*) / q(\theta^*, \theta^t)}{\pi(\theta^t) / q(\theta^t, \theta^*)} \\
= \frac{\pi(\theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^{(t-1)}_{i+1}, \ldots, \theta^t_d)}{\pi(\theta^t_1, \theta^t_2, \ldots, \theta^t_{i-1}, \theta^{(t-1)}_{i+1}, \ldots, \theta^t_d)} \\
= 1.
\]

As a result, all sampled values are always accepted when the proposal density is the full conditional of the Gibbs sampler.

### 2.5.5 Remark on algorithms

The efficiency of the Metropolis-Hastings algorithm depends on the choice of the proposal distribution. It is suggested that a good choice of the proposal density should be close to the target density \(\pi(\theta)\) but slightly heavier in the tails to obtain a high acceptance probability. Ideally, we want to explore all possible values of \(\theta\) without spending too long in only one small area of the distribution. In contrast, we do not need to worry about the acceptance probability in the Gibbs sampling as the candidate is always accepted. Nevertheless, we might face slow mixing problem when a chain moves around very slowly because the parameters are highly correlated.

### 2.6 Dealing with MCMC samples

In this section, we discuss some characteristics of MCMC samples which naturally arise. One important issue is the convergence in MCMC. It is important to remember that an approximate \(\theta^t\) can only be considered as a value sampled from the stationary distribution \(\pi(\theta)\) (the target distribution) after the Markov chain has converged. Another type of convergence is the convergence of the empirical average to the expectation,

\[
\frac{1}{N} \sum_{t=1}^{N} h(\theta^t) \to E_\pi[h(\theta)]. \quad (2.28)
\]

In this thesis, we place emphasis on the convergence of ergodic averaging in Equation (2.28) which is used for Bayesian point estimation. The following subsection provides an overview of the convergence in MCMC.
2.6.1 Convergence of the MCMC output

Strictly speaking, a Markov chain $\theta^t$ converges to the stationary distribution as $t \to \infty$. In practice, however, we often use a subset of the chain. Then it would be desirable to be able to say that the Markov chain has converged and the sample values from the subset can be considered as a sample from $\pi$. Several methods are commonly used for diagnosing convergence. They can be classified into two categories: graphical methods and numerical methods (or a combination of both). The most basic approach to diagnosing convergence of the MCMC output is to plot the approximate sample path $\theta^t$ and see how well our chain is mixing, or moving around the parameter space. Ideally, the plot should be oscillating very fast and show very little structure or trend. However, it is impossible to see from a plot of the sample path whether the chain has explored the entire support of the distribution. Furthermore, it is often that the graph of the raw sequence $\theta^t$ is not helpful for diagnosing convergence. Several methods for diagnostics in MCMC have been suggested. One can use the method in Gelman and Rubin (1992) when parallel chains are run. It is also natural to use time series methods to assess convergence as well. For instance, Geweke's spectral density diagnostic and Heidelberger and Welch’s convergence diagnostic, Brooks and Roberts (1998).

In a Monte Carlo setting, we concentrate on convergence of the empirical average (2.28) to $E_\pi[h(\theta)]$. The purpose of the convergence assessment is actually to determine whether the chain has explored all the features of $\pi$, e.g., all the modes. In the statistical literature, this convergence is often related to the mixing of the chain, Brooks and Roberts (1998). A chain is said to be poorly mixing if it stays in small regions of the parameter space for long periods of time as opposed to a well mixing chain that seems to explore around the space. In general, MCMC successive values are not independent, which makes the method converge slower than the Monte Carlo method. However, if a chain is rapidly mixing, the dependence is rapidly decaying over successive iterations and hence it converges faster. In other words, mixing is connected to the speed of forgetting the initial value or distribution of the Markov chain, see Tierney et al. (1994). We refer readers to the papers of Gelman and Rubin (1992), Cowles and Carlin (1996), Brooks and Roberts (1998) and Robert and Casella (2004), Chapter 12 for a more detailed review of diagnosing convergence methods.

2.6.2 Burn-in period

As a Markov chain needs to converge, the early values of the chain before convergence are called “burn-in”. The name burn-in comes from electronics. Many electronics components fail quickly. Therefore, a burn-in is done to eliminate the worst ones,
Brooks et al. (2011). The length of burn-in depends upon the rate of convergence which specifies how close the chain is to the stationary distribution. For practical purposes, there will be some value, $m$, such that after which all future values $\theta^m, \theta^{m+1}, \ldots$, appear to come from the same distribution. Thus we can assume that the Markov chain has converged and the values $\theta^m, \theta^{m+1}, \ldots, \theta^{m+n}$ can be considered as an approximate sample from the stationary distribution $\pi$. As a result, we discard early $m$ values of the sample and work with the values of the sample $\{\theta^t : t > m\}$. In general, it is unclear how much we should burn-in since we cannot determine exactly when the convergence occurs. Furthermore, it is difficult to estimate the rate of convergence and so to determine the length of the required burn in might not be feasible.

### 2.6.3 Effective Sample Size

Suppose we have a subset of a Markov chain $\{\theta^t : t = 1, \ldots, N\}$ which is ergodic and has stationary distribution $\pi$. Then the empirical average $\bar{h}_N = \sum_{t=1}^{N} h(\theta^t)$ is our estimator of $E_{\pi}[h(\theta)]$ according to the ergodic theorem. How good is this estimator? This estimator has bias and variance whose asymptotic forms are Green and Han (1992):

$$
E[\bar{h}_N] - E_{\pi}[h(\theta)] = \frac{1}{N} \sum_{t=1}^{N} \{E[h(\theta^t)] - E_{\pi}[h(\theta)]\} \\
\sim \frac{1}{N} \sum_{t=1}^{\infty} \{E[h(\theta^t)] - E_{\pi}[h(\theta)]\} \\
\text{Var}(\bar{h}_N) = \frac{1}{N} \sum_{s=1}^{N} \sum_{t=1}^{N} \text{Cov}(h(\theta^s), h(\theta^t)) \\
\sim \frac{\sigma^2}{N} \sum_{t=-\infty}^{N-1} \left(1 - \frac{|t|}{N}\right) \rho_t(h) \\
\sim \frac{\sigma^2}{N} \sum_{t=-\infty}^{\infty} \rho_t(h),
$$

where $\rho_t(h)$ is the autocorrelation function of the process $\{h(\theta^t)\}$ under the stationary distribution $\pi$ and $\sigma^2$ is the stationary variance of $h(\theta)$. The asymptotic variance is a factor

$$
\tau(h) = \sum_{t=-\infty}^{\infty} \rho_t(h)
$$

times what would be obtained if independent random sampling of $\theta$ from $\pi$ could be achieved, Green and Han (1992). We call $\tau(h)$ the integrated autocorrelation time.
As a result, for large $N$, the variance of the empirical mean is approximately

$$\text{Var}(\bar{h}_N) \sim \frac{1}{N} \text{Var}_\pi(h(\theta)) \tau(h). \quad (2.29)$$

In general, the integrated autocorrelation time cannot be determined directly, so it needs to be estimated (see Robert and Casella (2004), Section 12.6.1). For instance, it could be estimated by

$$\hat{\tau}(h) = 1 + \sum_{t=1}^{M} \hat{\rho}_t(h),$$

where $M = 2m + 1$ which is defined by

$$m = \max\{t \in \mathbb{N}_0 : \hat{\rho}_{2k}(h) + \hat{\rho}_{2k+1}(h) > 0, \text{ for all } k = 0, 1, \ldots, t\}$$

and the autocorrelations can be estimated by

$$\hat{\rho}_t(h) = \frac{\hat{\gamma}_t(h)}{\hat{\gamma}_0(t)}, \quad \text{where} \quad \hat{\gamma}_t(h) = \frac{1}{N} \sum_{i=1}^{N-t} (h(\theta^i - \bar{h}_N))(h(\theta^{i+t}) - \bar{h}_N).$$

This estimation was proposed in Priestley (1981).

If the Markov chain is ergodic and reversible, Geyer (1992), Theorem 2.1, then

$$\sqrt{N}(\bar{h}_N - E_\pi[h(\theta)]) \overset{d}{\to} N(0, \text{Var}_\pi(h(\theta)) \tau(h)) \quad \text{as} \quad N \to \infty$$

(which is similar to the Monte Carlo error (2.11)).

We can use the quantity derived from MCMC output called the Effective Sample Size (ESS), which gives the size of an approximate i.i.d sample in a chain, Robert and Casella (2004), Subsection 12.3.5. The ESS formula is given by

$$\text{ESS} = N/\tau(h). \quad (2.30)$$

As the approximate of the stationary variance of $h(\theta)$ is

$$\hat{\text{Var}}_\pi(h(\theta)) = \frac{1}{N} \sum_{t=1}^{N} (h(\theta^t) - \bar{h}_N)^2,$$
so the approximate variance of the empirical mean, (2.29) is obtained from

\[
\hat{\text{Var}}(\bar{h}_N) = \frac{1}{N^2} \sum_{t=1}^{N} (h(\theta^t) - \bar{h}_N)^2 \tau(h) \\
= \frac{1}{N \times \text{ESS}} \sum_{t=1}^{N} (h(\theta^t) - \bar{h}_N)^2,
\]

where ESS is given in Equation (2.30). Note that if the original sample is independent which means the autocorrelation time is 1, so the effective sample size ESS is the same as \( N \). Thus Equation (2.31) is exactly the same as the standard variance estimator.

There is a package called CODA available in the statistical R software, developed in Best et al. (1995). This package can be used to analyse the output of Metropolis-Hastings and Gibbs sampling algorithms. We thus estimate the value of effective sample size, ESS by using the following command,

\[
> \text{library(coda)} \\
> \text{effectiveSize}.
\]
Chapter 3

Optimisation Methods

3.1 Introduction

In this chapter, we provide a basic outline of some optimisation methods that could be used for searching for the approximate value $\hat{\theta}$ which minimises the expected loss function (2.1). For convenience, we describe optimisation methods as numerical techniques for dealing with a conventional optimisation problem rather than methods for minimisation of the expected loss function. Mathematically speaking, optimisation is either minimisation or maximisation but we pay attention to the minimisation problem:

$$\theta^* = \arg\min_{\theta} f(\theta),$$

(3.1)

where $\theta \in \mathbb{R}^n$ is a real vector with $n \geq 1$ components. The function $f$ is said to be the cost function. Note that it can be simply converted to the maximisation problem by replacing the cost function $f(\theta)$ by $-f(\theta)$,

$$\min_{\theta} f(\theta) \equiv \max_{\theta} (-f(\theta)).$$

Optimisation methods are used in many different contexts. In practice, users are responsible for choosing an algorithm that is appropriate for a specific application. There are two main approaches for the optimisation problem (3.1): deterministic numerical approaches and simulation methods. Deterministic numerical methods are more reliant on analytical properties of the cost function $f$ such as convexity, smoothness and boundedness than the simulation methods which are usually more forgiving of constraints on the domain of $\theta$ and on the function $f$. For some optimisation problems, using deterministic numerical methods can provide an exact solution, whilst simulation methods rarely achieve an exact solution. Nonetheless, the traditional deterministic numerical
methods, such as Newton-Raphson or quasi-Newton methods can suffer severely from trapping in local minima (maxima) for some problems. Simulation methods might be more appropriate for those problems.

In Section 3.2, we provide an overview of some deterministic methods, in particular ones available in R via the function “optim”. The overview of the deterministic methods in this chapter leans heavily on the numerical optimisation book written by Nocedal and Wright (1999). In Section 3.3, we describe the fundamental concept of simulated annealing (also available in R). In Section 3.4, we compare the performance of simulated annealing with the deterministic methods described in Section 3.2.

3.2 Deterministic optimisation methods

3.2.1 The Nelder-Mead method

The Nelder-Mead method was introduced in Nelder and Mead (1965) and is applicable to minimisation problems of \( n \) variables. It is also known as the Nelder-Mead (downhill) simplex search since the method is based on the simplex search. A simplex is the structure formed by \( (n + 1) \) points in a \( n \)-dimensional space. The simplex adapts itself to the local landscape and moves on to the neighbourhood of a minimum or the final minimum. The fundamental algorithm can be briefly described as follows: the function is evaluated at each point (vertex) of the simplex, and the vertex having the highest function value is then replaced by a new vertex with a lower function value. The main operations to locate such a vertex having a lower function value are reflection, contraction and expansion. If reflection has produced a new minimum, then we find a new simplex by using the expansion operation and restart the process, but if we have failed, we use contraction; see more details in Nelder and Mead (1965) and Nash (1990). The algorithm of Nelder-Mead method requires function evaluations without derivatives so it is suitable for functions which are not easily expressed as analytic forms, such as the output of simulations. However, it might take an unnecessarily large number of function evaluations to locate a solution. Therefore, it is not well customised to problems with a large number of variables. The major advantage of the Nelder-Mead method is that it works reasonably well for non-differentiable functions because it uses only function values. However, it is relatively slow and performs inefficiently in high dimensions.
3.2.2 Newton’s method

Newton’s method (also known as the Newton-Raphson method), is a method for finding the roots (or zeroes) of a real-valued function; \( \theta : f(\theta) = 0 \). In optimisation, a procedure also called Newton’s method is applied to the derivative of a function to find its zeros, \( f'(\theta) = 0 \). The idea of Newton’s Method is to approximate the function locally by the first three terms of its Taylor expansion, and to set the next iterate to be the minimiser of the approximation. The algorithm chooses a direction \( p \) and searches along this direction from the current iterate \( \theta_k \) for a new iterate \( \theta_{k+1} \) with a lower function value. Newton’s method produces the search direction which is derived from the second-order Taylor series approximation to \( f(\theta_k + p) \),

\[
f(\theta_k + p) \approx f(\theta_k) + p^T \nabla f(\theta_k) + \frac{1}{2} p^T \nabla^2 f(\theta_k) p,
\]

where \( \nabla \) and \( \nabla^2 \) are the first and the second derivatives of the function \( f \), respectively. We denote these three terms of the Taylor series expansion as \( m_k(p) \). Assuming that the Hessian \( \nabla^2 f(\theta_k) \) is positive definite (\( p^T \nabla^2 f(\theta_k) p > 0 \) for all \( p \neq 0 \)), we obtain the (Newton) direction by finding the vector \( p \) which minimises \( m_k(p) \) as

\[
p_k = - (\nabla^2 f(\theta_k))^{-1} \nabla f(\theta_k).
\]

The iterative scheme can be set up using the direction \( p_k \) defined by \( p = \theta_{k+1} - \theta_k \), thus

\[
\theta_{k+1} = \theta_k - (\nabla^2 f(\theta_k))^{-1} \nabla f(\theta_k).
\]

Newton’s method can often converge remarkably quickly, especially if the initial values are sufficiently close to the optimal solution. However, the main drawback of Newton’s method is the need for the Hessian \( \nabla^2 f(\theta_k) \) for which the explicit computation of the matrix of second derivatives can be troublesome and very expensive. Moreover, when \( \nabla^2 f(\theta_k) \) is not positive definite, the direction \( p \) may not even be defined since \( (\nabla^2 f(\theta_k))^{-1} \) may not exist. Another drawback is that if \( f(\theta) \) is not a convex function, Newton’s method may sometimes diverge or converge to saddle points and local minima.
3.2.3 Quasi-Newton methods

Quasi-Newton methods are alternatives to Newton’s method that do not require computation of the true Hessian, $\nabla^2 f(\theta_k)$ and yet still converge quickly. Instead of using the true Hessian $\nabla^2 f(\theta_k)$, they use an approximation of the Hessian denoted by $B_k$. One of the most popular quasi-Newton methods is the BFGS method, named by Broyden, Fletcher, Goldfarb and Shannon who invented the algorithm, and which is defined by

\[ B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{\lambda_k \lambda_k^T}{s_k^T s_k}, \]  

(3.2)

where

\[ s_k = \theta_{k+1} - \theta_k, \quad \lambda_k = \nabla f(\theta_{k+1}) - \nabla f(\theta_k). \]

The BFGS update (3.2) generates positive definite approximations whenever the initial approximation $B_0$ is positive definite and $s_k^T \lambda_k > 0$ Nocedal and Wright (1999). The BFGS method replaces the true Hessian by $B_k$ to formulate the search direction

\[ p_k = -B_k^{-1} \nabla f(\theta_k). \]

The quasi-Newton methods require only the gradient of the objective function (the first derivative of the function, $\nabla f(\theta)$). The second derivatives are not required. As a result, quasi-Newton methods are sometimes more efficient than Newton’s methods. Another member of the group of quasi-Newton methods discussed here is the limited-memory BFGS (L-BFGS), Nocedal (1980), Liu and Nocedal (1989). It is based on the BFGS method and updates using a formula which generates matrices using information from the last $m$ iterations where $m$ is a number specified by the user. The method uses the inverse BFGS formula to obtain an approximation of the inverse of the Hessian matrix, $H$,

\[ H_{k+1} = V_k^T H_k V_k + p_k s_k s_k^T, \]

where

\[ p_k = \frac{1}{\lambda_k^T s_k} \quad \text{and} \quad V_k = I - p_k \lambda_k s_k^T. \]

There is a variant of L-BFGS is called the “L-BFGS-B” method which allows box constraints, that is each variable can be given a lower and/or upper bound.
The L-BFGS-B is used to solve problems, such as:

$$\min_{\theta} f(\theta), \quad \text{subject to} \quad l \leq \theta \leq u.$$ 

Overall, the BFGS method performs quite as well as the Newton method does, and even better for non-smooth optimisation problems. Nevertheless, global convergence for general nonlinear cost functions is not established properly; it is not guaranteed that the iterates of BFGS method approach a stationary point of the problem from any starting point.

### 3.2.4 The conjugate gradient method

The conjugate gradient method is an algorithm for solving a large linear system of equations and can be modified to solve nonlinear optimisation problems. The linear conjugate gradient method was proposed by Hestenes and Stiefel (1952) as an iterative algorithm for solving linear systems with positive definite coefficient matrices.

Consider the linear system of equations,

$$A\theta = b,$$  \hspace{1cm} (3.3)

where $A$ is an $n \times n$ symmetric ($A^T = A$) and positive definite matrix ($\theta^T A \theta > 0$ for all non-zero vectors $\theta \in \mathbb{R}^n$). We denote the unique solution of Equation (3.3) by $\theta^*$. One of the noticeable properties of the conjugate gradient method is that it generates a set of vectors with a property known as conjugacy. A set of nonzero vectors $\{p_0, p_1, \ldots, p_l\}$ is said to be conjugate with respect to the symmetric positive definite matrix $A$ if

$$p_i^T A p_j = 0, \quad \text{for all} \ i \neq j.$$ 

Suppose we can expand solution $\theta^*$:

$$\theta^* = \sum_{i=1}^{n} \alpha_i p_i,$$

where the $\{p_i\}$ are the conjugate directions and so

$$b = A\theta^* = \sum_{i=1}^{n} \alpha_i A p_i.$$
For any $p_k$,
\[
p_k^T b = p_k^T A \theta^* = \sum_{i=1}^n \alpha_i p_k^T A p_i = \alpha_k p_k^T A p_k.
\]

This means that we find a sequence of $n$ conjugate directions and then compute the coefficients $\alpha_k$. The conjugate gradient method is regarded as an iterative method. We denote the initial guess by $\theta_0$ and without loss of generality we assume that $\theta_0 = 0$. Starting with $\theta_0$ we search for solution $\theta^*$ and in each iteration we need a metric to tell us whether we are closer to the solution. This metric comes from the fact that the solution is also the unique minimizer of the convex quadratic function:
\[
\phi(\theta) = \frac{1}{2} \theta^T A \theta - b^T \theta.
\]

Therefore, if the function $\phi(\theta)$ becomes smaller in an iteration, we are closer to $\theta^*$. As a result, we can use the conjugate gradient method as an algorithm for solving either the linear system of equations (3.3) or as an algorithm for minimising the convex quadratic function
\[
\phi(\theta) = \frac{1}{2} \theta^T A \theta - b^T \theta.
\]

(3.4)

The gradient of $\phi$ is then expressed by
\[
\nabla \phi(\theta) = A \theta - b.
\]

(3.5)

We denote the gradient of $\phi$ as the residual of the linear system, that is at $\theta = \theta_k$, we have
\[
r_k = A \theta_k - b.
\]

The conjugacy property allows us to minimise $\phi$ in $n$ steps by successively minimising it along the individual directions in a conjugate set. The method of conjugate direction can be described as follows:

Given a starting point $\theta_0 \in \mathbb{R}^n$ and a set of conjugate directions $\{p_0, p_1, \ldots, p_{n-1}\}$, we generate the sequence \{\theta_k\} by setting
\[
\theta_{k+1} = \theta_k + \alpha_k p_k,
\]

(3.6)

where $\alpha_k$ is the scalar minimiser of the quadratic function $\phi$ along $\theta_k + \alpha p_k$, given by
\[
\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k}.
\]

(3.7)
Theorem 5.1 in Nocedal and Wright (1999) shows that the sequence \( \{\theta_k\} \) generated by (3.6) and (3.7) converges to the solution of the linear system (3.5) in at most \( n \) steps. A conjugate direction method (3.6), (3.7) is based on any choice of the conjugate direction set \( \{p_0, p_1, \ldots, p_{n-1}\} \). We can generate \( p_k \) by using \( p_{k-1} \) without knowing all the other previous vectors \( p_0, p_1, \ldots, p_{k-2} \). We write

\[
p_k = -r_k + \beta_k p_{k-1},
\]

where the scalar \( \beta_k \) is obtained from

\[
\beta_k = \frac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}}.
\]

We have described the conjugate gradient method for minimising the convex quadratic function \( \phi \) in Equation (3.4). Now we present the conjugate gradient for a nonlinear minimisation problem. Indeed, many variants of conjugate gradient method have been suggested, for example the Polak-Ribière method and variants; see more details in Nocedal and Wright (1999). The nonlinear conjugate gradient method described here was introduced in Fletcher and Reeves (1964). Fletcher and Reeves extend the conjugate gradient method (3.8), (3.9) by following algorithm,

\[
p_{k+1} = \nabla f_{k+1} + \beta_{k+1}^{FR} p_k,
\]

where

\[
\beta_{k+1}^{FR} = \frac{\nabla f_{k+1}^T \nabla f_k}{\nabla f_k^T \nabla f_k}.
\]

The advantage of the conjugate gradient method over other methods is that it does not store a matrix. As a result, it is suitable for dealing with large optimisation problems. However, it tends to be more unstable than the BFGS method (it is unstable with respect to even small perturbations).
3.3 Simulated annealing

Simulated annealing (SA) is a generalisation of a Monte Carlo method used to simulate the annealing process that was introduced by Metropolis, Metropolis et al. (1953). The name annealing is borrowed from metallurgy where it refers to the manner of heating a solid and then cooling it slowly. In Kirkpatrick et al. (1983), they took the idea of the Metropolis algorithm and applied it to an optimisation problem in the context of a local search procedure. From the physics point of view, denoting the function to be minimised as energy \( E \) and \( T \) as a temperature which is to decrease slowly (annealing), simulated annealing can be explained as follows: An annealing process, initially at high temperature and disordered, is slowly cooled so that the system at any time is approximately in thermodynamic equilibrium. As cooling proceeds, the system becomes more ordered and approaches a frozen ground state at \( T = 0 \). Precisely, if the change in energy is negative, the energy state of the new configuration is lower and the new configuration is accepted. If the change in energy is positive, the new configuration has a higher energy state; however, it may still be accepted according to the Boltzmann probability factor:

\[
\mathbb{P} = \exp(-\Delta E/k_B T),
\]

where \( k_B \) is the Boltzmann’s constant, and \( T \) is the current temperature. Note that if the initial temperature of the system is too low or if cooling is done insufficiently slowly the system may become quenched, forming defects which means it is trapped in a local minimum energy state; see more details in Aarts and Korst (1988) and Kirkpatrick et al. (1983).

Simulated annealing for minimisation problem (3.1) uses the cost function \( f \) in place of the energy and specifies configurations by a set of parameters \( \theta^i \). Hence, we obtain the simulated annealing algorithm for searching the minimum of \( f \) as follows:

Given a temperature parameter \( T > 0 \) and a starting point \( \theta^0 \), the SA algorithm generates a succession of samples \( \theta^1, \theta^2, \ldots, \theta^i, \ldots \) tending to the global minimum of the cost function. New candidate samples are generated on a neighborhood of the current point \( \theta^i \) from the distribution, \( \pi \) such that

\[
\pi(\theta) \propto \exp(-f(\theta)/T).
\]

Note that a candidate sample can be simulated from a uniform distribution but it may be very slow since it does not take into account any specific feature of the cost function \( f \). A candidate sample \( \theta' \) is accepted or rejected according to the Metropolis criterion. We always accept \( \theta' \) if \( f(\theta') \leq f(\theta) \). On the other hand, if \( f(\theta') > f(\theta) \), \( \theta' \) may still be
accepted with probability $\rho \neq 0$. This allows the algorithm to escape the attraction of $\theta^t$ if $\theta^t$ is a local minimum of $f$. This method is in fact the Metropolis algorithm where the target distribution is proportional to $\exp(-f(\theta)/T)$.

**Algorithm 3.1.** The simulated annealing (SA) algorithm

1. Choose an arbitrary initial point $\theta^0$ and $T^0 > 0$
2. At iteration $t$:
   - Sample $\theta'$ from a distribution on the neighbourhood of $\theta^t$
   - Accept $\theta^{t+1} = \theta'$ with probability
     \[
     \rho^t = \min\{\exp(-\Delta f^t/T^t), 1\}
     \]
     where $\Delta f^t = f(\theta') - f(\theta^t)$ and $T^t$ is a parameter called temperature
     otherwise, take $\theta^{t+1} = \theta^t$.
3. Update $T^t$ to $T^{t+1}$
4. Repeat 2-3

It can be proved that there is a necessary and sufficient condition on the rate of decrease of the temperature such that the simulated annealing algorithm Algorithm 3.1 converges to the set of global minima (maxima), Robert and Casella (2004), Theorem 5.7. The difficulty in implementing the SA algorithm is that there is no obvious scheme for the temperature $T$. Annealing schedules (cooling schedules) give the sequence $(T^i)$; the choice of initial temperature, the number of iterations performed at each temperature. An annealing schedule may be developed by trial and error for given problem Kirkpatrick et al. (1983). Several papers have considered practical annealing schedules; see Geman and Geman (1984), Hajek (1988), Bélisle (1992), Stander and Silverman (1994), and Nourani and Andresen (1998).

The following are examples for annealing schedules for the temperature decrement; see Nourani and Andresen (1998).

1. **Logarithmic annealing schedule**:
   \[
   T^i = \frac{c}{\log(i + d)},
   \]
   where $i$ is the step count and $c$ is greater than or equal to the largest energy barrier, $\Delta E$ (or $-\Delta f(\theta)$) in the problem and $d$ is usually set equal to one. This cooling scheme was introduced in Geman and Geman (1984).
2. **Geometric annealing schedule**:

\[ T^i = \alpha^i T^0, \]

where \( \alpha^i \) is a constant close to, but smaller than 1.

3. **Linear annealing schedule**:

\[ T^i = T^0 - \eta i, \]

where \( \eta \) is a constant which describes the amount the temperature is reduced after each iteration.

The major advantage of the SA algorithm over other methods is an ability to avoid becoming trapped in local minima. As a result, it works comparatively well with multimodal cost functions. It can be very computationally expensive as although using small \( T \) increases the probability of acceptance, it can take a very long time to make a chain \( \theta^0, \theta^1, \theta^2, \ldots \), reach the target distribution.

### 3.4 Numerical example

In this section, we examine the performance of the optimisation methods described in Section 3.2 compared to simulated annealing when they are applied to a multimodal cost function. We implement the (minimisation) maximisation algorithms provided in R via the function “optim” in which the default method is an implementation of Nelder and Mead (N-M). The others methods; the BFGS, L-BFGS-B, conjugate gradient (CG) and simulated annealing (SA)(SANN in R) are also implemented via the optim function. We use the default annealing schedule in R which is to decrease according to the logarithmic cooling schedule as given in Bélisle (1992).
3.4.1 The test problem

The cost function used to investigate the performance of the optimisation methods is a mix of probability density functions of the Normal distribution defined by

\[
 f(\theta) = \sum_{i=1}^{n} w_i \frac{1}{\sqrt{2\pi \sigma_i^2}} \exp\left\{-\frac{1}{2\sigma_i^2} \sum_{j=1}^{n} (\theta_j - \mu_i)^2 \right\}, \text{ for } \theta = (\theta_1, \ldots, \theta_n) \in \mathbb{R}^n, \tag{3.10}
\]

where

\[
 w_i = i, \text{ for } i = 1, \ldots, n,
\]

\[
 \mu_i = (i, \ldots, i),
\]

\[
 \sigma_i^2 = \sigma^2, \text{ for } i = 1, \ldots, n.
\]

For example, the mix of 3 probability density functions of the Normal distribution where \( \theta \in \mathbb{R}^3 \), is

\[
 f(\theta) = \frac{1}{\sqrt{2\pi \sigma^2}} \exp\left\{-\frac{1}{2\sigma^2} \sum_{j=1}^{3} (\theta_j - 1)^2 \right\} + 2 \frac{1}{\sqrt{2\pi \sigma^2}} \exp\left\{-\frac{1}{2\sigma^2} \sum_{j=1}^{3} (\theta_j - 2)^2 \right\} + 3 \frac{1}{\sqrt{2\pi \sigma^2}} \exp\left\{-\frac{1}{2\sigma^2} \sum_{j=1}^{3} (\theta_j - 2)^2 \right\}.
\]

We apply the maximisation methods to find the maximum of the cost function (3.10). For each component \( i \) in (3.10) individually, the maximum is the mode, \( \mu_i \), so the overall mix possibly has \( n \) local maxima which are the modes or close to the modes of the Normal densities,

\[
 \mu_1 = (1, \ldots, 1),
\]

\[
 \vdots
\]

\[
 \mu_n = (n, \ldots, n).
\]
Therefore, the function $f$ in Equation (3.10) allows us to examine how optimisation methods perform for a multimodal function. For higher dimensions $n$, the maximum of $\theta$ is more complicated to derive analytically. However, if $\sigma^2$ is small enough, the maximum can possibly be $\theta^* = \mu_n = (n, \ldots, n)$ since there is no merging of the individual curves to make a new mode which is higher than $\mu_n$. For simplicity, we illustrate this by using the mix of 3 probability density functions of the Normal distribution in Equation (3.10) where $\theta$ is in the real line $\mathbb{R}$:

$$f(\theta) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2} (\theta - 1)^2 \right\} + 2 \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2} (\theta - 2)^2 \right\} + 3 \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{ -\frac{1}{2\sigma^2} (\theta - 3)^2 \right\}.$$ (3.11)

The plot of the function in Equation (3.11) is shown in Figure 3.1. We can see that if $\sigma^2$ is small enough, the maximum equals $\mu_3 = 3$ or very close to 3 as shown in the top plots of Figures 3.1. On the other hand, if $\sigma^2$ is larger, the function $f$ can transform to be almost unimodal, then $\mu_3$ is less likely to be the maximum as shown in the bottom plots of Figure 3.1.
The mix of 3 Normal densities with $\sigma^2 = 0.1$

$\theta^* = 3$
$\mu = 3$

The mix of 3 Normal densities with $\sigma^2 = 0.25$

$\theta^* = 2.998$
$\mu = 3$

The mix of 3 Normal densities with $\sigma^2 = 0.5$

$\theta^* = 2.959$
$\mu = 3$

The mix of 3 Normal densities with $\sigma^2 = 1$

$\theta^* = 2.763$
$\mu = 3$

Figure 3.1: Plots of the function $f$ using $\sigma^2 = 0.1$ (top left), $\sigma^2 = 0.25$ (top right), $\sigma^2 = 0.5$ (bottom left) and $\sigma^2 = 1$ (bottom right).
Consider the maximisation problem

\[
\max_{\theta} f(\theta), \quad \theta \in \mathbb{R}^n, n \geq 2,
\]

(3.12)

where \( f(\theta) \) is given by Equation (3.10). Although the simulated annealing algorithm is more likely to be appropriate for finding the global optimum than the others which are designed for finding the local optimum, it is more expensive than the others. We show the computational expensiveness of simulated annealing by using the R function “system.time” to calculate the used time of each method for searching for the maximum of the problem (3.12) in dimensions, \( n = 3, 4 \) and 5 in Table 3.1.

Table 3.1: The comparison of the used time of 100 iterations of the SA method to other deterministic methods while searching for maximum values of the function \( f \) with \( \sigma^2 = 0.25 \).

<table>
<thead>
<tr>
<th>( n )</th>
<th>Method</th>
<th>Used time (seconds)</th>
<th>Ratio of used time to SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>N-M</td>
<td>1.12</td>
<td>0.018</td>
</tr>
<tr>
<td></td>
<td>BFGS</td>
<td>0.69</td>
<td>0.011</td>
</tr>
<tr>
<td></td>
<td>L-BFGS-B</td>
<td>0.18</td>
<td>0.003</td>
</tr>
<tr>
<td></td>
<td>CG</td>
<td>0.56</td>
<td>0.009</td>
</tr>
<tr>
<td></td>
<td>SA</td>
<td>60.42</td>
<td>1.000</td>
</tr>
<tr>
<td>4</td>
<td>N-M</td>
<td>1.85</td>
<td>0.025</td>
</tr>
<tr>
<td></td>
<td>BFGS</td>
<td>0.50</td>
<td>0.007</td>
</tr>
<tr>
<td></td>
<td>L-BFGS-B</td>
<td>0.17</td>
<td>0.002</td>
</tr>
<tr>
<td></td>
<td>CG</td>
<td>0.38</td>
<td>0.005</td>
</tr>
<tr>
<td></td>
<td>SA</td>
<td>74.53</td>
<td>1.000</td>
</tr>
<tr>
<td>5</td>
<td>N-M</td>
<td>2.15</td>
<td>0.026</td>
</tr>
<tr>
<td></td>
<td>BFGS</td>
<td>0.49</td>
<td>0.006</td>
</tr>
<tr>
<td></td>
<td>L-BFGS-B</td>
<td>0.14</td>
<td>0.002</td>
</tr>
<tr>
<td></td>
<td>CG</td>
<td>0.28</td>
<td>0.003</td>
</tr>
<tr>
<td></td>
<td>SA</td>
<td>81.86</td>
<td>1.000</td>
</tr>
</tbody>
</table>

The SA method spends more time searching for the maximum than other methods do, in particular for higher dimensions. However, it tends to provide the global maximum while the others are often trapped in the local maxima. To illustrate this, we examine how often that the SA method provides the global maximum compared to the other methods.
3.4.2 The performance of the SA method for the multimodal problem

Loosely speaking, in $n$ dimensions, if $\sigma^2$ is small, the cost function $f(\theta)$ in Equation (3.11) has

$$\mu_n = (n, \ldots, n)$$

as the global maximum and $n - 1$ potential local maxima,

$$\mu_1 = (1, \ldots, 1),$$

$$\vdots$$

$$\mu_{n-1} = (n-1, \ldots, n-1).$$

For example, when $n = 3$, the global maximum is $(3,3,3)$ and the local maxima could be $(1,1,1)$ and $(2,2,2)$. We regard the value $\mu_n = (n, \ldots, n) \in \mathbb{R}^n$ as the exact solution of the problem (3.12). If the numerical result of the maximum $\theta^* = (\theta_1^*, \ldots, \theta_n^*)$ from each method is close enough to the exact solution $\mu_n = (n, \ldots, n)$, we say that the method can reach the global maximum. Here, the term “close enough” means that the distance between the maximum $\theta^* = (\theta_1^*, \ldots, \theta_n^*)$ and $\mu_n = (n, \ldots, n)$ is less than the distance between the maximum $\theta^* = (\theta_1^*, \ldots, \theta_n^*)$ and the other potential local maxima; $\mu_1 = (1, \ldots, 1), \ldots, \mu_{n-1} = (n-1, \ldots, n-1) \in \mathbb{R}^n$. We use the Euclidean distance between the maximum $\theta^*$ and $\mu_j \in \mathbb{R}^n$, given by

$$\|\theta^* - \mu_j\| = d_j = \sqrt{(\theta_1^* - j)^2 + \cdots + (\theta_n^* - j)^2}, \quad j = 1, \ldots, n, \quad (3.13)$$

to measure the distance between the maximum value $\theta^* = (\theta_1^*, \ldots, \theta_n^*)$ and potential local maxima including the global maximum. We find the shortest distance;

$$d = \min\{d_1, \ldots, d_n\}$$

If $d = d_n$, we can say that the maximum $\theta^* = (\theta_1^*, \ldots, \theta_n^*)$ is global otherwise the maximum $\theta^* = (\theta_1^*, \ldots, \theta_n^*)$ is local.
We examine the ability to reach the global maximum for each method by using the same starting points \( \theta_0 = (\theta_1, \ldots, \theta_n) \in \mathbb{R}^n \) for each maximisation method, and we then compute the distance \( d_j \) in Equation (3.13). We then check whether the maximum \( \theta^* = (\theta_1^*, \ldots, \theta_n^*) \) from each method is the global maximum by checking the distance \( d \). The maximum \( \theta^* = (\theta_1^*, \ldots, \theta_n^*) \) often depends on the starting point \( \theta_0 \). We cannot guarantee that the method can provide the global maximum if the starting point is already close to the global maximum \( \mu_n = (n, \ldots, n) \). Nonetheless, if the method uses a starting point which is close to one of the local maxima and yet still gives the global maximum, we are more convinced that such a method is more capable of establishing the global maximum. This implies that no matter what the value of the starting point \( \theta_0 \) is, a good method should be able to reach the global maximum (and not be trapped in the local maxima). We use 100 different starting points generated randomly from the uniform distribution as follows:

\[
\theta_0 = (\theta_1, \ldots, \theta_n),
\]

where \( \theta_i \sim U(-1, n + 2), \quad i = 1, \ldots, n. \)

We use the same values of the starting points for all maximisation methods to compute the maximum and then find the distance \( d \). The results from 100 trials that the shortest distance \( d \) is \( d_n \) are shown in the next subsection.
3.4.3 Results and discussion

Table 3.2: Numerical results of the maximisation problem using $\sigma^2 = 0.1$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n + 2)$, $i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>$n = 3$</th>
<th>$n = 4$</th>
<th>$n = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\theta_0$ giving $d = d_j$</td>
<td>no.of iterations</td>
<td>no.of iterations</td>
</tr>
<tr>
<td></td>
<td>$\theta_0$ giving $d = d_j$</td>
<td>for which $\theta^*$ gives $d = d_j$</td>
<td>for which $\theta^*$ gives $d = d_j$</td>
</tr>
<tr>
<td>N-M</td>
<td>39</td>
<td>31</td>
<td>30</td>
</tr>
<tr>
<td>BFGS</td>
<td>39</td>
<td>31</td>
<td>30</td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>39</td>
<td>31</td>
<td>30</td>
</tr>
<tr>
<td>CG</td>
<td>39</td>
<td>31</td>
<td>30</td>
</tr>
<tr>
<td>SA</td>
<td>35</td>
<td>25</td>
<td>40</td>
</tr>
</tbody>
</table>

$n = 3$

<table>
<thead>
<tr>
<th>Methods</th>
<th>no.of iterations for which $\theta^*$ gives $d = d_j$</th>
<th>$j = 1$</th>
<th>$j = 2$</th>
<th>$j = 3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-M</td>
<td>37</td>
<td>29</td>
<td>34</td>
<td></td>
</tr>
<tr>
<td>BFGS</td>
<td>39</td>
<td>31</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>39</td>
<td>31</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>CG</td>
<td>39</td>
<td>31</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>SA</td>
<td>35</td>
<td>25</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>

$n = 4$

<table>
<thead>
<tr>
<th>Methods</th>
<th>no.of iterations for which $\theta^*$ gives $d = d_j$</th>
<th>$j = 1$</th>
<th>$j = 2$</th>
<th>$j = 3$</th>
<th>$j = 4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-M</td>
<td>11</td>
<td>32</td>
<td>40</td>
<td>17</td>
<td></td>
</tr>
<tr>
<td>BFGS</td>
<td>15</td>
<td>37</td>
<td>33</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>15</td>
<td>37</td>
<td>33</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>CG</td>
<td>15</td>
<td>37</td>
<td>33</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td>SA</td>
<td>21</td>
<td>33</td>
<td>24</td>
<td>22</td>
<td></td>
</tr>
</tbody>
</table>

$n = 5$

<table>
<thead>
<tr>
<th>Methods</th>
<th>no.of iterations for which $\theta^*$ gives $d = d_j$</th>
<th>$j = 1$</th>
<th>$j = 2$</th>
<th>$j = 3$</th>
<th>$j = 4$</th>
<th>$j = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-M</td>
<td>5</td>
<td>23</td>
<td>31</td>
<td>32</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>BFGS</td>
<td>5</td>
<td>25</td>
<td>34</td>
<td>29</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>5</td>
<td>25</td>
<td>34</td>
<td>29</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>CG</td>
<td>5</td>
<td>25</td>
<td>34</td>
<td>29</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>SA</td>
<td>13</td>
<td>24</td>
<td>29</td>
<td>17</td>
<td>17</td>
<td></td>
</tr>
</tbody>
</table>

no.of iterations of $\theta_0$ giving $d = d_j$
Table 3.3: Numerical results of the maximisation problem using $\sigma^2 = 0.25$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n+2), i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>no. of iterations for which $\theta^*$ gives $d = d_j$</th>
<th>( j = 1 )</th>
<th>( j = 2 )</th>
<th>( j = 3 )</th>
<th>( j = 4 )</th>
<th>( j = 5 )</th>
<th>( j = 6 )</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N-M</td>
<td>42</td>
<td>35</td>
<td>23</td>
<td>23</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>BFGS</td>
<td>38</td>
<td>37</td>
<td>25</td>
<td>25</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>35</td>
<td>41</td>
<td>24</td>
<td>24</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>CG</td>
<td>38</td>
<td>35</td>
<td>22</td>
<td>22</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>SA</td>
<td>31</td>
<td>31</td>
<td>38</td>
<td>38</td>
<td>14</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>no. of iterations of $\theta_0$ giving $d = d_j$</td>
<td>40</td>
<td>36</td>
<td>24</td>
<td>24</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N-M</td>
<td>12</td>
<td>25</td>
<td>44</td>
<td>44</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>BFGS</td>
<td>15</td>
<td>26</td>
<td>42</td>
<td>42</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>15</td>
<td>26</td>
<td>42</td>
<td>42</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>CG</td>
<td>15</td>
<td>26</td>
<td>42</td>
<td>42</td>
<td>17</td>
<td>17</td>
<td>17</td>
</tr>
<tr>
<td>SA</td>
<td>16</td>
<td>28</td>
<td>25</td>
<td>25</td>
<td>31</td>
<td>31</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>no. of iterations of $\theta_0$ giving $d = d_j$</td>
<td>15</td>
<td>26</td>
<td>42</td>
<td>42</td>
<td>42</td>
<td>42</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>N-M</td>
<td>7</td>
<td>24</td>
<td>40</td>
<td>40</td>
<td>23</td>
<td>23</td>
<td>23</td>
</tr>
<tr>
<td>BFGS</td>
<td>7</td>
<td>29</td>
<td>38</td>
<td>38</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>7</td>
<td>29</td>
<td>38</td>
<td>38</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>CG</td>
<td>7</td>
<td>29</td>
<td>38</td>
<td>38</td>
<td>19</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>SA</td>
<td>14</td>
<td>22</td>
<td>34</td>
<td>34</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>no. of iterations of $\theta_0$ giving $d = d_j$</td>
<td>7</td>
<td>29</td>
<td>38</td>
<td>38</td>
<td>38</td>
<td>38</td>
</tr>
</tbody>
</table>
Table 3.4: Numerical results of the maximisation problem using $\sigma^2 = 0.5$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n+2), i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.

$$\begin{array}{|c|c|c|c|}
\hline
\text{Methods} & j = 1 & j = 2 & j = 3 \\
\hline
\text{N-M} & 0 & 32 & 68 \\
\text{BFGS} & 8 & 36 & 56 \\
\text{L-BFGS-B} & 1 & 29 & 70 \\
\text{CG} & 5 & 39 & 56 \\
\text{SA} & 2 & 2 & 96 \\
\hline
\text{no.of iterations of } \theta_0 \text{ giving } d = d_j & 39 & 24 & 37 \\
\hline
\end{array}$$

$$\begin{array}{|c|c|c|c|c|}
\hline
\text{Methods} & j = 1 & j = 2 & j = 3 & j = 4 \\
\hline
\text{N-M} & 10 & 35 & 35 & 20 \\
\text{BFGS} & 13 & 35 & 32 & 20 \\
\text{L-BFGS-B} & 12 & 35 & 32 & 21 \\
\text{CG} & 13 & 35 & 30 & 22 \\
\text{SA} & 15 & 29 & 28 & 28 \\
\hline
\text{no.of iterations of } \theta_0 \text{ giving } d = d_j & 13 & 35 & 31 & 21 \\
\hline
\end{array}$$

$$\begin{array}{|c|c|c|c|c|c|}
\hline
\text{Methods} & j = 1 & j = 2 & j = 3 & j = 4 & j = 5 \\
\hline
\text{N-M} & 3 & 27 & 39 & 19 & 12 \\
\text{BFGS} & 4 & 27 & 41 & 19 & 9 \\
\text{L-BFGS-B} & 4 & 27 & 41 & 19 & 9 \\
\text{CG} & 4 & 27 & 41 & 19 & 9 \\
\text{SA} & 14 & 22 & 35 & 16 & 13 \\
\hline
\text{no.of iterations of } \theta_0 \text{ giving } d = d_j & 4 & 27 & 41 & 19 & 9 \\
\hline
\end{array}$$
Table 3.5: Numerical results of the maximisation problem using $\sigma^2 = 1$ with 100 different starting points, $\theta_0 = (\theta_1, \ldots, \theta_n)$ where $\theta_i \sim U(-1, n+2)$, $i = 1, \ldots, n$, for $n = 3, 4$ and 5, respectively.

<table>
<thead>
<tr>
<th>Methods</th>
<th>$n = 3$</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$j = 1$</td>
<td>$j = 2$</td>
<td>$j = 3$</td>
<td></td>
</tr>
<tr>
<td>N-M</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BFGS</td>
<td>1</td>
<td>1</td>
<td>98</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CG</td>
<td>1</td>
<td>0</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td></td>
<td></td>
</tr>
<tr>
<td>no.of iterations of $\theta_0$ giving $d = d_j$</td>
<td>28</td>
<td>30</td>
<td>42</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>$n = 4$</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$j = 1$</td>
<td>$j = 2$</td>
<td>$j = 3$</td>
<td>$j = 4$</td>
</tr>
<tr>
<td>N-M</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>BFGS</td>
<td>1</td>
<td>14</td>
<td>12</td>
<td>73</td>
<td></td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>0</td>
<td>7</td>
<td>7</td>
<td>86</td>
<td></td>
</tr>
<tr>
<td>CG</td>
<td>1</td>
<td>11</td>
<td>12</td>
<td>76</td>
<td></td>
</tr>
<tr>
<td>SA</td>
<td>2</td>
<td>0</td>
<td>7</td>
<td>91</td>
<td></td>
</tr>
<tr>
<td>no.of iterations of $\theta_0$ giving $d = d_j$</td>
<td>18</td>
<td>36</td>
<td>33</td>
<td>13</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Methods</th>
<th>$n = 5$</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$j = 1$</td>
<td>$j = 2$</td>
<td>$j = 3$</td>
<td>$j = 4$</td>
<td>$j = 5$</td>
<td></td>
</tr>
<tr>
<td>N-M</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>66</td>
<td>33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BFGS</td>
<td>6</td>
<td>13</td>
<td>32</td>
<td>28</td>
<td>21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L-BFGS-B</td>
<td>3</td>
<td>10</td>
<td>23</td>
<td>23</td>
<td>41</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CG</td>
<td>6</td>
<td>13</td>
<td>31</td>
<td>27</td>
<td>23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA</td>
<td>4</td>
<td>4</td>
<td>12</td>
<td>31</td>
<td>49</td>
<td></td>
<td></td>
</tr>
<tr>
<td>no.of iterations of $\theta_0$ giving $d = d_j$</td>
<td>11</td>
<td>23</td>
<td>40</td>
<td>18</td>
<td>8</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Tables 3.2 - 3.3 present the results for the cases when the variances are small \( \sigma^2 = 0.1 \) and 0.25, respectively. They show that the simulated annealing (SA) method is better than the other methods because it gives more trials of the maximum value \( \theta^* \) giving the shortest distance \( d = d_n \) for \( n = 3, 4 \) and 5. This implies that there are more trails from the SA method give the global maxima which are \((3,3,3), (4,4,4,4)\) and \((5,5,5,5,5)\) for \( n = 3, 4 \) and 5, respectively. In fact, the number of trails of the maximum \( \theta^* \) near the global maximum provided by the BFGS, L-BFGS-B and CG methods is identical to the number of trials of the starting point \( \theta_0 \). This means that the maximum values \( \theta^* \) do not move far from the starting point values. The Nelder-Mead (N-M) method seems to be slightly better than the BFGS, L-BFGS-B and CG methods as it provides slightly higher number of trials of the maximum \( \theta^* \) which is close to the global maximum. Overall, the SA method is better than the others as it is less likely to be trapped in the local maxima.

Tables 3.4 - 3.5 present the results for the cases when the variances are relatively large. In these cases, local maxima can merge, so there is no obvious local maxima. Therefore, the cost function with \( \sigma^2 = 0.5 \) and 1 can be thought of the unimodal cost function where the global maximum is not at the exact modes; \((3,3,3), (4,4,4,4)\) and \((5,5,5,5,5)\) anymore. However, loosely speaking, the global maximum is located near the modes (see the example in Figure 3.1). For \( \sigma^2 = 0.5 \) in Table 3.4, all methods work comparatively well as we obtain more trails of the maximum \( \theta^* \) near the modes \((3,3,3)\) compared to the smaller values of \( \sigma^2 \). For \( n = 4 \) and 5 they seem to perform worse than for \( n = 3 \). When the cost function becomes more clearly unimodal with \( \sigma^2 = 1 \), all methods work relatively well, as shown in Table 3.5. In fact, all the methods work very well for \( n = 3 \) and become slightly worse for the higher dimensions, \( n = 4 \) and 5. From our example of the maximisation problem, the SA method works comparatively well in any circumstance.

In summary, when the cost function becomes unimodal, the simulated annealing method has made no difference compared to other methods. Therefore, in this case it is better to use other methods since they are faster than the simulated annealing. The advantage of the simulated annealing method is that it is more likely to overcome local maxima while the other methods are more likely to be trapped in local maxima, especially in higher dimensions.
3.5 Conclusion

The simulated annealing method is likely to be more expensive than the other deterministic methods. However, it is more likely to establish the global maximum (or minimum) in a multimodal optimisation problem. As a result, in this thesis, we use simulated annealing as the minimisation method searching for the simulation-based Bayes estimates under some non-standard loss function, while the deterministic methods will still have a role when it comes to making the uncertainty assessment using the jackknife idea in Chapter 7, as efficient as possible.
Chapter 4

The ETM Method for Bayes Estimates in One Dimension

4.1 Introduction

In Chapter 2, we have shown that there are some common loss functions for which the Bayes estimates can be derived analytically. However, there are several different possible loss functions suitable for different contexts. The choice of which loss functions to use for a given inference problem depends on the decision maker and the nature of the decision problems. For example, we might use the integrated squared difference loss function in order to avoid a label switching problem which will be described in Chapter 6 or we might use an asymmetric loss function such as the linex loss function for reliability analysis, Zellner (1986). Bayes estimates under some non-standard loss functions might not be available analytically. We could then find simulation-based Bayes estimates by approximating the expected loss function using a generated sample \( \theta \) and applying a minimisation method to search for \( \hat{\theta}^{*} \) which minimises the approximated expected loss function. This means we obtain the Bayes estimate from

\[
\hat{\theta}^{*} = \arg \min_{\theta} \mathbb{E}_{\pi} L(\hat{\theta}, \theta),
\]  

(4.1)

where the expected loss function \( \mathbb{E}_{\pi} L(\hat{\theta}, \theta) \) is estimated by

\[
\mathbb{E}_{\pi}[L(\hat{\theta}, \theta)] \approx \frac{1}{N} \sum_{j=1}^{N} L(\hat{\theta}, \theta^{j}),
\]  

(4.2)
according to either Monte Carlo integration or ergodic averaging of MCMC samples. As a result, computing the Bayes estimate could be computationally expensive because it involves two computational tasks; estimation and minimisation which are potentially computationally expensive. The term “computationally expensive” means that it takes long time to obtain $\hat{\theta}^*$. The estimation part which involves the approximation of $E_\pi[L(\hat{\theta}, \theta)]$ usually requires large $N$, so it can be computationally expensive. Moreover, if the loss function $L(\hat{\theta}, \theta)$ is complex, the estimation of the expected loss function is more expensive. In the meantime, the minimisation method which is used to search for $\hat{\theta}^*$ can also be computationally expensive, such as simulated annealing. More importantly, we might have to deal with both estimation and minimisation simultaneously because we estimate the expected loss function at $\hat{\theta}$ which is the candidate of $\hat{\theta}^*$ provided by the minimisation method. If we can separate the two computational tasks, we could compute the Bayes estimate $\hat{\theta}^*$ relatively quickly.

We present the “estimation-then-minimisation” (ETM) method computing Bayes estimates. To be able to apply the ETM method, we need to express a loss function in a decomposed form of the parameters $\hat{\theta}$ and $\theta$. Thus we separate the estimation from the minimisation as the expected loss function is expressed as the function of $\hat{\theta}$ and $E_\pi[h(\theta)]$ for some function $h$, for example the quadratic loss function (see Equation (1.4)). However, not every loss function can be expressed as easily in a decomposed form and hence we cannot apply the ETM method.

In this thesis, we want to extend the use of the ETM method for computing Bayes estimates under some non-standard loss functions, in particular to those loss functions in which the decomposed forms are not feasible. To do so, we approximate those loss functions in order to express them in forms of polynomials which the ETM method can be implemented. The idea of approximating a loss function for computing Bayes estimates is the new idea. However, the approximation of loss functions has been studied in other circumstances. For example, Christoffersen and Diebold (1996) used piecewise-linear approximation to the loss function for solving prediction problems. They shown that when the optimal predictor does not exist in a closed form, there is a different and complementary approach to find the optimal predictor. Instead of approximating the optimal predictor for the exact loss function, we could compute the exactly optimal predictor for an approximate loss function. In Singh et al. (2008), the linex loss function which was introduced by Varian (1975) is approximated by Lindley’s approximation technique, Lindley (1980) for estimating a Bayes estimator of the parameter of Generalized-Exponential distribution.
In this chapter, we present two well known methods for function approximation; Taylor series approximation and cubic spline interpolation. We want to apply the ETM method to compute the simulation-based Bayes estimates by using these two approximation representations of loss functions and show that we can obtain Bayes estimates as good as we can get from using the true representation in which the ETM could not be implemented.

4.2 The ETM method via Taylor series approximation

4.2.1 Taylor series approximation for loss function approximation

Mathematically speaking, a Taylor series is a representation of a function in the form of an infinite sum of terms that are calculated from the function’s derivatives at a single point. The Taylor series of a real or complex-valued function \( f(x) \) can be written as

\[
    f(x) = \sum_{n=1}^{\infty} \frac{f^{(n)}(a)}{n!} (x - a)^n,
\]

where \( f^{(n)}(a) \) denotes the \( n \)th derivative of \( f \) evaluated at the point \( a \). Note that if \( a = 0 \), the series is called a Maclaurin series. The \( n \)th partial sum of the Taylor series is the \( n \)th-degree Taylor polynomial of \( f \) at \( a \):

\[
    T_n(x) = f(a) + \frac{f'(a)}{1!} (x - a) + \frac{f''(a)}{2!} (x - a)^2 + \ldots + \frac{f^{(n)}(a)}{n!} (x - a)^n.
\]

To approximate a loss function \( L(\hat{\theta}, \theta) \) using Taylor series approximation, we consider the Taylor series approximation centred at \( \hat{\theta} \) as follows.

\[
    T_{n,\hat{\theta}}(\theta) = L(\hat{\theta}, \hat{\theta}) + \frac{L'(\hat{\theta}, \hat{\theta})}{1!} (\theta - \hat{\theta}) + \frac{L''(\hat{\theta}, \hat{\theta})}{2!} (\theta - \hat{\theta})^2 + \frac{L'''(\hat{\theta}, \hat{\theta})}{3!} (\theta - \hat{\theta})^3 + \ldots + \frac{L^{(n)}(\hat{\theta}, \hat{\theta})}{n!} (\theta - \hat{\theta})^n
\]

where \( L^{(n)}(\hat{\theta}) \) denotes the \( n \)th derivative with respect to \( \theta \) of \( L(\hat{\theta}, \theta) \) evaluated at the point \( \hat{\theta} \). Clearly, if \( \hat{\theta} = \theta \), the Taylor approximation of the loss function, \( T_{n,\hat{\theta}}(\theta) \) is zero (that is an important property of a loss function). For example, the quadratic loss function,

\[
    L(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2,
\]
where the derivatives with respect to \( \theta \) of order \( n \) are

\[
L'(\hat{\theta}, \theta) = -2(\hat{\theta} - \theta) \\
L''(\hat{\theta}, \theta) = 2 \\
L^{(n)}(\hat{\theta}, \theta) = 0 \quad \text{for } n = 3, 4, \ldots ,
\]

so that

\[
L'(\hat{\theta}, \hat{\theta}) = 0 \\
L''(\hat{\theta}, \hat{\theta}) = 2 \\
L^{(n)}(\hat{\theta}, \hat{\theta}) = 0 \quad \text{for } n = 3, 4, \ldots
\]

Therefore, the Taylor series approximation of the quadratic loss function is

\[
T_{2,\hat{\theta}}(\theta) = \frac{2}{2!}(\theta - \hat{\theta})^2 = (\hat{\theta} - \theta)^2,
\]

which is exact for the quadratic loss function.

For convenience, define coefficients of Taylor series \( \frac{L^{(i)}(\hat{\theta}, \hat{\theta})}{i!} = C_i^\hat{\theta} \) for \( i = 1, \ldots , n \). Therefore, the \( n \) terms of the Taylor series approximation of the loss function \( L(\hat{\theta}, \theta) \) is given by

\[
T_{n,\hat{\theta}}(\theta) = C_1^\hat{\theta}(\theta - \hat{\theta}) + C_2^\hat{\theta}(\theta - \hat{\theta})^2 + C_3^\hat{\theta}(\theta - \hat{\theta})^3 + \ldots + C_n^\hat{\theta}(\theta - \hat{\theta})^n.
\]

### 4.2.2 The ETM method

If the Taylor series approximation in Equation (4.3) approximates the loss function \( L(\hat{\theta}, \theta) \) well, we might be able to compute the Bayes estimate

\[
\hat{\theta}^T = \arg \min_\theta \mathbb{E}_\pi [T_{n,\hat{\theta}}(\theta)].
\]

We want to obtain the Bayes estimate \( \hat{\theta}^T \) as good as we could obtain from using the true loss function in Equation (4.1). More importantly, we can implement the ETM method to compute the Bayes estimate via the \( n \) terms of Taylor series approximation. We describe how to implement the ETM method as follows.
Consider

\[ E_\pi[T_{n,\hat{\theta}}(\theta)] = E_\pi[C_1^\theta(\theta - \hat{\theta}) + C_2^\theta(\theta - \hat{\theta})^2 + C_3^\theta E_\pi(\theta - \hat{\theta})^3 + \ldots + C_n^\theta(\theta - \hat{\theta})^n] \]

\[ = C_1^\theta E_\pi[(\theta - \hat{\theta})] + C_2^\theta E_\pi[(\theta - \hat{\theta})^2] + C_3^\theta E_\pi(\theta - \hat{\theta})^3 + \ldots + C_n^\theta E_\pi[(\theta - \hat{\theta})^n] \]

\[ = \left( C_1^\theta E_\pi[\theta] + C_2^\theta E_\pi[\theta^2] + C_3^\theta E_\pi[\theta^3] + \ldots + C_n^\theta E_\pi[\theta^n] \right) - \left( \frac{1}{1} C_1^\theta + \left( \frac{2}{1} C_2^\theta E_\pi[\theta] + \left( \frac{3}{1} C_3^\theta E_\pi[\theta^2] + \ldots + \left( \frac{n}{1} C_n^\theta E_\pi[\theta^{n-1}] \right) \right) \hat{\theta} \right) \]

\[ + \left( \frac{2}{2} C_2^\theta + \frac{3}{2} C_3^\theta E_\pi[\theta] + \ldots + \left( \frac{n}{2} C_n^\theta E_\pi[\theta^{n-2}] \right) \hat{\theta}^2 \right) \]

\[ - \left( \frac{3}{3} C_3^\theta + \frac{4}{3} C_4^\theta E_\pi[\theta] + \ldots + \left( \frac{n}{3} C_n^\theta E_\pi[\theta^{n-3}] \right) \hat{\theta}^3 \right) \]

\[ + \ldots \]

\[ + (-1)^{n-1} \left( \frac{n-1}{n-1} C_n^{\theta - 1} + \frac{n}{n-1} C_n^\theta E_\pi[\theta] \right) \hat{\theta}^{n-1} \]

\[ + (-1)^n C_n^\theta \hat{\theta}^n. \quad (4.4) \]

The ETM method using Taylor series approximation

Suppose we have \( N \) generated samples of the parameter \( \theta \) from the posterior distribution \( \pi \).

1. Estimation part

   (a) Estimate \( E_\pi[\theta^i] \) for \( i = 1, 2, \ldots, n \) by

   \[ E_\pi[\theta^i] \approx \frac{1}{N} \sum_{j=1}^{N} \theta_j^i \quad \text{for } i = 1, 2, \ldots, n. \]

2. Minimisation part

   (a) At iteration \( t \), evaluate \( C_i^{\hat{\theta}(t)} \) for \( i = 1, 2, \ldots, n \).

   (b) Get \( E_{\pi}[\theta^i] \) for \( i = 1, 2, \ldots, n \) from (1a) and \( C_i^{\hat{\theta}} \) for \( i = 1, 2, \ldots, n \) from (2a) to approximate \( E_{\pi}[T_{n,\hat{\theta}(t)}(\theta)] \) corresponding to the candidate value \( \hat{\theta}(t) \).

   (c) Repeat (2a) - (2b), to obtain \( \hat{\theta}_T \) which gives the minimum \( E_{\pi}[T_{n,\hat{\theta}}(\theta)] \).
In the next subsections, we will compare the Bayes estimate obtained from using the true loss functions to ones from using the Taylor series approximation. We refer to the computational method to compute the Bayes estimate using the true loss function in Equation (4.1) as the naive method.

4.2.3 Taylor series approximation for the linex loss function

In some decision problems, the use of symmetric loss functions might be inappropriate because at the same magnitude of error, a given positive error might be more serious than a given negative error or vice versa. For instance, in dam construction, an underestimate of the peak water level is usually much more serious than an overestimate. Therefore, for those cases using an asymmetric loss function is more desirable. The linex loss function is given by

\[ L(\hat{\theta}, \theta) = e^{c(\hat{\theta} - \theta)} - e^{c(\theta - \theta)} - 1, \quad c \in \mathbb{R}. \]  

(4.5)

The linex loss function was introduced in Varian (1975) and studied by several authors, such as Zellner (1986), Christoffersen and Diebold (1997) and Singh et al. (2005). For \( c > 0 \), it penalises an error almost exponentially for a positive error and almost linearly for a negative error (or vice versa for \( c < 0 \)) as shown in Figure 4.1.
We consider the linex loss function (4.5) with $c = 1$ so that

$$L(\hat{\theta}, \theta) = e^{(\hat{\theta} - \theta)} - (\hat{\theta} - \theta) - 1.$$  

(4.6)

By using Taylor series approximation of $f(x) = e^x$,

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} \ldots \text{ for all } x \in \mathbb{R}.$$

Then we have

$$e^{(\hat{\theta} - \theta)} = 1 + (\hat{\theta} - \theta) + \frac{(\hat{\theta} - \theta)^2}{2!} + \frac{(\hat{\theta} - \theta)^3}{3!} + \ldots,$$

and

$$L(\hat{\theta}, \theta) = e^{(\hat{\theta} - \theta)} - (\hat{\theta} - \theta) - 1 = \frac{(\hat{\theta} - \theta)^2}{2!} + \frac{(\hat{\theta} - \theta)^3}{3!} + \ldots.$$

Therefore, the $n$ terms of the Taylor series approximation for the linex loss function is

$$T_{n,\hat{\theta}}(\theta) = \frac{(\hat{\theta} - \theta)^2}{2!} + \frac{(\hat{\theta} - \theta)^3}{3!} + \ldots + \frac{(\hat{\theta} - \theta)^n}{n!} + \frac{(\hat{\theta} - \theta)^{n+1}}{(n+1)!}$$

$$= C_2(\hat{\theta} - \theta)^2 + C_3(\hat{\theta} - \theta)^3 + \ldots + C_n(\hat{\theta} - \theta)^n + C_{n+1}(\hat{\theta} - \theta)^{n+1}.$$  

(4.7)
Note that we use $C_i$ instead of $C_i^\theta$ in the Taylor series approximation (4.7) because in this case $C_i^\theta = \frac{1}{n}$ for all $i = 1, \ldots, n + 1$ which are independent from $\hat{\theta}$.

Choosing different values of $n$ in the Taylor series approximation in Equation (4.7) to approximate the linex loss function in Equation (4.6), the approximations are shown in Figure 4.2.

The linex loss function where $c = 1$ and the Taylor series approximation

![Plot of the linex loss function and Taylor series approximations](image)

Figure 4.2: Plot of the linex loss function and Taylor series approximations, $T_2 = T_{2,\hat{\theta}}(\theta), T_4 = T_{4,\hat{\theta}}(\theta), T_7 = T_{7,\hat{\theta}}(\theta)$ and $T_9 = T_{9,\hat{\theta}}(\theta)$. 
The Taylor series approximation in Equation (4.7) approximates the linex loss function relatively well, in particular on the exponential curve \((\hat{\theta} - \theta > 0)\). Unsurprisingly, the Taylor series approximation with terms of higher order approximates the linex loss function better than the lower order, for example \(T_{9,\hat{\theta}}(\theta)\) is the best compared to the others. Moreover, when \(n = 2\) and \(4\) which are the Taylor series of orders 3rd and 5th degrees (\(T_2\) and \(T_4\) in Figure 4.2), we can have negative function values at some point on the linear curve \((\hat{\theta} - \theta < 0)\) which is clearly undesirable.

For convenience, we use the linex Bayes estimate to refer the Bayes estimate under the linex loss function in Equation (4.6). We denote the linex Bayes estimate according to the method we use. We use \(\hat{\theta}^{LN}\) to denote the linex Bayes estimate obtained from the naive method, and use \(\hat{\theta}^{LT_n}\) to denote the linex Bayes estimate obtained from the ETM method using the \(n\)-terms Taylor series approximation.

The naive method:

\[
\hat{\theta}^{LN} = \arg \min_{\hat{\theta}} \mathbb{E}_\pi [e^{(\hat{\theta} - \theta)} - (\hat{\theta} - \theta) - 1] \tag{4.8}
\]

The ETM method:

\[
\hat{\theta}^{LT_n} = \arg \min_{\hat{\theta}} \mathbb{E}_\pi [T_{n,\hat{\theta}}(\theta)] \tag{4.9}
\]

where \(T_{n,\hat{\theta}}(\theta)\) is in Equation (4.7).

Before showing computational results of the linex Bayes estimate, we will show that the linex Bayes estimate can be derived analytically as follows.

Consider

\[
\mathbb{E}_\pi [L(\hat{\theta}, \theta)] = e^{c\hat{\theta}} \mathbb{E}_\pi [e^{-c\theta}] - c\hat{\theta} + c\mathbb{E}_\pi [\theta] - 1. \tag{4.10}
\]

Differentiate (4.10) with respect to \(\hat{\theta}\),

\[
\frac{d\mathbb{E}_\pi [L(\hat{\theta}, \theta)]}{d\hat{\theta}} = c \left( e^{c\hat{\theta}} \mathbb{E}_\pi [e^{-c\theta}] - 1 \right). \tag{4.11}
\]

Then equate (4.11) to zero and solve for \(\hat{\theta}\) as follows:

\[
e^{c\hat{\theta}} \mathbb{E}_\pi [e^{-c\theta}] - 1 = 0
\]

\[
\hat{\theta} = -(1/c) \log(\mathbb{E}_\pi [e^{-c\theta}]). \tag{4.12}
\]
Checking the minimum value, we find the second derivative of (4.10)

\[ \frac{d^2 E_\pi[L(\hat{\theta}, \theta)]}{d\hat{\theta}^2} = c^2 e^{-c\hat{\theta}} E_\pi[e^{-c\theta}] > 0, \quad \text{for all } \hat{\theta} \in \Omega. \]

Therefore, the value of \( \hat{\theta} \) that minimises the expected loss (4.10) is \(-\frac{1}{c} \log(E_\pi[e^{-c\theta}])\) provided that \( E_\pi[e^{-c\theta}] \) exists and is finite, Zellner (1986). As we consider the case when \( c = 1 \), the analytical linex Bayes estimate is \(-\log(E_\pi[e^{-\theta}])\).

**Simulation results of linex Bayes estimates**

Consider the Normal posterior distribution using the Normal-Normal conjugate model.

**Model N1:**

\[
\begin{align*}
  x_i | \theta &\sim \text{i.i.d. } N(10, 100), \quad i = 1, \ldots, 100, \\
  \theta &\sim N(0, 100), \\
  \theta | x &\sim N(10.97908, 0.990099). 
\end{align*}
\]

(4.13)

Using the moment generating function for a Normal distribution (see Equation (A.1) in Appendix), the analytical linex Bayes estimate with respect to the Normal posterior distribution is,

\[
\hat{\theta}^{LB} = -\log(E_\pi[e^{-\theta}]) = -\log(e^{-10.97908 + \frac{0.990099}{2}}) = 10.48403. \quad (4.14)
\]

Moreover, we can also compute \(-\log(E_\pi[e^{-\theta}])\) using the sample average of a sample \( \theta \) generated independently from the posterior distribution \( \pi \) according to the Monte Carlo method,

\[
\hat{\theta}^{LM} = -\log\left(\frac{1}{100,000} \sum_{j=1}^{100,000} e^{-\theta_j}\right) = 10.47743 \quad \text{where } \theta_j's \text{ are generated from (4.13)}. \quad (4.15)
\]

We implement the naive method using the linex loss function in Equation (4.8) and the ETM method via Taylor series approximation in Equation (4.9) with respect to the Normal posterior distribution, Model N1 to compute the linex Bayes estimates and the results are shown in Table 4.1.
Table 4.1: The comparison of the linex Bayes estimates $\hat{\theta}^{LN}$ and $\hat{\theta}^{LT}$ with respect to Model N1 with the sample size $N = 100,000$.

| The naive method $\hat{\theta}^{LN}$ | The ETM method $\hat{\theta}^{LT}$ | The absolute errors $|\hat{\theta}^{LN} - \hat{\theta}^{LT}|$ |
|-------------------------------------|-----------------------------------|----------------------------------|
| 10.4774                             | 10.9768                           | 0.4994                           |
| 1                                  | -501.620                          | 512.10                           |
| 2                                  | 10.6025                           | 0.1251                           |
| 3                                  | -499.839                          | 510.32                           |
| 4                                  | 10.5043                           | 0.0269                           |
| 5                                  | -497.317                          | 507.79                           |
| 6                                  | 10.4816                           | 0.0042                           |
| 7                                  | -493.144                          | 503.62                           |
| 8                                  | 10.4778                           | 0.0004                           |
| 9                                  | -4.76180                          | 486.66                           |
| 10                                 | 10.4778                           | 0.0004                           |
| 11                                 | -509.447                          | 519.92                           |
| 12                                 | 10.4775                           | 0.0001                           |
| 13                                 | 10.4772                           | 0.0002                           |
| 14                                 | 10.4774                           | 0.0000                           |
| 15                                 | 10.4774                           | 0.0000                           |
| 16                                 | 10.4773                           | 0.0001                           |
| 17                                 | 10.4774                           | 0.0000                           |
| 18                                 | 10.4775                           | 0.0001                           |
| 19                                 | 10.4775                           | 0.0001                           |
| 20                                 | 10.4773                           | 0.0001                           |
| 24                                 | 10.4774                           | 0.0000                           |
| 29                                 | 10.4773                           | 0.0000                           |
| 34                                 | 10.4774                           | 0.0000                           |
| 39                                 | 10.4774                           | 0.0000                           |
The results in Table 4.1 show that by using enough terms \( n \) of Taylor series, the ETM method gives the linex Bayes estimates \( \hat{\theta}^{LT_n} \) as good as the naive method \( \hat{\theta}^{LN} \) as they both are close to the analytical Bayes estimates \( \hat{\theta}^{LB} \) and the Bayes estimate based on the Monte Carlo method \( \hat{\theta}^{LM} \) in Equations (4.14) and (4.15), respectively. However, it can give completely wrong Bayes estimate for the small number of terms \( n \). Using \( n \geq 19 \) seems to be good enough to approximate the linex loss function. The linex Bayes estimates obtained from using small \( n \) are not as good as large \( n \), however when the linex Bayes estimate obtained from using \( n = 2, 4, 6, \) and, 8 are very bad compared to using \( n = 1, 3, 5, \) and 7. We illustrate this result by using the plots in Figures 4.3.

![The linex loss function where \( c = 1 \) and the Taylor series approximation](image)

Figure 4.3: Plot of the Taylor series approximations, \( T_7 = T_{7,\hat{\theta}}(\theta), T_8 = T_{8,\hat{\theta}}(\theta), T_{10} = T_{10,\hat{\theta}}(\theta), \) and \( T_{14} = T_{14,\hat{\theta}}(\theta) \) (for the model N1).

For the Normal posterior distribution, Model N1 where \( \hat{\theta} = 10.48 \) in Figure 4.3, the Taylor approximation \( T_{8,\hat{\theta}}(\theta) \) approximates the linex loss function better than \( T_{7,\hat{\theta}}(\theta) \) however, it gives a negative loss when \( \theta \) moves away from \( \hat{\theta} \) while \( T_{7,\hat{\theta}}(\theta) \) is still positive. Therefore, when we use \( \mathbb{E}_\pi[T_{8,\hat{\theta}}(\theta)] \) to search for the minimum, it is leading to the negative linex Bayes estimate. This also explains the Taylor approximations using \( n = 1, \ldots, 6 \). Although the Taylor series approximation \( T_{10,\hat{\theta}}(\theta) \) gives a negative loss, the linex Bayes estimate is not negative. This is because the generated samples of the
parameter \( \theta \in (6.45, 15.27) \) so that there is no negative loss provided by any \( \theta \).

Consider the Gamma posterior distribution using the Exponential-Gamma conjugate model.

**Model G:**

\[
x_i | \theta \sim \text{Exp}(3), \quad i = 1, \ldots, 100,
\theta \sim \text{Gamma}(2, 0.5).
\theta | x \sim \text{Gamma}(12, 3.308741).
\] (4.16)

Using the moment generating function for a Gamma distribution (see Equation (A.2) in Appendix), the analytical linex Bayes with respect to the Gamma posterior distribution is

\[
\hat{\theta}^{LB} = - \log(\mathbb{E}_\pi[e^{-\theta}]) = - \log \left( \frac{3.308741}{3.308741 + 1} \right)^{12} = 3.168936.
\] (4.17)

The linex Bayes estimate based on the Monte Carlo method is

\[
\hat{\theta}^{LM} = - \log(\frac{1}{100,000} \sum_{j=1}^{100,000} e^{-\theta^j}) = 3.170826 \quad \text{where} \quad \theta^j \text{'s are generated from (4.16)}
\] (4.18)

The simulation results of the linex Bayes estimates obtained from using the naive method and the ETM method with respect to Gamma posterior distribution in Model G, (4.16) are shown in Table 4.2.

The results in Table 4.2 are similar to the results of using the Normal posterior distribution; using the ETM method via Taylor series approximation with \( n \geq 15 \) gives the linex Bayes estimates \( \hat{\theta}^{LT_n} \) close to \( \hat{\theta}^{LN} \) and they both are close to the analytical Bayes estimates \( \hat{\theta}^{LB} \) in Equation (4.17) and the linex Bayes estimate based on Monte Carlo method \( \hat{\theta}^{LM} \) in Equation (4.18). There are some \( n \) that give negative values of the Bayes estimates. The reason for this result can be explained by using Figure 4.4 in a similar way as for Model N1.
Table 4.2: The comparison of the linex Bayes estimates $\hat{\theta}^{LN}$ and $\hat{\theta}^{LT_n}$ with respect to Model G with the sample size $N = 100,000$.

| The naive method $\hat{\theta}^{LN}$ | The ETM method $\hat{\theta}^{LT_n}$ | The absolute errors $|\hat{\theta}^{LN} - \hat{\theta}^{LT_n}|$ |
|---|---|---|
| 3.1709 | 3.6274 | 0.4565 |
| 2 | -493.67 | 496.84 |
| 3 | 3.3175 | 0.1467 |
| 4 | -446.43 | 449.60 |
| 5 | 3.2283 | 0.0574 |
| 6 | -147.57 | 150.74 |
| 7 | 3.1897 | 0.0188 |
| 8 | -486.22 | 489.39 |
| 9 | 3.1758 | 0.0049 |
| 10 | -498.93 | 502.10 |
| 11 | 3.1717 | 0.0009 |
| 12 | -505.96 | 509.13 |
| 13 | 3.1710 | 0.0001 |
| 14 | 3.1707 | 0.0002 |
| 15 | 3.1708 | 0.0001 |
| 16 | 3.1708 | 0.0001 |
| 17 | 3.1708 | 0.0001 |
| 18 | 3.1708 | 0.0001 |
| 19 | 3.1708 | 0.0001 |
| 20 | 3.1708 | 0.0001 |
| 24 | 3.1708 | 0.0001 |
| 29 | 3.1708 | 0.0001 |
| 34 | 3.1708 | 0.0001 |
| 39 | 3.1708 | 0.0001 |
The linex loss function where $c = 1$ and the Taylor series approximation

Figure 4.4: Plot of the Taylor series approximations, $T_7 = T_{7,\hat{\theta}}(\theta), T_8 = T_{8,\hat{\theta}}(\theta), T_{10} = T_{10,\hat{\theta}}(\theta)$ and $T_{14} = T_{14,\hat{\theta}}(\theta)$ (for the model G).

To show the efficiency of the ETM method, we use the function “system.time” available in R to estimate the used time for computing the linex Bayes estimates. The results are shown in Table 4.3. We also show that the ETM method via Taylor series approximation with an appropriate number of terms $n$ in Table 4.4.

The ETM method is faster than the naive method as shown in Table 4.3. The reason is that when approximating the expectation of the linex loss function, we have to deal with the exponential function which can be expensive. Therefore, using the ETM method which only involves addition and multiplication is cheaper. Using the ETM method with more terms $n$ to obtain a better Bayes estimate does not cost much compared to the naive method. The results in Table 4.4 show that the ETM method is very advantageous compared to the naive method, especially for a large number of samples $N$ because the cost is increasing very little compared to the naive method.
Table 4.3: The comparison of the used time (seconds) to calculate the linex Bayes estimates $\hat{\theta}_n^{LN}$ and $\hat{\theta}_n^{LT}$.

<table>
<thead>
<tr>
<th>The naive method $\hat{\theta}_n^{LN}$</th>
<th>The ETM method $\hat{\theta}_n^{LT}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>102.59</td>
<td>1 1.23</td>
</tr>
<tr>
<td>2 1.73</td>
<td>2 1.99</td>
</tr>
<tr>
<td>4 2.31</td>
<td>5 2.58</td>
</tr>
<tr>
<td>6 2.90</td>
<td>7 3.21</td>
</tr>
<tr>
<td>8 3.59</td>
<td>9 3.93</td>
</tr>
<tr>
<td>10 4.42</td>
<td>11 4.71</td>
</tr>
<tr>
<td>12 5.05</td>
<td>13 5.46</td>
</tr>
<tr>
<td>14 5.59</td>
<td>15 6.09</td>
</tr>
<tr>
<td>16 6.44</td>
<td>17 7.16</td>
</tr>
<tr>
<td>18 7.58</td>
<td>19 7.80</td>
</tr>
<tr>
<td>20 8.43</td>
<td>24 9.83</td>
</tr>
<tr>
<td>29 12.21</td>
<td>34 14.97</td>
</tr>
<tr>
<td>39 17.44</td>
<td></td>
</tr>
</tbody>
</table>

Table 4.4: The comparison of the linex Bayes estimates $\hat{\theta}_n^{LN}$ and $\hat{\theta}_n^{LT}$, and the times based on a generated sample of $\theta$ from Model N1 with the sample size $N$.

<table>
<thead>
<tr>
<th>Number of samples $N$</th>
<th>The linex Bayes estimates $\hat{\theta}_n^{LN}$</th>
<th>$\hat{\theta}_n^{LT}$</th>
<th>The used times (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100,000</td>
<td>10.4774</td>
<td>10.4775</td>
<td>102.59</td>
</tr>
<tr>
<td>200,000</td>
<td>10.4808</td>
<td>10.4807</td>
<td>203.27</td>
</tr>
<tr>
<td>300,000</td>
<td>10.4824</td>
<td>10.4825</td>
<td>290.17</td>
</tr>
</tbody>
</table>
4.2.4 Taylor series approximation for the quotient loss function

One of many possible loss functions that can be employed is a bounded loss function. Therefore, it might be useful to study the ETM method for a bounded loss function. In this section, we consider the loss function given by

\[
L(\hat{\theta}, \theta) = 1 - \frac{1}{1 + (\hat{\theta} - \theta)^2}. \tag{4.19}
\]

This loss function penalises error symmetrically and the error goes to 1 as \(|\hat{\theta} - \theta| \to \infty\) (see Figure 4.5). For convenience, we name the loss function in Equation (4.19) the \textit{quotient loss function}.

![The quotient loss function](image)

Figure 4.5: Plot of the quotient loss function.
Consider the Taylor series representation of the arctan function

\[
\arctan x = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{2n+1} = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} \ldots \quad \text{for } |x| < 1.
\]

We then have

\[
\frac{1}{1 + x^2} = \frac{d(\arctan x)}{dx} = 1 - x^2 + x^4 - x^6 + x^8 \ldots \quad \text{for } |x| < 1,
\]

and hence

\[
1 - \frac{1}{1 + x^2} = x^2 - x^4 + x^6 - \ldots \quad \text{for } |x| < 1.
\]

Suppose that $|\hat{\theta} - \theta| < 1$, so the quotient loss function can be written as a Taylor series representation as

\[
L(\hat{\theta}, \theta) = 1 - \frac{1}{1 + (\hat{\theta} - \theta)^2} = (\hat{\theta} - \theta)^2 - (\hat{\theta} - \theta)^4 + \ldots + (-1)^{n+1}(\hat{\theta} - \theta)^{2n} + \ldots \quad \text{for } |\hat{\theta} - \theta| < 1.
\]

Therefore, the $n$ terms of the Taylor series approximation for the quotient loss function is

\[
T_{n,\theta}(\theta) = (\hat{\theta} - \theta)^2 - (\hat{\theta} - \theta)^4 + \ldots + (-1)^{n+1}(\hat{\theta} - \theta)^{2n} = C_1(\hat{\theta} - \theta)^2 + C_2(\hat{\theta} - \theta)^4 + \ldots + C_n(\hat{\theta} - \theta)^{2n}, \quad (4.20)
\]

where $C_i = (-1)^{2i+1}$ for $i = 1, 2, \ldots, n$ which are independent from $\hat{\theta}$. The plot of the Taylor series approximation with some $n$ terms is shown in Figure 4.6.
The quotient loss function and the Taylor series approximation

Figure 4.6: Plot of the quotient loss function and the Taylor series approximations, $T_2 = T_{2,\hat{\theta}}(\theta), T_4 = T_{4,\hat{\theta}}(\theta), T_7 = T_{7,\hat{\theta}}(\theta)$ and $T_9 = T_{9,\hat{\theta}}(\theta)$.

Unlike the linex loss function, the Taylor series, $T_{n,\hat{\theta}}(\theta)$ approximates well only when $\theta$ is close to $\hat{\theta}$ and very badly when $\theta$ moves away from $\hat{\theta}$. Moreover, the Taylor series approximation is not bounded by 1 no matter how many terms $n$ we used. Using $n = 2$ and 4, we have the Taylor approximation tends to go to $-\infty$ while using $n = 7$ and 9, it tends to go to $\infty$.

For convenience, we use the quotient Bayes estimate to refer the Bayes estimate under the quotient loss function in Equation (4.19). We denote the quotient Bayes estimate according to the method we use. We use $\hat{\theta}^{QN}$ to denote the quotient Bayes estimate obtained from the naive method, and use $\hat{\theta}^{QT_n}$ to denote the quotient Bayes estimate obtained from the ETM method using the $n$-terms Taylor series approximation.

The naive method:

$$\hat{\theta}^{QN} = \arg\min_{\theta} \mathbb{E}_\pi[1 - \frac{1}{1 + (\hat{\theta} - \theta)^2}] \quad (4.21)$$
The ETM method:

\[
\hat{\theta}^{QT_n} = \arg \min_{\hat{\theta}} \mathbb{E}_n[T_{n,\hat{\theta}}(\theta)]
\] (4.22)

where \(T_{n,\hat{\theta}}(\theta)\) is in Equation (4.20).

The quotient Bayes estimate is difficult to derive analytically. Therefore, we will compare the quotient Bayes estimate obtained from using the naive method to ones obtained from using the ETM method via the Taylor series approximation.

Simulation results of linex Bayes estimates

The simulation results of the quotient Bayes estimates obtained from using the naive method and the ETM method with respect to Normal posterior distribution in Model N1, (4.13) and Gamma posterior distribution in Model G, (4.16) are shown in Tables 4.5 and 4.6, respectively.

Table 4.5: The comparison of the quotient Bayes estimates \(\hat{\theta}^{QN}\) and \(\hat{\theta}^{QT_n}\) with respect to Model N1 with the sample size \(N = 100,000\).

| The naive method \(\hat{\theta}^{QN}\) | The ETM method \(\hat{\theta}^{QT_n}\) | The absolute errors \(|\hat{\theta}^{QN} - \hat{\theta}^{QT_n}|\) |
|---|---|---|
| 10.9831 | 10.9769 | 0.0062 |
| 10.9831 | -489.052 | 500.04 |
| 10.9831 | 10.9717 | 0.0114 |
| 10.9831 | -494.083 | 505.07 |
| 10.9831 | 10.9717 | 0.0114 |
| 10.9831 | -494.083 | 505.07 |
| 10.9831 | 10.9179 | 0.0652 |
| 10.9831 | -491.270 | 501.25 |
| 10.9831 | 10.9544 | 0.0827 |
| 10.9831 | -497.581 | 508.56 |
| 10.9831 | 11.3182 | 0.3351 |
| 10.9831 | -494.383 | 505.37 |
| 10.9831 | 12.3081 | 1.3250 |
| 10.9831 | -496.327 | 507.31 |
| 10.9831 | 17.3089 | 6.3258 |
| 10.9831 | -495.149 | 506.13 |
| 10.9831 | 16.2505 | 5.2674 |
| 10.9831 | -487.537 | 498.52 |
| 10.9831 | 17.3646 | 6.3815 |
| 10.9831 | -508.284 | 519.27 |
Table 4.6: The comparison of the quotient Bayes estimates $\hat{\theta}^{LN}$ and $\hat{\theta}^{LT_n}$ with respect to Model G with the sample size $N = 100,000$.

| The naive method $\hat{\theta}^{Q_N}$ | The ETM method $\hat{\theta}^{Q_{T_n}}$ | The absolute errors $|\hat{\theta}^{Q_N} - \hat{\theta}^{Q_{T_n}}|$ |
|---|---|---|
| 3.4493 | 1 3.6258 | 0.1765 |
| | 2 -494.25 | 497.70 |
| | 3 4.0340 | 0.5847 |
| | 4 -498.06 | 501.51 |
| | 5 4.3899 | 0.9406 |
| | 6 -503.86 | 507.31 |
| | 7 4.6748 | 1.2255 |
| | 8 -506.41 | 509.86 |
| | 9 4.8840 | 1.4347 |
| | 10 -503.52 | 506.97 |
| | 11 5.0351 | 1.5858 |
| | 12 -512.62 | 516.07 |
| | 13 5.1471 | 1.6978 |
| | 14 -494.44 | 497.89 |
| | 15 5.2264 | 1.7771 |
| | 16 -494.53 | 497.98 |
| | 17 5.6630 | 2.2137 |
| | 18 -491.67 | 495.12 |
| | 19 6.3049 | 2.8556 |
| | 20 -493.36 | 496.81 |

The results in Tables 4.5 - 4.6 show that the quotient Bayes estimate from Model N1, $\hat{\theta}^{Q_N} = 10.9831$ is quite close to the posterior mean, 10.97908. In contrast, the quotient Bayes estimate from Model G, $\hat{\theta}^{Q_N} = 3.44930$ is not that close to the posterior mean; $12/3.308741 = 3.626757$. We regard the quotient estimate obtained from using the naive method $\hat{\theta}^{Q_N}$ as the good Bayes estimate of the parameter $\theta$. Unlike the linear loss function, the Taylor series approximation works quite poorly for the quotient loss function. The quotient Bayes estimates, $\hat{\theta}^{Q_{T_n}}$ with respect to Model N1 in Table 4.5 and Model G in Table 4.6 both are getting worse as more terms $n$ in Taylor series are used. This is because using more terms $n$ leads to higher approximated expected values for even number $n$ (or vice versa for odd number $n$). In addition, the Taylor series approximation for the quotient loss function works only for $\theta$ where $|\hat{\theta} - \theta| < 1$. However, in computation, we cannot guarantee that $|\hat{\theta} - \theta| < 1$ as a value of $\hat{\theta}$ is changeable according to a minimisation algorithm. As a result, the Taylor series approximation is not a suitable approach to use the ETM method for Bayes estimate under the quotient loss function and possibly any other bounded loss functions.
4.2.5 Discussion and conclusion

The ETM method can be implemented using the Taylor series approximation. It is relatively fast to compute the simulated-based Bayes estimate. But it can provide the accurate Bayes estimates only when Taylor series approximates the loss functions well. The number of terms $n$ Taylor series plays important role in loss approximation. We can see in Figures 4.3 and 4.4 that using more $n$, we approximate the linex function better and hence we obtain the better Bayes estimates. It might be more practical to know how many terms $n$ do we need to use in order to have a good approximation for a loss function beforehand. Ideally, we would like to have $L(\hat{\theta}, \theta) = T_n(\hat{\theta})$ for all $\theta \in \Theta$. In computation, we use $\theta^j$ for $j = 1, \ldots, N$ as the values of $\theta$ to approximate a loss function and all possible values of $\theta^j$ are in $[\min\{\theta^j\}, \max\{\theta^j\}]$. This means that we might only need a good approximation for $\theta \in [\min\{\theta^j\}, \max\{\theta^j\}]$. As a result, we concentrate $T_n(\hat{\theta})$ such that it approximates the loss function well at $\theta = \theta^j$. The limitation of using the Taylor series approximation is that in general, the Taylor series approximation for a function $f(x)$ at point $a$ does not necessarily converge for all $x$. We have shown that that the Taylor series approximation works reasonably well for the linex loss function but poorly for the quotient loss function. The reason is that Taylor series for the linex loss function at $\hat{\theta}$ converges for all $\theta \in \mathbb{R}$ while Taylor series for the quotient loss function at $\hat{\theta}$ converges only if $|\hat{\theta} - \theta| < 1$.

In conclusion, the Taylor series approximation allows us to implement the ETM method for computing the simulation-based Bayes estimates under some but not all non-standard loss functions. It is very useful when the generated sample of the parameter $\theta$ used in the estimation part is large because it is much cheaper than the naive method. It is unsurprising that a bad approximation from the Taylor series approximation leads to completely wrong Bayes estimates. Moreover, the Taylor series approximation sometimes requires higher order derivatives and that can be complicated for some loss functions. As a result, the ETM method via the Taylor series approximation might not be plausible.

4.3 The ETM method via cubic spline interpolation

We demonstrated that using a Taylor series approximation to represent loss functions sometimes allows us to apply the ETM method which is cheaper than the naive method. However, it failed to generate a good Bayes estimate under the bounded quotient loss function. Can we use other approximations to express a loss function that allows us to use the ETM method for computing the Bayes estimate?
Many numerical techniques are used to approximate a function in terms of polynomials; for example, interpolation, extrapolation and curve fitting. Interpolation is a method of constructing new data points within the range of a discrete set of (known) data points. One of the simplest methods is linear interpolation that takes two data points to interpolate and provides the linear interpolant (the linear function) which is the straight line between those points. It is quick and easy, but it is not very accurate. Polynomial interpolation is a generalisation of linear interpolation in which the linear interpolant is replaced by a polynomial of higher degree. Spline interpolation that is a form of interpolation where the interpolant is a special type of piecewise polynomial called a spline. The concept of spline interpolation is that it uses low-degree polynomials in each of the intervals, and chooses the polynomial pieces such that they fit smoothly together. Cubic splines are the most popular because they provide a smooth interpolated function that is continuous through to the second derivative, while higher-degree splines have instabilities inherent in high-degree polynomials Yakowitz and Szidarovszky (1989).

Nowadays, cubic spline interpolation is available in many computer programming languages and is thus convenient to implement. In R, the function called “splinefun” is provided for performing cubic spline interpolation of given data points. Using cubic spline interpolation to approximate loss functions might be more flexible and convenient than Taylor series approximation because it requires data points for doing interpolation. In this section, we show how to use cubic spline interpolation to approximate loss functions and then apply the ETM method to compute the Bayes estimates. We still consider two loss functions; the linex loss function and the quotient loss function described in Section 4.2 as examples to investigate performance of the ETM method via cubic spline approximation.

### 4.3.1 Cubic spline functions

Spline functions are piecewise polynomials which interpolate a specified data set. Let $[a, b]$ be a finite interval containing the points $a = x_1 < \ldots < x_n = b$. A spline function of degree $m$ with interpolation point $x_i$ for $i = 1, \ldots, n$, is a piecewise polynomial $s(x)$ satisfying:

1. $s(x)$ is $m - 1$ times differentiable at each point $x_i$, for $i = 1, \ldots, n$,
2. On each interval $[x_i, x_{i+1}]$, $s(x)$ is a polynomial of degree not exceeding $m$. 

81
In this subsection, we describe a cubic spline function which is a polynomial of degree \( m = 3 \). Given the data points 

\[(x_1, y_1), \ldots, (x_n, y_n),\]

the \( x_i \)'s are known as knots and \( y_i \) are corresponding function values at \( x_i \).

**Definition 4.1.** A cubic spline \( s(x) \) is a piecewise polynomial that satisfies the following conditions:

1. \( s(x) = s_i(x) \) is a cubic polynomial on each interval \([x_i, x_{i+1}]\) for \( i = 1, \ldots, n - 1 \).
2. \( s(x_i) = y_i \) for \( i = 1, \ldots, n \).
3. \( s(x), s'(x), s''(x) \) are continuous on \([a, b]\).

Therefore, on each subinterval \([x_i, x_{i+1}]\), a cubic spline has the form

\[
s_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad (4.23)
\]

for \( i = 1, \ldots, n - 1 \), where \( a_i, b_i, c_i, \) and \( d_i \) are to be determined. This means that the total number of unknowns is \( 4(n - 1) \). We can use the following constraints to solve for these unknowns.

1. Interpolation continuity:
   - \( s_i(x_i) = y_i \) for \( i = 1, \ldots, n - 1 \),
   - \( s_i(x_{i+1}) = y_{i+1} \) for \( i = 1, \ldots, n - 1 \).
2. Derivative continuity:
   - \( s'_i(x_i) = s'_{i+1}(x_{i+1}) \) for \( i = 1, \ldots, n - 2 \),
   - \( s''_i(x_i) = s''_{i+1}(x_{i+1}) \) for \( i = 1, \ldots, n - 2 \).

These constraints give \( 4(n - 1) - 2 \) equations in total so we need 2 more conditions. There are several constraints for a cubic spline interpolation. The standard choices are:

(i) \( s''_i(x_1) = s''_n(x_n) = 0 \) : (the natural cubic spline),
(ii) \( s'_i(x_1) = f'(x_1), s'(x_n) = f'(x_n) \) if the derivatives are known : (the clamped cubic spline).

In this thesis, we use natural cubic spline interpolation. The details of how to construct a cubic spline and the algorithm to find spline coefficients in Equation (4.23) are available in many numerical books; Atkinson and Han (2004), Mathews (1992), Yakowitz and Szidarovszky (1989).
4.3.2 Cubic spline functions to approximate loss functions

Spline functions are commonly used in interpolation problems. The main practical task of interpolation is finding some function from a (given) collection of knots $x_i$ and $y_i$. In this section, we want to use spline functions to express a loss function. This means that we know exactly what the true functions (loss functions) are and want to express them as spline polynomials. Therefore, data points used for interpolation to approximate the loss functions can be chosen arbitrarily. We define data points used for cubic spline interpolation to approximate the loss function as follows.

**Definition 4.2.** Given the knots $\theta_i$ for $i = 1, \ldots, n$ on a finite interval of $\theta \in \Theta$, where $\theta_1 < \theta_2 < \ldots < \theta_n$.

If we choose one of these knots, $\theta_m$ for some $m \in \{1, 2, \ldots, n\}$ as the “true value” of $\theta$ which means $\hat{\theta} = \theta_m$ and then evaluate a loss function $L(\hat{\theta}, \theta_i)$ for $i = 1, \ldots, n$ to obtain the values of the corresponding data points

$$L(\hat{\theta}, \theta_i) \equiv L_{i, \hat{\theta}}$$

where $L_{m, \hat{\theta}} = L(\hat{\theta}, \hat{\theta}) = 0$. Therefore, we obtain $n$ “data points” as interpolating points

$$(\theta_1, L_{1, \hat{\theta}}), \ldots, (\theta_m, 0), \ldots, (\theta_n, L_{n, \hat{\theta}})$$

to construct a cubic spline function to approximate the loss function $L(\hat{\theta}, \theta)$.

Note that we are interested in loss functions such that $L(\hat{\theta}, \theta) = f(|\hat{\theta} - \theta|)$, so no matter what values $\hat{\theta}$ are, we still have the same shape of loss function and thus also the cubic spline approximation.

We perform cubic spline interpolation using the data points defined in Definition 4.2 to construct the cubic spline functions which are given by

$$s_{i, \hat{\theta}}(\theta) = a_i + b_i(\theta - \theta_i) + c_i(\theta - \theta_i)^2 + d_i(\theta - \theta_i)^3, \quad \text{for} \quad \theta \in [\theta_i, \theta_{i+1}], \quad i = 1, \ldots, n - 1$$

Unfortunately, the cubic spline functions in Equation (4.24) are defined for $\theta$ in the finite intervals which lower or upper endpoints are the knots $\theta_i$ for $i = 1, \ldots, n - 1$. We want to use these cubic spline functions to approximate a loss function $L(\hat{\theta}, \theta)$ in which $\theta$ might be defined on the infinite interval $(-\infty, \infty)$. We therefore use $s_{i, \hat{\theta}}(\theta)$ for $\theta \in (-\infty, \theta_1]$ and $s_{n-1, \hat{\theta}}(\theta)$ for $\theta \in (\theta_n, \infty)$. Therefore, we can approximate the loss
function as follows.

\[
L(\hat{\theta}, \theta) \approx s_{1,\hat{\theta}}(\theta)\mathbb{I}_{D_0}(\theta) + \sum_{i=1}^{n-1} s_i,\hat{\theta}(\theta)\mathbb{I}_{D_i}(\theta) + s_{n-1,\hat{\theta}}(\theta)\mathbb{I}_{D_n}(\theta)
\]  

(4.25)

where

\[
D_0 = (-\infty, \theta_1] \\
D_i = (\theta_i, \theta_{i+1}] \quad \text{for } i = 1, \ldots, n - 1 \\
D_n = (\theta_n, \infty)
\]

and the indicator function defined by

\[
\mathbb{I}_{D_i}(\theta) = \begin{cases} 
1 & \text{if } \theta \in D_i \\
0 & \text{otherwise},
\end{cases}
\]

for \( i = 0, \ldots, n \).

As a result, we can approximate the loss function \( L(\hat{\theta}, \theta) \) in which the value of \( \hat{\theta} \) is the knot \( \theta_m \). This means that we can approximate the loss function as the function of \( \theta \) explicitly and \( \hat{\theta} \) implicitly. The plot in Figure 4.7 shows the cubic spline interpolation to approximate the linear loss function.

We can see in Figure 4.7 that on the exponential curve, the cubic spline \( s_{1,\hat{\theta}}(\theta) \) does not approximate the linear loss function well compared to the other cubic spline functions \( s_i,\hat{\theta}(\theta) \). To improve the approximation, we can add more knots in between \( \theta_1 \) and \( \theta_2 \) as shown in Figure 4.8. The more knots we use, the better approximation we have. However, we do not want to have too many knots as it means it is more expensive. The next questions are what are the values of the knots and how many of them do we use for performing interpolation to approximate a loss function? Although the data points for performing interpolation to approximate a loss function can be chosen arbitrarily, we still need to find a suitable set of knots in order to obtain a good approximation and hence an accurate approximation of the Bayes estimate.
The linex loss function and cubic spline approximation

Figure 4.7: Plot of the linex loss function and cubic spline approximation using 6 knots where \( s_1 = s_{1, \hat{\theta}}(\theta), \ldots, s_5 = s_{5, \hat{\theta}}(\theta) \).

The linex loss function and cubic spline approximation

Figure 4.8: Plot of the linex loss function and cubic spline approximation using 8 knots where \( s_1 = s_{1, \hat{\theta}}(\theta), \ldots, s_7 = s_{7, \hat{\theta}}(\theta) \).
4.3.3 How to find a suitable set of knots

The cubic spline function \( s_{i, \hat{\theta}}(\theta) \) is constructed to approximate the loss function \( L(\hat{\theta}, \theta) \) where \( \theta \) is between the two knots, \([\theta_i, \theta_{i+1}]\). Therefore, we can make sure that possible values of the parameter \( \theta \) belong in some intervals by using the knots given by,

\[
\min \theta = \theta_1 < \theta_2 < \ldots < \theta_{n-1} < \theta_n = \max \theta.
\]

To find the Bayes estimate using the ETM method we use the generated samples \( \theta_j \sim \pi \) so that we can use the quantile values of the samples \( \theta_j \) to be the knots to make sure that every \( \theta_j \) is in some interval. In R, quantiles are provided by the function “quantile” which produces sample quantiles corresponding to the given probabilities. The smallest observation corresponds to a probability of 0 and the largest to a probability of 1. The function quantile in R gives quantile samples corresponding to order 0, 0.25, 0.50, 0.75 and 1 by default and denoted by

\[
\theta_{0\%}, \theta_{25\%}, \theta_{50\%}, \theta_{75\%}, \theta_{100\%}.
\]

For simplicity, we use subscripts 1, 2, 3, 4 and 5 to denote the quantile samples corresponding to order 0, 0.25, 0.50, 0.75 and 1, respectively. Therefore, we propose the quantile knots denoted by

\[
\theta_1, \theta_2, \theta_3, \theta_4, \theta_5,
\]

for performing cubic spline interpolation. However, these quantile knots might not be enough interpolating points to construct \( s_{i, \hat{\theta}}(\theta) \)'s which approximate the loss function well. Our strategy is adding more knots in between each interval \([\theta_i, \theta_{i+1}]\) determined by the quantile knots to obtain the better approximation if necessary. The knot to be added is the midpoint of the lower and upper endpoints. We stop adding knots between the interval \([\theta_i, \theta_{i+1}]\) when we obtain \( s_{i, \hat{\theta}}(\theta) = L(\hat{\theta}, \theta) \). However, it might be impossible to obtain the exact value. We could use the error tolerance (Tol) to stop adding knots. This means we stop when an error is less than the error tolerance. We could use;

an absolute error = \(|\text{true loss value} - \text{cubic spline value}|\),

or alternatively,

a relative error = \(\frac{|\text{true loss value} - \text{cubic spline value}|}{|\text{true loss value}|}\)

which gives an indication of how good a measurement is relative to the size of the true value being measured. Furthermore, it might be pointless to add the knots \( \theta_i \) in between \([\theta_{i-1}, \theta_{i+1}]\) if \(|\theta_{i+1} - \theta_{i-1}|\) is very small. In this case, we also stop adding knots.
The following procedure shows how we find a suitable set of knots.

1. Set \( \hat{\theta} = \theta_3 \) and construct the cubic splines \( s_{i,\hat{\theta}}(\theta) \) using the following interpolating points
\[
(\theta_1, L_{1,\hat{\theta}}), (\theta_2, L_{2,\hat{\theta}}), (\theta_3, L_{3,\hat{\theta}}), (\theta_4, L_{4,\hat{\theta}}), (\theta_5, L_{5,\hat{\theta}}).
\]

2. Specify the error tolerance (Tol).

3. Consider the interval \([\theta_2, \theta_3]\),
   
   (a) set \( a = \theta_2, b = \theta_3 \),
   
   (b) compute the initial midpoint \( m_0 = \frac{a + b}{2} \),
   
   (c) compute the error,
   
   \[ \Delta L = |s_{i,\hat{\theta}}(m_0) - L(\hat{\theta}, m_0)|, \]
   
   (d) while \( \Delta L \geq \text{Tol} \),
   
   i. update the midpoint; \( m_t = \frac{m_{t+1} + b}{2} \)
   
   ii. compute the error
   
   \[ \Delta L = |s_{i,\hat{\theta}}(m_t) - L(\hat{\theta}, m_t)|, \]
   
   (e) stop adding \( m_t \) when \( \Delta L < \text{Tol} \), or stop adding \( m_t \) when \( |m_t - b| < \epsilon \) where \( \epsilon > 0 \) and small ,
   
   (f) repeat (3a) - (3e) by setting \( a = \theta_2, b = m_0 \).

4. Repeat step 3 by considering the interval \([\theta_1, \theta_2]\) using \( a = \theta_1, b = \theta_2 \) where the spline function \( s_{i,\hat{\theta}}(\theta) \) is derived from using the quantile knots and the midpoint values \( m \)'s obtained from the interval \([\theta_2, \theta_3]\).

5. Repeat step 3 by considering the interval \([\theta_3, \theta_4]\) by using \( a = \theta_3, b = \theta_4 \) where the spline function \( s_{i,\hat{\theta}}(\theta) \) is derived from using the quantile knots and the midpoint values \( m \)'s obtained from the intervals \([\theta_2, \theta_3]\) and \([\theta_1, \theta_2]\).

6. Repeat step 3 by considering the interval \([\theta_4, \theta_5]\) by using \( a = \theta_4, b = \theta_5 \) where the spline function \( s_{i,\hat{\theta}}(\theta) \) is derived from using the quantile knots and the midpoint values \( m \)'s obtained from the intervals \([\theta_2, \theta_3]\), \([\theta_1, \theta_2]\) and \([\theta_3, \theta_4]\).
4.3.4 The ETM method

We use the approximation in Equation (4.25) to define the function representing the loss function corresponding to the given \( \hat{\theta} \)

\[
S_{n,\hat{\theta}}(\theta) = s_{1,\hat{\theta}}(\theta)[I_{D_0}(\theta)] + \sum_{i=1}^{n-1} s_{i,\hat{\theta}}(\theta)[I_{D_i}(\theta)] + s_{n-1,\hat{\theta}}(\theta)[I_{D_n}(\theta)],
\]

(4.26)

where \( s_{i,\hat{\theta}}(\theta) \) is obtained from Equation (4.24). If the knots \( \theta_i \) for \( i = 1, \ldots, n \) are fixed and known, the coefficients \( a_i, b_i, c_i \) and \( d_i \) for \( i = 1, \ldots, n-1 \) of the cubic spline functions in Equation (4.24) depend on the \( L_{i,\hat{\theta}} \) determined by the value of \( \hat{\theta} \). Therefore, we have an alternative expression of the cubic spline functions to represent the loss function \( L(\hat{\theta}, \theta) \) as

\[
s_{i,\hat{\theta}}(\theta) = \alpha_i^\hat{\theta} + \beta_i^\hat{\theta} \theta + \gamma_i^\hat{\theta} \theta^2 + \delta_i^\hat{\theta} \theta^3 \quad \text{for} \quad \theta \in [\theta_i, \theta_{i+1}], \ i = 1, \ldots, n-1,
\]

(4.27)

where \( \alpha_i^\hat{\theta}, \beta_i^\hat{\theta}, \gamma_i^\hat{\theta} \) and \( \delta_i^\hat{\theta} \) are determined by the knots \( \theta_i \) and the corresponding values of \( L_{i,\hat{\theta}} \).

Consider

\[
\mathbb{E}_\pi[S_{n,\hat{\theta}}(\theta)] = \mathbb{E}_\pi \left[ s_{1,\hat{\theta}}(\theta)[I_{D_0}(\theta)] + \sum_{i=1}^{n-1} s_{i,\hat{\theta}}(\theta)[I_{D_i}(\theta)] + s_{n-1,\hat{\theta}}(\theta)[I_{D_n}(\theta)] \right]
\]

\[
= \mathbb{E}_\pi[s_{1,\hat{\theta}}(\theta)[I_{D_0}(\theta)]] + \sum_{i=1}^{n-1} \mathbb{E}_\pi[s_{i,\hat{\theta}}(\theta)[I_{D_i}(\theta)]] + \mathbb{E}_\pi[s_{n-1,\hat{\theta}}(\theta)[I_{D_n}(\theta)]].
\]

Using the cubic spline function defined in Equation (4.27), we have

\[
\mathbb{E}_\pi[s_{i,\hat{\theta}}(\theta)[I_{D_i}(\theta)]] = \alpha_i^\hat{\theta} \mathbb{E}_\pi[I_{D_i}(\theta)] + \beta_i^\hat{\theta} \mathbb{E}_\pi[\theta I_{D_i}(\theta)] + \gamma_i^\hat{\theta} \mathbb{E}_\pi[\theta^2 I_{D_i}(\theta)] + \delta_i^\hat{\theta} \mathbb{E}_\pi[\theta^3 I_{D_i}(\theta)].
\]

We can see that the estimation part, \( \mathbb{E}_\pi[I_{D_0}(\theta)], \mathbb{E}_\pi[\theta I_{D_0}(\theta)], \mathbb{E}_\pi[\theta^2 I_{D_0}(\theta)] \) and \( \mathbb{E}_\pi[\theta^3 I_{D_0}(\theta)] \) involves the generated sample of the parameter \( \theta \) while the minimisation part involves the cubic spline coefficients as they depend on the values of \( \hat{\theta} \). Therefore, we can apply the ETM method to find the Bayes estimate. However, as the cubic spline coefficients depend on the values of \( \hat{\theta} \) so if the value of \( \hat{\theta} \) changes, we will need to reconstruct cubic spline functions corresponding to the new \( \hat{\theta} \). As a result, we need to find the cubic spline coefficients for every new point \( \hat{\theta} \) in the minimisation stage. It sounds as though the ETM method via cubic spline approximation might not be an efficient approach to find the Bayes estimate. Nevertheless, calculating the cubic spline coefficients is relatively cheap and might be even cheaper than using the naive method to compute Bayes estimates.
The cubic spline function $S_{n, \hat{\theta}}(\theta)$ approximates the loss function $L(\hat{\theta}, \theta)$ in particular when $\hat{\theta}$ is one of the knots $\theta_i$. Can we still use this cubic spline function to approximate the loss function when $\hat{\theta}$ is not equal to any value of the $\theta_i$ for $i = 1, \ldots, n$? It depends on how good or bad the cubic spline approximation is. We consider the cubic spline function to approximate two loss functions in the following examples.

**Example 4.1.** Given a set of 6 knots $\{\theta_i\} = \{-2, -0.5, 0, 1, 2.5, 4\}$. Suppose $\hat{\theta} = 2$ and we have the data points for cubic interpolation as 

$$( -2, L_1, 2), (-0.5, L_2, 2), (0, L_3, 2), (1, L_4, 2), (2.5, L_5, 2), (4, L_6, 2).$$

(4.28)

We consider two loss functions; the quadratic loss function, $L(\hat{\theta}, \theta) = (\hat{\theta} - \theta)^2$ and the linear loss function, $L(\hat{\theta}, \theta) = e^{(\hat{\theta} - \theta) - (\hat{\theta} - \theta) - 1}$. We use the data points (4.28) to perform cubic spline interpolation related to these two loss functions. Then the cubic spline functions are

$$s_{i,2}(\theta) = \alpha_i^{\hat{\theta}} + \beta_i^{\hat{\theta}} \theta + \gamma_i^{\hat{\theta}} \theta^2 + \delta_i^{\hat{\theta}} \theta^3$$

for $\theta \in [\theta_i, \theta_{i+1}]$, $i = 1, \ldots, 5$.

Therefore,

$$S_{n, \hat{\theta}}(\theta) = s_{1,\hat{\theta}}(\theta)I_{D_0}(\theta) + \sum_{i=1}^{5} s_{i,\hat{\theta}}(\theta)I_{D_i}(\theta) + s_{5,\hat{\theta}}(\theta)I_{D_n}(\theta),$$

where

$$D_0 = (-\infty, -2],$$

$$D_1 = (-2, 0.5],$$

$$D_2 = (-0.5, 0],$$

$$D_3 = (0, 1],$$

$$D_4 = (1, 2.5],$$

$$D_5 = (2.5, 4],$$

$$D_6 = (4, \infty).$$

(4.29)

Consider, the cubic spline function $s_{4,2}(\theta)$. The cubic spline function to approximate the quadratic loss function is

$$s_{4,2}(\theta) = 1 - 2(\theta - 1) + (\theta - 1)^2$$

$$= 4 - 4\theta + \theta^2$$

$$= (2 - \theta)^2,$$
and to approximate the linex loss function is

\[ s_{4,2}(\theta) = 0.718 - 1.791(\theta - 1) + 1.176(\theta - 1)^2 - 0.169(\theta - 1)^3 \\
= 3.854 - 4.65\theta + 1.683\theta^2 - 0.169\theta^3. \]

We can see that the cubic spline function \( s_{4,2}(\theta) \) to approximate the quadratic loss function has the exact same form of the true loss function so that \( s_{4,2}(2) = 0 = L(2, 2) \). However, the cubic spline function \( s_{4,2}(\theta) \) to approximate the linex loss function, we have \( s_{4,2}(2) = -0.066 \neq L(2, 2) \).

From Example 4.1, we can see that when we consider the quadratic loss function, it is not necessary to use \( \hat{\theta} \) from the knots to perform interpolation to obtain a good approximation. However, in general, it is unlikely that we obtain the cubic spline function in the exact same form of the true loss function, especially for complex loss functions. We can make sure that the loss function evaluated at \( \hat{\theta} \) is zero by adding \( \hat{\theta} \) as an extra knot for interpolation. Therefore, instead of using data points in Equation (4.28), we could use

\[ (-2, L_{1,2}), (-0.5, L_{2,2}), (0, L_{3,2}), (1, L_{4,2}), (2,0), (2.5, L_{5,2}), (4, L_{6,2}). \]

As a result, we will have 7 knots and hence 6 cubic spline functions;

\[ s_{i,\hat{\theta}}(\theta) = \alpha_i^{\hat{\theta}} + \beta_i^{\hat{\theta}}\theta + \gamma_i^{\hat{\theta}}\theta^2 + \delta_i^{\hat{\theta}}\theta^3 \quad \text{for} \quad \theta \in [\theta_i, \theta_{i+1}], \quad i = 1, \ldots, 6, \]

where

\[
\begin{align*}
D_0 &= (-\infty, -2], \\
D_1 &= (-2, 0], \\
D_2 &= (-0.5, 0], \\
D_3 &= (0, 1], \\
D_4 &= (1, 2], \\
D_5 &= (2, 2.5], \\
D_6 &= (2.5, 4], \\
D_7 &= (4, \infty).
\end{align*}
\]

We can see that as the extra point \( (2,0) \) is added, we also have the extra intervals \( D_4 \) and \( D_5 \) determined by this point in Equation (4.30). The other intervals are the same as Equation (4.29). This makes a change in the estimation part namely \( \mathbb{E}_\pi[\mathbb{1}_{D_i}(\theta)] \), \( \mathbb{E}_\pi[\theta\mathbb{1}_{D_i}(\theta)] \), \( \mathbb{E}_\pi[\theta^2\mathbb{1}_{D_i}(\theta)] \) and \( \mathbb{E}_\pi[\theta^3\mathbb{1}_{D_i}(\theta)] \) for \( i = 4 \) and \( 5 \) while the estimation parts related to other intervals \( D_i \) are still the same subject to a change in notation as they all shift to the right.
In practice, the extra point is used temporarily for interpolating as the value of \( \hat{\theta} \) is changed iteratively in the minimisation stage. Consequently, if \( \hat{\theta} \) is changed, we have to not only do interpolation to find the new cubic spline coefficients but also estimation for extra intervals added by the extra knot. Although we do not entirely implement the ETM method because of the extra knot \( \hat{\theta} \), we can still separate the estimation part from the minimisation part related to the other knots. The following procedure provides step by step of how to implement the ETM method via cubic spline interpolation using \( \hat{\theta} \) as the extra knot.

**The ETM method using cubic spline interpolation**

Suppose we have \( N \) generated samples of the parameter \( \theta \) from the posterior distribution \( \pi \).

1. Find a set of suitable knots \( \{\theta_1, \ldots, \theta_n\} \).
2. Specify one of the knots from step 1 as an initial value of \( \hat{\theta} \) denoted by \( \hat{\theta}(0) \) and then evaluate the loss function corresponding to such \( \hat{\theta} \) to obtain interpolating points,

\[
(\theta_1, L_{1,\hat{\theta}}), \ldots, (\hat{\theta}, 0), \ldots, (\theta_n, L_{n,\hat{\theta}})
\]

3. **Estimation part**
   (a) Estimate \( E_{\pi}[\mathbb{I}_{D_i}(\theta)] \), \( E_{\pi}[\theta\mathbb{I}_{D_i}(\theta)] \), \( E_{\pi}[\theta^2\mathbb{I}_{D_i}(\theta)] \), and \( E_{\pi}[\theta^3\mathbb{I}_{D_i}(\theta)] \) by

\[
E_{\pi}[\mathbb{I}_{D_i}(\theta)] \approx \frac{1}{N} \sum_{\theta^i \in D_i} 1,
\]

\[
E_{\pi}[\theta\mathbb{I}_{D_i}(\theta)] \approx \frac{1}{N} \sum_{\theta^i \in D_i} \theta^i,
\]

\[
E_{\pi}[\theta^2\mathbb{I}_{D_i}(\theta)] \approx \frac{1}{N} \sum_{\theta^i \in D_i} \theta^{i2},
\]

\[
E_{\pi}[\theta^3\mathbb{I}_{D_i}(\theta)] \approx \frac{1}{N} \sum_{\theta^i \in D_i} \theta^{i3}.
\]

4. **Minimisation part**
   (a) At iteration \( t \), add the candidate \( \hat{\theta}^{(t)} \) as one of the knots from step 1 to obtain \( n + 1 \) knots.
   (b) Approximate the loss function \( L(\hat{\theta}^{(t)}, \theta) \) by \( s_{i, \hat{\theta}^{(t)}} \) for \( i = 1, \ldots, n \) to get the spline function \( S_{n, \hat{\theta}^{(t)}}(\theta) \) by using Equation (4.26).
(c) Specify \( j \) such that \( \hat{\theta} \in D_j \):
- if \( j = 0 \) or \( n \), use the approximations in (4.31) to approximate the \( \mathbb{E}_{\pi}[S_n, \hat{\theta}(\cdot)] \),
- else add two more intervals; \( D_{\hat{\theta}, j} = (\theta_j, \hat{\theta}(t)) \) and \( D_{\hat{\theta}, j+1} = (\hat{\theta}(t), \theta_{j+1}) \) and estimate two more expectations according to these intervals, then use the expectations from step (4a) including these two to approximate the \( \mathbb{E}_{\pi}[S_n, \hat{\theta}(\cdot)] \).

(d) Repeat steps (4b) - (4c) with the new candidate \( \hat{\theta}(t+1) \).

In the estimation part, if there are knots \( \theta_i \) and \( \theta_{i+1} \) for some \( i = 1, \ldots, n \) such that \( |\theta_{i+1} - \theta_i| \) is very small, the interval \( D_i \) is also very small. As a result, we may not be able to estimate \( \mathbb{E}_{\pi}[\mathbb{I}_{D_i}(\theta)] \), \( \mathbb{E}_{\pi}[\theta \mathbb{I}_{D_i}(\theta)] \), \( \mathbb{E}_{\pi}[\theta^2 \mathbb{I}_{D_i}(\theta)] \) and \( \mathbb{E}_{\pi}[\theta^3 \mathbb{I}_{D_i}(\theta)] \) using (4.31) if there is no \( \theta \) in the interval \( D_i \) (the empty interval). Instead of using zeros for these expectations, we could approximate by averaging neighbouring values:

\[
\begin{align*}
\mathbb{E}_{\pi}[\mathbb{I}_{D_i}(\theta)] & \approx \frac{1}{2} (\mathbb{E}_{\pi}[\mathbb{I}_{D_{i-1}}(\theta)] + \mathbb{E}_{\pi}[\mathbb{I}_{D_{i+1}}(\theta)]) , \\
\mathbb{E}_{\pi}[\theta \mathbb{I}_{D_i}(\theta)] & \approx \frac{1}{2} (\mathbb{E}_{\pi}[	heta \mathbb{I}_{D_{i-1}}(\theta)] + \mathbb{E}_{\pi}[	heta \mathbb{I}_{D_{i+1}}(\theta)]) , \\
\mathbb{E}_{\pi}[\theta^2 \mathbb{I}_{D_i}(\theta)] & \approx \frac{1}{2} (\mathbb{E}_{\pi}[	heta^2 \mathbb{I}_{D_{i-1}}(\theta)] + \mathbb{E}_{\pi}[	heta^2 \mathbb{I}_{D_{i+1}}(\theta)]) , \\
\mathbb{E}_{\pi}[\theta^3 \mathbb{I}_{D_i}(\theta)] & \approx \frac{1}{2} (\mathbb{E}_{\pi}[	heta^3 \mathbb{I}_{D_{i-1}}(\theta)] + \mathbb{E}_{\pi}[	heta^3 \mathbb{I}_{D_{i+1}}(\theta)]) .
\end{align*}
\]

4.3.5 Cubic spline approximation for the linex loss function

We apply the ETM method to compute the linex Bayes estimate by using

\[
\hat{\theta}^{LS_n} = \arg \min_{\hat{\theta}} \mathbb{E}_{\pi}[S_n, \hat{\theta}] .
\]

We initially use the quantile knots from the quantile values from the samples of the parameter \( \theta \) generated from the Normal and Gamma posterior distributions in Model N1, (4.13) and Model G, (4.16). We then find a set of suitable \( n \) knots determined by small values of the error tolerance (Tol). By Definition 4.2, the interpolating points for the linex loss function are

\[
(\theta_1, L_{1, \hat{\theta}}), \ldots, (\theta_n, L_{n, \hat{\theta}}) .
\]
where

\[ L_{i,\hat{\theta}} = e^{(\hat{\theta} - \theta_i)} - (\hat{\theta} - \theta_i) - 1, \quad \text{for } i = 1, \ldots, n. \]

The plots of cubic spline interpolation to approximate the linex loss function are shown in Figures 4.7 - 4.8. The Bayes estimates \( \hat{\theta}^{LS_n} \) according to number knots \( n \) are shown in Table 4.7 and Table 4.8 for Model N1 and Model G, respectively.

**Simulation results of linex Bayes estimates**

Table 4.7: The comparison of the linex Bayes estimates \( \hat{\theta}^{LN} \) and \( \hat{\theta}^{LS_n} \) with respect to Model N1 with the sample size \( N = 100,000 \).

| The naive method \( \hat{\theta}^{LN} \) | The ETM method \( \hat{\theta}^{LS_n} \) | The absolute errors \( |\hat{\theta}^{LN} - \hat{\theta}^{LS_n}| \) |
|--------------------------------------|----------------------------------|----------------------------------|
| Tol \( n \)                        | \hat{\theta}^{LS_n}             |                                  |
| N/A 5                               | 9.97560                         | 0.5018                           |
| \( 10^{-2} \) 16                     | 10.4753                         | 0.0021                           |
| \( 10^{-3} \) 29                     | 10.4804                         | 0.0030                           |
| \( 10^{-4} \) 46                     | 10.4804                         | 0.0030                           |
| \( 10^{-5} \) 67                     | 10.4775                         | 0.0001                           |
| \( 10^{-6} \) 93                     | 10.4775                         | 0.0001                           |

Table 4.8: The comparison of the linex Bayes estimates \( \hat{\theta}^{LN} \) and \( \hat{\theta}^{LS_n} \) with respect to Model G with the sample size \( N = 100,000 \).

| The naive method \( \hat{\theta}^{LN} \) | The ETM method \( \hat{\theta}^{LS_n} \) | The absolute errors \( |\hat{\theta}^{LN} - \hat{\theta}^{LS_n}| \) |
|--------------------------------------|----------------------------------|----------------------------------|
| Tol \( n \)                        | \hat{\theta}^{LS_n}             |                                  |
| N/A 5                               | 3.1464                          | 0.0245                           |
| \( 10^{-2} \) 16                     | 3.1736                          | 0.0027                           |
| \( 10^{-3} \) 26                     | 3.1729                          | 0.0020                           |
| \( 10^{-4} \) 40                     | 3.1710                          | 0.0001                           |
| \( 10^{-5} \) 61                     | 3.1710                          | 0.0001                           |
| \( 10^{-6} \) 82                     | 3.1710                          | 0.0001                           |

Table 4.9: The comparison of the used time (seconds) to calculate the linex Bayes estimates \( \hat{\theta}^{LN} \) and \( \hat{\theta}^{LS_n} \) with respect to Model N1.

<table>
<thead>
<tr>
<th>The naive method ( n )</th>
<th>The ETM method using ( S_{n,\hat{\theta}}(\theta) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>102.59</td>
<td>61.47</td>
</tr>
<tr>
<td>16</td>
<td>34.22</td>
</tr>
<tr>
<td>29</td>
<td>31.53</td>
</tr>
<tr>
<td>46</td>
<td>33.00</td>
</tr>
<tr>
<td>67</td>
<td>37.58</td>
</tr>
<tr>
<td>93</td>
<td>43.13</td>
</tr>
</tbody>
</table>

Table 4.10: The comparison of linex Bayes estimates and the used time based on a generated sample of $\theta$ from Model N1 with the sample size $N$.

<table>
<thead>
<tr>
<th>Number of samples $N$</th>
<th>The Bayes estimates</th>
<th>The used time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\hat{\theta}^{LN}$</td>
<td>$\hat{\theta}^{LS_n}$</td>
</tr>
<tr>
<td>100,000</td>
<td>10.4774</td>
<td>10.4775</td>
</tr>
<tr>
<td>200,000</td>
<td>10.4808</td>
<td>10.4807</td>
</tr>
<tr>
<td>300,000</td>
<td>10.4824</td>
<td>10.4827</td>
</tr>
</tbody>
</table>

The simulation results of the Bayes estimates under the linex loss function shown in Tables 4.7 - 4.8 suggest that cubic spline approximation with suitable number of knots $n$ provides the Bayes estimate $\hat{\theta}^{LS_n}$ as well as the naive method does. Moreover, the result in Table 4.9 confirms that although the cubic spline approximation does not allow us to separate the estimation and minimisation entirely, it is still relatively cheap compared to the naive method. This implies that constructing cubic spline functions even with extra estimation related to the candidate value of $\hat{\theta}$ in the ETM method is cheaper than doing estimation and minimisation simultaneously in the naive method.

Interestingly, we found that using more knots $n$ is not necessarily more expensive. We can see that using $n = 5$ is the most expensive value of $n$ and using $n = 16$ is slightly more expensive than $n = 29$ and $n = 46$. Nevertheless, the cost tends to be increasing when $n$ is more than 29. The reason for this phenomenon is that for each iteration in the minimisation stage, we have to add the candidate $\hat{\theta}$ as the extra knot between some intervals $D_j$ determined by starting knots and then compute expectations according to the two extra intervals $D_{\hat{\theta},j}$ and $D_{\hat{\theta},j+1}$ (see step (5d) in the step by step procedure of the ETM method). Therefore, these two extra intervals tend to have more generated samples when $n$ is smaller so calculating expectations according to these extra intervals involves more samples. If there are many generated samples in such intervals, it makes the cost of calculating expectations more expensive than computing cubic spline functions. This can answer why using $n = 5$ is the most expensive and using $n = 16$ is more expensive than $n = 29$ and $n = 46$. However, when we use many knots ($n \geq 29$), the cost tends to be increasing as the cost of interpolation dominates the cost of calculating expectations, for example $n = 93$ is more expensive than $n = 29$. 

94
We found that under both Model N1 and Model G, using Tol = 10^{-5} is suitable to compute the linex Bayes estimates as it is relatively quick to provide an accurate linex Bayes estimate. Then in Table 4.10, we show that it is more efficient to implement the ETM method via cubic spline approximation with a suitable number of knots $n = 67$ (obtained from using Tol = 10^{-5} in Model N1) for large number of samples. The result shows that we obtain the Bayes estimate from the ETM method as well as from the naive method but much more cheaply. However, it is more expensive than using Taylor approximation that uses only 7.80 and 8.22 seconds for $N = 100,000$ and 200,000, respectively (see Table 4.4).

4.3.6 Cubic spline approximation for the quotient loss function

The quotient Bayes estimate under the cubic spline approximation is

$$\hat{\theta}_{QS_n} = \arg\min_{\hat{\theta}} \mathbb{E}_{\pi}[S_{n,\hat{\theta}}].$$

Similar to the linex loss function, the interpolating points are

$$(\theta_1, L_{1,\hat{\theta}}), \ldots, (\theta_n, L_{n,\hat{\theta}}),$$

where

$$L_{i,\hat{\theta}} = 1 - \frac{1}{1 + (\hat{\theta} - \theta_i)^2}, \quad \text{for } i = 1, \ldots, n.$$ 

The plots of cubic spline interpolation to approximate the quotient loss function are shown in Figure 4.9 and 4.10.
The quotient loss function and cubic spline approximation

Figure 4.9: Plot of the quotient loss function and cubic spline approximation using 5 knots where \( s_1 = s_{1,\hat{\theta}}(\theta), \ldots, s_4 = s_{4,\hat{\theta}}(\theta) \)

The quotient loss function and cubic spline approximation

Figure 4.10: Plot of the quotient loss function and cubic spline approximation using 9 knots where \( s_1 = s_{1,\hat{\theta}}(\theta), \ldots, s_8 = s_{8,\hat{\theta}}(\theta) \)
We implement the ETM method the same way as we do for the linex loss function. The results of the quotient Bayes estimates are shown in Tables 4.11 for Model N1 and Table 4.12 for Model G.

**Simulation results of quotient Bayes estimates**

Table 4.11: The comparison of the quotient Bayes estimates $\hat{\theta}^{QN}$ and $\hat{\theta}^{QS_n}$ with respect to Model N1 with the sample size $N = 100,000$.

| The naive method $\hat{\theta}^{QN}$ | The ETM method $\hat{\theta}^{QS_n}$ | The absolute errors $|\hat{\theta}^{QN} - \hat{\theta}^{QS_n}|$ |
|------------------------------------|-------------------------------------|----------------------------------|
| 10.9831                            | N/A 5 11.5514                       | 0.5683                           |
| $10^{-2}$                          | 8 10.8502                           | 0.1329                           |
| $10^{-3}$                          | 13 11.0014                          | 0.0183                           |
| $10^{-4}$                          | 35 10.9830                          | 0.0001                           |
| $10^{-5}$                          | 48 10.9831                          | 0.0000                           |
| $10^{-6}$                          | 70 10.9831                          | 0.0000                           |

Table 4.12: The comparison of the quotient Bayes estimates $\hat{\theta}^{QN}$ and $\hat{\theta}^{QS_n}$ with respect to Model G with the sample size $N = 100,000$.

| The naive method $\hat{\theta}^{QN}$ | The ETM method $\hat{\theta}^{QS_n}$ | The absolute errors $|\hat{\theta}^{QN} - \hat{\theta}^{QS_n}|$ |
|------------------------------------|-------------------------------------|----------------------------------|
| 3.4493                             | N/A 5 3.1464                        | 0.3029                           |
| $10^{-2}$                          | 9 3.4189                            | 0.0304                           |
| $10^{-3}$                          | 19 3.4497                           | 0.0004                           |
| $10^{-4}$                          | 29 3.4491                           | 0.0002                           |
| $10^{-5}$                          | 46 3.4492                           | 0.0001                           |
| $10^{-6}$                          | 73 3.4492                           | 0.0001                           |

Table 4.13: The comparison of the used time (seconds) to calculate the quotient Bayes estimates $\hat{\theta}^{QN}$ and $\hat{\theta}^{QS_n}$ with respect to Model N1.

<table>
<thead>
<tr>
<th>The naive method $S_{\hat{\theta}}$</th>
<th>The ETM method using $S_{\hat{\theta}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>58.06</td>
</tr>
<tr>
<td>8</td>
<td>54.88</td>
</tr>
<tr>
<td>13</td>
<td>43.27</td>
</tr>
<tr>
<td>35</td>
<td>31.42</td>
</tr>
<tr>
<td>48</td>
<td>33.45</td>
</tr>
<tr>
<td>70</td>
<td>37.59</td>
</tr>
</tbody>
</table>
Table 4.14: The comparison of the quotient Bayes estimates and the used time based on a generated sample of $\theta$ from Model N1 with the sample size $N$.

<table>
<thead>
<tr>
<th>Number of samples $N$</th>
<th>The Bayes estimates $\hat{\theta}^{QN}$</th>
<th>The used time (seconds)</th>
<th>The naive method</th>
<th>The ETM method</th>
</tr>
</thead>
<tbody>
<tr>
<td>100,000</td>
<td>10.9831</td>
<td>50.61</td>
<td>33.45</td>
<td></td>
</tr>
<tr>
<td>200,000</td>
<td>10.9832</td>
<td>156.08</td>
<td>52.27</td>
<td></td>
</tr>
<tr>
<td>300,000</td>
<td>10.9817</td>
<td>214.26</td>
<td>66.11</td>
<td></td>
</tr>
</tbody>
</table>

The results in Tables 4.11 - 4.12 show that using the ETM method via cubic spline interpolation for the quotient loss function gives quotient Bayes estimates as good as the naive method does. We can see from these tables that the values of $\hat{\theta}^{QS_n}$ are close to $\hat{\theta}^{QN}$. The trend of the used time of the ETM method and the naive method under the quotient loss function shown in Table 4.13 is the same as the linex loss function described earlier. The ETM method is very efficient, especially for a large number of samples, $N$ provided an appropriate knots is used as shown in Table 4.14. For the quotient loss function, the ETM method via cubic spline approximation works very well while Taylor approximation fails to provide a good Bayes estimate.

4.3.7 Discussion and conclusion

The ETM method via cubic spline approximation might not entirely separate the estimation part from the minimisation part so it is more expensive than the ETM method via Taylor series approximation. However, cubic spline approximation seems to be more plausible to approximate a non-standard loss function than Taylor series approximation. The reason is that when $\hat{\theta}$ moves away from $\theta$, Taylor series approximation is more strict as it relies on convergence of Taylor series, while cubic spline approximation does not depend on this restriction. Moreover, the Bayes estimate can be very wrong for some number of terms $n$ in Taylor series, while cubic spline approximation gives relatively good Bayes estimate even for small number of knots $n$. The estimation part of the ETM method via Taylor series approximation is in the form (4.4) so that the estimation part of the linex loss function is different from the quotient loss function. Nevertheless, the estimation part of the ETM method via cubic spline approximation is always in the forms of $E_\pi[\theta^iI_{D_i}(\theta)]$ for $i = 0, \ldots, 3$ no matter what the loss function is. Therefore, cubic spline approximation is flexible to use for different loss functions as it uses the same form of estimation part. Cubic spline interpolating is quite convenient to do in many computer programming languages. Another possible disadvantage of Taylor series approximation is that it often requires higher order derivatives. For some loss functions, it might be difficult to get higher order derivatives.
On the contrary, cubic spline interpolation requires a loss function to be twice continuously differentiable (the first and the second derivatives are continuous) to satisfy the constraint for interpolation. We improve cubic spline approximation by using more knots and corresponding loss values which is very convenient.

In conclusion, cubic spline interpolation is preferable to Taylor series approximation to approximate non-standard loss functions for finding the Bayes estimates. One difficulty of using cubic spline approximation is to determine number of knots $n$ which is good enough for loss approximation and makes the ETM method cheap. In this thesis, we have determined the number of knots $n$ according to the error tolerance $(\text{Tol})$. We have found that using $\text{Tol} = 10^{-4}$ leads to a suitable set of knots which is good enough for Bayes estimates under the linex and the quotient loss functions. However, it is rather a trial and error approach. For these two loss functions, we have found that although using the same error tolerance, we could obtain a different number of knots. For other complex loss functions, we might have to use smaller error tolerance which will lead to more knots to obtain a good approximation. Using the ETM method might be cheaper than the naive method but we cannot guarantee success. As a result, cubic spline approximation might fail to make the ETM method cheaper than the naive method when we cannot find an appropriate set of knots.
Chapter 5

Bicubic Interpolation for Bayes Estimates in Two Dimensions

5.1 Introduction

We have shown that the ETM method works reasonably well to provide Bayes estimates in one dimension. It is common to extend and generalise the use of this method to higher dimensions of the parameter space. However, in some cases, it might not be necessary to consider very high dimensions because we could treat some irrelevant parameters as nuisance parameters. For example, we are not interested in an estimate of the allocation parameter $z$ in the mixture model which will be discussed in Chapter 6. Therefore, using the ETM method in a lower dimension may be adequate. In this chapter, we want to extend the ETM method to compute Bayes estimates of the parameter in two dimensions, $\theta = (\theta_1, \theta_2)$ where $\theta$ is in the parameter space $\Theta$. In two dimensions, the loss function can be written as

$$L(\hat{\theta}, \theta) = L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)).$$

This means the simulation-based Bayes estimate is obtained from

$$(\hat{\theta}_1, \hat{\theta}_2)^* = \arg \min_{(\hat{\theta}_1, \hat{\theta}_2)} \mathbb{E}_{\pi}[L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2))].$$

(5.1)
where the expected loss function is estimated by Monte Carlo integration or MCMC samples,

\[ E_{\pi} L[(\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)] \approx \frac{1}{N} \sum_{j=1}^{N} L[(\hat{\theta}_1, \hat{\theta}_2), (\theta_{1j}, \theta_{2j})]. \]

We have shown in Chapter 4 that cubic spline interpolation is a reasonable approximation to express loss functions in order to apply the ETM method to compute the Bayes estimates. Therefore, we want to use the idea of approximating a function based on interpolation used in one dimension to compute Bayes estimates in two dimensions in Equation (5.1).

In two dimensional interpolation, suppose we are given functional values \( f_{ij} \), with \( i = 1, \ldots, n_1 \) and \( j = 1, \ldots, n_2 \) which have corresponding given data points \((x_i, y_j)\). We then use an interpolation method to formulate a function \( f \) to evaluate a point \((x, y)\). In this thesis, however we want to approximate a loss function which is a function of the parameter of interest \( \theta = (\theta_1, \theta_2) \). Therefore, we use the notation \( \theta_{1i}, \theta_{2j} \) for given data points and \( f(\theta_1, \theta_2) \) for the function of two variables instead. The simplest interpolation in two dimensions is bilinear interpolation on the grid square. It is an extension of linear interpolation for interpolating functions of two variables. The concept of bilinear interpolation is to do linear interpolation first in one direction then do again it in the other direction, Press et al. (2009). A drawback of the bilinear method is that as the interpolating point moves from grid square to grid square, the gradient of the interpolated function changes discontinuously at the boundaries of each grid. We can make use of gradients and higher order derivatives to obtain smoothness. Bicubic interpolation with gradients and the cross derivative constraints at grid points brings the smoothness property. The bicubic interpolation method gives a smoother interpolated surface than surfaces obtained by bilinear interpolation. Bilinear and bicubic interpolation methods are also widely used as an image interpolation algorithm, Acharya and Tsai (2007) and Li and Orchard (2001). There is another technique for interpolation and smoothing technique, thin-plate splines. This is also the generalisation of splines which may be used in two or more dimensions to obtain a smooth surface. The concept of the thin-plate spline is providing the spline surface which represents a thin metal sheet that is constrained not to move at the grid points. The thin-plate spline method is usually used for interpolating and smoothing arbitrarily spaced points in the plane; see more details in Bookstein (1989), Hutchinson (1995), and Sibson and Stone (1991). In this chapter, we concentrate on bicubic interpolation.
5.2 Bicubic interpolation

The problem of interpolation in two dimensions can be described as follows. Suppose we are given a matrix of functional values \( f_{ij} \) with \( i = 0, \ldots, m_1 \) and \( j = 0, \ldots, m_2 \) corresponding to given points of \( \theta_1 \), values \( \theta_1i \), and \( \theta_2 \), values \( \theta_2j \), with \( i \) and \( j \) just mentioned. We assume that \( f_{ij} \) is some function of \( \theta_1i \) and \( \theta_2j \), \( f_{ij} = f(\theta_1i, \theta_2j) \). An interpolated surface \( p \) obtained from a method of interpolation in two dimensions approximates the function \( f \)

\[ p(\theta_1, \theta_2) \approx f(\theta_1, \theta_2). \]

We can use the interpolated surface \( p \) to estimate \( f \) at a new point \((\theta_1, \theta_2)\). We refer to those points used in interpolation as grid points \( \theta_1i \) and \( \theta_2j \). An important concept of interpolation in two dimensions is the grid square in which the point \((\theta_1, \theta_2)\) falls, that is, the four grid points that enclose the desired interior point as shown in Figure 5.1.

![Figure 5.1: The grid square from grid points \{\theta_{11}, \theta_{12}, \theta_{13}\} on \theta_1-axis and \{\theta_{21}, \theta_{22}, \theta_{23}\} on \theta_2-axis.](image)

The simplest interpolation in two dimensions is bilinear interpolation, however, it is not an attractive method because as the interpolating points move from grid square to
grid square the interpolated function value changes continuously but the gradient of the interpolated function changes discontinuously at the boundaries of each grid square, Press et al. (2009). We then use bicubic interpolation which requires derivatives and cross derivatives at the corner of a grid square. These requirements enforce the continuous change of the gradient of the interpolated function values as the interpolating point moves from one grid square to another. We describe how to do bicubic interpolation on a grid square as follows.

Suppose the function values $f$ and the derivatives $\frac{\partial f}{\partial \theta_1}$, $\frac{\partial f}{\partial \theta_2}$ and $\frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}$ are known at the four corners $(0,0)$, $(1,0)$, $(1,1)$ and $(0,1)$ of the unit square. Best of all is to know the derivatives analytically, otherwise they can be computed accurately at the four corners by numerical methods, for example the central difference method (see Smith (1985)).

The bicubic interpolated surface (also known as the bicubic formula for the function $f$) on the unit square can be written

$$p(\theta_1, \theta_2) = \sum_{k=0}^{3} \sum_{l=0}^{3} c_{kl} \theta_1^k \theta_2^l.$$  (5.2)

To formulate the bicubic surface $p$ in Equation (5.2), we need to determine 16 coefficients $c_{kl}$ for $k, l = 0, \ldots , 3$ by the following procedure.

We match $p(\theta_1, \theta_2)$ with the 4 function values $f$, 4 gradient values $\frac{\partial f}{\partial \theta_1}$, 4 gradient values $\frac{\partial f}{\partial \theta_2}$ and 4 cross derivative values $\frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}$ at the four corners as follows.

1. $f(0, 0) = p(0, 0) = c_{00}$
2. $f(1, 0) = p(1, 0) = c_{00} + c_{10} + c_{20} + c_{30}$
3. $f(0, 1) = p(0, 1) = c_{00} + c_{01} + c_{02} + c_{03}$
4. $f(1, 1) = p(1, 1) = \sum_{k=0}^{3} \sum_{l=0}^{3} c_{kl}$
5. $\frac{\partial f}{\partial \theta_1}(0, 0) = \frac{\partial p}{\partial \theta_1}(0, 0) = c_{10}$
6. $\frac{\partial f}{\partial \theta_1}(1, 0) = \frac{\partial p}{\partial \theta_1}(1, 0) = c_{10} + 2c_{20} + 3c_{30}$
7. $\frac{\partial f}{\partial \theta_1}(0, 1) = \frac{\partial p}{\partial \theta_1}(0, 1) = c_{10} + c_{11} + c_{12} + c_{13}$
8. $\frac{\partial f}{\partial \theta_1}(1, 1) = \frac{\partial p}{\partial \theta_1}(1, 1) = \sum_{k=1}^{3} \sum_{l=0}^{3} c_{kl} k$
9. $\frac{\partial f}{\partial \theta_2}(0, 0) = \frac{\partial p}{\partial \theta_2}(0, 0) = c_{01}$

10. $\frac{\partial f}{\partial \theta_2}(1, 0) = \frac{\partial p}{\partial \theta_2}(1, 0) = c_{01} + c_{11} + c_{31}$

11. $\frac{\partial f}{\partial \theta_2}(0, 1) = \frac{\partial p}{\partial \theta_2}(0, 1) = c_{01} + 2c_{02} + 3c_{03}$

12. $\frac{\partial f}{\partial \theta_2}(1, 1) = \frac{\partial p}{\partial \theta_2}(1, 1) = \sum_{k=0}^{3} \sum_{l=1}^{3} c_{kl} l$

13. $\frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(0, 0) = \frac{\partial^2 p}{\partial \theta_1 \partial \theta_2}(0, 0) = c_{11}$

14. $\frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(1, 0) = \frac{\partial^2 p}{\partial \theta_1 \partial \theta_2}(1, 0) = c_{11} + 2c_{21} + c_{31}$

15. $\frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(0, 1) = \frac{\partial^2 p}{\partial \theta_1 \partial \theta_2}(0, 1) = c_{11} + c_{12} + 3c_{13}$

16. $\frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(1, 1) = \frac{\partial^2 p}{\partial \theta_1 \partial \theta_2}(1, 1) = \sum_{k=0}^{3} \sum_{l=0}^{3} c_{kl} k l$

where the expressions above have used (5.2) and the following identities,

- $\frac{\partial p}{\partial \theta_1}(\theta_1, \theta_2) = \sum_{k=1}^{3} \sum_{l=0}^{3} c_{kl} k \theta_1^{k-1} \theta_2^{l}$

- $\frac{\partial p}{\partial \theta_2}(\theta_1, \theta_2) = \sum_{k=0}^{3} \sum_{l=1}^{3} c_{kl} \theta_1^{k} \theta_2^{l-1}$

- $\frac{\partial^2 p}{\partial \theta_1 \partial \theta_2}(\theta_1, \theta_2) = \sum_{k=1}^{3} \sum_{l=1}^{3} c_{kl} k \theta_1^{k-1} l \theta_2^{l-1}$

Writing the unknown parameters $c_{kl}$ in a vector,

$c = (c_{00}, c_{10}, c_{20}, c_{30}, c_{01}, c_{11}, c_{21}, c_{31}, c_{02}, c_{12}, c_{22}, c_{32}, c_{03}, c_{13}, c_{23}, c_{33})^T$

and known (given) values

$x = \left( f(0, 0), f(1, 0), f(0, 1), f(1, 1), \frac{\partial f}{\partial \theta_1}(0, 0), \frac{\partial f}{\partial \theta_1}(1, 0), \frac{\partial f}{\partial \theta_1}(0, 1), \frac{\partial f}{\partial \theta_1}(1, 1), \frac{\partial f}{\partial \theta_2}(0, 0), \frac{\partial f}{\partial \theta_2}(1, 0), \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(0, 0), \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(1, 0), \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(0, 1), \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2}(1, 1) \right)^T$
then the problem can be reformulated into a linear equation \( Ac = x \) where \( A \)’s inverse can be found explicitly, Press et al. (2009),

\[
A^{-1} = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & -3 & 3 & 0 & 0 & -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & -2 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & -3 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\
9 & -9 & -9 & 9 & 6 & 3 & -6 & -3 & 6 & 6 & 3 & -3 & 4 & 2 & 2 & 1 & 0 & 0 & 0 & 0 \\
-6 & 6 & 6 & -6 & -3 & 3 & 3 & -4 & 4 & -2 & 2 & 2 & -2 & -2 & 1 & 1 & 1 & 1 & 1 & 1 \\
2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 0 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
-6 & 6 & 6 & -6 & -4 & -2 & 4 & 2 & -3 & 3 & -3 & 3 & -2 & -1 & -2 & -1 & 0 & 0 & 0 & 0 \\
4 & -4 & -4 & -4 & 2 & 2 & -2 & 2 & -2 & 2 & -2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\end{pmatrix}
\]

Therefore, with the known vector \( x \) and the \( A \)’s inverse, we solve for \( c \) to obtain the bicubic formula (5.2) defined on the unit square \([0, 1] \times [0, 1]\). What about bicubic interpolation on a rectangle? Suppose we are given grid points \( \theta_{1i}, \theta_{1i+1} \) and \( \theta_{2j}, \theta_{2j+1} \) as 4 corners of the rectangle \([\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]\). We can use the same procedure with a small adjustment to obtain the bicubic formula. First of all we replace the four corners of the unit square \([0, 1] \times [0, 1]\) by the corners of the rectangle \([\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]\) (see Figure 5.2).

We make use of the procedure of bicubic interpolation on the unit square \([0, 1] \times [0, 1]\) as described earlier. Define new variables which each is between 0 and 1,

\[
u = (\theta_1 - \theta_{1i})/(\theta_{1i+1} - \theta_{1i}), \\
v = (\theta_2 - \theta_{2j})/(\theta_{2j+1} - \theta_{2j}).
\]

Then the bicubic formula of a variable \((u, v)\) defined in the unit square \([0, 1] \times [0, 1]\) can be written as

\[
p(u, v) = \sum_{k=0}^{3} \sum_{l=0}^{3} c_{kl} u^k v^l \quad \text{for } 0 \leq u \leq 1, \text{ and } 0 \leq v \leq 1.
\]
Figure 5.2: A point \((\theta_1, \theta_2)\) in a rectangle \([\theta_1^i, \theta_{1i+1}] \times [\theta_2^j, \theta_{2j+1}]\) and a point \((u, v)\) in the unit square \([0, 1] \times [0, 1]\).

It is equivalent to the bicubic formula as a function of a variable \((\theta_1, \theta_2)\) defined in the rectangle \([\theta_1^i, \theta_{1i+1}] \times [\theta_2^j, \theta_{2j+1}]\),

\[
p(\theta_1, \theta_2) = \sum_{k=0}^{3} \sum_{l=0}^{3} c_{kl} \left( \frac{\theta_1 - \theta_1^i}{\theta_{1i+1} - \theta_1^i} \right)^k \left( \frac{\theta_2 - \theta_2^j}{\theta_{2j+1} - \theta_2^j} \right)^l,
\]

for \(\theta_1^i \leq \theta_1 \leq \theta_{1i+1}\) and \(\theta_2^j \leq \theta_2 \leq \theta_{2j+1}\).

The coefficients \(c_{kl}\) for \(k, l = 0, \ldots, 3\) can be found by matching 16 expressions of the function values and its derivatives at four corners \([\theta_1^i, \theta_{1i+1}] \times [\theta_2^j, \theta_{2j+1}]\) similar to the unit square \([0, 1] \times [0, 1]\). Derivatives can be found by using a change of variables and the chain rule, Spiegel (1963),

\[
\frac{\partial f}{\partial u} = \frac{\partial f}{\partial \theta_1} \frac{\partial \theta_1}{\partial u}, \quad \frac{\partial f}{\partial v} = \frac{\partial f}{\partial \theta_2} \frac{\partial \theta_2}{\partial v},
\]

and cross derivatives are

\[
\frac{\partial^2 f}{\partial u \partial v} = \frac{\partial}{\partial u} \left( \frac{\partial f}{\partial v} \right) = \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2} \frac{\partial \theta_1}{\partial u} \frac{\partial \theta_2}{\partial v},
\]

where

\[
\theta_1 = u(\theta_{1i+1} - \theta_{1i}) + \theta_{1i},
\]

\[
\theta_2 = v(\theta_{2j+1} - \theta_{2j}) + \theta_{2j}.
\]
Note that the order of the derivative does not matter if the cross derivatives \( \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2} \) and \( \frac{\partial^2 f}{\partial \theta_2 \partial \theta_1} \) are continuous on a disk that contains the point \((\theta_1, \theta_2)\), Spiegel (1963). This means \( \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2} (\theta_1, \theta_2) = \frac{\partial^2 f}{\partial \theta_2 \partial \theta_1} (\theta_1, \theta_2) \). To obtain the coefficients \( c_{kl} \) for the bicubic formula (5.4) on the rectangle \([\theta_1, \theta_1+1] \times [\theta_2, \theta_2+1]\), we follow the same procedure as on the unit square. We find the coefficient vector \( c \) by using the \( A^{-1} \) in (5.3) in the linear equation \( Ac = x \) where the vector \( x \) consists of the function and the derivatives values evaluated at rectangular corners described earlier.

Suppose we are given grid points \( \theta_1 \) for \( i = 1, \ldots, n_1 \) and \( \theta_2 \) for \( j = 1, \ldots, n_2 \). Then we can formulate the bicubic surface on rectangles \([\theta_1, \theta_1+1] \times [\theta_2, \theta_2+1]\) for \( i = 1, \ldots, n_1-1 \) and \( j = 1, \ldots, n_2-1 \) denoted by \( p_{ij} \). If the derivatives match on boundaries, then we patch surfaces \( p_{ij} \) together to obtain the bicubic surface on the rectangle \([\theta_1, \theta_1+1] \times [\theta_2, \theta_2+1]\).

Bicubic formulas in Equations (5.2) and (5.4) are generally used for interpolation problems in two dimensions. In practice, interpolation is performed to approximate the unknown function by using a set of data points. In the next section, we describe how we apply bicubic interpolation to approximate loss functions which are certainly well-defined functions. As in one dimension, the main reason for using the bicubic formula to approximate or express the loss function is that the bicubic formula allows us to implement the ETM method to compute the Bayes estimate.

### 5.3 Bicubic interpolation to approximate loss functions

In this section, we detail how to use bicubic interpolation to express non-standard loss functions for parameter estimation in two dimensions. Recall a loss function for point estimation in one dimension, it is usually a function of two variables \( \hat{\theta} \) and \( \theta \) denoted by \( L(\hat{\theta}, \theta) \). The variable \( \hat{\theta} \) represents the point estimate of the true parameter \( \theta \). The properties of loss functions for one dimensional parameter estimation are also applied to two dimensions. The loss function is increasing as \( \|\hat{\theta} - \theta\| \to \infty \) and \( L(\hat{\theta}, \theta) = 0 \) iff \( \hat{\theta} = \theta \).

Cubic spline interpolation to approximate a loss function starts with specifying knots \( \theta_i \) for \( i = 1, \ldots, n \) which can be chosen arbitrarily. Similarly, in bicubic interpolation, we can specify grid points \( \theta_1 \) and \( \theta_2 \) for some \( i = 1, \ldots, n_1 \) and \( j = 1, \ldots, n_2 \). We define data points used for bicubic interpolation to approximate the loss function as follows.
Definition 5.1. Determine grid points \( \theta_1, \theta_2 \), for \( i = 1, \ldots, n_1 \) and \( j = 1, \ldots, n_2 \) on finite intervals of \((\theta_1, \theta_2) \in \Theta\), where

\[
\theta_1 < \theta_1 < \ldots < \theta_{n_1} \quad \text{and} \quad \theta_2 < \theta_2 < \ldots < \theta_{2n_2}.
\]

If we choose one of these grid points, \((\theta_{1m_1}, \theta_{2m_2})\) for some \( m_1 \in \{1, 2, \ldots, n_1\} \) and \( m_2 \in \{1, 2, \ldots, n_2\} \) as the true value of \((\theta_1, \theta_2)\) which means \((\hat{\theta}_1, \hat{\theta}_2) = (\theta_{1m_1}, \theta_{2m_2})\) and then evaluate a loss function \( L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) \) for \( i = 1, \ldots, n_1 \) and \( j = 1, \ldots, n_2 \) to obtain the values of the corresponding data points,

\[
L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) = L_{i,j}(\hat{\theta}_1, \hat{\theta}_2) \quad \text{for} \quad i = 1, \ldots, n_1 \quad \text{and} \quad j = 1, \ldots, n_2,
\]

where \( L_{m_1m_2}(\hat{\theta}_1, \hat{\theta}_2) = L((\hat{\theta}_1, \hat{\theta}_2), (\hat{\theta}_1, \hat{\theta}_2)) \). Therefore, we obtain \( n_1 \times n_2 \) grid points for interpolation,

\[
((\theta_1, \theta_2), L_{11}(\hat{\theta}_1, \hat{\theta}_2)), ((\theta_1, \theta_2), L_{12}(\hat{\theta}_1, \hat{\theta}_2)), \ldots, ((\theta_1, \theta_2), L_{1n_1}(\hat{\theta}_1, \hat{\theta}_2)),
\]

\[
((\theta_1, \theta_2), L_{21}(\hat{\theta}_1, \hat{\theta}_2)), ((\theta_1, \theta_2), L_{22}(\hat{\theta}_1, \hat{\theta}_2)), \ldots, ((\theta_1, \theta_2), L_{2n_2}(\hat{\theta}_1, \hat{\theta}_2)),
\]

\[
((\theta_1, \theta_2), L_{n_11}(\hat{\theta}_1, \hat{\theta}_2)), ((\theta_1, \theta_2), L_{n_12}(\hat{\theta}_1, \hat{\theta}_2)), \ldots, ((\theta_1, \theta_2), L_{n_1n_2}(\hat{\theta}_1, \hat{\theta}_2)),
\]

to construct bicubic functions to approximate the loss function \( L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) \).

For convenience, we use \( L_{ij} \) to denote \( L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) \). We then perform bicubic interpolation on the grid points defined in Definition 5.1 by using the following values to find the bicubic coefficients.

For each rectangle \([\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]\) for \( i = 1, \ldots, n_1 - 1 \) and \( j = 1, \ldots, n_2 - 1 \),

we evaluate 4 loss values at 4 corners of the rectangles

\[
L_{ij}, L_{i+1j}, L_{ij+1}, L_{i+1j+1},
\]

(5.5)

4 derivatives with respect to \( \theta_1 \)

\[
\frac{\partial L_{ij}}{\partial \theta_1}, \frac{\partial L_{i+1j}}{\partial \theta_1}, \frac{\partial L_{ij+1}}{\partial \theta_1}, \frac{\partial L_{i+1j+1}}{\partial \theta_1},
\]

(5.6)

4 derivatives with respect to \( \theta_2 \),

\[
\frac{\partial L_{ij}}{\partial \theta_2}, \frac{\partial L_{i+1j}}{\partial \theta_2}, \frac{\partial L_{ij+1}}{\partial \theta_2}, \frac{\partial L_{i+1j+1}}{\partial \theta_2},
\]

(5.7)

and 4 cross derivatives with respect to \( \theta_1 \) and \( \theta_2 \),

\[
\frac{\partial^2 L_{ij}}{\partial \theta_1 \partial \theta_2}, \frac{\partial^2 L_{i+1j}}{\partial \theta_1 \partial \theta_2}, \frac{\partial^2 L_{ij+1}}{\partial \theta_1 \partial \theta_2}, \frac{\partial^2 L_{i+1j+1}}{\partial \theta_1 \partial \theta_2},
\]

(5.8)
Therefore, for a given value of \((\hat{\theta}_1, \hat{\theta}_2)\), we can approximate the loss function by the bicubic formula as follows:

\[
p_{(\hat{\theta}_1, \hat{\theta}_2)}(\theta_1, \theta_2) = \sum_{k=0}^{3-1} \sum_{l=0}^{3-1} c_{klij} \left( \frac{\theta_1 - \hat{\theta}_{1i}}{\theta_{1i+1} - \hat{\theta}_{1i}} \right)^k \left( \frac{\theta_2 - \hat{\theta}_{2j}}{\theta_{2j+1} - \hat{\theta}_{2j}} \right)^l,
\]

where \((\theta_1, \theta_2) \in [\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]\), for \(i = 1, \ldots, n_1 - 1\) and \(j = 1, \ldots, n_2 - 1\). The bicubic coefficients \(c_{klij}\) are obtained from using the same method as we explained in section 5.2 by replacing the function values and derivatives of function \(f\) at the 4 corners of the rectangle \([0, 1] \times [0, 1]\) with the loss values and derivatives from Equations (5.5), (5.6), (5.7) and (5.8) for 4 corners of the rectangle \([\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]\). The subscripts \(ij\) of the coefficients \(c_{kl}\) indicate the corresponding rectangle \([\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]\).

There are \(n_1 \times n_2\) bicubic formulas to express the loss function \(L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2))\) for all possible values \((\theta_1, \theta_2)\). We use an indicator function to indicate the rectangle in which the point \((\theta_1, \theta_2)\) falls. Therefore, we can approximate the loss function by the bicubic formulas as follows:

\[
L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) \approx \sum_{i=1}^{n_1-1} \sum_{j=1}^{n_2-1} p_{(\hat{\theta}_1, \hat{\theta}_2)}(\theta_1, \theta_2) \mathbb{I}_{[\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]}(\theta_1, \theta_2)
\]

\[
= \sum_{i=1}^{n_1-1} \sum_{j=1}^{n_2-1} \sum_{k=0}^{3-1} \sum_{l=0}^{3-1} c_{klij} \left( \frac{\theta_1 - \hat{\theta}_{1i}}{\theta_{1i+1} - \hat{\theta}_{1i}} \right)^k \left( \frac{\theta_2 - \hat{\theta}_{2j}}{\theta_{2j+1} - \hat{\theta}_{2j}} \right)^l \mathbb{I}_{[\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]}(\theta_1, \theta_2),
\]

where

\[
\mathbb{I}_{[\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]}(\theta_1, \theta_2) = \begin{cases} 
1 & \text{if } (\theta_1, \theta_2) \in [\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}] \\
0 & \text{otherwise.}
\end{cases}
\]

How many grid points do we need to use for bicubic interpolation? We use the same idea as when we chose the knots for cubic spline interpolation to choose grid points. We initially use the quantile values from the generated samples of the parameter \((\theta_1, \theta_2)\) to be grid points for bicubic interpolation. We then add more grid points between the quantile grid points to improve the approximation as detailed in the next section.

### 5.4 How to find a suitable set of grid points

In general for interpolation, the more data points used, the better. As a result, using the quantile grid points might not be good enough. In this section, we describe how to find a suitable set of grid points.
The idea is similar to choosing the knots in cubic spline interpolation, we choose the quantile grid points corresponding to order 0, 0.25, 0.50, 0.75 and 1 denoted by subscripts 1, 2, 3, 4 and 5, respectively;

\[ \theta_{11}, \theta_{12}, \theta_{13}, \theta_{14}, \theta_{15}, \theta_{21}, \theta_{22}, \theta_{23}, \theta_{24}, \theta_{25}. \]

We then add more points on the \( \theta_1 \)-axis and the \( \theta_2 \)-axis if necessary. The following procedure shows how we find a suitable set of grid points.

1. Set \( (\hat{\theta}_1, \hat{\theta}_2) = (\theta_{13}, \theta_{23}) \) and perform bicubic interpolation to obtain \( p(\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2) \) using the following interpolating points \( ((\theta_{1i}, \theta_{2j}), L_{ij}) \), for \( i, j = 1, \ldots, 5 \).

2. Specify the error tolerance \( \text{Tol} \).

3. Consider the rectangle \( [\theta_{12}, \theta_{13}] \times [\theta_{22}, \theta_{23}] \),

   (a) set four corners of the considered rectangle,
   \[ a_1 = \theta_{12}, b_1 = \theta_{13}, \]
   \[ a_2 = \theta_{22}, b_2 = \theta_{23}, \]

   (b) compute the initial midpoint on \( \theta_1 \)-axis, \( m_{10} = \frac{a_1 + b_1}{2} \),
   (c) compute the error,
   \[ \Delta L = |p(\hat{\theta}_1, \hat{\theta}_2), (m_{10}, \theta_2) - L((\hat{\theta}_1, \hat{\theta}_2), (m_{10}, \theta_2))| \]
   (d) if \( \Delta L < \text{Tol} \), stop, else, continue the next step,
   (e) set four corners of the considered rectangle
   \[ a_1 = m_{10}, b_1 = \theta_{13}, \]
   \[ a_2 = \theta_{22}, b_2 = \theta_{23}, \]

   (f) compute the initial midpoint on \( \theta_2 \)-axis, \( m_{20} = \frac{a_2 + b_2}{2} \)
   (g) compute the error,
   \[ \Delta L = |p(\hat{\theta}_1, \hat{\theta}_2), (m_{10}, m_{20}) - L((\hat{\theta}_1, \hat{\theta}_2), (m_{10}, m_{20}))| \]
(h) if $\Delta L < \text{Tol}$, stop else, continue the next step,
(i) set four corners of the considered rectangle
\[
    a_1 = m_{10}, b_1 = \theta_{13} \\
    a_2 = m_{20}, b_2 = \theta_{23}
\]
(j) while $\Delta L \geq \text{Tol}$,
   i. repeat steps (3a) - (3i) where the mid points on $\theta_1$-axis and $\theta_2$-axis are
\[
    m_{1t} = \frac{m_1_{t-1} + b_1}{2} \\
    m_{2t} = \frac{m_2_{t-1} + b_2}{2}
\]
and the four corners are
\[
    a_1 = m_{1t-1}, b_1 = \theta_{13} \\
    a_2 = m_{2t-1}, b_2 = \theta_{23}
\]
(k) stop when $\Delta L < \text{Tol}$, or stop adding $m_1$, when $|m_1_i - b_1| < \epsilon$ and $m_2$ when $|m_2_i - b_2| < \epsilon$ where $\epsilon > 0$ and small.

4. Compute $d_1 = |b_1 - m_{1t}|$ and $d_2 = |b_2 - m_{2t}|$.

5. Add points between $[a_1, b_1]$ with the length $d_1$ and between $[a_2, b_2]$ with the length $d_2$ where $a_1, b_1, a_2, b_2$ are from step (3a).

6. Repeat steps 3 - 5 by considering the rectangle $[\theta_{1i}, \theta_{1i+1}] \times [\theta_{2i}, \theta_{2i+1}]$ where
\[
    a_1 = \theta_{1i}, b_1 = \theta_{1i+1} \\
    a_2 = \theta_{2i}, b_2 = \theta_{2i+1} \quad \text{for } i = 1, 3 \text{ and } 4, \text{ respectively.}
\]

We can illustrate how to add grid points between the quantile grid points on $\theta_1$-axis and $\theta_2$-axis in Figures 5.3 - 5.10.
Figure 5.3: Consider the rectangle $[\theta_{12}, \theta_{13}] \times [\theta_{22}, \theta_{23}]$.

Figure 5.4: Add the midpoint on $\theta_1$-axis and use four corner points on the green rectangle for bicubic interpolation.

Figure 5.5: Add the midpoint on $\theta_2$-axis and use four corner points on the green rectangle for bicubic interpolation.

Figure 5.6: Repeat adding the midpoint on $\theta_1$-axis and use the green rectangle for bicubic interpolation.
Figure 5.7: Repeat adding the midpoint on $\theta_2$-axis after adding the midpoint on $\theta_1$-axis and use the green rectangle for bicubic interpolation.

Figure 5.8: Obtain the smallest rectangle which has dimension $d_1 \times d_2$ related to the midpoints on $[\theta_{12}, \theta_{13}]$ and $[\theta_{22}, \theta_{23}]$.

Figure 5.9: Add all the points on the $\theta_1$-axis and $\theta_2$-axis of the rectangle $[\theta_{12}, \theta_{13}] \times [\theta_{22}, \theta_{23}]$.

Figure 5.10: Consider the other rectangles; $[\theta_{1i}, \theta_{1i+1}] \times [\theta_{2i}, \theta_{2i+1}]$ for $i = 1, 3$ and 4 to obtain all grid points.
5.5 The ETM method via bicubic interpolation

We use the approximation in Equation (5.9) to define the function representing the loss function corresponding to given \((\hat{\theta}_1, \hat{\theta}_2)\),

\[
P(\theta_1, \theta_2, n_{12}) = \sum_{i=1}^{n_1-1} \sum_{j=1}^{n_2-1} P(\theta_1, \theta_2, i_j)(\theta_1, \theta_2) I_{[\theta_1, \theta_2]}(\theta_1, \theta_2)
\]

\[
= \sum_{i=1}^{n_1-1} \sum_{j=1}^{n_2-1} \sum_{k=0}^{3} \sum_{l=0}^{3} c_{kl}i_j \left( \frac{\theta_1 - \theta_{1i}}{\theta_{1i+1} - \theta_{1i}} \right)^k \left( \frac{\theta_2 - \theta_{2i}}{\theta_{2i+1} - \theta_{2i}} \right)^l I_{[\theta_1, \theta_{1i+1}] \times [\theta_2, \theta_{2i+1}]}(\theta_1, \theta_2).
\]

(5.10)

If the bicubic approximation in Equation (5.10) approximates the loss function well, we might obtain the Bayes estimate

\[
(\hat{\theta}_1, \hat{\theta}_2)^P = \arg\min_{\theta_1, \theta_2} \mathbb{E}_\pi[P(\theta_1, \theta_2, n_{12})]\]

as accurately as we do from using the true loss function. Consider the expectation of the bicubic approximation in Equation (5.10),

\[
\mathbb{E}_\pi[P(\theta_1, \theta_2, n_{12})] = \sum_{i=1}^{n_1-1} \sum_{j=1}^{n_2-1} \sum_{k=0}^{3} \sum_{l=0}^{3} c_{kl}i_j \mathbb{E}_\pi \left[ \left( \frac{\theta_1 - \theta_{1i}}{\theta_{1i+1} - \theta_{1i}} \right)^k \left( \frac{\theta_2 - \theta_{2i}}{\theta_{2i+1} - \theta_{2i}} \right)^l I_{[\theta_1, \theta_{1i+1}] \times [\theta_2, \theta_{2i+1}]} \right].
\]

For convenience, we denote

\[
\mathbb{E}_{i,j}^{kl} = \mathbb{E}_\pi \left[ \left( \frac{\theta_1 - \theta_{1i}}{\theta_{1i+1} - \theta_{1i}} \right)^k \left( \frac{\theta_2 - \theta_{2i}}{\theta_{2i+1} - \theta_{2i}} \right)^l I_{[\theta_1, \theta_{1i+1}] \times [\theta_2, \theta_{2i+1}]} \right],
\]

for example,

\[
\mathbb{E}_{12}^{13} = \mathbb{E} \left[ \left( \frac{\theta_1 - \theta_{11}}{\theta_{12} - \theta_{11}} \right)^{13} \left( \frac{\theta_2 - \theta_{21}}{\theta_{23} - \theta_{21}} \right)^3 \right] \text{ for } (\theta_1, \theta_2) \in [\theta_{11}, \theta_{12]} \times [\theta_{21}, \theta_{23}].
\]

Similar to the cubic spline approximation, we will add the value \((\hat{\theta}_1, \hat{\theta}_2)\) as an extra grid point to perform bicubic interpolation in order to make sure that \(L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) = P(\hat{\theta}_1, \hat{\theta}_2, n_{12})(\theta_1, \theta_2) = 0\). As a result, we can not entirely implement the ETM method to compute the Bayes estimate. The following procedure provides step by step of how to implement the ETM method via bicubic spline interpolation using \((\hat{\theta}_1, \hat{\theta}_2)\) as the extra grid points.
The ETM method using bicubic interpolation

Suppose we have $N$ generated samples of the parameter $(\theta_1, \theta_2)$ from the posterior distribution $\pi$.

1. Find a set of grid points

$$\theta_{11} < \ldots \leq \theta_{1n_1},$$
$$\theta_{21} < \ldots \leq \theta_{2n_2},$$

for $i = 1, \ldots, n_1$ and $j = 1, \ldots, n_2$.

2. Specify one of the grid points from step 1 as an initial value of $(\hat{\theta}_1, \hat{\theta}_2)$ denoted by and then evaluate the loss function corresponding to such $\hat{\theta}$ to obtain,

$$((\theta_1, \theta_2), L_{11}), ((\theta_1, \theta_2), L_{12}), \ldots, ((\theta_1, \theta_2), L_{1n_2})$$
$$((\theta_1, \theta_2), L_{21}), ((\theta_1, \theta_2), L_{22}), \ldots, ((\theta_1, \theta_2), L_{2n_2}), \ldots,$$
$$((\theta_1, \theta_2), L_{n_11}), ((\theta_1, \theta_2), L_{n_12}), \ldots, ((\theta_1, \theta_2), L_{n_1n_2}).$$

3. Estimation part

(a) Estimate $E_{kl}^{ij}$ for $i = 1, \ldots, n_1 - 1$, $j = 1, \ldots, n_2 - 1$ and $k, l = 0, 1, 2, 3$ by using the generated samples $(\theta_1^{(n)}, \theta_2^{(n)})$ for $n = 1, \ldots, N$,

$$E_{ij}^{kl} = E \left[ \frac{(\theta_1 - \theta_{1i})^k}{\theta_{1i+1} - \theta_{1i}} \frac{(\theta_2 - \theta_{2j})^l}{\theta_{2j+1} - \theta_{2j}} \right]_{[\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]}$$

$$\approx \frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \theta_{1i}}{\theta_{1i+1} - \theta_{1i}} \right)^k \left( \frac{\theta_2^{(n)} - \theta_{2j}}{\theta_{2j+1} - \theta_{2j}} \right)^l,$$

where $(\theta_1^{(n)}, \theta_2^{(n)}) \in [\theta_{1i}, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]$ for $k, l = 0, 1, 2, 3$.

4. Minimisation part

(a) At iteration $t$, add the candidate $(\hat{\theta}_1^{(t)}, \hat{\theta}_2^{(t)})$ as an extra grid point to those listed in step 1 to. We then have

$$\theta_{11} < \ldots < \theta_{1i} < \hat{\theta}_1 < \theta_{1i+1} < \ldots < \theta_{1n_1}$$
$$\theta_{21} < \ldots < \theta_{2j} < \hat{\theta}_2 < \theta_{2j+1} < \ldots < \theta_{2n_2},$$

for $i = 1, \ldots, n_1$ and for $j = 1, \ldots, n_2$ as new grid points for bicubic interpolation.
(b) Approximate the loss function $L(\hat{\theta}^{(t)}_1, \hat{\theta}^{(t)}_2, (\theta_1, \theta_2))$ by $P(\hat{\theta}_1, \hat{\theta}_2, n_{1+1n_{2+1}}(\theta_1, \theta_2))$ by using Equation (5.10).

(c) Estimate $E_{kl}^{ij}$ for $i = 1, \ldots, n_1$, $j = 1, \ldots, n_2$ and $k, l = 0, 1, 2, 3$ by using the procedure of Estimation corresponding to the new grid point $\hat{\theta}_1$ and $\hat{\theta}_2$ (see Appendix C).

(d) Repeat steps (4b) - (4c) with the new candidate $(\hat{\theta}_1^{(t+1)}, \hat{\theta}_2^{(t+1)})$.

5. At iteration $t$, the Bayes estimate $P(\hat{\theta}_1, \hat{\theta}_2) = (\hat{\theta}_1^{(t)}, \hat{\theta}_2^{(t)})$ which gives the minimum $E_\pi[P(\hat{\theta}_1, \hat{\theta}_2, n_{1+1n_{2+1}}(\theta_1, \theta_2))]$ is the Bayes estimate.

Note that to implement the ETM method entirely, we replace steps (4a) - (4c) by one step that approximates the loss function $L((\hat{\theta}_1^{(t)}, \hat{\theta}_2^{(t)}), (\theta_1, \theta_2))$ by $P(\hat{\theta}_1, \hat{\theta}_2, n_{1+1n_{2+1}}(\theta_1, \theta_2))$.

5.6 The implementation of the ETM method

In the ETM method, we will estimate $E_{kl}^{ij} = \pi n_{1+1n_{2+1}}(\theta_1, \theta_2)$ for $i = 1, \ldots, n_1$, $j = 1, \ldots, n_2$ and for $k, l = 0, 1, 2, 3$ by using the generated samples $(\theta_1^{(n)}, \theta_2^{(n)})$ for $n = 1, \ldots, N$ as the values of the parameter $(\theta_1, \theta_2)$. We can guarantee that there is at least one value of $(\theta_1^{(n)}, \theta_2^{(n)}) \in [\hat{\theta}_1, \theta_{i+1}] \times [\hat{\theta}_2, \theta_{j+1}]$ for $i, j = 1, \ldots, 4$ if we use the quantile samples to be grid points (see Figure 5.11).
However, it might be the case that using the quantile grid points is not good enough to approximate the loss function so we have to add more grid points between \( \theta_1 \) and/or \( \theta_2 \) for \( i, j = 1, \ldots, 5 \) as described in Section 5.4. We might have a rectangle \([\hat{\theta}_1, \theta_{1i+1}] \times [\hat{\theta}_2, \theta_{2j+1}]\) for some \( i, j \) such that there is no point of \((\theta_1^{(n)}, \theta_2^{(n)})\) in such rectangle (see Figure 5.12).

We call that rectangle \([\hat{\theta}_1, \theta_{1i+1}] \times [\hat{\theta}_2, \theta_{2j+1}]\) the empty rectangle. How can we compute \( E^{kl}_{ij} \) where \([\hat{\theta}_1, \theta_{1i+1}] \times [\hat{\theta}_2, \theta_{2j+1}]\) is empty? We will use the dummy variable obtained from

\[
(\theta_1, \theta_2) = \left( \frac{\hat{\theta}_1 + \theta_{1i+1}}{2}, \frac{\hat{\theta}_2 + \theta_{2j+1}}{2} \right)
\]

to be the point in the empty rectangle (see Figure 5.13).
Figure 5.12: The empty rectangles after adding grid points between the quantile grid points.

The model for simulation

We will use the independent multivariate Normal distribution as the posterior distri-
bution, $\pi$ of the parameter $\theta = (\theta_1, \theta_2)$, where

$$(\theta_1, \theta_2|x) \sim N(\mu, \Sigma),$$

and the posterior hyperparameters

$$\mu = (\mu_1, \mu_2) \quad \text{and} \quad \Sigma = \begin{pmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{pmatrix}.$$

Therefore, we will simulate the sample of the parameter $\theta = (\theta_1, \theta_2)$ from Model N2 as follows.

Model N2:

$$(\theta_1, \theta_2|x) \sim N \left( (0, 6), \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \right). \quad (5.11)$$
5.7 Bicubic interpolation for the linex loss function

Define the linex loss function of two parameters as

$$L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) = e^{(\hat{\theta}_1 - \theta_1)} - (\hat{\theta}_1 - \theta_1) - 1 + e^{(\hat{\theta}_2 - \theta_2)} - (\hat{\theta}_2 - \theta_2) - 1.$$  \hspace{1cm} (5.12)

Similar to one dimension, we denote the linex Bayes estimate according to the method that means we use $((\hat{\theta}_1, \hat{\theta}_2))_{LN}$ to denote the linex Bayes estimate obtained from the naive method, and use $((\hat{\theta}_1, \hat{\theta}_2))_{LP}$ to denote the linex Bayes estimate obtained from the ETM method using the bicubic interpolation approximation.

The naive method:

$$((\hat{\theta}_1, \hat{\theta}_2))_{LN} = \arg \min_{\hat{\theta}} \mathbb{E}_n[e^{(\hat{\theta}_1 - \theta_1)} - (\hat{\theta}_1 - \theta_1) - 1 + e^{(\hat{\theta}_2 - \theta_2)} - (\hat{\theta}_2 - \theta_2) - 1]. \hspace{1cm} (5.13)$$
The ETM method:

\[
(\hat{\theta}_1, \hat{\theta}_2)_{LP} = \arg \min_{\hat{\theta}} \mathbb{E}_\pi\{P(\hat{\theta}_1, \hat{\theta}_2, n_1, n_2(\theta_1, \theta_2))\}, \quad (5.14)
\]

where \(P(\theta_1, \theta_2, n_1, n_2(\theta_1, \theta_2))\) is in Equation (5.10) with the grid points defined in Definition 5.1, where

\[
L_{ij, (\hat{\theta}_1, \hat{\theta}_2)} = e^{(\hat{\theta}_1 - \theta_1) - 1} + e^{(\hat{\theta}_2 - \theta_2) - 1},
\]

for \(i = 1, \ldots, n_1\) and \(j = 1, \ldots, n_2\).

The linex loss function (5.16) is an extension of the linex loss function in one dimension in Equation (4.6). It is, in fact, the addition of two linex loss functions in one dimension with parameters \(\theta_1\) and \(\theta_2\). Therefore, we can derive the analytical Bayes estimate of the linex loss function in two dimensions in the same way that we have done in one dimension (see Equations (4.10) - (4.12)). The analytical Bayes estimate under the linex loss function in Equation (5.12) is

\[
(\hat{\theta}_1, \hat{\theta}_2)_{LB} = (-\log(\mathbb{E}_\pi[e^{-\theta_1}]), -\log(\mathbb{E}_\pi[e^{-\theta_2}])).
\]

The linex Bayes estimate based on Monte Carlo method is obtained from

\[
(\hat{\theta}_1, \hat{\theta}_2)_{LM} = (-\log(\frac{1}{N} \sum_{n=1}^{N} e^{-\theta_1^{(n)}}), -\log(\frac{1}{N} \sum_{n=1}^{N} e^{-\theta_2^{(n)}})).
\]

By using Model N2 (5.11) as the posterior distribution, the analytical linex Bayes is

\[
(\hat{\theta}_1, \hat{\theta}_2)_{LB} = (-\log(e^{-\frac{1}{2}}), -\log(e^{-6+\frac{1}{2}})) = (-0.500, 5.500),
\]

and the linex Bayes estimate based on the Monte Carlo method using \(N = 100,000\) sample size is

\[
(\hat{\theta}_1, \hat{\theta}_2)_{LM} = (-\log(\frac{1}{100,000} \sum_{j=1}^{100,000} e^{-\theta_1^{(j)}}), -\log(\frac{1}{100,000} \sum_{j=1}^{100,000} e^{-\theta_2^{(j)}})) = (-0.501, 5.511).
\]
Simulation results of linex Bayes estimates

We have used the ETM method in (5.14) using bicubic interpolation with \((\hat{\theta}_1, \hat{\theta}_2)\) as the extra grid point and found that using just only the quantile grid points, \((n_1, n_2) = (5, 5)\), it is much slower than using the naive method. It turned out that the ETM method with the extra grid point took 942.69 seconds to compute the linex Bayes method \((\hat{\theta}_1, \hat{\theta}_2)^{LP}\) while the naive method in (5.13) took only 275.22 seconds to compute \((\hat{\theta}_1, \hat{\theta}_2)^{LN}\) where both are based on the generated sample size \(N = 100,000\). Therefore, the ETM method with the extra grid point is not an efficient approach to compute Bayes estimates in two dimensions. It might be because we cannot separate the estimation part and the minimisation part entirely when the \((\hat{\theta}_1, \hat{\theta}_2)\) is used as the extra grid point. Therefore, we use the ETM method without adding the extra grid point \((\hat{\theta}_1, \hat{\theta}_2)\) as it is likely to be cheaper than the naive method. This implies that we might not have zero loss at \((\hat{\theta}_1, \hat{\theta}_2)\).

We compare the linex Bayes estimates from the naive method and the ETM method by using the error given by the Euclidean distance between points;

\[
\|\hat{\theta}^{LN} - \hat{\theta}^{LP}\| = \sqrt{(\hat{\theta}_1^{LN} - \hat{\theta}_1^{LP})^2 + (\hat{\theta}_2^{LN} - \hat{\theta}_2^{LP})^2},
\]

where

\[
\hat{\theta}^{LN} = (\hat{\theta}_1^{LN}, \hat{\theta}_2^{LN}) = (\hat{\theta}_1, \hat{\theta}_2)^{LN},
\]

\[
\hat{\theta}^{LP} = (\hat{\theta}_1^{LP}, \hat{\theta}_2^{LP}) = (\hat{\theta}_1, \hat{\theta}_2)^{LP}.
\]

Table 5.1: The linex Bayes estimates \((\hat{\theta}_1, \hat{\theta}_2)^{LN}\) and \((\hat{\theta}_1, \hat{\theta}_2)^{LP}\) with respect to Model N2 with the sample size \(N = 100,000\).

<table>
<thead>
<tr>
<th>The naive method ((\hat{\theta}_1, \hat{\theta}_2)^{LN})</th>
<th>The ETM method ((\hat{\theta}_1, \hat{\theta}_2)^{LP})</th>
<th>The errors (|\hat{\theta}^{LN} - \hat{\theta}^{LP}|)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-0.50343, 5.5045)</td>
<td>(5,5)</td>
<td>(-0.48932, 5.5325)</td>
</tr>
<tr>
<td>10^{-2} (13,9)</td>
<td>(-0.50317, 5.5067)</td>
<td>0.00221</td>
</tr>
<tr>
<td>10^{-3} (16,15)</td>
<td>(-0.50525, 5.4981)</td>
<td>0.00665</td>
</tr>
<tr>
<td>10^{-4} (25,25)</td>
<td>(-0.52255, 5.4635)</td>
<td>0.04524</td>
</tr>
</tbody>
</table>
Table 5.2: The comparison of the used time (seconds) to calculate the linex Bayes estimates $\hat{\theta}_1, \hat{\theta}_2$${}_{LN}$ and $\hat{\theta}_1, \hat{\theta}_2$${}_{LP}$ with respect to Model N2.

<table>
<thead>
<tr>
<th>The naive method</th>
<th>The ETM method using $P(\hat{\theta}_1, \hat{\theta}_2, n_1, n_2)$ ($\theta_1, \theta_2$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(n_1, n_2)</td>
<td></td>
</tr>
<tr>
<td>228.51</td>
<td>28.27</td>
</tr>
<tr>
<td>(5,5)</td>
<td>103.17</td>
</tr>
<tr>
<td>(13,9)</td>
<td>228.23</td>
</tr>
<tr>
<td>(16,15)</td>
<td>618.67</td>
</tr>
<tr>
<td>(25,25)</td>
<td></td>
</tr>
</tbody>
</table>

First of all, we compare the linex Bayes estimate from the naive method $\hat{\theta}_1, \hat{\theta}_2$${}_{LN}$ to the analytical linex Bayes estimate $\hat{\theta}_1, \hat{\theta}_2$${}_{LB}$, and to the linex Bayes estimate based on Monte Carlo method $\hat{\theta}_1, \hat{\theta}_2$${}_{LM}$ by using the errors similar to Equation (5.15) as

$$
\|\hat{\theta}_1{_{LB}} - \hat{\theta}_1{_{LN}}\| = 0.005658171,
$$

$$
\|\hat{\theta}_1{_{LM}} - \hat{\theta}_1{_{LN}}\| = 0.006939373.
$$

We can see the linex Bayes estimate from the naive method $\hat{\theta}_1, \hat{\theta}_2$${}_{LN}$ is accurate to 2 decimal places. In Table 5.1, we found that the good linex Bayes estimate $\hat{\theta}_1, \hat{\theta}_2$${}_{LP}$ is from when $(n_1, n_2) = (13, 9)$ is used because it gives the least error (the error is 0.00221). If we compare this linex Bayes estimate to the analytical linex Bayes estimate $\hat{\theta}_1, \hat{\theta}_2$${}_{LB}$ and the linex Bayes estimate based on Monte Carlo method $\hat{\theta}_1, \hat{\theta}_2$${}_{LM}$, we found that the errors are

$$
\|\hat{\theta}_1{_{LB}} - \hat{\theta}_1{_{LP}}\| = 0.007412078
$$

$$
\|\hat{\theta}_1{_{LM}} - \hat{\theta}_1{_{LP}}\| = 0.004816524.
$$

This linex Bayes estimate $\hat{\theta}_1, \hat{\theta}_2$${}_{LP}$ is also accurate to 2 decimal places.

We expect to have the linex Bayes estimate obtained from using more grid point $(n_1, n_2)$ better than using fewer because of the less error tolerance (Tol). However, it turned out that using $(n_1, n_2) = (13, 9)$ is better than $(n_1, n_2) = (16, 15)$ and $(n_1, n_2) = (25, 25)$. Why does using more grid points not provide a better Bayes estimates? One of the possible reasons for this phenomenon is that when more grid points are used, the rectangles determined by those grid points are smaller, so that we have more empty rectangles. Our strategy to deal with this situation is using the value of $\left(\frac{\hat{\theta}_1 + \hat{\theta}_1 + 1}{2}, \frac{\hat{\theta}_2 + \hat{\theta}_2 + 1}{2}\right)$ as an dummy value to approximate the expected value corresponding to the empty rectangle $[\hat{\theta}_1, \hat{\theta}_1 + 1] \times [\hat{\theta}_2, \hat{\theta}_2 + 1]$. This might not be good enough to approximate $E^{kl}_{ij}$ and hence $E_{n_1}[P(\hat{\theta}_1, \hat{\theta}_2, n_1, n_2)]$. Therefore, using more grid points might not be able to provide a better linex Bayes estimate. An alternative to consider would be that we need a different way to deal with empty rectangles.
In Table 5.2, we found that the ETM method using the number of grid points \((n_1, n_2) = (13, 9)\) is the most desirable choice in this example because it is also faster than using the naive method. We then compare the results of the linex Bayes estimates and the used time from the ETM method using \((n_1, n_2) = (13, 9)\) to the naive method when the sample size \(N\) is large. The results are shown in Table 5.3.

Table 5.3: The comparison of linex Bayes estimates \((\hat{\theta}_1, \hat{\theta}_2)^{LP}\) which is obtained when \((n_1, n_2) = (13, 9)\) is used and the use time based on a generated sample of \((\theta_1, \theta_2)\) from Model N2 with the sample size \(N\).

<table>
<thead>
<tr>
<th>(N)</th>
<th>({(\hat{\theta}_1, \hat{\theta}_2)}^{LN})</th>
<th>({(\hat{\theta}_1, \hat{\theta}_2)}^{LP})</th>
<th>The naive method</th>
<th>The ETM method</th>
</tr>
</thead>
<tbody>
<tr>
<td>100,000</td>
<td>(-0.50343, 5.5045)</td>
<td>(-0.50317, 5.5067)</td>
<td>275.22</td>
<td>103.17</td>
</tr>
<tr>
<td>200,000</td>
<td>(-0.49092, 5.5120)</td>
<td>(-0.49557, 5.5060)</td>
<td>585.26</td>
<td>104.58</td>
</tr>
<tr>
<td>300,000</td>
<td>(-0.49694, 5.5004)</td>
<td>(-0.49759, 5.5047)</td>
<td>833.87</td>
<td>104.29</td>
</tr>
</tbody>
</table>

Using more samples, the naive method makes the linex Bayes estimate closer to the analytical linex Bayes estimate, while for the ETM method it makes almost no difference. However, the naive method tends to be much more expensive while the ETM method is not affected when the large sample size \(N\) is used.

5.8 Bicubic interpolation for the quotient loss function

Define the quotient loss function of two parameters as

\[
L((\hat{\theta}_1, \hat{\theta}_2), (\theta_1, \theta_2)) = 1 - \frac{1}{1 - ((\hat{\theta}_1 - \theta_1)^2 + (\hat{\theta}_2 - \theta_2)^2)}. \tag{5.16}
\]

We use \((\hat{\theta}_1, \hat{\theta}_2)^{QN}\) to denote the linex Bayes estimate obtained from the naive method, and use \((\hat{\theta}_1, \hat{\theta}_2)^{QP}\) to denote the quotient Bayes estimate obtained from the ETM method using the bicubic interpolation approximation.

The naive method :

\[
(\hat{\theta}_1, \hat{\theta}_2)^{QN} = \arg \min_{\hat{\theta}} \mathbb{E}_{\pi}[1 - \frac{1}{1 - ((\hat{\theta}_1 - \theta_1)^2 + (\hat{\theta}_2 - \theta_2)^2)}]. \tag{5.17}
\]

The ETM method :

\[
(\hat{\theta}_1, \hat{\theta}_2)^{QP} = \arg \min_{\hat{\theta}} \mathbb{E}_{\pi}[P_{(\hat{\theta}_1, \hat{\theta}_2), n_1 n_2}(\theta_1, \theta_2)] \tag{5.18}
\]
where $P(\hat{\theta}_1, \hat{\theta}_2), n_1 n_2(\theta_1, \theta_2)$ is in Equation (5.10) with the grid points defined in Definition 5.1, where

$$L_{ij,(\hat{\theta}_1, \hat{\theta}_2)} = 1 - \frac{1}{1 - ((\hat{\theta}_1 - \theta_1)^2 + (\hat{\theta}_2 - \theta_2)^2)},$$

for $i = 1, \ldots, n_1$ and $j = 1, \ldots, n_2$.

The quotient loss function (5.16) is also an extension of the quotient loss function in one dimension in Equation (4.19). It represents a bounded loss function in two dimensions. There is no closed form of the quotient loss function in Equation (5.16). Therefore, we could compare the quotient Bayes $(\hat{\theta}_1, \hat{\theta}_2)^{{QP}}$ obtained from the ETM method (5.18) to the quotient Bayes estimate $(\hat{\theta}_1, \hat{\theta}_2)^{{QN}}$ obtained from the naive method (5.17). We use the error similar to Equation (5.15) to compare the quotient Bayes estimates.

**Simulation results of quotient Bayes estimates**

Table 5.4: The quotient Bayes estimates $(\hat{\theta}_1, \hat{\theta}_2)^{{QN}}$ and $(\hat{\theta}_1, \hat{\theta}_2)^{{QP}}$ with respect to Model N2 with the sample size $N = 100,000$.

| The naive method $(\hat{\theta}_1, \hat{\theta}_2)^{{QN}}$ | The ETM method $(\hat{\theta}_1, \hat{\theta}_2)^{{QP}}$ | The errors $||\hat{\theta}^{{QN}} - \hat{\theta}^{{QP}}||$ |
|--------------------------------------------------|---------------------------------|-----------------------------|
| (0.00230, 6.01300) | N/A (5,5) | (0.01522, 5.3546) | 0.65853 |
| $10^{-2}$ | (9,6) | (0.01453, 6.0748) | 0.06300 |
| $10^{-3}$ | (19,13) | (0.00520, 6.0328) | 0.02001 |
| $10^{-4}$ | (31,18) | (0.00573, 6.0092) | 0.00512 |

Table 5.5: The comparison of the used time (seconds) to calculate the quotient Bayes estimates $(\hat{\theta}_1, \hat{\theta}_2)^{{QN}}$ and $(\hat{\theta}_1, \hat{\theta}_2)^{{QP}}$ with respect to Model N2.

<table>
<thead>
<tr>
<th>The naive method $(\hat{\theta}_1, \hat{\theta}_2)^{{QN}}$</th>
<th>The ETM method using $P(\hat{\theta}_1, \hat{\theta}_2), n_1 n_2(\theta_1, \theta_2)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>68.34</td>
<td>(5,5)</td>
</tr>
<tr>
<td></td>
<td>(9,6)</td>
</tr>
<tr>
<td></td>
<td>(19,13)</td>
</tr>
<tr>
<td></td>
<td>(31,18)</td>
</tr>
</tbody>
</table>

Unlike the linex loss function, using more grid points seems to be better than using fewer because it provides a quotient Bayes estimate $(\hat{\theta}_1, \hat{\theta}_2)^{{QP}}$ closer to $(\hat{\theta}_1, \hat{\theta}_2)^{{QN}}$. Why does using more grid points tends to be the better way to use the ETM method for the quotient loss function but not for the linex loss function? Suspect that the reason is because of the empty rectangles, we examine how many empty rectangles for computing $E^{k_l}_{1j}$ for the linex loss function and the quotient loss function and the results are shown in Table 5.6.
Table 5.6: The number of empty rectangles in bicubic interpolation for the linex loss function and the quotient loss function with the sample size $N = 100,000$.

<table>
<thead>
<tr>
<th>Tol</th>
<th>The linex loss function</th>
<th>The quotient loss function</th>
</tr>
</thead>
<tbody>
<tr>
<td>$10^{-2}$</td>
<td>14</td>
<td>1</td>
</tr>
<tr>
<td>$10^{-3}$</td>
<td>51</td>
<td>22</td>
</tr>
<tr>
<td>$10^{-4}$</td>
<td>201</td>
<td>125</td>
</tr>
</tbody>
</table>

We can see that the quotient loss function has fewer empty rectangles. Therefore, one possible reason is that the number of empty rectangles used in the quotient loss function might not be so big as to make a big effect on the approximation. Although using more grid points is better, when we compare the time used, we found that using $\text{Tol} = 10^{-2}$ which brings the number of grid point $(n_1, n_2) = (9, 6)$ is the best way to use the ETM method for computing the quotient Bayes estimate $(\hat{\theta}_1, \hat{\theta}_2)^{QP}$ as it is faster than the naive method. We then compare the ETM method with the number of grid point $(n_1, n_2) = (9, 6)$ to the naive method if more samples are used in Table 5.7. Similar to the results in the linex loss function in Table 5.3, when the large sample size $N$ is used, the naive method is much more expensive while the ETM method seems to be independent from the sample size.

Table 5.7: The comparison of linex Bayes estimates $(\hat{\theta}_1, \hat{\theta}_2)^{QP}$ which is obtained when $(n_1, n_2) = (9, 6)$ is used and the use time based on a generated sample of $(\theta_1, \theta_2)$ from Model N2 with the sample size $N$.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$(\hat{\theta}_1, \hat{\theta}_2)^{QN}$</th>
<th>$(\hat{\theta}_1, \hat{\theta}_2)^{QP}$</th>
<th>The used time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(0.00230, 6.01300)</td>
<td>(0.01453, 6.0748)</td>
<td>The naive method</td>
</tr>
<tr>
<td>100,000</td>
<td></td>
<td></td>
<td>68.34</td>
</tr>
<tr>
<td>200,000</td>
<td>(0.01010, 6.00131)</td>
<td>(0.02264, 6.0923)</td>
<td>115.75</td>
</tr>
<tr>
<td>300,000</td>
<td>(-0.01040, 5.99995)</td>
<td>(0.01150, 6.1087)</td>
<td>215.87</td>
</tr>
</tbody>
</table>

5.9 Discussion and conclusion

In one dimension, the ETM method via cubic spline interpolation with $\hat{\theta}$ as the extra knot is cheaper than the naive method provided a suitable number of knots. However, in two dimensions, the ETM method via bicubic interpolation with $(\hat{\theta}_1, \hat{\theta}_2)$ as the extra grid points is much more expensive than the naive method. Therefore, in two dimensions, we have used the ETM method without adding the grid point $(\hat{\theta}_1, \hat{\theta}_2)$ because we want to make the ETM method cheaper than the naive one. We have found that the ETM method via bicubic interpolation is a fairly good computational approach to compute Bayes estimates related to the linex loss function and the quotient loss function in two dimensions provided a suitable number of grid points.
An important key to make the ETM method efficient is the number of grid points used in bicubic interpolation. It is essential to find a suitable number of grid points which approximate a loss function good enough to obtain a good Bayes estimates and also make the ETM method faster than the naive method. We have used the quantile grid points and added more points in between these quantile points according to error tolerance (Tol) to find the suitable grid points \((n_1, n_2)\). Nevertheless, the number of grid points has no exact relation to the accuracy of the Bayes estimates. For the linex and the quotient loss functions, we have found that using the error tolerance \(\text{Tol} = 10^{-2}\) is the best choice to determine the number of grid points because it leads to rather better Bayes estimates in terms of accuracy and computational cost. The ETM method using too many grid points is slower than the naive method and might lead to a poorer Bayes estimate. One possible reasons is that it tends to have more empty rectangles when more grid points are used and our strategy used for approximation in the empty rectangles is not suitable.

In conclusion, the ETM method via bicubic interpolation works relatively well to compute the Bayes estimates under the linex loss function and the quotient loss function in two dimensions. However, there are two main concerns of using it. We could not use the extra grid point \((\hat{\theta}_1, \hat{\theta}_2)\) in bicubic interpolation, so we might not have zero loss value at \((\hat{\theta}_1, \hat{\theta}_2)\). Furthermore, we might need to find a better approach to estimate for the empty rectangles.
Chapter 6

Point Estimation in Bayesian Mixture Modelling

6.1 Introduction

Suppose that a random variable or vector, $X$ in a sample space $\mathcal{X}$ has a mixture distribution and that its distribution can be represented by a probability density function (or mass function in the case of discrete random variable $X$) of the form

$$f_\xi(x) = \sum_{j=1}^{k} \omega_j f(x|\theta_j),$$

(6.1)

where $\xi = (\omega_1, \ldots, \omega_k, \theta_1, \ldots, \theta_k)$, $k$ is finite, the mixture weights satisfy

$$\omega_j \geq 0, \quad \omega_1 + \ldots + \omega_k = 1,$$

and

$$\int_{\mathcal{X}} f(x|\theta_j)dx = 1, \quad \text{for } j = 1, \ldots, k.$$

The mixture model (6.1) is attractive for describing complex models or distributions in diverse areas of application (see examples in Titterington and Makov (1985), Chapter 2). However, it might require complex computational techniques for drawing inferences with the mixture model.
For frequentist mixture models, the maximum likelihood approach and the expectation-maximisation (EM) algorithm, provide a plausible approach for statistical inference for the mixture model. The EM algorithm was introduced as an iterative method for fitting finite mixture distributions in Dempster et al. (1977).

From the Bayesian perspective, mixture models have become increasingly popular since the development of posterior simulation techniques, especially Markov chain Monte Carlo (MCMC) methods, Hastings (1970). Various MCMC strategies have been proposed to provide a convenient way to make inferences on complicated mixture model posteriors, Diebolt and Robert (1994), Escobar and West (1995). In Bayesian analysis using a mixture model, one of the major problems is the estimation of the number of components \( k \) in Equation (6.1). For this problem, Marin et al. (2005) suggested some possible solutions such as reversible jump MCMC, Richardson and Green (1997) and birth-and-death processes, Stephens (2000a). However, Bayesian analysis using a mixture model is also often used when the number of components is assumed known. If exchangeable priors or likelihoods are used in the mixture model, then the posterior distribution will be invariant to permutations in the labelling of the parameters. This leads to the problem so-called *label switching*. A method to deal with label switching is to use a label invariant loss function. This method was introduced in Celeux et al. (2000). It is a decision theoretic approach that computes parameter estimates using algorithms that aim to minimise the posterior expectation of loss. This method removes the problem of label switching as the order of components does not matter in the algorithm. However, this method might not be suitable in some contexts where such invariant loss functions makes nonsense of statistical objectives, e.g., finding the parameters of the “first” component. Moreover, this method can be computationally expensive as it often involves complex loss functions and expensive minimisation algorithms such as simulated annealing.

In this chapter, we aim to provide point estimates of parameters for the Normal mixture distribution. The data used for analysis is the data representing velocities of 82 galaxies diverging from our galaxy. The galaxy data were used in Postman et al. (1986) and analysed under different mixture models by many researchers afterwards, for example, Roeder (1990), Richardson and Green (1997), Stephens (2000a), and Jasra et al. (2005). We estimate parameters of the Normal mixture model by following the work in Celeux et al. (2000) that used an invariant loss function for the predictive distribution. By using standard MCMC algorithms, we generate MCMC samples for estimation of the expected loss function and use simulated annealing to deal with the minimisation problem. We also address the question of providing uncertainty of those point estimates.
6.2 Mixture posterior distributions

In this section, we describe Bayesian modelling based on the Normal mixture model and a standard MCMC method for generating a parameter sample. Assume the data \( \mathbf{x} = (x_1, \ldots, x_n) \) are normally independently distributed so that their distributions can be represented by

\[
 f_\xi(x_i) \sim \sum_{j=1}^{k} \omega_j f_j(x_i|\mu_j, \sigma_j^2) = \sum_{j=1}^{k} \omega_j \frac{1}{\sqrt{2\pi \sigma_j^2}} \exp \left\{ -\frac{(x_i - \mu_j)^2}{2\sigma_j^2} \right\}, \tag{6.2}
\]

where \( k \) is fixed and known and \( \omega_j \)'s are the weights, \( \sum_{j=1}^{k} \omega_j = 1, \omega_j \geq 0 \). The subscript \( \xi \) refers to \( \omega, \mu, \sigma^2 \), where

\[
 \omega = (\omega_1, \ldots, \omega_k), \\
 \mu = (\mu_1, \ldots, \mu_k), \\
 \sigma^2 = (\sigma_1^2, \ldots, \sigma_k^2).
\]

If suitable priors are used, the posterior distribution can be sampled using MCMC methods such as the Metropolis-Hastings algorithm. For the Normal mixture model, it is common to use the following priors for the parameters \( \omega, \mu \) and \( \sigma^2 \):

1. \( \omega \sim \text{Dirichlet}(\delta_1, \ldots, \delta_k) \)

\[
p(\omega|k) = \frac{\Gamma(\delta_0)}{\Gamma(\delta_1) \cdots \Gamma(\delta_k)} \prod_{j=1}^{k} \omega_j^{\delta_j - 1}, \tag{6.3}
\]

where \( \delta_0 = \delta_1 + \ldots + \delta_k \).

2. \( \mu_j \sim N(\mu_0, \sigma_0^2) \) for \( j = 1, \ldots, k \), independently

\[
p(\mu_j|k) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp \left\{ -\frac{1}{2\sigma_0^2} (\mu_j - \mu_0)^2 \right\},
\]

so that

\[
p(\mu|k) = \prod_{j=1}^{k} \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp \left\{ -\frac{1}{2\sigma_0^2} (\mu_j - \mu_0)^2 \right\}. \tag{6.4}
\]
3. $\sigma_j^2 \sim \text{InvGamma}(\alpha, \beta)$ for $j = 1, \ldots, k$, independently

$$p(\sigma_j^2 | k) = \frac{\beta^\alpha}{\Gamma(\alpha)} (\sigma_j^2)^{-(\alpha+1)} \exp \left\{ -\frac{\beta}{\sigma_j^2} \right\},$$

so that

$$p(\sigma^2 | k) = \prod_{j=1}^{k} \frac{\beta^\alpha}{\Gamma(\alpha)} (\sigma_j^2)^{-(\alpha+1)} \exp \left\{ -\frac{\beta}{\sigma_j^2} \right\}. \quad (6.5)$$

In mixture modelling, latent variables are often used to represent subpopulations where population membership is not known but is inferred from the data, McLachlan and Peel (2004). We define the variable $z = (z_1, \ldots, z_n)$ as the latent variable used for allocating the components in the mixture model. We label the component $j$ if $z_i = j$, $j \in \{1, \ldots, k\}$ for the $i$th observation.

Suppose each $z_i$ is independently drawn from the distribution

$$\text{Prob}\{z_i = j\} = \omega_j \quad \text{for } j = 1, \ldots, k,$$

so the prior distribution of $z$ could be

$$p(z | k, \omega) = \prod_{i=1}^{n} (\omega_1 I\{z_i = 1\}(z_i) + \omega_2 I\{z_i = 2\}(z_i) + \cdots + \omega_k I\{z_i = k\}(z_i)), \quad (6.6)$$

where $I\{z_i = j\}$ is an indicator function defined by

$$I\{z_i = j\}(z_i) = \begin{cases} 
1 & \text{if } z_i = j, \\
0 & \text{otherwise}.
\end{cases}$$

Define

$$n_j = \sum_{i=1}^{n} I\{z_i = j\}(z_i), \quad \text{for } j = 1, \ldots, k, \quad (6.7)$$

then the prior distribution of $z$ in Equation (6.6) can be written as

$$p(z | k, \omega) = \prod_{j=1}^{k} \omega_j^{n_j}. \quad (6.8)$$
By using the allocation variable $z$, we can express the likelihood function for the Normal mixture distribution in the form

$$l(x|k, z, \omega, \mu, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2_{zi}}} \exp \left\{ -\frac{1}{2\sigma^2_{zi}} (x_i - \mu_{zi})^2 \right\}. \quad (6.9)$$

By Bayes’ Theorem applied to the priors in Equations (6.3), (6.4), (6.5) and (6.8), and the likelihood function in Equation (6.9), the posterior distribution can be expressed in the proportional form as follows:

$$\pi(\omega, z, \mu, \sigma^2 | k, x) \propto p(\omega | k)p(z | k, \omega)p(\mu | k)p(\sigma^2 | k)l(x | k, z, \omega, \mu, \sigma^2)$$

$$= \prod_{j=1}^{k} \omega_j^{\delta_j-1} \times \prod_{j=1}^{k} \omega_j^{n_j} \times \prod_{j=1}^{k} \frac{1}{\sqrt{2\pi\sigma^2_0}} \exp \left\{ -\frac{1}{2\sigma^2_0} (\mu_j - \mu_0)^2 \right\} \times \prod_{j=1}^{k} (\sigma^2_j)^{-(\alpha+1)} \exp \left\{ -\frac{\beta}{\sigma^2_j} \right\} \times \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2_{zi}}} \exp \left\{ -\frac{1}{2\sigma^2_{zi}} (x_i - \mu_{zi})^2 \right\}. \quad (6.10)$$

In the next subsection, we detail the conditional posterior distributions and the algorithms for generating a sample of the parameters $\omega, z, \mu$ and $\sigma^2$. 

### 6.2.1 Conditional posterior distributions and MCMC algorithms

By using suitable priors, the conditional posterior distributions of the parameters $\omega, \mu$ and $\sigma^2$ are the conjugate distributions. Therefore, we can use the standard MCMC method, Gibbs sampling to generate the samples of the parameter $\omega, \mu$ and $\sigma^2$. Meanwhile, we use the Metropolis-Hastings method to generate the sample of the latent variable, $z$ which assists us to allocate the component in the mixture distribution. We derive the conditional posterior distributions from the posterior distribution (6.10) for each parameter as follows:
The conditional posterior distribution of $\omega$:

$$
\pi(\omega|k, z, \mu, \sigma^2, x) \propto \prod_{j=1}^{k} \omega_j^{\delta_j-1} \prod_{j=1}^{k} \omega_j^{n_j} = \prod_{j=1}^{k} \omega_j^{\delta_j+n_j-1},
$$

where $n_j$'s are from Equation (6.7). Therefore, the conditional posterior distribution of $\omega$ is

$$
\omega|k, z, \mu, \sigma^2, x \sim \text{Dirichlet}(\delta_1 + n_1, \ldots, \delta_k + n_k).
$$

As a result, we can use Gibbs sampling to update the sample $\omega$.

**Algorithm 6.1.** Update $\omega^{(t)}$ by using the Gibbs sampling algorithm

Given $k, \omega^{(t-1)}, z^{(t-1)}$,

1. Find $n_j$, corresponding to $z_i = j$ for $i = 1, \ldots, n$ and $j = 1, \ldots, k$.
2. Generate

$$
\omega^{(t)} \sim \text{Dirichlet}(\delta_1 + n_1, \ldots, \delta_k + n_k).
$$

The conditional posterior distribution of $z$:

$$
\pi(z|k, \omega, \mu, \sigma^2, x) \propto \prod_{j=1}^{k} \omega_j^{n_j} \times \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi \sigma^2_{z_i}}} \exp \left\{ -\frac{1}{2\sigma^2_{z_i}} (x_i - \mu_{z_i})^2 \right\}. \quad (6.11)
$$

Mathematically speaking, the conditional posterior distribution of $z_i$ for $i = 1, \ldots, n$ in the form (6.11) is proportional to the Normal density;

$$
\text{Prob}\{z_i = j|k, \omega, \mu, \sigma^2, x\} \propto \frac{1}{\sqrt{2\pi \sigma^2_{z_i}}} \exp \left\{ -\frac{1}{2\sigma^2_{z_i}} (x_i - \mu_{z_i})^2 \right\}.
$$

As a result, we could use the Metropolis-Hasting method with the Normal density as the proposal distribution to update the sample $z$. 

134
Algorithm 6.2. Update \( z^{(t)} \) by using the Metropolis-Hastings algorithm

Given \( k, z^{(t-1)}, \omega^{(t-1)}, \mu^{(t-1)}, \sigma^2^{(t-1)} \)
1. Sample \( z'_i \) from the discrete uniform distribution on the integers \( \{1, 2, \ldots, k\} \) for \( i = 1, \ldots, n \).
2. Take \( z_i^{(t)} = z_i' \) with probability of acceptance, \( \alpha(z_i, z_i') = \min\{1, q\} \), where
   \[
   q = \frac{\omega z'_i}{\omega z_i} \frac{1}{\sqrt{2\pi\sigma^2_{z_i}}} \exp\left\{-\frac{1}{2\sigma^2_{z_i}} (x_i - \mu z'_i)^2\right\} \quad \text{for } i = 1, \ldots, n.
   \]
3. Update \( z^{(t)} = (z_1^{(t)}, \ldots, z_n^{(t)}) \).

The conditional posterior distribution of \( \mu_j \) for \( j = 1, \ldots, k \):

\[
\pi(\mu_j | k, \omega, z, \sigma^2, x) \propto \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left\{-\frac{1}{2\sigma_0^2} (\mu_j - \mu_0)^2\right\} \times \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2_{z_i}}} \exp\left\{-\frac{1}{2\sigma^2_{z_i}} (x_i - \mu z_i)^2\right\} \times \left(\frac{1}{\sqrt{2\pi\sigma^2_j}}\right)^{n_j} \exp\left[-\frac{1}{2\sigma^2_j} \sum_{i=1}^n (x_i - \mu_j)^2\right] \quad \text{where } n_j \text{ is defined in Equation (6.7)}
\]

\[
\propto \exp\left\{-\frac{1}{2\sigma^2_j} (\mu_j^2 + \mu_0^2 - 2\mu_0\mu_j) + \frac{1}{2\sigma^2_j} \sum_{i=1}^n (x_i^2 + \mu_j^2 - 2x_i\mu_j)\right\} \\
= \exp\left\{-\frac{\mu_j^2}{2} \left(\frac{1}{\sigma_0^2} + \frac{n_j}{\sigma^2_j}\right) + \mu_j \left(\frac{\mu_0}{\sigma_0^2} + \frac{\sum_{i=1}^n x_i}{\sigma^2_j}\right) - \left(\frac{\mu_0^2}{2\sigma_0^2} + \frac{\sum_{i=1}^n x_i^2}{2\sigma^2_j}\right)\right\} \\
= \exp\left\{-\frac{1}{2b_j}(\mu_j^2 - 2\mu_j a_j + a_j^2)\right\} = \exp\left\{-\frac{1}{2b_j}(\mu_j - a_j)^2\right\}.
\]

By following the proof in Murphy (2007), we can derive \( a_j \) and \( b_j \) as follows:
Matching coefficients of $\mu_j^2$, we have

\[
-\frac{\mu_j^2}{2b_j} = -\mu_j^2 \left( \frac{1}{\sigma_0^2} + \frac{n_j}{\sigma_j^2} \right)
\]
\[
\frac{1}{b_j} = \frac{1}{\sigma_0^2} + \frac{n_j}{\sigma_j^2}
\]
\[
b_j = \frac{\sigma_0^2 \sigma_j^2}{n_j \sigma_0^2 + \sigma_j^2} = \frac{1}{\sigma_j^2} + \frac{1}{\sigma_0^2}.
\]

(6.12)

Matching coefficients of $\mu_j$, we have

\[
-2\mu_j a_j = \mu_j \left( \frac{\sum_{i=1}^{n} x_i}{\sigma_j^2} + \frac{\mu_0}{\sigma_0^2} \right)
\]
\[
a_j = \frac{\sum_{i=1}^{n} x_i}{\sigma_j^2} + \frac{\mu_0}{\sigma_0^2} = \frac{\sigma_0^2 n_j \bar{x}_j + \sigma_j^2 \mu_0}{\sigma_0^2 \sigma_j^2}
\]
\[
= \left( \frac{n_j}{\sigma_j^2} + \frac{1}{\sigma_0^2} \right)^{-1} \left( \frac{\mu_0}{\sigma_0^2} + \frac{n_j \bar{x}_j}{\sigma_j^2} \right).
\]

(6.13)

Therefore, the conditional posterior of $\mu_j$ is

$$
\mu_j | \omega, z, \sigma^2, k \sim N(a_j, b_j),
$$

where $a_j$ is given by Equation (6.13) and $b_j$ is given by Equation (6.12).

**Algorithm 6.3.** Update $\mu^{(t)}$ by using the Gibbs sampling algorithm

<table>
<thead>
<tr>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Find $n_j$, corresponding to $z_i = j$ for $i = 1, \ldots, n$ and $j = 1, \ldots, k$.</td>
</tr>
<tr>
<td>2. Generate</td>
</tr>
</tbody>
</table>
| $$
\mu_j^{(t)} \sim N(a_j, b_j) \quad \text{for } j = 1, \ldots, k,
$$ |

where $a_j, b_j$ defined in (6.13) and (6.12)

<table>
<thead>
<tr>
<th>Step</th>
</tr>
</thead>
<tbody>
<tr>
<td>3. Update $\mu^{(t)} = (\mu_1^{(t)}, \ldots, \mu_k^{(t)})$.</td>
</tr>
</tbody>
</table>
The conditional posterior distribution of $\sigma_j^2$ for $j = 1, \ldots, k$:

$$
\pi(\sigma_j^2 | k, \omega, z, \mu, x) \propto (\sigma_j^2)^{-(\alpha+1)} \exp\left\{ -\frac{\beta}{\sigma_j^2} \right\} \times \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_{z_i}^2}} \exp\left\{ -\frac{1}{2\sigma_{z_i}^2} (x_i - \mu_{z_i})^2 \right\}
$$

$$
= (\sigma_j^2)^{-(\alpha+1)} \exp\left\{ -\frac{\beta}{\sigma_j^2} \right\} \times (\sigma_{z_i}^2)^{n_j - \alpha} \exp\left\{ -\frac{1}{2\sigma_{z_i}^2} \sum_{i=1}^{n} (x_i - \mu_{z_i})^2 \right\}
$$

$$
= (\sigma_{z_i}^2)^{-(\frac{n_j}{2} + \alpha + 1)} \exp\left\{ -\frac{1}{\sigma_{z_i}^2} \left( \beta + \sum_{i=1}^{n} \frac{(x_i - \mu_{z_i})^2}{2} \right) \right\}.
$$

Therefore, the posterior for $\sigma_j^2$ is

$$
\sigma_j^2 | k, \omega, z, \mu \sim \text{InvGamma} \left( \frac{n_j}{2} + \alpha, \beta + \sum_{i=1}^{n} \frac{(x_i - \mu_{z_i})^2}{2} \right).
$$

Algorithm 6.4. Update $\sigma^{2(t)}$ by using the Gibbs sampling algorithm

Given $k, \omega^{(t-1)}, z^{(t-1)}, \mu^{(t-1)}, \sigma^{2(t-1)}$

1. Find $n_j$, corresponding to $z_i = j$ for $i = 1, \ldots, n$ and $j = 1, \ldots, k$.
2. Generate

$$
\sigma_j^{2(t)} \sim \text{InvGamma} \left( \frac{n_j}{2} + \alpha, \beta + \sum_{i=1}^{n} \frac{(x_i - \mu_{z_i})^2}{2} \right), \text{ for } j = 1, \ldots, k.
$$

3. Update $\sigma^{2(t)} = (\sigma_1^{2(t)}, \ldots, \sigma_k^{2(t)})$.

We arrange the order of updating the parameters $\omega, z$ and $\mu, \sigma^2$ according their algorithms as follows:
Algorithm 6.5. Parameter updating scheme

1. Specify initial values
   \[ \omega^{(0)}, z^{(0)}, \mu^{(0)}, \sigma^2(0). \]

2. Generate \( \omega^{(1)} \) by Gibbs sampling from the conditional posterior distribution
   \[ \omega^{(1)} \sim \pi(\omega^{(1)} | k, z^{(0)}, \mu^{(0)}, \sigma^2(0), x) \]
   (see details in Algorithm 6.1).

3. Generate \( z^{(1)} \) given \( \omega^{(1)}, z^{(0)}, \mu^{(0)}, \sigma^2(0) \) from the Metropolis-Hastings algorithm from the conditional posterior distribution
   \[ z^{(1)} \sim \pi(z^{(1)} | k, \omega^{(1)}, z^{(0)}, \mu^{(0)}, \sigma^2(0), x) \]
   (see details in Algorithm 6.2).

4. Generate \( \mu^{(1)} \) by Gibbs sampling from the conditional posterior distribution,
   \[ \mu^{(1)} \sim \pi(\mu^{(1)} | k, \omega^{(1)}, z^{(1)}, \mu^{(0)}, \sigma^2(0), x) \]
   (see details in Algorithm 6.3).

5. Generate \( \sigma^2(1) \) by Gibbs sampling from the conditional posterior distribution,
   \[ \sigma^2(1) \sim \pi(\sigma^2(1) | k, \omega^{(1)}, z^{(1)}, \mu^{(1)}, \sigma^2(0), x) \]
   (see details in Algorithm 6.4).

6. Repeat step 2 to step 5 to obtain \( \omega^{(2)}, z^{(2)}, \mu^{(2)}, \sigma^2(2) \) and so on.

By following the updating scheme above, we generate the MCMC samples
\[ \{ \omega^1, z^1, \mu^1, \sigma^{2(1)}, \omega^2, z^2, \mu^2, \sigma^{2(2)}, \ldots, \omega^N, \mu^N, z^N, \sigma^{2(N)} \} \].
where

\[ \omega^i = (\omega_1, \ldots, \omega_k)^i, \]
\[ z^i = (z_1, \ldots, z_n)^i, \]
\[ \mu^i = (\mu_1, \ldots, \mu_k)^i, \]
\[ \sigma^{2i} = (\sigma_1^2, \ldots, \sigma_k^2)^i, \]

for \( i = 1, \ldots, N \). As the parameter \( z \) is used to allocate the components \( j = 1, \ldots, k \) in the mixture model, we can use Equation (6.7) which represents the number of \( z_i = j \) for \( i = 1, \ldots, n \) and \( i = 1, \ldots, k \) to study the MCMC samples of \( z \). Note that we will discard the initial 10% samples of runs as the burn-in.

### 6.3 The study of prior sensitivity and simulation results of the MCMC samples

We consider the Normal mixture posterior distribution using the galaxy data set. This data set consists of the velocities of 82 distant galaxy, diverging from our own galaxy. The histogram of the data set is shown in Figure 6.1.

![Histogram of the galaxy data](image)

**Figure 6.1:** Histogram of the galaxy data.
Assume the number of components $k$ is fixed and known. In this thesis, we assume that the number of components $k$ is small enough to make computation not so difficult. It is natural to consider the prior distributions as follows:

$$
\omega \sim \text{Dirichlet}(1, \ldots, 1),
$$

$$
\mu_j \sim \mathcal{N}(0, 100) \quad \text{for } j = 1, \ldots, k,
$$

$$
\sigma^2_j \sim \text{InvGamma}(1, 1) \quad \text{for } j = 1, \ldots, k.
$$

Consider $k = 4$ and 100,000 MCMC runs using the priors in (6.14). The MCMC samples of the parameters $\omega$, $\mu$ and $\sigma^2$ and the values of $n_j$ are shown in Figure 6.2.

We note that there are high values of the MCMC samples of the parameter $\sigma^2$. These values seem unreliable in fitting the Normal mixture model in Equation (6.2) where $x$ is the galaxy data set in Figure 6.1. The reason for this result might be explained as follows.

According to Algorithm 6.3, Algorithm 6.4 and the priors in (6.14), we generate MCMC samples of the parameters $\mu$ and $\sigma^2$ from

$$
\mu_j | k, \omega, z, \sigma^2, x \sim \mathcal{N} \left( \text{mean } = \frac{n_j \bar{x}_j}{0.01\sigma_j^2 + n_j}, \text{var } = \frac{\sigma_j^2}{0.01\sigma_j^2 + n_j} \right),
$$

$$
\sigma^2_j | k, \omega, z, \mu, x \sim \text{InvGamma} \left( \text{shape } = \frac{n_j}{2} + 1, \text{ scale } = 1 + \sum_{i=1}^{n} \frac{(x_i - \mu_i)^2}{2}, \right)
$$

for $j = 1, \ldots, k$, where $n_j$ is defined in Equation (6.7). If there is no data $x_i$ for $i = 1, \ldots, 82$ allocated to the component $k$, $n_j = 0$ for some $j = k$. As a result, we generate MCMC samples of the parameter $\sigma^2_j$ from the prior and generate MCMC samples of the parameter $\mu_j$ with mean zero and variance $100\sigma^2_j$. Furthermore, for $n_j \neq 0$, the generated sample $\mu_j$ might lead to a very high value of $\sigma^2_j$ if there is a big difference between $x_i$ and the generated $\mu_j$ because we generate the sample $\sigma^2_j$ from the Inverse-Gamma distribution whose scale parameter is high. We can see from the plot of probability density functions of the Inverse-Gamma distributions in Figure 6.3 that it is more likely to generate a high value sample $\sigma^2_j$ when the shape parameter is low and the scale parameter is high.
The values of $\eta_j$

Figure 6.2: Trace plots of 100,000 iterations of the MCMC samples using the non-informative priors.
The pdf of the Inverse–Gamma distributions with shape $\alpha$ and scale $\beta$

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{-\alpha-1} e^{-\beta x}$$

for $\alpha > 0$, $\beta > 0$, and $x > 0$.

**Figure 6.3:** The probability density functions of the Inverse-Gamma distributions with different values of the shape and scale parameters.

The simulation results using the priors in (6.14) shown in Figure 6.2 suggest that it might be more sensible to use an informative prior for the parameter $\sigma^2$. As a result, we study the sensitivity of the prior, in particular the Inverse-Gamma distribution. To do so, we simulate MCMC samples by using the same priors for the parameter $\omega$ and $\mu$ but with a different prior for the parameter $\sigma^2$; we use

$$\sigma_j^2 \sim \text{InvGamma}(\alpha, \beta) \quad \text{for } j = 1, \ldots, k.$$  \hspace{1cm} (6.15)

with different values of $\alpha$ and $\beta$.

Note that if $\sigma_j^2 \sim \text{InvGamma}(\alpha, \beta)$, the mean and the variance of $\sigma_j^2$ are

$$\mathbb{E}[\sigma_j^2] = \frac{\beta}{\alpha - 1} \quad \text{for } \alpha > 1$$

$$\text{Var}(\sigma_j^2) = \frac{\beta^2}{(\alpha - 1)(\alpha - 2)^2} \quad \text{for } \alpha > 2,$$

for $j = 1, \ldots, k$. We run MCMC where the prior of $\sigma^2$ is (6.17) with the parameters for the Inverse-Gamma distribution in Table 6.1.
Table 6.1: Prior distributions for $\sigma^2_j$ for $j = 1, \ldots, k$ and the values of means and variances.

<table>
<thead>
<tr>
<th>The priors</th>
<th>$E[\sigma^2_j]$</th>
<th>Var($\sigma^2_j$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvGam(3,1)</td>
<td>0.500</td>
<td>0.500</td>
</tr>
<tr>
<td>InvGam(4,0.5)</td>
<td>0.167</td>
<td>0.0208</td>
</tr>
<tr>
<td>InvGam(5,0.1)</td>
<td>0.025</td>
<td>0.0003</td>
</tr>
</tbody>
</table>

We could investigate the sensitivity of priors by comparing the means (or medians) of the MCMC samples, for example for $j = 1, \ldots, k$,

$$\text{Mean of } \mu_j = \frac{1}{N} \sum_{i=1}^{N} \mu^i_j,$$  \hspace{1cm} (6.16)

where $\mu^i_j$ is the component $j^{th}$ at the $i^{th}$ iteration in a MCMC run. However, the MCMC samples are mixing among the components as shown in Figure 6.2. As a result, using the mean given by Equation (6.16) might be meaningless as we could obtain the same means for all $j = 1, \ldots, k$ if all components are well mixing. We will use histograms of the MCMC samples to see where the means of the samples are without dealing with the mixing of the components in the MCMC samples.

We run 3 MCMC simulations using the priors in (6.14) as well as the Inverse-Gamma distribution with the parameters as shown in Table 6.1. The histograms of the MCMC samples are shown in Figures 6.4 - 6.6. For each row of the plots, we give 3 histograms obtained from the 3 runs of the MCMC where the first row is from using $\sigma^2_j \sim \text{InvGam}(1,1)$, the second row is from $\sigma^2_j \sim \text{InvGam}(3,1)$, the third row is from using $\sigma^2_j \sim \text{InvGam}(4,0.5)$ and the fourth row is from using $\sigma^2_j \sim \text{InvGam}(5,0.1)$ for $j = 1, \ldots, 4$.

The simulation results are explained as follows. In Figure 6.4 which shows the weights when $\sigma^2_j \sim \text{InvGam}(5,0.1)$ (fourth row), for $j = 1, \ldots, 4$, we can see clearly that the means of samples $\omega = (\omega_1, \omega_2, \omega_3, \omega_4)$ are about 0.1 and 0.8. This means that at least one component dominates the model with the high weight and at least one of the 4 components has very small weight. However, this effect becomes less clear when $\alpha$ is small and $\beta$ is high.

In Figure 6.5 which shows the means when $\sigma^2_j \sim \text{InvGam}(5,0.1)$, we see 3 different means of the samples $\mu = (\mu_1, \mu_2, \mu_3, \mu_4)$ from 3 simulations suggesting there is a mixing issue. In contrast, when $\sigma^2_j \sim \text{InvGam}(3,1)$, $\sigma^2_j \sim \text{InvGam}(4,0.5)$ and $\sigma^2_j \sim \text{InvGam}(1,1)$, we are less certain about the means. Nevertheless, we can see say that the means for two components are about 10 and 33, and there might be two components having means between 18 - 25.
In Figure 6.6 which shows the variance when \( \sigma^2_j \sim \text{InvGam}(5,0.1) \), we see 3 different means of the samples \( \sigma^2 = (\sigma^2_1, \sigma^2_2, \sigma^2_3, \sigma^2_4) \) from 3 simulations again suggesting a mixing issue. However, when \( \sigma^2_j \sim \text{InvGam}(3,1) \) and \( \sigma^2_j \sim \text{InvGam}(4,0.5) \), we have fairly similar results that when \( \sigma^2_j \sim \text{InvGam}(3,1) \), it is more likely to have higher values of the samples of \( \sigma^2_j \) than when \( \sigma^2_j \sim \text{InvGam}(4,0.5) \) is used. For \( \sigma^2_j \sim \text{InvGam}(1,1) \), we have very small mean but we could have very high values samples.

In general, we need to study the posterior distribution using a variety of prior distributions for checking prior sensitivity. Here we study the sensitivity to changes in the Inverse-Gamma distribution. Our results suggest that the results are sensitive to the choice of priors and therefore care must be taken when using this setup. Further study on the prior sensitivity analysis is required to determine the extent to which estimation.

In this thesis, we use the priors as follows.

\[
\omega \sim \text{Dirichlet}(1, \ldots, 1), \quad \mu_j \sim N(0, 100) \quad \text{for } j = 1, \ldots, k, \quad \sigma^2_j \sim \text{InvGamma}(4, 0.5) \quad \text{for } j = 1, \ldots, k. \tag{6.17}
\]

We will study point estimates of the Normal mixture model with either \( k = 3 \) or \( k = 4 \). We illustrate simulation results of the generated MCMC samples obtained from using the priors (6.17) with trace plots and histograms with the components \( k = 3 \) and \( k = 4 \) in Figures 6.7 - 6.10.
Figure 6.4: Histograms of the three MCMC samples of $\omega$ (in 3 columns) from using the priors in (6.17) with $\sigma_j^2 \sim \text{InvGam}(1,1)$ (first row), $\sigma_j^2 \sim \text{InvGam}(3,1)$ (second row), $\sigma_j^2 \sim \text{InvGam}(4,0.5)$ (third row) and $\sigma_j^2 \sim \text{InvGam}(5,0.1)$ (fourth row).
Figure 6.5: Histograms of the three MCMC samples of $\mu$ (in 3 columns) from using the priors in (6.17) with $\sigma^2_j \sim \text{InvGam}(1,1)$ (first row), $\sigma^2_j \sim \text{InvGam}(3,1)$ (second row), $\sigma^2_j \sim \text{InvGam}(4,0.5)$ (third row) and $\sigma^2_j \sim \text{InvGam}(5,0.1)$ (fourth row).
Figure 6.6: Histograms of the three MCMC samples of $\sigma^2$ (in 3 columns) from using the priors in (6.17) with $\sigma_j^2 \sim \text{InvGam}(1,1)$ (first row), $\sigma_j^2 \sim \text{InvGam}(3,1)$ (second row), $\sigma_j^2 \sim \text{InvGam}(4,0.5)$ (third row) and $\sigma_j^2 \sim \text{InvGam}(5,0.1)$ (fourth row).
For $k = 3$, the MCMC samples of the parameter $\mu$ in Figure 6.7 are clearly grouped into 3 components. For the trace plots in Figure 6.8, we want to see mixing of components as it can imply that we have explored all possible labellings of the parameters of the mixture model. Nevertheless, for $k = 3$, we hardly see that the samples are mixing among the components. One possible reason is that there are clearly 3 modes of the likelihood which are well separated so it might be difficult to make a sample jump up to the other components.

For $k = 4$, the MCMC samples of the parameter $\mu$ in Figure 6.9 seem to be grouped into 4 components but they are not as clear as in $k = 3$. For the trace plots in Figure 6.10, we can see mixing of the components. There are jumps of the samples among the components. As we mentioned earlier the mixing of the components in MCMC output which is also known as label switching could make the mean in Equation (6.16) meaningless. In the next section, we will discuss how to make inference on the parameters in the mixture posterior distribution without dealing with the label switching problem.
Figure 6.7: Histograms of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma_j^2$ for $j = 1, \ldots, 3$. 
Figure 6.8: Trace plots of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma_j^2$ for $j = 1, \ldots, 3$. 
Figure 6.9: Histograms of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma_j^2$ for $j = 1, \ldots, 4$. 
Figure 6.10: Trace plots of 100,000 iterations of the MCMC samples using the informative prior for the parameter $\sigma_j^2$ for $j = 1, \ldots, 4$. 
6.4 Inference for the mixture posterior distribution

In this section, we discuss point estimation of parameters $\xi = (\omega, \mu, \sigma^2)$ based on the Normal mixture distribution (6.2) using the MCMC sample generated from the algorithms described in the previous section. As label switching of the components is needed for MCMC convergence, it is inevitably difficult to make inference for the mixture posterior distribution. The term label switching was initially introduced for describing the invariance of the likelihood under relabelling of the mixture components, Redner and Walker (1984). The label switching in MCMC output makes ergodic averages of component specific quantities identical so it is impossible to form ergodic averages over labels, Celeux et al. (2000). We have shown that the MCMC samples of the parameters $\omega$, $\mu$ and $\sigma^2$ are mixing among the components $k$ (see Figures 6.2 and 6.10), so it is impossible to label the components. As a result, common practice for estimating parameters using the ergodic average of the MCMC samples is meaningless. For instance, we might not be able to use

$$\hat{\mu} \approx \frac{1}{N} \sum_{i=1}^{N} (\mu_1, \ldots, \mu_k)^i,$$

as the point estimate of the parameter $\mu$ because the MCMC samples $\mu_1, \ldots, \mu_k$ are mixing among the components $j = 1, \ldots, k$. A naive approach to cope with the label switching problem is to impose artificial identifiability constraints on the parameters for example by ordering the means in a normal mixture model ($\mu_1 < \ldots < \mu_k$). In fact, imposing the identifiability constraint is equivalent to changing the prior distribution because the prior distribution is limited by the ordering ($\mu_1 < \ldots < \mu_k$). It has been suggested that the ordering constraints can be carried out after the MCMC simulations have been completed provided the priors are exchangeable, Stephens (2000b). However, computing an average under the constraint may produce a value that is unrelated to the modes of the posterior, Marin et al. (2005).

Using a label invariant loss function has been suggested in Celeux et al. (2000) to deal with label switching. In general, it requires less programming effort than the relabelling method as discussed in Jasra et al. (2005). In this thesis, we consider the invariant loss function for the predictive distribution given by

$$L(\hat{\xi}, \xi) = \int_{\mathbb{R}} (f_{\hat{\xi}}(y) - f_{\xi}(y))^2 dy,$$  

(6.18)

where $f_{\xi}$ denotes the Normal mixture density (6.2). The loss function in Equation (6.18) is called the integrated squared difference loss function. We can see that by considering the loss function (6.18), the order of components in the parameter $\xi$ does
The point estimate of the Normal mixture model is then obtained as
\[ \hat{\xi}^* = \arg\min_{\xi} \mathbb{E}_\pi \left[ L(\hat{\xi}, \xi) \right]. \tag{6.19} \]

### 6.4.1 The ETM method for the point estimate of the parameter \( \xi \)

Assume that the order of integration can be interchanged, so the expected posterior loss function can be written in the form
\[ \mathbb{E}_\pi \left[ L(\hat{\xi}, \xi) \right] = \int_\mathcal{R} \mathbb{E}_\pi \left[ (f_{\hat{\xi}}(y) - f_{\xi}(y))^2 \right] dy, \tag{6.20} \]

The expected loss function in Equation (6.20) allows us to implement the ETM method by using its decomposed form as follows.
\[ \mathbb{E}_\pi \left[ L(\hat{\xi}, \xi) \right] = \int_\mathcal{R} f_{\hat{\xi}}(y)^2 dy - 2 \int_\mathcal{R} f_{\hat{\xi}}(y) \mathbb{E}_\pi [f_{\xi}(y)] dy + \int_\mathcal{R} \mathbb{E}_\pi [f_{\xi}(y)^2] dy \tag{6.21} \]

For the estimation part, we approximate the expectations by using the MCMC samples \( \xi^i = (\omega^i, \mu^i, \sigma^2_i) \) for \( i = 1, \ldots, N \), with ergodic averaging,
\[ \mathbb{E}_\pi [f_{\xi}(y)] \approx \frac{1}{N} \sum_{i=1}^{N} f_{\xi^i}(y), \tag{6.22} \]
\[ \mathbb{E}_\pi [f_{\xi}(y)^2] \approx \frac{1}{N} \sum_{i=1}^{N} f_{\xi^i}(y)^2. \tag{6.23} \]

We can use the numerical integration of the function to evaluate the integrals. In this thesis, we use the Simpson’s 1/3 rule as the numerical integration that can be explained as follows.

Suppose that the interval \([a, b] \in \mathcal{R}\) is divided into \( n \) subintervals, with \( n \) an even number. The composite Simpson’s rule is given by
\[ \int_a^b f(y) dy \approx \frac{h}{3} \left( f(y_0) + 2 \sum_{i=1}^{n/2-1} f(y_{2i}) + 4 \sum_{i=1}^{n/2} f(y_{2i-1}) + f(y_n) \right), \tag{6.24} \]
where \( h = (b - a)/n \) and \( y_j = a + jh \); for \( j = 0, 1, \ldots, n; \); see more details of the Simpson’s 1/3 rule in Atkinson and Han (2004).
For the minimisation part, we use the simulated annealing algorithm as the minimisation method with a geometric annealing schedule given by

\[ T_t = \rho T_{t-1}, \]

where \( T_0 = 10^{-5}, T_f = 10^{-8} \) and \( \rho = \left( \frac{T_f}{T_0} \right)^{1/M} \), where \( M = 1,000 \) is the number of iterations of simulated annealing (see Figure 6.11). Note that this annealing schedule is not the default setting in R.

![The annealing schedule](image)

Figure 6.11: Plot of the geometric annealing (cooling) schedule.

The following algorithm shows implementation of the simulated annealing (SA) algorithm to search for the point estimate \( \hat{\xi}^* \).

**Algorithm 6.6.** SA algorithm for the point estimates \( \hat{\xi}^* \)

1. Choose an arbitrary initial point, \( \hat{\xi}^{(0)} = (\hat{\omega}^{(0)}, \hat{\mu}^{(0)}, \hat{\sigma}^2(0)) \) and the temperature \( T^0 > 0 \).
2. At iteration \( t \), \( \hat{\xi}^{(t)} = (\hat{\omega}^{(t)}, \hat{\mu}^{(t)}, \hat{\sigma}^2(0)) \).
   (a) Update \( \hat{\omega}^{(t)} \) by:
3. Update

- sample $\hat{\omega}'$ from a distribution on the neighbourhood of $\hat{\omega}^{(t)}$ (see Algorithm B.1),
- use $\hat{\xi}_{\hat{\omega}} = (\hat{\omega}', \hat{\mu}^{(t)}, \hat{\sigma}^{2(t)})$ to estimate $\mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\omega}}, \xi) \right]$.
- accept $\hat{\omega}^{(t+1)} = \hat{\omega}'$ with probability
  \[
  \rho^t_{\hat{\omega}} = \min\{\exp(-\Delta \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\omega}}, \xi) \right] / T^t), 1\},
  \]
  where $\Delta \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\omega}}, \xi) \right] = \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\omega}}, \xi) \right] - \mathbb{E}_\pi \left[ L(\hat{\xi}^{(t)}, \xi) \right]$ otherwise, take $\hat{\omega}^{(t+1)} = \hat{\omega}^{(t)}$.

(b) Use $\hat{\xi}^{(t)} = (\hat{\omega}^{(t+1)}, \hat{\mu}^{(t)}, \hat{\sigma}^{2(t)})$ and update $\hat{\mu}^{(t)}$ by:
- sample $\hat{\mu}'$ from a distribution on the neighbourhood of $\hat{\mu}^{(t)}$ (see Algorithm B.2),
- use $\hat{\xi}_{\hat{\mu}} = (\hat{\omega}^{(t+1)}, \hat{\mu}', \hat{\sigma}^{2(t)})$ to estimate $\mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\mu}}, \xi) \right]$.
- accept $\hat{\mu}^{(t+1)} = \hat{\mu}'$ with probability
  \[
  \rho^t_{\hat{\mu}} = \min\{\exp(-\Delta \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\mu}}, \xi) \right] / T^t), 1\},
  \]
  where $\Delta \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\mu}}, \xi) \right] = \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\mu}}, \xi) \right] - \mathbb{E}_\pi \left[ L(\hat{\xi}^{(t)}, \xi) \right]$ otherwise, take $\hat{\mu}^{(t+1)} = \hat{\mu}^{(t)}$.

(c) Use $\hat{\xi}^{(t)} = (\hat{\omega}^{(t+1)}, \hat{\mu}^{(t+1)}, \hat{\sigma}^{2(t)})$ and update $\hat{\sigma}^{2(t)}$ by:
- sample $\hat{\sigma}^{2'}$ from a distribution on the neighbourhood of $\hat{\sigma}^{2(t)}$ (see Algorithm B.3),
- use $\hat{\xi}_{\hat{\sigma}^{2}} = (\hat{\omega}^{(t+1)}, \hat{\mu}^{(t+1)}, \hat{\sigma}^{2(t)})$ to estimate $\mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\sigma}^{2}}, \xi) \right]$.
- accept $\hat{\sigma}^{2(t+1)} = \hat{\sigma}^{2'}$ with probability
  \[
  \rho^t_{\hat{\sigma}^{2}} = \min\{\exp(-\Delta \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\sigma}^{2}}, \xi) \right] / T^t), 1\},
  \]
  where $\Delta \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\sigma}^{2}}, \xi) \right] = \mathbb{E}_\pi \left[ L(\hat{\xi}_{\hat{\sigma}^{2}}, \xi) \right] - \mathbb{E}_\pi \left[ L(\hat{\xi}^{(t)}, \xi) \right]$ otherwise, take $\hat{\sigma}^{2(t+1)} = \hat{\sigma}^{2(t)}$.

3. Update $T^t = T^{t+1}$.

4. Repeat 2 – 3.
6.4.2 Simulation results

The simulation results of point estimates of the parameter $\omega, \mu$ and $\sigma^2$ denoted by $\hat{\omega}^*, \hat{\mu}^*$ and $\hat{\sigma}^{2*}$ of the mixture model with the components $k = 3$ and $k = 4$ are presented in Tables 6.2 and 6.3, respectively.

Table 6.2: The point estimates of the parameters of the Normal mixture with the components $k = 3$ based on $N$ MCMC samples.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\hat{\omega}^* = (\hat{\omega}_1^<em>, \hat{\omega}_2^</em>, \hat{\omega}_3^*)$</th>
<th>$\hat{\mu}^* = (\hat{\mu}_1^<em>, \hat{\mu}_2^</em>, \hat{\mu}_3^*)$</th>
<th>$\hat{\sigma}^{2*} = (\hat{\sigma}_1^{2*}, \hat{\sigma}_2^{2*}, \hat{\sigma}_3^{2*})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50,000</td>
<td>(0.048, 0.859, 0.093)</td>
<td>(32.97, 21.40, 9.707)</td>
<td>(0.555, 4.466, 0.236)</td>
</tr>
<tr>
<td>100,000</td>
<td>(0.048, 0.092, 0.860)</td>
<td>(32.98, 9.705, 21.40)</td>
<td>(0.509, 0.221, 4.453)</td>
</tr>
</tbody>
</table>

Table 6.3: The point estimates of the parameters of the Normal mixture with the components $k = 4$ based on $N$ MCMC samples.

<table>
<thead>
<tr>
<th>$N$</th>
<th>$\hat{\omega}^* = (\hat{\omega}_1^<em>, \hat{\omega}_2^</em>, \hat{\omega}_3^<em>, \hat{\omega}_4^</em>)$</th>
<th>$\hat{\mu}^* = (\hat{\mu}_1^<em>, \hat{\mu}_2^</em>, \hat{\mu}_3^<em>, \hat{\mu}_4^</em>)$</th>
<th>$\hat{\sigma}^{2*} = (\hat{\sigma}_1^{2*}, \hat{\sigma}_2^{2*}, \hat{\sigma}_3^{2*}, \hat{\sigma}_4^{2*})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>50,000</td>
<td>(0.047, 0.675, 0.186, 0.092)</td>
<td>(33.09, 21.85, 19.73, 9.705)</td>
<td>(0.567, 4.625, 0.247, 0.236)</td>
</tr>
<tr>
<td>100,000</td>
<td>(0.047, 0.094, 0.693, 0.166)</td>
<td>(33.02, 9.683, 21.84, 19.73)</td>
<td>(0.544, 0.250, 4.545, 0.289)</td>
</tr>
</tbody>
</table>

We use two MCMC runs with the number of samples $N = 50,000$ and $N = 100,000$ for both $k = 3$ and $k = 4$. We found that using $N = 50,000$ gives different point estimates from using $N = 100,000$. Without labelling the order of the components, we could match the point estimates with the nearest values to compare the point estimates in both MCMC runs. For example, for $k = 3$, we could compare the point estimates in the second component from $N = 50,000$ with the third component from $N = 100,000$. We can see that they are quite similar. Similarly, for $k = 4$, we also found that the point estimates from both runs are similar. We could compare the results from both runs by substituting these point estimates in the mixture density where the order of the component does not matter. Therefore, we approximate the mixture density by

$$ f_{\hat{\xi}^*}(y) = \frac{1}{\sqrt{2\pi \hat{\sigma}_j^{2*}}} \exp \left\{ -\frac{(y - \hat{\mu}_j^*)^2}{2\hat{\sigma}_j^{2*}} \right\}, \quad \text{for } j = 1, \ldots, k = 3 \text{ and } 4, $$

where the point estimates $\xi^* = (\omega^*, \mu^*, \sigma^{2*})$ from Tables 6.2 - 6.3. We then plot the approximation overlaid the histogram of the galaxy data as shown in Figure 6.12.
Figure 6.12: The approximated mixture densities with the component \( k = 3 \) (top) and \( k = 4 \) (bottom) evaluated at the point estimate \( \hat{\xi}^* \).
We can see that the point estimates \( \hat{\xi}^* = (\hat{\omega}^*, \hat{\mu}^*, \hat{\sigma}^2) \) from both MCMC runs fit the Normal mixture model reasonably well. For \( k = 3 \), using \( N = 50,000 \) and \( N = 100,000 \) gives almost the same mixture densities while for \( k = 4 \), there is small different between these two runs.

We have shown that we could find the point estimates of the parameters in the mixture model without dealing with label switching in MCMC output by using the invariant loss function in Equation (6.18) following the work of Celeux et al. (2000). In this thesis, we aim to extend their work by presenting the uncertainty of these point estimates. It is important to specify how reliable these point estimates are. In general, point estimation is usually combined with interval estimation such as confidence interval to show the uncertainty of point estimates. However, it might be meaningless to state the confidence interval of each component of the parameter, \( \omega, \mu \) and \( \sigma^2 \) individually because of label switching in components of the MCMC samples. Therefore, 95% confidence intervals of \((\omega_1, \ldots, \omega_k), (\mu_1, \ldots, \mu_k), (\sigma_1^2, \ldots, \sigma_k^2)\) provided by the MCMC samples are incapable of assessing uncertainty. In the next section, we present how to assess uncertainty without dealing with label switching in MCMC samples.

6.5 The uncertainty of the point estimate of the mixture model

In this section, we describe how we might assess uncertainty of the point estimate of the mixture model presented in the previous section. In Bayesian statistics, we could use Bayesian credible intervals to present the uncertainty of point estimates by using the 2.5% and 97.5% quantiles of the MCMC samples. For example, for \( k = 3 \) if the MCMC samples behave similar to the MCMC samples in Figure 6.8, we could take the means as the point estimates of the samples and hence we could use the credible intervals to present the uncertainty of the point estimates as shown in Table 6.4.
Table 6.4: The sample means and the credible intervals constructed by using the 2.5% and 97.5% quantiles of the simulated samples with $N = 10,000$ for $k = 3$.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Components</th>
<th>Means</th>
<th>Credible intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\omega$</td>
<td>1</td>
<td>0.047</td>
<td>(0.013, 0.099)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.094</td>
<td>(0.042, 0.163)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.859</td>
<td>(0.779, 0.922)</td>
</tr>
<tr>
<td>$\mu$</td>
<td>1</td>
<td>33.00</td>
<td>(32.23, 33.77)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>9.703</td>
<td>(9.215, 10.21)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>21.40</td>
<td>(21.25, 21.57)</td>
</tr>
<tr>
<td>$\sigma^2$</td>
<td>1</td>
<td>0.444</td>
<td>(0.171, 1.098)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.208</td>
<td>(0.090, 0.471)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4.518</td>
<td>(3.295, 6.205)</td>
</tr>
</tbody>
</table>

However, due to the label switching problem in the MCMC samples, we might not be able to use 2.5% and 97.5% quantiles of the MCMC samples to assess the uncertainty. As we are not certain about the labelling of the point estimates, it is difficult to provide uncertainty using the 2.5% and 97.5% quantiles values. Similarly, we cannot provide 95% confidence intervals due to the label switching. For example, confidence intervals for the point estimate $\hat{\mu}^* = (\hat{\mu}_1^*, \hat{\mu}_2^*, \hat{\mu}_3^*, \hat{\mu}_4^*)$ are

$$
\hat{\mu}_1^* \pm 1.96 \times SE_1, \hat{\mu}_2^* \pm 1.96 \times SE_2, \hat{\mu}_3^* \pm 1.96 \times SE_3, \hat{\mu}_4^* \pm 1.96 \times SE_4
$$

if $\hat{\mu}_i^*$ are normally distributed and $SE_i$ for $i = 1, \ldots, 4$ are the standard errors of the point estimates $\hat{\mu}_1^*, \hat{\mu}_2^*, \hat{\mu}_3^*$ and $\hat{\mu}_4^*$, respectively. When the MCMC samples are mixing among the components, it is impossible to estimate the standard errors of the point estimates $\hat{\mu}_i^*$ for $i = 1, \ldots, 4$. We therefore try to construct a confidence interval of the point estimate regardless of component labelling. As the expected loss in the form (6.21) is label invariant, and it is a function of the point estimate $\hat{\xi} = (\hat{\omega}, \hat{\mu}, \hat{\sigma}^2)$, we will construct a confidence interval of the expected loss function evaluated at the point estimate $\hat{\xi}$. By this method, we could present how uncertainty of the point estimate $\hat{\xi}$ affects the expected loss function. This means that we might be more certain about the point estimate $\hat{\xi}$ when we have a small confident interval of the expected loss evaluated at $\hat{\xi}$.
The uncertainty of the point estimates $\hat{\xi}$ for the mixture model discussed here could be classified into two types. As they are obtained from the two simulation-based methods, there are two main sources of uncertainty: one is from estimation using MCMC samples and another is from minimisation using simulated annealing. Simulated annealing is not deterministic, each iteration tells us something about the function we are minimising potential. Suppose we implement simulated annealing with $M$ iterations. Therefore, we have $M$ candidate values of the Bayes estimates denoted by $\hat{\xi}^{(1)}, \hat{\xi}^{(2)}, \ldots, \hat{\xi}^{(M)}$. For example, the values of expected loss corresponding to $\hat{\xi}^{(j)}$ based on $N = 50,000$ MCMC samples as shown in Figure 6.13 for the components $k = 3$ and $k = 4$. We could say that the point estimate is chosen from $\hat{\xi}^{(j)}$ for some $j \in 1, \ldots, M$ which gives the minimum expected loss value and denoted by $\hat{\xi}^*$. It would be interesting to assess the uncertainty of $\hat{\xi}^*$ compared the other candidates of the Bayes estimates, $\hat{\xi}^{(j)}$’s generated by the simulated annealing algorithm. Our idea of assessing uncertainty is to find the lower bound and upper bound of the expected loss value evaluated at all possible values point estimate $\hat{\xi}$. By considering the point estimates generated by the simulated annealing algorithm, we then can present some form of uncertainty of the point estimates. To do so, we construct the uncertainty bound which is the approximation of the 95% confidence interval of the expected loss evaluated at each point estimate $\hat{\xi}^{(j)}$ for $i = 1, \ldots, M$. We discuss how to construct the uncertainty bound in details in the next subsection.
The approximation of $E[\text{Loss}]$ evaluated at $\hat{\xi}$ at each SA iteration.

Figure 6.13: The expected loss values based on $N = 50,000$ MCMC samples evaluated at each iteration from the SA algorithm of the mixture model with the components $k = 3$ (top) and $k = 4$ (bottom).
6.5.1 Uncertainty bounds of the expected loss function evaluated at point estimates

Treat the variable $\hat{\xi}$ as a known value and rewrite the expected loss function as the function of $\xi$,

$$h_\xi(\xi) = \mathbb{E}_\pi[L(\hat{\xi}, \xi)] = \mathbb{E}_\pi \int_R (f_\xi(y) - f_\xi(y))^2 dy,$$

so when we assume that the order of integration can be interchanged and the point estimate $\hat{\xi}$ is known,

$$\overline{h_\xi(\xi)} \approx \int_R \frac{1}{N} \sum_{i=1}^N (f_\xi(y) - f_\xi^i(y))^2 dy,$$

where the subscript $\hat{\xi}$ is used to denote the point estimate which is evaluated. In other words, the expected loss function in Equation (6.21) can be estimated by the ergodic average using the MCMC sample $\xi$, so

$$\mathbb{E}_\pi[L(\hat{\xi}, \xi)] \approx \overline{h_\xi(\xi)}.$$

If the sampled $\xi^i$ were independent, so we would estimate the variance of $\overline{h_\xi(\xi)}$ by

$$\text{Var}(\overline{h_\xi(\xi)}) = \frac{1}{N^2} \sum_{i=1}^N (h_\xi(\xi^i) - \overline{h_\xi(\xi)})^2,$$

where $N$ is the number of samples $\xi$. However, the MCMC samples, $\xi^i$ are correlated. The autocorrelation in the MCMC samples makes the estimated variance (6.27) likely to be underestimated. One possible way to correct the estimated variance is using the effective sample size (ESS) given by

$$\text{ESS} = N/\tau(h_\xi(\xi)),$$

where $\tau(h_\xi(\xi))$ is the autocorrelation time for $h_\xi(\xi)$ (see more details Subsection 2.6.3). The variance estimate for $\overline{h_\xi(\xi)}$ is approximated by

$$\text{Var}(\overline{h_\xi(\xi)}) = \frac{1}{N \times \text{ESS}} \sum_{i=1}^N (h_\xi(\xi^i) - \overline{h_\xi(\xi)})^2.$$
As a result, we could approximate a 95% confidence interval of the expected loss function evaluated at \( \hat{\xi} \) by

\[
\bar{h}_\xi(\xi) \pm 1.96 \sqrt{\frac{1}{N \times \text{ESS}} \sum_{i=1}^{N} (h_\xi(\xi_i) - \bar{h}_\xi(\xi))^2}.
\]  

(6.28)

Computing the expected loss function (6.26) is quite expensive because we need to do estimation for each \( \hat{\xi} \) we want to evaluate. By following the idea from the ETM method, we could reduce the cost by using the expected loss function in the decomposed form (6.21). The decomposed form of the expected loss function allows us to do estimation once for all point estimate values \( \hat{\xi} \). However, we cannot use the expected loss function in the decomposed form to construct the confidence interval in (6.28) because the MCMC samples \( \xi^i \) are used to estimate \( E_\pi[f_\xi(y)] \) and \( E_\pi[f_\xi(y)^2] \) not the expected loss function. In the next subsection, we detail how to approximate the uncertainty using the decomposed form of the expected loss function.

### 6.5.2 Crude uncertainty bounds of the expected loss function evaluated at point estimates

Define the decomposed form of the expected loss function,

\[
g_\xi(\xi) = \mathbb{E}_\pi \left[ L(\hat{\xi}, \xi) \right] = \int_{\mathcal{R}} f_\xi(y)^2 dy - 2 \int_{\mathcal{R}} f_\xi(y) \mathbb{E}_\pi[f_\xi(y)] dy + \int_{\mathcal{R}} \mathbb{E}_\pi[f_\xi(y)^2] dy.
\]

We have \( f_\xi(y) \) and \( f_\xi(y)^2 \) that are approximated by the ergodic averages,

\[
\mathbb{E}_\pi[f_\xi(y)] \approx \bar{f}_\xi(y) = \frac{1}{N} \sum_{i=1}^{N} f_\xi_i(y),
\]

\[
\mathbb{E}_\pi[f_\xi(y)^2] \approx \bar{f}_\xi(y)^2 = \frac{1}{N} \sum_{i=1}^{N} f_\xi_i(y)^2.
\]

We can estimate variances of \( f_\xi(y) \) and \( f_\xi(y)^2 \) by

\[
\text{Var}(f_\xi(y)) = \frac{1}{N \times \text{ESS}_1} \sum_{i=1}^{N} (f_\xi_i(y) - \bar{f}_\xi(y))^2,
\]

and

\[
\text{Var}(f_\xi(y)^2) = \frac{1}{N \times \text{ESS}_2} \sum_{i=1}^{N} (f_\xi_i(y)^2 - \bar{f}_\xi(y)^2)^2.
\]
By using variances of $\bar{f}_\xi(y)$ and $\bar{f}_\xi(y)^2$, we could possibly approximate the 95% confidence interval of the expected loss function evaluated at $\hat{\xi}$ by considering the 95% lower and upper pointwise limits of $\bar{f}_\xi(y)$ and $\bar{f}_\xi(y)^2$. Then we have

$$\bar{f}_\xi(y)^L \leq \bar{f}_\xi(y) \leq \bar{f}_\xi(y)^U,$$

$$\bar{f}_\xi(y)^2L \leq \bar{f}_\xi(y)^2 \leq \bar{f}_\xi(y)^2U,$$

where

$$\bar{f}_\xi(y)^L = \bar{f}_\xi(y) - 1.96\sqrt{\text{Var}(\bar{f}_\xi(y))}, \quad \bar{f}_\xi(y)^2L = \bar{f}_\xi(y)^2 - 1.96\sqrt{\text{Var}(\bar{f}_\xi(y)^2)} \quad (6.29)$$

$$\bar{f}_\xi(y)^U = \bar{f}_\xi(y) + 1.96\sqrt{\text{Var}(\bar{f}_\xi(y))}, \quad \bar{f}_\xi(y)^2U = \bar{f}_\xi(y)^2 + 1.96\sqrt{\text{Var}(\bar{f}_\xi(y)^2)}. \quad (6.30)$$

The approximated 95% confidence intervals from Equations (6.29) and (6.30) only present the uncertainty of the approximated expected loss function $\hat{g}_\xi(\hat{\xi})$ implicitly. However, we could use them to construct the uncertainty bound as follows:

$$\hat{g}_\xi(\xi) \in (\hat{g}_\xi(\xi)^L, \hat{g}_\xi(\xi)^U), \quad (6.31)$$

where

$$\hat{g}_\xi(\xi)^L = \int_\mathcal{R} \bar{f}_\xi(y)^2 \, dy - 2 \int_\mathcal{R} \bar{f}_\xi(y) \bar{f}_\xi(y)^U \, dy + \int_\mathcal{R} \bar{f}_\xi(y)^2L \, dy,$$

$$\hat{g}_\xi(\xi)^U = \int_\mathcal{R} \bar{f}_\xi(y)^2 \, dy - 2 \int_\mathcal{R} \bar{f}_\xi(y) \bar{f}_\xi(y)^L \, dy + \int_\mathcal{R} \bar{f}_\xi(y)^2U \, dy.$$

The uncertainty bound of the expected loss function evaluated at the point estimate $\hat{\xi}$ given by (6.31) is actually a crude bound because we use the ultimate limits of the expected functions $\bar{f}_\xi(y)$ and $\bar{f}_\xi(y)^2$ to obtain the lower and upper bounds.

### 6.5.3 Simulation results of uncertainty bounds

We have presented two ways to assess the uncertainty of the point estimate for a mixture model. They both use the expected loss value evaluated at the point estimate $\hat{\xi}$ in the simulated annealing algorithm. As there are two expressions for the expected loss function, we have two ways to present the uncertainty. The (computationally) expensive way is to use the uncertainty bound which is approximated by the 95% confidence interval (6.28) and the cheap way is to use the crude uncertainty bound (6.31).
If we computed the expected loss evaluated at $\hat{\xi}^*$ based on MCMC with $N = 10,000$ for $k = 3$, we have

$$E_\pi[L(\hat{\xi}^*, \xi)] = \bar{h}_{\hat{\xi}^*}(\xi) = g_{\hat{\xi}^*}(\xi) \approx 0.00466.$$  

The two types of uncertainty bounds are shown in Table 6.5.

Table 6.5: The approximated 95% confidence interval and the crude bound evaluated at $\hat{\xi}^*$ based on MCMC with $N = 10,000$ for $k = 3$.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Intervals</th>
</tr>
</thead>
<tbody>
<tr>
<td>The approximated 95% confidence interval</td>
<td>(0.00459, 0.00473)</td>
</tr>
<tr>
<td>The crude bound</td>
<td>(0.00294, 0.00638)</td>
</tr>
</tbody>
</table>

If we want to compare the credible intervals in Table 6.4 to the uncertainty bounds in Table 6.5, we need to plug in the lower and upper limits of the credible intervals of each parameter to the loss function. However, we cannot do that because the loss function is defined from the mixture density which has the constraint on the weights that $\sum_{j=1}^k \omega_j = 1$. For example, the lower limits of the credible intervals for the parameter $\omega$ are $(0.013, 0.042, 0.779)$, we cannot treat these limits as $(\hat{\omega}_1^*, \hat{\omega}_2^*, \hat{\omega}_3^*)$ because $\sum_{j=1}^3 \hat{\omega}_j^* = 0.834 \neq 1$. Consequently, we cannot compare the uncertainty bounds in Table 6.5 to the credible intervals in Table 6.4 directly as they are computed from different approaches. We could compare the two uncertainty bounds by using the width of the interval obtained from

upper limits – lower limits.

We show the widths of the uncertainty bounds as well as the widths of the credible intervals in Table 6.6.
Table 6.6: The widths of the credible intervals and the two uncertainty bounds; the approximated 95% confidence interval and the crude bound evaluated at $\hat{\xi}^*$ based on MCMC with $N = 10,000$ for $k = 3$.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Components</th>
<th>The width of the credible intervals</th>
<th>The width of the approximated 95% confidence interval</th>
<th>The width of the crude bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\omega$</td>
<td>1</td>
<td>0.09</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.14</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mu$</td>
<td>1</td>
<td>1.54</td>
<td>0.00014</td>
<td>0.00344</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\sigma^2$</td>
<td>1</td>
<td>0.93</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.38</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2.91</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

It is desirable to have a narrow width (for more precise estimates). For example, the credible interval of $\sigma^2$ of the component 3 is wider than the others. Therefore, we could say that the point estimate $\hat{\sigma}^2$ of the component 3 is less reliable than the others. As we mentioned that we might not be able to construct the credible intervals if there is a label switching problem. We could use the uncertainty bounds present the reliability of point estimates as they are constructed based on the expected loss function where the label of the component does not matter. In this approach, a narrower width is also more desirable than a wider one. We found that the approximated 95% confidence interval gives a much smaller relative width than the crude bound. Moreover, we could use these two uncertainty bounds to compare the uncertainty of the point estimates $\hat{\xi}^{(1)}, \hat{\xi}^{(2)}, \ldots, \hat{\xi}^{(M)}$ from the simulated annealing method by constructing the uncertainty bounds of the expected loss function evaluated at these point estimates. The results are shown in Figures 6.14 - 6.15.
Figure 6.14: The uncertainty bounds and the crude uncertainty bounds of the expected loss function based on $N = 50,000$ MCMC samples, evaluated at the $j$th iteration of the SA algorithm for the mixture model with the component $k = 3$. 

The uncertainty bounds of the expected loss evaluated at $\xi^{(j)}$.

The crude uncertainty bounds of the expected loss evaluated at $\xi^{(j)}$.

Figure 6.15: The uncertainty bounds and the crude uncertainty bounds of the expected loss function based on $N = 50,000$ MCMC samples, evaluated at the $j$th iteration of the SA algorithm for the mixture model with the component $k = 4$. 
The results in Figures 6.14 - 6.15 suggest that the uncertainty bound approximated by the 95% confidence interval seems to provide a more reliable uncertainty than the crude uncertainty bound as it is narrower. One of the possible reasons is the extreme case \( \hat{f}_\xi(y)^L \) or/and \( \hat{f}_\xi(y)^U \) are unnecessarily merged. As a result, the crude uncertainty bound might not be practical since the extreme cases (6.29) and (6.30) do not occur together at all values at \( y \)'s simultaneously.

### 6.5.4 Conclusion

In conclusion, although using the uncertainty bound is much more expensive than the crude uncertainty bound, it is a more reliable way to provide the uncertainty. Nevertheless, these two uncertainty bounds have the overlapping of the lower and upper bounds which leads to unconvincing decisions about the best point estimate \( \hat{\xi}^* \). Therefore, we cannot be certain that the point estimate \( \hat{\xi}^* \) is the best compared to others \( \hat{\xi}^{(j)} \) for \( j = 1, \ldots, M = 1,000 \). One possible reason is that although the simulated annealing algorithm is an acceptably good technique for the global optimisation problem, it is always possible to become stuck at the local minima. The point estimate and the expected values cannot be improved thus we can not get over the non-overlapping bound.

Although, the decomposed form of the expected loss function, \( \overline{g}_\xi(\xi) \) leads to cheap computation for constructing the crude uncertainty bound, it fails to provide a practical uncertainty bound. Meanwhile, variance estimation of the expected loss function using the function \( \overline{h}_\xi(\xi) \) is quite computationally expensive. Nevertheless, these two uncertainty bounds could only present how uncertainty of the point estimate \( \xi \) affects the expected loss function. What can we do to assess uncertainty of point estimates of a mixture model? We move onto an alternative in the next chapter.
Chapter 7

The Jackknife Method for Uncertainty of Point Estimates

7.1 Introduction

We have shown that we could present the uncertainty of the point estimate of the parameter \( \xi = (\omega, \mu, \sigma^2) \) without dealing with difficulty in label switching in MCMC output by using the uncertainty bound approximated by the 95\% confidence interval of the expected loss function evaluated at the point estimate \( \hat{\xi} \) under the integrated squared difference loss function of the Normal mixture model. We have found that computing the uncertainty bound is computationally expensive (it takes long time). We could use the crude uncertainty bound which is computationally cheaper, but it might not be practical. Nevertheless, both types of the uncertainty bounds cannot provide the uncertainty of the point estimates for each component \( k \); for example, we cannot assess the uncertainty of \( \hat{\mu}_j \) for each \( j = 1, \ldots, k \). In this chapter, we present a more general alternative approach to assess uncertainty of point estimates. We aim to present the uncertainty for point estimates found using Bayesian decision theory not only for a mixture model.

We use the idea from the jackknife method to assess uncertainty of Bayesian point estimates. We consider to apply the idea of the jackknife method because it requires no theoretical calculations and is always available no matter how complicated the estimator \( \hat{\theta} \) is. What is the jackknife method? A jackknife can be referred to as a foldable pocket knife which is easy to carry around. In statistics, the jackknife method is often referred to as the nonparametric estimation of statistical error of a statistic of interest. It was introduced in Quenouille (1949) with the intention of reducing the bias of the sample
estimate. It was developed further in Tukey (1958) as a general approach for testing hypotheses and calculating confidence intervals using the assumption that the jackknife replicates are considered identically and independently distributed; see Miller (1974). The jackknife method is also known as a resampling method for variance and bias estimation. Why resample? Resampling methods give a way to estimate the bias and variability of an estimator \( \hat{\theta} \) by using the values of \( \hat{\theta} \) on subsamples (jackknife samples) of \( x_1, \ldots, x_n \). There are two well known resampling methods; the jackknife method and the bootstrap method, Efron (1979). The basic concept of resampling is to estimate the precision of sample statistics based on removing data and then recalculating from subsets of available data (the jackknife method) or drawing randomly with replacement from a set of data points (the bootstrap method), Efron and Tibshirani (1994). The jackknife method is the simple resampling method, but is possibly computationally expensive. The reason is that the time required to compute the jackknife standard error for an estimator \( \hat{\theta} \) will depend on the time required to compute \( \hat{\theta} \) itself, especially for a large number of replicates. The advent of high performance computers makes the jackknife method more attractive. Although the jackknife method is relatively old method (compared to the bootstrap method), it is recently used in many problems, especially in econometrics (see Phillips and Yu (2005), Chiquoine and Hjalmarsson (2009), and Chambers (2013)). The idea of using subsamples from the jackknife method could be used for uncertainty estimation. For example, in Zhang et al. (2010), they used the concept of leaving out a sample from the jackknife method to perturb a data set in a method called paired-samples test algorithm (PST) to classify and assess the uncertainty multiple tumor types using gene expression information.

In this thesis, we also want to use the leaving out concept from the jackknife method to propose a new approach to assess the uncertainty of simulation-based Bayes estimates. To do so, we will compute the Bayes estimates called “jackknife-Bayes estimates” which are based on the concept of leaving one out from the jackknife method. Then we use the jackknife-Bayes estimates to visualise the uncertainty of the Bayes estimates. In this chapter, we present the fundamental concept of the jackknife method in Section 7.2 and describe how to obtain the jackknife-Bayes estimates in Sections 7.3. In Section 7.4, we describe the uncertainty assessment using the jackknife-Bayes estimates for the Normal mixture model. We then show the simulation results of Bayes estimates and the jackknife-Bayes estimates in Section 7.5.
7.2 The jackknife method

Let $\hat{\theta}$ be an estimator of $\theta$ based on i.i.d. random variables $X_1, \ldots, X_n$. This means that $\hat{\theta} = f(X_1, \ldots, X_n)$ for some function $f$. Let

$$\hat{\theta}_{(i)} = f(X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n)$$

be the corresponding recomputed statistic based on all but the $i$-th observation. Having observed $X_1 = x_1, \ldots, X_n = x_n$, we define jackknife samples as follows.

**Definition 7.1.** The jackknife samples are computed by leaving out one observation $x_i$ from $x = (x_1, x_2, \ldots, x_n)$ at a time:

$$x_{(i)} = (x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n).$$

**Definition 7.2.** The $i$th jackknife replication $\hat{\theta}_{(i)}$ of the statistic $\hat{\theta} = f(x)$ is

$$\hat{\theta}_{(i)} = f(x_{(i)}).$$

**Example 7.1.** Jackknife replication of the mean $\hat{\theta} = \bar{x}$

$$f(x_{(i)}) = \frac{1}{n-1} \sum_{j \neq i} x_j = \frac{(n\bar{x} - x_i)}{n-1} = \bar{x}_{(i)}.$$

The jackknife estimator of bias $\mathbb{E}(\hat{\theta}) - \theta$ is given by

$$\text{bias}^J = \frac{n-1}{n} \sum_{i=1}^{n} (\hat{\theta}_{(i)} - \hat{\theta}).$$

The jackknife estimator $\hat{\theta}^J$ of $\theta$ is given by

$$\hat{\theta}^J = \hat{\theta} - \text{bias}^J$$

$$= \hat{\theta} - \frac{n-1}{n} \sum_{i=1}^{n} (\hat{\theta}_{(i)} - \hat{\theta})$$

$$= \frac{1}{n} \sum_{i=1}^{n} (n\hat{\theta} - (n-1)\hat{\theta}_{(i)}).$$

(7.1)
Define the summands in Equation (7.1)

$$\tilde{\theta}_i = n\hat{\theta} - (n - 1)\hat{\theta}_{(i)} = \hat{\theta} + (n - 1)(\hat{\theta} - \hat{\theta}_{(i)}).$$

The values \(\tilde{\theta}_i\) are called pseudo-values. In Quenouille (1956), it has been suggested that, the pseudo-values can be viewed as a bias-corrected version of \(\hat{\theta}\). They are treated as if they were independent random variables with mean \(\theta\),

$$\hat{\theta}^J = \frac{1}{n} \sum_{i=1}^{n} \tilde{\theta}_i.$$ 

Therefore, the jackknife estimator of variance \(\hat{\theta}\) is given by

$$\overline{\text{Var}}^J(\hat{\theta}) = \overline{\text{Var}}(\hat{\theta}^J) = \frac{1}{n} \overline{\text{Var}}(\tilde{\theta}_i)$$

$$= \frac{1}{n(n-1)} \sum_{i=1}^{n} (\tilde{\theta}_i - \hat{\theta}^J)^2$$

$$= \frac{1}{n(n-1)} \sum_{i=1}^{n} (n\hat{\theta} - (n - 1)\hat{\theta}_{(i)} - n\hat{\theta} - \frac{n-1}{n} \sum_{i=1}^{n} \hat{\theta}_{(i)})^2$$

$$= \frac{1}{n(n-1)} \sum_{i=1}^{n} (-(n-1)(\hat{\theta}_{(i)} - \frac{n}{n} \sum_{i=1}^{n} \hat{\theta}_{(i)}))^2$$

$$= \frac{n-1}{n} \sum_{i=1}^{n} (\hat{\theta}_{(i)} - \hat{\theta}_{(·)})^2$$

where \(\hat{\theta}_{(·)} = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_{(i)}\).

**Example 7.2.** The pseudo-values of \(\hat{\theta} = \bar{x}\) are

$$n\bar{x} - (n - 1)\bar{x}_{(i)} = x_i \quad i = 1 \ldots , n.$$ 

Therefore, the jackknife estimation of the variance of the mean \(\hat{\theta} = \bar{x}\).

$$\overline{\text{Var}}^J(\bar{x}) = \frac{1}{n(n-1)} \sum_{i=1}^{n} (x_i - \bar{x})^2$$

$$= \frac{n-1}{n} \sum_{i=1}^{n} (\bar{x}_{(i)} - \bar{x}_{(·)})^2$$

which is the usual variance for the sample mean.
For most statistics, the jackknife estimator of variance tends asymptotically to the true value almost surely. The jackknife estimate is technically consistent:

$$\frac{\text{Var}_J(\hat{\theta})}{\text{Var}(\hat{\theta})} \to 1,$$

as $n \to \infty$ almost surely in particular for a smooth functional model, Shao and Wu (1989).

### 7.3 Jackknife-Bayes estimates

The key idea of the jackknife method is to leave out one observed datum $x_i$ and use the jackknife samples $x_{(i)}$ to compute the estimate of the parameter $\theta$. We can use the formula (7.2) to estimate the variance. In this thesis, we aim for assessing uncertainty of Bayesian point estimates. We only adopt the idea of leaving out observations (jack-knifing) instead of using the jackknife estimate of the variance to assess uncertainty of those point estimates. What we will do is computing Bayes estimates with respect to the reduced posterior distribution related to the $x_i$ deleted points for $i = 1, \ldots, n$ and using these Bayes estimates to visualise uncertainty of the Bayes estimate with respect to the full posterior distribution. The Bayes estimates with respect to the reduced posterior distribution are obtained from using a sample $\theta$ drawn from the posterior distribution which is derived from the likelihood leaving out observed data $x_i$. The Bayes estimate based on the jackknife samples $x_{(i)}$ is then called the jackknife-Bayes estimate denoted by $\hat{\theta}_J^{(i)}$. Therefore, the jackknife-Bayes estimate is obtained from

$$\hat{\theta}_J^{(i)} = \arg \min_{\hat{\theta}} \mathbb{E}_{\pi(\theta|x_{(i)})}[L(\hat{\theta}, \theta)],$$

where $\pi(\theta|x_{(i)})$ is the posterior distribution derived from the prior $p(\theta)$ and the likelihood function $l(x_{(i)}|\theta)$ and given by

$$\pi(\theta|x_{(i)}) = \frac{p(\theta)l(x_{(i)}|\theta)}{\int p(\theta)l(x_{(i)}|\theta) \, d\theta} = \frac{p(\theta)l(x_{(i)}|\theta)}{m_i}. \quad (7.3)$$

For convenience, we denote the posterior distribution (7.3) by $\pi^{(i)}$ and name it the $x_i$-delete posterior distribution. We can compute $n$ Bayes estimates $\hat{\theta}_J^{(1)}$ with respect to $x_i$-delete posterior distributions for $i = 1, \ldots, n$ in a similar approach to the Bayes estimate $\hat{\theta}$ with respect to the full posterior distribution $\pi(\theta|x)$. By the Ergodic Theorem 2.6 with limiting distribution $\pi^{(i)}$, we can estimate $\mathbb{E}_{\pi(\theta|x_{(i)})}[L(\hat{\theta}, \theta)]$ by ergodic averaging.
of MCMC samples $\hat{\theta}^j \sim \pi(\cdot)$,

$$E_{\pi(\theta|x_{(i)})}[L(\hat{\theta}, \theta)] \approx \frac{1}{N} \sum_{j=1}^{N} L(\hat{\theta}, \theta^j).$$

We then apply a minimisation method for searching for $\hat{\theta}^j_{(i)}$.

### 7.3.1 Importance sampling for computing jackknife-Bayes estimates

Computing the jackknife-Bayes estimates seems to be very expensive as we have to use $n$ different sets of the sample $\theta$ corresponding to of the $x_i$-delete posterior distributions for $i = 1, \ldots, n$. We can reduce the computational cost of sampling by using importance sampling. The fundamental concept of importance sampling is evaluating a Monte Carlo integral (2.8) by using a sample from the instrumental distribution instead of using a sample from the target distribution (see more details in Subsection 2.3.2).

What should we use for the instrumental distribution? At this point, we have already used (MCMC) samples generated from the (full) posterior distribution, $\pi = \pi(\theta|x)$ for computing the Bayes estimate. We could reuse these samples by using the full posterior distribution $\pi$ as the instrumental distribution in importance sampling. Therefore, we express the expected loss function with respect to the $x_i$-delete posterior distribution as the expected loss function with respect to the (full) posterior distribution, $\pi$ as follows:

$$E_{\pi(\cdot)}[L(\hat{\theta}, \theta)] = \int L(\hat{\theta}, \theta)\pi(\theta|x_{(i)})d\theta$$

$$= \int L(\hat{\theta}, \theta)\frac{\pi(\theta|x_{(i)})}{\pi(\theta|x)} \pi(\theta|x)d\theta$$

$$= \int L(\hat{\theta}, \theta)\frac{p(\theta)l(\theta|x_{(i)})}{p(\theta)l(\theta|x)} \pi(\theta|x)d\theta$$

$$= \frac{m}{m_i} \int L(\hat{\theta}, \theta)w_i \pi(\theta|x)d\theta$$

$$= \frac{m}{m_i} E_{\pi}[L(\hat{\theta}, \theta)w_i], \quad \text{(7.4)}$$

where

$$w_i = \frac{l(x_{(i)}|\theta)}{l(x|\theta)}$$

$$= \frac{1}{l(x_i|\theta)} \quad \text{if } x_i, i = 1, \ldots, n \text{ are independent}, \quad \text{(7.5)}$$
and

\[ m_i = \int p(\theta) l(x_i | \theta) d\theta = \int \frac{p(\theta) l(x_i | \theta)}{\pi(\theta | x)} \pi(\theta | x) d\theta = \int \frac{l(x_i | \theta)}{l(x | \theta) / m} \pi(\theta | x) d\theta. \]

Therefore,

\[ \frac{m}{m_i} = \frac{1}{\int \frac{l(x_i | \theta)}{l(x | \theta)} \pi(\theta | x) d\theta} = \frac{1}{\int w_i \pi(\theta | x) d\theta} = \frac{1}{E_\pi [w_i]} . \tag{7.6} \]

From Equations (7.4) and (7.6), we therefore estimate \( E_{\pi(\cdot)} [L(\hat{\theta}, \theta)] \) by using \( \frac{E_\pi [L(\hat{\theta}, \theta) w_i]}{E_\pi [w_i]} \).

We could use the same MCMC samples \( \theta^j \) for \( j = 1, \ldots, N \) which are in estimation \( E_\pi [L(\hat{\theta}, \theta)] \) to obtain \( E_\pi [L(\hat{\theta}, \theta) w_i] \) and \( E_\pi [w_i] \) by using

\[ \mathbb{E}_\pi [L(\hat{\theta}, \theta) w_i] \approx \frac{1}{N} \sum_{j=1}^{N} L(\hat{\theta}, \theta^j) w_i^j, \]

and

\[ \mathbb{E}_\pi [w_i] \approx \frac{1}{N} \sum_{j=1}^{N} w_i^j, \]

where

\[ w_i^j = \frac{l(x_i | \theta^j)}{l(x | \theta^j)}, \quad \text{for } i = 1, \ldots, n, \ j = 1, \ldots, N. \]

Therefore, the expected loss function with respect to the \( x_1 \)-delete posterior distribution in Equation (7.4) can be computed by reusing the generated MCMC samples from the full posterior distribution \( \pi \) using the weight in Equation (7.5).
By using importance sampling, we could estimate of the expected loss function relatively cheaply as we do not need to generate MCMC samples corresponding to the \( n \times_i \)-delete posterior distributions. Nonetheless, computing the jackknife-Bayes estimates could still be expensive because of choice of minimisation methods. We could replace the simulated annealing method which is often used for searching the Bayes estimate with respect to the full posterior distribution with cheaper deterministic methods to reduce the computational cost.

### 7.4 Assessing uncertainty using jackknife-Bayes estimates

We make use of the jackknife-Bayes estimates \( \hat{\theta}_{J(i)} \) with respect to the \( x_i \)-delete posterior distribution posterior distributions for \( i = 1, \ldots, n \) to visualise uncertainty of the Bayes estimate \( \hat{\theta} \) with respect to the full posterior distribution. To illustrate this, we consider the Bayes estimate \( \hat{\xi}^* = (\hat{\omega}^*, \hat{\mu}^*, \hat{\sigma}^2) \) under the squared difference loss function in Equation (6.18) from Chapter 6.

Suppose we have \( n \) observations, so there are \( n \) jackknife-Bayes estimates with respect to \( x_i \)-delete posterior distributions for \( i = 1, \ldots, n \). The jackknife-Bayes estimates are solutions of

\[
\hat{\xi}_{(i)} = \arg\min_{\xi} \mathbb{E}_{\pi(i)}[L(\hat{\xi}, \xi)],
\]

where

\[
\mathbb{E}_{\pi(i)}[L(\hat{\xi}, \xi)] = \mathbb{E}_{\pi(i)} \left[ \int_{\mathcal{R}} (f_{\hat{\xi}}(y) - f_{\xi}(y))^2 dy \right].
\]

By using importance sampling, we can estimate the expected loss function with respect to the deleted posterior distribution by

\[
\mathbb{E}_{\pi(i)}[L(\hat{\xi}, \xi)] = \frac{\mathbb{E}_{\pi}[L(\hat{\xi}, \xi)w_i]}{\mathbb{E}_{\pi}[w_i]} = \frac{\mathbb{E}_{\pi} \left[ \int_{\mathcal{R}} (f_{\hat{\xi}}(y) - f_{\xi}(y))^2 w_i dy \right]}{\mathbb{E}_{\pi}[w_i]}.
\]
Assume that the order of integration can be interchanged, so

\[
\mathbb{E}_{\pi(i)}[L(\hat{\xi}, \xi)] = \frac{\int R \mathbb{E}_{\pi} \left[ (f(\hat{\xi}(y)) - f(\xi(y))^2 w_i \right] dy}{\mathbb{E}_{\pi}[w_i]}
\]

\[
= \int R f(\hat{\xi}(y))^2 dy \mathbb{E}_{\pi}[w_i] - 2 \int R f(\hat{\xi}(y))\mathbb{E}_{\pi}[f(\xi(y)) w_i] dy + \int R \mathbb{E}_{\pi}[f(\xi(y))^2 w_i] dy
\]

\[
= \frac{\int R \mathbb{E}_{\pi} \left[ (f(\hat{\xi}(y)) - f(\xi(y))^2 w_i \right] dy}{\mathbb{E}_{\pi}[w_i]},
\]

where

\[
w_i = \frac{l(x_{1(i)}|\xi)}{l(x|\xi)}
\]

\[
= \sqrt{2\pi\sigma^2_z} \exp \left\{ \frac{1}{2\sigma^2_z} (x_i - u_i^j)^2 \right\} \text{ from Equation (6.9)},
\]

and \(z_i\) is the latent variable used for allocating the components in the mixture model. Therefore, by using the MCMC samples \(\xi_j^i\) for \(j = 1, \ldots, N\) generated from the full posterior distribution \(\pi\), we can estimate the expected loss function with respect to the \(x_i\)-delete posterior distributions for \(i = 1, \ldots, n\) by

\[
\mathbb{E}_{\pi}[w_i] \approx \frac{1}{N} \sum_{j=1}^{N} \sqrt{2\pi\sigma^2_z} \exp \left\{ \frac{1}{2\sigma^2_z} (x_i - u_i^j)^2 \right\},
\]

\[
\mathbb{E}_{\pi}[f(\xi(y)) w_i] \approx \frac{1}{N} \sum_{j=1}^{N} f(\xi^i(y)) \sqrt{2\pi\sigma^2_z} \exp \left\{ \frac{1}{2\sigma^2_z} (x_i - u_i^j)^2 \right\},
\]

\[
\mathbb{E}_{\pi}[f(\xi(y))^2 w_i] \approx \frac{1}{N} \sum_{j=1}^{N} f(\xi^i(y))^2 \sqrt{2\pi\sigma^2_z} \exp \left\{ \frac{1}{2\sigma^2_z} (x_i - u_i^j)^2 \right\}.
\]

Note that we approximate integrals in Equation (7.7) by using the composite Simpson’s 1/3 rule as in Equation (6.24) in Subsection 6.4.1. We implement the simulated annealing algorithm to search for the jackknife-Bayes estimates \(\hat{\xi}^J(i)\) by using Equation (7.8) as the cost functions for \(i = 1, \ldots, n\) the same way as we do for the Bayes estimate \(\hat{\xi}^\ast\). We then assess the uncertainty of the Bayes estimate \(\hat{\xi}^\ast\) by investigating the values of the jackknife-Bayes estimate \(\hat{\xi}^J(i)\). Note that there are two sources of uncertainty involving the computed Bayes estimate \(\hat{\xi}^\ast\); the uncertainty from using the MCMC samples for estimating the expected loss function and from using the simulated annealing which is the simulation-based minimisation method. Therefore, the jackknife-Bayes estimates have similar sources of uncertainty to the Bayes estimate as they are computed from the same procedure.
We note that the jackknife-Bayes estimates, $\hat{\xi}_J^{(i)}$, and the Bayes estimate $\hat{\xi}^*$ have similar sources of uncertainty; from MCMC samples in estimation and from the simulated annealing method in minimisation. In order to check out the reliability of the uncertainty from jackknife-Bayes estimates, instead of using the galaxy data, we use the simulated data. Thus we could be more certain that the simulated annealing method does provide the Bayes estimates and the jackknife-Bayes estimates correctly. In the next section, we present the simulation results of the uncertainty using the jackknife-Bayes estimates using the simulated data.

7.5 The uncertainty of the Bayes estimates of the Normal mixture model: the simulation study

In this section, we want to investigate the uncertainty of the Bayes estimates $\hat{\xi}^*$ obtained from Equation (6.19) using the idea from the jackknife method. We use the same simulation procedure described in Chapter 6 but use the simulated data instead of the galaxy data. We consider the simulated data generated from

\[
x_{110}, \ldots, x_{10} \stackrel{i.i.d}{\sim} \mathcal{N}(10, 1)
\]

\[
x_{11}, \ldots, x_{70} \stackrel{i.i.d}{\sim} \mathcal{N}(22, 2)
\]

\[
x_{71}, \ldots, x_{82} \stackrel{i.i.d}{\sim} \mathcal{N}(33, 2),
\]

where $x_1 < \ldots < x_{82}$. By using the simulated data in (7.9), we could regard

\[
\hat{\omega} = \left(\frac{10}{82}, \frac{60}{82}, \frac{12}{82}\right) = (0.122, 0.732, 0.146)
\]

\[
\hat{\mu} = (10, 22, 33)
\]

\[
\hat{\sigma}^2 = (1, 2, 2)
\]

as the point estimates of the Normal mixture model in Equation (6.2). We indicated the issue of prior sensitivity in our simulation setup which could affect point estimation, so before we consider the uncertainty of the Bayes estimates using the jackknife-Bayes estimates, we investigate the variability of the simulation results by using different datasets simulated from (7.9) and compare the Bayes estimates to the expected values in (7.10). After that we discuss the uncertainty using the jackknife-Bayes estimates. We use the 500 simulated datasets of $x_i$ for $i = 1, \ldots, 82$ and compute Bayes estimates based on MCMC simulations which each simulation has $N = 50,000$. The histogram of the 500 simulated datasets is shown in Figure 7.1 and the simulation results are shown in Table 7.1.
Table 7.1: The first quantiles (Q1), medians, means and the third quartiles (Q3) of the Bayes estimates obtained from 500 simulations using 500 simulated datasets.

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Expected values</th>
<th>Q1</th>
<th>Medians</th>
<th>Means</th>
<th>Q3</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\omega$</td>
<td>0.122</td>
<td>0.128</td>
<td>0.129</td>
<td>0.126</td>
<td>0.129</td>
</tr>
<tr>
<td></td>
<td>0.732</td>
<td>0.717</td>
<td>0.718</td>
<td>0.708</td>
<td>0.719</td>
</tr>
<tr>
<td></td>
<td>0.146</td>
<td>0.153</td>
<td>0.153</td>
<td>0.167</td>
<td>0.154</td>
</tr>
<tr>
<td>$\mu$</td>
<td>10</td>
<td>9.779</td>
<td>10.02</td>
<td>10.52</td>
<td>10.25</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>32.55</td>
<td>32.89</td>
<td>32.27</td>
<td>33.21</td>
</tr>
<tr>
<td>$\sigma^2$</td>
<td>1</td>
<td>0.533</td>
<td>0.707</td>
<td>0.752</td>
<td>0.917</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.008</td>
<td>1.272</td>
<td>1.376</td>
<td>1.558</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1.697</td>
<td>1.933</td>
<td>6.314</td>
<td>2.209</td>
</tr>
</tbody>
</table>
The results in Table 7.1 could show the variability in Bayes estimates of parameters \(\omega\), \(\mu\) and \(\sigma^2\). We can see that the mean value of parameter \(\sigma^2\) in the third component, 6.314 is quite different from the expected value, 2. This is because of very high values of the Bayes estimates obtained from simulations in which convergence had not been achieved. However, mostly, our simulation method works reasonably well as we cannot see much of the variability in the Bayes estimates.

Now we turn our attention to the 82 jackknife-Bayes estimates corresponding the \(x_i\)-delete posterior distributions. It is more practical to use the jackknife-Bayes estimates from the simulations which give sensible Bayes estimates to study the uncertainty. To present the uncertainty using the jackknife-Bayes estimates, instead of using the results from 500 simulations, we depict the jackknife-Bayes estimates from stable chains to assess the uncertainty of the Bayes estimates. We plot the Bayes estimates and 82 jackknife-Bayes estimates obtained from 30 simulations based on 30 stable chains in Figures 7.2 - 7.7 to show the uncertainty.
The Bayes estimates and the jackknife-Bayes estimates of the parameter $\mu$

Figure 7.2: The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x_{(i)}$ for $i = 1, \ldots, 82$ of the parameter $\mu$ where the vertical red lines are the expected values of the parameter: three components (above) and component 1 (bottom).
Figure 7.3: The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x_{(i)}$ for $i = 1, \ldots, 82$ of the parameter $\mu$ where the vertical red lines are the expected values of the parameter: component 2 (above) and component 3 (bottom).
Figure 7.4: The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x_{(i)}$ for $i = 1, \ldots, 82$ of the parameter $\sigma^2$ where the vertical red lines are the expected values of the parameter: three components (above) and component 1 (bottom).
Figure 7.5: The Bayes estimates (solid green dots and solid blue dots represent component 2 and component 3, respectively) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x_{(i)}$ for $i = 1, \ldots, 82$ of the parameter $\sigma^2$ where the vertical red lines are the expected values of the parameter.
The Bayes estimates and the jackknife–Bayes estimates of the parameter $\omega$

The Bayes estimates and the jackknife–Bayes estimates of the parameter $\omega_1$

Figure 7.6: The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x_{(i)}$ for $i = 1, \ldots, 82$ of the parameter $\omega$ where the vertical red lines are the expected values of the parameter: three components (above) and component 1 (bottom).
The Bayes estimates and the jackknife–Bayes estimates of the parameter $\omega_2$

The Bayes estimates and the jackknife–Bayes estimates of the parameter $\omega_3$

Figure 7.7: The Bayes estimates (solid green dots) and the jackknife-Bayes estimates (black circles) corresponding to deleted simulated values $x_{(i)}$ for $i = 1, \ldots, 82$ of the parameter $\omega$ where the vertical red lines are the expected values of the parameter: component 2 (above) and component 3 (bottom).
Discussion of the simulation results

The jackknife-Bayes estimates could present how the Bayes estimates could be when one of data $x_i$ for $i = 1, \ldots, 82$ is deleted. Therefore, when we plot the Bayes estimates and the jackknife-Bayes estimates, we expect to see thick clouds of the jackknife-Bayes estimates in which the Bayes estimates are in the middle of those clouds and they are centred around the expected values. We could use Figures 7.2 - 7.7 to show the performance of the method and to visualise the uncertainty of the Bayes estimates.

In Figures 7.2 - 7.3, we can see that the Bayes estimates (solid green dots) of the parameter $\mu$ are centred around the expected values (vertical red lines). This suggests that there are small differences between the Bayes estimates and the expected values. In these figures, we also see that the jackknife-Bayes estimates (black circles) are centred around the Bayes estimates. We notice that the jackknife-Bayes estimates of $\mu_2$ are precisely very close to each other, so they are clustered together in thicker clouds than the jackknife-Bayes estimates of $\mu_1$ and $\mu_3$. One possible reason for this result is that there are 60 observations (simulated data), $x_{11}, \ldots, x_{70}$ generated from the second component of the mixture model, so leaving one observation does not have much effect on the estimates of $\mu_2$.

In Figure 7.4 - 7.5, the Bayes estimates of the parameter $\sigma^2$ are also centred around the expected values. These figures shown that the Bayes estimates of the parameter $\sigma^2$ are equally good in every simulation because the solid green dots representing the Bayes estimates are not much different from the expected values. However, we found that there are some simulations that provide some of the 82 jackknife-Bayes estimates of $\sigma^2$ which are very different from the Bayes estimates (see the above plot of Figure 7.4). Why do we have some of the 82 jackknife-Bayes estimates that are very different from the Bayes estimates and the other jackknife-Bayes estimates? Computing jackknife-Bayes estimates involves two computational stages; estimation and minimisation. In each simulation, we use the same MCMC sample in the estimation stage to compute both the Bayes estimates and the 82 jackknife-Bayes estimates. Therefore, the occurrence of the outliers of the jackknife-Bayes estimates is most likely caused by simulated annealing which is the simulation optimisation method used in the minimisation stage.

In Figures 7.6 - 7.7, the Bayes estimates of the parameter $\omega$ are close to the expected values. Nonetheless, we found that they are not quite centred around the expected values. The Bayes estimates of the parameters $\omega_1$ and $\omega_3$ tend to be overestimated whilst the Bayes estimates of the parameter $\omega_2$ tend to be underestimated. This suggests that there is a systematic trend rather than a random variation centred around the expected values. As a result, although we can see thick clouds of the jackknife-Bayes estimates of $\omega$ in which the Bayes estimates are mostly in the middle of those
clouds, the Bayes estimates and the jackknife-Bayes estimates might be biased. We could express the differences between the Bayes estimates and the expected values in terms percentage errors (on average) from

\[
\frac{\text{The average difference}}{\text{The expected value}} \times 100.
\]

It turns out that the percentage errors of \(\omega_1, \omega_2\) and \(\omega_3\) are 5.24, -1.88, and 5.05, respectively. Why does the method tend to give the overestimates of \(\omega_1\) and \(\omega_3\) and the underestimates of \(\omega_2\)? Computing the estimates of the parameter \(\omega\) involves estimation and minimisation similar to the other parameters. In the estimation stage, the MCMC sample of the parameter \(\omega\) is generated from the Dirichlet distribution which seems to be suitable as the parameter \(\omega\) has the constraint, \(\sum_{j=1}^{3} \omega_j = 1\). In the minimisation stage, due to this constraint, we have to generate \(\hat{\omega}'\) from a distribution of the neighbourhood of \(\hat{\omega}^{(t)}\) in a rather unusual approach in the simulated annealing algorithm (see Algorithm 6.6 and Algorithm B.1). This procedure in our method might affect the estimates of the parameter \(\omega\).

In summary, our method seems to work reasonably well to provide the Bayes estimates and the jackknife-Bayes estimates for the parameter \(\mu\) and \(\sigma^2\) as we could see the thick clouds of the jackknife-Bayes estimates in which the Bayes estimates are mostly in the middle of those clouds and they are centred around the expected values. However, it is doubtful whether the method works well for the estimates of the parameter \(\omega\) because even though there is a small discrepancy between the estimated values and the expected values, it suggests a systematic trend of the estimates away from the expected values. Overall the method shows promise, but there are areas in which further investigation are required. These include investigating the very high values of the jackknife-Bayes estimates occurred in some simulations and the Bayes estimates of the parameter \(\omega\). It is interesting to further investigate these two issues by taking a closer look at the simulated annealing algorithm.
7.6 Conclusion

We have shown that we could use the clouds of the jackknife-Bayes estimates to visualise the uncertainty of the Bayes estimates. The jackknife-Bayes estimates have the same source of uncertainty as the Bayes estimates because they are computed from the same procedures. An advantage of using the jackknife-Bayes estimates in the mixture model is that label switching is not a problem. Therefore, using the jackknife-Bayes estimates allows us to assess the uncertainty of Bayes estimates for each component without dealing with identifying the number of the components. The idea from the jackknife method to assess uncertainty could also be used in other models where Bayes estimates are based on a decision-theoretic viewpoint.

Computing jackknife-Bayes estimates could be complicated and computationally expensive because we have to deal with $n$ different $x_i$ deleted-posterior distributions corresponding to observations $x_1, \ldots, x_n$. In this thesis, we have presented that it could be relatively easy and computationally cheap by using importance sampling. Another issue of computing jackknife-Bayes estimates is the minimisation method. While the simulated annealing method tends to provide the global minimum, it is computationally expensive. We can avoid an expensive cost from using the simulated annealing method by using deterministic minimisation methods to compute the jackknife-Bayes estimates. The jackknife-Bayes estimates are usually not too far away from the Bayes estimate. As a result, we can avoid the problem of trapping in local minima that often occurs in some deterministic minimisation methods by using the value of the Bayes estimate (from the simulated annealing method) as the initial guess in those deterministic algorithms to search for the jackknife-Bayes estimates.

In this thesis, we have considered using the jackknife-Bayes estimates in the mixture model for $k = 3$. However, for higher number of components $k$, it might be difficult to assess the uncertainty using this approach. Jackknife-Bayes estimates could overlap among $k$ components. The jackknife-Bayes estimates could group as a big cloud instead of $k$ clouds. Therefore, we might not be able to group those jackknife-Bayes estimates to assess the uncertainty of Bayes estimates in each component.
Chapter 8

Conclusions

In this thesis, we have considered several aspects of the study of Bayes estimates under some nonstandard loss functions using simulation methods. The study has been divided roughly into three purposes. First, we have used the estimation-then-minimisation (ETM) method to compute Bayes estimate under some nonstandard loss functions. Second, we have explored point (Bayes) estimates and the uncertainty for a mixture posteriors based on MCMC samples. Third, we have proposed the idea from the jackknife method to assess the uncertainty of Bayes estimates.

One main aspect which has been engaged throughout the thesis is the ETM method. We have considered this method as an efficient computational approach to compute Bayes estimates. Therefore, we have explored the use of this method to other two nonstandard loss functions; the linex loss function and the quotient loss function. Typically, these two loss function are not expressed in the decomposable form. As a result, the ETM method cannot be implemented straightforwardly. We have used two methods for function approximation; the Taylor series approximation and the cubic spline approximation to transform the loss functions into the form where the ETM method could be implemented. We have compared efficiency of the ETM method using these two approximations to the naive method using the true loss function. We have demonstrated that the ETM method via these two approximation is faster than the naive method. Furthermore, we have found that the ETM method via the Taylor series approximation is faster than the cubic spline approximation. The reason is that although we can separate the estimation part from the minimisation part, they are not entirely separated because of adding an extra knot. The extra knot which we added is actually a candidate value of the Bayes estimate. By this way, we can be certain that we have the zero loss value evaluated at the candidate Bayes estimate. Therefore, when we implement the ETM method, we still need to do estimation and minimisation
simultaneously but the computation is still much less than using the naive method. The ETM method via the Taylor series approximation is faster, however it failed to provide a good Bayes estimate under the quotient loss function. In fact, the Taylor series approximation is not suitable for any bounded loss function. In contrast, the cubic spline approximation seems to work well for both loss functions. It seems to be more flexible and versatile because the following reasons. First, it only uses derivatives of order 2 while the Taylor series approximation usually requires higher than order 2. Second, while Taylor series approximation cannot be used to approximate a bounded loss function, on the other hand, cubic spline functions are derived from a set of data points which are exactly determined from a loss function, so loosely speaking, it can approximate any loss function. A key issue that makes the cubic spline approximation works efficiently and effectively in terms of the accuracy and the computational cost is a number of knots for interpolation. We have used the quantile values from generated samples as the knots to perform interpolation to approximate a loss function. By using these knots, we definitely have cubic spline functions which are defined for all possible values of the parameter sample. However, using just only 5 knots from the quantile values might be not good enough to approximate loss functions. Using more knots could improve the approximation. Our strategy to find an appropriate set of knots is to add points between the quantile knots according to a setup error tolerance. The more knots we use, the better approximation we have. However, using too many knots makes the ETM method no longer the efficient method as it is more expensive than naive method. In this thesis, we have not presented a decent strategy to find a set of appropriate knots which guarantees to make the ETM method cheaper than the naive method. The method we have used to find the knots is a trial and error method.

We have explored the use of ETM method for point estimation in two dimensions. We have come up with using bicubic interpolation. Interpolation in two dimensions is performed using grid points. We have used the quantile values from generated samples of two parameters as the initial grid points. We then add more points between them similar to one dimension but the procedure of adding them is more complicated. Similar to cubic spline interpolation in one dimension, we have also considered an extra grid point to guarantee that we have zero loss at a candidate Bayes estimate. Nonetheless, the ETM method with the extra grid points is much slower than the naive method. As a result, we have not used the extra grid points to perform bicubic interpolation. We have considered the linex loss function and the quotient loss function which are defined for two dimensions. We have demonstrated that for these two loss function, the ETM method via bicubic interpolation works fairly well. We have obtained reasonably good Bayes estimates compared to ones from the naive method. However, the Bayes estimates are not good enough to convince that the ETM method via bicubic interpolation works efficiently and effectively.
The second aspect of the thesis concerns the analysis of a mixture posterior distribution with a known number of components which has been widely studied using MCMC algorithms. One difficulty on making inferences with mixture posterior distributions is label switching in the MCMC output. One of the methods to deal with label switching for point estimation is to use statistical decision theory with a label invariant loss function. By this approach, the point estimate which minimises the expected loss function with respect to the posterior distribution could be determined. We have used ergodic averaging to estimate the posterior expected loss and have applied simulated annealing as the minimisation method to compute point estimates. We have discussed in detail how to implement algorithms to compute point estimates under the integrated squared difference loss function for the Bayesian mixture using the galaxy dataset. We have used the decomposed form of the expected (integrated squared difference) loss function to implement the ETM method for computing the point estimates.

Moreover, we have investigated uncertainty of the point estimates for the mixture posterior distribution using the galaxy dataset. Due to label switching, we have considered the uncertainty of the expected loss evaluated at Bayes estimates which is label invariant. We have constructed two uncertainty bounds based on MCMC samples. The uncertainty bound which is approximated by a 95% confidence interval of the expected loss evaluated at Bayes estimates is very computationally expensive, while the crude uncertainty bound which is derived from the decomposed form of the expected loss function is computationally-cheap approximation. We have found that the crude uncertainty bound is likely to be less reliable than the approximated 95% confidence interval. Nevertheless, these two types of the uncertainty bounds cannot show the uncertainty of the point estimates in each component.

The third aspect of the thesis concerns a more general alternative method to present the uncertainty. We have proposed the method of jackknifing to show the uncertainty of point estimates or Bayes estimates found using Bayesian decision theory. We have computed jackknife-Bayes estimates corresponding to \( n \) observations \( x_1, \ldots, x_n \) which are the minimisers of the reduced posterior expected loss function corresponding to the \( x_i \)-delete observation. Then they could be used to visualise the uncertainty of Bayes estimates. By using jackknife-Bayes estimates, we could show the uncertainty of point estimates for the mixture model without coping with label switching. One of the most important issues of using the jackknife-Bayes estimates is the computational cost. Besides the relaxation of labelling, we can compute the jackknife-Bayes estimates by using importance sampling where the full posterior distribution is the instrumental distribution and hence the computational cost is relatively cheap. We could also apply a deterministic minimisation method to speed up the jackknife approach. The simulation study have shown that our method works reasonable well, however the estimates of the parameter \( \omega \) tend to be defective as there is a systematic trend of the estimates away
from the expected values. Further investigation, in particular the simulated annealing method applied in this problem is required.

We might not need to implement the ETM method under uncomplicated loss functions as the naive method could provide more accurate Bayes estimates. However, if we apply the simulated annealing method to search for Bayes estimates under uncomplicated loss functions, the computational cost still be relatively expensive notwithstanding. Although we could use deterministic optimisation methods to reduce the cost, we might not obtain a good Bayes estimate as we could get from simulated annealing because deterministic optimisation methods often get stuck at a local minima unless we have good initial points. In future work, we could try to use the ETM method implemented together with simulated annealing to provide a Bayes estimate and then use such value as the initial point for a deterministic method with the naive method. It would be interesting to investigate how quickly the algorithm run and how accurate the Bayes estimate is. Moreover, it would be interesting to apply the ETM method to compute Bayes estimates under some non-standard loss functions together with the study of the uncertainty using the jackknife-Bayes estimates. In this thesis, we have used the jackknife-Bayes estimates based on a deleted-one observation. In future, we could also study the use of the jackknife-Bayes estimates based on a deleted-\( m \) observation where \( 1 < m < n \) and \( n \) is the number of observations. Besides, we might apply the bootstrap method, Efron (1979) which is more recent than the jackknife method, Quenouille (1949) to assess the uncertainty of Bayes estimates. Finally, it is interesting to investigate the use of interpolation methods to the problem which we could not specify a suitable loss function as they are commonly used for approximation of an unknown function.
Appendix A

The moment generating function of $\theta$ when $\theta|x$ has Normal distribution with mean $\mu$ and variance $\sigma^2$:

$$M_\theta(t) = \mathbb{E}_x[e^{t\theta}] = \int_{-\infty}^{\infty} e^{t\theta} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(\theta-\mu)^2} d\theta.$$ 

Define $z = \frac{\theta - \mu}{\sigma}$, so $\theta = z\sigma + \mu$ and $d\theta = \sigma dz$. Then we get

$$M_\theta(t) = \int_{-\infty}^{\infty} e^{t(z\sigma + \mu)} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}z^2} \sigma dz$$
$$= e^{t\mu} \int_{-\infty}^{\infty} e^{tz\sigma} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(z\sigma)^2} dz$$
$$= e^{t\mu} \int_{-\infty}^{\infty} e^{(\frac{z\sigma}{2})^2} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(z\sigma)^2} dz$$
$$= e^{t\mu} e^{\frac{(\sigma^2 t)^2}{2}}$$
$$= e^{t\mu + \frac{1}{2}\sigma^2 t^2}. \quad (A.1)$$

The moment generating function $\theta$ when $\theta|x$ has Gamma distribution with shape $\alpha$ and rate $\beta$:

$$M_\theta(t) = \mathbb{E}_x[e^{t\theta}] = \int_0^{\infty} e^{t\theta} \frac{\beta^\alpha}{\Gamma(\alpha)} \theta^{\alpha-1} e^{-\beta \theta} d\theta$$
$$= \int_0^{\infty} \frac{\beta^\alpha}{\Gamma(\alpha)} \theta^{\alpha-1} e^{-(\beta-t)\theta} d\theta$$
$$= \frac{\beta^\alpha}{(\beta-t)^\alpha} \int_0^{\infty} (\beta - t)^\alpha \theta^{\alpha-1} e^{-(\beta-t)\theta} d\theta$$
$$= \left( \frac{\beta}{\beta-t} \right)^\alpha. \quad (A.2)$$
Appendix B

**Algorithm B.1. Update $\hat{\omega}(t)$**

Given $\hat{\omega}(t)$.

1. Generate $\epsilon \sim N(0, 0.5)$.
2. Sample $i$ from the discrete uniform distribution on the integers $\{1, \ldots, k\}$.
3. Take $\alpha_i$ where
   \[
   \alpha_j = \begin{cases} 
   \epsilon & \text{if } i = k, \\
   -\frac{\epsilon}{k-1} & \text{otherwise}
   \end{cases}
   \]
   for $j = 1, \ldots, k$.
4. $\epsilon^* = (\alpha_1, \alpha_2, \ldots, \alpha_k)$
5. Update $\hat{\omega}' = \hat{\omega}(t) + \epsilon^*$

**Algorithm B.2. Update $\hat{\mu}(t)$**

Given $\hat{\mu}(t) = (\hat{\mu}_1(t), \hat{\mu}_2(t), \ldots, \hat{\mu}_k(t))$.

1. Generate $\hat{\mu}_j' \sim N(\hat{\mu}_j(t), 0.5)$ for $j = 1, \ldots, k$.
2. Update $\hat{\mu}' = (\hat{\mu}_1', \hat{\mu}_2', \ldots, \hat{\mu}_k')$
Algorithm B.3. Update $\hat{\sigma}^{2(t)}$

Given $\hat{\sigma}^{2(t)} = (\hat{\sigma}_1^{2(t)}, \hat{\sigma}_2^{2(t)}, \ldots, \hat{\sigma}_k^{2(t)})$.

1. Generate $\hat{\sigma}_j' \sim \text{N}(\hat{\sigma}_j^{2(t)}, 0.5)$ for $j = 1, \ldots, k$.

2. Take

   $$\hat{\sigma}_j' = \begin{cases} 
   \hat{\sigma}_j' & \text{if } \hat{\sigma}_j' > 0, \\
   -\hat{\sigma}_j' & \text{otherwise}
   \end{cases}$$

   for $j = 1, \ldots, k$.

3. Update $\hat{\sigma}' = (\hat{\sigma}_1', \hat{\sigma}_2', \ldots, \hat{\sigma}_k')$. 
Appendix C

Algorithm C.1. Estimation corresponding to the new grid points $\hat{\theta}_1$ and $\hat{\theta}_2$

1. Find location $r$ and $s$ such that

$$\theta_{1r} \leq \hat{\theta}_1 < \theta_{1r+1},$$

$$\theta_{2s} \leq \hat{\theta}_2 < \theta_{2s+1},$$

for $1 \leq r \leq n_1$ and $1 \leq s \leq n_2$.

2. Pass some values of $E_{ij}^{kl}$ for $k, l = 0, \ldots, 3$ to $\hat{E}_{ij}^{kl}$ as follows.

\[
\begin{align*}
\hat{E}_{ij}^{kl} &= E_{ij}^{kl} \quad \text{for } i < r \text{ and } j < s \\
\hat{E}_{i+1j}^{kl} &= E_{ij}^{kl} \quad \text{for } i > r \text{ and } j < s \\
\hat{E}_{ij+1}^{kl} &= E_{ij}^{kl} \quad \text{for } i < r \text{ and } j > s \\
\hat{E}_{i+1j+1}^{kl} &= E_{ij}^{kl} \quad \text{for } i > r \text{ and } j > s.
\end{align*}
\]

In this step, we introduce the new notation for expected values, $\hat{E}_{ij}^{kl}$ to avoid confusion. We use the exact expected values with some changes in indices according to the new grid points $\hat{\theta}_1$ and $\hat{\theta}_2$.

3. Estimate expected values $\hat{E}_{ij}^{kl}$ for either $i = r$ or $j = s$ as follows.

- For $i = r$ and $j \neq s$

\[
\frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \theta_1}{\theta_1 - \theta_1^i} \right)^k \left( \frac{\theta_2^{(n)} - \theta_2}{\theta_{2j+1} - \theta_2} \right)^l \approx \begin{cases} \hat{E}_{ij}^{kl} & \text{for } j < s \\ \hat{E}_{ij+1}^{kl} & \text{for } j > s + 1. \end{cases}
\]


– use \((\theta_1^{(n)}, \theta_2^{(n)}) \in [\hat{\theta}_1, \theta_{1i+1}] \times [\theta_{2j}, \theta_{2j+1}]\) to obtain
\[
\frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \hat{\theta}_1}{\theta_{1i+1} - \theta_1} \right)^k \left( \frac{\theta_2^{(n)} - \theta_{2j}}{\theta_{2j+1} - \theta_2} \right)^l \approx \left\{ \begin{array}{ll}
\hat{\theta}_{ij}^{kl} & \text{for } j < s \\
\hat{\theta}_{i+1j}^{kl} & \text{for } j > s + 1.
\end{array} \right.
\]

• For \(i \neq r\) and \(j = s\)
  – use \((\theta_1^{(n)}, \theta_2^{(n)}) \in [\theta_1, \theta_{1i+1}] \times [\theta_{2j}, \hat{\theta}_2]\) to obtain
\[
\frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \theta_1}{\theta_{1i+1} - \theta_1} \right)^k \left( \frac{\theta_2^{(n)} - \hat{\theta}_2}{\theta_{2j+1} - \theta_2} \right)^l \approx \left\{ \begin{array}{ll}
\hat{\theta}_{ij}^{kl} & \text{for } i < r \\
\hat{\theta}_{i+1j}^{kl} & \text{for } i > r + 1,
\end{array} \right.
\]

• For \(i = r\) and \(j = s\),

  – use \((\theta_1^{(n)}, \theta_2^{(n)}) \in [\theta_1, \hat{\theta}_1] \times [\theta_{2j}, \hat{\theta}_2]\) to obtain
\[
\hat{\theta}_{ij}^{kl} \approx \frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \hat{\theta}_1}{\theta_1 - \theta_1} \right)^k \left( \frac{\theta_2^{(n)} - \hat{\theta}_2}{\theta_2 - \theta_2} \right)^l,
\]

  – use \((\theta_1^{(n)}, \theta_2^{(n)}) \in [\hat{\theta}_1, \theta_{1i+1}] \times [\theta_{2j}, \hat{\theta}_2]\) to obtain
\[
\hat{\theta}_{i+1j}^{kl} \approx \frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \hat{\theta}_1}{\theta_{1i+1} - \theta_1} \right)^k \left( \frac{\theta_2^{(n)} - \hat{\theta}_2}{\theta_{2j+1} - \theta_2} \right)^l,
\]

  – use \((\theta_1^{(n)}, \theta_2^{(n)}) \in [\hat{\theta}_1, \theta_{1i+1}] \times [\hat{\theta}_2, \theta_{2j+1}]\) to obtain
\[
\hat{\theta}_{ij+1}^{kl} \approx \frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \hat{\theta}_1}{\theta_1 - \theta_1} \right)^k \left( \frac{\theta_2^{(n)} - \hat{\theta}_2}{\theta_{2j+1} - \theta_2} \right)^l,
\]

  – use \((\theta_1^{(n)}, \theta_2^{(n)}) \in [\hat{\theta}_1, \theta_{1i+1}] \times [\hat{\theta}_2, \theta_{2j+1}]\) to obtain
\[
\hat{\theta}_{i+1j+1}^{kl} \approx \frac{1}{N} \sum_{n=1}^{N} \left( \frac{\theta_1^{(n)} - \hat{\theta}_1}{\theta_{1i+1} - \theta_1} \right)^k \left( \frac{\theta_2^{(n)} - \hat{\theta}_2}{\theta_{2j+1} - \theta_2} \right)^l.
\]

201


