BLACK

Bayesian Analysis of Linear Spatio-Temporal Models

Linda Garside

A thesis submitted for the degree of Doctor of Philosophy at the

University of Newcastle upon Tyne

October 24, 2003

"Nothing would be done at all

if a man waited till he could do it so well

that no one could find fault with it"

CARDINAL JOHN HENRY NEWMAN (1801-1890)

Abstract

Spatio-temporal models provide a mechanism for analysing data that occurs naturally in space and time such as pollution levels, functional magnetic resonance imaging data and temperature data. These models aim to capture the important features of the space time structure that can be overlooked by examining the spatial and temporal features separately. In this thesis a dynamic linear model (DLM) is used to describe a lattice Markov spatio-temporal system with Markov chain Monte Carlo (MCMC) techniques used to obtain estimates for the model parameters from the marginal posterior distributions.

This thesis is concerned with the modelling of the latent structure of a Bayesian spatio-temporal model with a view to improving parameter inference, smoothing and prediction. The equilibrium distribution of a time stationary system will be examined, paying particular attention to edge effects and the effect of grid coarsening. In order to develop an effective MCMC algorithm the latent process is integrated out of the model. These techniques are illustrated using both simulated data and North Atlantic ocean temperature data.

Acknowledgements

The work in this thesis has been possible due to the funding provided by EPSRC. A huge thank you to everyone who has helped and supported me during the past three years, especially my supervisor Darren Wilkinson for his assistance and patience. I would also like to thank Johann for all his help both with the proof reading and for all the hugs to cheer me up. Thanks to June for a never ending supply of tea, Lynda and Marilyn for listening, my parents for their support, Chadders for proof reading some of the technical work and everyone in the office for making me laugh.

Contents

1 Background and Motivation **1**

1.1 Introduction 1

1.2 A Brief History of Space Time Modelling 2

1.3 MCMC in Spatio-Temporal Models 9

1.4 Structure of the Thesis 10

2 The Model and Notation **12**

2.1 Introduction 12

2.2 The Dynamic Linear Model 13

2.2.1 Examples 15

2.3 Notation 17

2.4 STAR Models 19

2.4.1 STARMA models 19

2.5 The observation equation 20

2.6 Problem specific declarations 21

2.6.1 The Latent structure 21

2.6.2 Building the evolution weight matrix: 1+1D model 22

2.6.3 Building the evolution weight matrix: 2+1D model 24

2.7 Discussion 27

3 Edge Effects **29**

3.1 Introduction 29

3.2 The Conditional Distribution 34
3.2.1 The Stationary Variance Matrix 35
3.3 1+1D model 38
3.3.1 Numerical Example 41
3.4 2+1D model 41
3.4.1 Numerical Example 47
3.5 Extensions 49
3.5.1 Changing the state precision 49
3.5.2 Changing the Mean 52
3.6 Discussion 53

4 Markov Chain Monte Carlo Simulation Techniques **54**
4.1 Introduction 54
4.2 Gibbs Sampling 55
4.2.1 The Gibbs Algorithm 56
4.2.2 Full conditionals for the lattice DLM 56
4.2.3 Implementation 65
4.2.4 Example 65
4.2.5 Discussion 67
4.3 Data Augmentation 72
4.3.1 Data Augmentation algorithm 73
4.3.2 Implementation 73
4.3.3 Example 74
4.3.4 Discussion 76
4.4 Metropolis-Hastings 77
4.4.1 Metropolis-Hastings Algorithm 78
4.4.2 Independent Parameter Updates 78
4.4.3 Dependent Parameter Updates 79
4.4.4 Implementation 80

4.4.5 Example: 1+1D model 80
4.4.6 Example: 2+1D model 83
4.4.7 Discussion 86
4.5 Parallel chains approach 88
4.5.1 Example 88
4.6 Introducing the mean parameter 90
4.6.1 Example 91
4.7 Discussion 93

5 The Coarse Model **95**
5.1 Introduction 95
5.2 The Second Order Markov Process 96
5.3 The Conditional Distribution 97
5.3.1 Creating a reference table 101
5.4 1+1D model 101
5.4.1 Numerical Example 103
5.5 2+1D Model 104
5.5.1 Numerical Example 107
5.6 MCMC Example 107
5.6.1 1+1D model 108
5.6.2 2+1D model (1) 112
5.6.3 2+1D model (2) 116
5.7 Discussion 119

6 Examination of Temporal and Spatial Dependencies **121**
6.1 Introduction 121
6.2 Temporal parameter 122
6.2.1 Example 122
6.3 The spatial dependence parameter 128

6.3.1 Example 129
6.4 Examining the spatial and temporal dependencies together 135
6.4.1 Example 1 : Small data set 135
6.4.2 Example 2 : Large data set 136
6.5 Discussion 136

7 Case Studies **139**
7.1 Introduction 139
7.2 Case Study 1: Small Data Set 140
7.2.1 Results 142
7.3 Case Study 2: Large Data Set 150
7.3.1 Results 152
7.4 Discussion 159

8 Discussion and Further work **161**

A Reference Tables of Edge Weights for the 2+1D Model in the Fine and Coarse Grid Setting. **170**
A.1 Reference table for the fine model 171
A.2 Reference table for the coarse model 176

B Extracts from the computer code **180**

List of Figures

1.1 *Spatial locations of the underlying grid process are denoted by the black dots. One standard deviation ellipses corresponding to the covariance matrix of the Gaussian smoothing kernels are also shown.* 7

2.1 *Conditional independence graph for the DLM showing the relationship between the state parameters* $\theta^{(t)}$ *and observations* $X^{(t)}$. 13

2.2 *Evolution through time of the latent process in the 0+1D model.* 15

2.3 *Evolution through time of the latent process in the 1+1D model.* 16

2.4 *Evolution through time of the latent process in the 2+1D model.* 16

2.5 *The parent structure for the 1+1D spatio-temporal model far (a) a directed graph (b) an undirected graph.* 19

2.6 *The graphical representation of the hidden layer in the 1+1D model.* 23

2.7 *The four edge weights* p_{11}, p_{12}, p_{21} *and* p_{22} *for the hidden layer in the 2+1D model.* 25

2.8 *The graphical representation of the hidden layer in the 2+1D model for a second order parent system.* 28

3.1 *Precision of the system on a* 10×10 *spatial grid at time* $T = 20$ *with unadjusted edge weights for (a)* $\alpha = 0.7$, *(b)* $\alpha = 0.9$, *(c)* $\alpha = 0.95$, *(d)* $\alpha = 0.99$. 31

3.2 *Precision of the system on a* 10×10 *spatial grid at time T=20 with scaled edge weights for (a)* $\alpha = 0.7$, *(b)* $\alpha = 0.9$, *(c)* $\alpha = 0.95$, *(d)* $\alpha = 0.99$. . . . 33

3.3 *Precision of the system for a 100 spatial node model at* $T = 20$ *and* $\alpha = 0.95$ *with (a) no edge adjustments, (b) scaled weights and (c) conditional weights.* 42

3.4 *Precision of the system on a* 10×10 *spatial grid at time t=20 with fine edge adjustments for (a)* $\alpha = 0.7$, *(b)* $\alpha = 0.9$, *(c)* $\alpha = 0.95$ *and (d)* $\alpha = 0.99$. . 48

4.1 *The evolution of the 1+1D model showing both forward and backward dependencies for the central node* $\theta_i^{(t)}$. 57

4.2 *The trace, density and auto correlation plots for the two precision parameters* τ_s *and* τ_o *for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the Gibbs sampler implemented in BUGS with 20,000 iterations.* 67

4.3 *The trace, density and auto correlation plots for the two precision parameters* τ_s *and* τ_o *for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the Gibbs sampler implemented in 'C' with 20,000 iterations.* 68

4.4 *The trace, density and autocorrelation plots for the two precision parameters* τ_s *and* τ_o *for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the data augmentation approach with 20,000 iterations.* 75

4.5 *The trace, density and auto correlation plots for the two precision parameters* τ_s *and* τ_o *for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the adjusted edge weight model and the Metropolis-Hastings approach with 20,000 iterations.* 81

4.6 *The trace, density and auto correlation plots for the two precision parameters* τ_s *and* τ_o *for the 2+1D model with a* 10×10 *spatial grid evolving over 20 time periods generated using the Gibbs sampler implemented in BUGS with 20,000 iterations.* 84

4.7 *The trace, density and autocorrelation plots for the two precision parameters τ_s and τ_o for the 2+1D model with a 10×10 spatial grid evolving over 20 time periods generated using the Metropolis-Hastings algorithm.* 87

4.8 *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model with a 10×10 spatial grid evolving over 20 time periods generated using a parallel approach with eight chains each of 5000 iterations.* 89

4.9 *The trace,density and auto correlation plots for the two precision parameters τ_s, τ_o and the mean parameter μ for the 2+1D model with a 10×10 spatial grid evolving over 20 time periods generated using Metropolis-Hastings.* 92

5.1 *The graphical representation of the hidden layer in the 1+1D model over two time periods.* 97

5.2 *The second order Markov process for the fine 1+1D model.* 98

5.3 *The graphical representation of the 1+1D coarse model which uses the second order Markov process.* 102

5.4 *Precision of the 1+1D system for a 100 spatial node model at T=20 using the coarse model for (a) $\alpha = 0.7$, (b) $\alpha = 0.9$, (c) $\alpha = 0.95$, (d) $\alpha = 0.99$.* . . 104

5.5 *The graphical representation of the 2+1D coarse model which uses the second order Markov process.* 104

5.6 *Precision of the system on a 10×10 spatial grid at time $T = 20$ using the coarse model for (a) $\alpha = 0.7$, (b) $\alpha = 0.9$, (c) $\alpha = 0.95$, (d) $\alpha = 0.99$.* . . . 108

5.7 *The trace and density plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the coarse model with a complete set of data.* 110

5.8 *The trace and density plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the coarse model with thinned data.* 111

5.9 *Plots of the mean of the simulated latent structure at time period $t = 9$.* . . 113

5.10 *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model on a* 10×10 *spatial spatial evolving over 20 time periods generated using the coarse model with a complete set of data.* 114

5.11 *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model on a* 10×10 *spatial spatial evolving over 20 time periods generated using the coarse model with thinned data.* 115

5.12 *Surface plots generated on the fine* 10×10 *spatial grid using the estimate from the coarsened model at (a) Time period 0, (b) Time period 4, (c) Time period 14 and (d) Time period 19.* 117

5.13 *Predictive surface plot for the generated* 10×10 *model.* 118

5.14 *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model with a* 20×20 *spatial grid evolving over 20 time periods generated using a parallel approach with eight chains each of 5000 iterations.* 119

6.1 *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is* $Beta(1,1)$ *and is shown by the dashed line on the density plot.* 123

6.2 *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is* $Beta(2,3)$ *and is shown by the dashed line on the density plot.* 124

6.3 *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is* $Beta(3,2)$ *and is shown by the dashed line on the density plot.* 125

6.4 *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is* $Beta(1,5)$ *and is shown by the dashed line on the density plot.* 126

6.5 *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is* $Beta(5,1)$ *and is shown by the dashed line on the density plot.* 127

6.6 *The five parent system for the 2+1D model.* 128

6.7 *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(1,1)$ *and is shown by the dashed line on the density plot.* 130

6.8 *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(2,3)$ *and is shown by the dashed line on the density plot.* 131

6.9 *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(3,2)$ *and is shown by the dashed line on the density plot.* 132

6.10 *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(1,5)$ *and is shown by the dashed line on the density plot.* 133

6.11 *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(5,1)$ *and is shown by the dashed line on the density plot.* 134

6.12 *Trace and density plots for the state precision, observation precision spatial and temporal dependency parameter from an MCMC run with 500,000 iterations on a* 5×5 *spatial grid over ten time periods. The prior distribution for the both the spatial and temporal dependency parameter is* $Beta(1,1)$ *and is shown by the dashed line on the density plots.* 137

6.13 *Trace and density plots for the state precision, observation precision spatial and temporal dependency parameter from an MCMC run with 500,000 iterations on a* 10×10 *spatial grid over twenty time periods. The prior distribution for the both the spatial and temporal dependency parameter is* $Beta(1,1)$ *and is shown by the dashed line on the density plots.* 138

7.1 *Spatial location of the data with shade representing time.* 141

7.2 *Trace and density plots for ocean temperature data modelled using the fine model on a* 3×8 *spatial grid.* 143

7.3 *The coarse and fine grid points for the second model.* 144

7.4 *Trace and density plots for ocean temperature data modelled using the fine model on a* 6×15 *spatial grid.* 145

7.5 *Trace and density plots for ocean temperature data modelled using the coarse model on a* 3×8 *spatial grid.* 146

7.6 *Trace and density plots for ocean temperature data modelled using the coarse model on a* 6×30 *spatial grid.* 148

7.7 *Mean temperature surface for 1982 on a* 12×60 *spatial grid.* 149

7.8 *Mean temperature surface for 1985 on a* 12×60 *spatial grid.* 149

7.9 *Mean temperature surface for 1988 on a* 12×60 *spatial grid.* 150

7.10 *Predicted mean temperature surface for 1989 on a* 12×60 *spatial grid.* . . . 150

7.11 *Spatial location of the data with shade representing time.* 151

7.12 *Trace and density plots for ocean temperature data modelled using the fine model on a* 4×4 *spatial grid.* 153

7.13 *Trace and density plots for ocean temperature data modelled using the coarse model on a* 4×4 *spatial grid after the removal of 100 iterations as burn-in* 155

7.14 *Trace and density plots for ocean temperature data modelled using the fine model on a* 8×8 *spatial grid.* 156

7.15 *Trace and density plots for ocean temperature data modelled using the coarse model on a* 16×16 *spatial grid.* 158

7.16 *Mean temperature surface for (a) 1959 (b) 1973 (c) 1979 (d) 1983 on a* 32×32 *spatial grid.* 159

7.17 *Predicted mean temperature surface for 1985 on a* 32×32 *spatial grid.* . . 160

List of Tables

3.1 *Stationary precision and the associated range for a 10 × 10 spatial model at time $T = 20$ with unadjusted edge weights.* 32

3.2 *Stationary Precision and the associated range for 10 × 10 spatial model at time $T = 20$ with scaled edge weights.* 32

3.3 *Extract of the reference table for the 2+1D model with fine edge weights showing the edge and precision adjustments for six values of temporal dependence.* 37

3.4 *Stationary Precision and the associated range for a 100 spatial node model at time $T = 20$ with no edge weights.* 43

3.5 *Stationary Precision and the associated range for a 100 spatial node model at time $T = 20$ with scaled edge weights.* 43

3.6 *Stationary Precision and the associated range for a 100 spatial node model at time $T = 20$ with conditional edge weights.* 43

3.7 *Stationary Precision and the associated range for 10 × 10 spatial model at time $T = 20$ with adjusted edge weights.* 49

4.1 *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in BUGS.* 66

4.2 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in BUGS.* 66

4.3 *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in 'C'* 67

4.4 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in 'C'* 68

4.5 *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the data augmentation approach.* . . 74

4.6 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the data augmentation approach.* 75

4.7 *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Metropolis-Hastings approach.* . 81

4.8 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Metropolis-Hastings approach.* 82

4.9 *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Gibbs sampler implemented in BUGS.* 85

4.10 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Gibbs sampler implemented in BUGS.* 85

4.11 *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm.* . . 86

4.12 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm.* 86

4.13 *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm in a parallel approach.* 90

4.14 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm in a parallel approach.* 90

4.15 *The mean and standard deviation for the precision and mean parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods.* 92

4.16 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods.* 93

5.1 *Stationary precision and the associated range for a 100 spatial node model at time* $T = 20$ *with coarse edge weights.* 103

5.2 *Stationary precision and the associated range for a 10 × 10 spatial model at time* $T = 20$ *using the coarse model.* 107

5.3 *Summary statistics for the generated 1+1D data.* 110

5.4 *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1d model with 100 spatial nodes evolving through 20 time periods using the coarse model with thinned data.* 112

5.5 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1d model with 100 spatial nodes evolving through 20 time periods using the coarse model with thinned data.* 112

5.6 *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.* 113

5.7 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.* 114

5.8 *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 20×20 *spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.* 117

5.9 *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 20×20 *spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.* 118

7.1 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 3×8 *spatial grid.* 143

7.2 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 3×8 *spatial grid.* 144

7.3 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 6×15 *spatial grid.* 145

7.4 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 6×15 *spatial grid.* 146

7.5 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 3×8 *spatial grid.* 147

7.6 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 3×8 *spatial grid.* 147

7.7 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 6×30 *spatial grid.* 147

7.8 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 6×30 *spatial grid.* 148

7.9 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 4×4 *spatial grid.* 153

7.10 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 4×4 *spatial grid.* 154

7.11 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 4×4 *spatial grid.* 155

7.12 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 4×4 *spatial grid.* 156

7.13 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 8×8 *spatial grid.* 157

7.14 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 8×8 *spatial grid.* 157

7.15 *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 16×16 *spatial grid.* 157

7.16 *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 16×16 *spatial grid.* 158

A.1 *The reference table for the fine 2+1D model.* 175

A.2 *The reference table for the coarse 2+1D model.* 179

Chapter 1

Background and Motivation

1.1 Introduction

Spatio-temporal models are used to describe data that is spread through space and time, e.g. Functional magnetic resonance imaging (FMRI) (Genovese 1999), pollution levels, wind speed (Haslett and Raftery 1989; Tonellato 1998) or other environmental data (Haung and Cressie 1996). Modelling the spatial or temporal component of the data is relatively easy however modelling the spatial and temporal parts simultaneously is significantly more complex. Christakos (1991) discusses space time modelling and states;

"Typically space time data processing problems have been handled under some convenient but rather simplistic assumption of treating space and time separately ... in environmental pollution monitoring and research the existing models either apply traditional methods of classical statistics that are incapable to capture important features of the space time structure, or have been designed to handle problems that are significantly different in nature than those arising in the spatio-temporal data processing context."

Spatio-temporal data is dealt with in two main ways; in a spatio-temporal framework as discussed in this thesis or with the examination of spatial and temporal trends separately. In the latter case, if either of the trends are found to be negligible then the data is examined in a purely spatial or temporal setting as in Haas (1995) and Handcock &

Wallis (1994).

In this thesis spatio-temporal models are examined within the dynamic linear model (DLM) framework using Bayesian and Markov Chain Monte Carlo (MCMC) methods to identify the models on a spatial lattice grid evolving through time. Interest in spatio-temporal models originated in the mid 1970's and continues today. There have been numerous different approaches which have developed along side the development of computers with the increased use of Markov Chain Monte Carlo approaches. The main methods developed are presented briefly in chronological order in this chapter along with an introduction to Markov Chain Monte Carlo methods. This chapter concludes with a description of the structure of this thesis.

1.2 A Brief History of Space Time Modelling

Interest in spatio-temporal modelling started in the mid 1970's with the development of space-time auto-regressive moving-average (STARMA) models by Cliff & Ord (1975) and Pfeifer & Deutsch (1980*d*). Since then other methods have been developed and described in the literature including; linked time series models as in Haslett & Raftery (1989); space time random fields as in Christakos (1991); space time decomposition as in Høst et al. (1995); the use of temporal pre-whitening and spatial deformation as in Meiring et al. (1998); hierarchical models using Markov random fields as in Waller et al. (1997) and Wikle et al. (1998); unobserved field and measurement error as in Brown et al. (2001) and Huang & Cressie (1996); process convolution as in Higdon (1998); dynamic models as in Stroud et al. (2001).

The discrete space time formulation by Cliff & Ord (1975) and Pfeifer & Deutsch (1980*d*) develops a STARMA model of the form

$$\begin{aligned} \mathbf{z}(t) &= \sum_{k=1}^{p}\sum_{l=0}^{\lambda_k}\phi_{kl}W^{(l)}\mathbf{z}(t-k) \\ &\quad -\sum_{k=1}^{q}\sum_{l=0}^{m_k}\theta_{kl}W^{(l)}\epsilon(t-k)+\epsilon(t) \end{aligned}$$

where $\mathbf{z}(t)$ is the vector of observations at time t. The evolution of the observations is given by the sum of AR and MA models. $W^{(l)}$ is a weight matrix of order l, ϕ_{kl} and θ_{kl} are the AR and MA parameters respectively. Further details of this model are given in Chapter 2. The identification of STARMA models, the form of the covariance matrix and the stationarity of these models is examined in detail by Pfeifer & Deutsch in a number of different papers published in 1980, Pfeifer & Deutsch (1980*c*), Pfeifer & Deutsch (1980*b*) and Pfeifer & Deutsch (1980*a*).

Haslett & Raftery (1989) examine a linked time series model for wind speed data from Ireland. Here interest is in the power available at each site which can be calculated from the wind speed data. Inference for mean wind speed at spatial location k is based on a single model for the entire space time process

$$X_{it} = \mu_{it} + \nabla^{-d}\phi(B)^{-1}\theta(B)\epsilon_{it}.$$

Where B is the backwards shift operator defined such that $B\epsilon_{it} = \epsilon_{i,t-1}$, $\nabla = (1-B)$, ∇^{-d} is defined by the binomial series expansion of $(1-B)^{-d}$, $\mu_{i,t}$ is the mean wind speed at location i, time t and ϵ_t are independent identically distributed (i.i.d.) multivariate Normal with mean zero and variance $\sigma_\epsilon^2\mathbf{R}$ where $\mathbf{R}$ is a matrix whose elements are defined to be a function of distance between sites. Estimation of power at a new site is done in two stages, firstly the distribution of wind speed is estimated and then it is translated to the associated kinetic energy. The second order moment structure of the space time process is approximately isotropic and stationary in space and time. The parameters in the model are estimated using maximum likelihood methods.

Christakos (1991) develops a space-time random field (S/TRF) model. The aim of this method is to describe the correlation structure of space non-homogeneous/time non-stationary processes and derive the optimal estimators for data simultaneously in space and time. This model is a generalisation of the discrete model to the continuous space time context. The S/TRF model has been studied in more detail and applied to environmental data by Christakos (1992), Christakos & Bogaert (1996), Vyas & Christakos (1997), Christakos & Vyas (1998) and Christakos (1998).

Høst et al. (1995) develop a model using space time decomposition. The space time process $Y(x,t)$ is decomposed into three random fields representing the mean $M(x,t)$, standard deviation (or scale) $S(x,t)$ and residual $U(x,t)$, where U has mean 0 and variance 1.

$$Y(x,t) = M(x,t) + S(x,t)U(x,t).$$

This method then goes on to treat M, S and U as independent stochastic fields and treats the spatial and temporal effects as separable. Hence for the separable model the mean field $M(x,t)$ is given by

$$M(x,t) = F(x) + \eta(t)1$$

where $\eta(t)$ represents zero mean temporal noise and $F(x)$ is the average mean spatial field. The scale field is defined by

$$S(x,t) = H(x)\kappa(t)$$

where $\kappa(t)$ represents unit mean temporal noise and $H(x)$ represents the time invariant spatial scale field.

Hierarchical models have been used by a number of people including Waller et al. (1997) and Wikle et al. (1998), the two examples discussed here use a Bayesian approach to develop a hierarchical model to different data sets. Waller et al. (1997) introduce a Bayesian approach to model spatio-temporal disease rate data using a regression model, μ_{ilt} the parameter of interest is defined to be the log of the relative risk

$$\mu_{ilt} = log\psi_{ilt}.$$

μ_{ilt} is modelled by

$$\mu_{ilt} = x_l^T\beta + z_i^T\omega + \theta_i^{(t)} + \phi_i^{(t)}$$

where l is the subgroup, i the spatial location and t the time period, x_l^T is the design vector, β is a parameter vector, z_i is a specified parametric function, $\theta_i^{(t)}$ represents the

random effect and $\phi_i^{(t)}$ represents the spatial effect at spatial location i and time period t. The Bayesian modelling is carried out in three stages:

1. Specify the likelihood for vector of observed counts y given the vector of relative risk ψ.

2. Specify a prior model over possible ψ's.

3. Use MCMC to yield the marginal posterior distribution $\psi|y$.

Wikle et al. (1998) also uses a Bayesian approach and the parameters are simulated using a Gibbs sampler with a nested Metropolis-Hastings step. The model is developed in five stages:

1. Define the state process

$$Z(s,t) \sim N\left(Y(s,t), \sigma_{s,t}^2\right)$$

where the observational data $Z(s,t)$ is modelled by a Normal distribution with mean $Y(s,t)$ (which is defined in Stage 2) and measurement error variances $\sigma_{s,t}^2$.

2. Define the large and small scale features via the definition of $Y(s,t)$.

$$Y(s,t) = \mu(s) + M(t;\beta(s)) + X(s,t) + \gamma(s,t)$$

where $\mu(s)$ is the site specific mean, $M(t;\beta(s))$ defines the large scale temporal model with site specific parameters $\beta(s)$, $X(s,t)$ is the short time scale dynamic process and $\gamma(s,t)$ represents the zero mean noise random variable.

3. Define the spatial structures and dynamics. Assume the model is of STARMA form and μ, β and X are mutually exclusive conditional on hyper-parameters $\theta_3 = (\theta_\mu, \theta_\beta, \theta_X)$;

$$[\mu, \beta, X|\theta_3] = [\mu|\theta_\mu][\beta|\theta_\beta][X|\theta_X],$$

$[\mu\,|\theta_\mu]$ and $[\beta\,|\theta_\beta]$ are defined using Markov random fields and $[X\,|\theta_X]$ is defined as a stochastic model expressed as

$$\prod_t [X(t+1)\,|X(t), \theta_X]\,.$$

4. Define prior distributions for parameters μ, β and X.

5. Define prior distributions for the hyper-parameters θ_3.

Once the model has been fully defined a Gibbs sampler with nested Metropolis-Hastings step is used to simulate the parameter estimates. These two methods allow larger and more complex models to be analysed compared with previous methods.

Temporal pre-whitening and spatial deformation is the method presented by Meiring et al. (1998) for the analysis of grid-cell hourly ozone levels. Their approach represents a non separable space time correlation structure which allows for spatially varying mean fields and diurnal variability in the correlation of residuals. This method uses three steps to develop the space time model:

1. Transform the data so variations about the site mean are approximately symmetric; in this example an auto-regressive filter of order two is used to temporally pre-whiten the residuals after the removal of hourly means.

2. Model the spatially varying mean structure and residual temporal correlation. The pre-whitened residual $Y_t\,(x_i)$ is assumed to be uncorrelated in time but correlated in space. The model is given by

$$Y_t\,(x) = Q_t(x) + \epsilon_t(x)$$

where the spatial correlation between $Q_t(x)$ and $Q_t(u)$ is a smooth non stationary function of spatial location and $\epsilon_t(x)$ is the measurement error.

3. Use a spatial deformation approach to model spatially non stationary spatial correlation in pre-whitened residuals. The spatial dispersion $D_k(u,k)$ assumes that daily

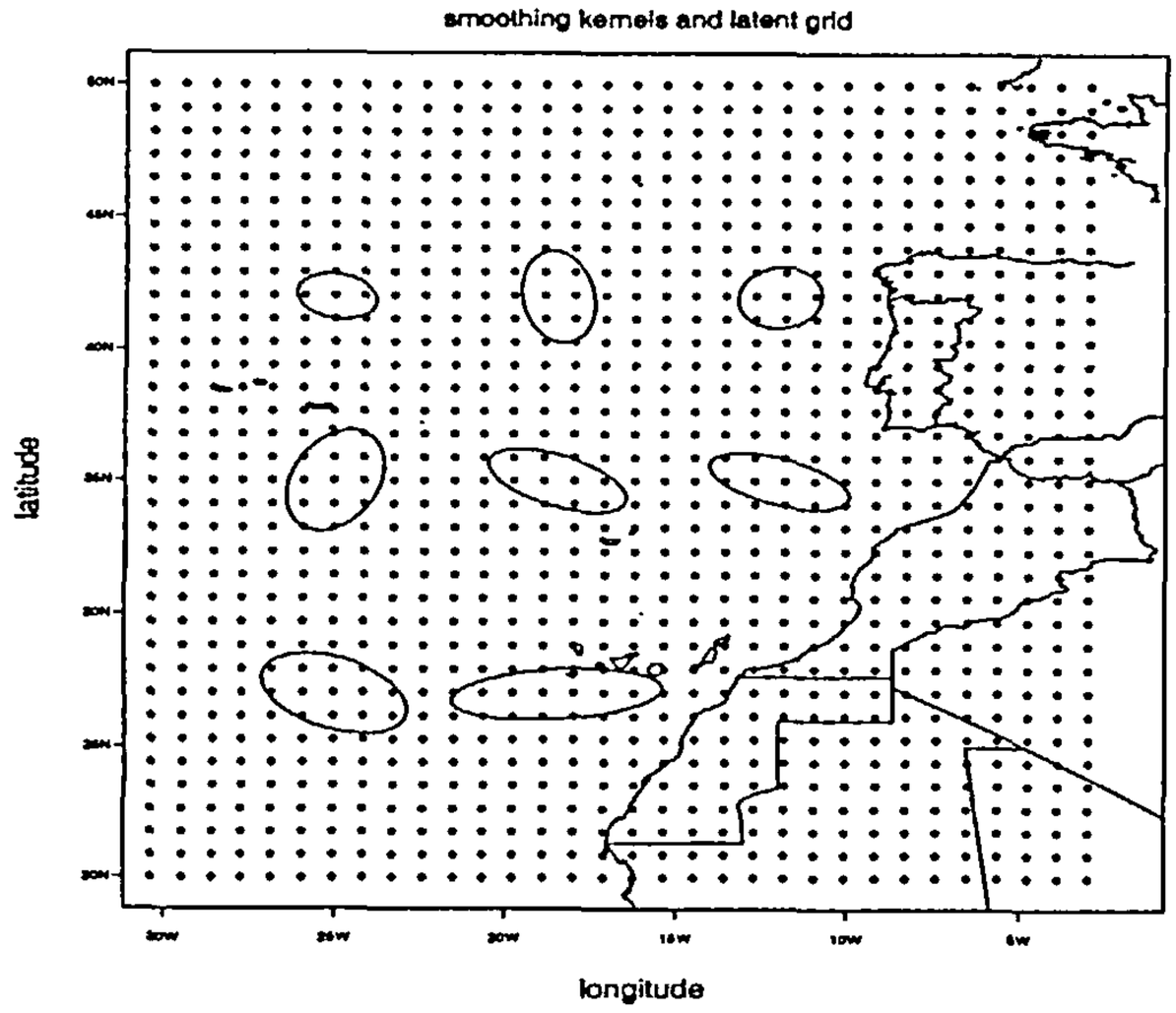


Figure 1.1: *Spatial locations of the underlying grid process are denoted by the black dots. One standard deviation ellipses corresponding to the covariance matrix of the Gaussian smoothing kernels are also shown.*

time slices of residuals for a given hour are independent spatial samples from the process $Y_t(x)$ at this time period thus providing replications to use in estimating the spatial correlation at each hour.

Higdon (1998) develops a process convolution approach which focuses on the large scale properties of the field and changes over time on a lattice grid. The temperature field is defined by

$$y(s,t) = z(s,t) + \epsilon(s,t)$$

where $z(s,t)$ denotes a smooth Gaussian process which is defined to be the convolution of a Gaussian white noise process and $\epsilon(s,t)$ is an independent error process. The smooth process $z(s,t)$ is constructed by taking the convolution of a 3-D lattice process, $z(s,t) = \sum K_s(s-\omega_j, t-\tau_j) \cdot x_j$. The convolution kernels depend on location s. This paper considers a separable kernel so K_s is represented as a product of a spatial kernel and temporal kernel. The estimation of the kernel K_s is achieved using MLE techniques which are computationally demanding so the kernel is estimated at a small number of spatial

locations and then interpolated for the remaining locations. The example in this paper models the temperature of the North Atlantic using eight kernels (see Figure 1.1), the size and shape of these kernels is related to the prior information about the temperature of the ocean and known currents in the area. The posterior distribution for this model is explored using MCMC techniques.

Brown et al. (2001) and Huang & Cressie (1996) develop models which use an unobserved field and measurement error modelling spatio-temporal data. Brown et al. (2001) examines radar data which measures the amount of rainfall. In this example the parameters are estimated using maximum likelihood. These parameters are then used to construct the mean square error predictors of the latent process $\lambda(x,t)$ and then predict the unobserved values of $Y(x,t)$ as

$$Y(x,t) = \begin{cases} \lambda(x,t) + \beta\mu(x,t) & R_{i,t} > 0 \\ 0 & \text{otherwise} \end{cases}$$

where $\lambda(x,t)$ is the latent process, $\mu_{i,y} = \log R_{it}$ and R_{it} is the radar reading. By defining the model in this way they can take into account days with zero rainfall. Huang & Cressie (1996) examines snow water equivalent data where $Y(x,t)$ is modelled as the sum of an unobserved field and a measurement error to give

$$Y(x,t) = S(x,t) + \epsilon(x,t)$$

where $S(x,t)$ is expressed as a p^{th} order AR process

$$S(s,t) = \sum_{i=1}^{p} \phi_i(s) A_i(t).$$

$S(x,t)$ is the unobserved field with spatio-temporal dependence, $\epsilon(x,t)$ is the measurement error which is spatially and temporally uncorrelated, $\phi_i(s)$ is the deterministic basis function and $A_i(t)$ represents the univariate time series.

Stroud et al. (2001) uses a dynamic linear model to define the process

$$\begin{aligned} Y_t &= X_t\beta_t + \epsilon_t, \\ \beta_t &= G_t\beta_{t-1} + \omega_t. \end{aligned}$$

In this paper the authors are examining environmental data, they cast a locally weighted mixture model in a state space framework. This allows for the use of computationally efficient posterior inference for the Gaussian state space model and the inclusion of trends, seasonal effects and auto-regressions. Other authors who have used the DLM to define the spatio-temporal model include Wikle et al. (1997), Tonellato (1998), Wikle & Cressie (1999), Tonellato (2001) and Garside & Wilkinson (2003). The models discussed above represent a selection of methods for analysing spatio-temporal data. This is not an exhaustive list nor are the examples listed the only reference for that technique. Other approaches include the use of Gaussian Markov random fields (Carrol *et al.*, 1997) and Kalman Filtering (Wikle and Cressie, 1999).

The model used in this thesis is a lattice-Markov spatio-temporal model which was considered by some of the above authors. This spatio-temporal model has computational advantages compared to the continuous field models. Many of the models described above use regressive parameters which are not used in the models in this thesis however these regressive parameters can be easily embedded into a larger hierarchical model as mentioned by Waller et al. (1997) and Wikle et al. (1998). The model described in the following chapters examines the main spatio-temporal process ignoring any other possible regressors which may influence the model.

1.3 MCMC in Spatio-Temporal Models

Markov Chain Monte Carlo methods have been used in spatio-temporal modelling by an increasing number of authors with the development of computers over the past ten years. MCMC techniques are used to explore the posterior distribution of the model parameters via stochastic simulation and are particularly useful in situations where the likelihood is not easily defined. Some of the examples in Section 1.2 used MCMC techniques whereas others used maximum likelihood estimation or Kalman filtering approaches.

Hierarchical models similar to those used by Waller et al. (1997) and Wikle et al.

(1998) are popular approaches for using MCMC techniques in spatio-temporal modelling. This approach is used with a variety of different models for the spatio-temporal data. Knorr-Held & Besag (1998) combine models for existing longitudinal and spatial data into a binomial distribution and then use a Metropolis-Hastings algorithm for the simulation of parameters. Cressie & Mugglin (2000) and Mugglin et al. (2002) use a Markov random field representation for the spatio-temporal data and then apply the Metropolis-Hastings algorithm for the simulation of parameter estimates. Genovese (1999) also uses a hierarchical model with Metropolis-Hastings simulation techniques whilst Wikle et al. (1997) uses a hierarchical approach with the Gibbs algorithm for exploring the posterior distribution of the model parameters.

Tonellato (1998) uses a dynamic linear model representation for spatio-temporal wind speed data, this paper examines the same data as Haslett & Raftery (1989). The parameter estimates were then estimated using a Gibbs sampler to allow a fully Bayesian analysis of the model. Carrol et al. (1997) also uses a Gibbs sampler for the MCMC parameter simulations. Sanso & Guenni (1999) use a truncated Normal distribution to represent the rainfall in Venezuela at 80 spatial locations recorded over ten years. They use the covariance structure to model spatial dependence and a data augmentation algorithm for the simulation from the joint posterior distribution of the parameters.

1.4 Structure of the Thesis

This thesis examines the structure of the spatio-temporal data by using the dynamic linear model (DLM) as described in Chapter 2. In this chapter the model is defined along with other basic definitions that are required for future chapters. The model described can suffer from edge effects where there is not a full set of spatial information available at the previous time period. Chapter 3 examines three methods for handling these edge effects: ignoring them by only using the information that is available; scaling the edge weights to take into account the reduction of information; and the use of a conditional representation

that adjusts both the edge weights and the associated precision of the system for these edge nodes. MCMC techniques for the model estimation are introduced in Chapter 4. Three methods of simulation are considered; Gibbs sampling which exploits the use of conjugate priors but is not practical to use in 2+1D models, data augmentation which can suffer from poor mixing in models with a lack of data and finally a Metropolis-Hastings approach. One problem with spatio-temporal models is they tend to be large and thus are computationally time consuming. Chapter 5 examines a method for reducing the size of the model by considering a coarse grid approach as discussed in Higdon et al. (2003). The main property of the method discussed is the parameter estimates from this coarse model are consistent and share interpretation with the fine model developed in Chapter 3. Chapter 6 extends the examples from Chapter 4 by investigating the spatial and temporal dependency parameters. Chapter 7 contains two case studies which use the techniques developed in this thesis to examine two spatio-temporal data sets. The data is ocean temperatures taken from two different regions of the North Atlantic ocean. The results from the fine and coarse models are examined and resulting surface plots produced. This thesis concludes in Chapter 8 with a discussion and presentation of possible future work.

Chapter 2

The Model and Notation

2.1 Introduction

This thesis examines dynamic lattice Markov models where the data is spread through space and time with the latent process being of a spatio-temporal auto-regressive (STAR) form. The model is initially described using observations through time at one spatial location (0+1D model), observations through time at a set of spatial locations in one dimension (1+1D model) and finally a set of spatial observations in two dimensions through time (2+1D model). The dynamic linear model (DLM) is developed for these systems using the standard notation as in West & Harrison (1997) with the conditional distribution for a given point in space at time t being dependent upon the same spatial location and its immediate neighbours at the previous time period.

This chapter introduces the dynamic linear model and describes it in a spatio-temporal setting for the three models above. The underlying latent structure is then developed further for each of these models. The underlying latent structure is important in the building of this model as it describes how the system evolves. Once the structure of this process is understood it is necessary to introduce the notation that allows the process to be neatly described mathematically. The latent structure is a member of the STAR family which is described in Section 2.4. Once the general model has been defined the model

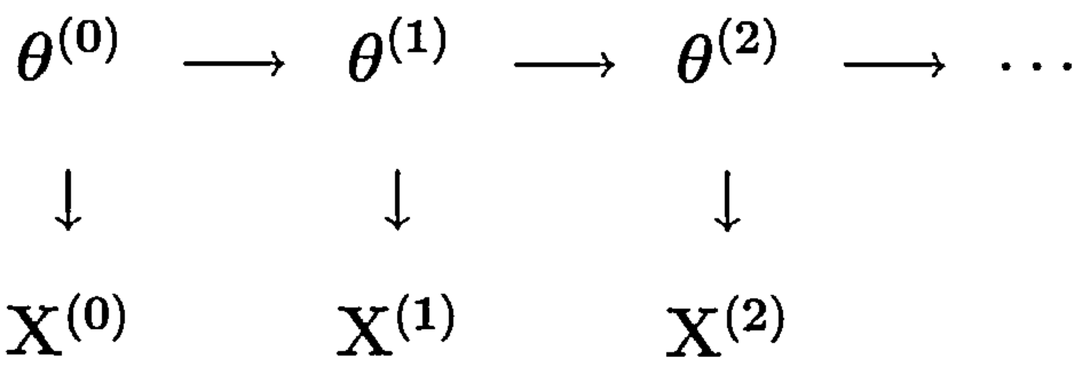


Figure 2.1: *Conditional independence graph for the DLM showing the relationship between the state parameters $\theta^{(t)}$ and observations $X^{(t)}$.*

specific parameters can be developed (the evolution and design matrices) thus entirely defining the DLM for the spatio-temporal model.

2.2 The Dynamic Linear Model

Within the DLM the vector $\mathbf{X^{(t)}}$ represents the observable data points and the vector $\boldsymbol{\theta^{(t)}}$ represents the state of the system (or latent structure) at the given time t. The relationship between the observations and state parameters can be seen in the conditional independence graph shown in Figure 2.1.

Each latent node $\theta^{(t)}$ depends on the latent node directly before it $\theta^{(t-1)}$ and the observations $X^{(t)}$ depend on the latent node $\theta^{(t)}$ i.e. the observations at time t depend on the latent node at time t. This conditional independence graph is represented mathematically as

$$\begin{aligned} \boldsymbol{\theta^{(t)}} &= G_t\boldsymbol{\theta^{(t-1)}} + w_t \quad \text{where} \quad w_t \sim N(0, W_t) \\ \mathbf{X^{(t)}} &= F_t\boldsymbol{\theta^{(t)}} + v_t \quad \text{where} \quad v_t \sim N(0, V_t), \end{aligned}$$

which is equivalent to

$$\begin{aligned} \boldsymbol{\theta^{(t)}} \Big| \boldsymbol{\theta^{(t-1)}} &\sim N\left(G_t\boldsymbol{\theta^{(t-1)}}, W_t\right) \\ \mathbf{X^{(t)}} \Big| \boldsymbol{\theta^{(t)}} &\sim N\left(F_t\boldsymbol{\theta^{(t)}}, V_t\right). \end{aligned}$$

These two equations form the DLM and are referred to as the system equation (or latent structure) and the observation equation. Within the system equation $\boldsymbol{\theta}^{(\mathbf{t})}$ is a vector which has length N, where N is the number of spatial nodes at time t, which describes the underlying latent structure at time t. G_t is the transition (or evolution) matrix which has dimensions $N \times N$, this matrix determines how information is transferred through the latent structure. W_t is a matrix containing the evolution error with dimensions $N \times N$. Within the observation equation $\mathbf{X}^{(\mathbf{t})}$ is a vector of length N containing the observations at time t if there is an observation available at each latent grid point. F_t is the design matrix which has dimensions $N \times N$ for a model with N observations and N latent grid points, it maps the observations to the latent structure while V_t is the matrix containing the observational error with dimension $N \times N$. If there is not a full set of observations available the dimensions of F_t and V_t correspond to the length of the observation vector.

These two equations fully determine the evolution of the DLM. Here the variances, evolution and design matrices are allowed to evolve through time. In this thesis a simplified DLM is examined which is assumed to have constant variance and design matrices through time, the DLM is written as

$$\begin{aligned} \boldsymbol{\theta}^{(\mathbf{t})} \left| \boldsymbol{\theta}^{(\mathbf{t-1})} \right. &\sim N\left(G\boldsymbol{\theta}^{(\mathbf{t-1})}, \tau_s^{-1}\mathrm{I}\right) \\ \mathbf{X}^{(\mathbf{t})} \left| \boldsymbol{\theta}^{(\mathbf{t})} \right. &\sim N\left(F\boldsymbol{\theta}^{(\mathbf{t})}, \tau_o^{-1}\mathrm{I}\right) \end{aligned} \tag{2.1}$$

where τ_s is the precision for the latent structure (state precision), τ_o is the observational precision and I is an identity matrix. In this model setting the innovation variance to be white noise is a pragmatic choice. The spatial dependence in the model is introduced by convolving the process with the transition matrix G. Deviations from this white noise model may introduce identifiability issues as well as making the model computationally more complex. In the models considered in this thesis the two precisions are defined to be constant through space and time. The evolution and design matrix are defined for the spatial structure at each time period. These equations are used to define a number of different spatio-temporal models as elaborated below.

2.2.1 Examples

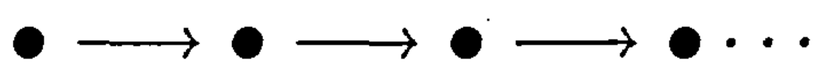
The examples below illustrate the latent structure of three models, the purely temporal case and then two spatio-temporal cases which look at models with one and two spatial dimensions evolving through time. In the following examples it is assumed that there is a full set of observations available and the design matrix F is defined as the identity matrix with dimensions $N \times N$. This design matrix maps the observations to the latent structure.

Time series model (0+1D model)

Consider an AR(1) model with noise,

$$\theta^{(t)} = a\theta^{(t-1)} + \epsilon_t, \qquad \epsilon_t \sim N\left(0, \sigma^2\right)$$

where a is a constant and ϵ_t represents white noise. The latent structure is a single spatial node for each time period where a single spatial node at time t is influenced by the spatial node at time $t-1$.



Time 0 Time 1 Time 2 Time 3 ...

Figure 2.2: *Evolution through time of the latent process in the 0+1D model.*

Figure 2.2 shows how information is passed through this system. Using the DLM this is depicted by Figure 2.1 where the superscript denotes the time period that the node is in. The design matrix here is a 1×1 identity matrix as there is only one spatial node in the system, similarly the evolution matrix is a 1×1 matrix.

Space time model (i) (1+1D model)

Extending the above model to a vector of latent nodes in space evolving through time yields the 1+1D model.

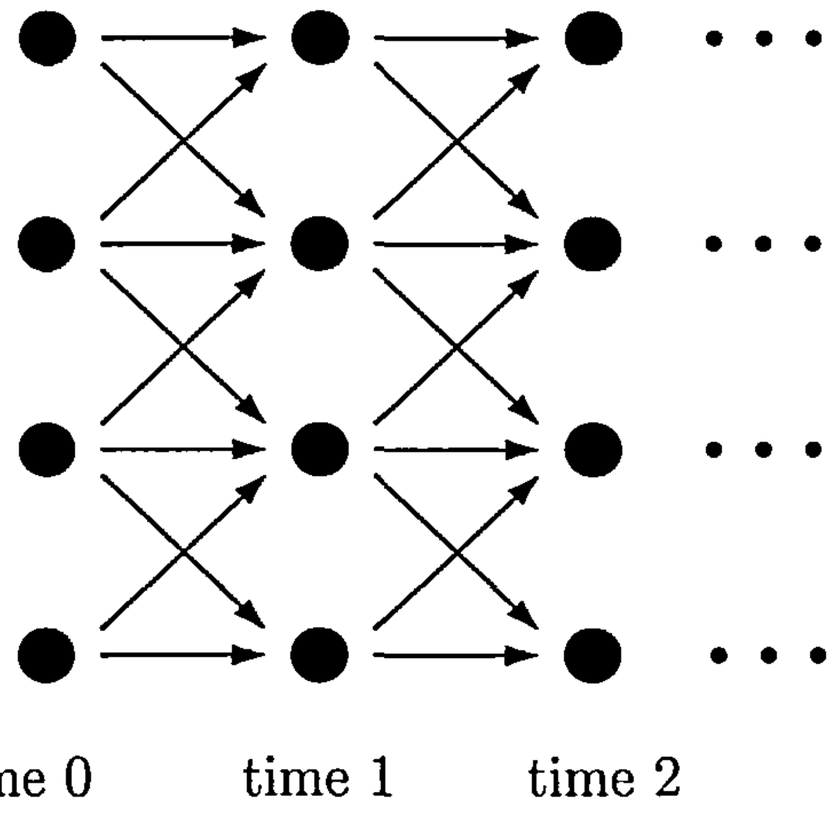


Figure 2.3: *Evolution through time of the latent process in the 1+1D model.*

Figure 2.3 shows a basic way in which information can be passed through this 1+1D system. The labelling for the DLM is now $\theta_i^{(t)}$ and $X_i^{(t)}$ where t represents the time period and i represents the spatial location. Both $\mathbf{X^{(t)}}$ and $\boldsymbol{\theta}^{(\mathbf{t})}$ are ordered vectors of length N where N is the number of spatial nodes. The design matrix F is a $N \times N$ identity matrix, similarly the evolution matrix G is also an $N \times N$ matrix.

Space time model (ii) (2+1D model)

Extending the above model further develops the latent structure with nodes in two spatial dimensions evolving through time. Figure 2.4 shows an example of how this three

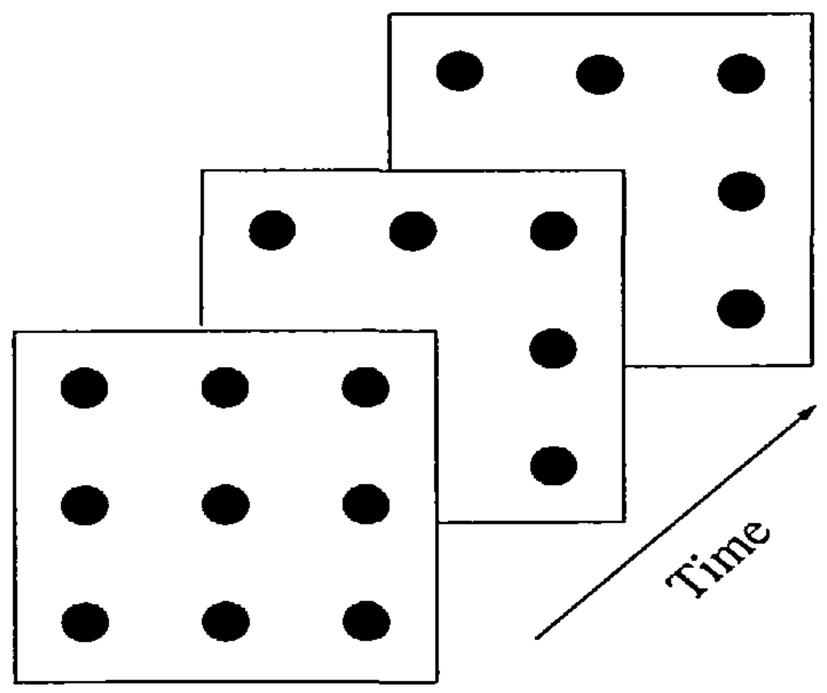


Figure 2.4: *Evolution through time of the latent process in the 2+1D model.*

dimensional problem can be depicted. Each time period has a spatial latent structure which is defined to be a square or rectangular lattice. The way in which information is transferred through the system is not shown here for simplicity, this is examined in more detail in section 2.6.3. The labelling for the DLM is now $\theta_{i,j}^{(t)}$ and $X_{i,j}^{(t)}$ where t represents the time period and i and j are the spatial coordinates of the latent node or observation. Both $\mathbf{X^{(t)}}$ and $\boldsymbol{\theta}^{\mathbf{(t)}}$ are ordered vectors of length N, where $N = nm$ and n is the number of spatial nodes in the horizontal direction whilst m is the number of spatial nodes in the vertical direction. The latent vector at time t is given by $\boldsymbol{\theta}^{\mathbf{(t)}} = \left(\theta_{0,0}^{(t)}, \theta_{0,1}^{(t)}, \ldots, \theta_{0,m-1}^{(t)}, \theta_{1,0}^{(t)}, \ldots, \theta_{n-1,m-1}^{(t)}\right)$, the observation vector at time t is similarly defined. The design matrix F is a $N \times N$ identity matrix, similarly the evolution matrix G is also an $N \times N$ matrix.

2.3 Notation

The notation for the latent system $\left(\boldsymbol{\theta}^{\mathbf{(t)}}\right)$ and observations $\left(\mathbf{X^{(t)}}\right)$ has been described in Section 2.2. This notation is now extended to fully describe the evolution of the system.

The location of an observation X on the lattice is given by $X_{i,j}^{(t)}$ where t denotes the time period, (i, j) represents the spatial coordinate location on the two dimensional spatial lattice (in the 1+1D model the spatial location is given solely by i) and the vector of observations at time t is denoted by $\mathbf{X^{(t)}}$. The latent structure is defined similarly with $\theta_{i,j}^{(t)}$ representing the latent node at time period t, the spatial location given by the coordinate (i, j) and the vector of latent nodes at time t is given by $\boldsymbol{\theta}^{\mathbf{(t)}}$. The evolution of a latent node is determined by the surrounding nodes at the previous time period.

In a purely spatial setting these surrounding nodes are defined to be neighbours if the conditional distribution of θ_i depends on them (Besag, 1974). These neighbours are then defined to be zero, one or two step (order) neighbours. In a spatio-temporal setting the zero step neighbour of node $\theta_i^{(t+1)}$ is defined to be the node $\theta_i^{(t)}$ i.e. the node itself at the previous time point, the one step neighbour is a direct neighbour to

node $\theta_i^{(t)}$ and a two step neighbour is a diagonal neighbour in models with two spatial dimensions as defined in Pfeifer & Deutsch (1980*d*). Cressie & Mugglin (2000) examine non lattice spatio-temporal data and define the first order neighbour if the sites share a common boundary and a second order neighbour to be a neighbour of the first order neighbour. These neighbours are then grouped into neighbourhoods, Pfeifer & Deutsch (1980*d*) define a neighbourhood of order l to be the set of all neighbours which are of order l. Whereas an extension of the purely spatial neighbourhoods defined by Besag (1974) to a spatio-temporal setting yields a first order neighbourhood defined as the set of zero and one step neighbours to a node (see Royle et al. (1998) for a spatial example and Bailey et al. (1997) and Wikle et al. (1998) for a spatio-temporal model) and a second order neighbourhood is defined as an extension of this first order neighbourhood by including the second order neighbours in the 2+1D model as in Garside & Wilkinson (2003). The extension of the spatial definition of neighbours to the spatio-temporal setting although possible is not ideal. In a purely spatial setting the neighbours of a given node are the same in both a directed and undirected graph whereas in the spatio-temporal setting the neighbours differ between the directed and undirected graphs. Figure 2.5 (a) shows that node P has one zero order and two first order neighbours from the previous time period whereas Figure 2.5 (b) shows node P to have eight neighbours from the previous, current and next time period comprising of four first order neighbours and four second order neighbours. This undirected graph shows the conditional independence graph which is used in Chapter 4. Instead of using the spatio-temporal extension of neighbours to describe the evolution of the system through time each node has a set of parents defined which allows the information to be passed through the system.

Definition: Parents

The parents of node $\theta_i^{(t)}$ *are denoted by* $\text{pa}\left(\theta_i^{(t)}\right)$ *and represent the set of nodes which the distribution of* $\theta_i^{(t)}$ *is conditional upon.*

As with the spatio-temporal extension of the neighbour structure define the parent

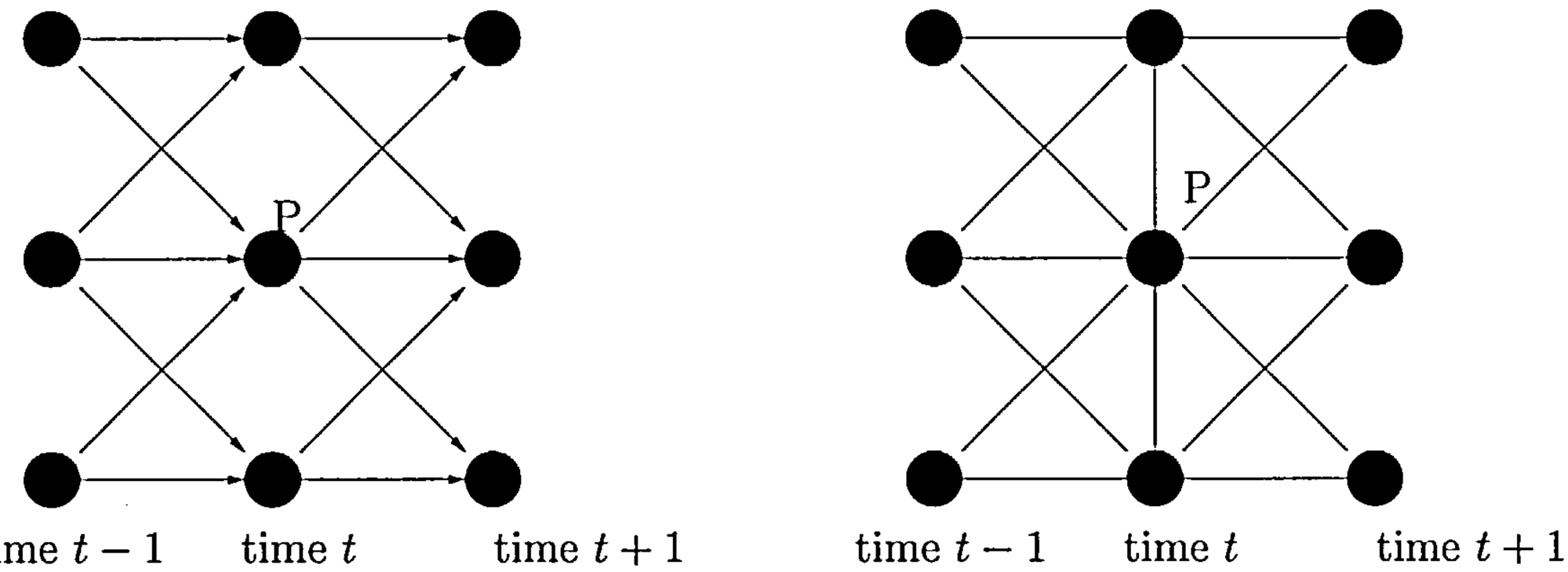


Figure 2.5: *The parent structure for the 1+1D spatio-temporal model far (a) a directed graph (b) an undirected graph.*

nodes to be located zero, one or two steps from the node of interest. The zero step parent of node $\theta_i^{(t+1)}$ is defined to be the node $\theta_i^{(t)}$ i.e. the node itself at the previous time point, the one step parent is a direct neighbour to node $\theta_i^{(t)}$ at the previous time period and a two step parent is a diagonal neighbour to the node of interest at the previous time period. Further model specific notation is introduced in later chapters as it is required.

2.4 STAR Models

The STAR model is a member of the family of STARMA (spatio-temporal auto-regressive moving average) models. The latent process of the spatio-temporal models examined in this thesis are of STAR form Cressie (1991) and Pfeifer & Deutsch (1980*d*) observed irregularly and with error.

2.4.1 STARMA models

This family of models uses a hierarchical spatial ordering of neighbours of each site and a sequence of weighting matrices. Using the notation from Pfeifer & Deutsch (1980*d*) the

STARMA$\left(p_{\lambda_1,\ldots\lambda_p}, q_{m_1,\ldots m_q}\right)$ model takes the form

$$\begin{aligned}\mathbf{z}(t) &= \sum_{k=1}^{p}\sum_{l=0}^{\lambda_k}\phi_{kl}W^{(l)}\mathbf{z}(t-k) \\ &\quad -\sum_{k=1}^{q}\sum_{l=0}^{m_k}\theta_{kl}W^{(l)}\epsilon(t-k)+\epsilon(t) \qquad (2.2)\end{aligned}$$

where $\mathbf{z}(t)$ is the $N \times 1$ vector of observations at the N spatial locations and time t, $\epsilon(t)$ are i.i.d. zero mean random variables with variance $\sigma^2 I$, p is the auto-regressive order, q is the moving average order, λ_k is the spatial order of the k^{th} auto-regressive term, m_k is the spatial order of the k^{th} moving average term and ϕ_{kl} and θ_{kl} are parameters. $W^{(l)}$ is the weight matrix with elements $w_{ij}^{(l)}$ that are non zero if and only if sites i and j are l^{th} order neighbours, $W^{(0)}$ is the identity matrix. The weights must reflect the hierarchical ordering of the spatial neighbours, i.e. first order neighbours are "closer" than second order neighbours and thus should have a larger weight $w_{i,j}^{(l)}$.

When $q = 0$ the model is a STAR model. The latent structure of the DLM $\theta^{(t)} = G\theta^{(t-1)} + \tau_s^{-1}$ can be written as a STAR(1_1) model with $k = 1$, $\phi_l = \alpha/2$ and $G = W^{(0)} + W^{(1)}$ if a first order neighbourhood structure is being considered for the model, i.e.

$$\mathbf{z}(t) = \phi_{10}W^{(0)}\mathbf{z}(t-1) + \phi_{11}W^{(1)}\mathbf{z}(t-1) + \epsilon(t)\,. \qquad (2.3)$$

The stationarity of STAR models is examined in Pfeifer & Deutsch (1980*c*), the STAR(1_1) model is stationary if $|\phi_{10}| + |\phi_{11}| < 1$. In this DLM model $|\phi_{10}| + |\phi_{11}| = \alpha$ where $\alpha \in [0, 1)$ thus the stationarity constraint is met and the DLM is a stationary STAR(1_1) model. Further work by Pfeifer & Deutsch (1980*b*) examines the residuals from STAR models and Pfeifer & Deutsch (1980*a*) examines how to identify and interpret first order STARMA models.

2.5 The observation equation

All of the examples used previously in this chapter have been described for models with a complete set of data where the design matrix is an $N \times N$ identity matrix. In situations

where there is not a full set of data available the design matrix F is a $N_o \times N$ matrix at each time period where N_o is the number of observations and N is the number of latent points with elements f_{ij} non zero if and only if there exists an observation at spatial location (i, j) in that time period. In situations where the data does not naturally match to the grid structure of the underlying latent process each observation is mapped to the nearest grid location. Problems that can occur with multiple observations being assigned to the same grid point are examined in Chapter 5. The design matrix F provides a mapping of the data to the underlying latent structure.

2.6 Problem specific declarations

Before the system equation can be written for the 0+1D, 1+1D and 2+1D models the evolution matrix needs to be determined. In order to define the evolution matrix the way in which information is passed through the latent structure must be determined. This is achieved by building the latent structure along a lattice grid which evolves through space and time using information from parent nodes. The evolution through space and time is developed as an extension to the purely spatial models as examined by Besag (1974) and Bartlett (1978) and the space time models of Pfeifer & Deutsch (1980*d*).

2.6.1 The Latent structure

The latent structure is developed on a lattice grid where the latent points occur at each intersection of the grid. Two models are developed further in this thesis, the 1+1D model and the 2+1D model, in both cases the conditional distribution of $\theta_i^{(t)} \Big| \text{pa}\left(\theta_i^{(t)}\right)$ describes the evolution of the latent structure.

1+1D model

In the 1+1D model the parents $\text{pa}\left(\theta_i^{(t)}\right)$ of node $\theta_i^{(t)}$ are $\theta_{i-1}^{(t-1)}$, $\theta_i^{(t-1)}$ and $\theta_{i+1}^{(t-1)}$. The amount of weight to be given to each of these parent nodes is discussed in Section 2.6.2

2+1D models

In the 2+1D model the parents $pa\left(\theta_{i,j}^{(t)}\right)$ of node $\theta_{i,j}^{(t)}$ are defined as the set of five nodes, the spatial node itself and its four direct parent nodes at the previous time $t-1$

$$\text{pa}\left(\theta_{i,j}^{(t)}\right) = \left(\theta_{i-1,j}^{(t-1)}, \theta_{i,j-1}^{(t-1)}, \theta_{i,j}^{(t-1)}, \theta_{i,j+1}^{(t-1)}, \theta_{i+1,j}^{(t-1)}\right).$$

This model is not ideal as it suffers from lattice coordinate system artifacts. A more obvious system is to use the second order parent nodes as well i.e. this model also includes information from the diagonal parent nodes, thus $\theta_{i,j}^{(t)}$ is conditioned on it parents.

$$\text{pa}\left(\theta_{i,j}^{(t)}\right) = \left(\theta_{i-1,j-1}^{(t-1)}, \theta_{i-1,j}^{(t-1)}, \theta_{i-1,j+1}^{(t-1)}, \theta_{i,j-1}^{(t-1)}, \theta_{i,j}^{(t-1)}, \theta_{i,j+1}^{(t-1)}, \theta_{i+1,j-1}^{(t-1)}, \theta_{i+1,j}^{(t-1)}, \theta_{i+1,j+1}^{(t-1)}, \right).$$

Now the parents for node $\theta_{i,j}^{(t)}$ have been defined the evolution matrix G can be determined for these models together with some i.i.d. noise.

The evolution of the latent structure is determined by the value of the latent structure at the previous time point. The latent structure is initialised at time $t = 0$ as a grid of independent Normal values

$$\theta^{(0)} \sim N\left(\mu, \tau_s^{-1}\right)$$

where μ is the mean of the system and τ_s is the state precision. For $t > 0$ $\theta_{i,j}^{(t)}$ is defined by a linear combination of its parents. The latent structure is allowed to run to equilibrium before including the observations and examining the space-time structures of the model, thus the initial choice of τ_s^{-1} as the precision of the latent structure at time $t = 0$ is unimportant.

2.6.2 Building the evolution weight matrix: 1+1D model

The introduction of a *hidden layer* to the model at time periods $t - \frac{1}{2}$ which is perfectly offset from the lattice at times $t-1$ and t (see Figure 2.6) allows information from a latent point at time $t-1$ to be passed diagonally up or down to a node in the hidden layer and then passed diagonally up or down to the next time point. The introduction of this hidden layer provides a tool which simplifies the model building process and motivates

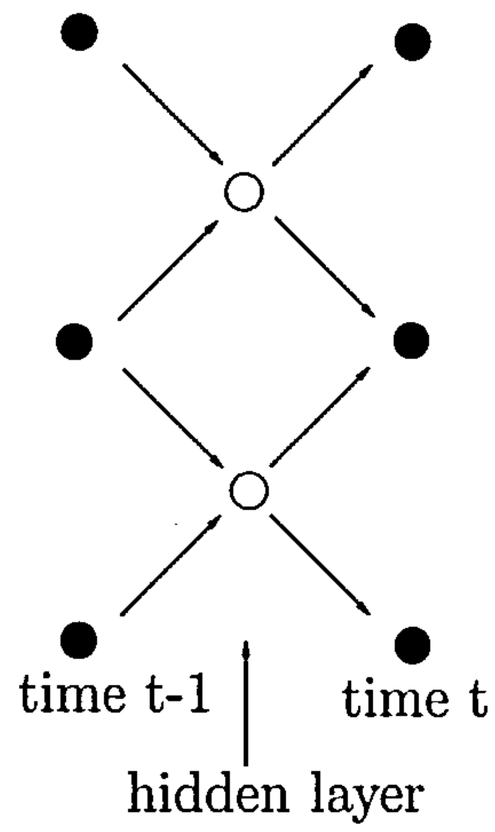


Figure 2.6: *The graphical representation of the hidden layer in the 1+1D model.*

the development of the binomial weight model. Let the proportion of information moving up be represented by p_1 and the proportion of information moving down be represented by p_2 using these proportions the value of a given latent node can be expressed as

$$\begin{aligned} \theta_i^{(t)} \left| \text{pa}\left(\theta_i^{(t)}\right) \right. &= N\left(G\theta_i^{(t-1)}, \tau_s^{-1}\right) \\ &= N\left(\alpha\left(p_1^2\theta_{i-1}^{(t-1)} + 2p_1p_2\theta_i^{(t-1)} + p_2^2\theta_{i+1}^{(t-1)}\right), \tau_s^{-1}\right) \end{aligned} \tag{2.4}$$

where $\sum_i p_i = 1$. This model has three parameters; one free $p_{i,j}$, α and τ_s. The free $p_{i,j}$ controls the degree of anisotropy, α, an auto-regressive parameter, controls the temporal dependence and τ_s controls the degree of innovation noise. This model has three free parameters, one free p_i, α and τ_s. Where p_i controls the degree of anisotropy, α is the auto-regressive parameter which controls the temporal dependence within the system and τ_s is the state precision which controls the degree of innovation noise.

Thus for a single node the weight matrix g is given by these proportions is

$$g = \alpha \begin{pmatrix} p_1^2 \\ 2p_1p_2 \\ p_2^2 \end{pmatrix} \tag{2.5}$$

The evolution matrix G is a $N \times N$ matrix where N is the total number of spatial nodes in the system. Define g_i to be the i^{th} element of g, using this definition the evolution

matrix G is given by

$$G = \begin{pmatrix} g_2 & g_3 & 0 & 0 & 0 & \cdots \\ g_1 & g_2 & g_3 & 0 & 0 & \cdots \\ 0 & g_1 & g_2 & g_3 & 0 & \cdots \\ 0 & 0 & g_1 & g_2 & g_3 & \cdots \\ \vdots & \vdots & & & \ddots & \\ 0 & & \cdots & & g_1 & g_2 \end{pmatrix} \tag{2.6}$$

If $p_1 = p_2 = \frac{1}{2}$ then the weight matrix g is

$$g = \alpha \begin{pmatrix} \frac{1}{4} \\ \frac{1}{2} \\ \frac{1}{4} \end{pmatrix}$$

This model is referred to as the isotropic binomial weight model. Further details of the conditional distribution of node $\theta_i^{(t)}$ are given in Chapter 3 where the evolution of nodes along the edge of the system is examined.

2.6.3 Building the evolution weight matrix: 2+1D model

As in the 1+1D model a *hidden layer* is introduced to the 2+1D model at time period $t - \frac{1}{2}$ (see Figure 2.8) to allow information to be passed between the time periods. This hidden layer is the same shape as the lattice for the model and is perfectly offset from the lattice at times $t - 1$ and t. Imposing a first order (four parent) spatial model with edge weights $p_{i,j}$ (defined as in Figure 2.7) to this system and then marginalising out the hidden layer leads to the second order (nine parent) spatio-temporal model given below. The difference between the 1+1D and 2+1D model is that information is now passed in one of four directions as detailed in Figure 2.7.

Using these proportions the distribution of a latent node conditional upon its parents

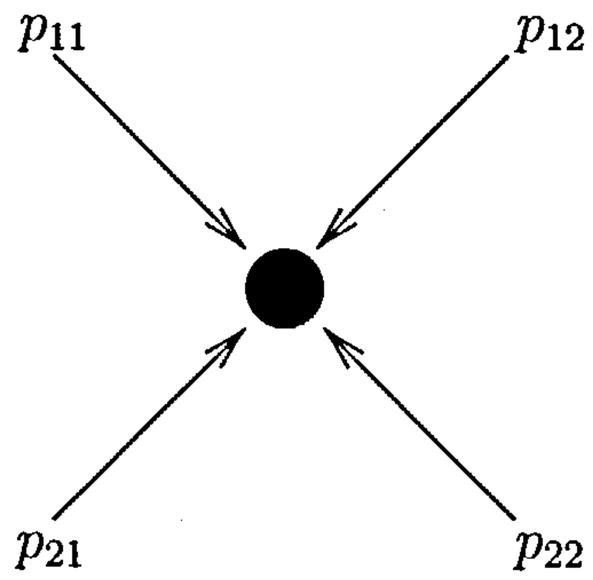


Figure 2.7: *The four edge weights p_{11}, p_{12}, p_{21} and p_{22} for the hidden layer in the 2+1D model.*

is given by

$$\begin{aligned}\theta_{i,j}^{(t)}\left|\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\right. &= N\left(\alpha\left(p_{11}^2\theta_{i-1,j-1}^{(t-1)}+2p_{11}p_{12}\theta_{i-1,j}^{(t-1)}+p_{12}^2\theta_{i-1,j+1}^{(t-1)}\right.\right.\\ &\quad +2p_{11}p_{12}\theta_{i,j-1}^{(t-1)}+\left(2p_{11}p_{22}+2p_{21}p_{12}\right)\theta_{i,j}^{(t-1)}+2p_{12}p_{22}\theta_{i,j+1}^{(t-1)}\\ &\quad \left.\left.+p_{21}^2\theta_{i+1,j-1}^{(t-1)}+2p_{21}p_{22}\theta_{i+1,j}^{(t-1)}+p_{22}^2\theta_{i+1,j+1}^{(t-1)}\right),\tau_s^{-1}\right),\end{aligned} \tag{2.7}$$

where $\Sigma_{i,j}p_{i,j}=1$. This model has five parameters; three free $p_{i,j}$, α and τ_s. The three free $p_{i,j}$ control the degree of anisotropy, α, an auto-regressive parameter, controls the temporal dependence and τ_s controls the degree of innovation noise. Thus for a single node the weight matrix g is given by

$$g = \alpha\begin{pmatrix} p_{11}^2 & 2p_{11}p_{12} & p_{12}^2 \\ 2p_{11}p_{21} & 2p_{11}p_{22}+2p_{21}p_{12} & 2p_{12}p_{22} \\ p_{21}^2 & 2p_{21}p_{22} & p_{22}^2 \end{pmatrix}. \tag{2.8}$$

The evolution matrix G is a $N\times N$ matrix where $N=nm$; the total number of spatial nodes in the system. The evolution matrix G is written in terms of the elements of g where g_{ij} is the $(i,j)^{th}$ element of g. The evolution matrix is partitioned into four smaller matrices,

$$G = \left(\begin{array}{c|c|c|c|c} A & B & \underline{0} & \underline{0} & \cdots \\ \hline C & A & B & \underline{0} & \\ \hline \underline{0} & C & A & B & \\ \hline \vdots & & & & \ddots \end{array}\right) \tag{2.9}$$

where $\underline{0}$ is the zero matrix and A B, and C are matrices defined by

$$A = \begin{pmatrix} g_{22} & g_{23} & 0 & 0 & \dots & & \\ g_{12} & g_{22} & g_{23} & 0 & \dots & & \\ 0 & g_{12} & g_{22} & g_{23} & \dots & & \\ 0 & 0 & g_{12} & g_{22} & \dots & & \\ \vdots & \vdots & \vdots & \vdots & & & \\ 0 & & \dots & & & g_{12} & g_{22} \end{pmatrix}, \tag{2.10}$$

$$B = \begin{pmatrix} g_{32} & g_{33} & 0 & 0 & \dots & & \\ g_{31} & g_{32} & g_{33} & 0 & \dots & & \\ 0 & g_{31} & g_{32} & g_{33} & \dots & & \\ 0 & 0 & g_{31} & g_{32} & \dots & & \\ \vdots & \vdots & \vdots & \vdots & & & \\ 0 & & \dots & & & g_{31} & g_{32} \end{pmatrix}, \tag{2.11}$$

$$C = \begin{pmatrix} g_{12} & g_{13} & 0 & 0 & \dots & & \\ g_{11} & g_{12} & g_{13} & 0 & \dots & & \\ 0 & g_{11} & g_{12} & g_{13} & \dots & & \\ 0 & 0 & g_{11} & g_{12} & \dots & & \\ \vdots & \vdots & \vdots & \vdots & & & \\ 0 & & \dots & & & g_{11} & g_{12} \end{pmatrix}. \tag{2.12}$$

If $p_{11} = p_{12} = p_{21} = p_{22} = \frac{1}{4}$ then the weight matrix g is given by

$$g = \alpha \begin{pmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{pmatrix}.$$

This model is referred to as the isotropic binomial weight model. Further details of the conditional distribution of node $\theta_i^{(t)}$ are given in Chapter 3 where the evolution of nodes along the edge of the system is examined.

The latent structure for both the 1+1D and 2+1D model is now fully defined. The weight matrix g has been given in the general form and in the isotropic binomial weight form for both models. Writing the latent structure for the 1+1D model with binomial weights in the form of Equation 2.3 yields

$$\boldsymbol{\theta^{(t)}} = \phi_{01}\mathbf{W^{(0)}}\boldsymbol{\theta^{(t-1)}} + \phi_{11}\mathbf{W^{(1)}}\boldsymbol{\theta^{(t-1)}} + \boldsymbol{\epsilon}(t)$$

where $\phi_{01} = \frac{\alpha}{2}$, $\phi_{11} = \frac{\alpha}{2}$, $W^{(0)}$ is the identity matrix, $W^{(1)}$ has elements $w_{ij}^{(1)} = \frac{1}{2}$ if site (i,j) is a first order neighbour of $\theta_{k,l}^{(t)}$ and $\epsilon(t) \sim N(0, \tau_s^{-1})$. Thus the properties of stationarity and invertibility of the STARMA models apply to the latent structure of this DLM.

2.7 Discussion

This chapter has defined the dynamic linear equation and the notation used in the analysis of spatio-temporal models. Early work on spatio-temporal modelling led to the development of STARMA models. The latent structure of the DLM can be expressed in STAR form. The evolution matrix G has been developed through the introduction of a *hidden layer* which is then used in defining the conditional distribution $\theta_i^{(t)} \left| \text{pa}\left(\theta_i^{(t)}\right) \right.$ which allows information to be transferred through the system.

In the models discussed in the chapter α is the parameter controlling the temporal dependence in the model but there is not a parameter determining the spatial dependence. The spatial dependence parameter β is introduced and investigated in Chapter 6.

These ideas form the basis for the work discussed in future chapters which examines the techniques used to describe the evolution of nodes which lie along the edge of the system, parameter simulation and the development of more time efficient methods such as the use of parallel computing clusters and coarsening the latent model.

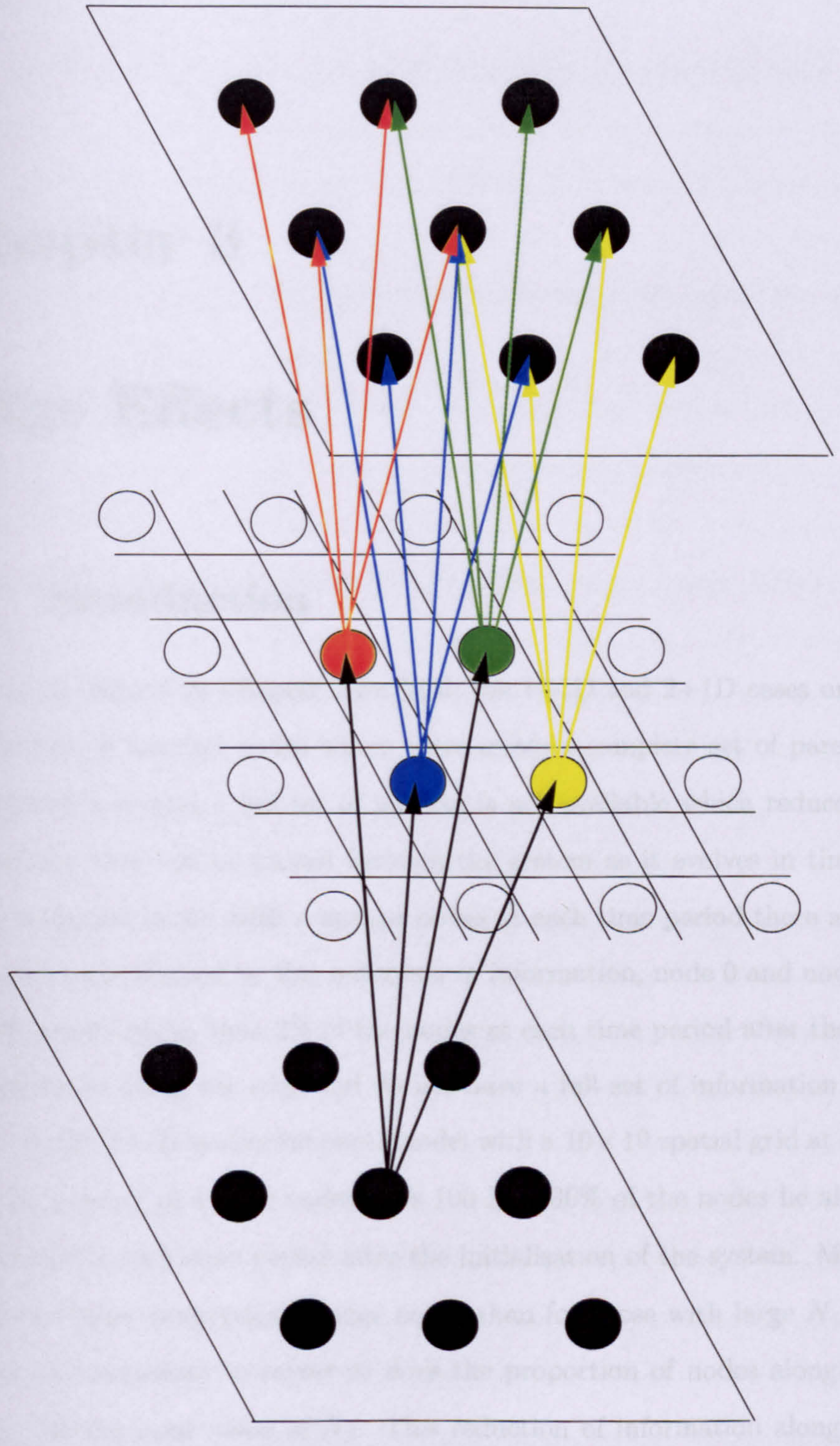


Figure 2.8: *The graphical representation of the hidden layer in the 2+1D model for a second order parent system.*

Chapter 3

Edge Effects

3.1 Introduction

The model defined in Chapter 2 for both the 1+1D and 2+1D cases only relates to the distribution of internal nodes where there exists a complete set of parent nodes. Along the edge of a system a full set of parents is not available which reduces the amount of information that can be passed through the system as it evolves in time. In the 1+1D spatio-temporal model with n spatial nodes at each time period there are only two edge sites which are affected by this reduction in information, node 0 and node $n-1$. If there are 100 spatial nodes then 2% of the nodes at each time period after the initialisation of the system lie along the edge and do not have a full set of information from the parent nodes. In the 2+1D spatio-temporal model with a 10×10 spatial grid at each time period the total number of spatial nodes N is 100 here 36% of the nodes lie along the edges of the system for each time period after the initialisation of the system. Models with small N have a higher proportion of edge nodes than for those with large N, similarly as the number of dimensions increases so does the proportion of nodes along the edge of the system (for the same value of N). This reduction of information along the edge of the system is referred to as an edge effect. Within spatio-temporal models these edge effects are an important factor in the design of models due to the large proportion of nodes which

lie along the edge of these systems.

There are several approaches to dealing with edge effects on finite lattices which are discussed in the literature including in Cressie (1991), Lawson et al. (1999), Besag & Kooperberg (1995) and Berliner et al. (1999). The methods discussed are generally explained in relation to purely spatial models but they can be extended to the dynamic spatio-temporal context. The main methods discussed are: to set the edge weights corresponding to missing edge nodes to zero i.e. ignore the missing nodes as in Cressie (1991) (referred to as the unadjusted model), to set the edge weights corresponding to missing nodes to zero and then scale the remaining edge weights appropriately as in Lawson et al. (1999) and Berliner et al. (1999) (referred to as the scaled model), to consider the conditional distribution for the edge sites as mentioned in Cressie (1991) (referred to as the adjusted model), to use a guard region i.e. embedding the model of interest into a larger grid as in Cressie (1991) and Lawson et al. (1999) or to define a new approach by expanding the spatial neighbourhood structure for edge nodes as defined in Besag & Kooperberg (1995). Embedding the model into a larger region is computationally infeasible for spatio-temporal models as the guard region significantly increases the dimensions of the model under consideration and thus increases the computational cost considerably. Altering the neighbourhood structure along the edge of the system as in Besag & Kooperberg (1995) is not easily extended to the spatio-temporal context and thus is not considered in this thesis. The first three methods listed above are examined in detail in this thesis.

Consider a system which is approximately stationary in time, Figure 3.1 shows the precision surfaces for a simulated 2+1D model at $T = 20$ on a 10×10 spatial grid using a nine parent model with four different values for the temporal dependence α. Along the edge of the system the edge weights corresponding to missing parents were set to zero, this system is not stationary in space and the precision of the edges and corners is significantly larger than the internal spatial points. As $\alpha \rightarrow 1$ this difference in precision becomes larger, the range of precision for the four cases is shown in Table 3.1. The size of this range increases from 0.0349894 for $\alpha = 0.7$ to 0.1946258 for $\alpha = 0.99$ with the

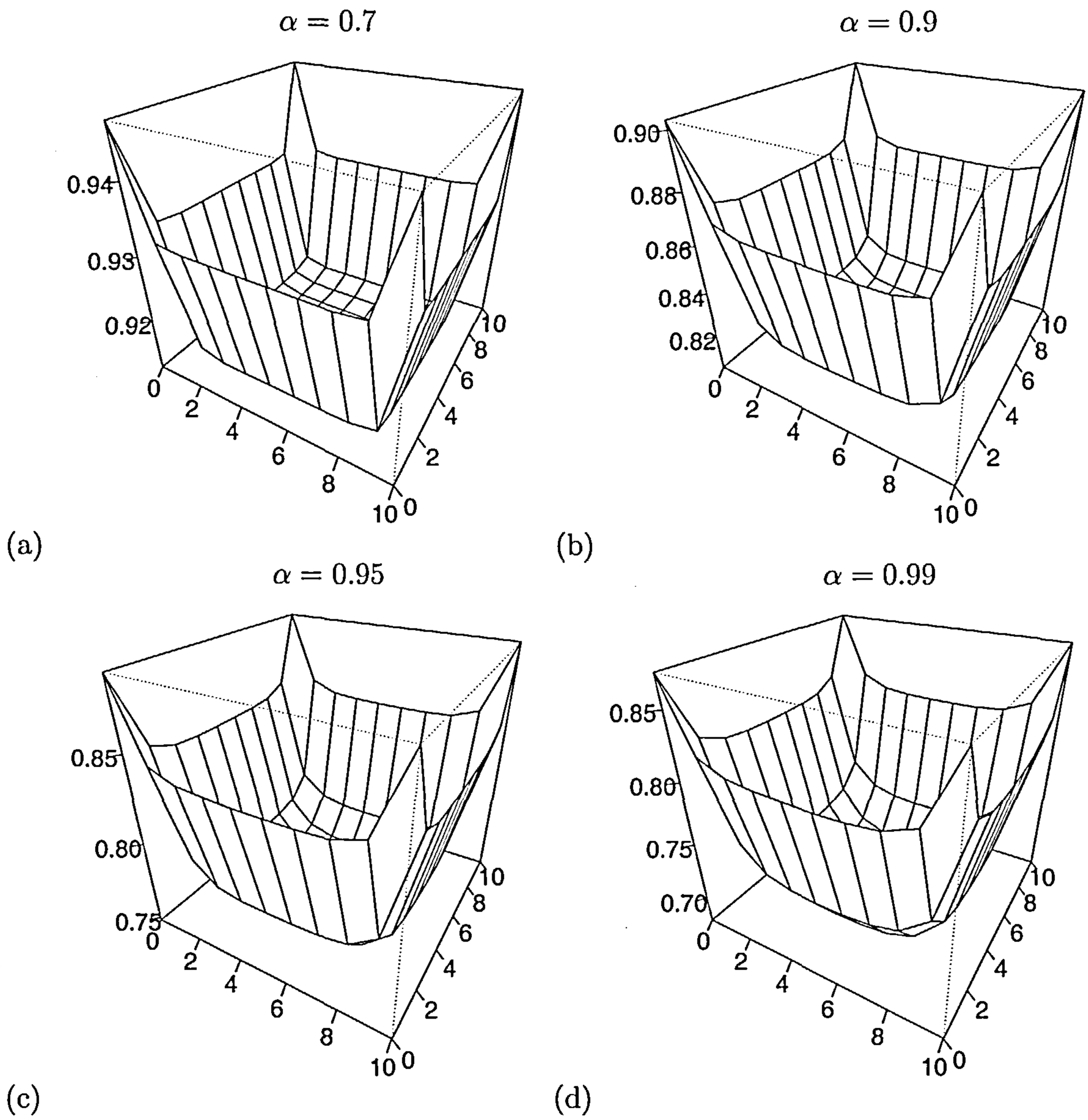


Figure 3.1: *Precision of the system on a* 10×10 *spatial grid at time* $T = 20$ *with unadjusted edge weights for (a)* $\alpha = 0.7$*, (b)* $\alpha = 0.9$*, (c)* $\alpha = 0.95$*, (d)* $\alpha = 0.99$.

precision being greater than the stationary precision. Figure 3.1 shows that the precision is significantly larger along the edges and at the corners of the system, this is due to the reduction of information being passed through the system along the edges, thus a method which adjusts for these edge effects is required. The model under consideration is $\theta^{(t)}|\theta^{(t-1)} \sim N\left(G\theta^{(t-1)}, \tau_s^{-1} I\right)$, where the state precision τ_s is constant in space and time and G is the matrix of edge weights, it is this matrix which needs adjusting along

α	Stationary Precision	Range	Size of range
0.7	0.9119323	0.9119337 : 0.9469231	0.0349894
0.9	0.8039998	0.8045777 : 0.9030634	0.0990563
0.95	0.7429752	0.7482521 : 0.8876131	0.1393610
0.99	0.627286	0.6781992 : 0.8728250	0.1946258

Table 3.1: *Stationary precision and the associated range for a 10 × 10 spatial model at time $T = 20$ with unadjusted edge weights.*

α	Stationary Precision	Range	Size of range
0.7	0.9119323	0.8166405 : 0.9119270	0.0952865
0.9	0.8039998	0.6202312 : 0.8036434	0.1834121
0.95	0.7429752	0.5295794 : 0.7446787	0.2150993
0.99	0.627286	0.4302563 : 0.6683411	0.2380848

Table 3.2: *Stationary Precision and the associated range for 10 × 10 spatial model at time $T = 20$ with scaled edge weights.*

the edges of the system to allow for the missing parent nodes.

Table 3.1 gives the theoretical stationary precision for the four values of temporal dependence along with the range of values the state precision takes in the simulated system for the unadjusted model, the final column in the table gives the size of this range. These results are shown graphically in Figure 3.1. Here the edge weights for the corner node $\theta_{0,0}^{(t)}$ are given by

$$\alpha\left(\frac{1}{16}\theta_{1,1}^{(t-1)}+\frac{1}{8}\left(\theta_{0,1}^{(t-1)}+\theta_{1,0}^{(t-1)}\right)+\frac{1}{4}\theta_{0,0}^{(t-1)}\right).$$

Similarly for the edge node $\theta_{0,3}^{(t)}$ the edge weights are given by

$$\alpha\left(\frac{1}{16}\left(\theta_{1,2}^{(t-1)}+\theta_{1,4}^{(t-1)}\right)+\frac{1}{8}\left(\theta_{0,2}^{(t-1)}+\theta_{0,4}^{(t-1)}+\theta_{1,3}^{(t-1)}\right)+\frac{1}{4}\theta_{0,3}^{(t-1)}\right).$$

For the internal nodes all nine parents are used in the calculation of the edge weights, for $\theta_{i,j}^{(t)}$ where (i,j) are the coordinates of an internal node the edge weights are given by

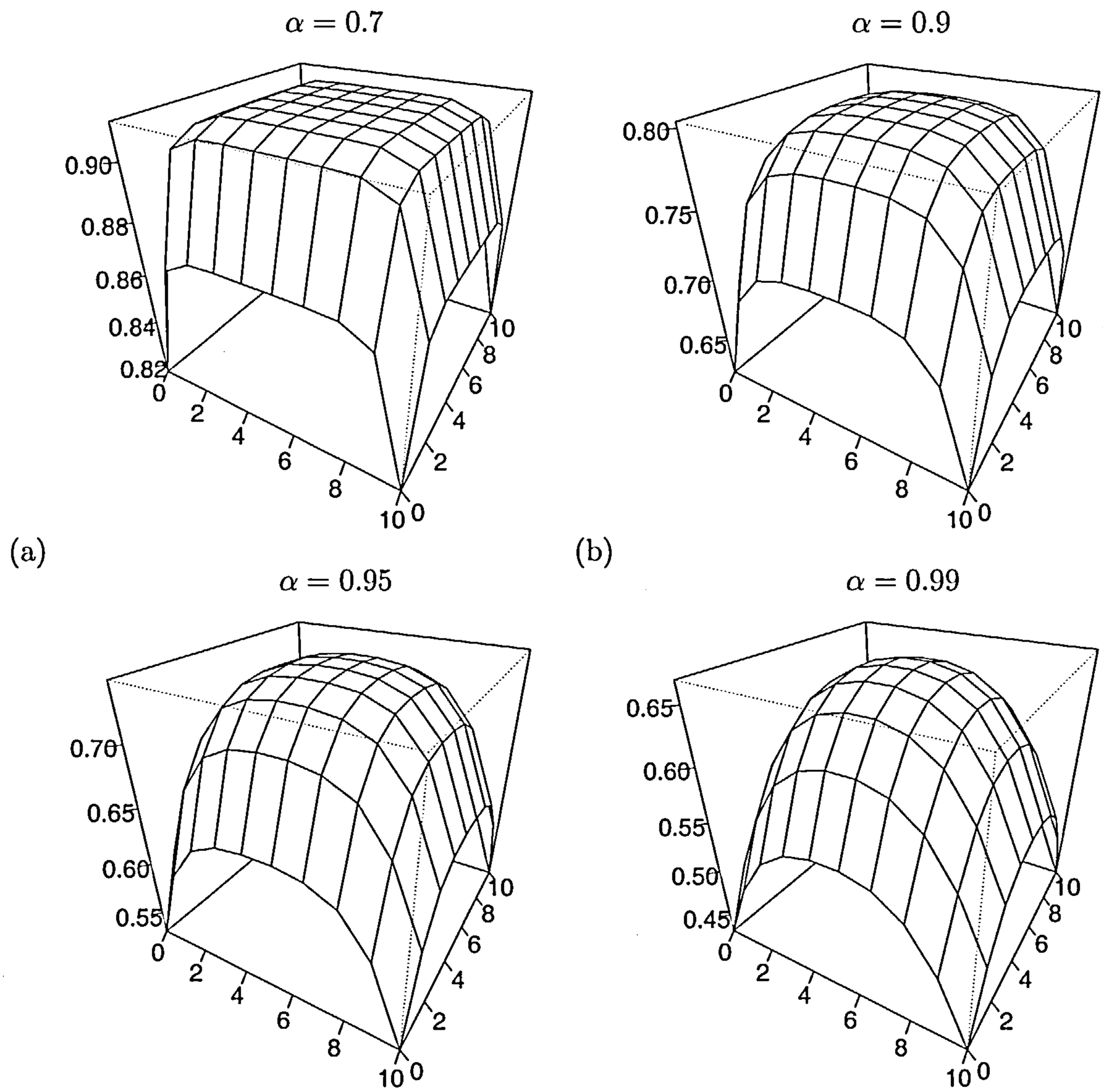


Figure 3.2: *Precision of the system on a* 10×10 *spatial grid at time T=20 with scaled edge weights for (a)* $\alpha = 0.7$*, (b)* $\alpha = 0.9$*, (c)* $\alpha = 0.95$*, (d)* $\alpha = 0.99$.

$$\alpha\left(\frac{1}{16}\left(\theta_{i-1,j-1}^{(t-1)}+\theta_{i-1,j+1}^{(t-1)}+\theta_{i+1,j-1}^{(t-1)}+\theta_{i+1,j+1}^{(t-1)}\right)\right.$$
$$\left.+\frac{1}{8}\left(\theta_{i-1,j}^{(t-1)}+\theta_{i,j-1}^{(t-1)}+\theta_{i,j+1}^{(t-1)}+\theta_{i+1,j}^{(t-1)}\right)+\frac{1}{4}\theta_{i,j}^{(t-1)}\right).$$

The scaled model has the same edge weights for the internal nodes but the corner and edge weights are scaled by $\frac{16}{9}$ and $\frac{4}{3}$ respectively to make the sum of the edge weights

equal α. This scaled model results in edge effects which are still non stationary in space, see Figure 3.2. For this simple scaling Table 3.2 shows that the system is less stationary than the unadjusted model with the range of values being larger. The size of the range varies between 0.0952865 for $\alpha = 0.7$ and 0.2380848 for $\alpha = 0.99$ compared with 0.0349894 and 0.1946258 for the unadjusted model. As the temporal dependence increases Figure 3.2 shows the precision surfaces become more peaked. This suggests the scaled model over adjusts the edge weights as the precision is much smaller at the corners and along the edges of the system than for the internal nodes.

The approach examined in this chapter uses modifications of the conditional distributions along the edges of the system to define the latent structure. For known temporal dependence the process is linear Gaussian and the time stationary distribution is easily obtained. The stationary variance is required to calculate this conditional distribution for a specific model. Two examples are considered the 1+1D and 2+1D models with zero mean and unit precision and the parent structure defined by the binomial weight matrix. Two extensions for the model are then considered; non-unit precision and non-zero mean cases.

3.2 The Conditional Distribution

Examination of the conditional distribution of the system allows not only the distribution of the internal nodes to be known but also the distribution of the nodes along the edge. Recall that only adjusting the weights applied to the parents of a node in the scaled model caused the size of the precision range to be increased so now consider an approach that scales both the edge weights and the corresponding precision of the latent structure.

Given a multivariate Normal distribution $Z = (X, Y)$, where
$Z \sim N\left(\begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}, \begin{pmatrix} \Sigma_x & \Sigma_{xy} \\ \Sigma_{yx} & \Sigma_y \end{pmatrix}\right)$ with sub-vectors $X \sim N(\mu_x, \Sigma_x)$ and
$Y \sim N(\mu_y, \Sigma_y)$ then the conditional distribution of $X|Y$ is given by

$$X|Y \sim N\left(\mu_x + \Sigma_{x,y}\Sigma_y^{-1}(X - \mu_y), \Sigma_x - \Sigma_{x,y}\Sigma_y^{-1}\Sigma_{y,x}\right). \tag{3.1}$$

This equation explicitly shows the relationship between X and Y in the conditional distribution with the mean of the distribution given by $\mu_{\mathbf{x}} + \Sigma_{x,y}\Sigma_y^{-1}(X - \mu_{\mathbf{y}})$ and the variance given by $\Sigma_x - \Sigma_{x,y}\Sigma_y^{-1}\Sigma_{y,x}$. The first term of this distribution is often described as the conditional expectation and the second as the conditional variance.

The latent process $\theta^{(t)}$ represents a stochastic process such that the system equation is a time homogeneous Markov chain where

$$\left(\theta^{(\mathbf{t})} \middle| \theta^{(\mathbf{t-1})}, \theta^{(\mathbf{t-2})}, \ldots, \theta^{(\mathbf{0})}\right) = \left(\theta^{(\mathbf{t})} \middle| \theta^{(\mathbf{t-1})}\right) \sim N\left(G\theta^{(\mathbf{t-1})}, \tau_s^{-1} I\right).$$

This expression describes the conditional distribution adequately for the internal nodes but does not indicate how to handle edge effects. Define l to be the spatial location of a node and $\text{pa}\left(\theta_l^{(t)}\right)$ to be the vector of parents of node l as defined in Chapter 2. The conditional distribution for the spatio-temporal dynamic linear model is given by

$$\begin{aligned}
\theta_l^{(t)} | \,\text{pa}\left(\theta_l^{(t)}\right) \;\sim\; & N\left(E\left(\theta_l^{(t)}\right) + Cov\left(\theta_l^{(t)}, \text{pa}\left(\theta_l^{(t)}\right)\right)\right. \\
& \times Var\left(\text{pa}\left(\theta_l^{(t)}\right)\right)^{-1}\left(\text{pa}\left(\theta_l^{(t)}\right) - E\left(\text{pa}\left(\theta_l^{(t)}\right)\right)\right) \\
& , Var\left(\theta_l^{(t)}\right) - Cov\left(\theta_l^{(t)}, \text{pa}\left(\theta_l^{(t)}\right)\right) \\
& \left. \times Var\left(\text{pa}\left(\theta_l^{(t)}\right)\right)^{-1} Cov\left(\text{pa}\left(\theta_l^{(t)}\right), \theta_l^{(t)}\right)\right). \qquad (3.2)
\end{aligned}$$

This equation describes both the conditional distribution for the 1+1D model, where l is a single number representing spatial location and the 2+1D model where l is a coordinate representing the spatial location.

3.2.1 The Stationary Variance Matrix

In order to use the distribution defined by Equation (3.2) the time stationary variance matrix Σ for the system is necessary. From equation (2.1) $\theta^{(\mathbf{t})} = G\theta^{(\mathbf{t-1})} + \epsilon^{(t-1)}$ where $\epsilon^{(t-1)}$ is white noise with $\mathrm{E}\left(\epsilon^{(t-1)}\right) = 0$ and $\mathrm{Var}\left(\epsilon^{(t-1)}\right) = \tau_s^{-1} I$. Hence for the stationary

variance matrix Σ

$$\begin{aligned} Var\left(\theta^{(t)}\right) &= Var\left(G\theta^{(t-1)}\right) + \tau_s^{-1} I \\ &= GVar\left(\theta^{(t-1)}\right) G' + \tau_s^{-1} I \\ \text{so that } \Sigma &= G\Sigma G' + \tau_s^{-1} I \end{aligned}$$

this equation uses the identity $Var\left(\theta^{(t)}\right) = Var\left(\theta^{(t-1)}\right)$ for a stationary system. Trying a solution of the form $\Sigma = \tau_s^{-1}(I - GG')^{-1}$ gives

$$\begin{aligned} \tau_s^{-1}(I - GG')^{-1} &= \tau_s^{-1} G(I - GG')^{-1} G' + \tau_s^{-1} I \\ \Leftrightarrow (I - GG')^{-1} &= G(I - GG')^{-1} G' + I \\ \Leftrightarrow I &= G(I - GG')^{-1} G'(I - GG') + I - GG' \\ \Leftrightarrow GG' &= G(I - GG')^{-1} G'(I - GG') \\ \Leftrightarrow G &= G(I - GG')^{-1}(I - G'G) \\ \Leftrightarrow I &= (I - GG')^{-1}(I - G'G). \end{aligned}$$

This is true provided $GG' = G'G$, hence the stationary variance matrix is

$$\tau_s^{-1}(I - GG')^{-1}. \tag{3.3}$$

The models considered in this chapter have the evolution matrix G which is defined with the property $G = G'$, thus for all of the models considered in this thesis the variance matrix is given by Equation 3.3 since $GG' = G'G$ is satisfied.

The stationary variance matrix depends on the temporal dependence parameter through the weight matrix G thus the edge weights applied by using Equation (3.2) also depend on the temporal dependence parameter. In order to reduce the computational time in the MCMC simulation a reference table with these edge adjustments is required otherwise for each iteration in the MCMC simulation new edge weights and conditional variances will need to be calculated if the temporal dependence is one of the parameters being simulated.

Creating a reference table

Consider zero mean models with unit precision in either 1+1D or 2+1D. Calculating the stationary variance matrix and then the conditional expectation and conditional variance for each iteration in MCMC simulation is computationally demanding so consider creating the table for $\alpha \in [0, 1)$ in steps of 0.01. A simple R routine (http://www.r-project.org/) is used to calculate $\Sigma = (I - GG')^{-1}$ the stationary variance matrix for each α . This matrix is considered stationary when there is no noticeable change in the values obtained by increasing the system size. In this thesis the variance matrix is considered to be stationary when there is no change in the values obtained for the stationary variance matrix to seven decimal places by increasing the system size. Within R the relevant variance sub-matrix for a given latent point can then be read from Σ and the edge weights and conditional variances calculated using Equation 3.2. These results can then be output in tabular form as a reference table to be read into the MCMC simulation program.

α	Edge Weights						Precision Adjustment	
	Edge nodes			Corner Nodes			Edge nodes	Corner nodes
	2 Step	1 Step	0 Step	2 Step	1 Step	0 Step		
0.90	0.0615	0.1312	0.2476	0.0674	0.1380	0.2699	0.9741	0.9486
0.91	0.0625	0.1336	0.2513	0.0688	0.1409	0.2751	0.9731	0.9465
0.92	0.0635	0.1360	0.2551	0.0704	0.1440	0.2806	0.9721	0.9443
0.93	0.0646	0.1386	0.2590	0.0721	0.1472	0.2863	0.9711	0.9419
0.94	0.0657	0.1413	0.2631	0.0739	0.1508	0.2924	0.9700	0.9393
0.95	0.0670	0.1443	0.2673	0.0760	0.1546	0.2990	0.9688	0.9364

Table 3.3: *Extract of the reference table for the 2+1D model with fine edge weights showing the edge and precision adjustments for six values of temporal dependence.*

Table 3.3 is an extract of the full reference table for the 2+1D model with six different values of temporal dependence, the full table is given in Appendix A.1. The columns

relating to the edge weights give the conditional edge weights for the parents of nodes which lie along the edge or corner of the system. The zero, one and two step value relates to the spatial point at the previous time period, its first order parent nodes and its second order parent nodes. The precision adjustments, when multiplied by the precision for the model, give the conditional precision for nodes along the edge and at the corners of the system. The corresponding values for the internal nodes are obtained directly from the weight matrix (G) and the state precision.

3.3 1+1D model

Consider a 1+1D model with binomial weights and for simplicity, with zero mean and unit precision. The latent structure is defined by

$$\theta^{(t)} \sim N\left(G\theta^{(t-1)}, I\right).$$

Using the isotropic binomial weight model the evolution matrix G is defined as in Equation (2.6) where the elements of G are given by the elements of the edge weight vector g. For this isotropic binomial weight model the edge weights for a particular node are given by,

$$g = \alpha\left(\frac{1}{4}, \frac{1}{2}, \frac{1}{4}\right).$$

For the internal nodes $\text{pa}\left(\theta_i^{(t)}\right) = \left(\theta_{i-1}^{(t-1)}, \theta_i^{(t-1)}, \theta_{i+1}^{(t-1)}\right)$ where i is the spatial location of an internal node and at the edges of the system $\text{pa}\left(\theta_0^{(t)}\right) = \left(\theta_0^{(t-1)}, \theta_1^{(t-1)}\right)$ and $\text{pa}\left(\theta_{n-1}^{(t)}\right) = \left(\theta_{n-2}^{(t-1)}, \theta_{n-1}^{(t-1)}\right)$.

Figure 3.3 (a) and (b) show the precision plots for fixed temporal dependence $\alpha = 0.9$ for the unadjusted and scaled models, this system is not stationary and the edge weights do need adjusting. In order to use Equation (3.2) we require the stationary variance and covariance for the model. Since $G = G'$ the stationary variance for the full model can be obtained for fixed α using Equation (3.3) and the method described in Section 3.2.1.

From Σ the variance matrix σ for a particular node is given by

$$\sigma = \begin{pmatrix} v_0 & v_1 & v_2 \\ v_1 & v_0 & v_1 \\ v_2 & v_1 & v_0 \end{pmatrix}.$$

Hence only three values are needed (v_0, v_1 and v_2) in order to calculate the covariance between nodes. Since an isotropic binomial edge weight model is being considered $Cov(\theta_i^{(t)}, \theta_{i-1}^{(t-1)}) = Cov(\theta_i^{(t)}, \theta_{i+1}^{(t-1)})$ thus only two values are necessary for the conditional distribution (as defined in Chapter 2);

0 Step

$$\begin{aligned} Cov\left(\theta_i^{(t)}, \theta_i^{(t-1)}\right) &= Cov\left(\frac{\alpha}{4}\left(\theta_{i-1}^{(t-1)} + 2\theta_i^{(t-1)} + \theta_{i+1}^{(t-1)}\right), \theta_i^{(t-1)}\right) \\ &= \frac{\alpha}{4}\left(v_1 + 2v_0 + v_1\right) \\ &= \frac{\alpha}{2}\left(v_1 + v_0\right) \\ &\equiv C_0, \end{aligned}$$

1 Step

$$\begin{aligned} Cov\left(\theta_i^{(t)}, \theta_{i+1}^{(t-1)}\right) &= Cov\left(\frac{\alpha}{4}\left(\theta_{i-1}^{(t-1)} + 2\theta_i^{(t-1)} + \theta_{i+1}^{(t-1)}\right), \theta_{i+1}^{(t-1)}\right) \\ &= \frac{\alpha}{4}\left(v_2 + 2v_1 + v_0\right) \\ &\equiv C_1. \end{aligned}$$

These two values along with σ provide all the information required to calculate the conditional distribution for the edge nodes. For the zero mean model both $E\left(\text{pa}\left(\theta_i^{(t)}\right)\right)$ and $E\left(\theta_i^{(t)}\right)$ equal zero. The conditional distribution for $\theta_i^{(t)} | \text{pa}\left(\theta_i^{(t)}\right)$ can now be calculated using Equation (3.2). This allows the edge weights to be adjusted according to the number of parent nodes available as well as adjusting the system variance to take into account the reduction of information available for nodes along the edge of the system.

Using Equation (3.2) the adjusted edge weight for node 0 is given by

$$\begin{aligned} E\left(\theta_0^{(t)} \middle| \mathrm{pa}\left(\theta_0^{(t)}\right)\right) &= Cov\left(\theta_0^{(t)}, \mathrm{pa}\left(\theta_0^{(t)}\right)\right) Var\left(\mathrm{pa}\left(\theta_0^{(t)}\right)\right)^{-1} \mathrm{pa}\left(\theta_0^{(t)}\right) \\ &= Cov\left(\theta_0^{(t)}, \left(\theta_0^{(t-1)}, \theta_1^{(t-1)}\right)\right) Var\left(\theta_0^{(t-1)}, \theta_1^{(t-1)}\right)^{-1} \left(\theta_0^{(t-1)}, \theta_1^{(t-1)}\right) \\ &= (C_0, C_1) \begin{pmatrix} v_0 & v_1 \\ v_1 & v_0 \end{pmatrix}^{-1} \left(\theta^{(t-1)}\right). \end{aligned}$$

The corresponding conditional variance is given by

$$\begin{aligned} Var\left(\theta_0^{(t)} | \mathrm{pa}\left(\theta_0^{(t)}\right)\right) &= Var\left(\theta_0^{(t)}\right) - Cov\left(\theta_0^{(t)} \mathrm{pa}\left(\theta_0^{(t)}\right)\right) \\ &\quad \times Var\left(\mathrm{pa}\left(\theta_0^{(t)}\right)\right)^{-1} Cov\left(\mathrm{pa}\left(\theta_0^{(t)}\right), \theta_0^{(t)}\right) \\ &= Var\left(\theta_0^{(t)}\right) - Cov\left(\theta_0^{(t)}, \left(\theta_0^{(t-1)}, \theta_1^{(t-1)}\right)\right) \\ &\quad \times Var\left(\theta_0^{(t-1)}, \theta_1^{(t-1)}\right)^{-1} Cov\left(\left(\theta_0^{(t-1)}, \theta_1^{(t-1)}\right), \theta_0^{(t)}\right) \\ &= v_0 - (C_0, C_1) \begin{pmatrix} v_0 & v_1 \\ v_1 & v_0 \end{pmatrix}^{-1} \begin{pmatrix} C_0 \\ C_1 \end{pmatrix} \\ &= v_0 - \frac{1}{v_o^2 - v_1^2} (C_0, C_1) \begin{pmatrix} v_0 & -v_1 \\ -v_1 & v_0 \end{pmatrix} \begin{pmatrix} C_0 \\ C_1 \end{pmatrix} \\ &= v_0 - \frac{1}{v_o^2 - v_1^2} (C_0, C_1) \begin{pmatrix} C_0 v_0 - C_1 v_1 \\ -C_0 v_1 + C_1 v_0 \end{pmatrix} \\ &= v_0 - \frac{1}{v_o^2 - v_1^2} \left(C_0^2 v_0 - 2C_0 C_1 v_1 + C_1^2 v_0\right) \\ &= v_0 - \frac{1}{v_o^2 - v_1^2} \left(\left[\frac{a}{2}(v_1 + v_0)\right]^2 v_0 - 2\left[\frac{a}{2}(v_1 + v_0)\right] \right. \\ &\quad \left. \times \left[\frac{\alpha}{4}(v_2 + 2v_1 + v_0)\right] v_1 + \left[\frac{\alpha}{4}(v_2 + 2v_1 + v_0)\right]^2 v_0 \right). \end{aligned}$$

Since the binomial weight model is being used these equations are essentially the same for node n. For the internal nodes the edge weights are those defined by G and the conditional variance is unchanged.

Both the conditional expectation and conditional variance depend on the value of the temporal dependence. If a model with fixed temporal dependence is being considered

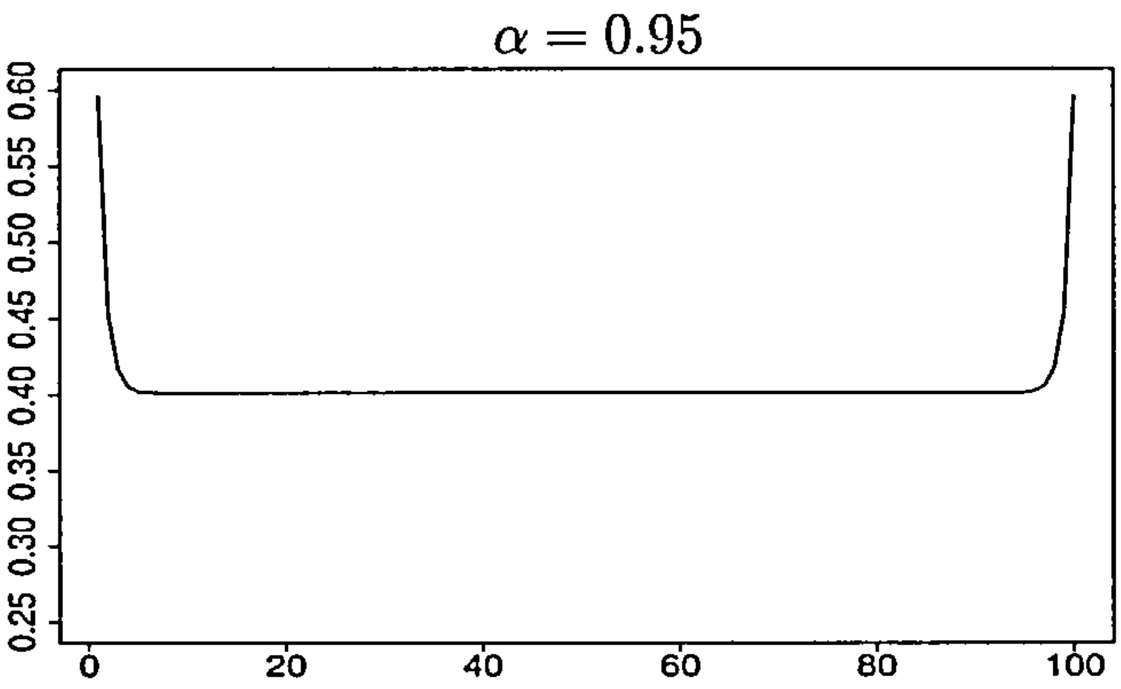
then the above distribution only needs to be calculated once for the edges of the system. However if the temporal dependence is to be simulated the distribution would need to be calculated for each iteration, increasing the complexity and time required for the MCMC simulation. In this situation the reference table mentioned in Section 3.2 is a sensible approach. The routines used to create this reference table do not need to calculate the conditional distribution for the internal nodes as these are obtained directly from the evolution matrix and the state precision.

3.3.1 Numerical Example

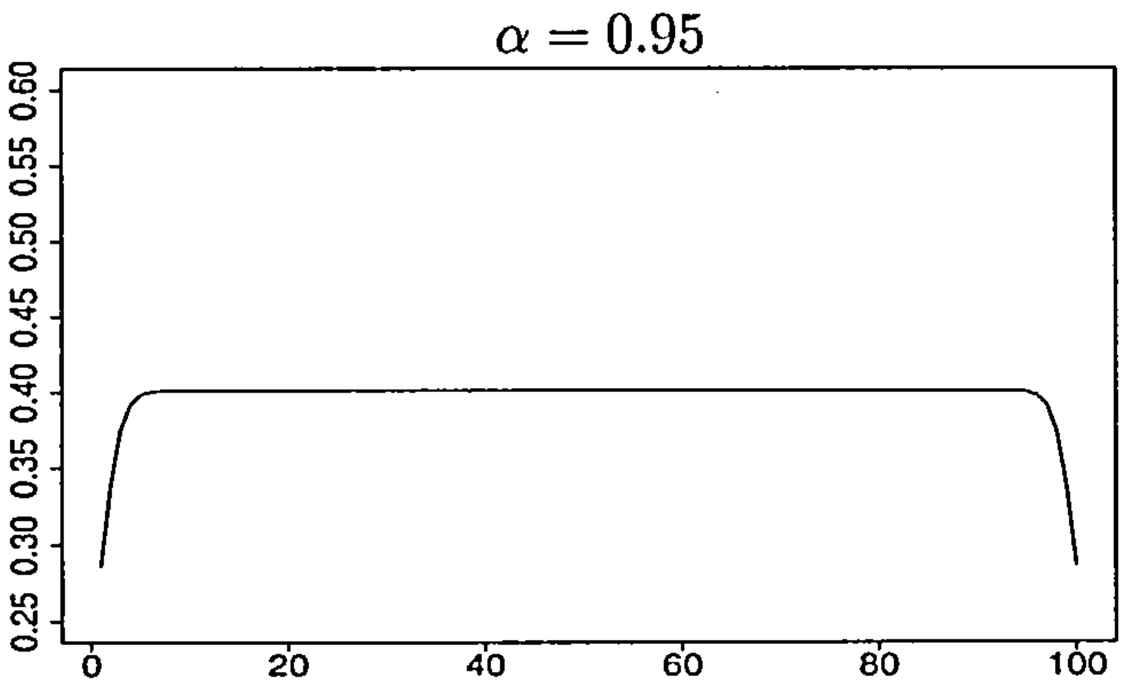
Consider the zero mean unit precision 1+1D model with 100 spatial nodes evolving over twenty time periods defined for four values of temporal dependence using the unadjusted, scaled and conditional models. Table 3.4 shows the results for the unadjusted model, here the precision is over estimated for all values of the temporal dependence and the range of values increases as $\alpha \to 1$ from 0.0538716 for $\alpha = 0.7$ to 0.2371421 for $\alpha = 0.99$. The scaled edge weight model (see Table 3.5) shows that the range varies with α increasing for $\alpha = 0.7$ and 0.9 then decreasing for $\alpha = 0.95$ and 0.99. The third model which exploits the conditional distribution of the edge nodes and under estimates the true precision for each temporal dependence. The range of values is considerably smaller than for the other models 0.0000000 for $\alpha = 0.7$ and 0.0520667 for $\alpha = 0.99$ (See Table 3.6). These results can been seen in Figure 3.3 for temporal dependence $\alpha = 0.95$. Hence for the 1+1D model the scaled model over corrects for the edge effects and has a larger range than the conditional model so the conditional model produces the most stationary distribution in space.

3.4 2+1D model

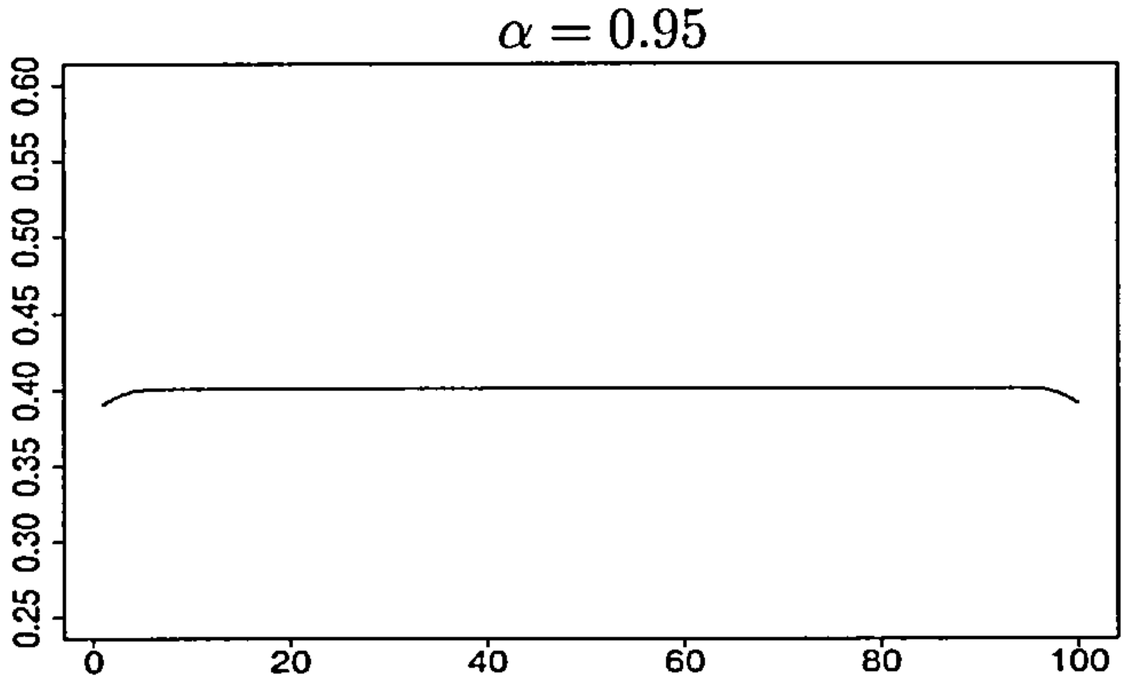
The extension of the 1+1D model to the 2+1D model follows directly from the method above with the introduction of more parent nodes to the model. Here the spatial dimen-



(a)



(b)



(c)

Figure 3.3: *Precision of the system for a 100 spatial node model at* $T = 20$ *and* $\alpha = 0.95$ *with (a) no edge adjustments, (b) scaled weights and (c) conditional weights.*

α	Stationary Precision	Range	Size of range
0.7	0.7713946	0.7713948 : 0.8252664	0.0538716
0.9	0.5144359	0.5161237 : 0.6633636	0.1472399
0.95	0.3854864	0.4009572 : 0.5963794	0.1954222
0.99	0.1867609	0.2871524 : 0.5242945	0.2371421

Table 3.4: *Stationary Precision and the associated range for a 100 spatial node model at time $T = 20$ with no edge weights.*

α	Stationary Precision	Range	Size of range
0.7	0.7713946	0.6866929 : 0.7713948	0.0847019
0.9	0.5144359	0.3963897 : 0.5161237	0.1197340
0.95	0.3854864	0.2863019 : 0.4009572	0.1146553
0.99	0.1867609	0.1903682 : 0.2871524	0.0967842

Table 3.5: *Stationary Precision and the associated range for a 100 spatial node model at time $T = 20$ with scaled edge weights.*

α	Stationary Precision	Range	Size of range
0.7	0.7713946	0.7713948 : 0.7713948	0.0000000
0.9	0.5144359	0.514859 : 0.5161237	0.0012676
0.95	0.3854864	0.3906899 : 0.4009572	0.0102673
0.99	0.1867609	0.2350857 : 0.2871524	0.0520667

Table 3.6: *Stationary Precision and the associated range for a 100 spatial node model at time $T = 20$ with conditional edge weights.*

sions are represented by i and j and the temporal dimension by t where $i = 0, 1, \ldots, n-1$, $j = 0, 1, \ldots, m-1$ and $t = 0, 1, \ldots, T-1$. In this model the edge corrections need to be calculated for both corner nodes (e.g. $(i = 0, j = 0)$, $(i = 0, j = m-1)$) and edge nodes (e.g. $(i = 0, j = 1, 2...m-2)$). In this example consider the nine parent model introduced

in Chapter 2 with zero mean and unit precision where the latent model is defined by

$$\theta^{(t)} \sim N\left(G\theta^{(t-1)}, I\right).$$

Using the isotropic binomial weight model the evolution matrix G is defined as in Equation (2.9) where the elements of G are given by the elements of the edge weight vector g. For this isotropic binomial weight model the edge weights for a particular node are given by,

$$g = \alpha \begin{pmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{pmatrix}.$$

For the internal nodes the vector of parent nodes is defined to be $\text{pa}\left(\theta_{i,j}^{(t)}\right) = \left(\theta_{i-1,j-1}^{(t-1)}, \theta_{i-1,j}^{(t-1)}, \theta_{i-1,j+1}^{(t-1)}, \theta_{i,j-1}^{(t-1)}, \theta_{i,j}^{(t-1)}, \theta_{i,j+1}^{(t-1)}, \theta_{i+1,j-1}^{(t-1)}, \theta_{i+1,j}^{(t-1)}, \theta_{i+1,j+1}^{(t-1)}\right)$ where i, j is the spatial location of an internal node. Similarly for edge and corner nodes $\text{pa}\left(\theta_{i,j}^{(t)}\right)$ represents the vector of parent nodes for node i, j where in this case i, j represents a node along the edge of the system.

Again the stationary variance matrix Σ is needed, since $G = G'$ this can be calculated using Equation (3.3) and the method described in Section 3.2.1. From Σ the variance matrix σ for a particular node can be extracted

$$\sigma = \left[\begin{array}{ccc|ccc|ccc} v_{00} & v_{01} & v_{02} & v_{01} & v_{11} & v_{12} & v_{02} & v_{12} & v_{22} \\ v_{01} & v_{00} & v_{01} & v_{11} & v_{01} & v_{11} & v_{12} & v_{02} & v_{12} \\ v_{02} & v_{01} & v_{00} & v_{12} & v_{11} & v_{01} & v_{22} & v_{12} & v_{02} \\ \hline v_{01} & v_{11} & v_{12} & v_{00} & v_{01} & v_{02} & v_{01} & v_{11} & v_{12} \\ v_{11} & v_{01} & v_{11} & v_{01} & v_{00} & v_{01} & v_{11} & v_{01} & v_{11} \\ v_{12} & v_{11} & v_{12} & v_{02} & v_{01} & v_{00} & v_{12} & v_{11} & v_{01} \\ \hline v_{02} & v_{12} & v_{22} & v_{01} & v_{11} & v_{12} & v_{00} & v_{01} & v_{02} \\ v_{12} & v_{02} & v_{12} & v_{11} & v_{01} & v_{11} & v_{01} & v_{00} & v_{01} \\ v_{22} & v_{12} & v_{02} & v_{12} & v_{11} & v_{01} & v_{02} & v_{01} & v_{00} \end{array}\right] \begin{array}{l} i-1, j-1 \\ i-1, j \\ i-1, j+1 \\ i, j-1 \\ i, j \\ i, j+1 \\ i+1, j-1 \\ i+1, j \\ i+1, j+1. \end{array}$$

Hence $v_{00}, v_{01}, v_{02}, v_{11}, v_{12}$, and v_{22} are required in order to calculate the covariance

between individual points. These values are obtained by solving (3.3), this is easily done by using an R routine as in the 1+1D case (see Section 3.3).

In the isotropic binomial weight model all four edges have the same weight, as do the four corners, hence $\mathrm{Cov}\left(\theta_{i,j}^{(t)}, \theta_{i+1,j}^{(t-1)}\right) = \mathrm{Cov}\left(\theta_{i,j}^{(t)}, \theta_{i-1,j}^{(t-1)}\right) = \mathrm{Cov}\left(\theta_{i,j}^{(t)}, \theta_{i,j+1}^{(t-1)}\right) = \mathrm{Cov}\left(\theta_{i,j}^{(t)}, \theta_{i,j-1}^{(t-1)}\right)$. Similarly the covariances between $\theta_{i,j}^{(t)}$ and the diagonal neighbours are the same, thus only C_0=$\mathrm{Cov}\left(\theta_{i,j}^{(t)}, \theta_{i,j}^{(t-1)}\right)$, C_1=$\mathrm{Cov}\left(\theta_{i,j}^{(t)}, \theta_{i+1,j}^{(t-1)}\right)$ and C_2=$\mathrm{Cov}\left(\theta_{i,j}^{(t)}, \theta_{i+1,j+1}^{(t-1)}\right)$, the zero, one and two step covariances, respectively, need to be calculated;

0 Step

$$\begin{aligned}
Cov\left(\theta_{i,j}^{(t)}, \theta_{i,j}^{(t-1)}\right) &= Cov\left(\frac{\alpha^2}{2^4}\left(\theta_{i-1,j-1}^{(t-1)} + 2\theta_{i-1,j}^{(t-1)} + \theta_{i-1,j+1}^{(t-1)} + 2\theta_{i,j-1}^{(t-1)} + 4\theta_{i,j}^{(t-1)}\right.\right. \\
&\qquad \left.\left. +2\theta_{i,j+1}^{(t-1)} + \theta_{i+1,j-1}^{(t-1)} + 2\theta_{i+1,j}^{(t-1)} + \theta_{i+1,j+1}^{(t-1)}\right), \theta_{i,j}^{(t-1)}\right) \\
&= \frac{\alpha^2}{2^4}\left(v_{11} + 2v_{01} + v_{11} + 2v_{01} + 4v_{00} + 2v_{01} + v_{11} + 2v_{01} + v_{11}\right) \\
&= \frac{\alpha^2}{2^4}\left(4v_{00} + 8v_{01} + 4v_{11}\right) \\
&= \frac{\alpha^2}{4}\left(v_{00} + 2v_{01} + v_{11}\right) \\
&\equiv C_0, \qquad (3.4)
\end{aligned}$$

1 Step

$$\begin{aligned}
Cov\left(\theta_{i,j}^{(t)}, \theta_{i+1,j}^{(t-1)}\right) &= Cov\left(\frac{\alpha^2}{2^4}\left(\theta_{i-1,j-1}^{(t-1)} + 2\theta_{i-1,j}^{(t-1)} + \theta_{i-1,j+1}^{(t-1)} + 2\theta_{i,j-1}^{(t-1)} + 4\theta_{i,j}^{(t-1)}\right.\right. \\
&\qquad \left.\left. +2\theta_{i,j+1}^{(t-1)} + 2\theta_{i+1,j-1}^{(t-1)} + \theta_{i+1,j}^{(t-1)} + 2\theta_{i+1,j+1}^{(t-1)}\right), \theta_{i+1,j}^{(t-1)}\right) \\
&= \frac{\alpha^2}{2^4}\left(v_{12} + 2v_{02} + v_{12} + 2v_{11} + 4v_{01} + 2v_{11} + v_{01} + 2v_{00} + v_{01}\right) \\
&= \frac{\alpha^2}{2^4}\left(2v_{00} + 6v_{01} + 2v_{02} + 4v_{11} + 2v_{12}\right) \\
&= \frac{\alpha^2}{2^3}\left(v_{00} + 3v_{01} + v_{02} + 2v_{11} + v_{12}\right) \\
&\equiv C_1, \qquad (3.5)
\end{aligned}$$

2 Step

$$\begin{aligned} Cov\left(\theta_{i,j}^{(t)}, \theta_{i+1,j+1}^{(t-1)}\right) &= Cov\left(\frac{\alpha^2}{2^4}\left(\theta_{i-1,j-1}^{(t-1)} + 2\theta_{i-1,j}^{(t-1)} + \theta_{i-1,j+1}^{(t-1)} + 2\theta_{i,j-1}^{(t-1)} + 4\theta_{i,j}^{(t-1)}\right.\right. \\ &\qquad \left.\left. +2\theta_{i,j+1}^{(t-1)} + 2\theta_{i+1,j-1}^{(t-1)} + \theta_{i+1,j}^{(t-1)} + 2\theta_{i+1,j+1}^{(t-1)}\right), \theta_{i+1,j+1}^{(t-1)}\right) \\ &= \frac{\alpha^2}{2^4}(v_{22} + 2v_{12} + v_{02} + 2v_{12} + 4v_{11} + 2v_{01} + v_{02} + 2v_{01} + v_{00}) \\ &= \frac{\alpha^2}{2^4}(v_{00} + 4v_{01} + 2v_{02} + 4v_{11} + 4v_{12} + v_{22}) \\ &\equiv C_2. \end{aligned} \tag{3.6}$$

Using these values the conditional expectation and the conditional variance can be calculated along the edges and at the corners of the system using Equation (3.2).

Conditional Expectation

The symmetry of the chosen binomial weight model means the conditional expectation only needs to be calculated once for the corners and once for the edges. However if the model did not have this symmetry this value would have to be calculated individually for each corner and each edge.

Using Equation (3.2) the conditional expectation is given by

$$\begin{aligned} E\left(\theta_{i,j}^{(t)} \middle| \text{pa}\left(\theta_{i,j}^{(t)}\right)\right) &= E\left(\theta_{i,j}^{(t)}\right) + Cov\left(\theta_{i,j}^{(t)}, \text{pa}\left(\theta_{i,j}^{(t)}\right)\right) \\ &\quad \times Var\left(\text{pa}\left(\theta_{i,j}^{(t)}\right)\right)^{-1}\left(\text{pa}\left(\theta_{i,j}^{(t)}\right) - E\left(\text{pa}\left(\theta_{i,j}^{(t)}\right)\right)\right) \end{aligned}$$

where $\text{pa}\left(\theta_{i,j}^{(t)}\right)$ is a vector of length four, six or nine depending upon the spatial location of θ i.e. a corner, edge or internal node. In this zero mean model $E\left(\theta_{i,j}^{(t)}\right)$ and $E\left(\text{pa}\left(\theta_{i,j}^{(t)}\right)\right)$ equal zero so the adjusted edge weights are given by

$$\begin{aligned} E\left(\theta_{i,j}^{(t)} \middle| \text{pa}\left(\theta_{i,j}^{(t)}\right)\right) &= Cov\left(\theta_{i,j}^{(t)}, \text{pa}\left(\theta_{i,j}^{(t)}\right)\right) \\ &\quad \times Var\left(\text{pa}\left(\theta_{i,j}^{(t)}\right)\right)^{-1}\left(\text{pa}\left(\theta_{i,j}^{(t)}\right)\right). \end{aligned} \tag{3.7}$$

Conditional Variance

The conditional variance is required to update the edges of the system and due to the symmetry of this binomial weight model only one value for the corners and one value for the edges needs to be calculated. From Equation (3.2) the conditional variance is given by

$$\begin{aligned} Var\left(\theta_{i,j}^{(t)} \middle| \text{pa}\left(\theta_{i,j}^{(t)}\right)\right) &= Var\left(\theta_{i,j}^{(t)}\right) - Cov\left(\theta_{i,j}^{(t)}, \text{pa}\left(\theta_{i,j}^{(t)}\right)\right) \\ &\quad \times Var\left(\text{pa}\left(\theta_{i,j}^{(t)}\right)\right)^{-1} Cov\left(\text{pa}\left(\theta_{i,j}^{(t)}\right), \theta_{i,j}^{(t)}\right). \end{aligned} \tag{3.8}$$

Again $\text{pa}\left(\theta_{i,j}^{(t)}\right)$ is a vector of length four, six or nine depending upon the spatial location of $\theta_{i,j}^{(t-1)}$.

The conditional distribution fully describes the evolution of the latent model for the internal nodes and the edge nodes for fixed α. If α is allowed to vary then the reference table produced by R (see Section 3.2) for the 2+1D model should be used for the MCMC simulation.

3.4.1 Numerical Example

As with the unadjusted and scaled edge weight models examined in Section 3.2 this conditional edge weight model can be used to produce surface plots of the precision of these models. Consider a zero mean unit precision simulated 2+1D model at time $T = 20$ on a 10×10 spatial grid defined for four values of temporal dependence using the conditional distribution to calculate the edge adjustments.

The results shown in Figure 3.4 and Table 3.7 show that the model is more stationary using these edge weights with the size of the range varying between 0.0009933 for $\alpha = 0.7$ and 0.0416968 for $\alpha = 0.99$ compared with 0.0952865 and 0.2380848 for the scaled model and 0.0349894 and 0.1946258 for the unadjusted model (see Tables 3.1 and 3.2). This adjusted model still shows some non-stationarity at the edge of the system but not as pronounced as for the unadjusted model and it is also more tightly bounded than in the

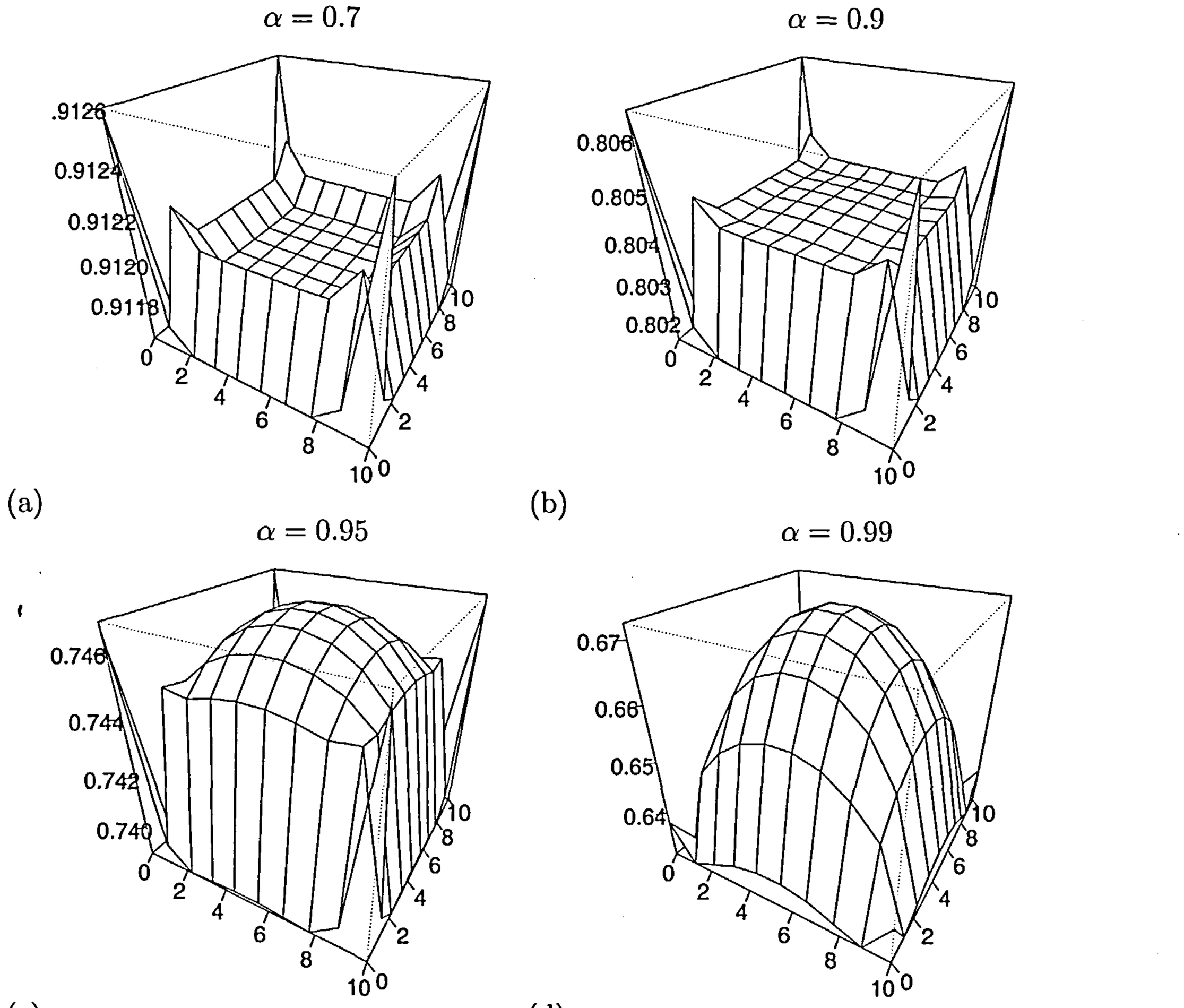


Figure 3.4: *Precision of the system on a 10 × 10 spatial grid at time t=20 with fine edge adjustments for (a)* $\alpha = 0.7$*, (b)* $\alpha = 0.9$*, (c)* $\alpha = 0.95$ *and (d)* $\alpha = 0.99$.

other two models (Figures 3.1, 3.2 and 3.4). Both the scaled model (Figure 3.2) and the conditional model (Figure 3.4) show the precision along the edges to be larger than the precision for internal nodes suggesting they have over corrected for the edge effects, however the conditional model has lower precision at the corners and the size of the range is very small compared with the other two methods. Hence for the 2+1D model the conditional distribution approach yields the most stationary distribution of the methods considered.

α	Stationary Precision	Range	size of range
0.7	0.9119323	0.9116086 : 0.9126019	0.0009933
0.9	0.8039998	0.8014747 : 0.8064991	0.0050244
0.95	0.7429752	0.7388498 : 0.7468511	0.0080013
0.99	0.627286	0.6306729 : 0.6723697	0.0416968

Table 3.7: *Stationary Precision and the associated range for 10 × 10 spatial model at time $T = 20$ with adjusted edge weights.*

3.5 Extensions

The adjusted weights can be calculated for $\alpha \in [0, 1)$ in steps of 0.01 using a basic R routine producing a reference table which can be used for any system where the state precision equals one. It is highly unlikely that real data will have $\tau_s = 1$ and so in this situation a method for converting the edge weights from $\tau_s = 1$ to $\tau_s^* \neq 1$ is necessary. Much real life data is also unlikely to have zero mean e.g. mean ocean temperature or pollution levels, so knowing how to adjust for a non zero mean in the model would be very useful.

3.5.1 Changing the state precision

A generalisation of the model discussed in this chapter is to consider models with non unit precision. The development of this model follows as in Section 3.2. In order to develop the conditional distribution the stationary variance matrix is required. From Equation 3.3 the stationary variance matrix is given by $\tau_s^{-1}(I - GG')^{-1}$. Define V to be the variance matrix with elements $v_{i,j}$ for the model with unit precision and $\tilde{V}$ to be the variance matrix with elements $\tilde{v}_{i,j}$ for the model with non unit state precision. The variance matrices are linked via the equation

$$\tilde{V}\frac{1}{\tau_s}V$$

i.e. the elements of the matrices are linked via the relationship

$$\tilde{v}_{i,j}\frac{1}{\tau_s}v_{i,j}. \tag{3.9}$$

Using this relationship the zero, one and two step covariances can be calculated as before;

$$C_0' = \frac{\alpha}{2^4}(\tilde{v}_{00} + 8\tilde{v}_{01} + 4\tilde{v}_{11}).$$

Using (3.9)

$$\begin{aligned} C_0' &= \frac{\alpha}{4}(\frac{v_{00}}{\tau_s} + 2\frac{v_{01}}{\tau_s} + \frac{v_{11}}{\tau_s}) \\ &= \frac{\alpha}{4\tau_s}(v_{00} + 2v_{01} + v_{11}) \\ &= \frac{C_0}{\tau_s} \end{aligned} \tag{3.10}$$

where C_0' is the zero step covariance for the model with $\tau_s \neq 1$. Similarly

$$C_1' = \frac{C_1}{\tau_s}, \tag{3.11}$$

$$C_2' = \frac{C_2}{\tau_s}. \tag{3.12}$$

Now the stationary variance matrix and covariance relations have been defined for the general model the conditional distribution can be defined. The conditional expectation is

given by,

$$
\begin{aligned}
E\left(\theta_{i,j}^{(t)}|\,\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\right) &= Cov\left(\theta_{i,j}^{(t)},\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\right) Var\left(\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\right)^{-1}\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\\
&= Cov(\theta_{i,j}^{(t)},(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)}))\\
&\quad \times Var(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)})^{-1}\\
&\quad \times(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)})\\
&= (C_0',C_1',C_1',C_2')\tilde{V}^{-1}(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)})\\
&= \left(\frac{1}{\tau_s}C_0,\frac{1}{\tau_s}C_1,\frac{1}{\tau_s}C_1,\frac{1}{\tau_s}C_2\right)\left(\frac{1}{\tau_s}V\right)^{-1}\\
&\quad \times(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)})\\
&= \frac{1}{\tau_s}(C_0,C_1,C_1,C_2)\tau_s^{*}V^{-1}(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)})\\
&= (C_0,C_1,C_1,C_2)V^{-1}(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)})\\
&= E\left(\theta_{i,j}^{(t)}|\,\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\right)\Big|_{\tau_s=1}. \qquad (3.13)
\end{aligned}
$$

Equation 3.13 shows that the conditional expectation is unaffected by changes in the state precision. Similarly the conditional variance is given by,

$$
\begin{aligned}
Var\left(\theta_{i,j}^{(t)}|\theta_{i,j}^{(t)}\right) &= Var\left(\theta_{i,j}^{(t)}\right) - Cov\left(\theta_{i,j}^{(t)},\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\right)\\
&= \tilde{v}_{00} - Cov(\theta_{i,j}^{(t)},(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)}))\\
&\quad \times Var(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)})^{-1}\\
&\quad \times Cov(\theta_{i,j}^{(t)},(\theta_{i,j}^{(t-1)},\theta_{i,j+1}^{(t-1)},\theta_{i+1,j}^{(t-1)},\theta_{i+1,j+1}^{(t-1)}))'\\
&= \tilde{v}_{00} - (C_0',C_1',C_1',C_2)\tilde{V}^{-1}(C_0',C_1',C_1',C_2')'\\
&= \frac{1}{\tau_s}v_{00} - \left(\frac{1}{\tau_s}C_0,\frac{1}{\tau_s}C_1,\frac{1}{\tau_s}C_1,\frac{1}{\tau_s}C_2\right)\left(\frac{1}{\tau_s}V\right)^{-1}\\
&\quad \times\left(\frac{1}{\tau_s}C_0,\frac{1}{\tau_s}C_1,\frac{1}{\tau_s}C_1,\frac{1}{\tau_s}C_2\right)'\\
&= \frac{1}{\tau_s}v_{00} - \frac{1}{\tau_s}(C_0,C_1,C_1,C_2)\tau_s V^{-1}\frac{1}{\tau_s}(C_0,C_1,C_1,C_2)'\\
&= \frac{1}{\tau_s}\left(v_{00} - (C_0,C_1,C_1,C_2)V^{-1}(C_0,C_1,C_1,C_2)'\right)\\
&= \frac{1}{\tau_s}Var\left(\theta_{i,j}^{(t)}|\,\mathrm{pa}\left(\theta_{i,j}^{(t)}\right)\right)\Big|_{\tau_s=1}. \qquad (3.14)
\end{aligned}
$$

Equation 3.14 shows that the conditional variance is scaled by non unit precision. Hence once the updates for the model where $\tau_s = 1 \; \forall \; \alpha$ are known the model for any value of τ_s can be updated by transforming the conditional precisions by $\frac{1}{\tau_s}$ and using the same edge weights.

3.5.2 Changing the Mean

A further generalisation of this model is to consider models with non-zero mean. Returning to Equation (3.2) the conditional expectation is given by

$$E\left(\theta_{i,j}^{(t)} | \operatorname{pa}\left(\theta_{i,j}^{(t-1)}\right)\right) = E\left(\theta_{i,j}^{(t)}\right) + \underbrace{Cov\left(\theta_{i,j}^{(t)}, \operatorname{pa}\left(\theta_{i,j}^{(t)}\right)\right) Var\left(\operatorname{pa}\left(\theta_{i,j}^{(t)}\right)\right)^{-1}}_{a^T} \times \left(\operatorname{pa}\left(\theta_{i,j}^{(t)}\right) - E\left[\operatorname{pa}\left(\theta_{i,j}^{(t-1)}\right)\right]\right)$$

where a^T is the edge weights calculated in Section 3.4 giving

$$\begin{aligned} &= E\left(\theta_{i,j}^{(t)}\right) + a^T\left[\operatorname{pa}\left(\theta_{i,j}^{(t)}\right) - E\left(\operatorname{pa}\left(\theta_{i,j}^{(t)}\right)\right)\right] \\ &= E\left(\theta_{i,j}^{(t)}\right) - a^T E\left(\operatorname{pa}\left(\theta_{i,j}^{(t)}\right)\right) + a^T \operatorname{pa}\left(\theta_{i,j}^{(t)}\right) \\ &= \mu\left(1 - a^T 1\right) + a^T \operatorname{pa}\left(\theta_{i,j}^{(t)}\right) \end{aligned} \tag{3.15}$$

for the stationary distribution. Note that $a^T\mathbf{1}$ is the sum of the edge weights and hence equals α for internal nodes. The system equation can now be expressed as

$$\theta_{i,j}^{(t)} \left| \operatorname{pa}\left(\theta_{i,j}^{(t)}\right) \right. \sim N\left(G \operatorname{pa}\left(\theta_{i,j}^{(t)}\right)(1-\alpha)\,\mu, \tau_s^{-1} I\right) \tag{3.16}$$

for internal nodes and

$$\theta_{i,j}^{(t)} \left| \operatorname{pa}\left(\theta_{i,j}^{(t)}\right) \right. \sim N\left(a^T \operatorname{pa}\left(\theta_{i,j}^{(t)}\right)\left(1 - a^T 1\right)\mu, \tau_s^{-1} I\right) \tag{3.17}$$

along the edges.

Equations (3.16) and (3.17) give the distribution of the latent nodes for systems with non-zero mean, Equation (3.15) gives the conditional expectation for this model. The edge weights are given by the sum of $\mu\left(1 - a^T 1\right)$ and the edge weights from the zero mean model. Thus models with zero mean can be easily extended to allow for non-zero

mean data by increasing the edge weights by $\mu\left(1 - a^T\mathbf{1}\right)$. The conditional variance is unaffected by changes in the mean of the data. The table of edge weights generated for the zero mean model can still be used in this non-zero mean model along with the addition of $\mu\left(1 - a^T\mathbf{1}\right)$ at each location.

3.6 Discussion

The way in which edge effects are handled affects the stationarity of the underlying latent model. This chapter has compared three approaches to dealing with these edge effects; unadjusted, scaled and conditional models.

Setting the missing edge nodes to zero in both 1+1D and 2+1D models leads to a non stationary precision surface with the edges and corners having a considerably larger precision associated with them.

The scaled model produces precision surfaces which over correct for the edge effects. In the 1+1D model the size of the range appeared to be bounded, in the 2+1D model the size of the range was larger for the scaled model than for the unadjusted model.

The conditional model produced precision surfaces with the smallest range for both the 1+1D and 2+1D situations. When the temporal dependence is small this model results in a stationary system.

The edge weights and conditional variances are easily generated using an R routine for $\alpha \in [0, 1)$ in steps of 0.01 for a zero mean and unit precision. Sections 3.5.1 and 3.5.2 provide the tools for transforming these edge weights and precisions for models with non zero mean and/or non unit precision, thus for a model with given G which is constant in space and time this reference table can be used in the MCMC schemes to allow for edge effects in any model. This is especially useful if the state precision and temporal dependence are variables being simulated.

Chapter 4

Markov Chain Monte Carlo Simulation Techniques

4.1 Introduction

Spatio-temporal models defined by the DLM have a number of parameters which require estimation; the mean of the latent structure μ, the precision of the latent structure τ_s, observational precision τ_o and temporal dependence parameter α. This chapter is an introduction to MCMC simulation for DLM's which initially concentrates on the estimation of the precision parameters. Later examples extend the simulation scheme to include the mean parameter. The spatial and temporal dependency parameters are examined in Chapter 6.

Markov chain Monte Carlo (MCMC) methods provide a mechanism for simulating realisations from a target distribution, $\pi(\phi)$. MCMC simulation generates a sample from $\pi(\phi)$ by simulating a Markov chain with this as its stationary distribution. Details of the theory behind MCMC simulations is widely covered in the literature both in a general form and described specifically for different applications. For a general introduction to MCMC see Gamerman (1997), Besag et al. (1995) and Besag (2001). A selection of papers discussing MCMC methods in spatio-temporal modelling were discussed in Section 1.3

in a more general framework. Hurn et al. (1999) examine block updating in MCMC and Robert (1995) examines the convergence of MCMC chains. Wilkinson & Yeung (2002) and Wilkinson & Yeung (2001) both deal with Bayesian computation for large linear models. Parallel processing approaches for MCMC are discussed in Wilkinson (2004) and Whiley & Wilson (2002). Higdon et al. (2003) examines MCMC approaches for computationally intensive models and this issue is discussed in more detail in Chapter 5.

The MCMC simulation techniques are used in the Bayesian framework to explore the posterior distribution of the model parameters. Parameter estimates derived from these simulations can then be used to completely specify a DLM for the model. Bayes theorem for the posterior probability density function of a parameter vector ϕ given a data vector x is

$$\begin{aligned} \pi(\phi|x) &= \frac{\pi(\phi)\,L(\phi|x)}{f(x)} \\ &= \frac{\pi(\phi)\,L(\phi|x)}{\int_{\Phi}\pi(\phi)\,L(\phi|x)\,dx} \end{aligned} \tag{4.1}$$

where $L(\phi|x)$ is the likelihood function, the function $f(x)$ is not a function of ϕ and is integrated over the sample space of ϕ. Hence the posterior can be written as

$$\begin{aligned} \pi(\phi|x) &\propto \pi(\phi)\,L(\phi|x) \\ \text{posterior} &\propto \text{prior} \times \text{likelihood.} \end{aligned}$$

The key difficulty here from an analytic viewpoint is the difficulty of performing the integration required to normalise this density and further integration is required to obtain marginal distributions of functions of ϕ of particular interest.

4.2 Gibbs Sampling

Gibbs sampling generates samples from the posterior for ϕ by sampling from the full conditionals for each component of ϕ in turn. The components of ϕ are defined as ϕ_i

where $i = 1, \ldots, d$ and d is the total number of model parameters. Simulation from the full conditional is given by $\pi(\phi \,|\phi_{-i}, x)$ where $\phi_{-i} = \{\phi_i, \ldots \phi_{i-1}, \phi_{i+1}, \ldots \phi_d\}$.

4.2.1 The Gibbs Algorithm

Assume the full conditionals are of closed form i.e. semi conjugate priors have been specified specified which means the full conditional distributions are of standard form, and further are of the same form as the independent prior specifications. The Gibbs algorithm is then

1. Initialise counter to $j = 1$. Initialise the state of the chain $\phi^{(0)} = \left(\phi_1^{(0)}, \ldots, \phi_d^{(0)}\right)'$ in the support of $\pi(\phi \,|x)$

2. Obtain a new value $\phi^{(j)}$ from $\phi^{(j-1)}$ by successive simulation from the full conditionals.

$$
\begin{aligned}
\phi_1^{(j)} &\sim \pi\left(\phi_1 \left| \phi_2^{(j-1)}, \ldots, \phi_d^{(j-1)}, x \right.\right) \\
\phi_2^{(j)} &\sim \pi\left(\phi_1 \left| \phi_1^{(j)}, \phi_3^{(j-1)} \ldots, \phi_d^{(j-1)}, x \right.\right) \\
&\vdots \\
\phi_d^{(j)} &\sim \pi\left(\phi_1 \left| \phi_1^{(j)}, \ldots, \phi_{d-1}^{(j)}, x \right.\right)
\end{aligned}
$$

3. Increase counter to $j + 1$ and repeat step 2.

This algorithm was utilised by Geman & Geman (1984) for models with the Gibbs distribution and then extended to this general form by Gelfand & Smith (1990).

4.2.2 Full conditionals for the lattice DLM

In order to use Gibbs sampling for the lattice DLM model the full conditionals are required. These are obtained by specifying semi-conjugate prior distributions for the parameters. Consider the 1+1D model with known temporal dependence α and mean μ and unknown precisions τ_s and τ_o. The model has T time periods $(t = 0, 1, \ldots, T-1)$ and n spatial nodes $(i = 0, 1, \ldots, n-1)$.

The DLM for this model with isotropic binomial edge weights is given by

$$\theta_i^{(t+1)} \left| \text{pa}\left(\theta_i^{(t+1)}\right) \right. \sim N\left(g\,\text{pa}\left(\theta_i^{(t+1)}\right), \tau_s^{-1}\right) \tag{4.2}$$

$$X^{(t)} \left| \theta^{(t)} \right. \sim N\left(F\theta^{(t)}, \tau_o^{-1}\right). \tag{4.3}$$

The evolution matrix g for a given node is defined as in Equation (2.5) where $g' = \alpha\left(\frac{1}{4}, \frac{1}{2}, \frac{1}{4}\right)$ and F is the design matrix that maps observations to the underlying latent structure. In this example the unadjusted edge weight model from Chapter 3 is used thus nodes which lie along the edge of the system are influenced only by the parent nodes which exist with no adjustments for the reduction of information available.

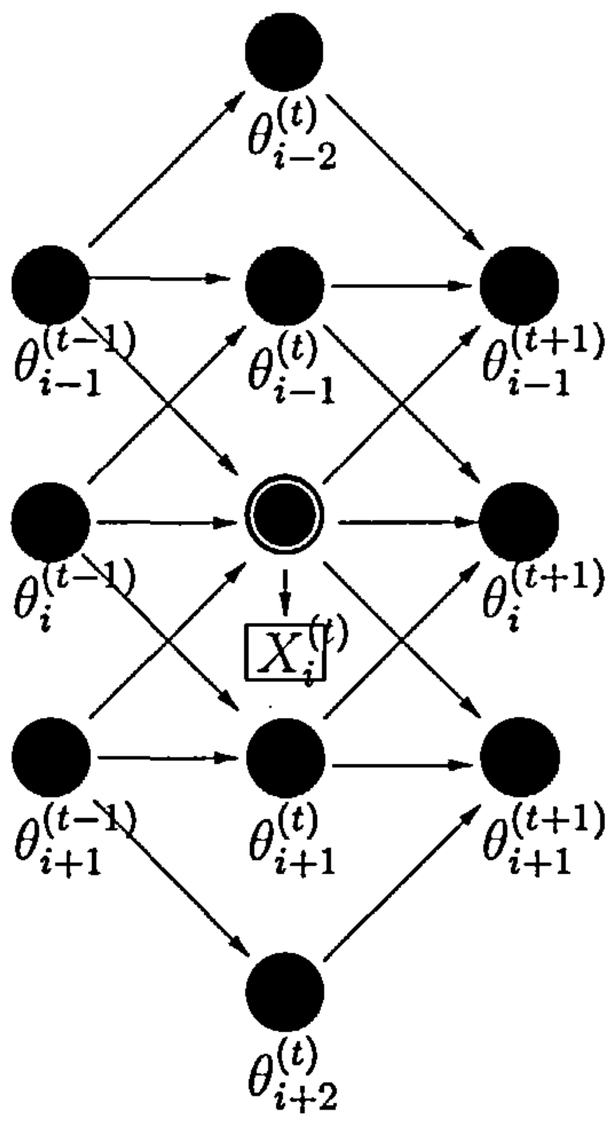


Figure 4.1: *The evolution of the 1+1D model showing both forward and backward dependencies for the central node* $\theta_i^{(t)}$.

Semi-conjugate prior distributions for this model are given by,

$$\tau_s \sim \Gamma(a_s, b_s), \tag{4.4}$$

$$\tau_o \sim \Gamma(a_o, b_o), \tag{4.5}$$

$$\theta_i^{(t)} \left| \text{pa}\left(\theta_i^{(t)}\right), \tau_s \right. \sim N\left(\theta_i^{(t)}; g\,\text{pa}\left(\theta_i^{(t)}\right), \tau_s^{-1}\right). \tag{4.6}$$

The likelihood $L(\theta; X, \tau_o)$ for this model where $\theta = \left(\theta^{(0)}, \theta^{(1)}, \ldots, \theta^{(t-1)}\right)$ and

$X = \left(X^{(0)}, X^{(1)}, \ldots, X^{(t-1)}\right)$ is given by

$$\begin{aligned} L\left(\theta; X, \tau_o\right) &= \prod_{t=0}^{T-1}\prod_{i=0}^{n-1}\left(\frac{\tau_o}{2\pi}\right)^{\frac{1}{2}} \exp\left[-\frac{1}{2}\tau_o\left(X_i^{(t)} - \theta_i^{(t)}\right)^2\right] \\ &\propto \tau_o^{N/2} \exp\left[-\frac{\tau_o}{2}\sum_{t=0}^{T-1}\sum_{i=0}^{n-1}\left(X_i^{(t)} - \theta_i^{(t)}\right)^2\right]. \end{aligned} \tag{4.7}$$

Thus the posterior distribution is given by

$$\begin{aligned} \pi\left(\theta_i^{(t)}, \tau_o, \tau_s \middle| X\right) \propto\ & \tau_o^{N/2} \exp\left[-\frac{\tau_o}{2}\sum_{t=0}^{T-1}\sum_{i=0}^{n-1}\left(X_i^{(t)} - \theta_i^{(t)}\right)^2\right] \\ & \times \tau_s^{N/2} \exp\left\{-\frac{\tau_s}{2}\sum_{t=0}^{T-1}\sum_{i=0}^{n-1}\left[\theta_i^{(t)} - g\,\mathrm{pa}\left(\theta_i^{(t)}\right)\right]^2\right\} \\ & \times \tau_s^{a_s-1}\tau_o^{a_o-1} \exp\left(-b_s\tau_s\right) \exp\left(-b_o\tau_o\right). \end{aligned} \tag{4.8}$$

From the posterior distribution the full conditionals for the precision parameters can be obtained and are given by

$$\begin{aligned} \tau_o \ :\ & \tau_o^{N/2+a_o-1} \exp\left\{-\left[b_o + \frac{\tau_o}{2}\sum_{t=0}^{T-1}\sum_{i=0}^{n-1}\left(\theta_i^{(t)} - X_i^{(t)}\right)^2\right]\right\} \\ & \sim \Gamma\left(\frac{N}{2} + a_o, b_o + \frac{1}{2}\sum_{t=0}^{T-1}\sum_{i=0}^{n-1}\left(\theta_i^{(t)} - X_i^{(t)}\right)^2\right), \end{aligned} \tag{4.9}$$

$$\begin{aligned} \tau_s \ :\ & \tau_s^{N/2+a_s-1} \exp\left(-\left\{b_s + \frac{\tau_s}{2}\sum_{t=0}^{T-1}\sum_{i=0}^{n-1}\left[\theta_i^{(t)} - g\,\mathrm{pa}\left(\theta_i^{(t)}\right)\right]^2\right\}\right) \\ & \sim \Gamma\left(\frac{N}{2} + a_s, b_s + \frac{1}{2}\sum_{t=0}^{T-1}\sum_{i=0}^{n-1}\left[\theta_i^{(t)} - g\,\mathrm{pa}\left(\theta_i^{(t)}\right)\right]^2\right). \end{aligned} \tag{4.10}$$

Figure 4.1 depicts the conditional independence graph for a given node $\theta_i^{(t)}$ and its "Markov blanket". The full conditional for any node depends on its parents, children and co-parents. This full conditional can be obtained by taking the nodes conditional density

and multiplying it by the conditional densities of its children:

$$\begin{aligned}
\pi\left(\theta_i^{(t)}|\cdot\right) &\sim N\left(X_i^{(t)};\theta_i^{(t)},\tau_o\right) N\left(\theta_i^{(t)};g\,\mathrm{pa}\left(\theta_i^{(t)}\right),\tau_s^{-1}\right) N\left(\theta_i^{(t+1)};g\,\mathrm{pa}\left(\theta_i^{(t+1)}\right),\tau_s^{-1}\right) \\
&\quad \times N\left(\theta_{i+1}^{(t+1)};g\,\mathrm{pa}\left(\theta_{i+1}^{(t+1)}\right),\tau_s^{-1}\right) N\left(\theta_{i-1}^{(t+1)};g\,\mathrm{pa}\left(\theta_{i-1}^{(t+1)}\right),\tau_s^{-1}\right). \\
&\propto \tau_o^{1/2}\exp\left[-\frac{\tau_o}{2}\left(X_i^{(t)}-\theta_i^{(t)}\right)^2\right]\tau_s^{1/2}\exp\left\{-\frac{\tau_s}{2}\left[\theta_i^{(t)}-g\,\mathrm{pa}\left(\theta_i^{(t)}\right)\right]^2\right\} \\
&\quad \times\tau_s^{1/2}\exp\left\{-\frac{\tau_s}{2}\left[\theta_{i-1}^{(t+1)}-g\,\mathrm{pa}\left(\theta^{(t+1)}\right)\right]^2\right\} \\
&\quad \times\tau_s^{1/2}\exp\left\{-\frac{\tau_s}{2}\left[\theta_i^{(t+1)}-g\,\mathrm{pa}\left(\theta_i^{(t+1)}\right)\right]^2\right\} \\
&\quad \times\tau_s^{1/2}\exp\left\{-\frac{\tau_s}{2}\left[\theta_{i+1}^{(t+1)}-g\,\mathrm{pa}\left(\theta_{i+1}^{(t+1)}\right)\right]^2\right\}.
\end{aligned}$$

Interest is in $\theta_i^{(t)}$ which is embedded in the last three terms of Equation (4.11). Firstly expand these terms to obtain an expression involving $\theta_i^{(t)}$ then substitute into Equation (4.11) to obtain the full conditional for $\theta_i^{(t)}$

$$\begin{aligned}
&N\left(\theta_i^{(t+1)};g\,\mathrm{pa}\left(\theta_i^{(t+1)}\right),\tau_s^{-1}\right) N\left(\theta_{i+1}^{(t+1)};g\,\mathrm{pa}\left(\theta_{i+1}^{(t+1)}\right),\tau_s^{-1}\right) N\left(\theta_{i-1}^{(t+1)};g\,\mathrm{pa}\left(\theta_{i-1}^{(t+1)}\right),\tau_s^{-1}\right) \\
&\propto \exp\left(-\frac{\tau_s}{2}\left\{\left[\theta_i^{(t+1)}-g\,\mathrm{pa}\left(\theta_i^{(t+1)}\right)\right]^2+\left[\theta_{i+1}^{(t+1)}-g\,\mathrm{pa}\left(\theta_{i+1}^{(t+1)}\right)\right]^2\right.\right. \\
&\qquad \left.\left.+\left[\theta_{i-1}^{(t+1)}-g\,\mathrm{pa}\left(\theta_{i-1}^{(t+1)}\right)\right]^2\right\}\right) \\
&\propto \exp\left\{-\frac{\tau_s}{2}\left[\left(\left\{\theta_i^{(t+1)^2}-2\theta_i^{(t+1)}\left[\frac{\alpha}{4}\left(\theta_{i-1}^{(t)}+\theta_{i+1}^{(t)}\right)+\frac{\alpha}{2}\theta_i^{(t)}\right]+\left[\frac{\alpha}{4}\left(\theta_{i-1}^{(t)}\theta_{i+1}^{(t)}\right)+\frac{\alpha}{2}\theta_i^{(t)}\right]^2\right\}\right)\right.\right. \\
&\qquad +\left(\left\{\theta_{i+1}^{(t+1)^2}-2\theta_{i+1}^{(t+1)}\left[\frac{\alpha}{4}\left(\theta_i^{(t)}+\theta_{i+2}^{(t)}\right)+\frac{\alpha}{2}\theta_{i+1}^{(t)}\right]+\left[\frac{\alpha}{4}\left(\theta_i^{(t)}\theta_{i+2}^{(t)}\right)+\frac{\alpha}{2}\theta_{i+1}^{(t)}\right]^2\right\}\right) \\
&\qquad \left.\left.+\left(\left\{\theta_{i-1}^{(t+1)^2}-2\theta_{i-1}^{(t+1)}\left[\frac{\alpha}{4}\left(\theta_{i-2}^{(t)}+\theta_i^{(t)}\right)+\frac{\alpha}{2}\theta_{i-1}^{(t)}\right]+\left[\frac{\alpha}{4}\left(\theta_{i-2}^{(t)}\theta_i^{(t)}\right)+\frac{\alpha}{2}\theta_{i-1}^{(t)}\right]^2\right\}\right)\right]\right\}.
\end{aligned}$$

Dropping the constant terms and expanding the end term for each part of the equation yields

$$\begin{aligned}
&\propto \exp\left(-\frac{\tau_s}{2}\left\{\frac{\alpha^2}{4}\theta_i^{(t)^2}+\frac{2\alpha}{2}\theta_i^{(t)}\left[\frac{\alpha}{4}\left(\theta_{i-1}^{(t)}+\theta_{i+1}^{(t)}\right)\right]-\frac{2\alpha}{2}\theta_i^{(t)}\theta_i^{(t+1)}\right.\right. \\
&\qquad +\frac{\alpha^2}{16}\theta_i^{(t)^2}+\frac{2\alpha}{4}\theta_i^{(t)}\left(\frac{\alpha}{2}\theta_{i+1}^{(t)}+\frac{\alpha}{4}\theta_{i+2}^{(t)}\right)-\frac{2\alpha}{4}\theta_i^{(t)}\theta_{i+1}^{(t+1)} \\
&\qquad \left.\left.+\frac{\alpha^2}{16}\theta_i^{(t)^2}+\frac{2\alpha}{4}\theta_i^{(t)}\left(\frac{\alpha}{2}\theta_{i-1}^{(t)}+\frac{\alpha}{4}\theta_{i-2}^{(t)}\right)-\frac{2\alpha}{4}\theta_i^{(t)}\theta_{i-1}^{(t+1)}\right\}\right) \\
&\propto \exp\left[-\frac{\tau_s}{2}\left(\frac{3\alpha^2}{8}\theta_i^{(t)^2}-2\theta_i^{(t)}\left\{g\,\mathrm{pa}\left(\theta_i^{(t+2)}\right)-\frac{\alpha^2}{4}\left[\theta_{i-1}^{(t)}+\theta_{i+1}^{(t)}+\frac{1}{4}\left(\theta_{i-2}^{(t)}+\theta_{i+2}^{(t)}\right)\right]\right\}\right)\right].
\end{aligned}$$

Expanding the rest of the full conditional and substituting back in gives

$$\begin{aligned}
\pi\left(\theta_i^{(t)}|\cdot\right) &\propto \left\{-\frac{1}{2}\left[\tau_s\left[\theta_i^{(t)2} - 2\theta_i^{(t)} g\,\mathrm{pa}\left(\theta_i^{(t+1)}\right)\right] + \tau_o\left(\theta_i^{(t)2} - 2\theta_i^{(t)} X_i^{(t)}\right) + \tau_s\left(\frac{3\alpha^2}{8}\theta_i^{(t)2}\right.\right.\right. \\
&\qquad \left.\left.\left. -2\theta_i^{(t)}\left\{g\left(\mathrm{pa}\,\theta_i^{(t+2)}\right) - \frac{\alpha^2}{4}\left[\theta_{i-1}^{(t)} + \theta_{i+1}^{(t)} + \frac{1}{4}\left(\theta_{i-2}^{(t)} + \theta_{i+2}^{(t)}\right)\right]\right\}\right)\right]\right\} \\
&\propto exp\left\{-\frac{1}{2}\left[\theta_i^{(t)2}\left(\tau_s\left[1+\frac{3\alpha^2}{8}\right] + \tau_o\right) - 2\theta_i^{(t)}\left(\tau_o X_i^{(t)} + \tau_s\left\{g\,\mathrm{pa}\left(\theta_i^{(t)}\right)\right.\right.\right. \\
&\qquad \left.\left.\left. +g\,\mathrm{pa}\left(\theta_i^{(t+2)}\right) - \frac{\alpha^2}{4}\left[\theta_{i-1}^{(t)} + \theta_{i+1}^{(t)} + \frac{1}{4}\left(\theta_{i-2}^{(t)} + \theta_{i+2}^{(t)}\right)\right]\right\}\right)\right]\right\} \\
&\sim N\left(\frac{\tau_o X_i^{(t)} + \tau_s\left\{g\,\mathrm{pa}\left(\theta_i^{(t)}\right) + g\,\mathrm{pa}\left(\theta_i^{(t+2)}\right) - \frac{\alpha^2}{4}\left[\theta_{i-1}^{(t)} + \theta_{i+1}^{(t)} + \frac{1}{4}\left(\theta_{i-2}^{(t)} + \theta_{i+2}^{(t)}\right)\right]\right\}}{\tau_s\left(1+\frac{3\alpha^2}{8}\right) + \tau_o}\right. \\
&\qquad \left., \left[\tau_s\left(1+\frac{3\alpha^2}{8}\right) + \tau_o\right]^{-1}\right). \qquad (4.11)
\end{aligned}$$

This expression is only valid for internal nodes, for nodes which lie along the edge of the system the full conditional needs to take into account the reduction in the number of parent nodes that are available. Thus the full conditional needs to be calculated separately along the boundaries of the system, taking into account the changed structure at both the spatial and temporal boundary. In each case nodes which do not exist are dropped from the expression. The boundaries which need to be considered are the initial and final time point, spatial locations 0, 1, $n-2$ and $n-1$ as well as the spatial boundaries at time $t = 0$

Time $t = 0$

$$\begin{aligned}
\pi\left(\theta_i^{(0)}|\cdot\right) &\sim N\left(X_i^{(0)}; \theta_i^{(0)}, \tau_o\right) N\left(\theta_i^{(0)}; 0, \tau_s^{-1}\right) N\left(\theta_i^{(1)}; g\,\mathrm{pa}\left(\theta_i^{(1)}\right), \tau_s^{-1}\right) \\
&\quad \times N\left(\theta_{i+1}^{(1)}; g\,\mathrm{pa}\left(\theta_{i+1}^{(1)}\right), \tau_s^{-1}\right) N\left(\theta_{i-1}^{(1)}; g\,\mathrm{pa}\left(\theta_{i-1}^{(1)}\right), \tau_s^{-1}\right) \\
&\propto \exp\left[-\frac{1}{2}\left(\tau_s\left\{\theta_i^{(0)2} + \left[\theta_i^{(1)} - g\,\mathrm{pa}\left(\theta_i^{(1)}\right)\right]^2 + \left[\theta_{i+1}^{(1)} - g\,\mathrm{pa}\left(\theta_{i+1}^{(1)}\right)\right]^2\right.\right.\right. \\
&\qquad \left.\left.\left. + \left[\theta_{i-1}^{(1)} - g\,\mathrm{pa}\left(\theta_{i-1}^{(1)}\right)\right]^2\right\} + \tau_o\left[X_i^{(0)} - \theta_i^{(0)}\right]\right)\right].
\end{aligned}$$

Expanding as before yields

$$\sim N\left(\frac{\tau_o X_i^{(0)} + \tau_s\left\{g\,\mathrm{pa}\left(\theta_i^{(2)}\right) - \frac{\alpha^2}{4}\left[\theta_{i-1}^{(0)} + \theta_{i+1}^{(0)} + \frac{1}{4}\left(\theta_{i-2}^{(0)} + \theta_{i+2}^{(0)}\right)\right]\right\}}{\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o}\right.$$

$$\left., \left[\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o\right]^{-1}\right). \tag{4.12}$$

The spatial boundaries also effect the distribution at $t = 0$, since the model is isotropic the derivation at location $n-1$ at time 0 is easily obtained from the derivation at spatial location 0 at time 0 similarly the derivation of the distribution at spatial location $n-2$ is easily obtained from the derivation at spatial location 1.

Time $t = 0$, spatial location $i = 0$

$$\begin{aligned}
\pi\left(\theta_0^{(0)}\,|\cdot\right) &\sim N\left(X_0^{(0)};\theta_0^{(0)},\tau_o\right) N\left(\theta_0^{(0)};0,\tau_s^{-1}\right) N\left(\theta_0^{(1)};g\,\mathrm{pa}\left(\theta_0^{(1)}\right),\tau_s^{-1}\right) \\
&\quad \times N\left(\theta_1^{(1)};g\,\mathrm{pa}\left(\theta_1^{(1)}\right),\tau_s^{-1}\right) \\
&\sim N\left(\frac{\tau_o X_0^{(0)} + \tau_s\left[g\,\mathrm{pa}\left(\theta_0^{(2)}\right) - \frac{\alpha^2}{4}\left(\theta_1^{(0)} + \frac{1}{4}\theta_2^{(0)}\right)\right]}{\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o}\right. \\
&\qquad \left., \left[\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o\right]^{-1}\right).
\end{aligned} \tag{4.13}$$

Time $t = 0$, spatial location $i = n - 1$

$$\begin{aligned}
\pi\left(\theta_{n-1}^{(0)}|\cdot\right) &\sim N\left(X_{n-1}^{(0)};\theta_{n-1}^{(0)},\tau_o\right) N\left(\theta_{n-1}^{(0)};0,\tau_s^{-1}\right) N\left(\theta_{n-1}^{(1)};g\,\mathrm{pa}\left(\theta_{n-1}^{(1)}\right),\tau_s^{-1}\right) . \\
&\quad \times N\left(\theta_{n-2}^{(1)};g\,\mathrm{pa}\left(\theta_{n-2}^{(1)}\right),\tau_s^{-1}\right) \\
&\sim N\left(\frac{\tau_o X_{n-1}^{(0)} + \tau_s\left[g\,\mathrm{pa}\left(\theta_{n-1}^{(2)}\right) - \frac{\alpha^2}{4}\left(\theta_{n-2}^{(0)} + \frac{1}{4}\theta_{n-3}^{(0)}\right)\right]}{\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o}\right. \\
&\qquad \left., \left[\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o\right]^{-1}\right) . \qquad (4.14)
\end{aligned}$$

Time $t = 0$, spatial location $i = 1$

$$\begin{aligned}
\pi\left(\theta_1^{(0)}|\cdot\right) &\sim N\left(X_1^{(0)};\theta_1^{(0)},\tau_o\right) N\left(\theta_1^{(0)};0,\tau_s^{-1}\right) N\left(\theta_1^{(1)};g\,\mathrm{pa}\left(\theta_1^{(1)}\right),\tau_s^{-1}\right) \\
&\quad \times N\left(\theta_2^{(1)};g\,\mathrm{pa}\left(\theta_2^{(1)}\right),\tau_s^{-1}\right) N\left(\theta_0^{(1)};g\,\mathrm{pa}\left(\theta_0^{(1)}\right),\tau_s^{-1}\right) \\
&\sim N\left(\frac{\tau_o X_1^{(0)} + \tau_s\left\{g\,\mathrm{pa}\left(\theta_1^{(2)}\right) - \frac{\alpha^2}{4}\left[\theta_0^{(0)} + \theta_2^{(0)} + \frac{1}{4}\left(\theta_3^{(0)}\right)\right]\right\}}{\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o}\right. \\
&\qquad \left., \left[\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o\right]^{-1}\right) . \qquad (4.15)
\end{aligned}$$

Time $t = 0$, spatial location $i = n - 2$

$$\begin{aligned}\pi\left(\theta_{n-2}^{(0)}|\cdot\right) &\sim N\left(X_{n-2}^{(0)};\theta_{n-2}^{(0)},\tau_o\right)N\left(\theta_{n-2}^{(0)};0,\tau_s^{-1}\right)N\left(\theta_{n-2}^{(1)};g\,\mathrm{pa}\left(\theta_{n-2}^{(1)}\right),\tau_s^{-1}\right)\\ &\quad\times N\left(\theta_{n-1}^{(1)};g\,\mathrm{pa}\left(\theta_{n-1}^{(1)}\right),\tau_s^{-1}\right)N\left(\theta_{n-3}^{(1)};g\,\mathrm{pa}\left(\theta_{n-3}^{(1)}\right),\tau_s^{-1}\right)\\ &\sim N\left(\frac{\tau_o X_{n-2}^{(0)}+\tau_s\left\{g\,\mathrm{pa}\left(\theta_{n-2}^{(2)}\right)-\frac{\alpha^2}{4}\left[\theta_{n-3}^{(0)}+\theta_{n-1}^{(0)}+\frac{1}{4}\left(\theta_{n-4}^{(0)}\right)\right]\right\}}{\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o}\right.\\ &\quad\left.,\left[\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o\right]^{-1}\right).\end{aligned}\tag{4.16}$$

Spatial boundary

As above once the distribution for spatial location 0 has been determined the distribution for spatial location $n-1$ is easily obtained, similarly once the distribution for spatial location 1 has been determined the distribution for spatial location $n-2$ is easily obtained. The full conditional distributions for nodes along the spatial boundary for time periods $t > 0$ are given below.

Spatial location $i = 0$

$$\begin{aligned}\pi\left(\theta_0^{(t)}|\cdot\right) &\sim N\left(X_0^{(t)};\theta_0^{(t)},\tau_o\right)N\left(\theta_0^{(t)};g\,\mathrm{pa}\left(\theta_0^{(t)}\right),\tau_s^{-1}\right)N\left(\theta_0^{(t+1)};g\,\mathrm{pa}\left(\theta_0^{(t+1)}\right),\tau_s^{-1}\right)\\ &\quad\times N\left(\theta_1^{(t+1)};g\,\mathrm{pa}\left(\theta_1^{(t+1)}\right),\tau_s^{-1}\right)\\ &\sim N\left(\frac{\tau_o X_0^{(t)}+\tau_s\left[g\,\mathrm{pa}\left(\theta_0^{(t)}\right)+g\,\mathrm{pa}\left(\theta_0^{(t+2)}\right)-\frac{\alpha^2}{4}\left(\theta_1^{(t)}+\frac{1}{4}\theta_2^{(t)}\right)\right]}{\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o}\right.\\ &\quad\left.,\left[\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o\right]^{-1}\right).\end{aligned}\tag{4.17}$$

Spatial location $i = n - 1$

$$\begin{aligned}
\pi\left(\theta_{n-1}^{(t)}|\cdot\right) &\sim N\left(X_{n-1}^{(t)};\theta_{n-1}^{(t)},\tau_o\right) N\left(\theta_{n-1}^{(t)};g\,\mathrm{pa}\left(\theta_{n-1}^{(t)}\right),\tau_s^{-1}\right) N\left(\theta_{n-1}^{(t+1)};g\,\mathrm{pa}\left(\theta_{n-1}^{(t+1)}\right),\tau_s^{-1}\right) \\
&\quad \times N\left(\theta_{n-2}^{(t+1)};g\,\mathrm{pa}\left(\theta_{n-2}^{(t+1)}\right),\tau_s^{-1}\right) \\
&\sim N\left(\frac{\tau_o X_{n-1}^{(t)}+\tau_s\left[g\,\mathrm{pa}\left(\theta_{n-1}^{(t)}\right)+g\,\mathrm{pa}\left(\theta_{n-1}^{(t+2)}\right)-\frac{\alpha^2}{4}\left(\theta_{n-2}^{(t)}+\frac{1}{4}\theta_{n-3}^{(t)}\right)\right]}{\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o}\right. \\
&\qquad \left., \left[\tau_s\left(1+\frac{5\alpha^2}{16}\right)+\tau_o\right]^{-1}\right). \qquad (4.18)
\end{aligned}$$

Spatial location $i = 1$

$$\begin{aligned}
\pi\left(\theta_1^{(t)}|\cdot\right) &\sim N\left(X_1^{(t)};\theta_1^{(t)},\tau_o\right) N\left(\theta_1^{(t)};g\,\mathrm{pa}\left(\theta_1^{(t)}\right),\tau_s^{-1}\right) N\left(\theta_1^{(t+1)};g\,\mathrm{pa}\left(\theta_1^{(t+1)}\right),\tau_s^{-1}\right) \\
&\quad \times N\left(\theta_2^{(t+1)};g\,\mathrm{pa}\left(\theta_2^{(t+1)}\right),\tau_s^{-1}\right) N\left(\theta_0^{(t+1)};g\,\mathrm{pa}\left(\theta_0^{(t+1)}\right),\tau_s^{-1}\right). \\
&\sim N\left(\frac{\tau_o X_1^{(t)}+\tau_s\left\{g\,\mathrm{pa}\left(\theta_1^{(t)}\right)+g\,\mathrm{pa}\left(\theta_1^{(t+2)}\right)-\frac{\alpha^2}{4}\left[\theta_0^{(t)}+\theta_2^{(t)}+\frac{1}{4}\left(\theta_3^{(t)}\right)\right]\right\}}{\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o}\right. \\
&\qquad \left., \left[\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o\right]^{-1}\right). \qquad (4.19)
\end{aligned}$$

Spatial location $i = n - 2$

$$\begin{aligned}
\pi\left(\theta_{n-2}^{(t)}|\cdot\right) &\sim N\left(X_{n-2}^{(t)};\theta_{n-2}^{(t)},\tau_o\right) N\left(\theta_{n-2}^{(t)};g\,\mathrm{pa}\left(\theta_{n-2}^{(t)}\right),\tau_s^{-1}\right) N\left(\theta_{n-2}^{(t+1)};g\,\mathrm{pa}\left(\theta_{n-2}^{(t+1)}\right),\tau_s^{-1}\right) \\
&\quad \times N\left(\theta_{n-1}^{(t+1)};g\,\mathrm{pa}\left(\theta_{n-1}^{(t+1)}\right),\tau_s^{-1}\right) N\left(\theta_{n-3}^{(t+1)};g\,\mathrm{pa}\left(\theta_{n-3}^{(t+1)}\right),\tau_s^{-1}\right). \\
&\sim N\left(\frac{\tau_o X_{n-2}^{(t)}+\tau_s\left\{g\,\mathrm{pa}\left(\theta_{n-2}^{(t)}\right)+g\,\mathrm{pa}\left(\theta_{n-2}^{(t+2)}\right)-\frac{\alpha^2}{4}\left[\theta_{n-3}^{(t)}+\theta_{n-1}^{(t)}+\frac{1}{4}\left(\theta_{n-3}^{(t)}\right)\right]\right\}}{\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o}\right. \\
&\qquad \left., \left[\tau_s\left(1+\frac{3\alpha^2}{8}\right)+\tau_o\right]^{-1}\right). \qquad (4.20)
\end{aligned}$$

The final time period

At time $t-1$ the distribution is the same for all spatial locations and is given by,

$$\pi\left(\theta_i^{(t-1)}|\cdot\right) \sim N\left(X_i^{(t-1)};\theta_i^{(t-1)},\tau_o\right) N\left(\theta_i^{(t-1)};g\,\mathrm{pa}\left(\theta_i^{(t-1)}\right),\tau_s^{-1}\right) \tag{4.21}$$

$$\sim N\left(\frac{\tau_o X_i^{(t-1)}+\tau_s g\,\mathrm{pa}\left(\theta_i^{(t-1)}\right)}{\tau_s+\tau_o},(\tau_s+\tau_o)^{-1}\right). \tag{4.22}$$

Now that the full conditionals have been determined for the entire model the Gibbs sampler can be applied. In each iteration the parameters are simulated in turn. In this 1+1D model the parameters which need to be simulated are $\phi = \left(\tau_o,\tau_s,\theta_0^{(0)},\ldots,\theta_{n-1}^{(0)},\theta_0^{(1)},\ldots,\theta_{n-1}^{(T-1)}\right)$. These parameters are simulated using the algorithm above and the appropriate full conditional for updating the latent nodes.

4.2.3 Implementation

The implementation of the above Gibbs sampler is done using two approaches, using the BUGS package (Speigalhalter *et al.* 1996) and by writing the algorithm in 'C' . The BUGS package generates realisations from the posterior distribution. It builds the required full conditionals from the specification of the model, prior distributions, and data which are provided by the operator, once it has determined the full conditional it uses the Gibbs algorithm to generate realisations from the marginal posterior distributions.

The operator has little control over the order in which the conditional distributions are sampled in BUGS and by writing the corresponding program in 'C' control in this ordering is possible. Using 'C' the full conditionals have to take account the boundary conditions in the model as the BUGS program does. Both approaches were used in the following model on the same set of data so that they could be compared.

4.2.4 Example

Consider a 1+1D model which is defined by Equation 2.1 with generated data. The data is generated from a model with mean $\mu = 0$, state precision $\tau_s = 0.2$, observation

precision $\tau_o = 0.1$ and temporal dependence parameter $\alpha = 0.7$ on 100 spatial nodes evolving over 20 time periods. The Gibbs sampler is used to simulate the two precisions from the model with known mean and temporal dependence parameter. The unadjusted edge weight model from Chapter 3 is used in this example. The results for this simulation are presented from both the BUGS code (Figure 4.2 and Tables 4.1 and 4.2) and the 'C' code (Figure 4.3 and Tables 4.3 and 4.4), for both methods the MCMC chains have 1000 iterations as burn-in which has been removed and the plots are for the following 20,000 iterations. The posterior distribution for the model generated by BUGS suggests $\tau_s = 0.17$ and $\tau_o = 0.11$ whereas the posterior distribution for the model generated by the 'C' code suggests $\tau_s = 0.18$ and $\tau_o = 0.11$. Both methods give estimates of τ_s which are slightly lower than the true value of 0.2 and both estimates of τ_o are slightly higher than the true value of 0.1.

	Mean	SD	Naive SE	Time-series SE
τ_s	0.1716	0.02608	1.844e-04	0.0003101
τ_o	0.1116	0.01170	8.273e-05	0.0001426

Table 4.1: *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in BUGS.*

	2.5%	25%	50%	75%	97.5%
τ_s	0.12886	0.1538	0.1685	0.1858	0.2328
τ_o	0.09168	0.1036	0.1105	0.1183	0.1380

Table 4.2: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in BUGS.*

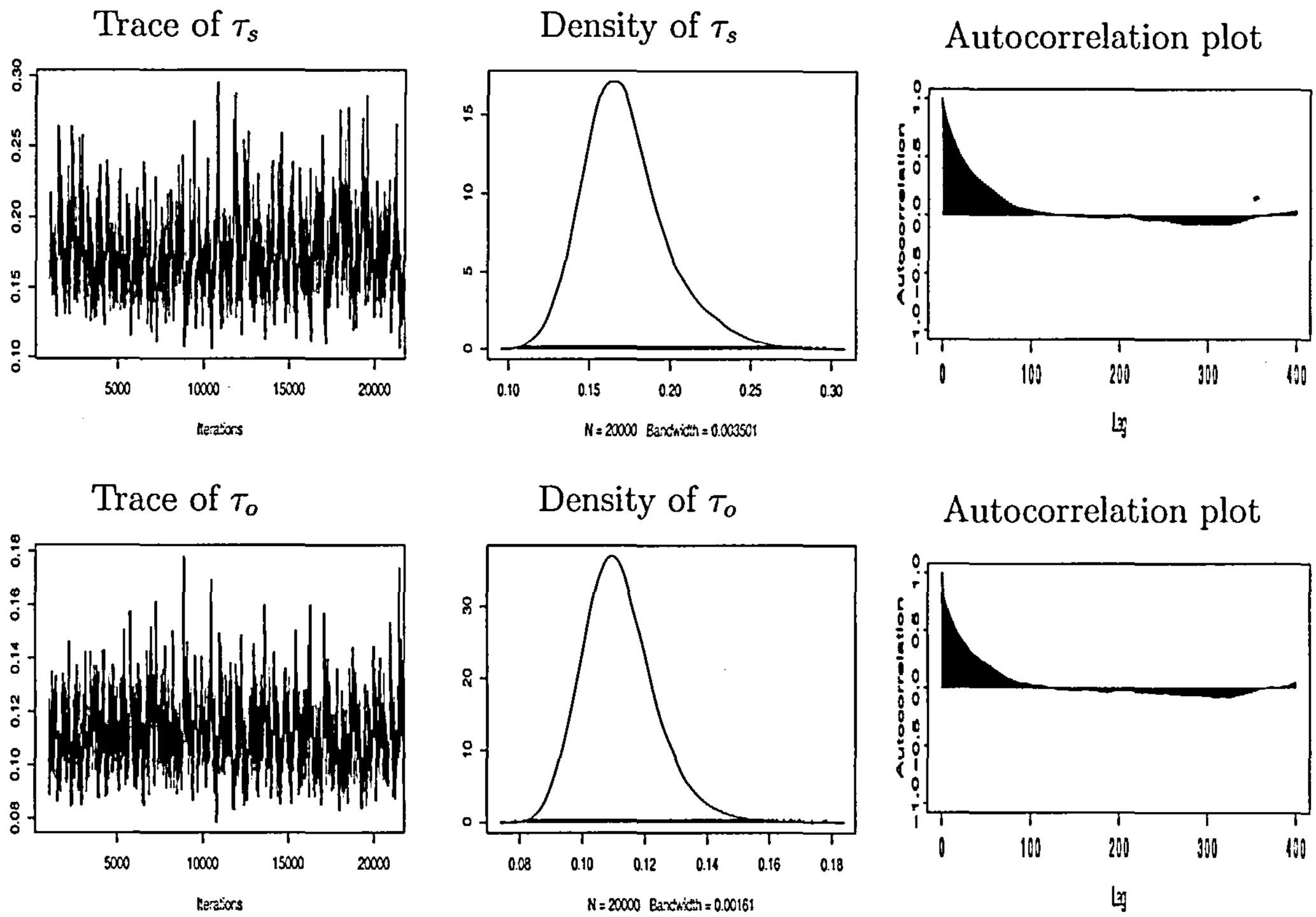


Figure 4.2: *The trace, density and auto correlation plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the Gibbs sampler implemented in BUGS with 20,000 iterations.*

	Mean	SD	Naive SE	Time-series SE
τ_s	0.1802	0.02798	0.0001978	0.0003510
τ_o	0.1072	0.01049	0.0000742	0.0001275

Table 4.3: *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in 'C' .*

4.2.5 Discussion

Examination of the results from the two Gibbs sampler approaches suggests that the 'C' program is simulating from the correct posterior distribution whilst providing more flexibility than the Gibbs sampler implemented in BUGS. The convergence of these chains

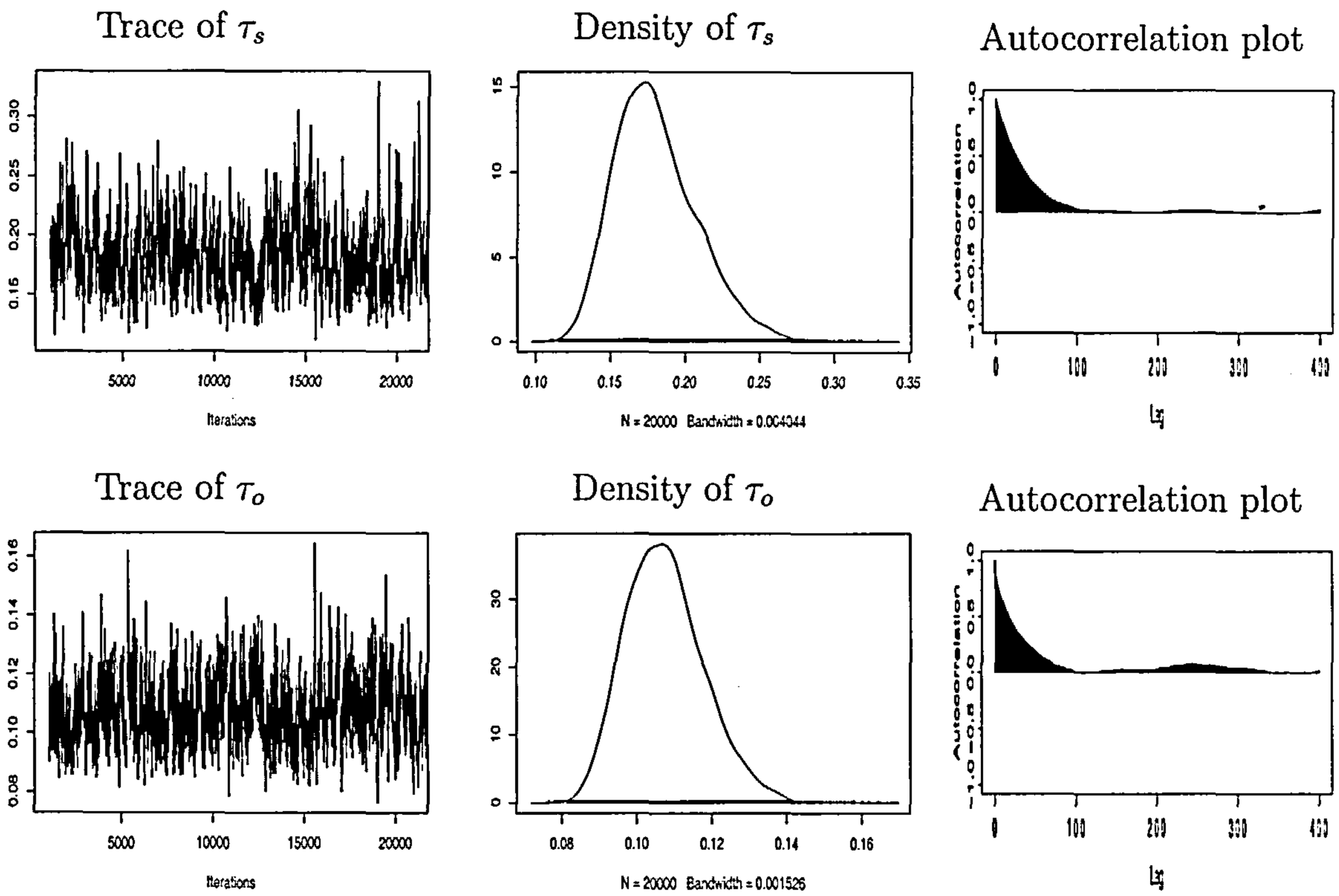


Figure 4.3: *The trace, density and auto correlation plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the Gibbs sampler implemented in 'C' with 20,000 iterations.*

	2.5%	25%	50%	75%	97.5%
τ_s	0.1350	0.16009	0.1767	0.1971	0.2430
τ_o	0.0893	0.09965	0.1064	0.1136	0.1301

Table 4.4: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Gibbs sampler implemented in 'C' .*

can be examined using the CODA library in R. The convergence diagnostics used are the Geweke convergence diagnostic, Rafferty and Lewis convergence diagnostic and Heidelberger and Welch stationarity and interval halfwidth test. The Geweke diagnostic tests for stationarity of chains by checking for a constant mean and variance along the entire

length of the chain where large $|Z|$ scores represent a lack of convergence whilst small $|Z|$ indicate convergence may have been reached. The Rafferty and Lewis diagnostic estimates the length of the required burn-in and the minimum number of iterations required N. The efficiency of the chain can be assessed by considering the dependence statistic I, values of $I > 5$ indicate convergence problems. The Heidelberger and Welch convergence diagnostic tests the stationarity of the chain. The convergence diagnostics for the Gibbs sampler implemented in BUGS are:

```
GEWEKE CONVERGENCE DIAGNOSTIC (Z-score)
=======================================
Iterations used = 1001:21000
Thinning interval = 1
Sample size per chain = 20000
Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5
 taus   tauo
 4.542 -6.128

RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC
========================================
Iterations used = 1001:21000
Thinning interval = 1
Sample size per chain = 20000
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95
      Burn-in  Total Lower bound  Dependence
      (M)      (N)   (Nmin)       factor (I)
 taus 64       75664 3746         20.2
 tauo 49       48314 3746         12.9

HEIDELBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS
===============================================================
Iterations used = 1001:21000
Thinning interval = 1
Sample size per chain = 20000
Precision of halfwidth test = 0.1
     Stationarity start     p-value
     test         iteration
taus failed       NA        0.000642
tauo failed       NA        0.001501

     Halfwidth Mean Halfwidth
     test
taus <NA>      NA   NA
tauo <NA>      NA   NA
```

The corresponding diagnostics for the Gibbs algorithm using the 'C' program are given below:

```
GEWEKE CONVERGENCE DIAGNOSTIC (Z-score)
=======================================
Iterations used = 1000:20999
Thinning interval = 1
Sample size per chain = 20000
Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5
 tau_s  tau_o
 7.849 -9.000

RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC
========================================
Iterations used = 1000:20999
Thinning interval = 1
Sample size per chain = 20000
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95
       Burn-in  Total Lower bound  Dependence
       (M)      (N)   (Nmin)       factor (I)
 tau_s 35       34560 3746          9.23
 tau_o 32       39352 3746         10.50

HEIDELBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS
===============================================================
Iterations used = 1000:20999
Thinning interval = 1
Sample size per chain = 20000
Precision of halfwidth test = 0.1
      Stationarity start     p-value
      test         iteration
tau_s failed       NA        3.02e-03
tau_o failed       NA        1.13e-05
      Halfwidth Mean Halfwidth
      test
tau_s <NA>       NA   NA
tau_o <NA>       NA   NA
```

The Gweke convergence diagnostic for the Gibbs sampler implemented in BUGS (4.52 and -6.128) is smaller than for the Gibbs sampler implemented in the 'C' program (7.89 and -9.00) which suggests the Gibbs sampler implemented in BUGS approach has performed

better. In both models the dependence statistic I from the Raftery and Lewis convergence diagnostic is considerably larger than 5 which suggests convergence problems in the chains. Both models failed the Heidelberger and Welch stationarity and interval halfwidth tests. All of the tests above show that the convergence could be improved in these two models and that longer runs are required.

The Gibbs algorithm was applied to the unadjusted edge weight model, which was shown in Chapter 3 to produce a non stationary precision surface. The adjusted edge weight model is a more realistic model to use in describing the evolution of the edge nodes, however this increases the complexity of the simulation code and does not seem feasible using the BUGS program. However it is possible to extend the 'C' code to take into account the conditional edge weights. The derivation of the full conditionals for nodes in the latent structure is straightforward once the full conditional independence graph and boundary conditions have been identified. Extending the Gibbs sampler to the 2+1D model is possible but the posterior distribution is more complicated to determine due to the edge effects within the 2+1D spatio-temporal models. The order in which the the edge effects are described can prevent the full conditionals from being determined when using the BUGS package. The diagnostics for this Gibbs sampler has highlighted a convergence problem, a method which can improve the convergence of the chains is required. A tidier method which also has many other benefits for these high dimensional models is to consider a data augmentation approach.

4.3 Data Augmentation

Data augmentation is similar to Gibbs sampling, but in the lattice DLM model the nodes of the latent structure are updated in a block rather than individually by augmenting the model with the missing latent process.

4.3.1 Data Augmentation algorithm

In the data augmentation method ϕ is partitioned into ξ and η with $\phi = (\xi, \eta_1, \ldots, \eta_k)$ where $k+1 = d$ and ξ is a vector which is updated as a block. The algorithm is as follows;

1. Initialise counter to $j = 1$. Initialise the state of the chain $\phi^{(0)} = \left(\xi^{(0)}, \eta_1^{(0)}, \ldots, \eta_d^{(0)}\right)'$ in the support of $\pi(\phi)$

2. Obtain a new value $\phi^{(j)}$ from $\phi^{(j-1)}$ by successive simulation of ξ and η.

$$
\begin{aligned}
\xi^{(0)} &\sim \pi\left(\xi \,|\eta_1, \ldots, \eta_{d-1}\right) \\
\eta_1^{(j)} &\sim \pi\left(\eta_1 \left|\xi, \eta_2^{(j-1)}, \ldots, \eta_{d-1}^{(j-1)}\right.\right) \\
\eta_2^{(j)} &\sim \pi\left(\eta_1 \left|\xi, \eta_1^{(j)}, \eta_3^{(j-1)} \ldots, \eta_{d-1}^{(j-1)}\right.\right) \\
&\vdots \\
\eta_{d-1}^{(j)} &\sim \pi\left(\eta_1 \left|\xi, \eta_1^{(j)}, \ldots, \eta_{d-2}^{(j)}\right.\right)
\end{aligned}
$$

3. Increase counter to $j+1$ and repeat step 2.

In the models examined in this thesis, ξ represents the latent process θ and η_i represents the parameters in the model where $i \in (1, k)$, k is the total number of parameters of interest. This algorithm was developed by Tanner & Wong (1987).

4.3.2 Implementation

The implementation of the above data augmentation sampler is done by adapting the Gibbs algorithm written in `'C'` for the previous section. The updates for the two precision parameters remain the same whilst the update for the latent structure changes. In the Gibbs sampler each latent node is updated in turn whilst in the data augmentation method all of the latent nodes are updated in a block with the aid of `GDAGsim` a free software library. This library is written for `'C'` programmers who wish to analyse Gaussian directed acyclic graph models and is available from (`http://www.staff.ncl.ac.uk/d.j.wilkinson/`

software/gdagsim/). The software builds the sparse precision matrix for latent variables conditional on observations. The library uses the GNU Scientific Library (GSL) (http//source.redhat.com/gsl/) and depends on the sparse matrix library Meschach (ftp://ftpmaths.anu.ed.au/pub/meschach/meschach.html) however the user does not need to make direct calls to this library. GDAGsim makes it easy to implement MCMC schemes for large Gaussian systems without having to deal with the underlying matrix implementation. For more details and list of library functions see Wilkinson (2001).

In order to update the latent structure as a block build this latent structure by calling the library functions gdag_add_root and gdag_add_node which are used to build the latent structure through space and time and gdag_add_observation to add the observations to the model. Once the latent process has been defined the precision parameters are updated as in the Gibbs 'C' program.

4.3.3 Example

	Mean	SD	Naive SE	Time-series SE
τ_s	0.1722	0.02687	0.0002833	0.0004915
τ_o	0.1114	0.01175	0.0001238	0.0002158

Table 4.5: *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the data augmentation approach.*

Consider a 1+1D model which is defined by Equation 2.1 with simulated data. The data is simulated from a model with mean $\mu = 0$, state precision $\tau_s = 0.2$, observation precision $\tau_o = 0.1$ and temporal dependence parameter $\alpha = 0.7$ on 100 spatial nodes evolving over 20 time periods. The data augmentation sampler is used to simulate the two precisions from the model with known mean and temporal dependence parameter. Again the unadjusted edge weight model from Chapter 3 is used in this example.

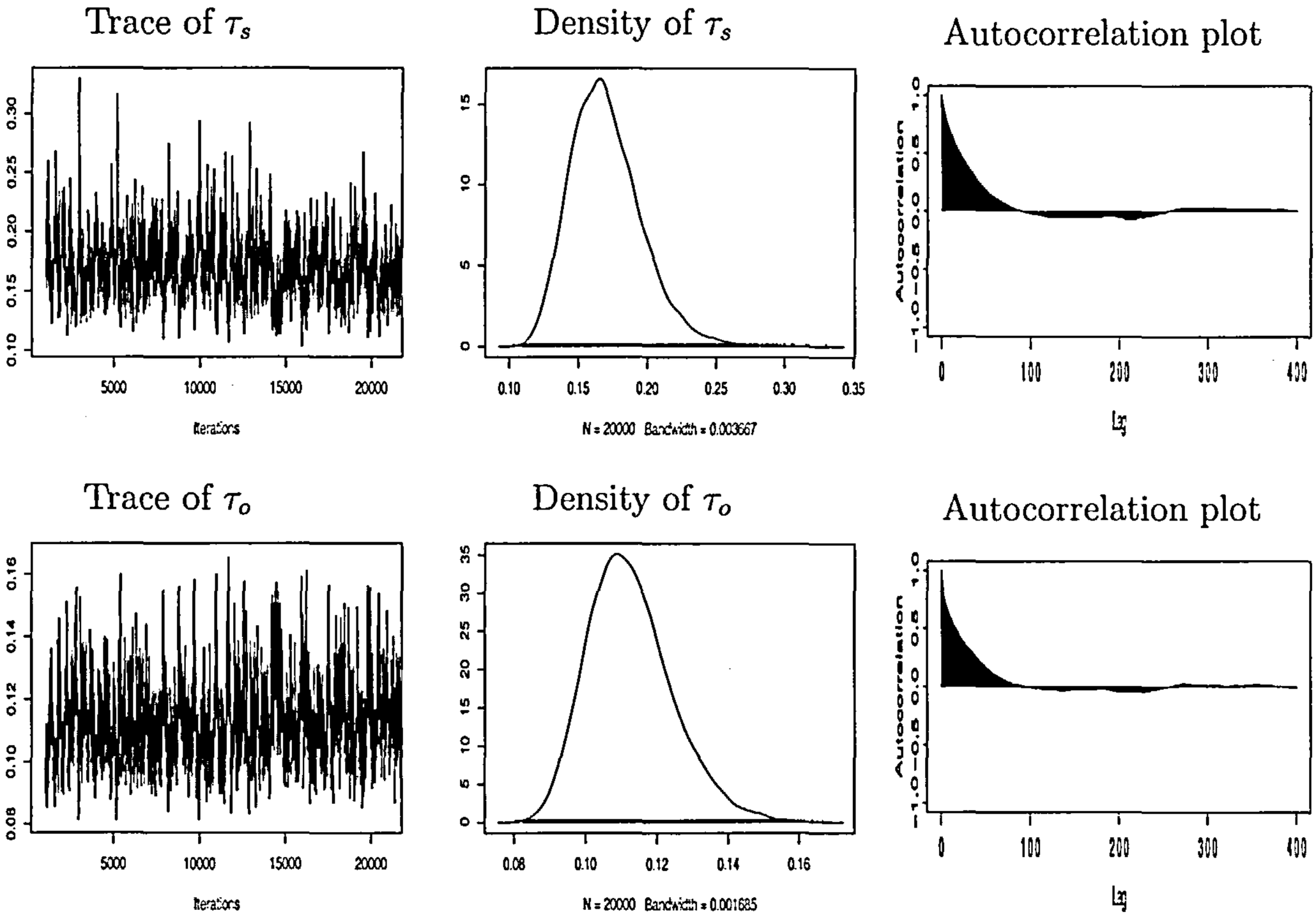


Figure 4.4: *The trace, density and autocorrelation plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the data augmentation approach with 20,000 iterations.*

	2.5%	25%	50%	75%	97.5%
τ_s	0.12863	0.1533	0.1692	0.1877	0.2328
τ_o	0.09174	0.1031	0.1100	0.1183	0.1383

Table 4.6: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the data augmentation approach.*

Figure 4.4 shows the MCMC chain for 20,000 iterations after the first 1000 iterations have been removed as burn-in and Tables 4.5 and 4.6 give the summary statistics and quartile ranges for the two chains. Both chains show good mixing and convergence. The posterior distribution suggests $\tau_s = 0.17$ which is lower than the true value of $\tau_s = 0.2$

and $\tau_o = 0.11$ which is higher than the true value of $\tau_o = 0.1$. These values are consistent with the estimates obtained using the Gibbs algorithm.

4.3.4 Discussion

The results for the data augmentation algorithm are given in Figure 4.4 showing the 10,000 iteration chain with no burn-in and Tables 4.5 and 4.6. This model uses the unadjusted edge weights which account for the discrepancy between the true precision parameter values and the simulated ones. The mixing in this model is better than that of the simulations from the Gibbs algorithm with convergence occurring much sooner. The convergence diagnostics for this model are given below:

```
GEWEKE CONVERGENCE DIAGNOSTIC (Z-score)
=======================================
Iterations used = 1000:20999
Thinning interval = 1
Sample size per chain = 20000
Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5
 tau_s  tau_o
 3.767 -2.656

RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC
========================================
Iterations used = 1000:20999
Thinning interval = 1
Sample size per chain = 20000
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95
       Burn-in  Total Lower bound  Dependence
       (M)      (N)   (Nmin)       factor (I)
 tau_s 36       39432 3746         10.5
 tau_o 42       43458 3746         11.6
```

```
HEIDELBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS
================================================================
Iterations used = 1000:20999
Thinning interval = 1
Sample size per chain = 20000
Precision of halfwidth test = 0.1
      Stationarity start     p-value
      test          iteration
tau_s failed        NA        2.70e-06
tau_o failed        NA        5.86e-07
      Halfwidth Mean Halfwidth
      test
tau_s <NA>       NA   NA
tau_o <NA>       NA   NA
```

The diagnostics for this model suggest that the convergence has not been reached and that a longer run is required. Comparing the diagnostics for this model with the ones from the Gibbs algorithm shows that the data augmentation approach has performed better. The $|Z|$ scores from the Geweke convergence diagnostic are smaller for the data augmentation approach than the Gibbs algorithm as are the corresponding dependence factors from the Rafferty and Lewis convergence diagnostic whilst both the Gibbs algorithm and data augmentation approach failed the Heidelberger and Welch stationarity test.

4.4 Metropolis-Hastings

The Metropolis-Hastings approach originates from work by Metropolis *et al.* (1953) and Hastings (1970) and allows samples to be drawn from the distribution π via a Markov chain. A transition kernel $p(\phi, \zeta)$ is constructed such that π is the equilibrium distribution of the chain. The transition kernel consists of two elements; the arbitrary transition kernel $q(\phi, \zeta)$ and an acceptance probability $A(\phi, \zeta)$. Hastings (1970) proposed

$$A(\phi, \zeta) = \min\left\{1, \frac{\pi(\zeta)q(\zeta, \phi)}{\pi(\phi)q(\phi, \zeta)}\right\}$$

for the acceptance probability as it defines a reversible chain. If $q(\phi, \zeta)$ is aperiodic and irreducible then the algorithm below defines a chain with transition kernel p and limiting

distribution π.

4.4.1 Metropolis-Hastings Algorithm

1. Initialise counter to $j = 1$ and set arbitrary initial values $\phi^{(0)}$.

2. Move chain to a new value ζ simulated from the density $q(\phi^{(j-1)}, \cdot)$.

3. Evaluate the acceptance probability of the move $A(\phi^{(j-1)}, \zeta)$. If this move is accepted $\phi^{(j)} = \zeta$. If it is not accepted $\phi^{(j)} = \phi^{(j-1)}$and the chain does not move.

4. Change counter from j to $j + 1$ and return to step 2 until convergence.

In the models considered so far in this chapter there have been two unknown variables, τ_s and τ_o. MCMC techniques are used to to estimate these parameters. Consider a Metropolis-Hastings random walk update for each parameter and obtain the acceptance probability from the prior distributions as described in the following subsections.

4.4.2 Independent Parameter Updates

Assume the two precision parameters are independent and then consider Gamma priors for these parameters,

$$\tau_i \sim \Gamma(a_i, b_i). \tag{4.23}$$

Using these priors the acceptance probability, A, for the update is

$$A = \frac{[\tau_s^*]\,[\tau_o^*]\,[x|\phi^*]\,f(\phi|\phi^*)}{[\tau_s]\,[\tau_o]\,[x|\phi]\,f(\phi^*|\phi)} \tag{4.24}$$

where $f(\phi^*|\phi)$ is the proposal density for the parameters. Since the proposal density is symmetric, then (4.24) becomes

$$A = \frac{[\tau_s^*]\,[\tau_o^*]\,[x|\phi^*]}{[\tau_s]\,[\tau_o]\,[x|\phi]}. \tag{4.25}$$

To obtain a rapidly mixing chain, random walk updates are performed on the log precisions and so the density of the log of a gamma distribution is used in the acceptance probability. For numerical stability the log acceptance probability, $a = \log A$, is used.

$$\begin{aligned} a &= [a_s \log b_s - \log\Gamma(a_s) + b_s\lambda_s^* + a_s \exp\lambda_s^*] - [a_s \log b_s - \log\Gamma(a_s) \\ &\quad + a_s\lambda_s + b_s \exp\lambda_s] + [a_o \log b_o - \log\Gamma(a_o) + a_o\lambda_o^* + b_o \exp\lambda_o^*] \\ &\quad - [a_o \log b_o - \log\Gamma(a_o) + a_o\lambda_o + b_o \exp\lambda_o] \\ &\quad + \text{marginal log likelihood ratio.} \end{aligned} \tag{4.26}$$

The marginal log likelihood ratio is calculated using one of the `GDAGsim` library functions. The ratio is obtained by subtracting the marginal log likelihood of the previous model from the marginal log likelihood of the proposed model.

4.4.3 Dependent Parameter Updates

Bad mixing in the above case can be due to high negative posterior correlation between the posterior distributions of the precision components. This can be improved by considering a bivariate update of these components. If the precisions have a posterior correlation of approximately $-\rho$ consider an update of the form

$$\begin{aligned} \log\tau_o^* &= \log\tau_o + y, \\ \log\tau_s^* &= \log\tau_s + z, \\ y &\sim N(0, g^{-1}), \\ z &\sim N\left(-\rho y, \frac{1}{g(1-\rho^2)}\right) \end{aligned} \tag{4.27}$$

where g is the precision for the random walk update and the log precisions are updated in the same way as for the independent model above. The random walk update is still symmetric with the prior distributions for the parameters unchanged, so the acceptance probability remains unchanged, and is given by Equation (4.26).

4.4.4 Implementation

The Metropolis-Hastings algorithm is implemented using a `'C'` program which calls the `GDAGsim` library. The latent nodes and observations are added in the same way as in the data augmentation method using the new parameter values that are obtained from the random walk updates. Once the model has been built the `GDAGsim` library function `gdag_mloglik` is used to obtain the marginal log likelihood of the observed data which is integrated over the distribution of the latent variables. This value along with information from the prior distributions of the parameters is used to calculate the acceptance probability which determines if the new parameter values are to be accepted or if the previous parameter values are to be kept.

4.4.5 Example: 1+1D model

Consider a 1+1D model which is defined by Equation 2.1 with simulated data. The data is simulated from a model with mean $\mu = 0$, state precision $\tau_s = 0.2$, observation precision $\tau_o = 0.1$ and temporal dependence parameter $\alpha = 0.7$ on a 100 spatial nodes evolving over 20 time periods. The Metropolis-Hastings sampler is used to simulate the two log precisions, where $\lambda_i = \log(\tau_i)$ from the model with known mean and temporal dependence parameter. The adjusted edge weight model from Chapter 3 is used in this example along with random walk updates where

$$\begin{aligned} \lambda_s^* &= \lambda_s + N(0, 400^{-1}), \\ \lambda_o^* &= \lambda_o + N(0, 400^{-1}). \end{aligned}$$

The MCMC chain for 20,000 iterations with no burn-in is shown in Figure 4.5 the corresponding summary statistics and quartile ranges are given in Tables 4.7 and 4.8. In the Metropolis-Hastings method the algorithm simulates the log precisions thus the actual parameter estimates are $\log(\tau_s) = -1.6094$ and $\log(\tau_o) = -2.3026$. The chains in Figure 4.5 show reasonable mixing and convergence. The posterior distribution suggests $\log(\tau_s) = -1.7$ which is lower than the true value of $\log(\tau_s)$ and $\log(\tau_o) = -2.2$ which

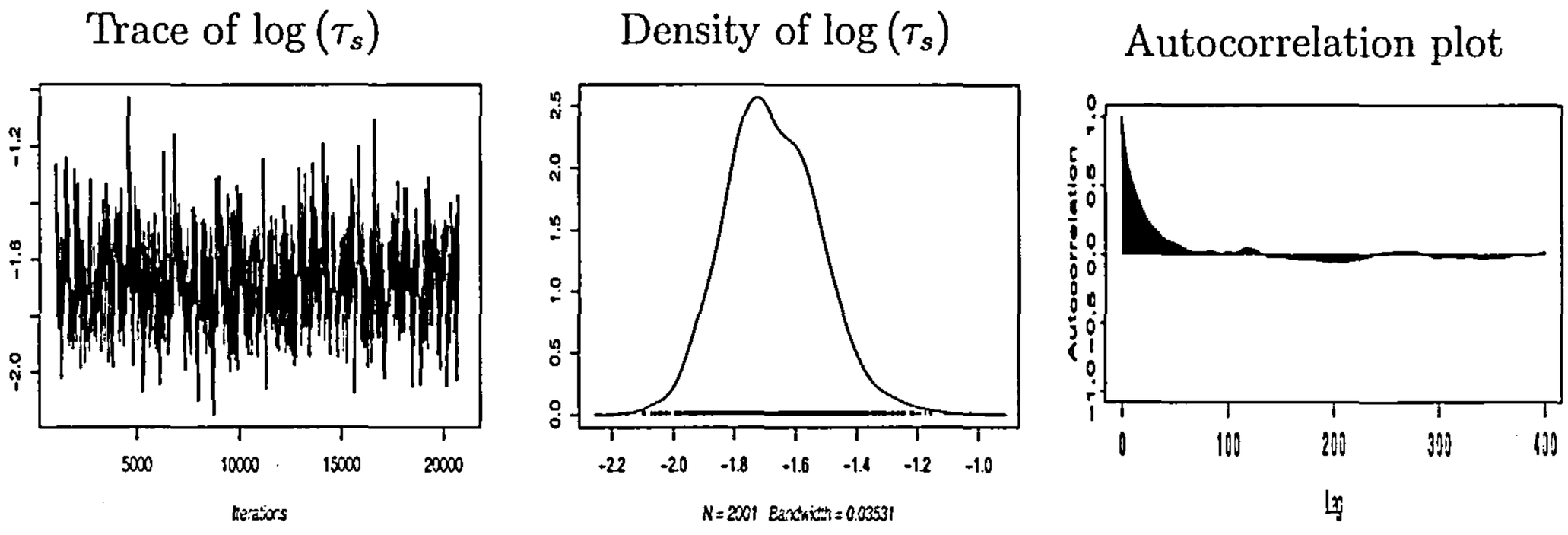


Figure 4.5: *The trace, density and auto correlation plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the adjusted edge weight model and the Metropolis-Hastings approach with 20,000 iterations.*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.676	0.1523	0.0010770	0.001893
$\log(\tau_o)$	-2.202	0.0976	0.0006902	0.001220

Table 4.7: *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Metropolis-Hastings approach.*

is higher than the true value of $\log(\tau_o)$. These values are equivalent to $\tau_s = 0.182$ and $\tau_o = 0.11$ which are consistent with the estimates obtained using the Gibbs algorithm or the data augmentation algorithm. The convergence statistics for the 1+1D model are

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.950	-1.783	-1.684	-1.576	-1.365
$\log(\tau_o)$	-2.382	-2.269	-2.207	-2.141	-1.999

Table 4.8: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1D model with 100 spatial nodes evolving through 20 time periods using the Metropolis-Hastings approach.*

given below and it can be seen that this model has performed more efficiently than the previous two approaches.

```
GEWEKE CONVERGENCE DIAGNOSTIC (Z-score)
========================================
Iterations used = 1000:21000
Thinning interval = 1
Sample size per chain = 20001
Fraction in 1st window = 0.1
Fraction in 2nd window = 0.5
lambda_s lambda_o
  -2.447    2.528

RAFTERY AND LEWIS CONVERGENCE DIAGNOSTIC
========================================
Iterations used = 1000:21000
Thinning interval = 1
Sample size per chain = 20001
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95
          Burn-in  Total Lower bound  Dependence
          (M)      (N)   (Nmin)       factor (I)
lambda_s  28       29909 3746         7.98
lambda_o  27       29602 3746         7.90
```

```
HEIDELBERGER AND WELCH STATIONARITY AND INTERVAL HALFWIDTH TESTS
================================================================
Iterations used = 1000:21000
Thinning interval = 1
Sample size per chain = 20001
Precision of halfwidth test = 0.1
          Stationarity start     p-value
          test          iteration
lambda_s passed         1000       0.0535
lambda_o passed         1000       0.0877
          Halfwidth Mean  Halfwidth
          test
lambda_s passed     -1.68 0.00371
lambda_o passed     -2.20 0.00239
```

The diagnostics above show that this model has performed reasonably well but the chain length should be extended to 30,000 iterations. Comparing these diagnostics with those for the Gibbs algorithm and data augmentation approach shows that this Metropolis Hastings algorithm has lower $|Z|$ score from the Geweke convergence statistics and the dependence factor from the Rafferty and Lewis diagnostic than either the Gibbs algorithm or the data augmentation approach. It is also the only method that has passed the Heidelberger and Welch stationarity test. The convergence diagnostics from this model show that the Metropolis-Hastings method has performed much better than either the Gibbs sampler or data augmentation approach.

4.4.6 Example: 2+1D model

Consider a 2+1D model which is defined by Equation 2.1 with simulated data. The data is simulated from a model with mean $\mu = 0$, state precision $\tau_s = 0.2$, observation precision $\tau_o = 0.1$ and temporal dependence parameter $\alpha = 0.7$ on a 10×10 spatial grid evolving over 20 time periods. Both the Gibbs approach implemented in BUGS and the Metropolis-Hastings algorithm are used to generate the posterior distributions for the precision parameters.

The results from the BUGS program are given in Figure 4.6 and Tables 4.9 and 4.10.

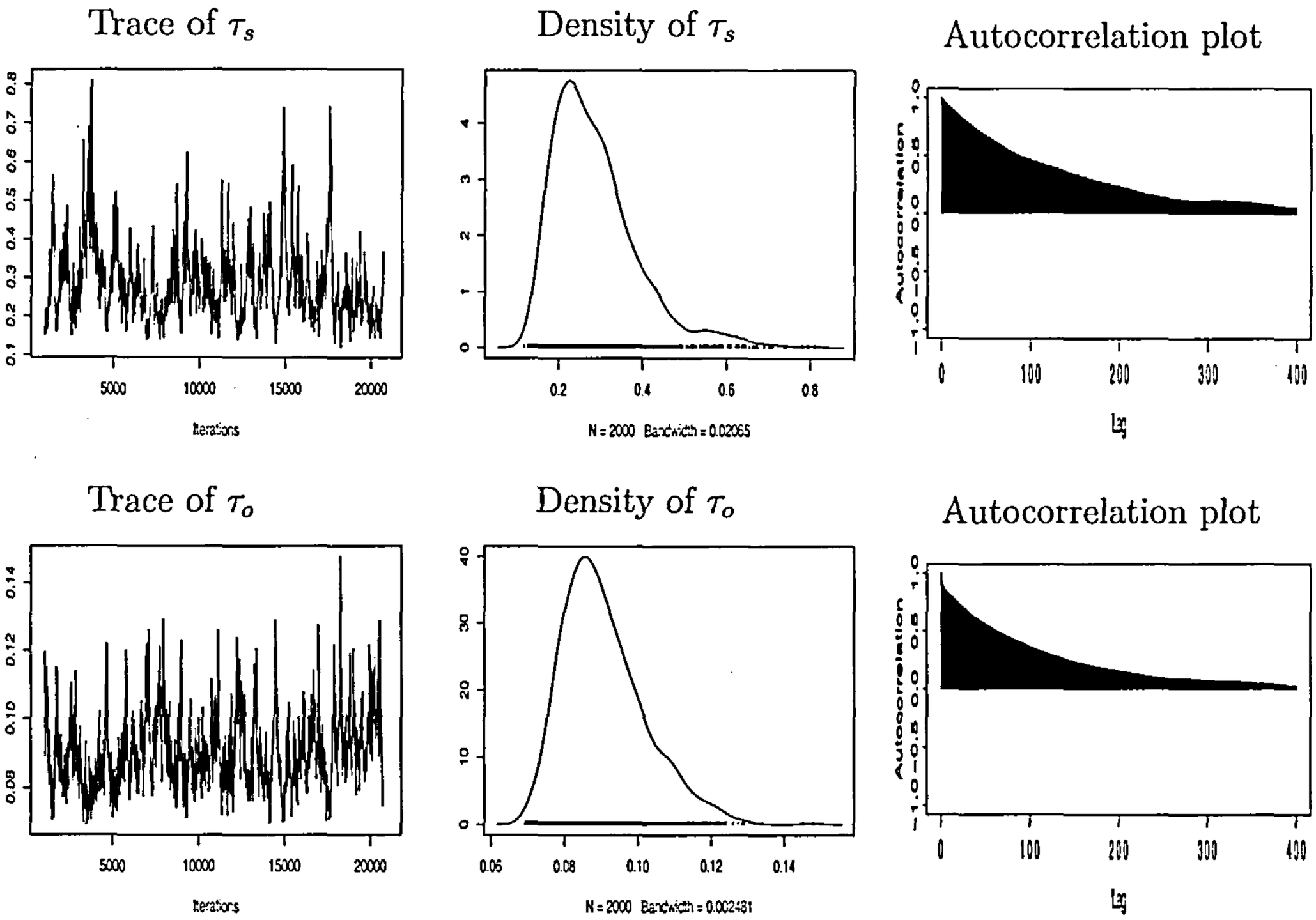


Figure 4.6: *The trace, density and auto correlation plots for the two precision parameters τ_s and τ_o for the 2+1D model with a 10×10 spatial grid evolving over 20 time periods generated using the Gibbs sampler implemented in BUGS with 20,000 iterations.*

The summary statistics for the posterior distributions suggest $\tau_s = 0.27$ which is larger than the actual value of τ_s and $\tau_o = 0.09$ which is lower than the true value of τ_o. The difference between these parameter estimates and the true values is probably caused by edge effects, the BUGS program uses the unadjusted edge weights and takes no account of the reduction of information available.

The Metropolis-Hastings sampler is used to simulate the two log precisions, where $\lambda_i = \log(\tau_i)$ from the model with known mean and temporal dependence parameter. The adjusted edge weight model from Chapter 3 is used in this example along with random walk updates where

$$\begin{aligned} \lambda_s^* &= \lambda_s + N(0, 400^{-1}), \\ \lambda_o^* &= \lambda_o + N(0, 400^{-1}). \end{aligned}$$

	Mean	SD	Naive SE	Time-series SE
τ_s	0.28737	0.10273	0.0022971	0.0041160
τ_o	0.09052	0.01110	0.0002483	0.0004156

Table 4.9: *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Gibbs sampler implemented in BUGS.*

	2.5%	25%	50%	75%	97.5%
τ_s	0.1551	0.2140	0.26822	0.33336	0.5678
τ_o	0.0739	0.0824	0.08854	0.09674	0.1175

Table 4.10: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Gibbs sampler implemented in BUGS.*

The results from this model generated with 10,000 iterations are shown in Figure 4.7 and Tables 4.11 and 4.12. In the Metropolis-Hastings method the algorithm simulates the log precisions thus the actual parameter estimates are $\log(\tau_s) = -1.6094$ and $\log(\tau_o) = -2.3026$. The chains in Figure 4.5 show reasonable mixing and convergence. The posterior distribution suggests $\log(\tau_s) = -1.2$ which is higher than the true value of $\log(\tau_s)$ and $\log(\tau_o) = -2.4$ which is lower than the true value of $\log(\tau_o)$. These values are equivalent to $\tau_s = 0.3$ and $\tau_o = 0.09$ which are consistent with the estimates obtained using the Gibbs algorithm.

Comparing Figures 4.6 and 4.7 the Metropolis-Hastings approach appears to have better mixing than the Gibbs algorithm. The autocorrelation plots also show that the Metropolis-Hastings approach has performed better than the Gibbs algorithm. Examination of the corresponding figures for the 1+1D model (see Figures 4.2 and 4.5) shows that the trace plots look to be mixing equally as well and that the autocorrelation plots are similar. As the models increase in size the convergence of the posterior distributions

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.169	0.3998	0.0028268	0.007139
$\log(\tau_o)$	-2.443	0.1196	0.0008454	0.001473

Table 4.11: *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.808	-1.451	-1.223	-0.9484	-0.2527
$\log(\tau_s)$	-2.644	-2.528	-2.452	-2.3699	-2.1829

Table 4.12: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm.*

becomes more of a problem. In a small 1+1D model the Gibbs sampler implemented using BUGS is a suitable approach for the simulation of the posterior distribution, whereas in larger 2+1D models the Metropolis-Hastings approach may be more suitable.

4.4.7 Discussion

The Metropolis-Hastings algorithm provides a tool for generating posterior distributions for the parameters in the DLM which can easily handle edge effects and allow them to be corrected. The results for both the 1+1D and 2+1D models show that the results from the Metropolis-Hastings algorithm is consistent with the results from the other approaches and that as in the data augmentation approach convergence occurs almost immediately. The Metropolis-Hastings algorithm is easier to implement than the Gibbs algorithm in the 2+1D model as the full conditionals do not need to be calculated for model parameters and the `GDAGsim` library functions enable the complex marginal log likelihood to be obtained

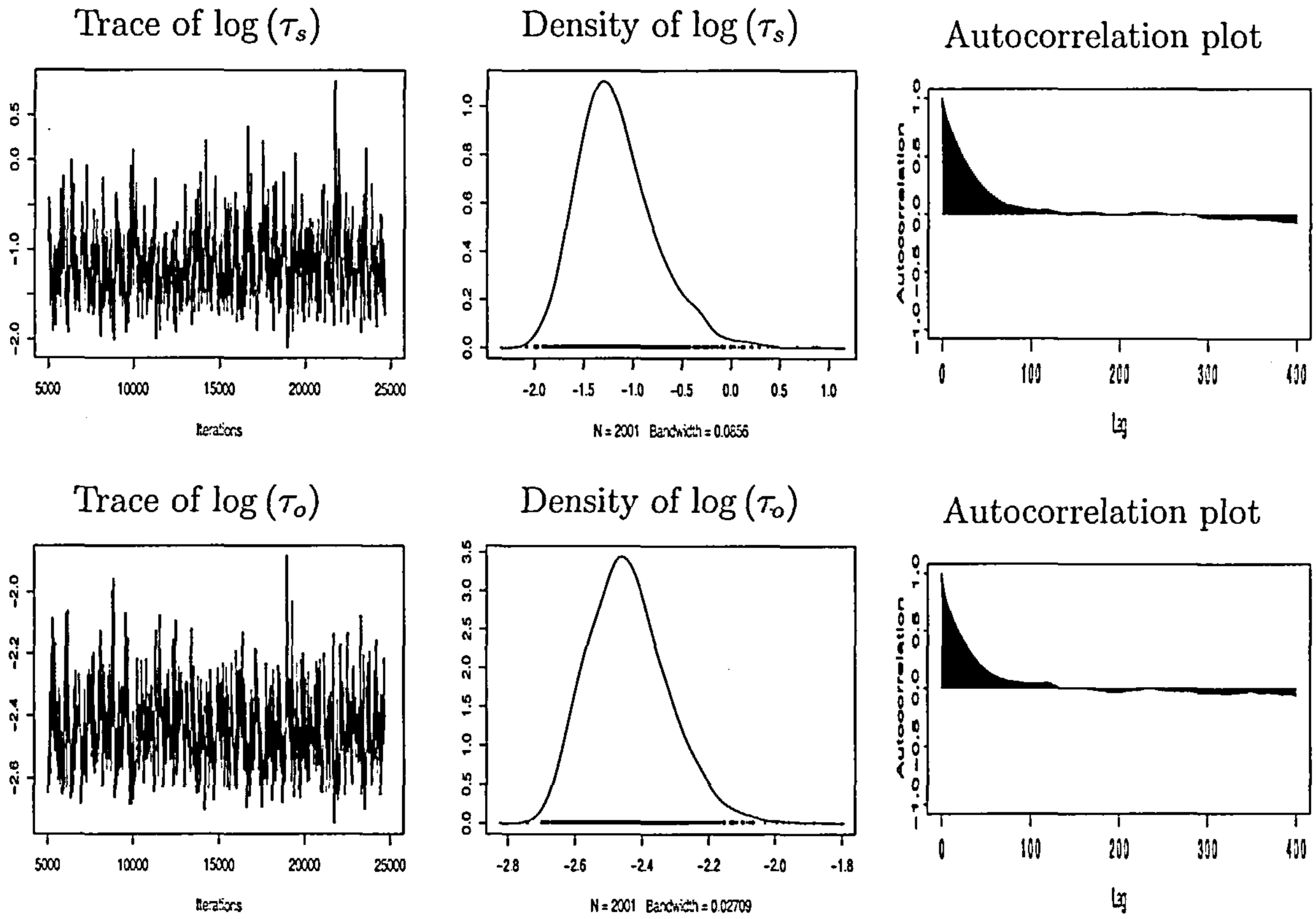


Figure 4.7: *The trace, density and autocorrelation plots for the two precision parameters τ_s and τ_o for the 2+1D model with a 10×10 spatial grid evolving over 20 time periods generated using the Metropolis-Hastings algorithm.*

easily.

The convergence diagnostics for the 1+1D model have shown that the Metropolis-Hastings algorithm performs much better than either the data augmentation approach or the Gibbs algorithm. The 2+1D model with a 10×10 spatial grid is affected by the edge effects and the parameter estimated are expected to be better for a model with a larger spatial area. Unfortunately increasing the spatial area also significantly increases the amount of memory the MCMC simulation program requires thus a method for reducing the computational time is needed.

4.5 Parallel chains approach

The MCMC simulations described above are computationally intensive and the simulation sizes increases dramatically with the dimensions of the model. This implies that a method of reducing the computational time is required. Whiley & Wilson (2002) and Wilkinson (2004) examine the use of parallel algorithms for MCMC simulation which allows larger models to be analysed compared with the size of model that can feasibly be analysed on a serial machine. Both papers highlight the importance of convergence in the selection of the most appropriate method of parallisation for a model. There are two approaches to applying a parallel method to MCMC programs, to run the chain across n processors as a series process splitting the program between the n processors or to run the program in a parallel fashion where each of the n processors runs the same program and the resulting simulation chains can then be combined for analysis. Two further libraries are required for this parallel approach, MPI (`http://www-unix.mcs.anl.gov/mpi/`) and SPRNG (`http://sprng.cs.fsu.edu/`). MPI is the message passing interface which allows information to be passed between each of the processor nodes whilst SPRNG is a random number generator which allows independent streams of random numbers to be generated for each processor. The MCMC chains examined in this chapter all had fast convergence which suggests that the parallel approach is suitable for these models, thus running the processes on a Beowulf cluster with eight processor nodes allows the number of iterations overall to be reduced and thus reduces the computation time required.

4.5.1 Example

Consider a 2+1D model which is defined by Equation 2.1 with simulated data. The data is simulated from a model with mean $\mu = 0$, state precision $\tau_s = 0.2$, observation precision $\tau_o = 0.1$ and temporal dependence parameter $\alpha = 0.7$ on a 10×10 spatial grid evolving over 20 time periods. The Metropolis-Hastings algorithm is used to generate the posterior distributions for the precision parameters in a parallel setting which generates eight chains

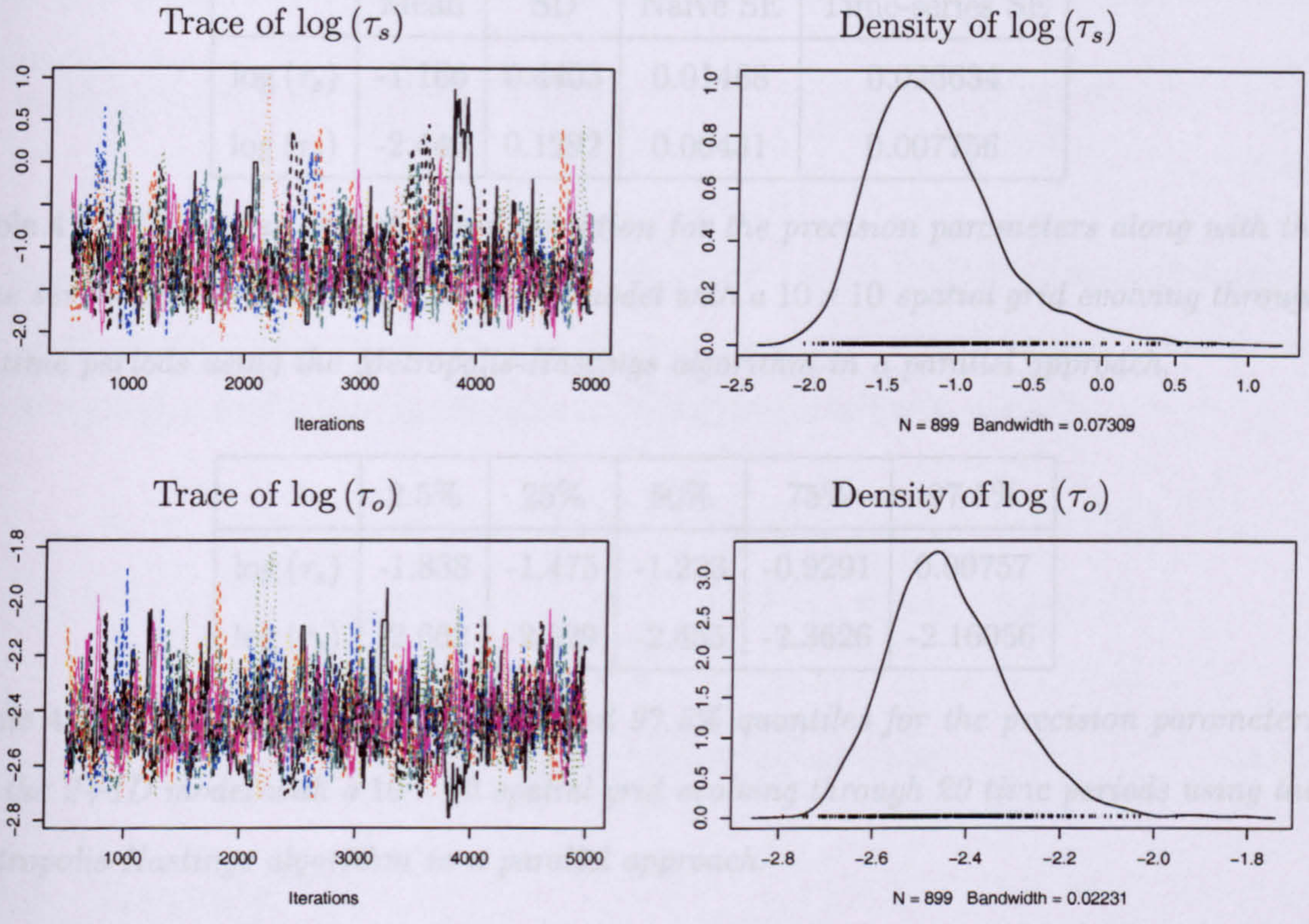


Figure 4.8: *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model with a 10×10 spatial grid evolving over 20 time periods generated using a parallel approach with eight chains each of 5000 iterations.*

with 5000 iterations for each chain.

Figure 4.8 and Tables 4.13 and 4.14 give the results for this model using a parallel approach. Figure 4.8 shows good mixing and convergence of the eight chains after the removal of the burn-in period which was 250 iterations. This model suggests parameter estimates of $\log(\tau_s) = -1.2$ which is higher than the true value of $\log(\tau_s)$ and $\log(\tau_o) = -2.4$ which is lower than the true value of $\log(\tau_o)$. These estimates are the same as those suggested by the Metropolis-Hastings algorithm applied to a single chain. This parallel approach provides a mechanism for reducing the time needed to produce parameter estimators from the simulated posterior distributions.

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.166	0.4403	0.01468	0.026634
$\log(\tau_o)$	-2.440	0.1292	0.00431	0.007756

Table 4.13: *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm in a parallel approach.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.838	-1.475	-1.223	-0.9291	0.09757
$\log(\tau_s)$	-2.662	-2.529	-2.455	-2.3626	-2.16056

Table 4.14: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using the Metropolis-Hastings algorithm in a parallel approach.*

4.6 Introducing the mean parameter

The models examined previously have only been interested in sampling from the posterior distribution of the two precision parameters, in this section consider the 2+1D model where interest is in sampling from the precision and mean parameters posterior distributions. Assume the precision and mean parameters are independent and then consider Gamma priors for the precision parameters and a Normal prior for the mean parameter,

$$\tau_i \sim \Gamma(a_i, b_i),$$
$$\mu \sim N(c, d). \tag{4.28}$$

Using these priors the acceptance probability, A, for the update is

$$A = \frac{[\tau_s^*]\,[\tau_o^*]\,[\mu^*]\,[x|\phi^*]\,f(\phi|\phi^*)}{[\tau_s]\,[\tau_o]\,[\mu]\,[x|\phi]\,f(\phi^*|\phi)} \tag{4.29}$$

where $f(\phi^*|\phi)$ is the proposal density for the parameters. Since the proposal density is symmetric, then (4.24) becomes

$$A = \frac{[\tau_s^*]\,[\tau_o^*]\,[\mu^*]\,[x|\phi^*]}{[\tau_s]\,[\tau_o]\,[\mu]\,[x|\phi]}. \tag{4.30}$$

To obtain a rapidly mixing chain, random walk updates are performed on the log precisions and so the density of the log of a gamma distribution is used in the acceptance probability. For numerical stability the log acceptance probability, $a = \log A$, is used.

$$\begin{aligned} a = {} & [a_s \log b_s - \log\Gamma(a_s) + b_s\lambda_s^* + a_s \exp\lambda_s^*] - [a_s \log b_s - \log\Gamma(a_s) \\ & + a_s\lambda_s + b_s \exp\lambda_s] + [a_o \log b_o - \log\Gamma(a_o) + a_o\lambda_o^* + b_o \exp\lambda_o^*] \\ & - [a_o \log b_o - \log\Gamma(a_o) + a_o\lambda_o + b_o \exp\lambda_o] - \frac{1}{2d}\left[\mu^*\left(\mu^* - 2c\right)\right] \\ & + \frac{1}{2d}\left[\mu\left(\mu - 2c\right)\right] + \text{marginal log likelihood ratio.} \end{aligned} \tag{4.31}$$

4.6.1 Example

Consider a 2+1D model which is defined by Equation 2.1 with simulated data. The data is simulated from a model with mean $\mu = 0$, state precision $\tau_s = 0.2$, observation precision $\tau_o = 0.1$ and temporal dependence parameter $\alpha = 0.7$ on a 10×10 spatial grid evolving over 20 time periods. The Metropolis-Hastings sampler is used to simulate the two log precisions and the mean parameter, where $\lambda_i = \log(\tau_i)$ and μ are from the model with known temporal dependence parameter. The adjusted edge weight model from Chapter 3 is used in this example along with random walk updates where

$$\begin{aligned} \lambda_s^* &= \lambda_s + N(0, 400^{-1}), \\ \lambda_o^* &= \lambda_o + N(0, 400^{-1}), \\ \mu^* &= \mu + N(0, 100^{-1}). \end{aligned}$$

The MCMC chain for 10,000 iterations with no burn-in is shown in Figure 4.9 the corresponding summary statistics and quartile ranges are given in Tables 4.15 and 4.16. The Metropolis-Hastings method updates the log precisions thus the actual parameter

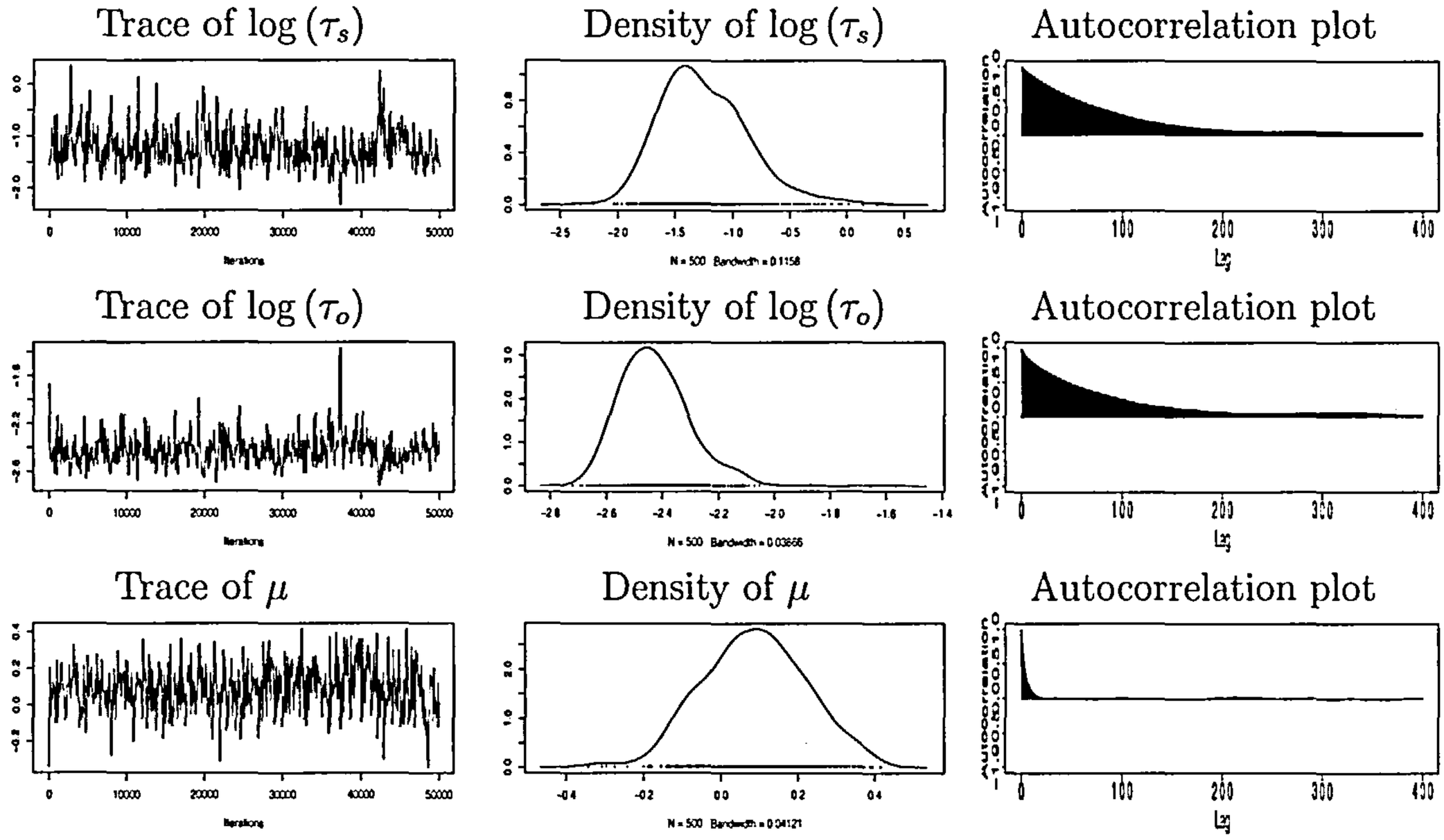


Figure 4.9: *The trace,density and auto correlation plots for the two precision parameters τ_s, τ_o and the mean parameter μ for the 2+1D model with a 10×10 spatial grid evolving over 20 time periods generated using Metropolis-Hastings.*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.24535	0.3902	0.003902	0.006851
$\log(\tau_o)$	-2.41882	0.1337	0.001337	0.002338
μ	0.09686	0.1328	0.001328	0.001940

Table 4.15: *The mean and standard deviation for the precision and mean parameters along with the time series standard error for the 2+1D model with a 10×10 spatial grid evolving through 20 time periods.*

values are $\log(\tau_s) = -1.6094$ and $\log(\tau_o) = -2.3026$. The chains in Figure 4.9 show reasonable mixing and convergence. The posterior distribution suggests $\log(\tau_s) = -1.3$ which is higher than the true value of $\log(\tau_s)$, $\log(\tau_o) = -2.4$ which is lower than the true value of $\log(\tau_o)$ and $\mu = 0.1$ which is higher than the true value of μ.

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.8751	-1.523790	-1.28794	-1.0179	-0.3434
$\log(\tau_o)$	-2.6360	-2.510051	-2.43341	-2.3423	-2.1244
μ	-0.1526	0.006876	0.09374	0.1867	0.3599

Table 4.16: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods.*

4.7 Discussion

This chapter has compared three approaches for simulating the model parameters. The three approaches used were Gibbs sampling, data augmentation and a Metropolis-Hastings method. These three methods were compared using a 1+1D model with 100 spatial nodes evolving over 20 time periods. The Gibbs algorithm uses the full conditionals to sample from the posterior distribution. The full conditionals are easy to obtain for the internal nodes in both the 1+1D model and 2+1D model however the corresponding full conditionals along the edges of the system can be more complicated to obtain. Data augmentation was the second approach considered, this model performed better than the Gibbs algorithm. The mixing and convergence of the parameter estimates was not ideal so the parameters were simulated using a Metropolis-Hastings approach. The chains obtained using this simulation technique mixed much better than those which used the Gibbs algorithm or the data augmentation approach.

The Gibbs algorithm applied to the 1+1D model required 9.8 Megabytes of memory to run on a computer whereas the same model using data augmentation required 10 Megabytes of memory and the Metropolis-Hastings method required 16 Megabytes of memory. Although the Metropolis-Hastings approach requires more computer memory the resulting chains performed better by the convergence statistics than the other two approaches and it required a smaller chain length. Computationally all three methods were easy to implement: the Gibbs algorithm due to the BUGS package which builds the full

conditionals for the model; the data augmentation and Metropolis-Hastings approaches rely on the `GDAGsim` library which is capable of calculating the maximum log likelihood of the model at each iteration.

The examples examined in future chapters all use the Metropolis-Hastings approach for MCMC simulation, this method being easy to implement with the aid of the `GDAGsim` library. This approach has performed the best according to the convergence statistics and the increase in computational size is justified by the reduced chain size that is required.

Chapter 5

The Coarse Model

5.1 Introduction

The model used in Chapter 3 adjusts for the edges of the system and in the case of the 2+1D model produces an approximate stationary precision surface for the system which in turn leads to improved parameter estimation. This edge effect model is computationally demanding within the MCMC framework, so a method to reduce the computational time is desirable. One approach is to consider modelling on a coarse grid (Higdon *et al.* 2003). In this coarse model every other spatial node is considered for the 1+1D system and one in four spatial nodes are considered for the 2+1D system at each time period (Figures 5.3 and 5.5).

The aim of this model is to generate parameter estimates which are consistent and share interpretation between coarse and fine scales. The fine model can be approximated by a coarse model which is a Markov process of order two. In the 1+1D case each node has four parents. Here information is passed from three nodes at time period $t-2$ and one node from time period $t-1$ to the node of interest at time period t. Similarly the 2+1D case uses a system with 10 parents with nine nodes from time period $t-2$ and one node from the time period $t-1$ to the node of interest at time period t.

As in Chapter 3 the model can suffer from edge effects where parents are missing along

the edge of the system. These edge effects need to be considered and appropriate edge adjustments used, in this model there are more spatial nodes that lie along the edge of the system due to the second order dependence through time. All of the spatial nodes at time $t = 1$ suffer from these edge effects as information can only be passed from the previous time period since the second order Markovian process can not be used in this time period. As well as dealing with a larger proportion of edge nodes in the coarse model the edge weights for the internal nodes are not as obvious as for the fine model. Not only do the weights have to be applied in space but due to the second order dependency they must also be applied across time periods. One approach is to calculate the conditional distribution for any node given its parents at time $t-2$ and $t-1$ as in the method used for the fine model in Chapter 3.

In order to generate the required edge weights and adjusted variances the stationary distribution is required. These values can be generated for a general zero mean unit precision model and then stored in a reference table ready to be read into the MCMC simulation program as described in Chapter 3 for the fine model. This stationary distribution is obtained from the joint covariance matrix and then the rows and columns corresponding to the nodes of interest are used to form the variance covariance matrix for the coarse model. Numerical examples of the precision surfaces are given for both the 1+1D and 2+1D models and the corresponding MCMC examples are given at the end of this chapter.

5.2 The Second Order Markov Process

To understand why a second order Markov process is required for the coarse model consider first the 1+1D model which uses every other spatial node at each time point (the circled nodes in Figure 5.1). If a first order Markov process was used to approximate the fine model, the latent node $\theta_i^{(t)}$ would be conditional on one node $\theta_i^{(t-1)}$ and this model would be unable to describe the spatial dynamics of how the system evolves. The

use of a second order Markov chain allows information from nodes two time periods away to be used in the evolution of the system. Figure 5.1 shows the 1+1D model for three time periods and the hidden layers (see Chapter 2) at time periods $t - \frac{1}{2}$ and $t - \frac{3}{2}$. The circled nodes are the nodes which are available in the coarsened model and the arrows show the paths that can be used to pass information through the system. Node $\theta_i^{(t)}$ has parents $\text{pa}\left(\theta_i^{(t)}\right) = \left(\theta_{i-2}^{(t-2)}, \theta_i^{(t-2)}, \theta_{i+2}^{(t-2)}, \theta_i^{(t-1)}\right)$. The second order Markov process can be similarly constructed for the 2+1D model, here $\theta_{i,j}^{(t)}$ has parents $\text{pa}\left(\theta^{(t)}\right) = \left(\theta_{i-2,j-2}^{(t-2)}, \theta_{i-2,j}^{(t-2)}, \theta_{i-2,j+2}^{(t-2)}, \theta_{i,j-2}^{(t-2)}, \theta_{i,j}^{(t-2)}, \theta_{i,j+2}^{(t-2)}, \theta_{i+2,j-2}^{(t-2)}, \theta_{i+2,j}^{(t-2)}, \theta_{i+2,j+2}^{(t-2)}, \theta_i^{(t-1)}\right)$.

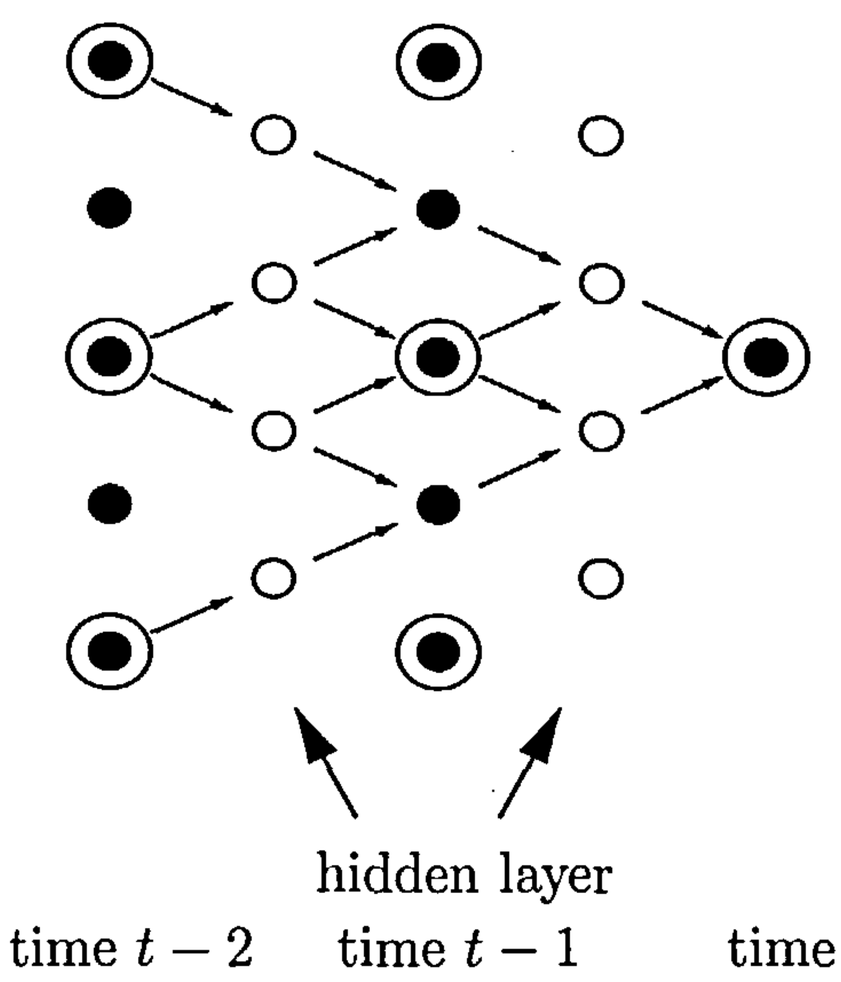


Figure 5.1: *The graphical representation of the hidden layer in the 1+1D model over two time periods.*

5.3 The Conditional Distribution

For this coarse model the nodes of interest are every other node in each spatial dimension at each time period. In Figures 5.3 and 5.5 the nodes of interest for the 1+1D and 2+1D models have been circled. To distinguish between the coarse and fine models let $\tilde{\theta}$ be nodes on the coarse scale and θ be nodes on the fine scale. Therefore $\tilde{\theta}_i^{(t)}$ is the node of interest and $pa\left(\tilde{\theta}_i^{(t)}\right)$ is the vector of its parents. The conditional distribution of interest

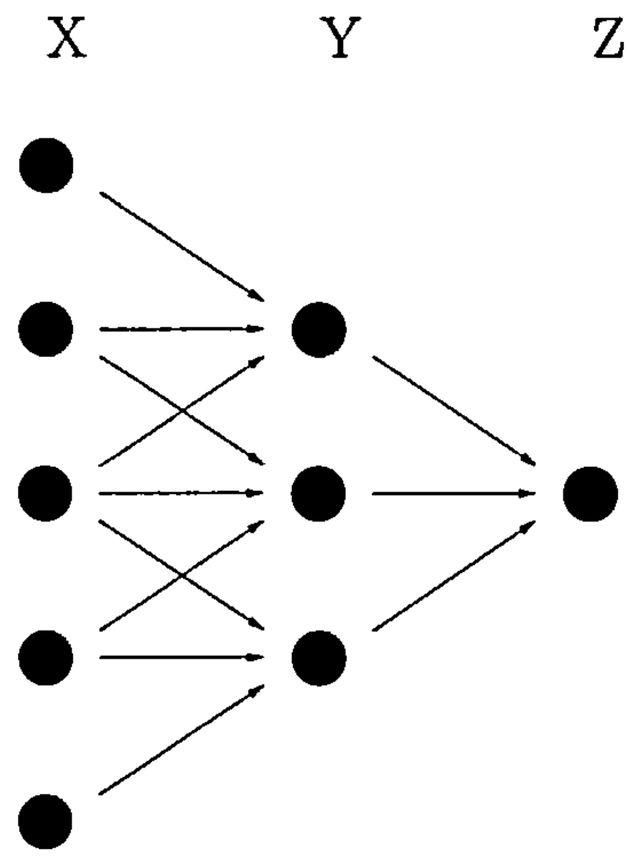


Figure 5.2: *The second order Markov process for the fine 1+1D model.*

is $\tilde{\theta}_i^{(t)} \left| pa\left(\tilde{\theta}_i^{(t)}\right) \right.$ which is given by

$$\begin{aligned}\tilde{\theta}_i^{(t)}|pa\left(\tilde{\theta}_i^{(t)}\right) \sim\ & N\left(E\left(\tilde{\theta}_i^{(t)}\right) + Cov\left(\tilde{\theta}_i^{(t)}, pa\left(\tilde{\theta}_i^{(t)}\right)\right)\right. \\ & \times Var\left(pa\left(\tilde{\theta}_i^{(t)}\right)\right)^{-1}\left(pa\left(\tilde{\theta}_i^{(t)}\right) - E\left(pa\left(\tilde{\theta}_i^{(t)}\right)\right)\right) \\ & , Var\left(\tilde{\theta}_i^{(t)}\right) - Cov\left(\tilde{\theta}_i^{(t)}, pa\left(\tilde{\theta}_i^{(t)}\right)\right) \\ & \left.\times Var\left(pa\left(\tilde{\theta}_i^{(t)}\right)\right)^{-1} Cov\left(pa\left(\tilde{\theta}_i^{(t)}\right), \tilde{\theta}_i^{(t)}\right)\right).\end{aligned} \tag{5.1}$$

The distribution of $\tilde{\theta}_i^{(t)}$ is not known however it can be determined form the joint distribution of the the second order Markov process for a fine model where, in the 1+1D situation the node of interest has eight parent nodes or in the 2+1D model it has thirty four parent nodes. Using the 1+1D model the coarse model can be formulated as follows. Consider the situation depicted in Figure 5.2 if node Z is normally distributed with mean 0 and variance $V_Z = v_0$, Y is from a multivariate Normal distribution with the mean of each node being 0 and variance matrix

$$V_y = \begin{pmatrix} v_0 & v_1 & v_2 \\ v_1 & v_0 & v_1 \\ v_2 & v_1 & v_0 \end{pmatrix}.$$

and X is from a multivariate Normal distribution with the mean of each node being 0

and variance matrix

$$V_x = \begin{pmatrix} v_0 & v_1 & v_2 & v_3 & v_4 \\ v_1 & v_0 & v_1 & v_2 & v_3 \\ v_2 & v_1 & v_0 & v_1 & v_2 \\ v_3 & v_2 & v_1 & v_0 & v_1 \\ v_4 & v_3 & v_2 & v_1 & v_0 \end{pmatrix}.$$

Then the joint distribution of these three variables is multivariate Normal with mean 0 for each node and joint variance matrix given by

$$\left(\begin{array}{c|c|c} V_x & V_{xy} & V_{xz} \\ \hline V_{yx} & V_y & V_{yz} \\ \hline V_{zx} & V_{zy} & V_z \end{array}\right). \tag{5.2}$$

The sub matrices V_x, V_y and V_z are given above and V_{xy}, V_{yz} and V_{xz} are the covariances between the three distributions.

$$\begin{aligned} V_{xy} &= Cov(X, Y) && (5.3) \\ V_{yz} &= Cov(Y, Z) && (5.4) \\ V_{xz} &= Cov(X, Z) \\ &= Cov(X, Y)\, Var(Y)^{-1}\, Cov(Y, Z), && (5.5) \end{aligned}$$

since X is independent of Z given Y. Define B to be the evolution matrix which maps distribution X to distribution Y and D to be the evolution matrix which maps distribution Y to distribution Z. Then the covariances in Equation (5.2) are given by,

$$\begin{aligned} V_{X,Y} &= Cov(X, Y) \\ &= Cov(X, BX + \epsilon) \\ &= Cov(X, BX) \\ &= V_x B', && (5.6) \end{aligned}$$

where ϵ is a white noise factor. Similarly

$$\begin{aligned} V_{Y,Z} &= Cov(Y, Z) \\ &= Cov(Y, DY + \epsilon) \\ &= Cov(Y, DY) \\ &= V_y D', \end{aligned} \tag{5.7}$$

where D is a weight matrix. Using the properties of conditional independence

$$\begin{aligned} V_{X,Z} &= Cov(X, Z) \\ &= Cov(X, Y)\, Var(Y)^{-1}\, Cov(Y, Z) \\ &= V_{xy} V_y^{-1} V_{yz} \\ &= V_x B' V_y^{-1} V_y D' \\ &= V_x B' D'. \end{aligned} \tag{5.8}$$

The coarse model is obtained by using the only the nodes which are parents in the coarse model. In this coarse joint model each node has mean zero and variance matrix obtained by extracting the lines and columns of variance matrix (Equation (5.2)) which relate to the parent nodes. The method above can be applied to Equation (5.1) where the joint distribution over the three time periods is developed. The three covariances required to develop the joint covariance matrix are

$$V_{t-2,t-1} = V_{t-2} B',$$

$$V_{t-1,t} = V_{t-1} D',$$

$$V_{t-2,t} = V_{t-2} B' D'.$$

Using these identities the conditional distribution for the coarse model is obtained by using only the rows and columns of the variance covariance matrix which correspond to nodes which are parents in the coarse model along with the parents that are available at a given point. The 2+1D model is obtained in a similar manner by formulating the second order Markov process for the fine model and then extracting the corresponding coarse model covariance matrix.

5.3.1 Creating a reference table

As in the previous Chapter it is computationally beneficial to have a reference table with the corresponding edge adjustments and conditional variances for various values of temporal dependence, α. Sections 3.5.1 and 3.5.2 showed how these values should be transformed for models with non zero mean or non unit precision. Again consider a reference table for $\alpha \in [0, 1)$ in steps of 0.01. The stationary variance matrix is given by $\Sigma = (I - GG')^{-1}$ where G is the weight matrix for the fine model. The variance matrices V_{t-2}, V_{t-1} and V_t are now defined using the elements from Σ. The covariances are calculated using Equations (5.6), (5.7) and (5.8) where B and D are given weight matrices, thus the joint covariance matrix is known. The adjusted expectation and conditional variance can now be calculated using Equation 5.1 with the associated edge weight for each parent and the conditional precision for each node for each value of temporal dependence being stored in the reference table (see Table A.2).

5.4 1+1D model

Consider the zero mean 1+1D model with unit precision and binomial weights with n spatial nodes and T time periods as defined in Section 2.6.2. The corresponding coarse model for a node at time period t is defined by its parents at time periods $t-2$ and $t-1$. The 1+1D model with second order Markov process is shown in Figure 5.3 the parents of node $\theta_i^{(t)}$ in the coarse model are those nodes which have been circled.

By building the fine model depicted in Figure 5.3 (using all 9 nodes) the distribution of the coarse model can be determined from the corresponding second order fine model where the parents of $\theta_i^{(t)}$ come from both time $t-1$ and time $t-2$

i.e. $pa\left(\tilde{\theta}_i^{(t)}\right) = \left(\tilde{\theta}_{i-1}^{(t-2)}, \tilde{\theta}_i^{(t-2)}, \tilde{\theta}_{i+1}^{(t-2)}, \tilde{\theta}_i^{(t-1)}\right) \equiv \left(\theta_{i-2}^{(t-2)}, \theta_i^{(t-2)}, \theta_{i+2}^{(t-2)}, \theta_i^{(t-1)}\right)$. Once the full joint variance matrix has been determined the rows and columns corresponding to the coarse model can be extracted to form the joint variance matrix for the coarse model for use in the conditional distribution.

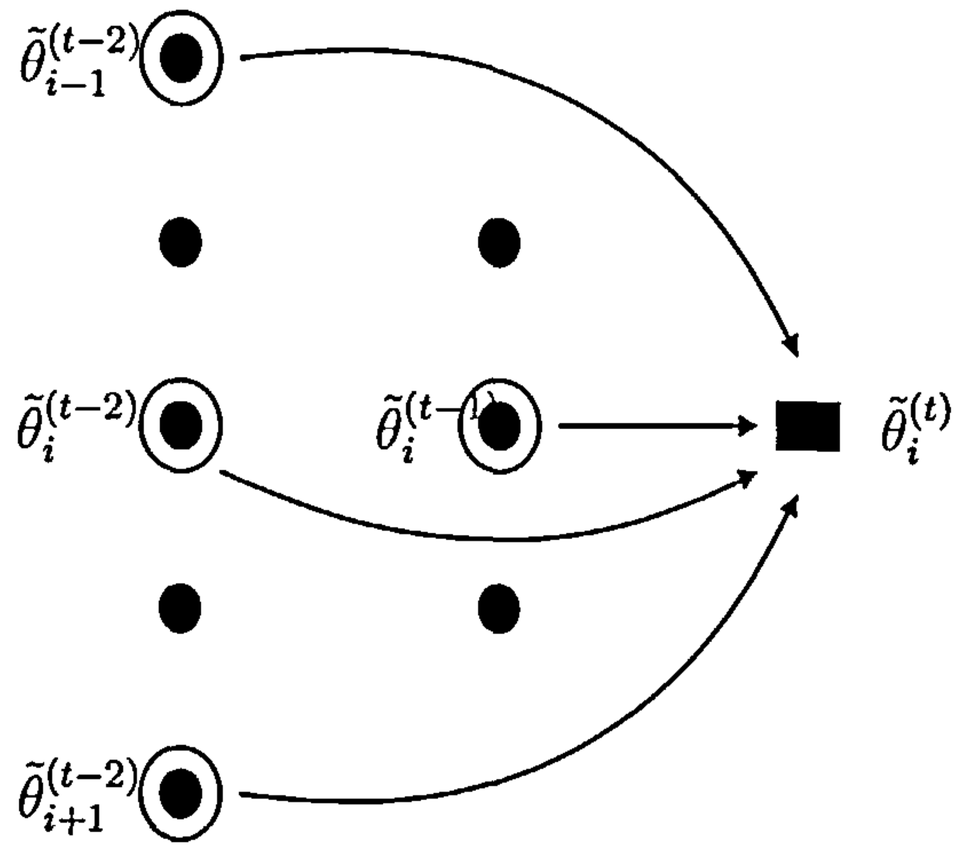


Figure 5.3: *The graphical representation of the 1+1D coarse model which uses the second order Markov process.*

The joint covariance matrix for the fine model is given by

$$\begin{pmatrix} V_{t-2} & V_{t-1,t-2} & V_{t,t-2} \\ & V_{t-1} & V_{t,t-1} \\ & & V_t \end{pmatrix}, \tag{5.9}$$

where V_{t-2}, V_{t-1}, and V_t, are known, hence only the covariances $V_{t-1,t-2}$, $V_{t,t-2}$ and $V_{t,t-1}$ need to be calculated using Equations 5.6, 5.7 and 5.8.

$$\begin{aligned} V_{t-1,t-2} &= V_{t-2}B', \\ V_{t-1,t} &= V_{t-1}D', \\ V_{t-2,t} &= V_{t-2}B'D' \end{aligned}$$

where B and D are isotropic binomial weight matrices for this model, for a given node the weight matrices are given by b and d;

$$b = \frac{\alpha}{4} \begin{pmatrix} 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 2 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 \end{pmatrix} \begin{matrix} \theta_{i-1}^{(t-1)} \\ \theta_{i}^{(t-1)} \\ \theta_{i+1}^{(t-1)} \end{matrix},$$

$$d = \frac{\alpha}{4}\begin{pmatrix} 1 & 2 & 1 \end{pmatrix}.$$

The joint variance matrix is now known for the fine model, the variance matrix for the coarse model is obtained by extracting the values relating to $\theta_{i-2}^{(t-2)}$, $\theta_i^{(t-2)}$, $\theta_{i+2}^{(t-2)}$, $\theta_i^{(t-1)}$ and $\theta_i^{(t)}$. These values can now be used in Equation 5.1, e.g. if the node of interest is in the centre of the system all 5 parents are used to calculate the conditional distribution for the model of interest however along the edges at time $t = 1$, and spatial locations 0 and $n-1$ fewer parents are available, thus the conditional distribution must be adjusted accordingly.

5.4.1 Numerical Example

The model above applied to a grid with 100 spatial nodes and 20 time periods yields the precision surfaces shown in Figure 5.4. Table 5.1 gives the theoretical stationary precision for the corresponding fine model along with the range of values the state precision takes in the simulated system for the coarse model, the final column gives the size of this range. Figure 5.4, along with Table 5.1, shows that the surface is approximately stationary with the size of the range being relatively small 0.001070816 for $\alpha = 0.7$ and 0.04086080 for $\alpha = 0.99$, which are slightly larger than the size of the range given by the fine model using conditional edge adjustments (see Table 3.6).

α	Stationary Precision	Range	Size of range
0.7	0.7713946	0.7724805 0.7735514	0.001070816
0.9	0.5144359	0.5229797 0.5310948	0.008115075
0.95	0.3854864	0.4012195 0.4160731	0.01485352
0.99	0.1867609	0.2365524 0.2774132	0.04086080

Table 5.1: *Stationary precision and the associated range for a 100 spatial node model at time $T = 20$ with coarse edge weights.*

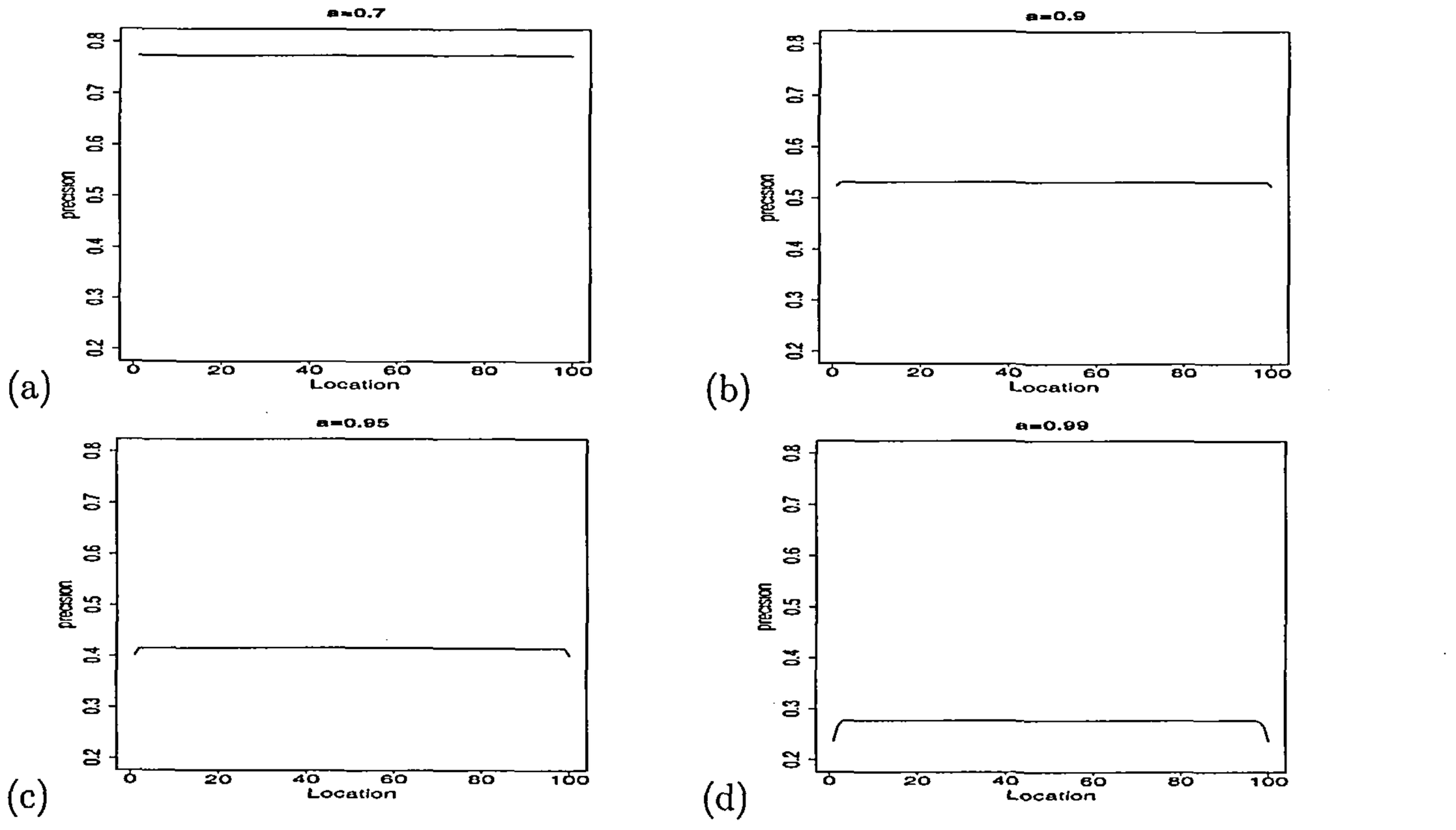


Figure 5.4: *Precision of the 1+1D system for a 100 spatial node model at T=20 using the coarse model for (a)* $\alpha = 0.7$, *(b)* $\alpha = 0.9$, *(c)* $\alpha = 0.95$, *(d)* $\alpha = 0.99$.

5.5 2+1D Model

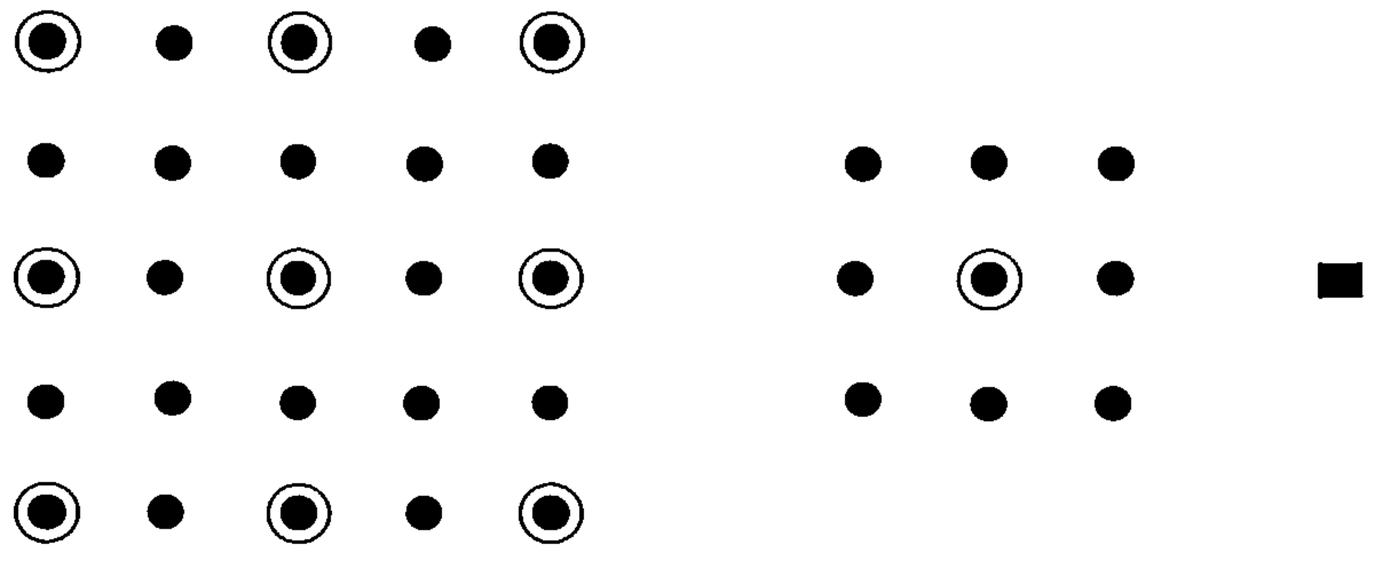


Figure 5.5: *The graphical representation of the 2+1D coarse model which uses the second order Markov process.*

For the 2+1D model the same procedure as in Section 5.4 is used but now the parents of node $\theta_i^{(t)}$ come from a 5×5 grid at time $t-2$ and a 3×3 grid at time $t-1$. The circled

nodes in Figure 5.5 are the parent nodes of interest in the 2+1D coarse model. The joint covariance matrix is calculated in the same way as for the 1+1D model;

$$\begin{aligned} V_{t-1,t-2} &= V_{t-2}B', \\ V_{t-1,t} &= V_{t-1}D', \\ V_{t-2,t} &= V_{t-2}B'D'. \end{aligned}$$

where B and D are weight matrices for the model and matrices b and d are the weight matrices for a given node.

The weight matrix for b is

$$b = \frac{\alpha}{16}\left(\begin{array}{ccc|ccc|ccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
\hline
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
4 & 2 & 0 & 2 & 1 & 0 & 0 & 0 & 0 \\
2 & 4 & 2 & 1 & 2 & 1 & 0 & 0 & 0 \\
0 & 2 & 4 & 0 & 1 & 2 & 0 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 & 0 \\
\hline
1 & 0 & 0 & 2 & 0 & 0 & 1 & 0 & 0 \\
2 & 1 & 0 & 4 & 2 & 0 & 2 & 1 & 0 \\
1 & 2 & 1 & 2 & 4 & 2 & 1 & 2 & 1 \\
0 & 1 & 2 & 0 & 2 & 4 & 0 & 1 & 2 \\
0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 & 1 \\
\hline
0 & 0 & 0 & 1 & 0 & 0 & 2 & 0 & 0 \\
0 & 0 & 0 & 2 & 1 & 0 & 4 & 2 & 0 \\
0 & 0 & 0 & 1 & 2 & 1 & 2 & 4 & 2 \\
0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 & 4 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 2 \\
\hline
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1
\end{array}\right),$$

The weight matrix d is given by,

$$d = \frac{\alpha}{16}\begin{pmatrix} 1 & 2 & 1 & 2 & 4 & 2 & 1 & 2 & 1 \end{pmatrix}.$$

The joint variance matrix is now known for the fine model, the variance matrix for the coarse model is obtained by extracting the values relating to $\theta^{(t-2)}_{i-2,j-2}$, $\theta^{(t-2)}_{i-2,j}$, $\theta^{(t-2)}_{i-2,j+2}$, $\theta^{(t-2)}_{i,j-2}$, $\theta^{(t-2)}_{i,j}$, $\theta^{(t-2)}_{i,j+2}$, $\theta^{(t-2)}_{i+2,j-2}$, $\theta^{(t-2)}_{i+2,j}$, $\theta^{(t-2)}_{i+2,j+2}$ and $\theta^{(t-1)}_{i,j}$. These values can now be used in Equation 5.1, e.g. if the node of interest is in the centre of the system then all 10 parents are used to determine the conditional distribution for that node. The approach used in Chapter 3 to deal with edge effects can be applied to this model using the appropriate number of parents depending on the nodes spatial location.

5.5.1 Numerical Example

The model above applied to a grid with a 10×10 spatial grid and 20 time periods (i.e. the coarse model uses a 5×5 spatial grid) yields the precision surfaces shown in Figure 5.6. This figure along with Table 5.2 shows that the surface is approximately stationary with the size of range being relatively small 0.0002872 for $\alpha = 0.7$ and 0.0279085 for $\alpha = 0.99$, which are slightly larger than the size of the range given by the fine model using conditional edge adjustments (see Table 3.7).

α	Stationary Precision	Range	Size of range
0.7	0.9119323	0.9121857 : 0.9124729	0.0002872
0.9	0.8039998	0.8076038 : 0.8116600	0.0040562
0.95	0.7429752	0.7518639 : 0.7614151	0.0095512
0.99	0.627286	0.6571248 : 0.6850333	0.0279085

Table 5.2: *Stationary precision and the associated range for a 10 × 10 spatial model at time $T = 20$ using the coarse model.*

5.6 MCMC Example

In this Section examples for both the 1+1D and 2+1D model are presented. Both examples use the same generated data as in the examples from Chapter 4 to allow comparisons

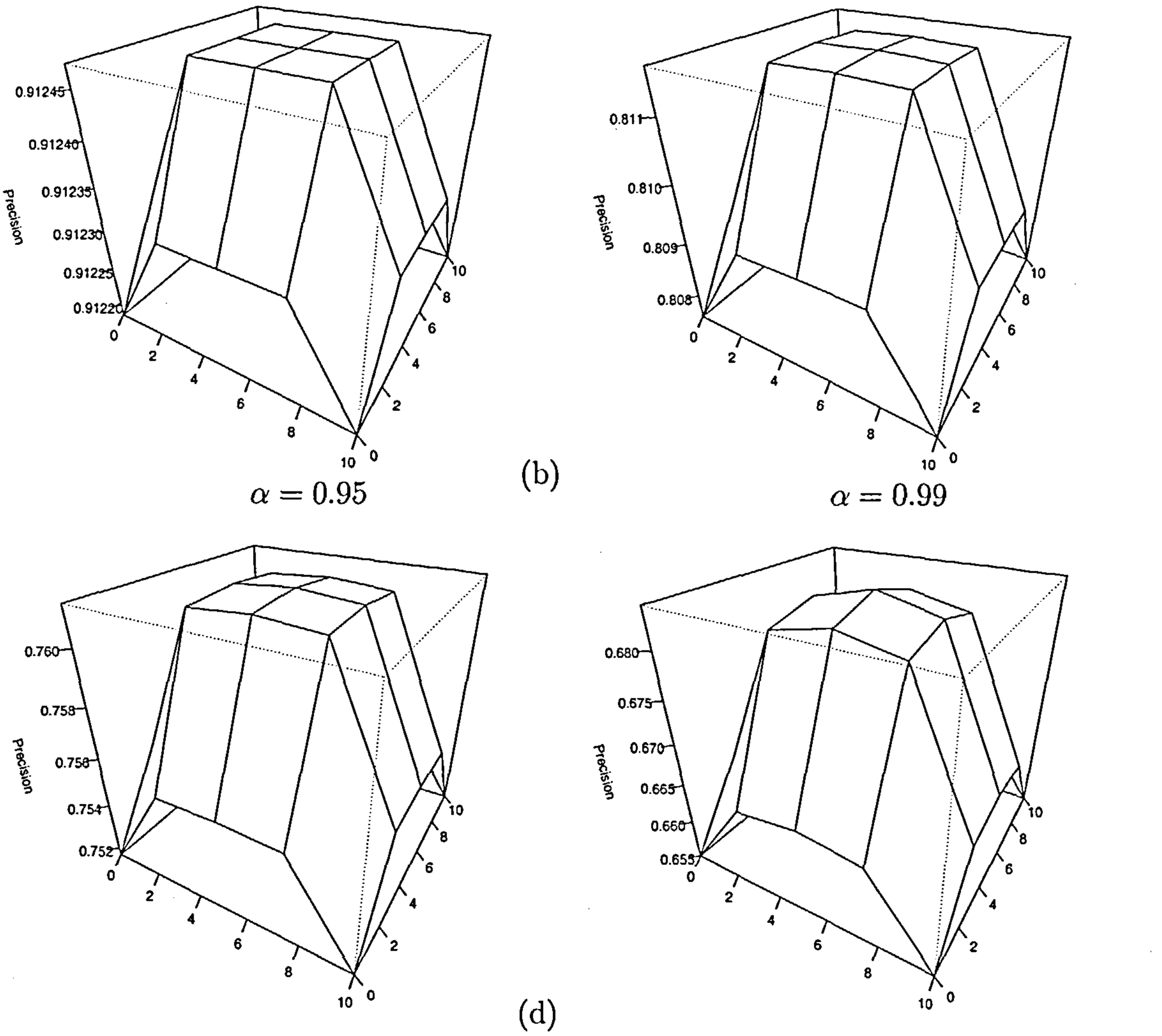


Figure 5.6: *Precision of the system on a* 10×10 *spatial grid at time* $T = 20$ *using the coarse model for (a)* $\alpha = 0.7$*, (b)* $\alpha = 0.9$*, (c)* $\alpha = 0.95$*, (d)* $\alpha = 0.99$.

between the fine and coarse methods when simulating from the posterior distribution of the precision parameters. This coarse approach will hopefully provide a good estimate for the precision parameters whilst reducing the computational time.

5.6.1 1+1D model

Consider a 1+1D model with 100 spatial nodes evolving over 20 time periods. The data was generated from a model with $\tau_s = 0.2$, $\tau_o = 0.1$, $\mu = 0.0$ and $\alpha = 0.7$. In order to use

the coarse model on a 50 spatial node grid evolving over 20 time periods the data must be mapped to the coarse grid. This is done by mapping the data to the nearest coarse grid point prior to carrying out the simulations. The mean of the data and the temporal dependence are considered known whilst the two precisions are considered unknown and are simulated using the Metropolis-Hastings method discussed in Section 4.4. A Gamma prior is used for both precision values, $\Gamma(0.01, 0.01)$. The random walk update for the two parameters where τ_i^* is the proposed value is given by,

$$\tau_s^* = \tau_s + x,$$

$$\tau_o^* = \tau_o + y$$

where

$$x \sim N(0, 100^{-1}),$$

$$y \sim N(0, 400^{-1}).$$

The memory used by the program for the coarse model is 16 Megabytes whereas the memory used by the fine model was 31 Megabytes, which results in a a considerable reduction in the computational time for the model if the coarse approach is adopted. The results for this model are shown in Figure 5.7. The mixing in this model is good however the chain has not converged on the correct values, $\log(\tau_s) = -1.6$ and $\log(\tau_o) = -2.3$.

Table 5.3 shows the summary statistics for the generated data, the log of the inverse variance is -2.77. Examination of Figure 5.7 suggests that the variance of the data has pulled the observational precision away from the true value. This is probably due to the increased number of observations at each spatial location, in this 1+1D model there are two observations at each spatial location in each time period. One approach is to consider thinning the data to the same size as the coarse grid. Repeating the above MCMC simulation with the coarsened data yields the trace and density plots in Figure 5.8. These plots show good mixing and the true parameter values lie within the range of the posterior distribution.

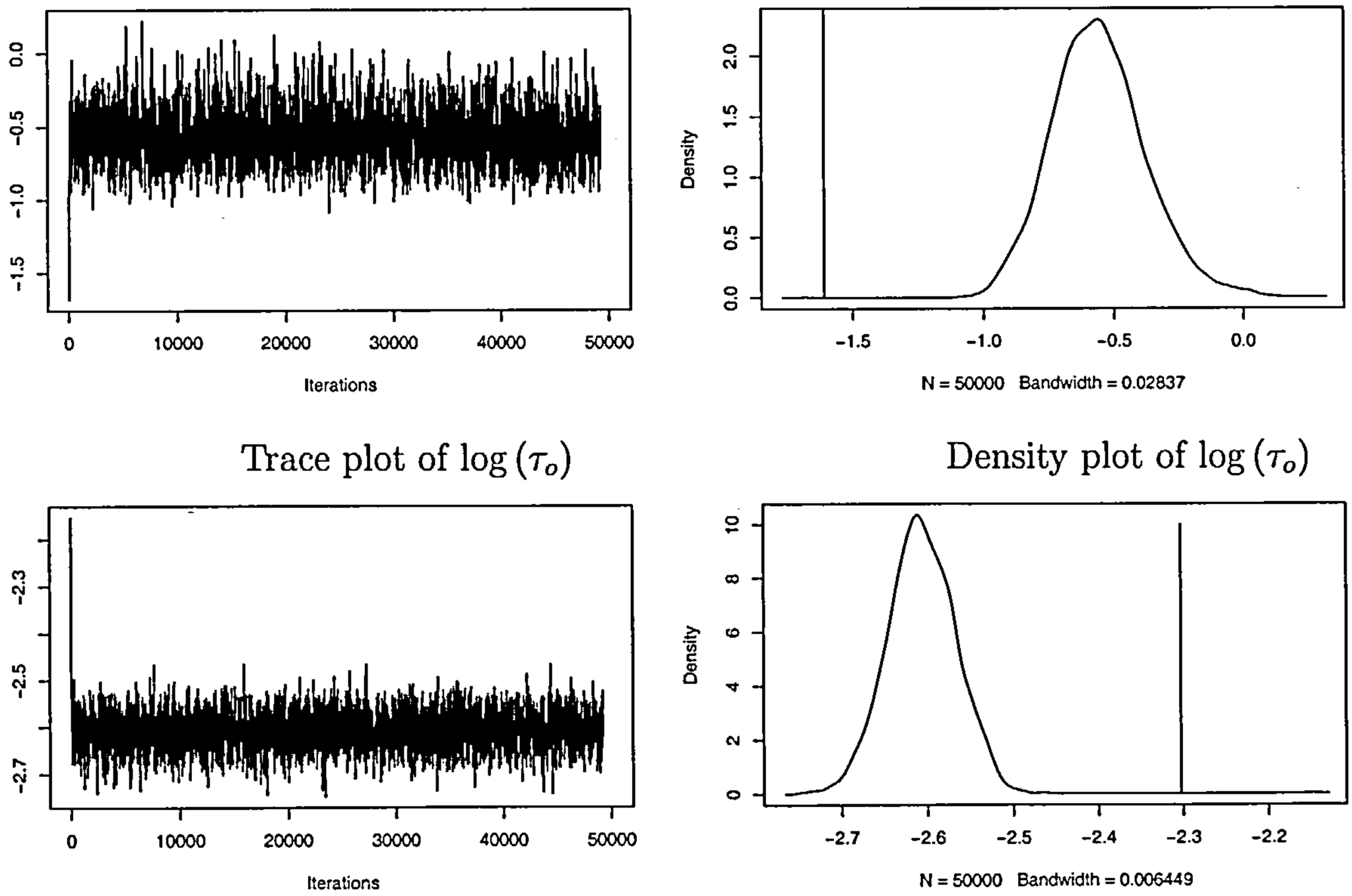


Figure 5.7: *The trace and density plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the coarse model with a complete set of data.*

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	Var
-13.95000	-2.81400	-0.06284	-0.11390	2.61300	12.51000	15.97770

Table 5.3: *Summary statistics for the generated 1+1D data.*

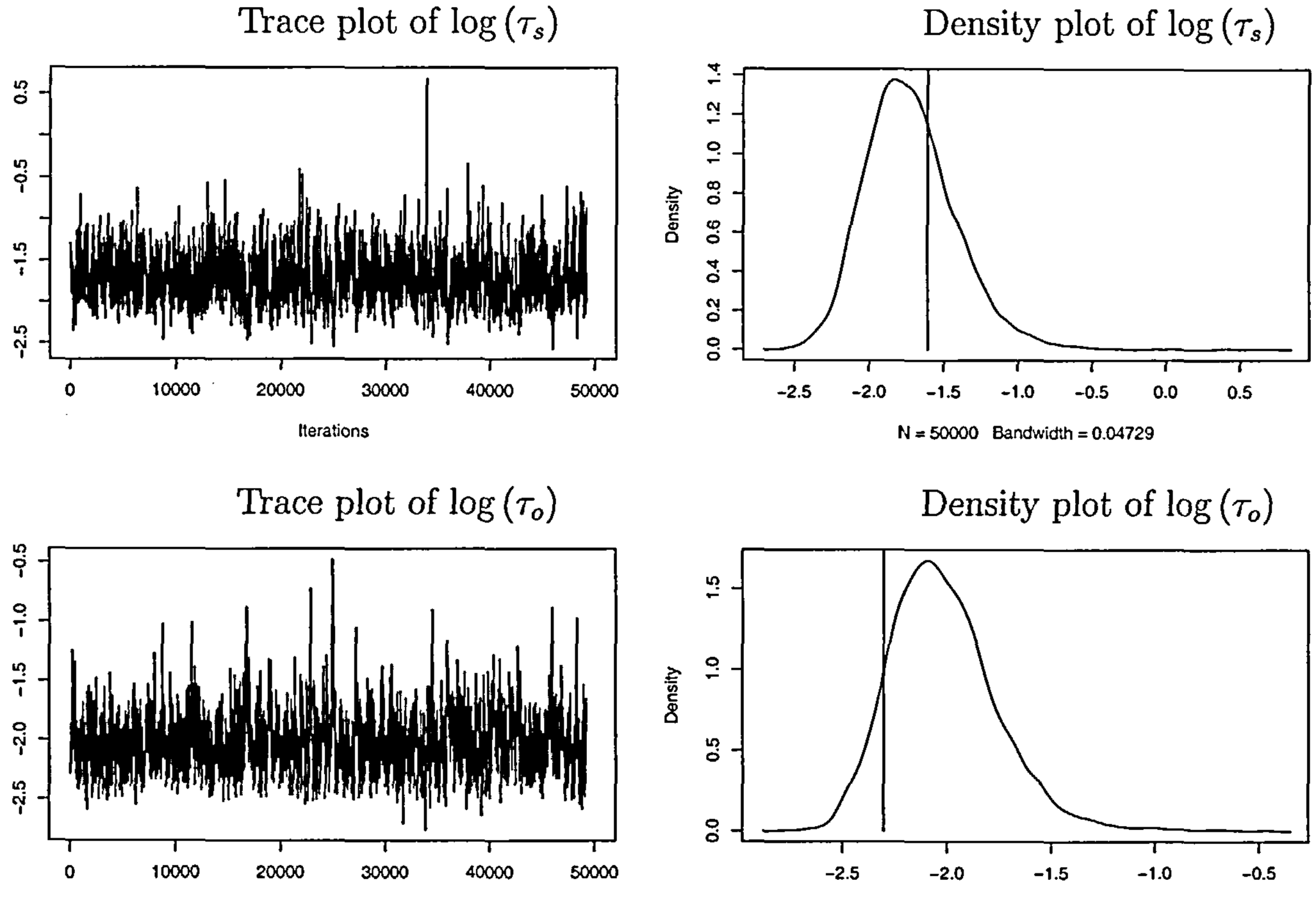


Figure 5.8: *The trace and density plots for the two precision parameters τ_s and τ_o for the 1+1D model with 100 spatial nodes evolving over 20 time periods generated using the coarse model with thinned data.*

The reduction of grid points using the coarse model reduces the computational time required for the MCMC simulations, however for the simulations to converge in the right area the number of observations should not exceed the total number of grid points in the model. Tables 5.4 and 5.5 show summary statistics and quartiles corresponding to the model in Figure 5.8, from these results the model suggests that the parameter estimates for this model are $\log(\tau_s) = -1.7$ and $\log(\tau_o) = -2.0$. Using `GDAGsim` the corresponding fine model model on a 100 spatial node and twenty time period grid can be built with these estimates. The mean of the underlying latent model at time period $t = 4$ is shown in Figure 5.9. The data was originally generated from a random distribution hence the random dispersion of the latent means through time is the expected distribution for the

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.730	0.3058	0.001368	0.002376
$\log(\tau_o)$	-2.018	0.2585	0.001156	30.526755

Table 5.4: *The mean and standard deviation for the precision parameters along with the time series standard error for the 1+1d model with 100 spatial nodes evolving through 20 time periods using the coarse model with thinned data.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-2.243	-1.939	-1.758	-1.554	-1.058
$\log(\tau_o)$	-2.443	-2.196	-2.046	-1.875	-1.435

Table 5.5: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 1+1d model with 100 spatial nodes evolving through 20 time periods using the coarse model with thinned data.*

latent structure. Similar plots from other time periods showed a similar random dispersion as did the plot for predictive time $t = 20$.

5.6.2 2+1D model (1)

Consider a 2+1D model with 100 spatial nodes on a 10×10 spatial grid evolving over 20 time periods. The data was generated from a model with $\tau_s = 0.2$, $\tau_o = 0.1$, $\mu = 0.0$ and $\alpha = 0.7$. In order to use the coarse model on a 5×5 spatial grid evolving over 20 time periods the data must be mapped to the coarse grid. This is done by mapping the data to the nearest spatial coarse grid point prior to carrying out the simulations. The mean of the data and the temporal dependence are considered known whilst the two precisions are considered unknown and are simulated using the Metropolis-Hastings method discussed in Section 4.4. A Gamma prior is used for both precision values, $\Gamma(0.01, 0.01)$. The random

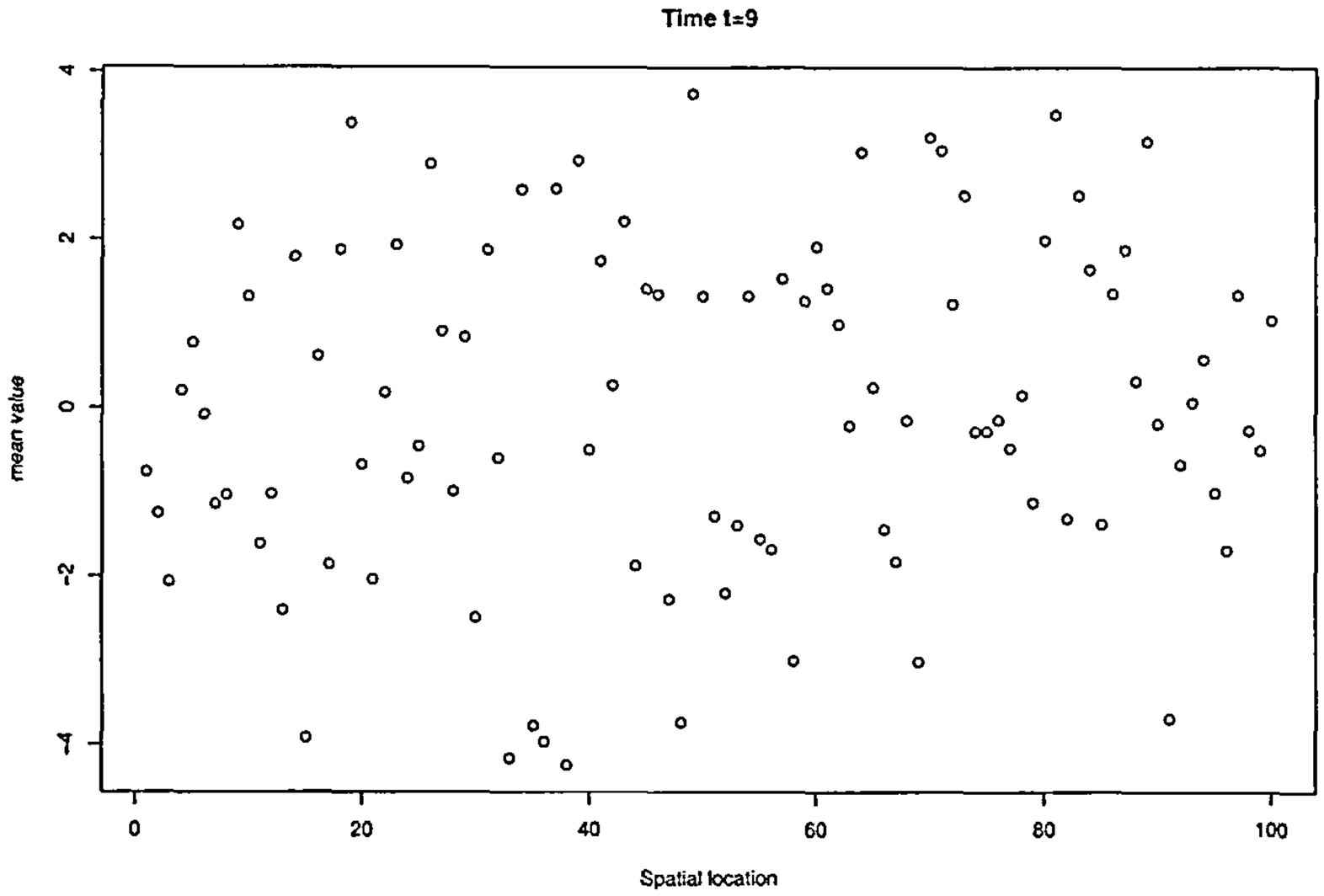


Figure 5.9: *Plots of the mean of the simulated latent structure at time period* $t = 9$.

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.873	0.2505	0.003653	0.00651
$\log(\tau_o)$	-1.985	0.2423	0.003534	0.04177

Table 5.6: *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 10×10 *spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.*

walk update for the two parameters where τ_i^* is the proposed value is given by,

$$\tau_s^* = \tau_s + x,$$

$$\tau_o^* = \tau_o + y$$

where

$$x \sim N(0, 100^{-1}),$$

$$y \sim N(0, 400^{-1}).$$

This coarse model is computationally much smaller than the fine model (5 Megabytes compared with 32 Megabytes). Figure 5.10 shows the MCMC simulation for the model

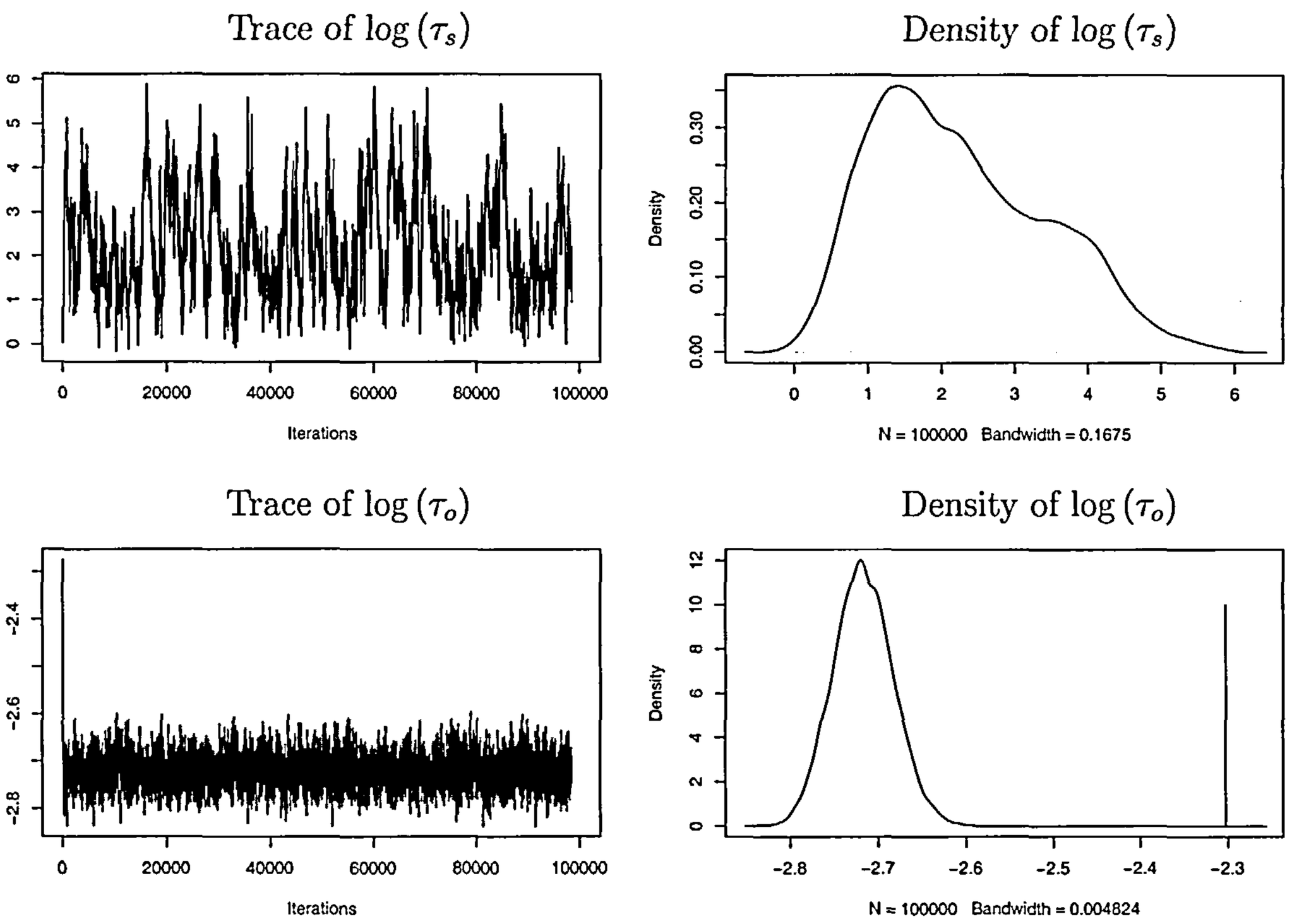


Figure 5.10: *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model on a 10×10 spatial spatial evolving over 20 time periods generated using the coarse model with a complete set of data.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-2.301	-2.050	-1.900	-1.717	-1.319
$\log(\tau_s)$	-2.369	-2.156	-2.011	-1.854	-1.406

Table 5.7: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a 10×10 spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.*

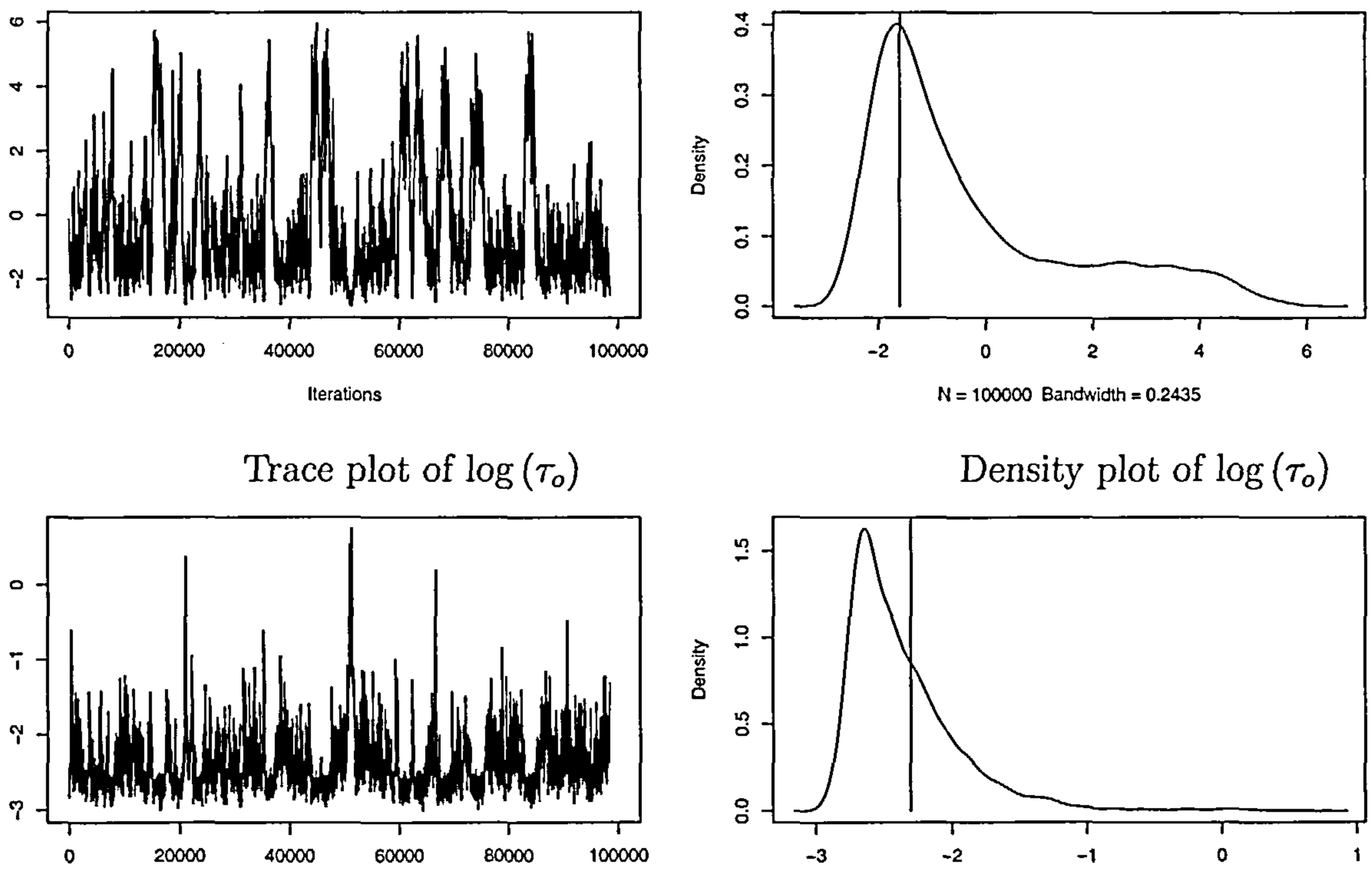


Figure 5.11: *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model on a* 10×10 *spatial spatial evolving over 20 time periods generated using the coarse model with thinned data.*

with a complete set of data, as in the 1+1D model the chains do not converge to the actual parameter values. Figure 5.11 shows the corresponding trace and density plots for the model with the thinned data where only one observation for each coarse grid point is used. Due to the small size of this coarse model (5×5 spatial grid evolving over 20 time periods) the MCMC simulation is not perfect and better results would be expected from a larger model. Examination of Figure 5.11 suggests parameter estimates of $\log(\tau_s) = -1.6$ and $\log(\tau_o) = -2.4$, these estimates were chosen by examining the posterior distributions and selecting the mode of the model. The parameter estimates given by the summary statistics and quartile ranges in Tables 5.7 and 5.6 would suggest parameter estimates which are slightly different. Using these values from Figure 5.11 the

underlying latent structure can be generated along with the surface plots of the mean of the latent structure as shown in Figure 5.12. These plots show that the mean values are randomly spread through space as expected and there are no obvious patterns in the data. The predictive surface plot for time $t = 20$ (Figure 5.13) is much smoother as there is no information from data at that time period available.

5.6.3 2+1D model (2)

Consider a 2+1D model with 400 spatial nodes on a 20×20 spatial grid evolving over 20 time periods. The data was generated from a model with $\tau_s = 0.2$, $\tau_o = 0.1$, $\mu = 0.0$ and $\alpha = 0.7$. In order to use the coarse model on a 10×10 spatial grid evolving over 20 time periods the data must be mapped to the coarse grid. This is done by mapping the data to the nearest spatial coarse grid point, the data is also thinned to the size of the latent grid prior to carrying out the simulations. The mean of the data and the temporal dependence are considered known whilst the two precisions are considered unknown and are simulated using the Metropolis-Hastings method discussed in Section 4.4. A Gamma prior is used for both precision values, $\Gamma(0.01, 0.01)$. The random walk update for the two parameters where τ_i^* is the proposed value is given by,

$$\tau_s^* = \tau_s + x,$$

$$\tau_o^* = \tau_o + y$$

where

$$x \sim N(0, 25^{-1}),$$

$$y \sim N(0, 25^{-1}).$$

The results for the MCMC are given in Figure 5.14 and Tables 5.8 and 5.9 for this model, the distribution is less skewed than for the model with a 10×10 spatial grid evolving over 20 time periods and the parameter estimates form the posterior distribution for this model are $\log(\tau_s) = -1.9$ and $\log(\tau_o) = -2.0$. Again these parameters can be used to

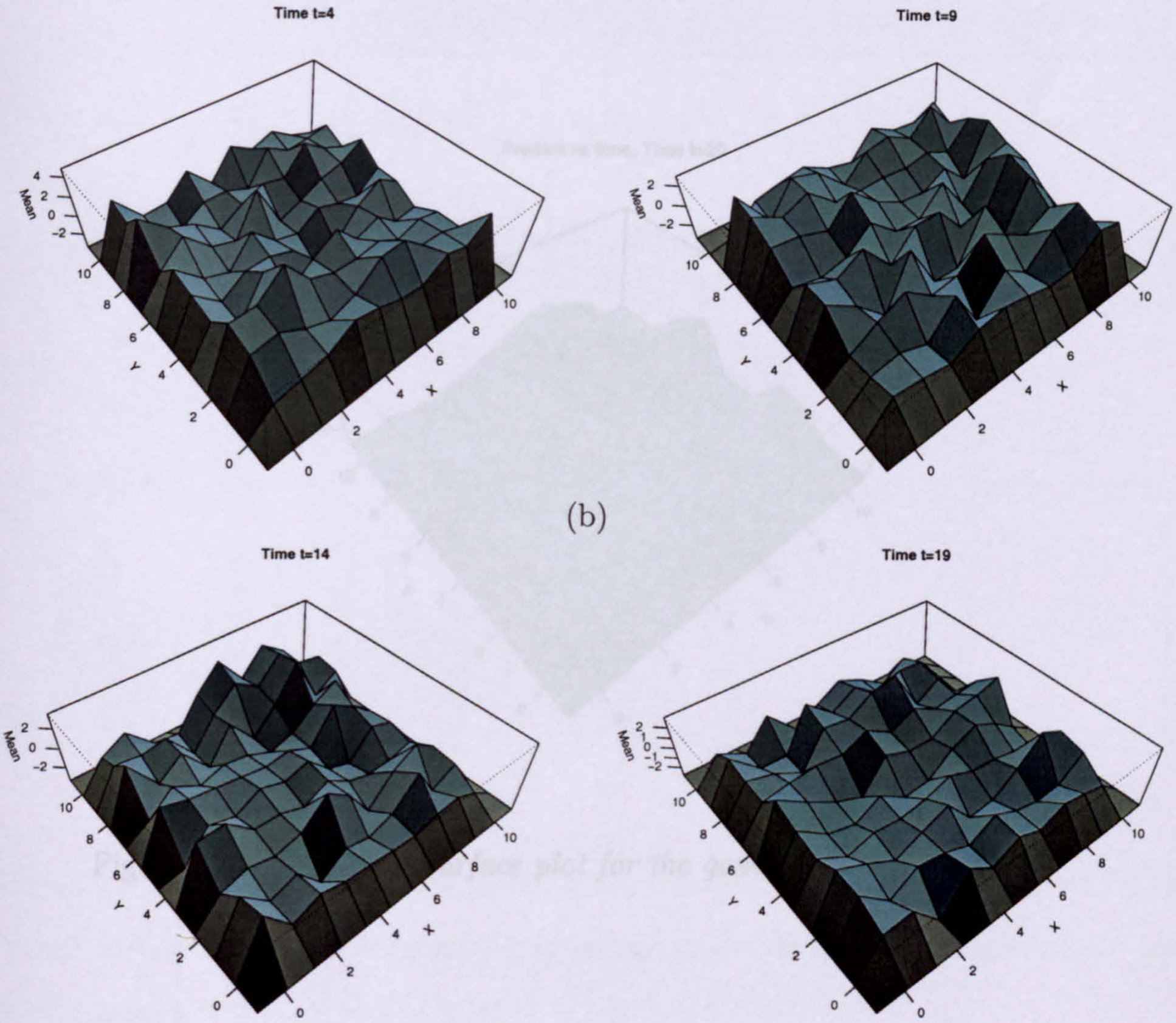


Figure 5.12: *Surface plots generated on the fine* 10×10 *spatial grid using the estimate from the coarsened model at (a) Time period 0, (b) Time period 4, (c) Time period 14 and (d) Time period 19.*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.871	0.2520	0.008956	0.01621
$\log(\tau_o)$	-1.986	0.2432	0.008641	0.01560

Table 5.8: *The mean and standard deviation for the precision parameters along with the time series standard error for the 2+1D model with a* 20×20 *spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.*

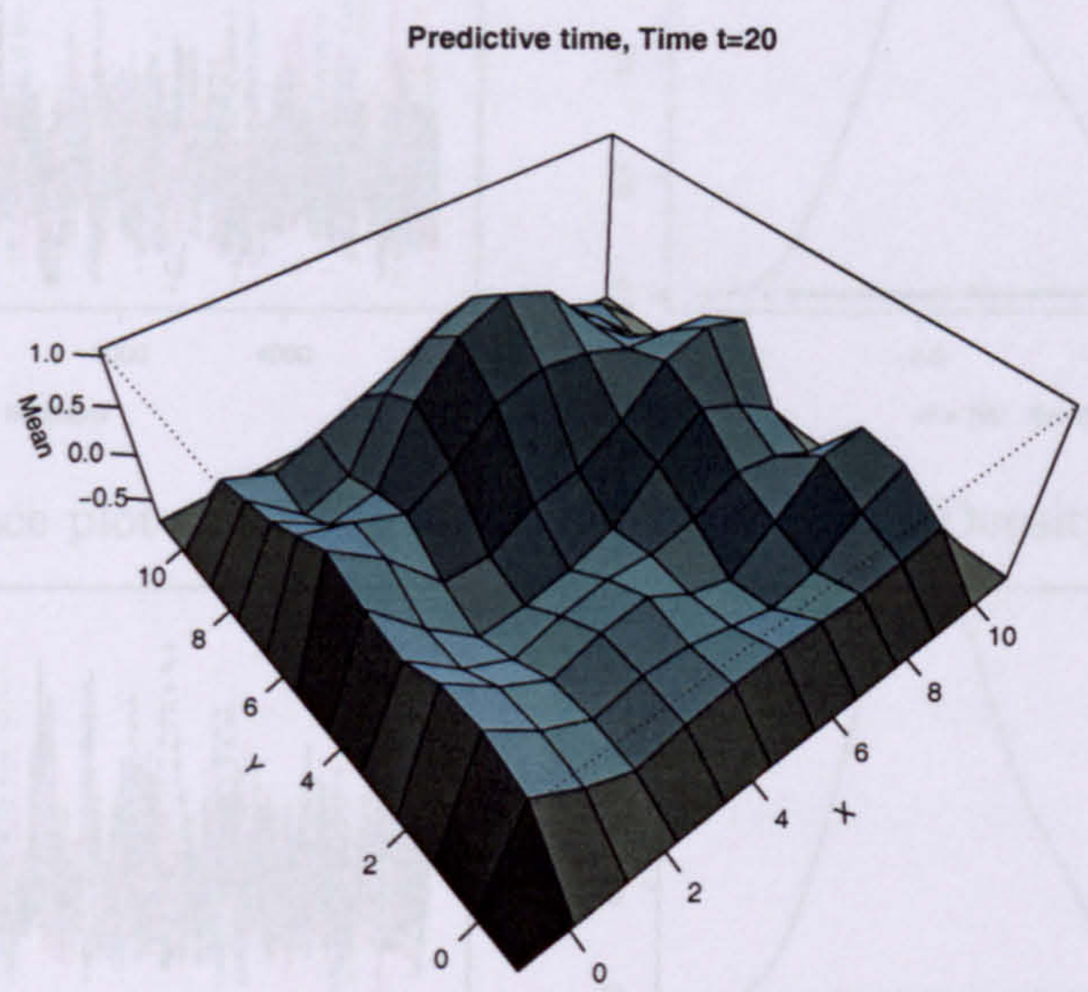


Figure 5.13: *Predictive surface plot for the generated* 10×10 *model.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-2.301	-2.050	-1.896	-1.715	-1.311
$\log(\tau_s)$	-2.371	-2.159	-2.012	-1.852	-1.409

Table 5.9: *The 2.5%, 25%, 50%,75% and 97.5% quantiles for the precision parameters for the 2+1D model with a* 20×20 *spatial grid evolving through 20 time periods using a parallel approach with eight chains each of 5000 iterations.*

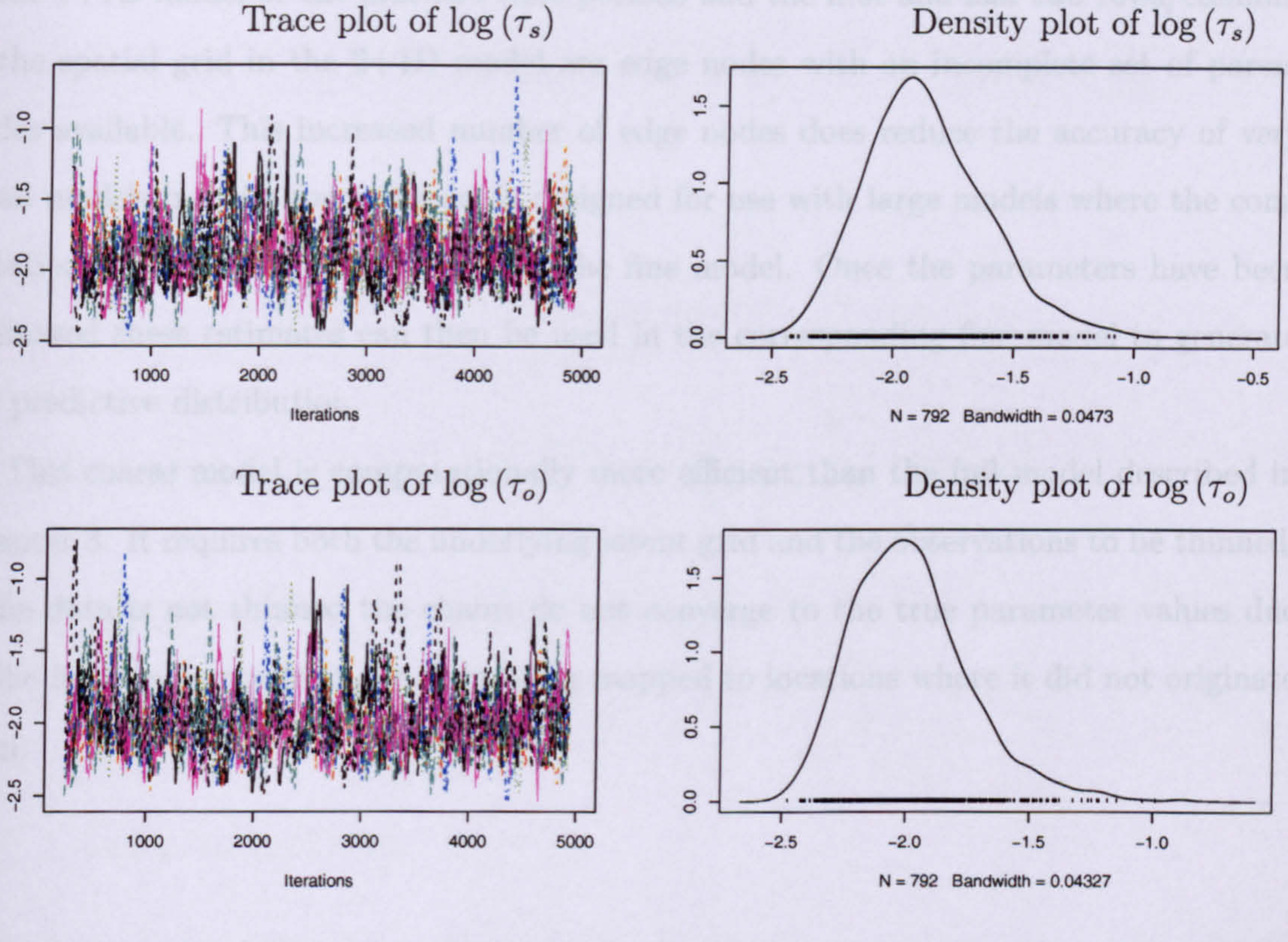


Figure 5.14: *The trace and density plots for the two precision parameters τ_s and τ_o for the 2+1D model with a 20×20 spatial grid evolving over 20 time periods generated using a parallel approach with eight chains each of 5000 iterations.*

generate mean surface plots of the latent structure but as in the previous model the data is generated randomly across the latent structure and no obvious patterns in the data can be seen.

5.7 Discussion

The model is developed in the same way as the fine model in Chapter 3 with the development of a reference table for edge weights and associated conditional precisions for each location within the model. Edge effects are increased in this model due to the increased number of nodes which are classified as edge nodes with the introduction of the second order Markov process; the first two time periods and the two nodes at each edge

of the 1+1D model or the first two time periods and the first and last two rows/columns of the spatial grid in the 2+1D model are edge nodes with an incomplete set of parent nodes available. This increased number of edge nodes does reduce the accuracy of very small models but the coarse model is designed for use with large models where the computation time is unacceptably long for the fine model. Once the parameters have been estimated these estimates can then be used in the corresponding fine model to generate the predictive distribution.

This coarse model is computationally more efficient than the full model described in Chapter 3. It requires both the underlying latent grid and the observations to be thinned, if the data is not thinned the chains do not converge to the true parameter values due to the large influence from the data being mapped to locations where it did not originate from.

Chapter 6

Examination of Temporal and Spatial Dependencies

6.1 Introduction

In Chapter 4 a number of 1+1D and 2+1D models were examined using three MCMC methods for model simulation. The final example estimated both precision parameters and the mean parameter using the Metropolis-Hastings method. This chapter extends the model further by using the Metropolis-Hastings method to simulate the precision parameters, the temporal parameter α and an additional spatial parameter β. This chapter is concerned with whether the spatial and temporal dependency parameters can be determined from the data or if they are too weakly identified to be estimated empirically.

To develop the DLM for this model consider the 2+1D scaled edge weight model from Chapter 3 where the weights along the edge of the system are scaled so that they sum to 1. The spatial and temporal dependence parameters are initially examined separately and then this chapter concludes with an example that considers updating both of these dependency parameters together.

6.2 Temporal parameter

The temporal parameter α has been the only parameter controlling spatial and temporal dependence in the models examined previously. The model under consideration is again defined by Equation (2.1) and the evolution matrix G is defined as in Equation (2.9) by the elements of g. For a given node the weights are given by

$$g = \alpha \begin{pmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{pmatrix} \tag{6.1}$$

where α is the temporal dependence parameter. This model has a "canonical" spatial dependence, i.e. for a fixed degree of temporal dependence, the degree of spatial dependence is fixed.

6.2.1 Example

Consider a 2+1D model with scaled edge weights, known mean but unknown precision and temporal dependence. The Metropolis-Hastings method is used to simulate the unknown variables, as in Chapter 4 the prior distribution for the precision parameter is a *Gamma* distribution. The prior distribution for the temporal dependence is chosen to be a *Beta* distribution since $\alpha \in [0, 1)$,

$$\begin{aligned} \tau_s &\sim \Gamma(0.01, 0.01), \\ \tau_o &\sim \Gamma(0.01, 0.01), \\ \alpha &\sim Beta(a, b). \end{aligned}$$

The model examined is generated on a 5×5 spatial grid over ten time points, five different prior distributions for α are considered, $\alpha \sim Beta(1, 1)$, $\alpha \sim Beta(2, 3)$, $\alpha \sim Beta(3, 2)$, $\alpha \sim Beta(1, 5)$ and $\alpha \sim Beta(5, 1)$. The model uses generated data with parameters $\tau_s = 0.2$, $\tau_o = 0.1$ and $\alpha = 0.7$ which are to be simulated and $\mu = 0$ is considered known.

Figures 6.1 to 6.5 show the results for the five different prior distributions applied to the temporal dependence parameter. The vertical line on each of the density plots represents

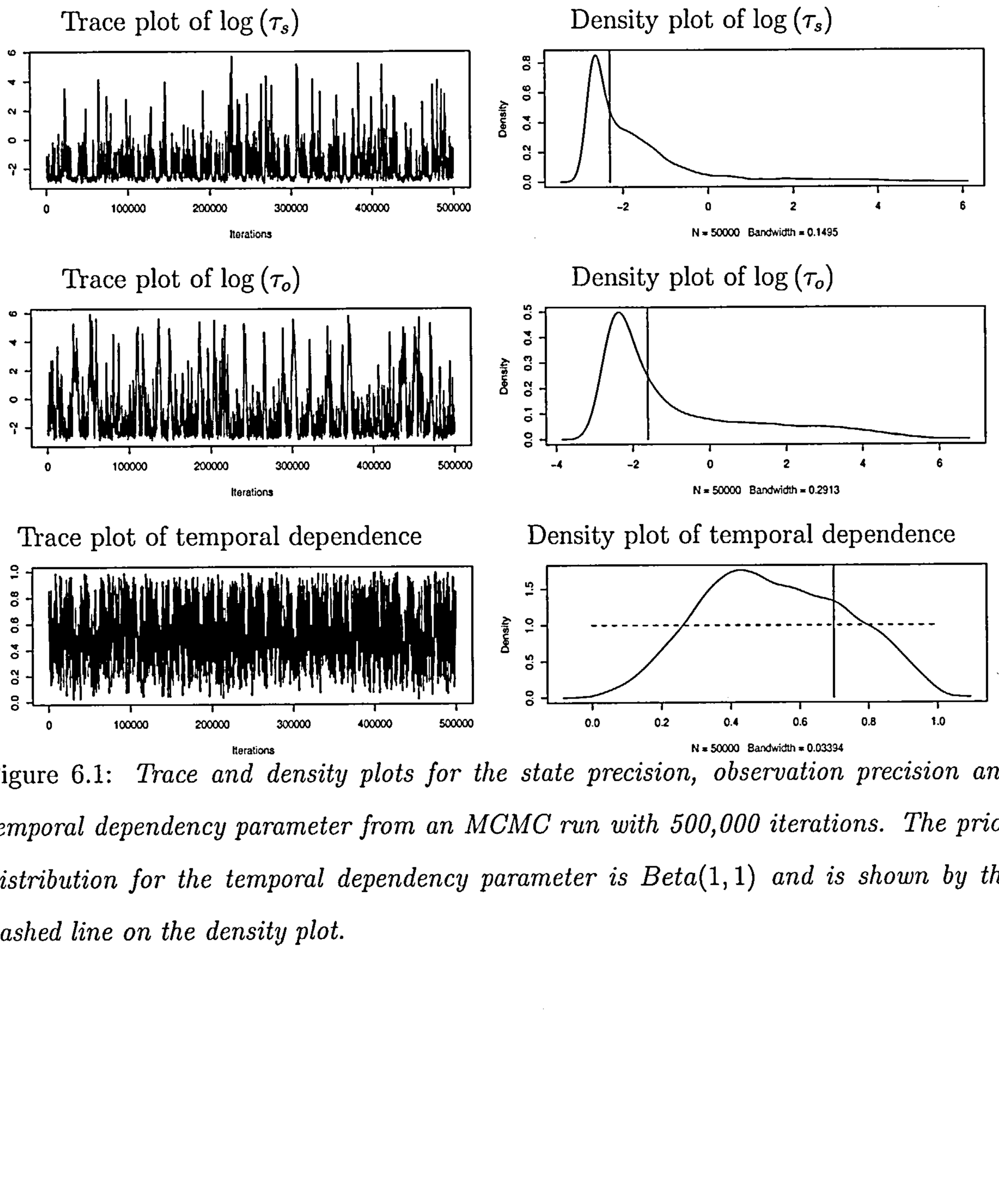


Figure 6.1: *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is Beta*$(1,1)$ *and is shown by the dashed line on the density plot.*

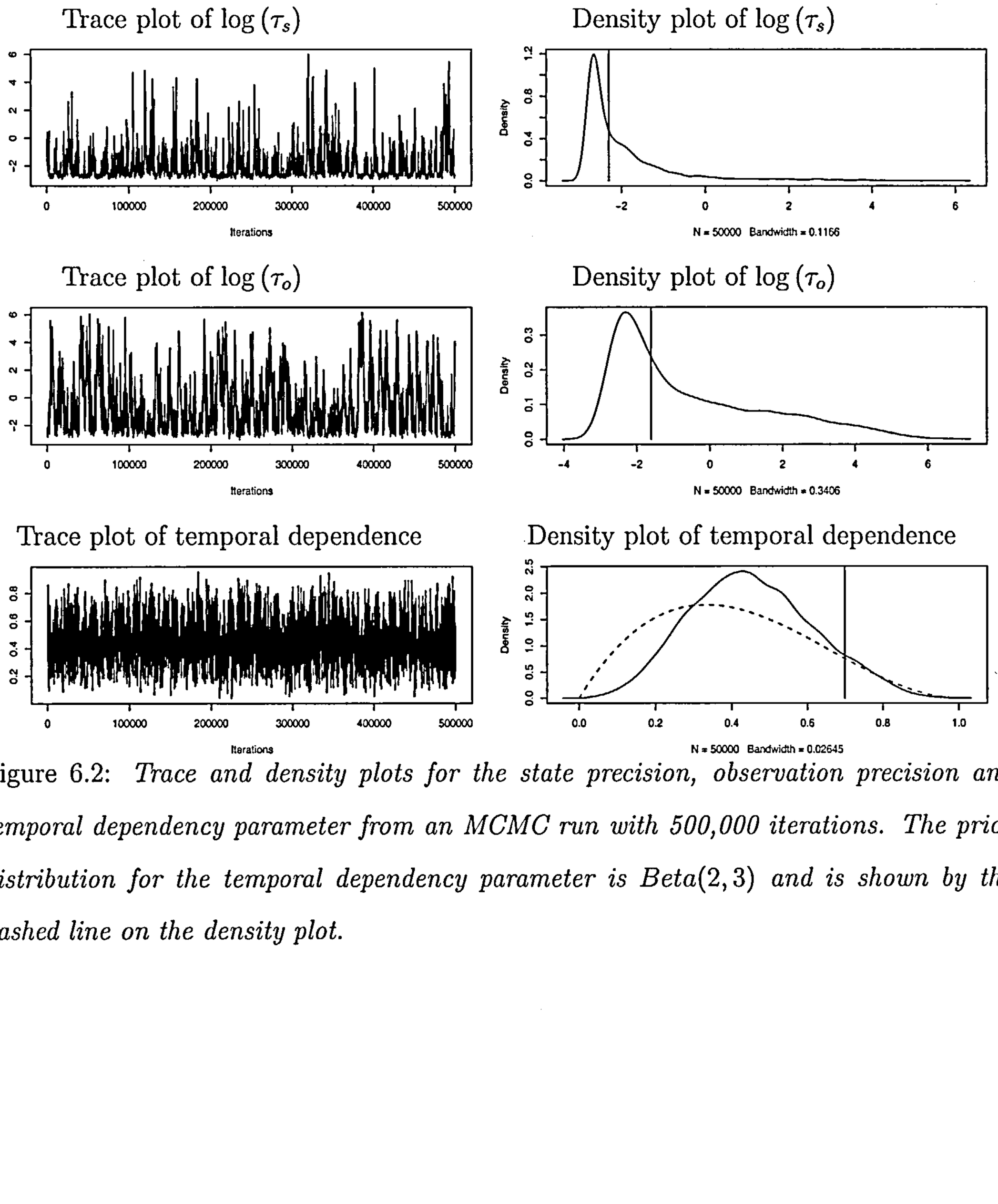


Figure 6.2: *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is Beta*$(2,3)$ *and is shown by the dashed line on the density plot.*

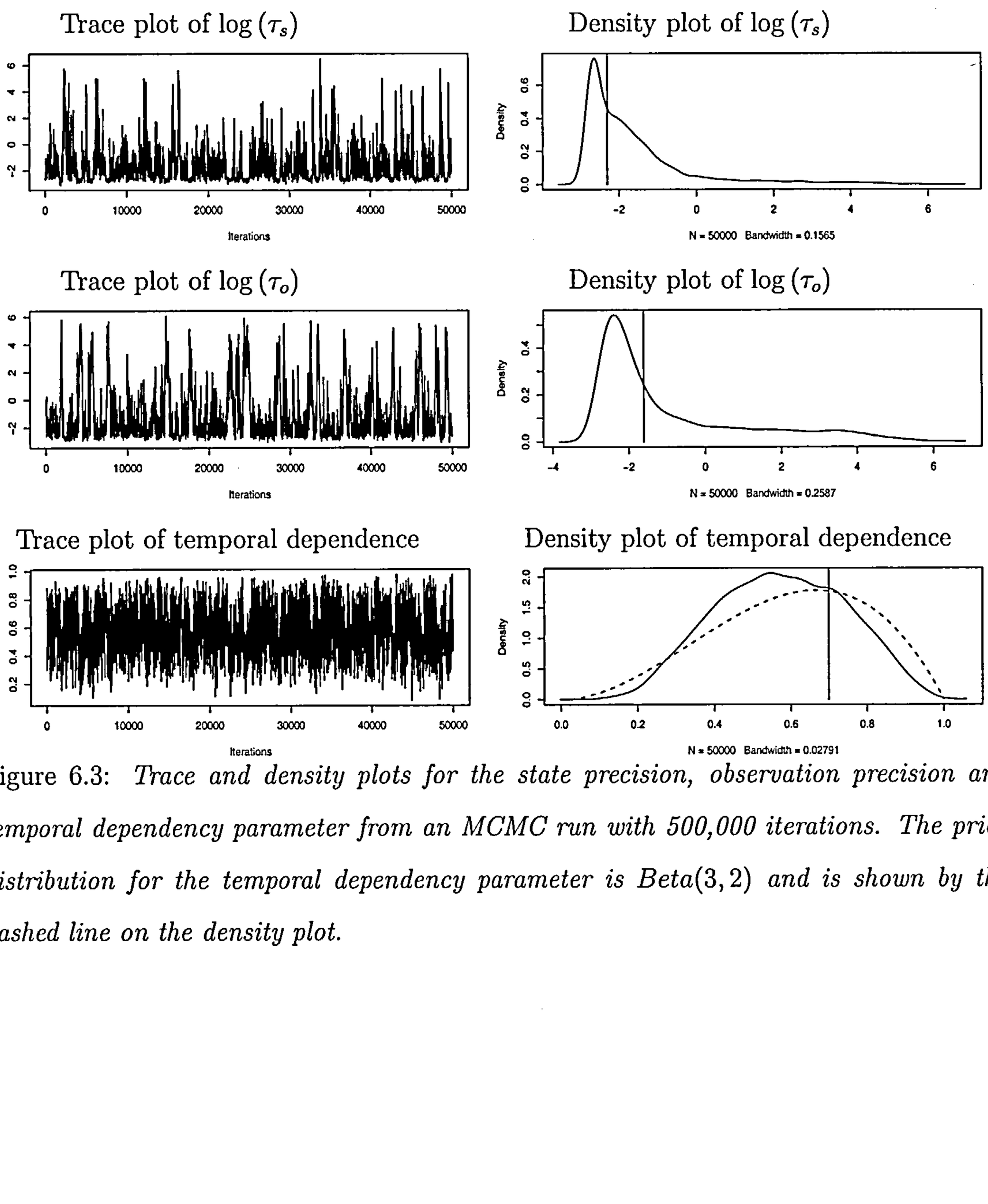


Figure 6.3: *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is Beta*$(3, 2)$ *and is shown by the dashed line on the density plot.*

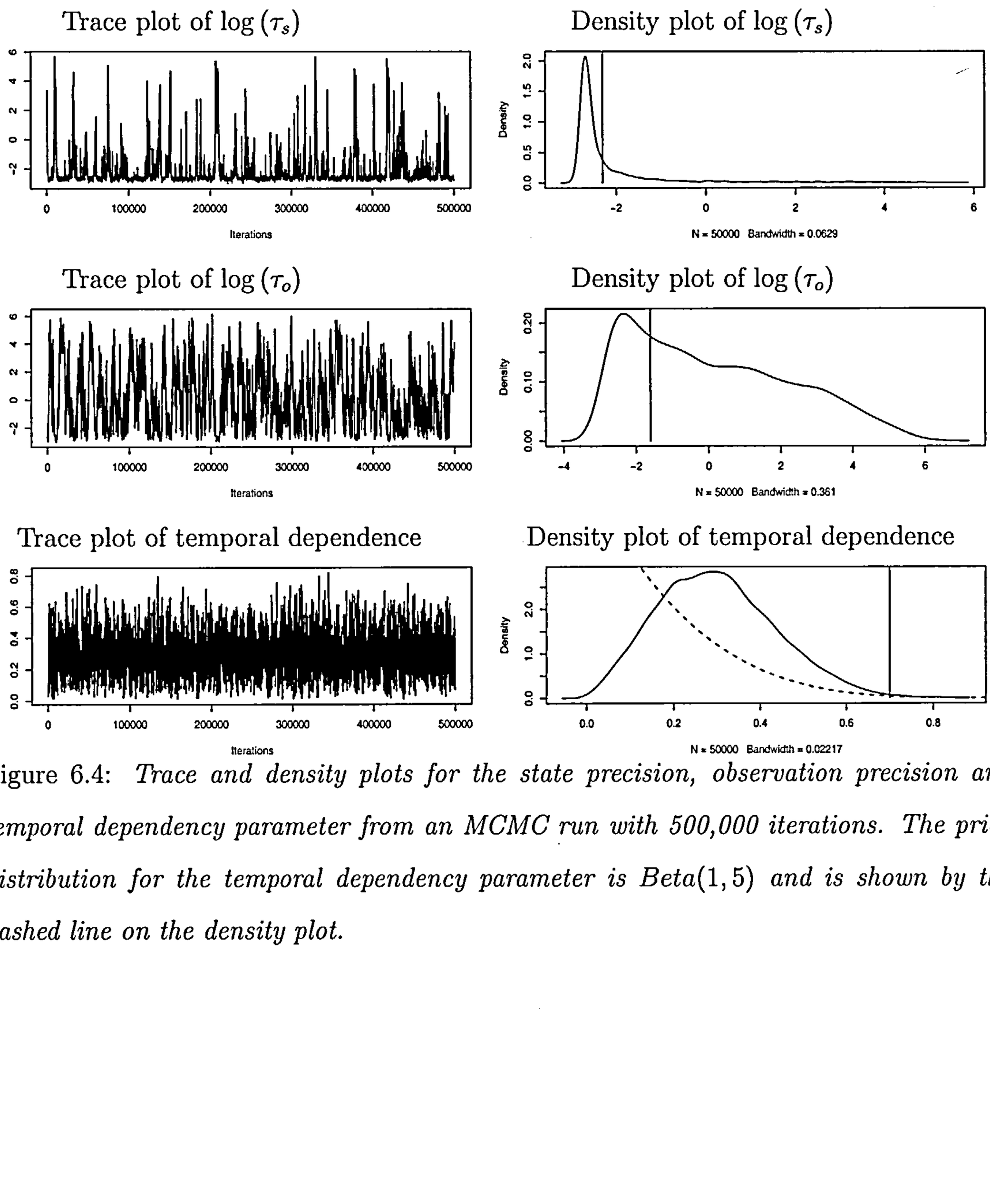


Figure 6.4: *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is* $Beta(1,5)$ *and is shown by the dashed line on the density plot.*

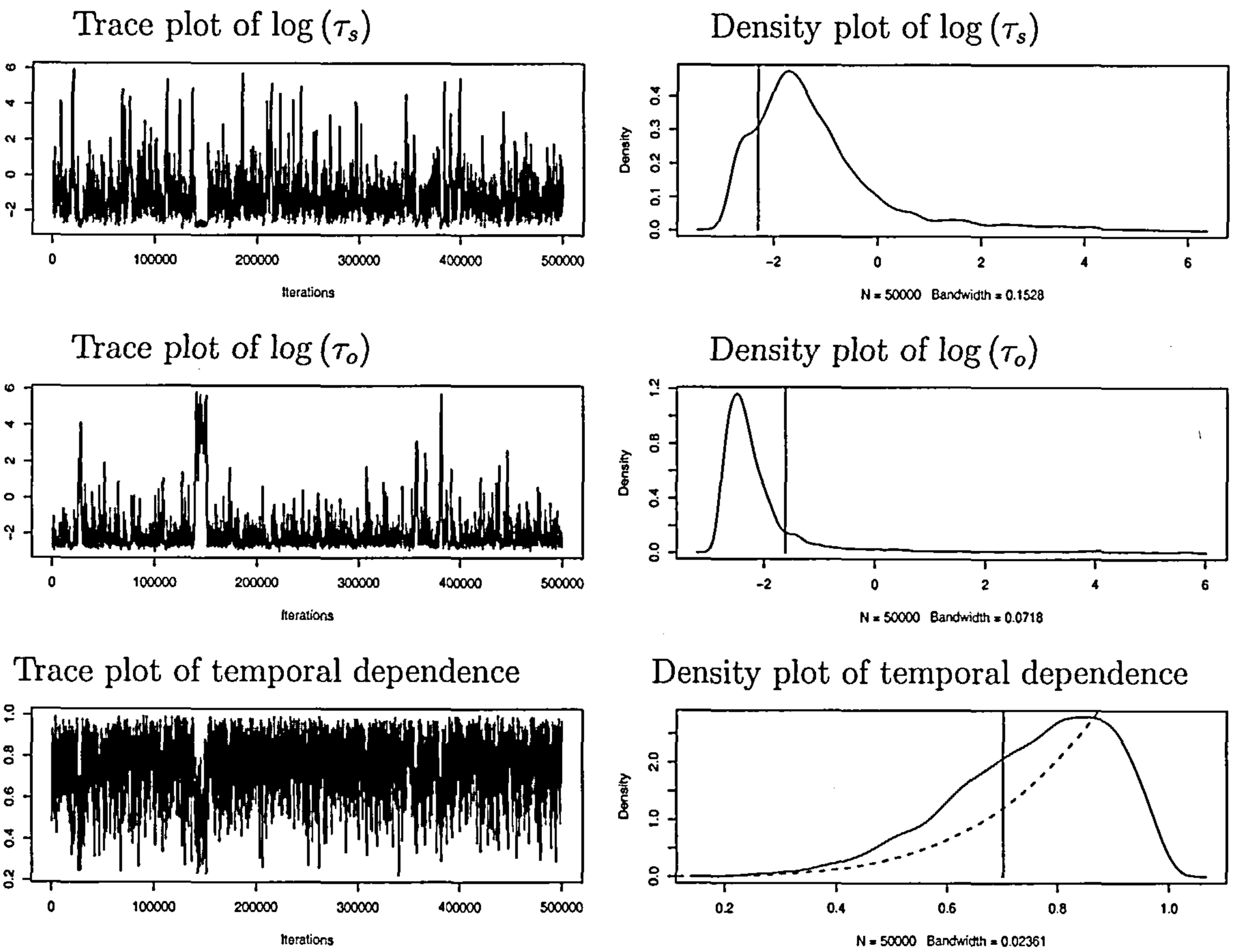


Figure 6.5: *Trace and density plots for the state precision, observation precision and temporal dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the temporal dependency parameter is* $Beta(5, 1)$ *and is shown by the dashed line on the density plot.*

the true parameter value from the simulated data and the dashed line on the density plot for the temporal dependence parameter represents the prior distribution. These figures show that there are some mixing problems in the models. The trace and density plots for the two precision values show that even when the temporal parameter is well estimated these values are not, this is partly due to using the scaled edge weight model which is unable to correctly describe the evolution along the edges of the system. These figures also show that the posterior distribution for the temporal dependence parameter is very similar to the prior distribution used, thus the temporal dependency parameter is weakly identified by the data.

6.3 The spatial dependence parameter

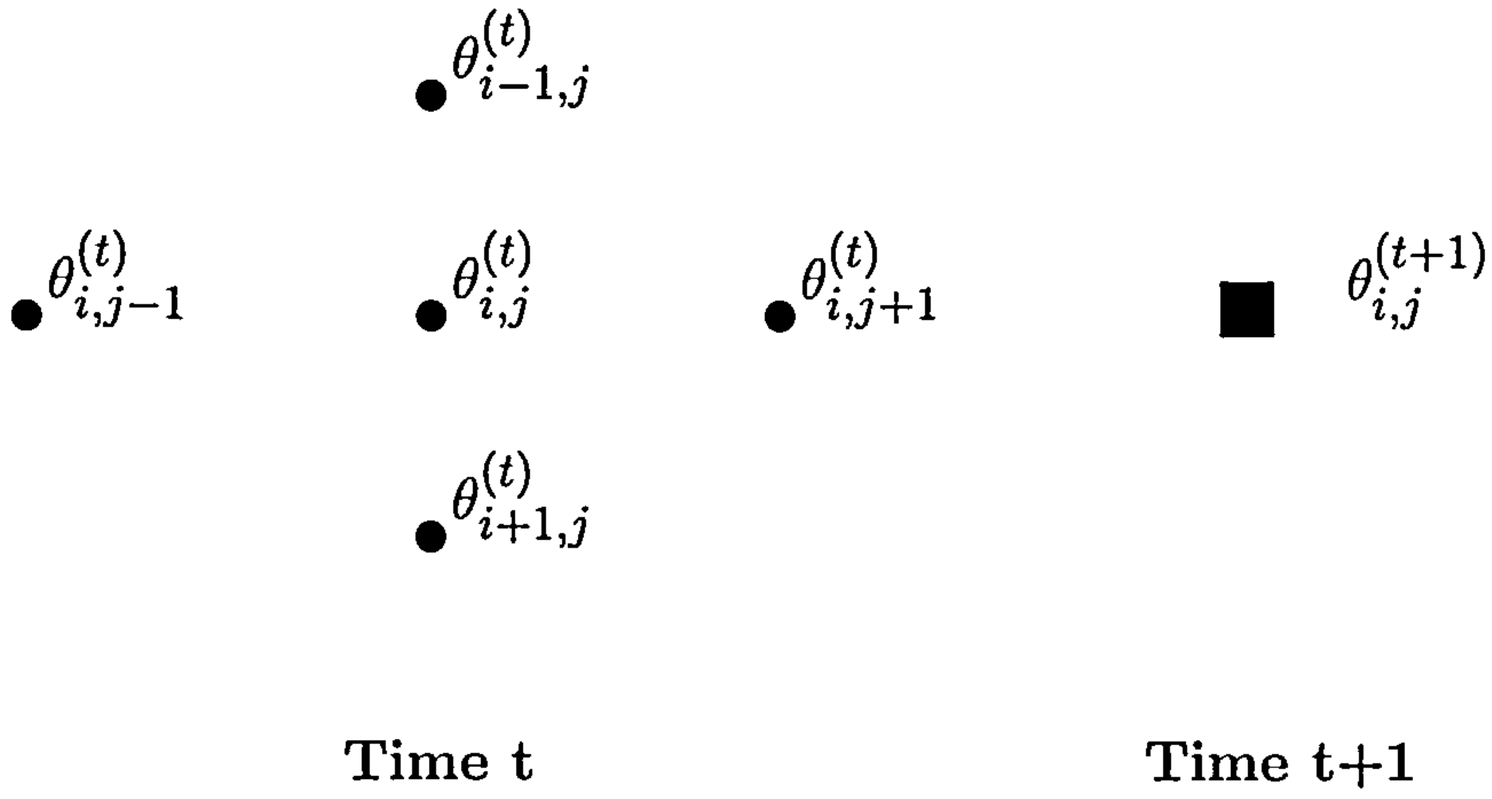


Figure 6.6: *The five parent system for the 2+1D model.*

The spatial dependence parameter β has not appeared explicitly in the model however it is contained within the evolution matrix G. In order to introduce the spatial dependence parameter to the model consider a system with five parents (see Figure 6.6). Define the weight matrix g for a given node in this five parent model to be

$$g = \alpha \begin{pmatrix} 0 & \frac{\beta}{5} & 0 \\ \frac{\beta}{5} & 1-\frac{4\beta}{5} & \frac{\beta}{5} \\ 0 & \frac{\beta}{5} & 0 \end{pmatrix}.$$

Where $\beta \in [0,1)$ this representation allows the weight matrix to represent a number of situations from information only being passed from node $\theta^{(t-1)}_{i,j}$ to node $\theta^{(t)}_{i,j}$ or to the model where none of the information comes from the node of interest at the previous time period. In the binomial weight model developed in Section 2.6.3 a nine parent model was developed where the second order parents had half the weight that the first order parents had. Extending this spatial dependence model to the nine parent model gives the weight matrix for a particular node as

$$g = \alpha \begin{pmatrix} \frac{\beta}{14} & \frac{\beta}{7} & \frac{\beta}{14} \\ \frac{\beta}{7} & 1-\frac{6\beta}{7} & \frac{\beta}{7} \\ \frac{\beta}{14} & \frac{\beta}{7} & \frac{\beta}{14} \end{pmatrix}.$$

if we make the simplifying assumption that the corners have half the weight of the edges, as for the canonical representation. The spatial dependency parameter is still defined as $\beta \in [0, 1)$ and the nine parent model is capable of representing the nine parent isotropic binomial weight model as defined in Chapter 2.

6.3.1 Example

Consider a 2+1D model with scaled edge weights, known mean and temporal dependence but unknown precision and spatial dependence. The Metropolis-Hastings method is used to simulate the unknown variables, as in Chapter 4 the prior distribution for the precision parameter is a *Gamma* distribution. The prior distribution for the spatial dependence is chosen to be a *Beta* distribution since $\beta \in [0, 1)$. In order to use the generated data from the previous example the spatial dependence parameter for the binomial weight model needs to be calculated,

$$g = \alpha \begin{pmatrix} \frac{\beta}{14} & \frac{\beta}{7} & \frac{\beta}{14} \\ \frac{\beta}{7} & 1 - \frac{6\beta}{7} & \frac{\beta}{7} \\ \frac{\beta}{14} & \frac{\beta}{7} & \frac{\beta}{14} \end{pmatrix} = \alpha \begin{pmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{pmatrix}$$

when $\beta = 0.875$.

The Metropolis-Hastings method is used to simulate the unknown variables, as in Chapter 4 the prior distribution for the precision parameter is a *Gamma* distribution. The prior distribution for the spatial dependence is chosen to be a *Beta* distribution since $\beta \in [0, 1)$,

$$\begin{aligned} \log(\tau_s) &\sim \Gamma(0.01, 0.01), \\ \log(\tau_o) &\sim \Gamma(0.01, 0.01), \\ \beta &\sim Beta(a, b). \end{aligned}$$

The model examined is on a 5×5 spatial grid over ten time periods, five different prior distributions for β are considered, $\beta \sim Beta(1, 1)$, $\beta \sim Beta(2, 3)$, $\beta \sim Beta(3, 2)$,

$\beta \sim Beta\,(1,5)$ and $\beta \sim Beta\,(5,1)$. The model uses generated data with parameters $\tau_s = 0.2$, $\tau_o = 0.1$ and $\beta = 0.875$ which are to be simulated and $\mu = 0$ and $\alpha = 0.7$ are considered known.

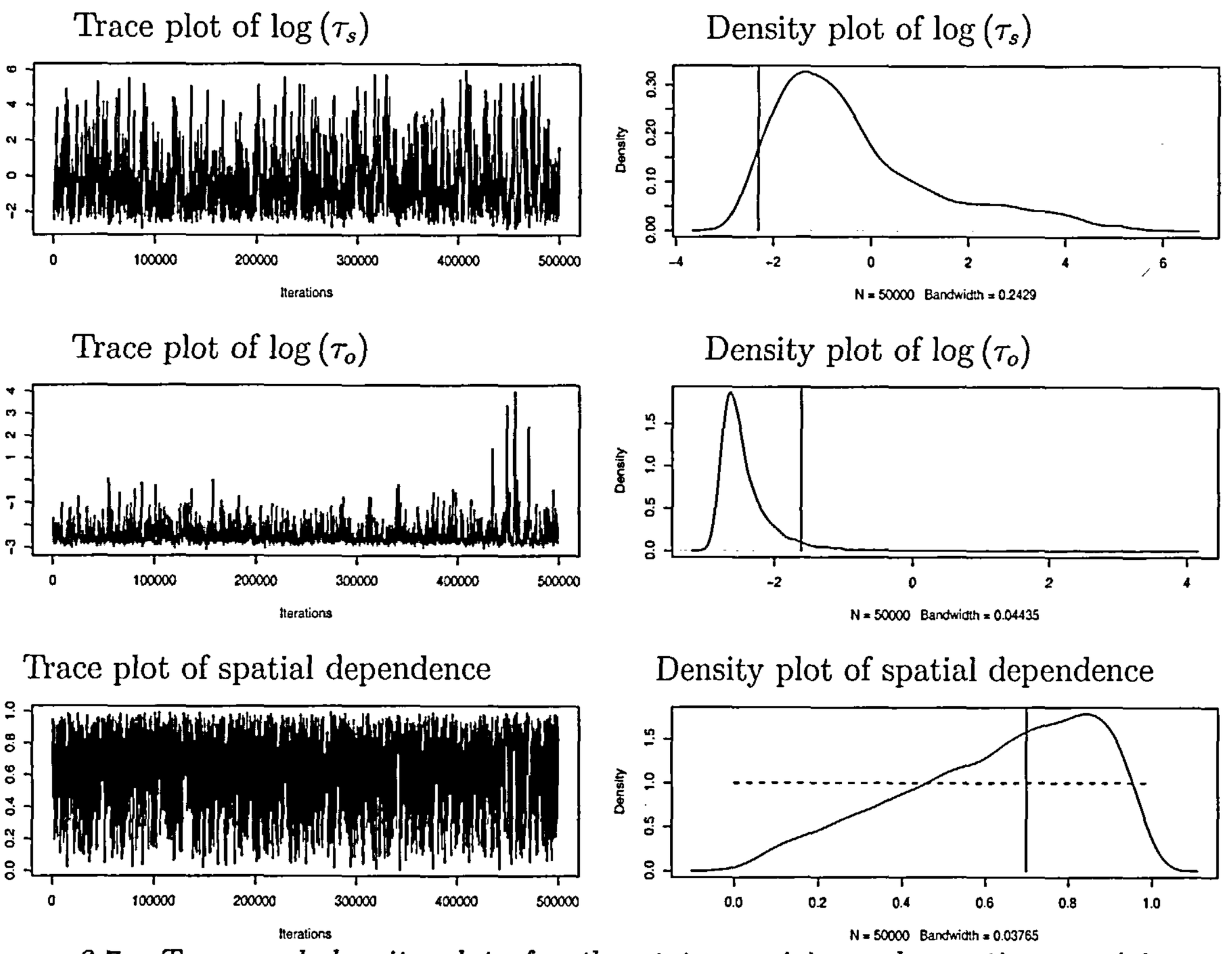


Figure 6.7: *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(1,1)$ *and is shown by the dashed line on the density plot.*

Figures 6.7 to 6.11 show the results for the five different prior distributions applied to the spatial dependence parameter. The vertical line on each of the density plots represents the true parameter value from the simulated data and the dashed line on the density plot for the spatial dependence parameter represents the prior distribution. These figures show that there are some mixing problems in these models. The trace and density plots for the two precision values show that even when the temporal parameter is well estimated

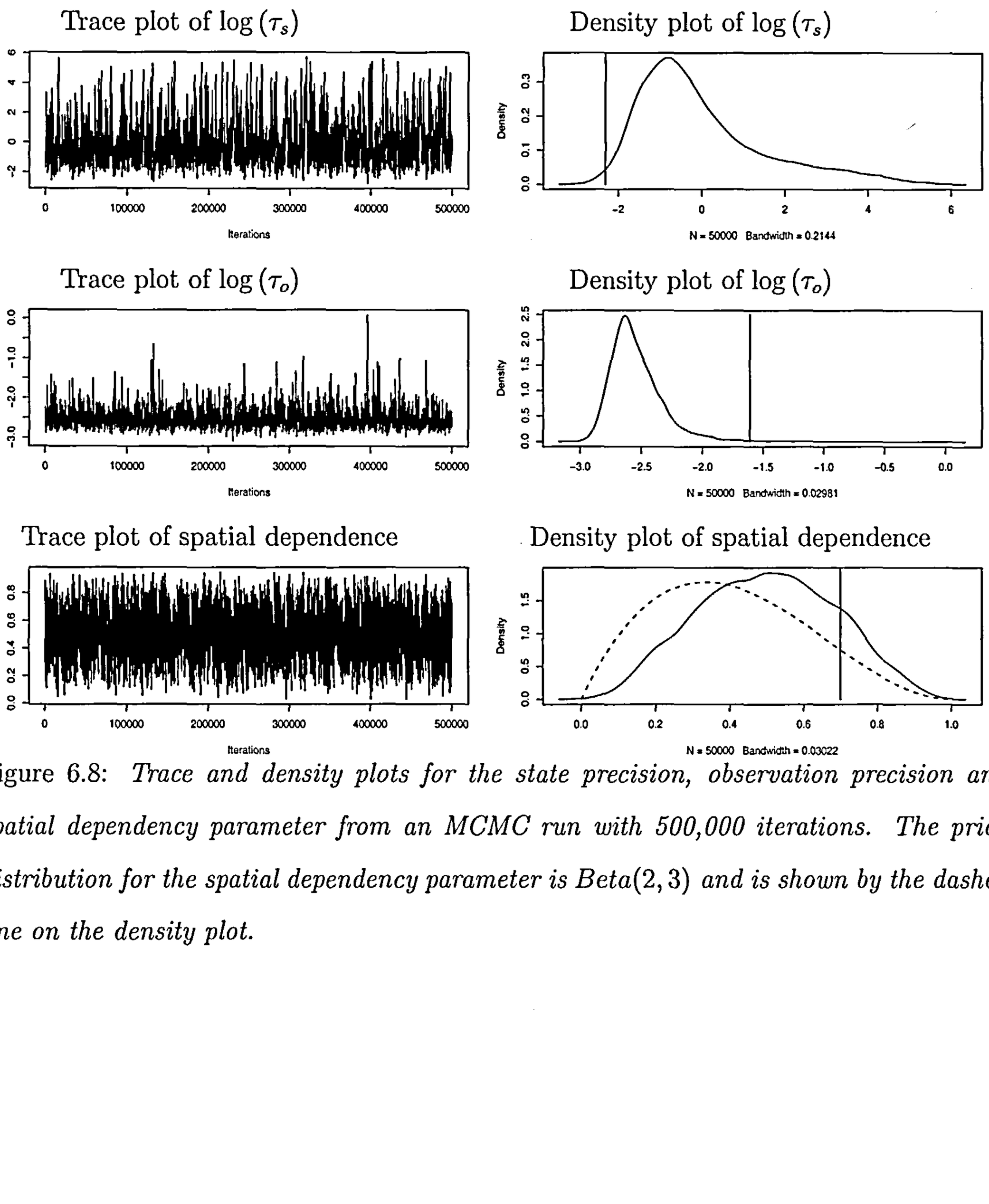


Figure 6.8: *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(2, 3)$ *and is shown by the dashed line on the density plot.*

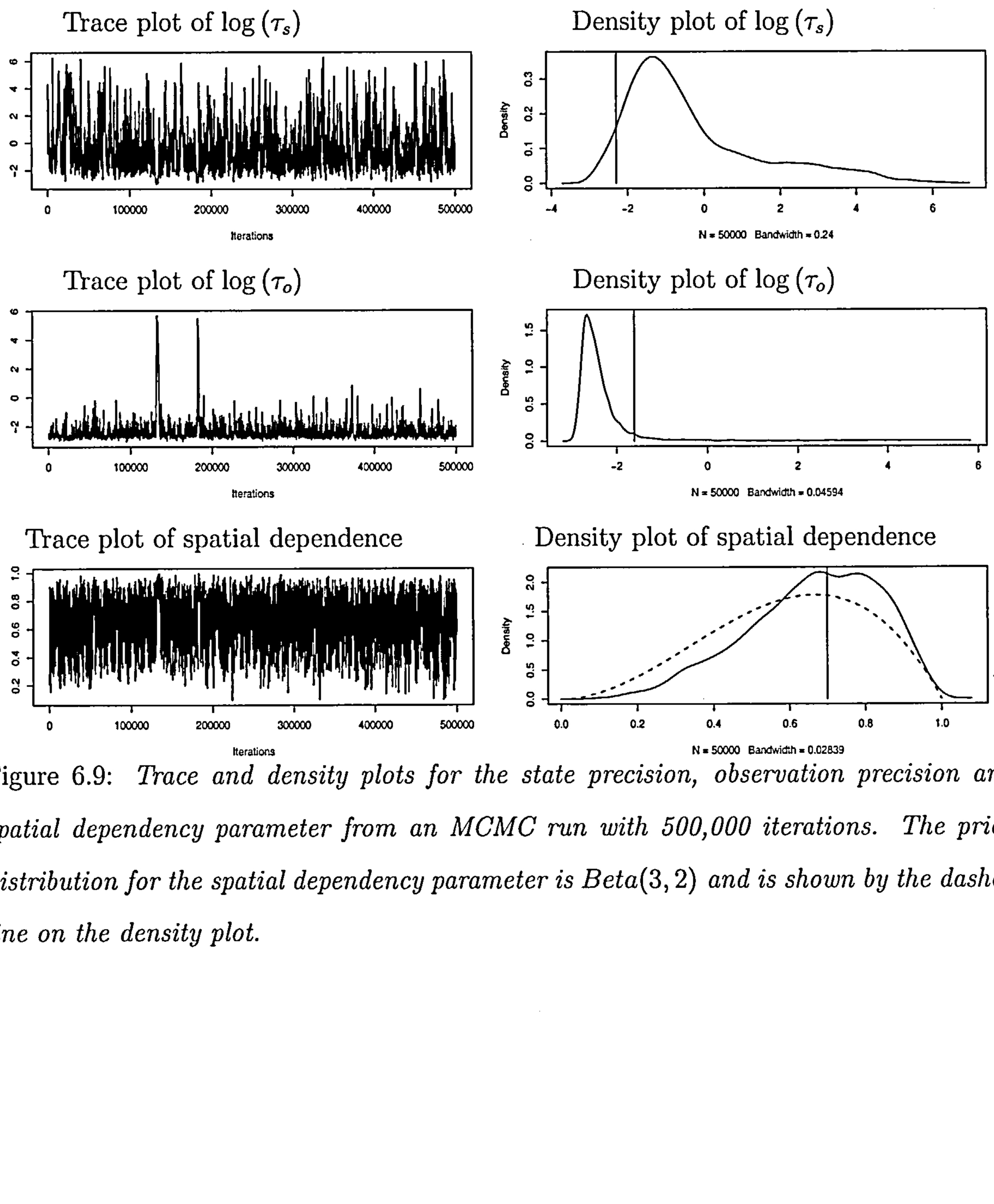


Figure 6.9: *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(3, 2)$ *and is shown by the dashed line on the density plot.*

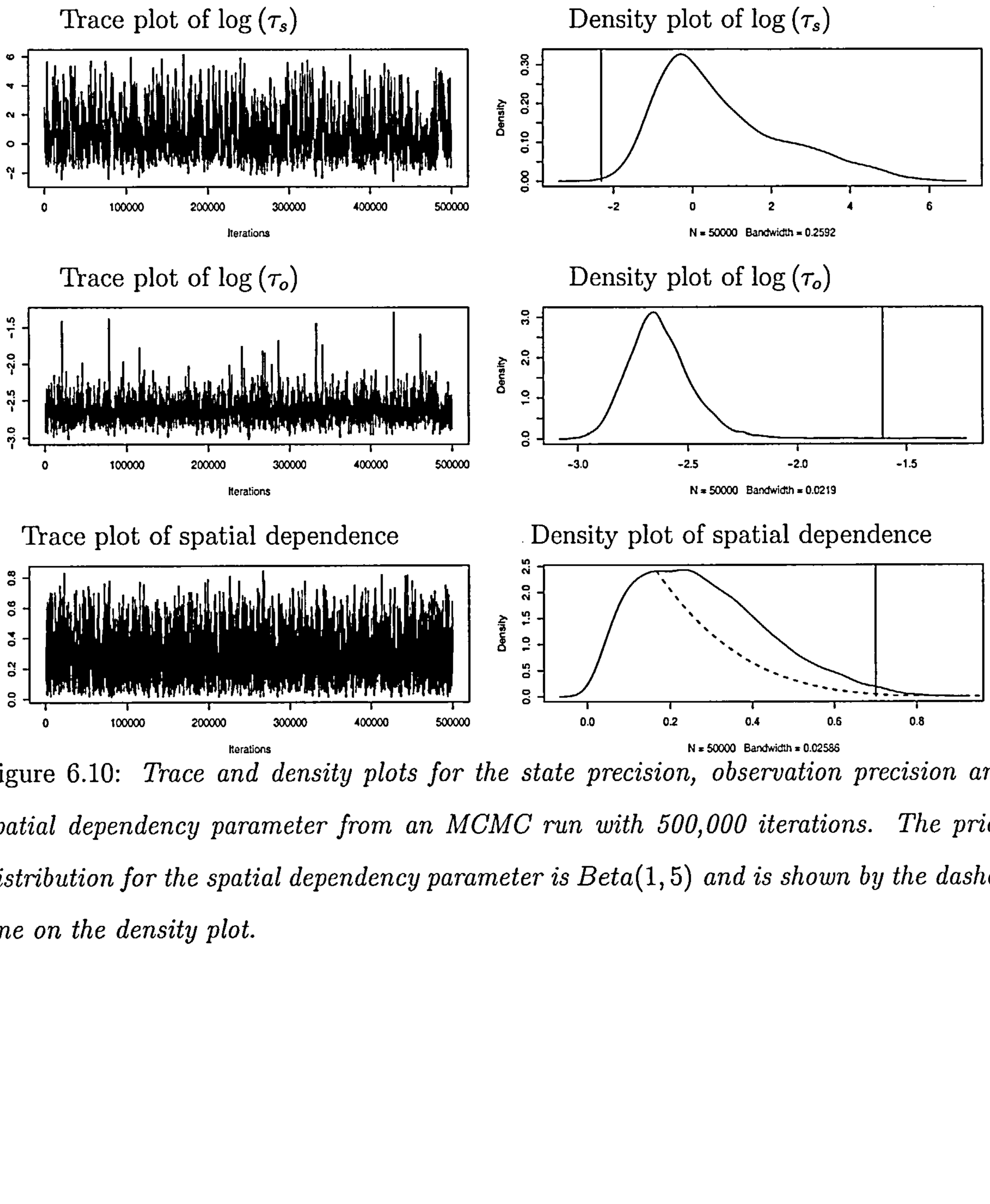


Figure 6.10: *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(1,5)$ *and is shown by the dashed line on the density plot.*

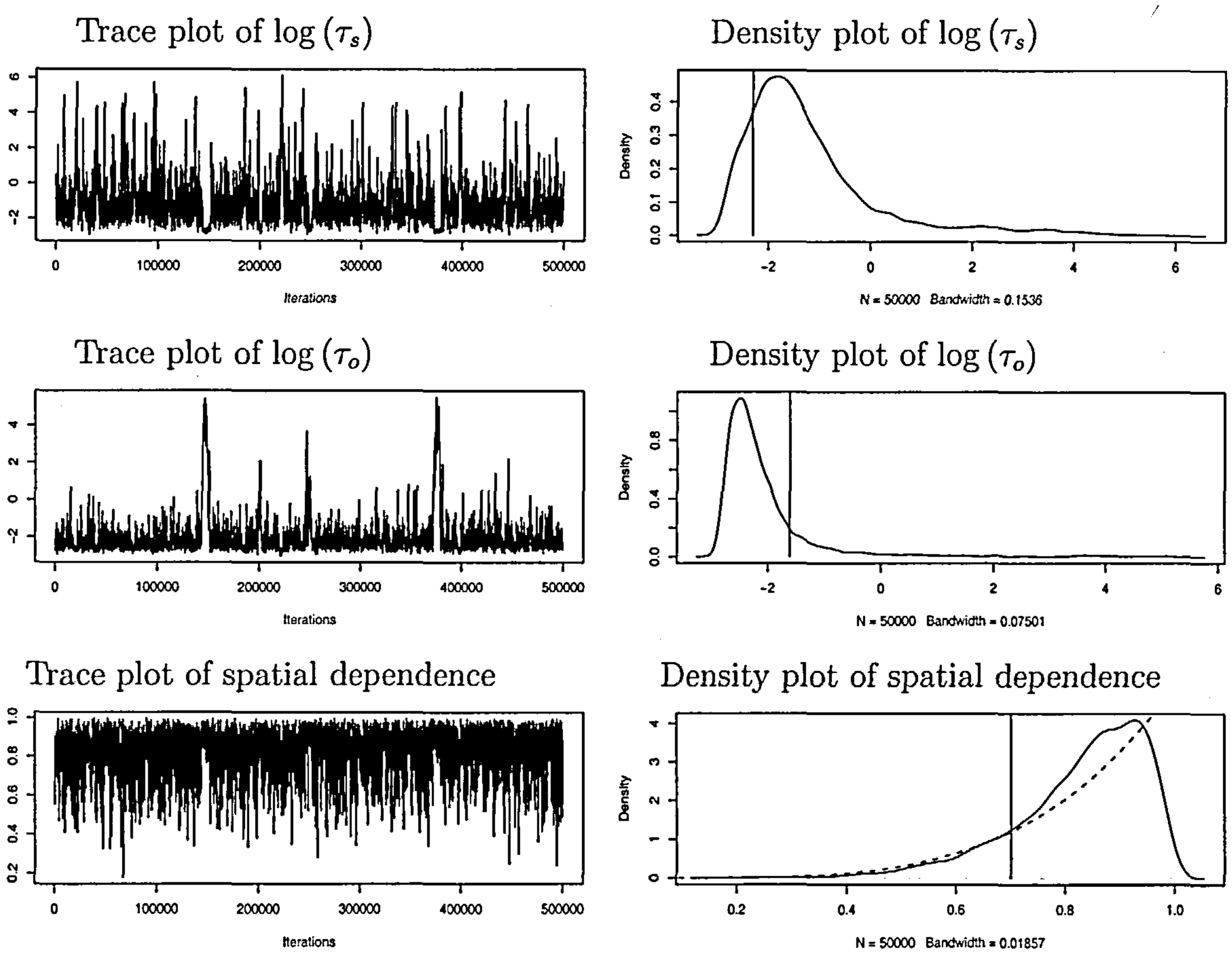


Figure 6.11: *Trace and density plots for the state precision, observation precision and spatial dependency parameter from an MCMC run with 500,000 iterations. The prior distribution for the spatial dependency parameter is* $Beta(5, 1)$ *and is shown by the dashed line on the density plot.*

these values are not, this is partly due to using the scaled edge weight model. These figures also show that the posterior distribution for the spatial dependence parameter is highly correlated with the prior distribution used thus the spatial dependency parameter is weakly identifiable from the data.

6.4 Examining the spatial and temporal dependencies together

In this section two examples are examined as extensions to the previous sections, the first example is on a small generated data set described on a 5×5 spatial grid over ten time periods with four of the parameters in the model treated as unknown (τ_s, τ_o, α and β) and the mean parameter μ treated as known. The second example is on a large generated data set described on a 10×10 spatial grid over twenty time periods and has four parameters in the model treated as unknown (τ_s, τ_o, α and β) and the mean parameter μ treated as known.

In both models the prior distributions are given by

$$\begin{aligned} \tau_s &\sim \Gamma(0.01, 0.01), \\ \tau_o &\sim \Gamma(0.01, 0.01), \\ \alpha &\sim Beta(1,1), \\ \beta &\sim Beta(1,1) \end{aligned}$$

i.e both the spatial and temporal dependency parameters have a weak prior associated with them.

6.4.1 Example 1 : Small data set

Figure 6.12 shows the trace and density plots for the small model. There are some mixing problems present and it can be seen that the marginal posterior distribution for the temporal dependency parameter is strongly influenced by the prior distribution. The parameter estimation for the spatial parameter is not as good as in the model with fixed temporal dependence however given these mixing problems the chain for the spatial dependence has performed reasonably well.

6.4.2 Example 2 : Large data set

Figure 6.13 shows the trace and density plots for the large model. There are some mixing problems present and it can be seen that the marginal posterior distribution for the temporal dependency parameter is strongly influenced by the prior distribution. The parameter estimation for the spatial parameter is not as good as in the model with fixed temporal dependence.

6.5 Discussion

The choice of prior distribution strongly influences the marginal posterior distributions of α and β. Both α and β are only weakly identified by the data. When both the spatial and temporal dependency parameters are simulated simultaneously the trace plots for both examples show that there are some mixing issues which require investigation. In the examples discussed in this chapter it appears as though the failure to identify the spatial or temporal parameters could be due to bad mixing involving the precision parameters. Further investigations which treated the precision parameters as known still highlighted these identifiability problems with spatial and temporal dependence parameters. This suggests that the approach adopted elsewhere in this thesis, using just one parameter to represent the spatio-temporal dependence is probably a sensible strategy.

To extend this model to have conditional edge weights would require the reference table for edge adjustments to be extended so that there is a set of adjustments for each possible combination of α and β. This would require the reference table to have 10,000 rows of information. The calculation of such a database requires a significant amount of computational time as well as increasing the computational time of the MCMC simulation `'C'` code at each iteration and as a consequence it is not feasible to consider it in this thesis.

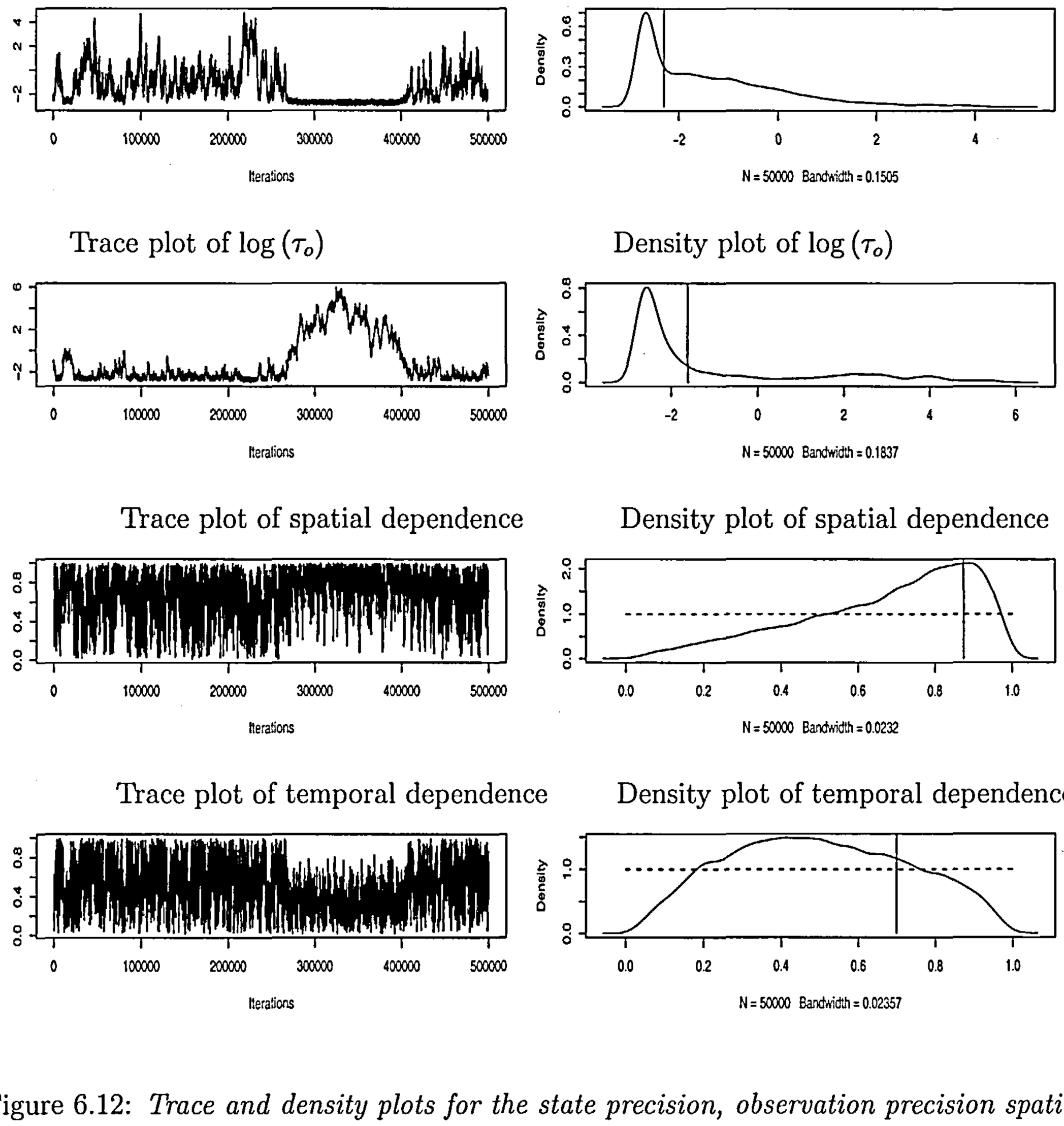
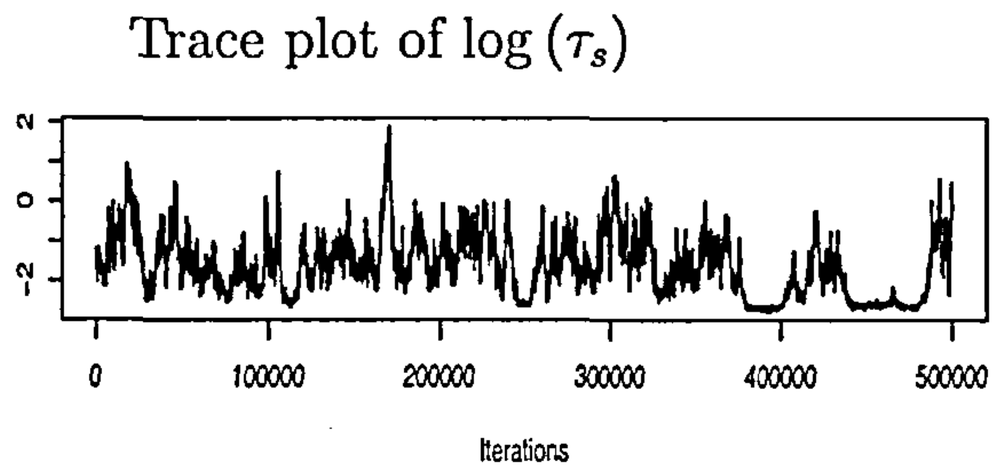
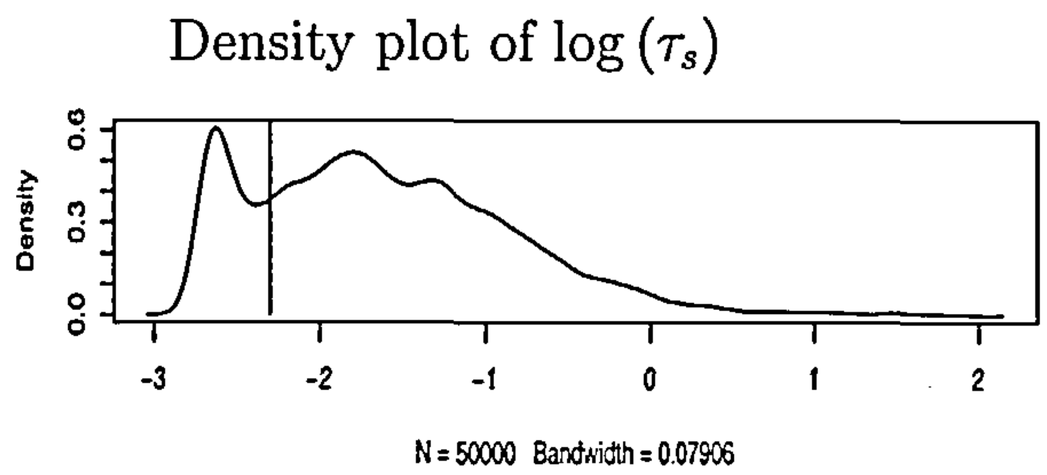
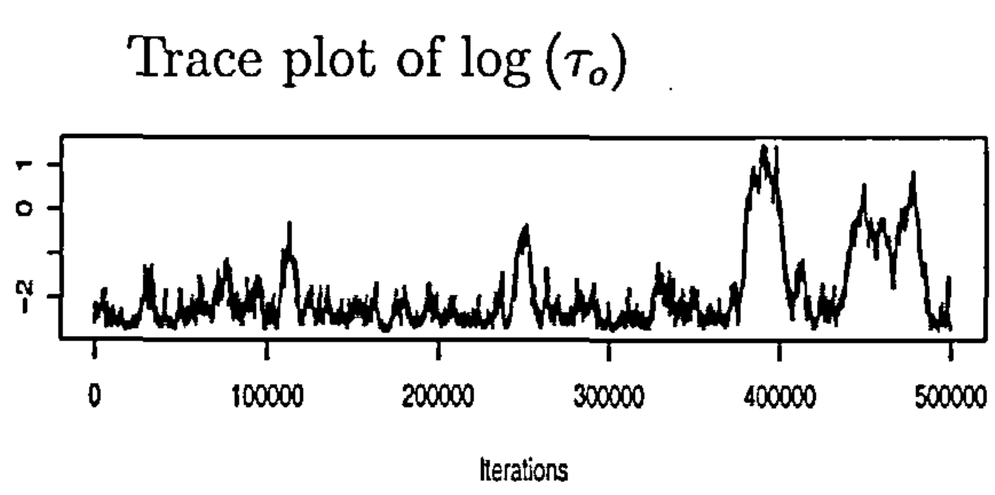
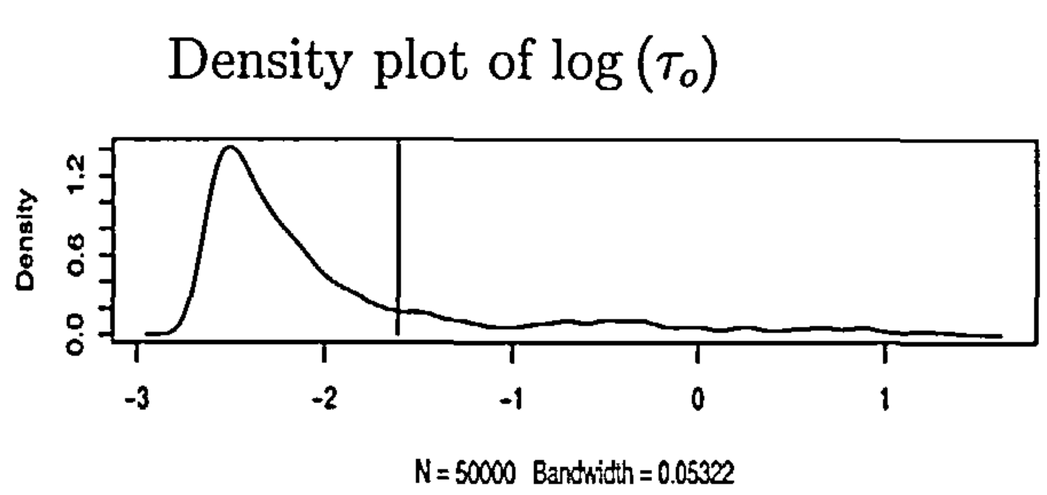


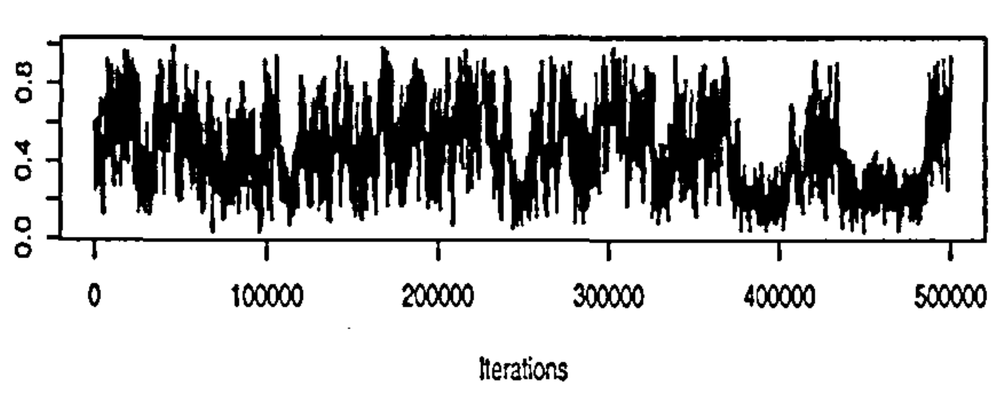
Figure 6.12: *Trace and density plots for the state precision, observation precision spatial and temporal dependency parameter from an MCMC run with 500,000 iterations on a* 5×5 *spatial grid over ten time periods. The prior distribution for the both the spatial and temporal dependency parameter is* $Beta(1, 1)$ *and is shown by the dashed line on the density plots.*

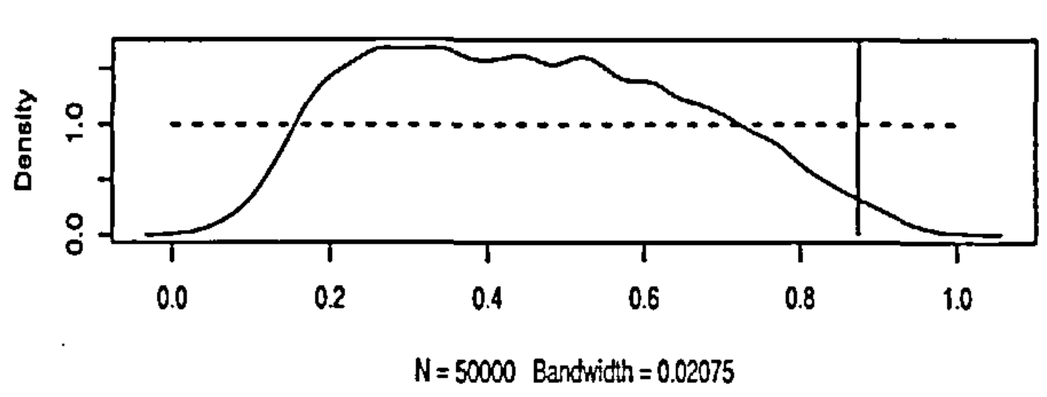


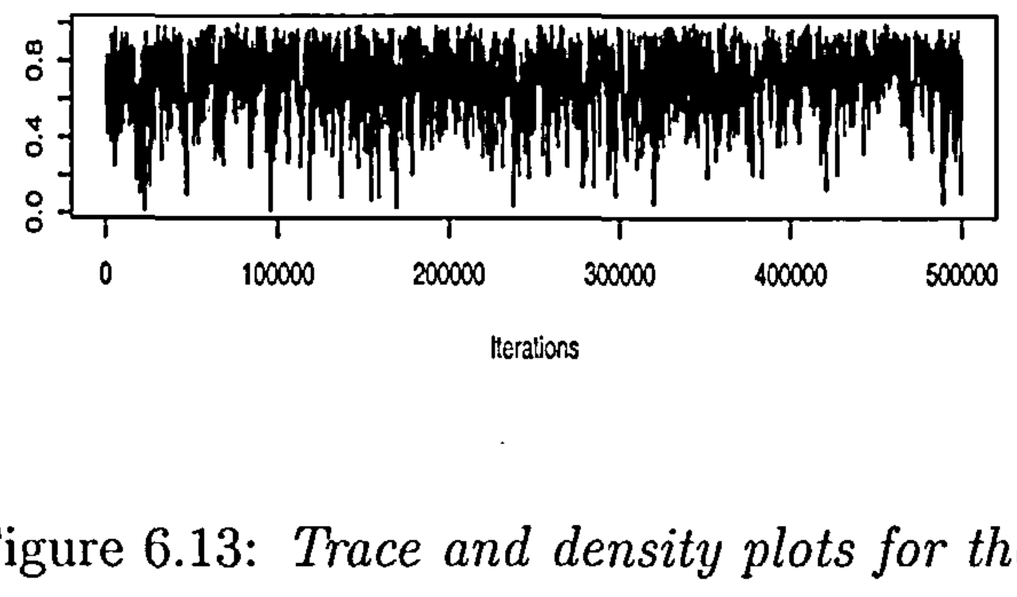












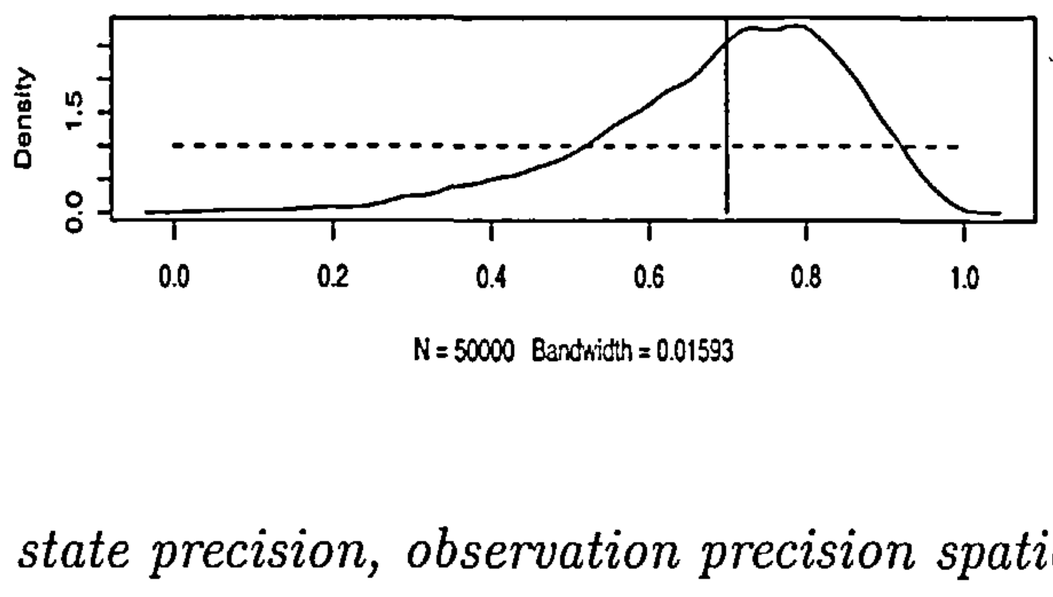


Figure 6.13: *Trace and density plots for the state precision, observation precision spatial and temporal dependency parameter from an MCMC run with 500,000 iterations on a* 10×10 *spatial grid over twenty time periods. The prior distribution for the both the spatial and temporal dependency parameter is* $Beta(1, 1)$ *and is shown by the dashed line on the density plots.*

Chapter 7

Case Studies

7.1 Introduction

The models discussed in Chapters 3 and 5 have been tested in the context of simulated data. In this chapter these models are applied to two real data sets. These examples use ocean temperature data and apply both the fine and coarse models. The data used in this chapter has appeared in the literature. See Higdon (1998), Stroud et al. (2001) and Lavine & Loizier (1999).

The first data set is a thin strip of the North Atlantic taken from 20 to 30 degrees latitude and −80 to −20 degrees longitude and the second set is a "square" region from 20 to 50 degrees north of the Equator and running from −30 to 0 degrees longitude. The data is obtained from the National Oceanographic data centre (NODC) `http://www.nodc.noaa.gov/`. This centre collects data from around the world which is available to download from the World Wide Web. Ocean station data (NANSEN) was considered for the two data sets from the variables available. Location (in degrees, minutes and seconds), time of the observation (day, month and year), depth of the observation and temperature of the observation were the variables of interest. These variables were extracted from the NODC data file using a simple Python script (van Rossum & Drake). Python is available from `http://www.python.org/`. The data was then converted into a

usable form in R. Here the observations were indexed according to their time and location for depths of approximately 100m. The resulting data file consists of the location index and the observation value ready to be read into an appropriate 'C' program for MCMC analysis. The data is continuous in both space and time and must be filtered so that it can be mapped onto the underlying latent grid. The data is filtered so that at most one observation exists for each latent grid point with resolution 1 degree by 1 degree. If more than one observation exists at a grid point then its spatial location defined by degrees minutes and seconds is used to identify the observation which is nearest to the grid point. If multiple observations exist at this spatial location only one of them is retained. This method for thinning the data is also used to map the data onto smaller grids to reduce the effect of having multiple observations at a grid location in the smaller models.

Initially a fine grid analysis on a small sample of the data is used to determine suitable mixing parameters for the Metropolis Hastings algorithm. Then the coarse model is applied to the full set of data which corresponds a fine grid of resolution 1 degree by 1 degree. The parameter estimates from the coarse model are then used to produce surface plots of the temperature, by carrying out interpolation using the fine model

7.2 Case Study 1: Small Data Set

The data in this example is taken from 20 to 30 degrees latitude and forms a rectangular region running from -80 to -20 degrees longitude. The data is recorded over 10 years from 1979 to 1988. In this model data is used from the entire year as temperatures at this latitude are approximately constant throughout the year. The location of the data is shown in Figure 7.1 where shade indicates time. The observations are indexed according to their time and spatial location for depths of approximately 100m.

It is assumed that the data can be modelled by the DLM as defined in Chapter 2 with isotropic binomial edge weights. Interest is in simulating the posterior distributions for the state precision τ_s, observation precision τ_o and mean temperature μ. The temporal

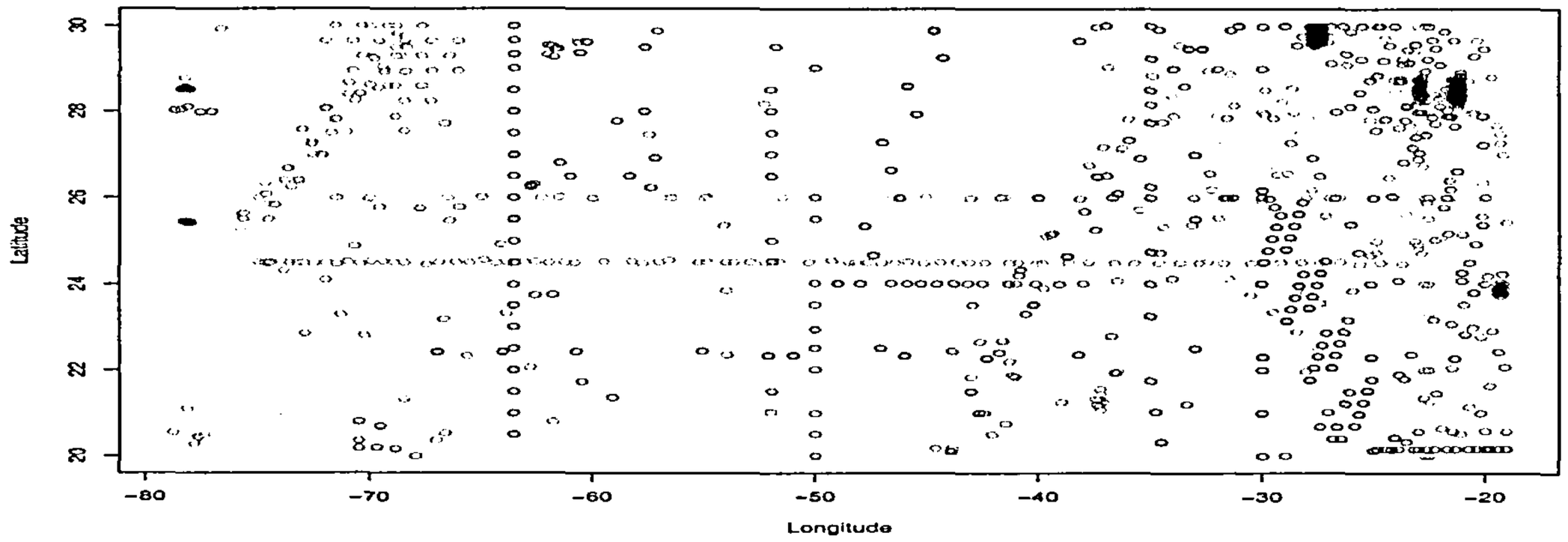


Figure 7.1: *Spatial location of the data with shade representing time.*

dependence parameter is considered fixed as the data contains little information regarding the temporal dependence. For this example there is no expert information for the value of α thus in order to demonstrate this procedure the temporal dependence is fixed at $\alpha = 0.9$. Due to the large amount of data available In both examples in this chapter the prior distribution for the precision parameters has been changed to a more diffuse distribution. This prior distribution is more uninformative than the prior distributions used in the previous chapters. Vague prior distributions are considered for the two precision parameters and the mean parameter,

$$\begin{aligned} \tau_s &\sim Gamma\,(0.001, 0.001)\,, \\ \tau_o &\sim Gamma\,(0.001, 0.001)\,, \\ \mu &\sim N\,(0, \infty)\,. \end{aligned} \tag{7.1}$$

Recall the models examined in Chapter 4 all used the same random walk update irrespective of the underlying latent grid size i.e. the model using a small grid is used to determine the random walk updates for the corresponding models on a larger latent grid. Thus in this chapter the initial analysis is done on a small spatial grid of dimensions 3×8 evolving over the 10 years (one time period per year) to reduce the time needed to determine the random walk updates. This model reduces the spatial grid to one quarter

of its original size thus reducing the computational time required to determine a suitable value for the random walk update. Once the mixing parameters (random walk updates) have been determined a much finer grid can be used to describe the model and determine the parameter estimates. Three examples are presented in this study, the initial example is a test model run on a small (3×8) spatial grid using the fine model, the second example compares a fine model on a 6×15 spatial grid with the corresponding coarse model on a 3×8 spatial grid and the final example uses the coarse approach on a 6×30 spatial grid which is equivalent to the fine approach on a 1 degree by 1 degree grid for this data set the parameter estimates from this coarse model are then used to build surface plots of the temperature for the corresponding fine model.

7.2.1 Results

Initial model

The first model uses a small grid and is used to determine possible mixing parameters for the Metropolis Hastings algorithm. The data is filtered to a 6×15 grid which results in 252 observations being mapped onto the 3×8 grid. Filtering the data to this 3×8 spatial grid leaves only 82 observation for the analysis and the resulting MCMC simulation does not converge thus a larger data set is preferable. Using data from the 6×15 spatial grid gives 252 observations in the model. By using this larger data set it is possible for a grid point to have four observations associated with it at any given time and may cause the resulting MCMC chain to be heavily influenced by the data. The results for this model are shown in Figure 7.2 and Tables 7.1 and 7.2, these results were obtained using a $N(0, 100^{-1})$ random walk update. The trace plots in Figure 7.2 have been thinned by a factor of 10 for storage reasons, they show reasonable mixing and suggest that this random walk update is suitable for the larger models. Table 7.1 gives the mean and standard deviation for the precision and mean temperature parameters along with the associated time series standard error for these parameters. The mean temperature proposed from the posterior distribution of μ is 20.74 and the actual mean of the data is 20.75 which suggests that

the posterior distribution for μ is correct.

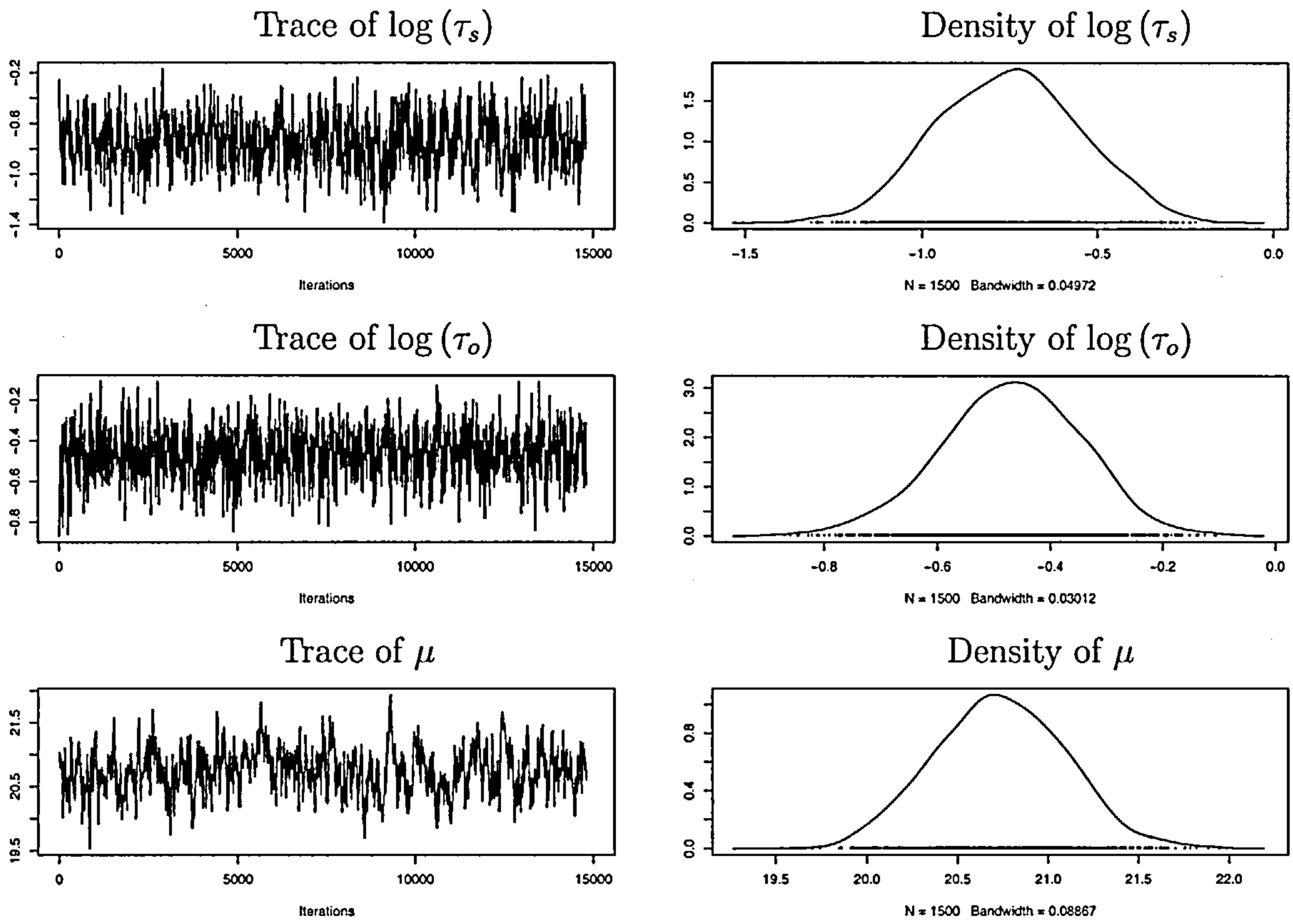


Figure 7.2: *Trace and density plots for ocean temperature data modelled using the fine model on a* 3×8 *spatial grid.*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-0.7505	0.2025	0.005228	0.008316
$\log(\tau_o)$	-0.4664	0.1246	0.003218	0.004116
μ	20.7435	0.3611	0.009324	0.016868

Table 7.1: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 3×8 *spatial grid.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.1378	-0.8944	-0.7480	-0.6102	-0.3715
$\log(\tau_o)$	-0.7212	-0.5452	-0.4637	-0.3809	-0.2368
μ	20.0464	20.4917	20.7372	20.9947	21.4417

Table 7.2: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 3×8 *spatial grid.*

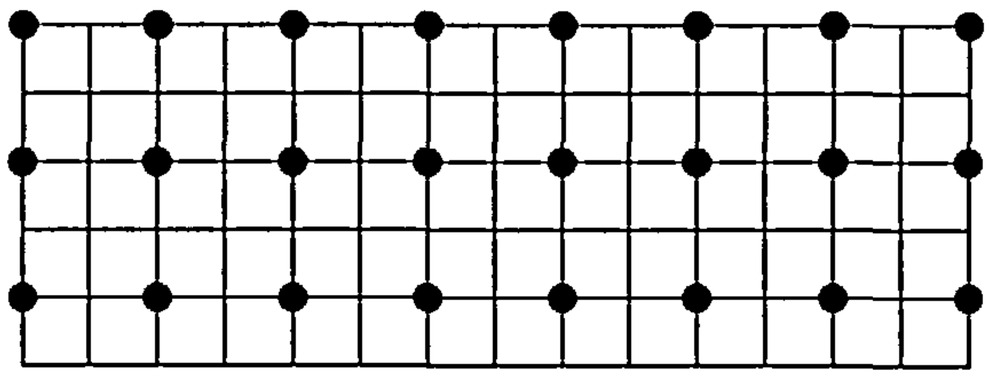


Figure 7.3: *The coarse and fine grid points for the second model.*

Second model

The second model compares a fine and coarse approach for the simulation of the posterior distributions. The fine model is developed on a 6×15 spatial grid and the corresponding coarse model is defined on a 3×8 spatial grid. Figure 7.3 shows the grid lines for the fine model with the corresponding nodes for the coarse model represented by the bullet points. As in the above model the data is filtered to a 6×15 grid for the fine model which results in 252 observations being used in this model, the data for the the corresponding coarse model is mapped to a 3×8 spatial grid which results in 82 observations being used. Both models use a $N(0, 100^{-1})$ random walk update for all three parameters and the prior distributions as listed in Equation (7.1). The results for the fine model are shown in Figure 7.4 and Tables 7.3 and 7.4, the corresponding results for the coarse model are shown in Figure 7.5 and Tables 7.5 and 7.6. Comparing the mean values for the posterior distributions of the three parameters shows a discrepancy between the two models. The fine model suggests $\log(\tau_s) = -0.54$, $\log(\tau_o) - 0.56 =$ and $\mu = 20.76$

whereas the coarse model suggests $\log(\tau_s) = 0.01$, $\log(\tau_o) = -0.82$ and $\mu = 20.71$. This discrepancy is partially due to the difference in the data used, in both models the number of observations used is significantly smaller than the total number of latent nodes.

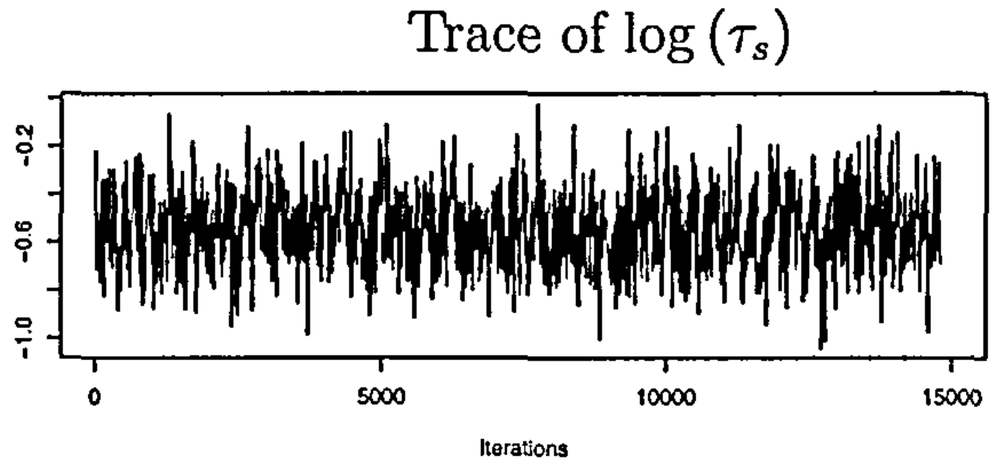


Figure 7.4: *Trace and density plots for ocean temperature data modelled using the fine model on a* 6×15 *spatial grid.*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-0.5406	0.1592	0.0012997	0.002339
$\log(\tau_o)$	-0.5576	0.1201	0.0009805	0.001783
μ	20.7589	0.2394	0.0019545	0.003433

Table 7.3: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 6×15 *spatial grid.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-0.8382	-0.6524	-0.5448	-0.4353	-0.2063
$\log(\tau_o)$	-0.7948	-0.6383	-0.5555	-0.4733	-0.3310
μ	20.2891	20.5981	20.7618	20.9202	21.2268

Table 7.4: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 6×15 *spatial grid.*

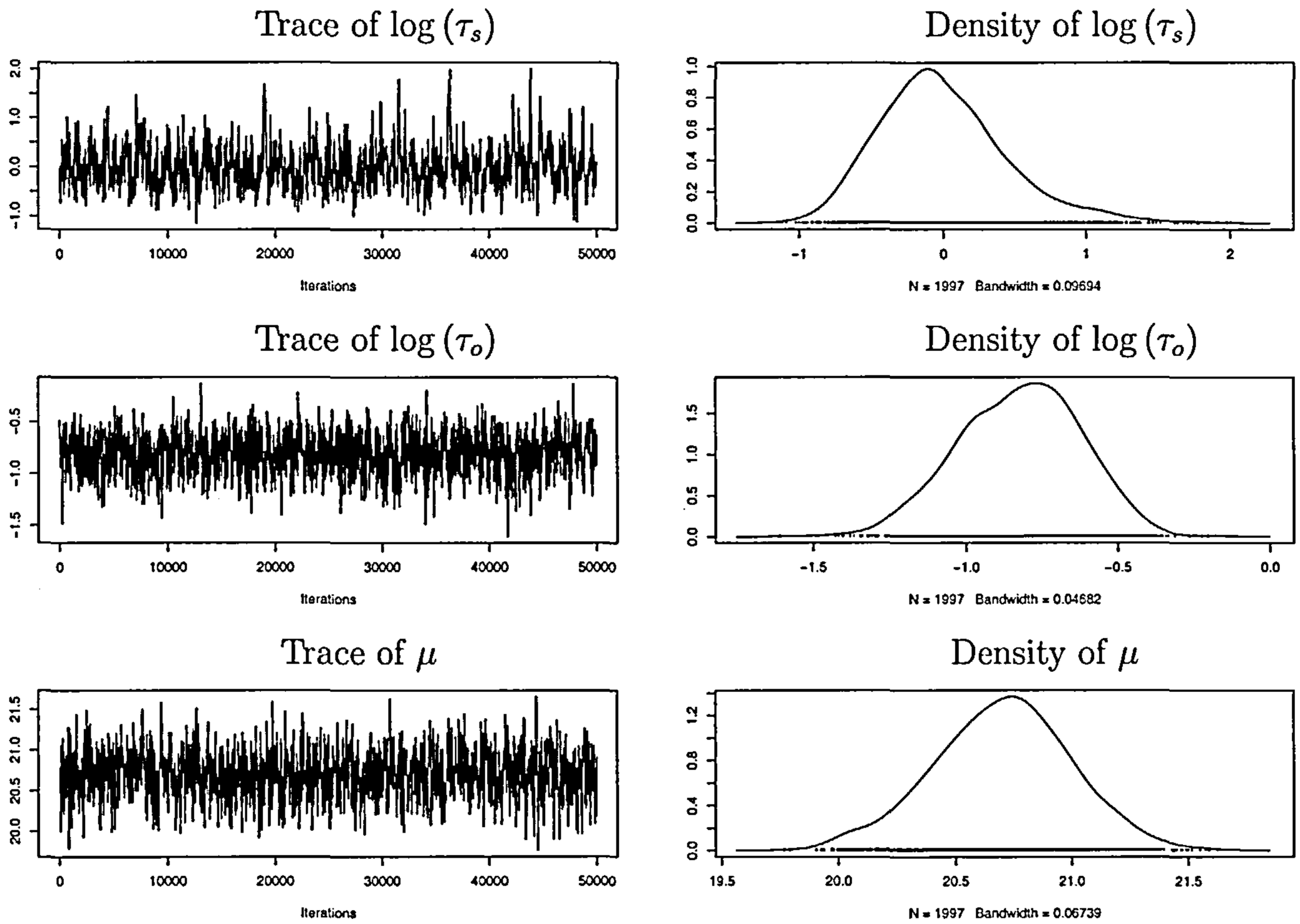


Figure 7.5: *Trace and density plots for ocean temperature data modelled using the coarse model on a* 3×8 *spatial grid.*

Final model

The final model uses a coarse approach applied to a 6×30 grid which corresponds to the fine model on a 12×60 grid which is a resolution of 1 degree $\times$ 1 degree. The observations

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	0.008574	0.4555	0.004559	0.008156
$\log(\tau_o)$	-0.817392	0.2012	0.002014	0.003516
μ	20.708989	0.2926	0.002929	0.005351

Table 7.5: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 3×8 *spatial grid.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-0.7519	-0.3024	-0.03938	0.2664	1.0815
$\log(\tau_o)$	-1.2245	-0.9513	-0.81133	-0.6762	-0.4421
μ	20.1172	20.5161	20.70965	20.9077	21.2793

Table 7.6: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 3×8 *spatial grid.*

are mapped to the coarse grid which results in 358 observations being used. The MCMC simulation is run for 2000 iterations on 8 nodes, the first 250 results are discarded as burn in which leaves 14000 simulations for analysis.

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-0.6065	0.12966	0.003099	0.005598
$\log(\tau_o)$	-0.6175	0.09086	0.002172	0.003939
μ	20.6086	0.12274	0.002934	0.005280

Table 7.7: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 6×30 *spatial grid.*

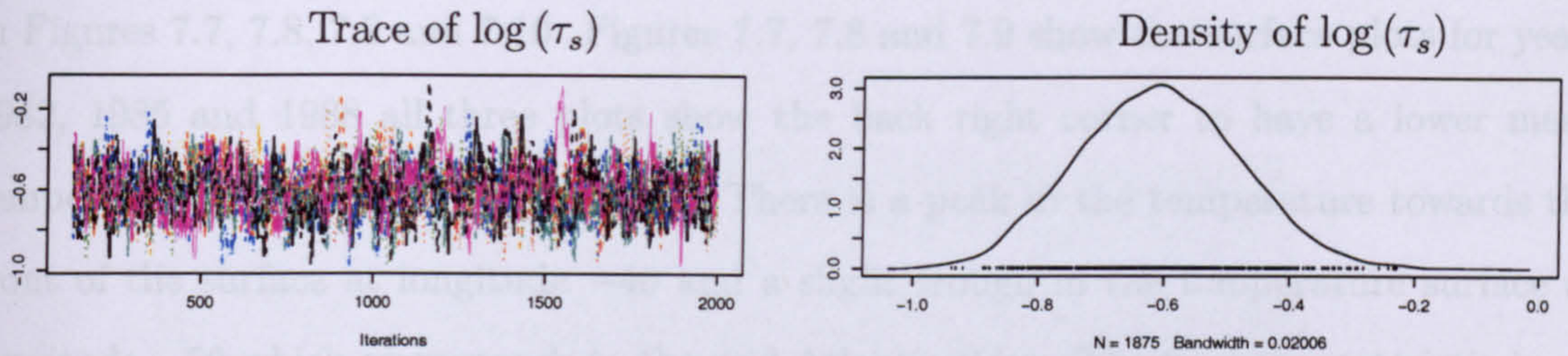


Figure 7.6: *Trace and density plots for ocean temperature data modelled using the coarse model on a* 6×30 *spatial grid.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-0.8487	-0.6961	-0.6088	-0.5216	-0.3496
$\log(\tau_o)$	-0.7973	-0.6785	-0.6159	-0.5556	-0.4423
μ	20.3648	20.5271	20.6093	20.6950	20.8448

Table 7.8: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 6×30 *spatial grid.*

Figure 7.6 shows the resulting MCMC chain for this model, Tables 7.7 and 7.8 show the summary statistics and quantile ranges for these posterior distributions. This model suggests the log precisions are -0.6 and the mean temperature is $\mu = 20.6$. These values are used to form the mean temperature surface plots for the fine model which are shown

in Figures 7.7, 7.8, 7.9 and 7.10. Figures 7.7, 7.8 and 7.9 show the surface plots for years 1982, 1985 and 1988 all three plots show the back right corner to have a lower mean temperature than the surrounding area. There is a peak in the temperature towards the front of the surface at longitude -40 and a slight trough in the temperature surface at longitude -50 which corresponds to the mid Atlantic ridge. The final characteristic in all of these surface plots is a minor ridge occurring at longitude -60. The predictive mean surface in Figure 7.10 is smoother than the other plots due to there being no data available for this time period. The predictive surface does maintain the main characteristics which were listed for the surface plots from 1982, 1985, and 1988.

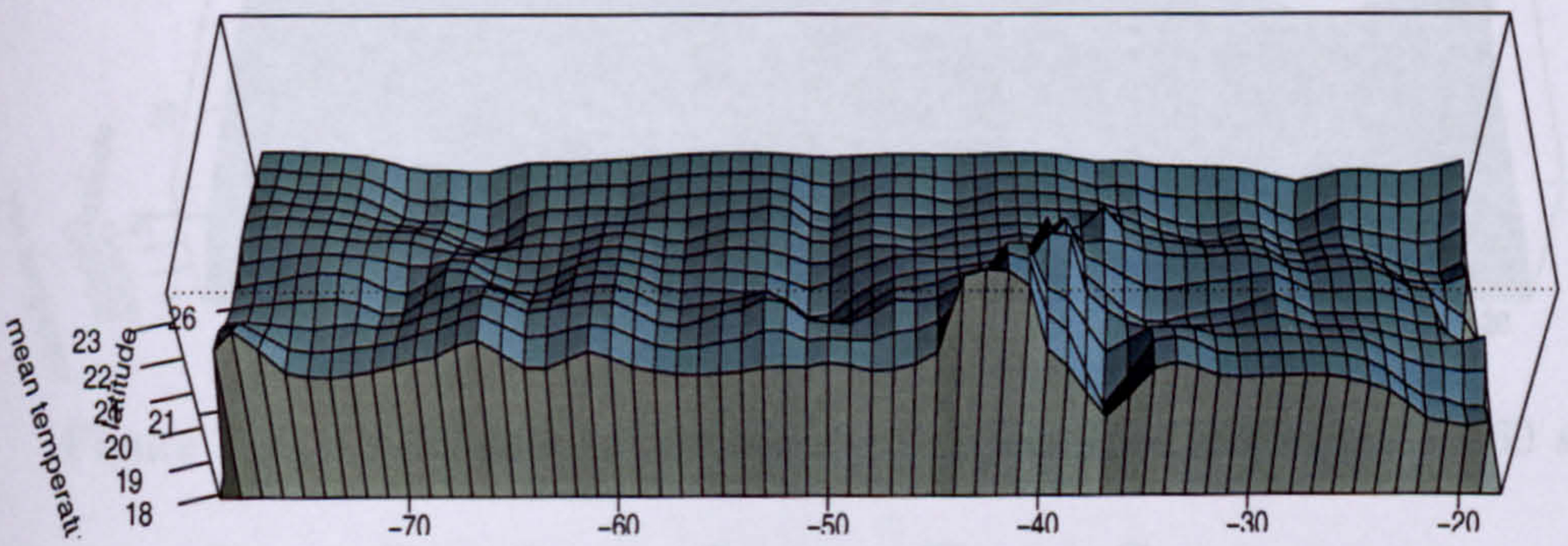


Figure 7.7: *Mean temperature surface for 1982 on a* 12×60 *spatial grid.*

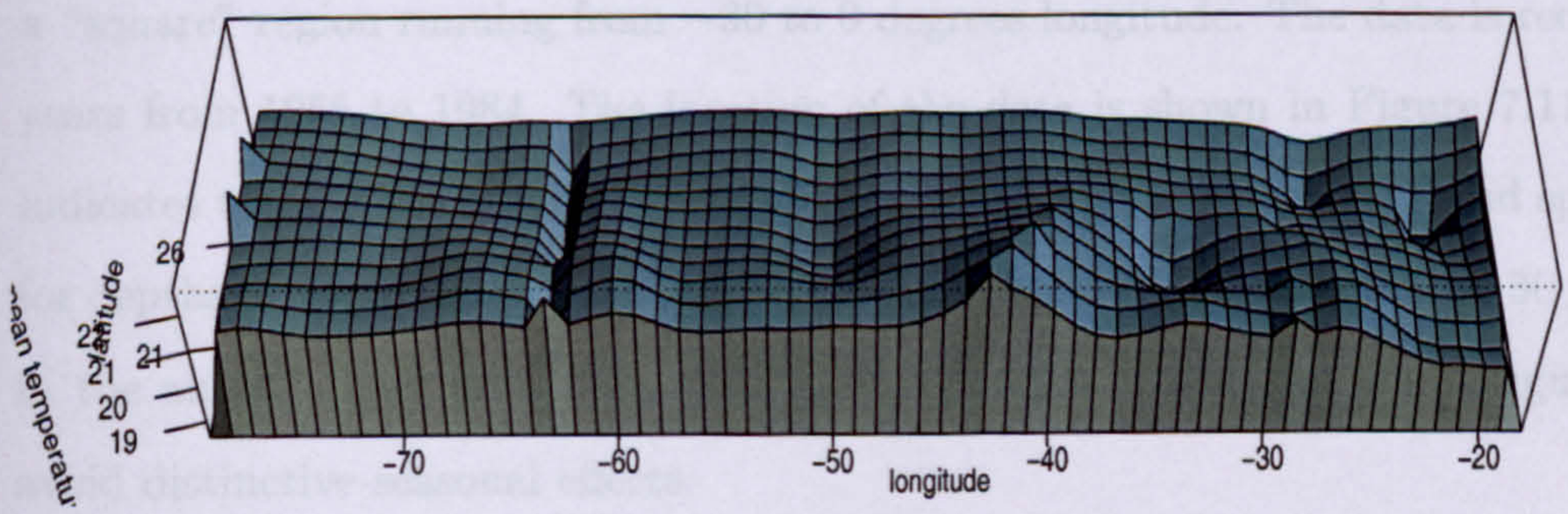


Figure 7.8: *Mean temperature surface for 1985 on a* 12×60 *spatial grid.*

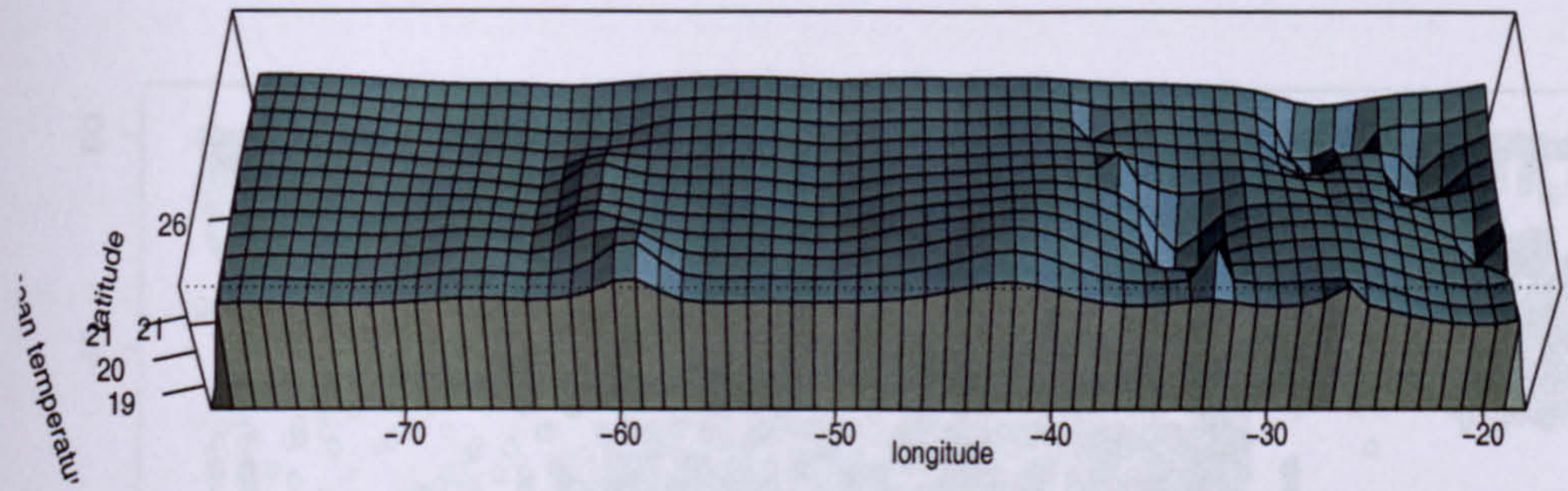


Figure 7.9: *Mean temperature surface for 1988 on a* 12×60 *spatial grid.*

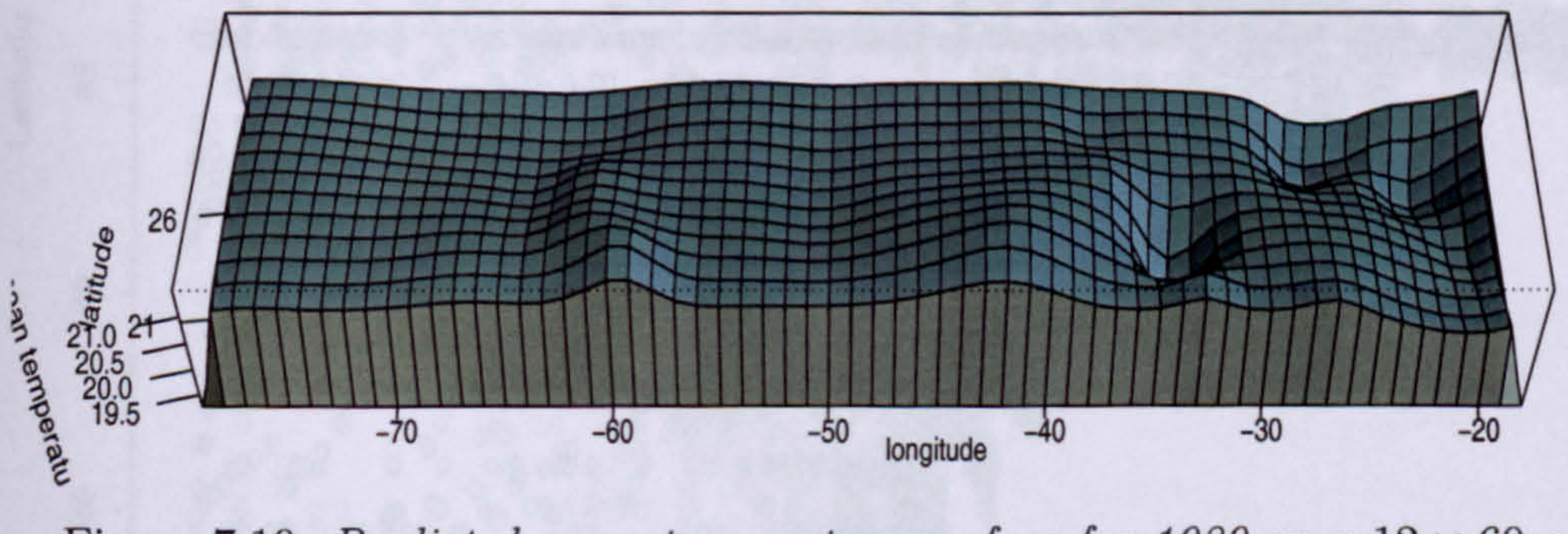


Figure 7.10: *Predicted mean temperature surface for 1989 on a* 12×60 *spatial grid.*

7.3 Case Study 2: Large Data Set

The data in this example is taken from 20 to 50 degrees north of the Equator and forms a "square" region running from -30 to 0 degrees longitude. The data is recorded over 30 years from 1955 to 1984. The location of the data is shown in Figure 7.11 where shade indicates time. The observations are indexed according to their time and spatial location for depths of approximately 100m. Figure 7.11 shows all the data for the 30 years however in the analysis only data from the summer months (June, July and August) is used to avoid distinctive seasonal effects.

It is assumed that the data can be modelled by the DLM as defined in Chapter 2 with isotropic binomial edge weights. Interest is in simulating the posterior distributions for the state precision τ_s, observation precision τ_o and mean temperature μ. The temporal dependence parameter is considered fixed at $\alpha = 0.9$ as the data contains little information

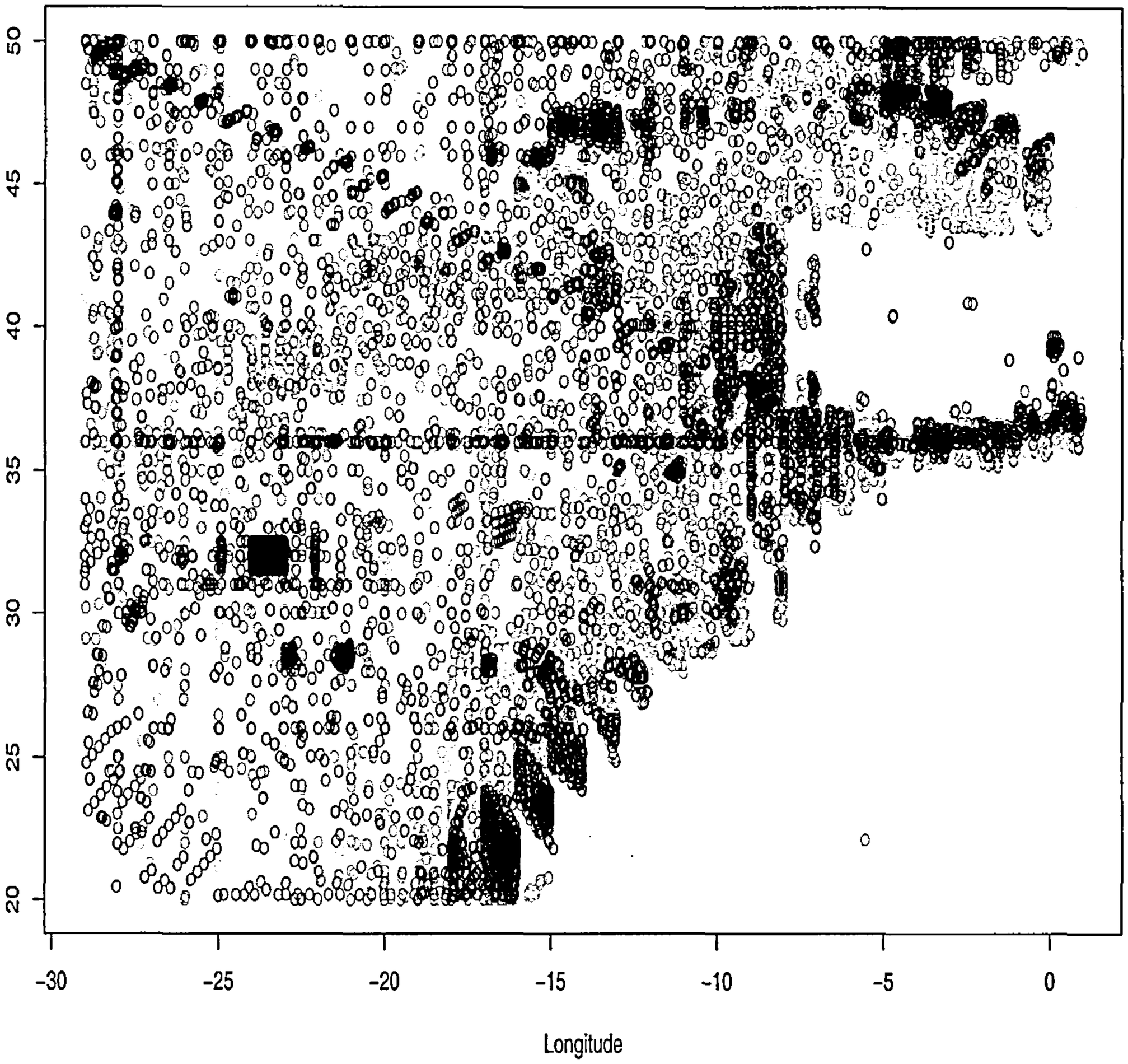


Figure 7.11: *Spatial location of the data with shade representing time.*

regarding the temporal dependence. Vague prior distributions are considered for the two precision parameters and the mean parameter

$$\begin{aligned} \tau_s &\sim Gamma\,(0.001, 0.001)\,, \\ \tau_o &\sim Gamma\,(0.001, 0.001)\,, \\ \mu &\sim N\,(0, \infty)\,. \end{aligned} \tag{7.2}$$

Initial analysis is done on a small spatial grid of dimensions 4×4 evolving over all the time periods. This model reduces the spatial grid to one eighth of its original size to determine suitable mixing parameters for the Metropolis Hastings algorithm. Once these mixing parameters have been determined a much finer grid can be used. The models presented in this study are the initial fine model run on a 4×4 spatial grid, the second model is for a 8×8 spatial grid with results presented for both the fine and corresponding coarse model and then concludes with the coarse model on a 16×16 spatial grid which is equivalent to the fine model with resolution 1 degree by 1 degree.

7.3.1 Results

Initial model

The first model uses a small grid and is used to determine possible mixing parameters for the Metropolis Hastings algorithm. The data is filtered to a 8×8 grid which results in 693 observations being mapped onto the 4×4 grid. Filtering the observations to a 4×4 spatial grid used 299 observations however the MCMC chain resulting from using this very small data set has problems converging thus the larger data set is used in this initial model. The results for this model are shown in Figure 7.12 and Tables 7.9 and 7.10, these results were obtained using a $N(0, 25^{-1})$ random walk update for μ and $\log(\tau_s)$ and a $N(0, 100^{-1})$ random walk update for $\log(\tau_o)$. The trace plots in Figure 7.12 have been thinned by a factor of 10 for storage reasons. They show reasonable mixing and suggest that this random walk update is suitable for the larger models. Table 7.9 gives the mean and standard deviation for the precision and mean temperature parameters along with the associated time series standard error for these parameters. The mean temperature proposed from the posterior distribution of μ is 14.25 and the actual mean of the data is 14.3 which suggests that the posterior distribution for μ has converged in the correct area.

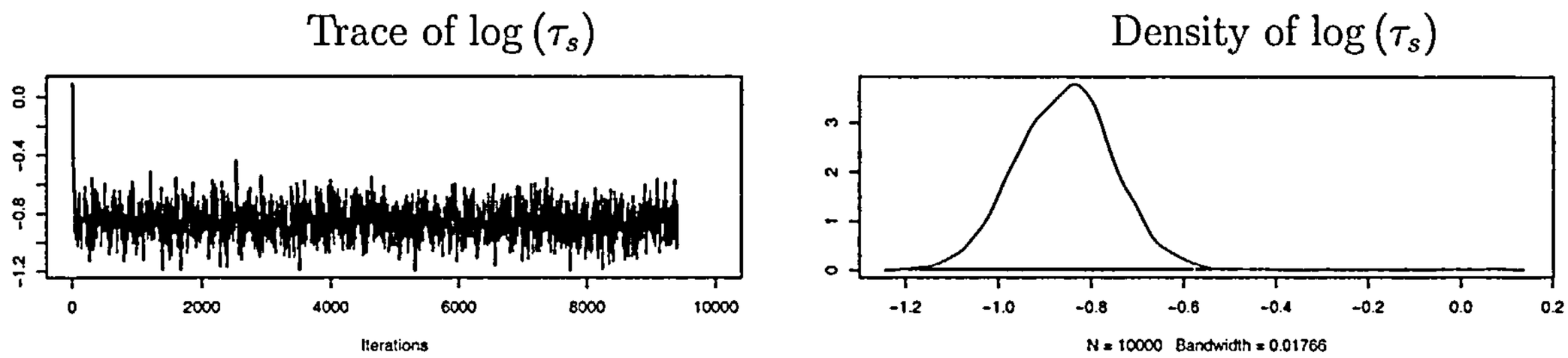


Figure 7.12: *Trace and density plots for ocean temperature data modelled using the fine model on a* 4×4 *spatial grid.*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-0.8526	0.11220	0.0011220	0.002038
$\log(\tau_o)$	-0.8154	0.06601	0.0006601	0.001205
μ	14.2521	0.30778	0.0030778	0.005522

Table 7.9: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 4×4 *spatial grid.*

Second model

The second model compares a fine and coarse approach for the simulation of the posterior distributions. The coarse model is developed on a 4×4 grid which corresponds to a

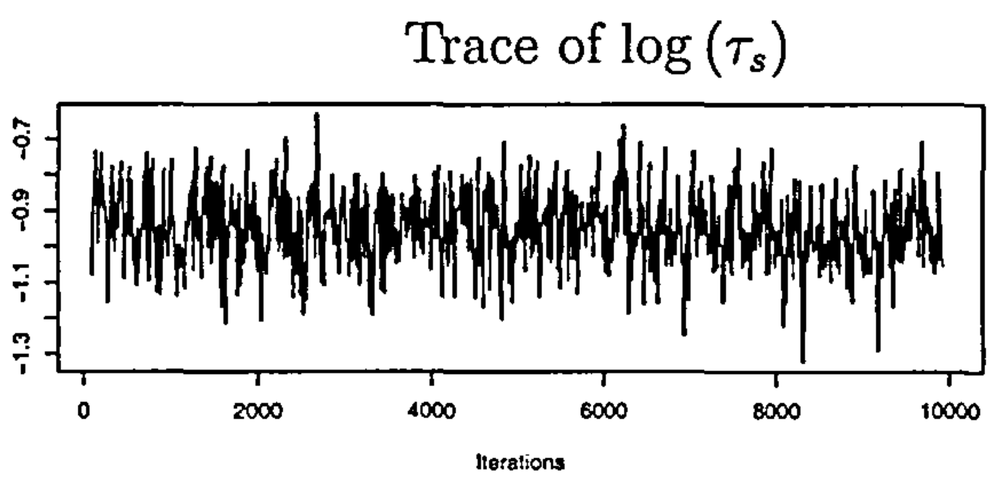
	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.0596	-0.9256	-0.8513	-0.7847	-0.6393
$\log(\tau_o)$	-0.9474	-0.8606	-0.8136	-0.7705	-0.6877
μ	13.6301	14.0434	14.2562	14.4611	14.8507

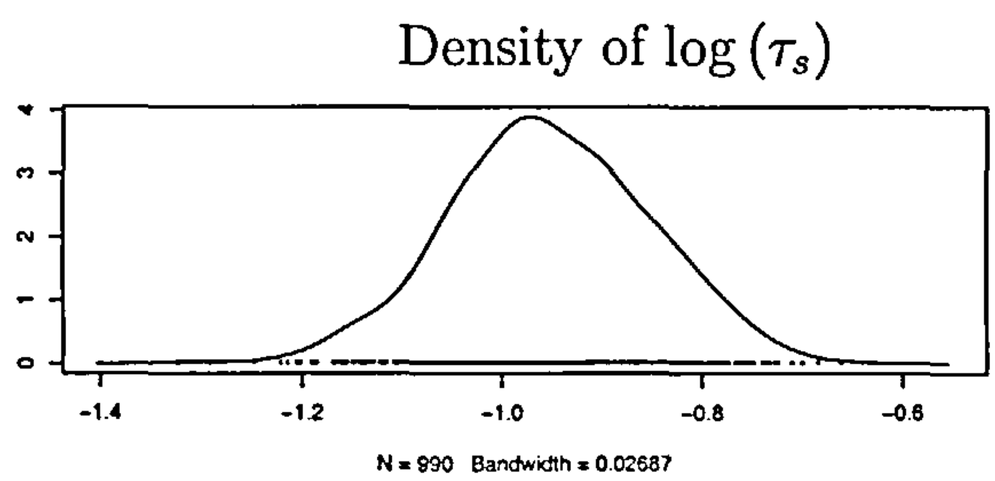
Table 7.10: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 4×4 *spatial grid.*

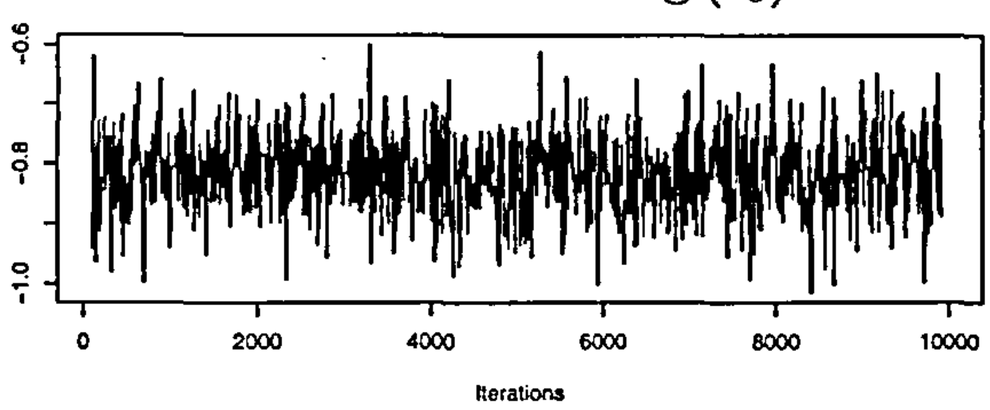
fine model run on a 8×8 grid. The data is filtered to a 8×8 grid which results in 693 observations being mapped onto the 4×4 grid for the coarse model. As in the above model if the data is filtered to the 4×4 spatial grid the coarse model has trouble converging thus the larger data set is used. Both models use a $N(0, 25^{-1})$ random walk update for μ and $\log(\tau_s)$ and a $N(0, 100^{-1})$ random walk update for $\log(\tau_o)$ and the prior distributions as listed in Equation (7.2). The results for the coarse model are shown in Figure 7.13 and Tables 7.11 and 7.12, the corresponding results for the fine model are shown in Figure 7.14 and Tables 7.13 and 7.14. Comparing the mean values for the posterior distributions of the three parameters shows a discrepancy between the two models for the precision parameters although the posterior distribution for the mean parameter is consistent between the two models. The coarse model suggests $\log(\tau_s) = -0.95$, $\log(\tau_o) = -0.82$ and $\mu = 14.4$ whereas the fine model suggests $\log(\tau_s) = -0.62$, $\log(\tau_o) - 0.31 =$ and $\mu = 14.46$. This discrepancy is due to the data which is filtered to map onto a 8×8 grid which means in the coarse model some locations are going to have more than one observation available which causes discrepancies in the simulated posterior distribution (Section 5.6.1).

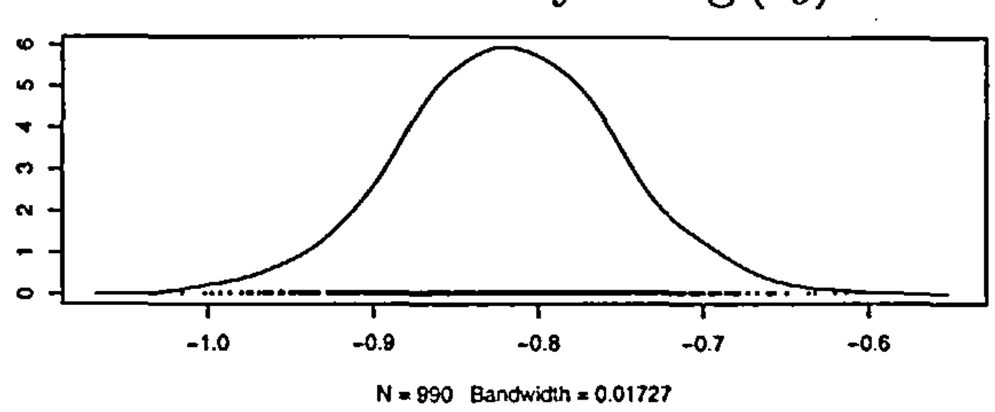
Final model

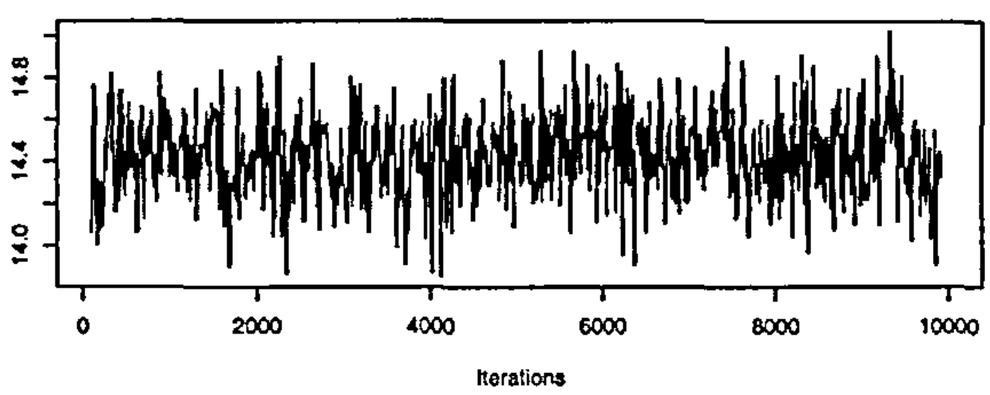
The final model uses a coarse approach applied to a 16×16 grid which corresponds to the fine model on a 32×32 grid which is a resolution of 1 degree $\times$ 1 degree. The data is filtered to a 16×16 spatial grid and results in 1377 observations being used. The MCMC











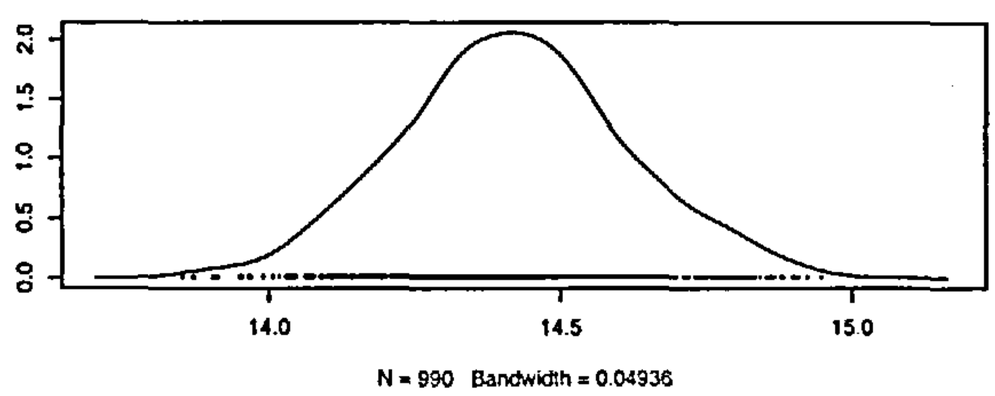


Figure 7.13: *Trace and density plots for ocean temperature data modelled using the coarse model on a* 4×4 *spatial grid after the removal of 100 iterations as burn-in*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-0.9497	0.10071	0.003201	0.004798
$\log(\tau_o)$	-0.8163	0.06535	0.002077	0.002240
μ	14.4240	0.19493	0.006195	0.008940

Table 7.11: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 4×4 *spatial grid.*

simulation is run for 5000 iterations on 8 nodes, the first 500 results are discarded as burn in which leaves 36000 simulations for analysis.

Figure 7.15 shows the resulting MCMC chain for this model, Tables 7.15 and 7.16 show the summary statistics and quantile ranges for these posterior distributions. This model

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.1480	-1.0201	-0.9553	-0.882	-0.7571
$\log(\tau_o)$	-0.9494	-0.8608	-0.8170	-0.774	-0.6861
μ	14.0571	14.2988	14.4218	14.547	14.8151

Table 7.12: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 4×4 *spatial grid.*

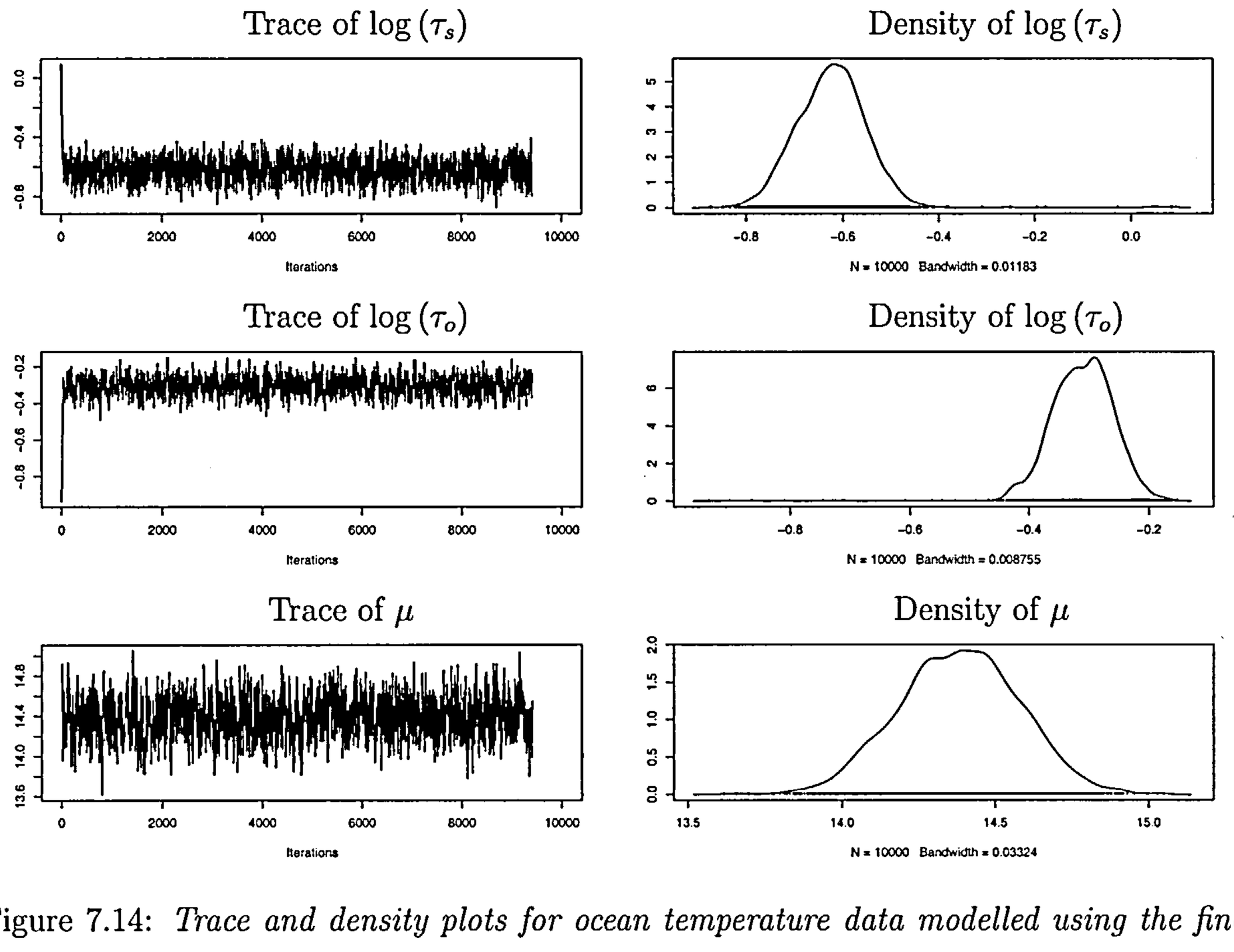


Figure 7.14: *Trace and density plots for ocean temperature data modelled using the fine model on a* 8×8 *spatial grid.*

suggests $\log(\tau_s) = -1.05$, $\log(\tau_o) = -0.3$ and $\mu = 14.4$. These values are used to form the mean temperature surface plots for the fine model which are shown in Figures 7.16 and 7.17. Figure 7.16 shows the surface plots for years (a) 1959 (b) 1973 (c) 1979 and (d)1983, from these plots the area which represents Africa is easily spotted at the front

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-0.6214	0.07542	0.0007542	0.001367
$\log(\tau_o)$	-0.3075	0.05557	0.0005557	0.001002
μ	14.3871	0.19789	0.0019789	0.003580

Table 7.13: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the fine model on a* 8×8 *spatial grid.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-0.7576	-0.6699	-0.6212	-0.5755	-0.4871
$\log(\tau_o)$	-0.4136	-0.3414	-0.3055	-0.2716	-0.2130
μ	14.0077	14.2547	14.3891	14.5221	14.7663

Table 7.14: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the fine model on a* 8×8 *spatial grid.*

	Mean	SD	Naive SE	Time-series SE
$\log(\tau_s)$	-1.057	0.06379	0.002927	0.005261
$\log(\tau_o)$	-0.301	0.05179	0.002376	0.003849
μ	14.424	0.11992	0.005502	0.009946

Table 7.15: *The mean and standard deviation for the precision and mean temperature parameters along with the time series standard error for ocean temperature data modelled using the coarse model on a* 16×16 *spatial grid.*

right hand corner bounded by a ridge of warmer water along the coastline. These surface plots also show a slight variation in the back left corner and an increase in temperature around the Canary Islands in the front left corner of the surface plots. The predictive surface plot in Figure 7.17 is similar to these surface plots and as in the small ocean

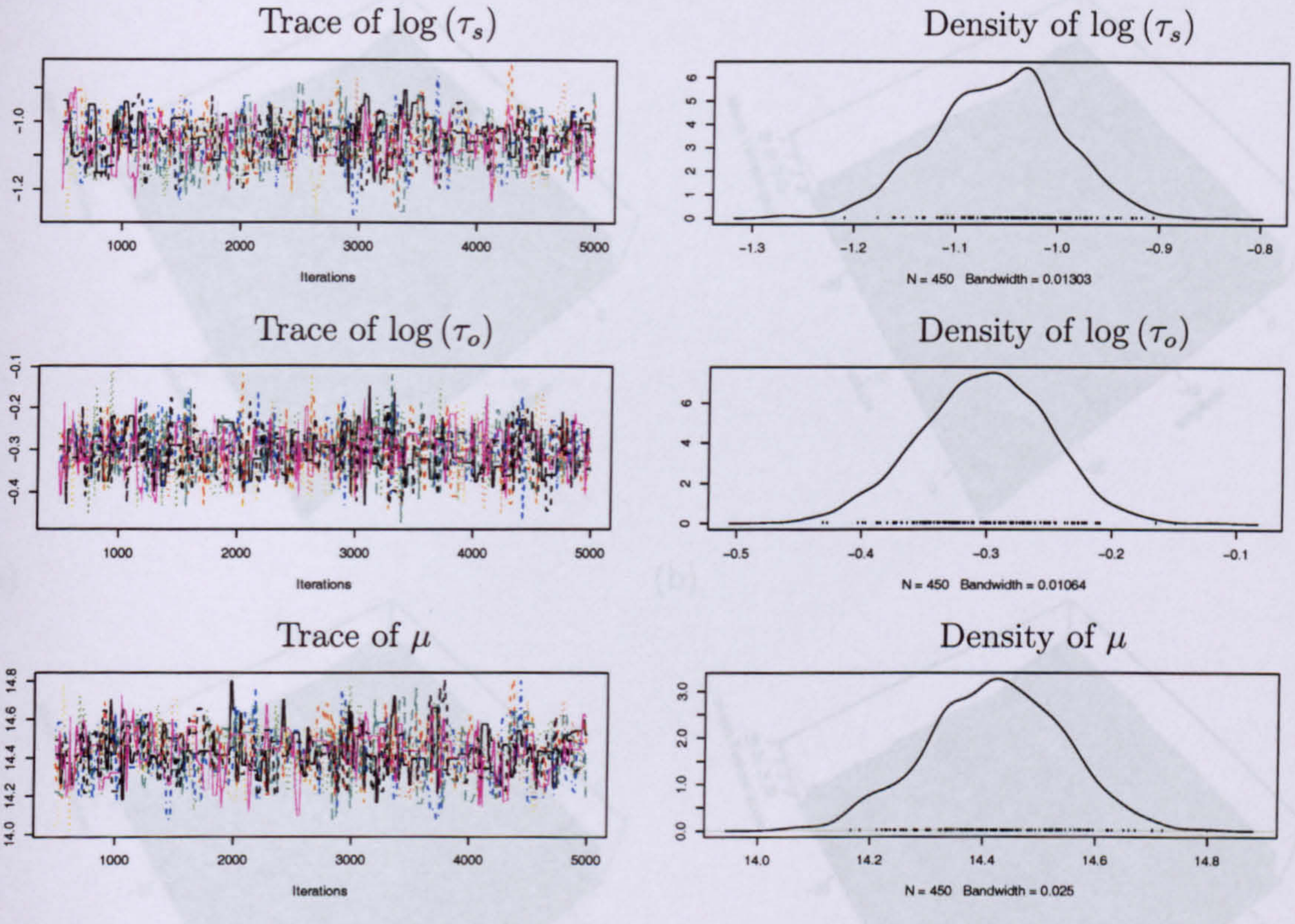


Figure 7.15: *Trace and density plots for ocean temperature data modelled using the coarse model on a* 16×16 *spatial grid.*

	2.5%	25%	50%	75%	97.5%
$\log(\tau_s)$	-1.1862	-1.1006	-1.0552	-1.0157	-0.938
$\log(\tau_o)$	-0.4054	-0.3348	-0.2998	-0.2641	-0.202
μ	14.1825	14.3426	14.4252	14.5070	14.654

Table 7.16: *The 2.5%, 25%, 50%, 75% and 97.5% quantiles for the precision and mean temperature parameters for ocean temperature data modelled using the coarse model on a* 16×16 *spatial grid.*

example is much smoother due to there being no observations available for this predictive time .

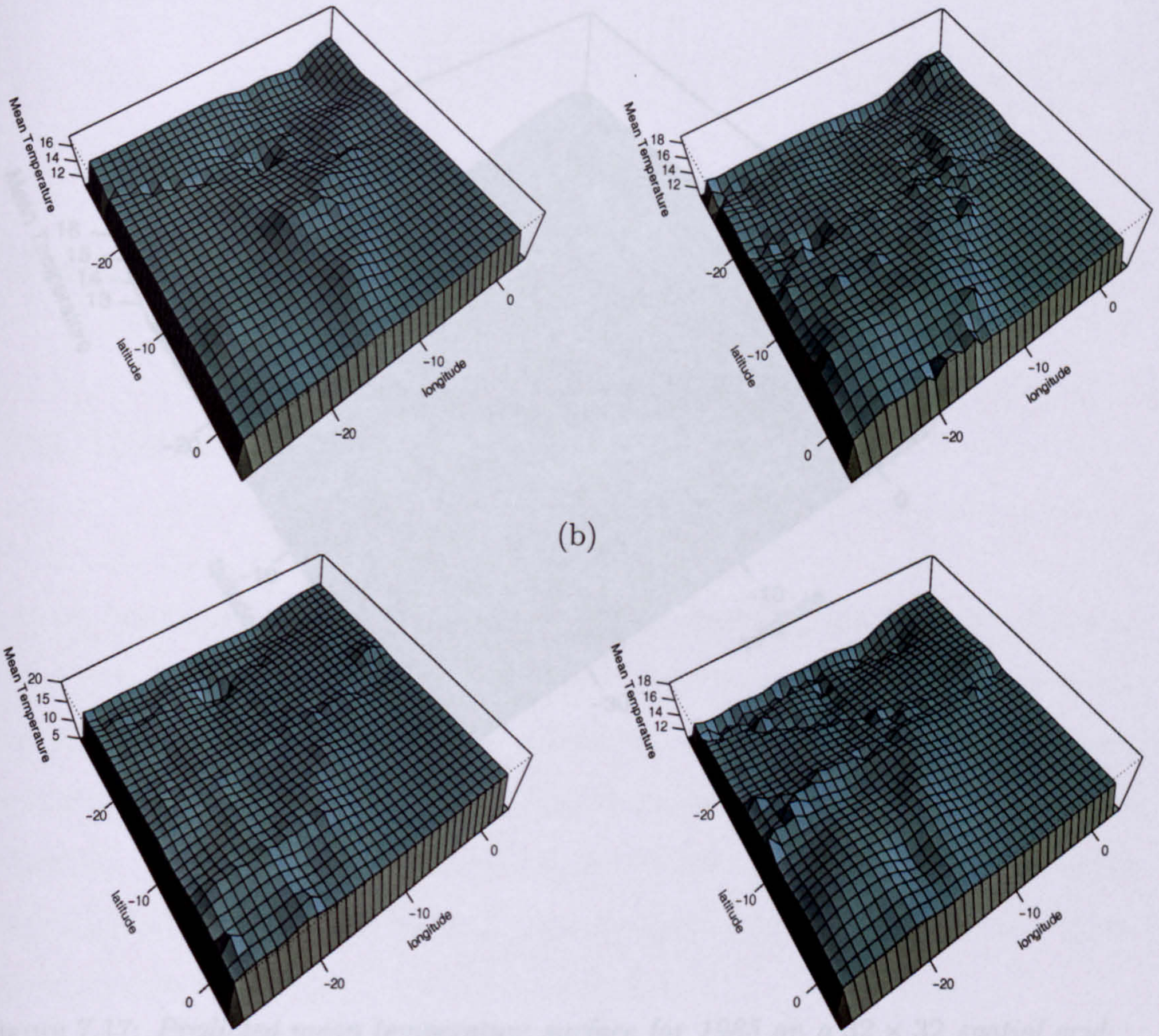


Figure 7.16: *Mean temperature surface for (a) 1959 (b) 1973 (c) 1979 (d) 1983 on a* 32×32 *spatial grid.*

7.4 Discussion

Both studies described in this chapter have examined ocean temperature data and have used the approaches discussed in Chapters 3 and 5 for modelling the spatio-temporal data and using the techniques discussed in Chapter 4 to simulate the posterior distributions of the model parameters.

The first data set was for a thin rectangular area of the Atlantic ocean where there was

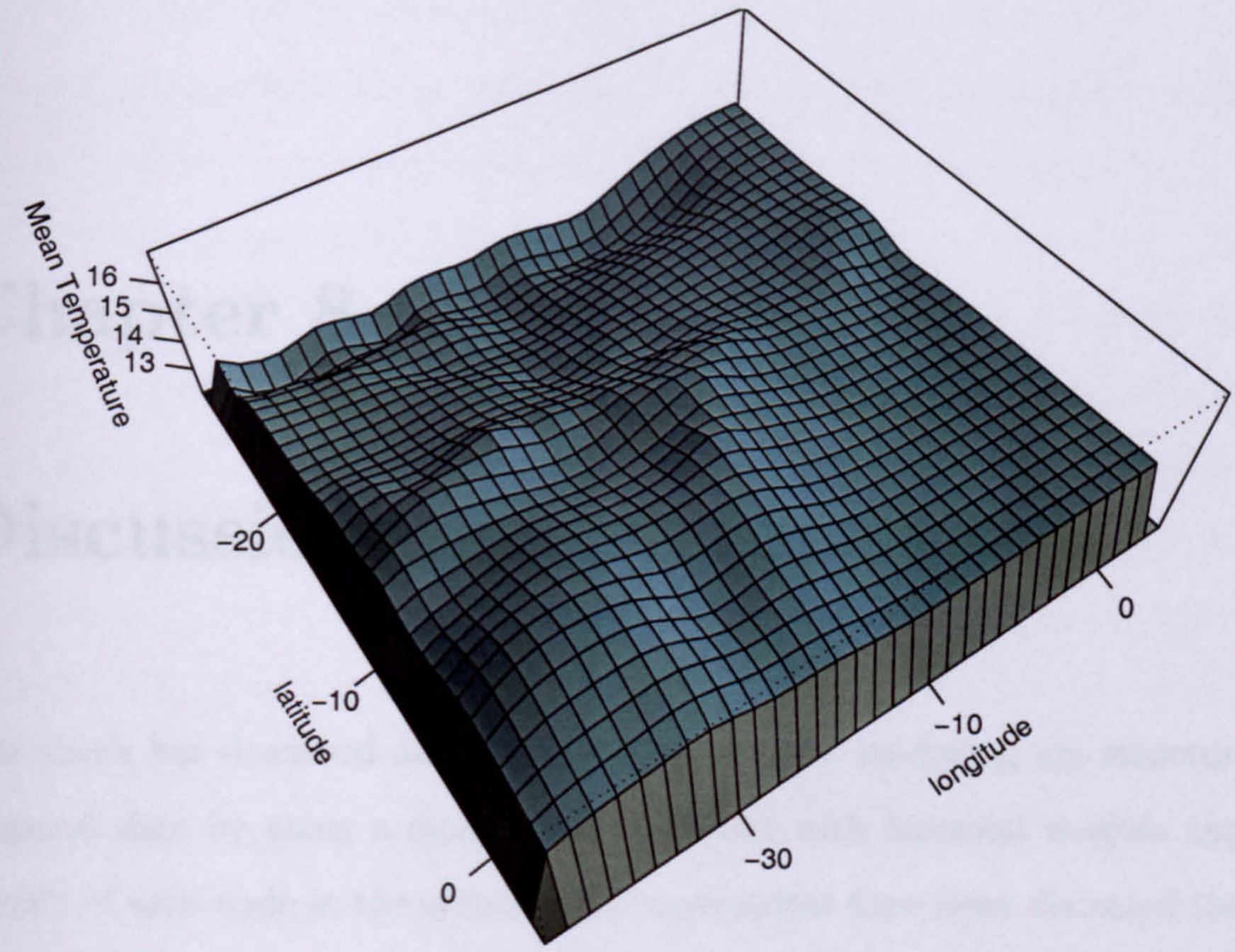


Figure 7.17: *Predicted mean temperature surface for 1985 on a* 32×32 *spatial grid.*

no land mass in the mode whereas the second model was for a square region of the North Atlantic which covered parts of Africa and Spain as well as the Atlantic ocean. The large amount of land in the second model (and the corresponding lack of temperature data) probably caused this model to perform less well than the first model. This land mass could also have contributed to the discrepancies between the coarse and fine models.

Chapter 8

Discussion and Further work

This thesis has described an hierarchical method for modelling the structure of spatio-temporal data by using a dynamic linear model with binomial weights applied to the parents of each node in the system. Two approaches have been discussed the first using a fine grid representation for the data the second applying a coarsened grid to the data to reduce the computational time. Model parameters are simulated from their posterior distributions using Markov chain Monte Carlo techniques. This thesis concluded with two examples using North Atlantic ocean temperature data.

The spatio-temporal model was defined in Chapter 2, the evolution of the system through time is described by defining a set of parents for each node. In this thesis information was passed through the system by using binomial edge weights applied to the parent nodes. Initially three models were discussed the 0+1D model, 1+1D model and finally the 2+1D model. The latter two models were then discussed in further detail in later chapters.

Chapter 3 examined three different approaches to dealing with edge effects and concluded that the use of conditional multivariate Normal methods to determine the edge adjustments was the best approach. This method adjusts both the edge weights and the associated state precision. The edge adjustments were generated using R and are listed in Table A.1 for the fine model and Table A.2 for the coarse model. The edge adjustments

were also examined to see the effect of having non unit precision and non zero mean, changing the precision had no effect on the adjusted edge weight and it scaled the conditional precision. Altering the mean value results in the edge weights being increased by $\mu\left(1 - a^T 1\right)$ and no change to the conditional precision.

The fine model from Chapter 3 whilst providing sensible parameter estimates via MCMC techniques was computationally intensive due to the number of latent points required for spatio-temporal examples. Chapter 5 provides a way to coarsen the latent structure and reduce the computational time required to generate parameter estimates. This approach along with parallel processing techniques allows large data sets to be examined in a relatively short space of time (compared with a serial processor with the fine model). Unfortunately the coarse model like the fine model is sensitive to edge effects and is unreliable for small models.

The thesis concludes with a number of examples; the examination of temporal and spatial dependencies and two case studies. The first example concluded that the use of a single parameter to represent spatial and temporal dependence was adequate. The two case studies examined two different areas of the Atlantic ocean the first, near the equator, was entirely ocean whilst the second had large areas of land in the spatial grid.

There are several natural extensions to the work presented in this thesis; further investigation into the spatial and temporal dependency parameters, the optimisation of the coarse model including investigation into the minimum grid size for comparable solutions to the fine model and the use and comparison of other data sets. Chapter 1 gave a brief overview of spatio-temporal models which have appeared in the literature, many of these models included regression parameters which have not been examined in this thesis. A natural extension would be to develop this spatio-temporal model to allow for the simulation of regression parameters by embedding this model into a larger hierarchical model. Chapters 5 and 7 required observations to be mapped to lattice grid points, the way in which this is done is very important and requires further investigation, particularly in models where the observations occur irregularly in space and time.

Bibliography

Bailey, B. A., Berliner, L. M., Collins, W. & Kiehl, J. T. (1997), Modeling the spatial and temporal distribution of cloud cover, available from http://www.stat.uiuc.edu/~babailey/.

Bartlett, M. (1978), 'Nearest neighbour models in the analysis of field experiments', *Journal of the Royal Statistical Society, Series B* **40**(2), 147–174.

Berliner, L. M., Royal, J. A., Wikle, C. K. & Milliff, R. F. (1999), Bayesian methods in the atmospheric sciences, *in* J. M. Bernado, J. O. Berger & A. F. M. Smith, eds, 'Bayesian Statistics 6', Oxford University Press, pp. 83–100.

Besag, J. (1974), 'Spatial interaction and the statistical analysis of lattice systems', *Journal of the Royal Statistical Society, Series B* **36**(2), 192–236.

Besag, J. (2001), Markov chain Monte Carlo for statistical inference, Center for Statistics and the Social Sciences working paper.

Besag, J. & Kooperberg, C. (1995), 'On conditional and intrinsic autoregressions', *Biometrika* **82**(4), 733–46.

Besag, J., Green, P., Higdon, D. & Mengersen, K. (1995), 'Bayesian computation and stochastic systems', *Statistical Science* **10**(Issue 1), 3–41.

Brown, P. E., Diggle, P. J., Lord, M. E. & Young, P. C. (2001), 'Space time calibration of radar rainfall data', *Applied Statistics* **50**(Part 2), 224–241.

Carrol, R. J., Chen, R., George, E. I., Newton, H. J., Schmiediche, H. & Wang, N. (1997), 'Ozone exposure and population density in Harris county, Texas', *Journal of the American Statistical Society* **92**(438), 392–438. including comment.

Christakos, G. (1991), 'On certain classes of spatiotemporal random fields with applications to space-time data processing', *IEEE Transactions on Systems, Man and Cybernetics* **24**(4), 861–875.

Christakos, G. (1992), *Random Field Models in Earth Sciences*, Academic Press, Inc.

Christakos, G. (1998), 'Spatiotemporal information systems in soil and environmental sciences', *Geoderma* **85**, 141–179.

Christakos, G. & Bogaert, P. (1996), 'Spatiotemporal analysis of spring water ion processes derived from measurements at the Dyle Basin in Belgium', *IEEE Transactions on Geoscience and Remote Sensing* **34**(3), 626–642.

Christakos, G. & Vyas, V. M. (1998), 'A composite space/time approach to studying ozone distributions over the Eastern United States', *Atmospheric Environment* **32**(16), 2845–2857.

Cliff, A. D. & Ord, J. K. (1975), 'Model building and the analysis of spatial data in human geography', *Journal of the Royal Statistical Society, Series B* **37**(3), 297–348.

Cressie, N. A. & Mugglin, A. S. (2000), Spatio-temporal hierarchical modeling of an infectious disease from (simulated) count data., *in* 'Computational Statistics'.

Cressie, N. A. C. (1991), *Statistics for Spatial Data*, John Wiley and Sons Inc.

Gamerman, D. (1997), *Markov Chain Monte Carlo. Stochastic simulation for Bayesian inference*, Chapman & Hall.

Garside, L. M. & Wilkinson, D. J. (2003), Dynamic lattice-markov spatio-temporal models for environmental data, *in* J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid,

D. Heckerman, A. F. . M. Smith & M. West, eds, 'Bayesian Statistics 7', Oxford University Press, pp. 535–542.

Gelfand, A. E. & Smith, A. F. M. (1990), 'Sampling-based approaches to calculating marginal densities', *Journal of the American Statistical Society* **85**(410), 398–409.

Geman, S. & Geman, D. (1984), 'Stochastic relaxation, Gibbs distributions and the Bayesian restoration of image', *IEEE transactions on pattern analysis and machine intelligence* **6**, 721–741.

Genovese, C. R. (1999), Functional magnetic resonance imaging and spatio-temporal inference, *in* J. M. Bernado, J. O. Berger & A. F. M. Smith, eds, 'Bayesian Statistics 6', Oxford University press, pp. 225–274.

Genovese, C. R. (2000), 'A Bayesian time course model for functual magnetic resonance imaging data', *Journal of the American Statistical Society* **95**(451), 691–719.

Haas, T. C. (1995), 'Local prediction of a spatio-temporal process with an application to wet sulfate deposition', *Journal of the American Statistical Society* **90**(432), 1189–1199.

Handcock, M. S. & Wallis, J. R. (1994), 'An approach to statistical spatial-temporal modeling of meteorological fields', *Journal of the American Statistical Society* **89**(426), 368–378.

Haslett, J. & Raftery, A. E. (1989), 'Space-time modelling with long-memory dependence: assessing Ireland's wind power resource', *Applied Statistics* **38**(1), 1–50.

Hastings, W. K. (1970), 'Monte Carlo sampling methods using Markov chains and their applications', *Biometrika* **57**(1), 97–109.

Higdon, D. (1998), 'A process-convolution approach to modelling temperatures in the North Atlantic ocean', *Environmental and Ecological Statistics* **5**, 173–190.

Higdon, D., Lee, H. & Holloman, C. (2003), Markov chain Monte Carlo-based approaches for inference in computationally intensive inverse problems, *in* J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. . M. Smith & M. West, eds, 'Bayesian Statistics 7', Oxford University Press, pp. 181–197.

Høst, G., Omre, H. & Switzer, P. (1995), 'Spatial interpolation errors for monitoring data', *Journal of the American Statistical Society* **90**(431), 853–861.

Huang, H.-C. & Cressie, N. (1996), 'Spatio-temporal prediction of snow water equivalent using the Kalman filter', *Computational Statistics & Data Analysis* **22**, 159–175.

Hurn, M. A., Rue, H. & Sheehan, N. A. (1999), 'Block updating in constrained Markov chain Monte Carlo sampling', *Statistics and probability letters* **41**, 353–361.

Knorr-Held, L. & Besag, J. (1998), 'Modelling risk from a disease in time and space', *Statistics in Medicine* **17**, 2045–2060.

Lavine, M. & Loizier, S. (1999), 'A Markov random field spatio-temporal analysis of ocean temperature', *Environmental and Ecological Statistics* **6**, 249–273.

Lawson, A., Biggeri, A. & Dreassi, E. (1999), Edge effects in disease mapping, *in* A. Lawson, A. Biggeri, D. Bohning, E. Lesaffre, J.-F. Viel & R. Bertollini, eds, 'Disease mapping and Risk Assessment for Public health', John Wiley and Sons Inc., chapter 6, pp. 85–98.

Meiring, W., Guttorp, P. & Sampson, P. D. (1998), 'Space-time estimation of grid-cell hourly ozone levels for assessment of a deterministic model', *Environmental and Ecological Statistics* **5**, 197–222.

Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. (1953), 'Equation of state calculations by fast computer machines', *Journal of Chemical Physics* **21**, 1087–1092.

Mugglin, A. S., Cressie, N. & Gemmell, I. (2002), 'Hierarchical statistical modelling of influenza epidemic dynamics in space and time', *Statistics in Medicine* **21**, 2703–2721.

Pfeifer, P. E. & Deutsch, S. J. (1980*a*), 'Identification and interpretation of first order space-time ARMA models', *Technometrics* **22**(3), 397–408.

Pfeifer, P. E. & Deutsch, S. J. (1980*b*), 'Independence and sphericity test for the residuals of space-time ARMA models', *Communications in Statistics. Part B. Simulation and Computation* **B(9)5**, 533–549.

Pfeifer, P. E. & Deutsch, S. J. (1980*c*), 'Stationarity and invertibility regions for low order STARMA models', *Communications in Statistics. Part B. Simulation and Computation* **B(9)5**, 551–562.

Pfeifer, P. E. & Deutsch, S. J. (1980*d*), 'A three stage iterative procedure for space-time modeling', *Technometrics* **22**(1), 35–47.

Robert, C. P. (1995), 'Convergence control methods for Markov chain Monte Carlo algorithms', *Statistical Science* **10**(3), 231–253.

Royle, J. A., Berliner, L. M. & Wikle, C. K. (1998), A Hierarchical spatial model for constructing wind fields from scatterometer data in the Labrador sea, *in* C. Gatsonis, R. E. Kass, B. Carlin, A. Cariquiry, A. Gelman, I. Verdinelli & M.West, eds, 'Case Studies in Bayesian Statistics IV', Springer-Verlag, pp. 367–381.

Sanso, B. & Guenni, L. (1999), 'Venezuelan rainfall data analysed by using a Bayesian space-time model', *Applied Statistics* **48**(Part 3), 345–362.

Scott, S. L. (2002), 'Bayesian methods for hidden Markov models: Recursive computing in the 21st century', *Journal of the American Statistical Society* **97**(457), 337–351.

Spiegelhalter, D. J., Thomas, A., Best, N. G. & Gilks, W. R. (1996), *BUGS: Bayesian inference Using Gibbs Sampling, Version 0.5, (version ii)*, MRC Biostatistics Unit, Cambridge, UK, Cambridge, UK.

Stroud, J. R., Muller, P. & Sanso, B. (2001), 'Dynamic models for spatiotemporal data', *Journal of the Royal Statistical Society, Series B* **63**(4), 673–689.

Tanner, M. A. & Wong, W. H. (1987), 'The calculation of posterior distributions by data augmentation', *Journal of the American Statistical Society* **82**(398), 528–540.

Tonellato, S. (1998), Spatial prediction with Space-time models, Rapporti di Ricerca (Technical Report) 2.

Tonellato, S. F. (2001), 'A multivariate time series model for the analysis and prediction of carbon monoxide atmospheric concentrations', *Applied Statistics* **50**(Part 2), 187–200.

van Rossum, G. & Drake, F. L. (2003), *Introduction to Python*, Network Theory Ltd.

Venables, W. N., Smith, D. & R Development core team (2003), *An introduction to R. Version 1.8.1.*

Vyas, V. M. & Christakos, G. (1997), 'Spatiotemporal analysis and mapping of sulfate decomposition data over Eastern U.S.A', *Atmospheric Environment* **31**(21), 3623–3633.

Waller, K. A., Carlin, B. P., Xia, H. & Gelfand, A. E. (1997), 'Hierarchical spatio-temporal mapping of disease rates', *Journal of the American Statistical Society* **92**(438), 607–617.

West, M. & Harrison, J. (1997), *Bayesian Forecasting and Dynamic Models*, 2 edn, Springer, New York.

Whiley, M. & Wilson, S. P. (2002), Parallel algorithms for Markov Chain Monte Carlo methods in latent spatial Gaussian models, https://hera.rz.hu-berlin.de/compstat2002/userinfolistpapers.php4#P.

Wikle, C., Berliner, M. & Cressie, N. (1998), 'Hierarchical Bayesian space-time models', *Environmental and Ecological Statistics* **5**, 117–154.

Wikle, C. K. & Cressie, N. (1999), 'A dimension reduced approach to space-time Kalman filtering', *Biometrika* **86**(4), 815–829.

Wikle, C. K., Millif, R. F., Nychka, D. & Berliner, L. M. (1997), Spatio-temporal hierarchical Bayesian modeling: Tropical ocean surface winds, Technical report, University of Missouri.

Wilkinson, D. (2001), GDAGsim 0.2 user guide, Statistics Preprint STA01,1, University of Newcastle.

Wilkinson, D. & Yeung, S. (2001), A sparse matrix approach to Bayesian computation in large linear models, Statistics Preprint STA01,2, University of Newcastle.

Wilkinson, D. J. (2004), Parallel Bayesian computation, *in* E. J. Kontoghiorghes, ed., 'Handbook of Parallel Computing and Statistics', Statistics: Textbooks and Monographs, Marcel Dekker, New York, pp. xx–xx. To appear.

Wilkinson, D. J. & Yeung, S. K. (2002), 'Conditional simulation from highly structured Gaussian systems, with applications to blocking-MCMC for the Bayesian analysis of very large linear models', *Statistics and Computing* **12**, 287–300.

Appendix A

Reference Tables of Edge Weights for the 2+1D Model in the Fine and Coarse Grid Setting.

A.1 Reference table for the fine model

Table A.1 is the complete reference table for the edge adjustments used in the fine model with isotropic binomial weights applied to the internal nodes for $\alpha \in [0, 1)$. The edge weights are given for nodes that lie along the edge of the system where 0 step is the weight applied to the spatial node itself at the previous time period, 1 step is the weight applied to the spatial node which is a first order parent at the previous time period and 2 step is the weight applied to the spatial node which is a second order parent at the previous time period. The precision adjustments are given in the last two columns of the table. The internal nodes have edge weights defined by the evolution matrix G and the precision is the unaffected.

α	Edge Weights						Precision Adjustment	
	Edge nodes			Corner Nodes			Edge nodes	Corner nodes
	2 Step	1 Step	0 Step	2 Step	1 Step	0 Step		
0.01	0.0006	0.0013	0.0025	0.0006	0.0013	0.0025	1.0000	1.0000
0.02	0.0013	0.0025	0.0050	0.0013	0.0025	0.0050	1.0000	1.0000
0.03	0.0019	0.0038	0.0075	0.0019	0.0038	0.0075	1.0000	1.0000
0.04	0.0025	0.0050	0.0100	0.0025	0.0050	0.0100	1.0000	0.9999
0.05	0.0031	0.0063	0.0125	0.0031	0.0063	0.0125	0.9999	0.9999
0.06	0.0038	0.0075	0.0150	0.0038	0.0075	0.0150	0.9999	0.9998
0.07	0.0044	0.0088	0.0175	0.0044	0.0088	0.0175	0.9999	0.9998
0.08	0.0050	0.0100	0.0200	0.0050	0.0100	0.0200	0.9998	0.9997
0.09	0.0056	0.0113	0.0225	0.0056	0.0113	0.0225	0.9998	0.9997
0.10	0.0063	0.0125	0.0250	0.0063	0.0125	0.0250	0.9998	0.9996
0.11	0.0069	0.0138	0.0275	0.0069	0.0138	0.0275	0.9997	0.9995

α	Edge Weights						Precision Adjustment	
	Edge nodes			Corner Nodes			Edge nodes	Corner nodes
	2 Step	1 Step	0 Step	2 Step	1 Step	0 Step		
0.12	0.0075	0.0150	0.0300	0.0075	0.0150	0.0301	0.9997	0.9994
0.13	0.0081	0.0163	0.0325	0.0081	0.0163	0.0326	0.9996	0.9993
0.14	0.0088	0.0175	0.0351	0.0088	0.0176	0.0351	0.9995	0.9992
0.15	0.0094	0.0188	0.0376	0.0094	0.0188	0.0376	0.9995	0.9990
0.16	0.0100	0.0201	0.0401	0.0100	0.0201	0.0401	0.9994	0.9989
0.17	0.0106	0.0213	0.0426	0.0107	0.0213	0.0427	0.9993	0.9987
0.18	0.0113	0.0226	0.0451	0.0113	0.0226	0.0452	0.9992	0.9986
0.19	0.0119	0.0239	0.0476	0.0119	0.0239	0.0477	0.9991	0.9984
0.20	0.0125	0.0251	0.0502	0.0126	0.0252	0.0503	0.9991	0.9983
0.21	0.0132	0.0264	0.0527	0.0132	0.0264	0.0528	0.9990	0.9981
0.22	0.0138	0.0277	0.0552	0.0138	0.0277	0.0554	0.9989	0.9979
0.23	0.0144	0.0289	0.0577	0.0145	0.0290	0.0579	0.9987	0.9977
0.24	0.0151	0.0302	0.0603	0.0151	0.0303	0.0605	0.9986	0.9975
0.25	0.0157	0.0315	0.0628	0.0157	0.0315	0.0631	0.9985	0.9973
0.26	0.0163	0.0328	0.0654	0.0164	0.0328	0.0656	0.9984	0.9970
0.27	0.0169	0.0340	0.0679	0.0170	0.0341	0.0682	0.9983	0.9968
0.28	0.0176	0.0353	0.0704	0.0177	0.0354	0.0708	0.9981	0.9965
0.29	0.0182	0.0366	0.0730	0.0183	0.0367	0.0734	0.9980	0.9963
0.30	0.0189	0.0379	0.0755	0.0190	0.0380	0.0760	0.9978	0.9960
0.31	0.0195	0.0392	0.0781	0.0196	0.0393	0.0786	0.9977	0.9957
0.32	0.0201	0.0405	0.0807	0.0202	0.0406	0.0812	0.9975	0.9955
0.33	0.0208	0.0418	0.0832	0.0209	0.0420	0.0838	0.9974	0.9952
0.34	0.0214	0.0431	0.0858	0.0215	0.0433	0.0865	0.9972	0.9948
0.35	0.0220	0.0444	0.0884	0.0222	0.0446	0.0891	0.9970	0.9945

α	Edge Weights						Precision Adjustment	
	Edge nodes			Corner Nodes			Edge nodes	Corner nodes
	2 Step	1 Step	0 Step	2 Step	1 Step	0 Step		
0.36	0.0227	0.0457	0.0910	0.0229	0.0459	0.0917	0.9969	0.9942
0.37	0.0233	0.0470	0.0935	0.0235	0.0473	0.0944	0.9967	0.9939
0.38	0.0240	0.0484	0.0961	0.0242	0.0486	0.0971	0.9965	0.9935
0.39	0.0246	0.0497	0.0987	0.0248	0.0499	0.0997	0.9963	0.9931
0.40	0.0253	0.0510	0.1013	0.0255	0.0513	0.1024	0.9961	0.9928
0.41	0.0259	0.0523	0.1039	0.0262	0.0526	0.1051	0.9959	0.9924
0.42	0.0265	0.0537	0.1065	0.0268	0.0540	0.1078	0.9957	0.9920
0.43	0.0272	0.0550	0.1092	0.0275	0.0554	0.1105	0.9954	0.9916
0.44	0.0278	0.0564	0.1118	0.0282	0.0567	0.1133	0.9952	0.9911
0.45	0.0285	0.0577	0.1144	0.0289	0.0581	0.1160	0.9950	0.9907
0.46	0.0291	0.0591	0.1171	0.0295	0.0595	0.1188	0.9947	0.9903
0.47	0.0298	0.0604	0.1197	0.0302	0.0609	0.1215	0.9945	0.9898
0.48	0.0305	0.0618	0.1224	0.0309	0.0623	0.1243	0.9943	0.9893
0.49	0.0311	0.0632	0.1250	0.0316	0.0637	0.1271	0.9940	0.9888
0.50	0.0318	0.0646	0.1277	0.0323	0.0651	0.1299	0.9937	0.9883
0.51	0.0324	0.0659	0.1304	0.0330	0.0666	0.1328	0.9935	0.9878
0.52	0.0331	0.0673	0.1331	0.0337	0.0680	0.1356	0.9932	0.9873
0.53	0.0338	0.0687	0.1358	0.0344	0.0695	0.1385	0.9929	0.9867
0.54	0.0344	0.0702	0.1385	0.0351	0.0709	0.1414	0.9926	0.9862
0.55	0.0351	0.0716	0.1412	0.0358	0.0724	0.1443	0.9923	0.9856
0.56	0.0358	0.0730	0.1439	0.0365	0.0739	0.1472	0.9920	0.9850
0.57	0.0364	0.0744	0.1466	0.0372	0.0753	0.1501	0.9917	0.9844
0.58	0.0371	0.0759	0.1494	0.0380	0.0768	0.1531	0.9913	0.9838
0.59	0.0378	0.0773	0.1521	0.0387	0.0784	0.1561	0.9910	0.9831

α	Edge Weights						Precision Adjustment	
	Edge nodes			Corner Nodes			Edge nodes	Corner nodes
	2 Step	1 Step	0 Step	2 Step	1 Step	0 Step		
0.60	0.0385	0.0788	0.1549	0.0394	0.0799	0.1591	0.9906	0.9825
0.61	0.0392	0.0803	0.1577	0.0402	0.0814	0.1621	0.9903	0.9818
0.62	0.0398	0.0817	0.1605	0.0409	0.0830	0.1652	0.9899	0.9811
0.63	0.0405	0.0832	0.1633	0.0417	0.0846	0.1683	0.9896	0.9804
0.64	0.0412	0.0847	0.1661	0.0424	0.0861	0.1714	0.9892	0.9796
0.65	0.0419	0.0863	0.1689	0.0432	0.0877	0.1745	0.9888	0.9789
0.66	0.0426	0.0878	0.1718	0.0440	0.0894	0.1777	0.9884	0.9781
0.67	0.0433	0.0893	0.1746	0.0448	0.0910	0.1809	0.9880	0.9773
0.68	0.0440	0.0909	0.1775	0.0456	0.0926	0.1841	0.9875	0.9764
0.69	0.0447	0.0924	0.1804	0.0464	0.0943	0.1874	0.9871	0.9756
0.70	0.0454	0.0940	0.1833	0.0472	0.0960	0.1907	0.9867	0.9747
0.71	0.0462	0.0956	0.1863	0.0480	0.0977	0.1940	0.9862	0.9738
0.72	0.0469	0.0973	0.1892	0.0488	0.0995	0.1974	0.9857	0.9729
0.73	0.0476	0.0989	0.1922	0.0497	0.1013	0.2008	0.9852	0.9719
0.74	0.0484	0.1005	0.1952	0.0505	0.1030	0.2043	0.9847	0.9709
0.75	0.0491	0.1022	0.1982	0.0514	0.1049	0.2078	0.9842	0.9699
0.76	0.0498	0.1039	0.2013	0.0523	0.1067	0.2114	0.9837	0.9688
0.77	0.0506	0.1056	0.2043	0.0532	0.1086	0.2150	0.9831	0.9677
0.78	0.0514	0.1074	0.2074	0.0541	0.1105	0.2187	0.9826	0.9666
0.79	0.0521	0.1092	0.2105	0.0550	0.1125	0.2225	0.9820	0.9654
0.80	0.0529	0.1110	0.2137	0.0560	0.1145	0.2263	0.9814	0.9642
0.81	0.0537	0.1128	0.2169	0.0570	0.1166	0.2302	0.9808	0.9629
0.82	0.0545	0.1146	0.2201	0.0580	0.1187	0.2342	0.9801	0.9616
0.83	0.0553	0.1165	0.2234	0.0590	0.1208	0.2382	0.9795	0.9602
0.84	0.0561	0.1185	0.2267	0.0601	0.1230	0.2424	0.9788	0.9588

α	Edge Weights						Precision Adjustment	
	Edge nodes			Corner Nodes			Edge nodes	Corner nodes
	2 Step	1 Step	0 Step	2 Step	1 Step	0 Step		
0.85	0.0570	0.1205	0.2300	0.0612	0.1253	0.2466	0.9781	0.9573
0.86	0.0578	0.1225	0.2334	0.0623	0.1277	0.2510	0.9773	0.9557
0.87	0.0587	0.1246	0.2368	0.0635	0.1301	0.2555	0.9766	0.9541
0.88	0.0596	0.1267	0.2403	0.0647	0.1326	0.2602	0.9758	0.9523
0.89	0.0605	0.1289	0.2439	0.0660	0.1352	0.2650	0.9749	0.9505
0.90	0.0615	0.1312	0.2476	0.0674	0.1380	0.2699	0.9741	0.9486
0.91	0.0625	0.1336	0.2513	0.0688	0.1409	0.2751	0.9731	0.9465
0.92	0.0635	0.1360	0.2551	0.0704	0.1440	0.2806	0.9721	0.9443
0.93	0.0646	0.1386	0.2590	0.0721	0.1472	0.2863	0.9711	0.9419
0.94	0.0657	0.1413	0.2631	0.0739	0.1508	0.2924	0.9700	0.9393
0.95	0.0670	0.1443	0.2673	0.0760	0.1546	0.2990	0.9688	0.9364
0.96	0.0683	0.1474	0.2718	0.0784	0.1590	0.3061	0.9674	0.9332
0.97	0.0699	0.1509	0.2766	0.0813	0.1640	0.3142	0.9660	0.9295
0.98	0.0718	0.1550	0.2818	0.0850	0.1701	0.3236	0.9642	0.9250
0.99	0.0743	0.1602	0.2879	0.0906	0.1785	0.3358	0.9621	0.9189
1.00	0.0807	0.1702	0.2977	0.1068	0.1988	0.3612	0.9580	0.9055

Table A.1: *The reference table for the fine 2+1D model.*

A.2 Reference table for the coarse model

Table A.2 is the complete reference table for the edge adjustments used in the coarse model for $\alpha \in [0, 1)$. The edge weights are given for internal nodes, nodes along the edge of the system, nodes which are at a corner of the system and nodes that are in time period 1 where 0 step is the weight applied to the spatial node itself at time period $t-2$, 1 step is the weight applied to the spatial node which is a first order parent at the time period $t-2$, 2 step is the weight applied to the spatial node which is a second order parent at time period $t-2$ and $t-1$ the adjustment for the spatial node at the previous time period. The precision adjustments for these four situations are given in the last four columns of the table.

α	Edge Weights												Precision Adjustment			
	Internal nodes				Edge nodes nodes				Corner Nodes				Time	Internal	Edge	Corner	time
	2 Step	1 Step	0 Step	t-1	2 Step	1 Step	0 Step	t-1	2 Step	1 Step	0 Step	t-1	1	nodes	nodes	nodes	0
0.01	0.0000	0.0000	0.0000	0.0025	0.0000	0.0000	0.0000	0.0025	0.0000	0.0000	0.0000	0.0025	0.0025	1.0000	1.0000	1.0000	1.0000
0.02	0.0000	0.0000	0.0000	0.0050	0.0000	0.0000	0.0000	0.0050	0.0000	0.0000	0.0000	0.0050	0.0050	1.0000	1.0000	1.0000	1.0000
0.03	0.0000	0.0000	0.0001	0.0075	0.0000	0.0000	0.0001	0.0075	0.0000	0.0000	0.0001	0.0075	0.0075	0.9999	0.9999	0.9999	0.9999
0.04	0.0000	0.0000	0.0001	0.0100	0.0000	0.0000	0.0001	0.0100	0.0000	0.0000	0.0001	0.0100	0.0100	0.9999	0.9999	0.9999	0.9999
0.05	0.0000	0.0001	0.0002	0.0125	0.0000	0.0001	0.0002	0.0125	0.0000	0.0001	0.0002	0.0125	0.0125	0.9998	0.9998	0.9998	0.9998
0.06	0.0000	0.0001	0.0003	0.0150	0.0000	0.0001	0.0003	0.0150	0.0000	0.0001	0.0003	0.0150	0.0150	0.9997	0.9997	0.9997	0.9997
0.07	0.0000	0.0001	0.0004	0.0175	0.0000	0.0001	0.0004	0.0175	0.0000	0.0001	0.0004	0.0175	0.0175	0.9996	0.9996	0.9996	0.9996
0.08	0.0000	0.0002	0.0005	0.0200	0.0000	0.0002	0.0005	0.0200	0.0000	0.0002	0.0005	0.0200	0.0200	0.9995	0.9995	0.9995	0.9995
0.09	0.0000	0.0002	0.0006	0.0225	0.0000	0.0002	0.0006	0.0225	0.0000	0.0002	0.0006	0.0225	0.0225	0.9994	0.9994	0.9994	0.9994
0.10	0.0000	0.0002	0.0008	0.0250	0.0000	0.0002	0.0008	0.0250	0.0000	0.0002	0.0008	0.0250	0.0251	0.9992	0.9992	0.9992	0.9992
0.11	0.0000	0.0003	0.0009	0.0276	0.0000	0.0003	0.0009	0.0276	0.0000	0.0003	0.0009	0.0276	0.0276	0.9991	0.9991	0.9991	0.9991
0.12	0.0001	0.0003	0.0011	0.0301	0.0001	0.0003	0.0011	0.0301	0.0001	0.0003	0.0011	0.0301	0.0301	0.9989	0.9989	0.9989	0.9989
0.13	0.0001	0.0004	0.0013	0.0326	0.0001	0.0004	0.0013	0.0326	0.0001	0.0004	0.0013	0.0326	0.0326	0.9987	0.9987	0.9987	0.9987
0.14	0.0001	0.0005	0.0015	0.0351	0.0001	0.0005	0.0015	0.0351	0.0001	0.0005	0.0015	0.0351	0.0352	0.9985	0.9985	0.9985	0.9985
0.15	0.0001	0.0005	0.0018	0.0376	0.0001	0.0005	0.0018	0.0376	0.0001	0.0005	0.0018	0.0376	0.0377	0.9982	0.9982	0.9982	0.9982
0.16	0.0001	0.0006	0.0020	0.0402	0.0001	0.0006	0.0020	0.0402	0.0001	0.0006	0.0020	0.0402	0.0403	0.9980	0.9980	0.9980	0.9980
0.17	0.0001	0.0007	0.0023	0.0427	0.0001	0.0007	0.0023	0.0427	0.0001	0.0007	0.0023	0.0427	0.0428	0.9977	0.9977	0.9977	0.9977
0.18	0.0001	0.0008	0.0026	0.0453	0.0001	0.0008	0.0026	0.0453	0.0001	0.0008	0.0026	0.0453	0.0454	0.9974	0.9974	0.9974	0.9974
0.19	0.0002	0.0009	0.0029	0.0478	0.0002	0.0009	0.0029	0.0478	0.0002	0.0009	0.0029	0.0478	0.0479	0.9972	0.9972	0.9972	0.9971
0.20	0.0002	0.0010	0.0032	0.0503	0.0002	0.0010	0.0032	0.0504	0.0002	0.0010	0.0032	0.0504	0.0505	0.9968	0.9968	0.9968	0.9968
0.21	0.0002	0.0011	0.0035	0.0529	0.0002	0.0011	0.0035	0.0529	0.0002	0.0011	0.0035	0.0529	0.0531	0.9965	0.9965	0.9965	0.9965
0.22	0.0002	0.0012	0.0038	0.0555	0.0002	0.0012	0.0038	0.0555	0.0002	0.0012	0.0038	0.0555	0.0557	0.9962	0.9962	0.9962	0.9961
0.23	0.0002	0.0013	0.0042	0.0580	0.0002	0.0013	0.0042	0.0580	0.0002	0.0013	0.0042	0.0580	0.0583	0.9958	0.9958	0.9958	0.9958
0.24	0.0003	0.0014	0.0046	0.0606	0.0003	0.0014	0.0046	0.0606	0.0003	0.0014	0.0046	0.0606	0.0609	0.9954	0.9954	0.9954	0.9954
0.25	0.0003	0.0015	0.0050	0.0632	0.0003	0.0015	0.0050	0.0632	0.0003	0.0015	0.0050	0.0632	0.0635	0.9950	0.9950	0.9950	0.9950
0.26	0.0003	0.0017	0.0054	0.0658	0.0003	0.0017	0.0054	0.0658	0.0003	0.0017	0.0054	0.0658	0.0661	0.9946	0.9946	0.9946	0.9946
0.27	0.0003	0.0018	0.0058	0.0684	0.0003	0.0018	0.0058	0.0684	0.0003	0.0018	0.0058	0.0684	0.0688	0.9942	0.9942	0.9942	0.9941
0.28	0.0004	0.0019	0.0063	0.0710	0.0004	0.0019	0.0063	0.0710	0.0004	0.0019	0.0063	0.0710	0.0714	0.9937	0.9937	0.9937	0.9937
0.29	0.0004	0.0021	0.0068	0.0736	0.0004	0.0021	0.0068	0.0736	0.0004	0.0021	0.0068	0.0736	0.0741	0.9933	0.9933	0.9933	0.9932
0.30	0.0004	0.0023	0.0073	0.0762	0.0004	0.0023	0.0073	0.0762	0.0004	0.0023	0.0073	0.0762	0.0768	0.9928	0.9928	0.9928	0.9927
0.31	0.0005	0.0024	0.0078	0.0788	0.0005	0.0024	0.0078	0.0788	0.0005	0.0024	0.0078	0.0788	0.0795	0.9923	0.9923	0.9923	0.9922
0.32	0.0005	0.0026	0.0083	0.0815	0.0005	0.0026	0.0083	0.0815	0.0005	0.0026	0.0083	0.0815	0.0822	0.9918	0.9918	0.9918	0.9917
0.33	0.0005	0.0028	0.0088	0.0841	0.0005	0.0028	0.0088	0.0841	0.0005	0.0028	0.0089	0.0841	0.0849	0.9912	0.9912	0.9912	0.9911
0.34	0.0006	0.0030	0.0094	0.0868	0.0006	0.0030	0.0094	0.0868	0.0006	0.0030	0.0094	0.0868	0.0876	0.9907	0.9907	0.9907	0.9906
0.35	0.0006	0.0031	0.0100	0.0894	0.0006	0.0032	0.0100	0.0895	0.0006	0.0032	0.0100	0.0895	0.0904	0.9901	0.9901	0.9901	0.9900

α	Edge Weights													Precision Adjustment			
	Internal nodes				Edge nodes nodes				Corner Nodes				Time	Internal	Edge	Corner	time
	2 Step	1 Step	0 Step	t-1	2 Step	1 Step	0 Step	t-1	2 Step	1 Step	0 Step	t-1	1	nodes	nodes	nodes	0
0.36	0.0007	0.0033	0.0106	0.0921	0.0007	0.0034	0.0106	0.0921	0.0007	0.0034	0.0106	0.0921	0.0931	0.9895	0.9895	0.9895	0.9894
0.37	0.0007	0.0036	0.0112	0.0948	0.0007	0.0036	0.0112	0.0948	0.0007	0.0036	0.0113	0.0948	0.0959	0.9889	0.9889	0.9889	0.9887
0.38	0.0007	0.0038	0.0119	0.0975	0.0007	0.0038	0.0119	0.0975	0.0007	0.0038	0.0119	0.0975	0.0987	0.9883	0.9883	0.9882	0.9881
0.39	0.0008	0.0040	0.0125	0.1002	0.0008	0.0040	0.0126	0.1002	0.0008	0.0040	0.0126	0.1002	0.1015	0.9876	0.9876	0.9876	0.9874
0.40	0.0009	0.0042	0.0132	0.1030	0.0009	0.0042	0.0133	0.1030	0.0009	0.0042	0.0133	0.1030	0.1044	0.9869	0.9869	0.9869	0.9867
0.41	0.0009	0.0045	0.0139	0.1057	0.0009	0.0045	0.0140	0.1057	0.0009	0.0045	0.0140	0.1057	0.1072	0.9862	0.9862	0.9862	0.9860
0.42	0.0010	0.0047	0.0147	0.1085	0.0010	0.0047	0.0147	0.1085	0.0010	0.0047	0.0147	0.1085	0.1101	0.9855	0.9855	0.9855	0.9852
0.43	0.0010	0.0050	0.0154	0.1112	0.0010	0.0050	0.0155	0.1112	0.0010	0.0050	0.0155	0.1112	0.1130	0.9848	0.9848	0.9847	0.9845
0.44	0.0011	0.0053	0.0162	0.1140	0.0011	0.0053	0.0162	0.1140	0.0011	0.0053	0.0163	0.1140	0.1160	0.9840	0.9840	0.9840	0.9837
0.45	0.0012	0.0055	0.0170	0.1168	0.0012	0.0056	0.0171	0.1168	0.0012	0.0056	0.0171	0.1168	0.1189	0.9833	0.9832	0.9832	0.9828
0.46	0.0012	0.0058	0.0178	0.1196	0.0012	0.0059	0.0179	0.1196	0.0012	0.0059	0.0179	0.1197	0.1219	0.9825	0.9824	0.9824	0.9820
0.47	0.0013	0.0061	0.0187	0.1224	0.0013	0.0062	0.0187	0.1225	0.0013	0.0062	0.0188	0.1225	0.1249	0.9816	0.9816	0.9816	0.9811
0.48	0.0014	0.0065	0.0196	0.1253	0.0014	0.0065	0.0196	0.1253	0.0014	0.0065	0.0197	0.1253	0.1279	0.9808	0.9808	0.9807	0.9802
0.49	0.0015	0.0068	0.0205	0.1282	0.0015	0.0068	0.0205	0.1282	0.0015	0.0068	0.0206	0.1282	0.1310	0.9799	0.9799	0.9798	0.9793
0.50	0.0016	0.0071	0.0214	0.1310	0.0016	0.0071	0.0215	0.1311	0.0016	0.0071	0.0215	0.1311	0.1341	0.9790	0.9790	0.9789	0.9784
0.51	0.0017	0.0075	0.0224	0.1339	0.0017	0.0075	0.0224	0.1340	0.0017	0.0075	0.0225	0.1340	0.1372	0.9781	0.9781	0.9780	0.9774
0.52	0.0018	0.0078	0.0233	0.1369	0.0018	0.0079	0.0234	0.1369	0.0018	0.0079	0.0235	0.1369	0.1403	0.9772	0.9771	0.9771	0.9764
0.53	0.0019	0.0082	0.0244	0.1398	0.0019	0.0082	0.0244	0.1399	0.0019	0.0082	0.0245	0.1399	0.1435	0.9762	0.9761	0.9761	0.9753
0.54	0.0020	0.0086	0.0254	0.1428	0.0020	0.0086	0.0255	0.1428	0.0020	0.0086	0.0256	0.1429	0.1467	0.9752	0.9752	0.9751	0.9743
0.55	0.0021	0.0090	0.0265	0.1458	0.0021	0.0091	0.0266	0.1458	0.0021	0.0091	0.0266	0.1459	0.1500	0.9742	0.9741	0.9740	0.9732
0.56	0.0022	0.0094	0.0276	0.1488	0.0022	0.0095	0.0277	0.1488	0.0022	0.0095	0.0278	0.1489	0.1533	0.9732	0.9731	0.9730	0.9720
0.57	0.0024	0.0099	0.0287	0.1518	0.0024	0.0099	0.0288	0.1519	0.0024	0.0099	0.0289	0.1519	0.1566	0.9721	0.9720	0.9719	0.9709
0.58	0.0025	0.0103	0.0299	0.1549	0.0025	0.0104	0.0300	0.1550	0.0025	0.0104	0.0301	0.1550	0.1600	0.9710	0.9709	0.9708	0.9697
0.59	0.0027	0.0108	0.0310	0.1580	0.0027	0.0109	0.0312	0.1580	0.0027	0.0109	0.0313	0.1581	0.1634	0.9699	0.9698	0.9696	0.9684
0.60	0.0028	0.0113	0.0323	0.1611	0.0028	0.0114	0.0324	0.1612	0.0028	0.0114	0.0326	0.1613	0.1669	0.9687	0.9686	0.9685	0.9671
0.61	0.0030	0.0118	0.0335	0.1642	0.0030	0.0119	0.0337	0.1643	0.0030	0.0119	0.0339	0.1644	0.1704	0.9676	0.9674	0.9672	0.9658
0.62	0.0031	0.0123	0.0348	0.1674	0.0031	0.0124	0.0350	0.1675	0.0031	0.0124	0.0352	0.1676	0.1740	0.9663	0.9662	0.9660	0.9645
0.63	0.0033	0.0129	0.0362	0.1706	0.0033	0.0130	0.0364	0.1707	0.0033	0.0130	0.0366	0.1708	0.1776	0.9651	0.9649	0.9647	0.9631
0.64	0.0035	0.0134	0.0375	0.1738	0.0035	0.0135	0.0378	0.1740	0.0035	0.0135	0.0380	0.1741	0.1813	0.9638	0.9636	0.9634	0.9616
0.65	0.0037	0.0140	0.0390	0.1771	0.0037	0.0141	0.0392	0.1773	0.0037	0.0142	0.0394	0.1774	0.1851	0.9625	0.9623	0.9621	0.9601
0.66	0.0039	0.0146	0.0404	0.1804	0.0039	0.0148	0.0407	0.1806	0.0039	0.0148	0.0409	0.1807	0.1889	0.9612	0.9609	0.9607	0.9586
0.67	0.0041	0.0153	0.0419	0.1837	0.0041	0.0154	0.0422	0.1839	0.0042	0.0154	0.0425	0.1841	0.1928	0.9598	0.9595	0.9593	0.9570
0.68	0.0044	0.0159	0.0434	0.1871	0.0044	0.0161	0.0438	0.1873	0.0044	0.0161	0.0441	0.1875	0.1967	0.9584	0.9581	0.9578	0.9553
0.69	0.0046	0.0166	0.0450	0.1905	0.0046	0.0168	0.0454	0.1908	0.0046	0.0168	0.0457	0.1910	0.2007	0.9569	0.9566	0.9563	0.9536
0.70	0.0049	0.0173	0.0466	0.1940	0.0049	0.0176	0.0470	0.1943	0.0049	0.0176	0.0474	0.1945	0.2049	0.9554	0.9551	0.9547	0.9519

α	Edge Weights												Precision Adjustment			
	Internal nodes				Edge nodes nodes				Corner Nodes				Time	Internal	Edge	Corner	time
	2 Step	1 Step	0 Step	t-1	2 Step	1 Step	0 Step	t-1	2 Step	1 Step	0 Step	t-1	1	nodes	nodes	nodes	0
0.71	0.0052	0.0181	0.0483	0.1975	0.0052	0.0184	0.0487	0.1978	0.0052	0.0184	0.0492	0.1981	0.2091	0.9539	0.9535	0.9532	0.9501
0.72	0.0055	0.0189	0.0500	0.2010	0.0055	0.0192	0.0505	0.2014	0.0055	0.0192	0.0510	0.2017	0.2133	0.9523	0.9519	0.9515	0.9482
0.73	0.0058	0.0197	0.0518	0.2046	0.0058	0.0200	0.0523	0.2050	0.0058	0.0200	0.0528	0.2054	0.2177	0.9507	0.9503	0.9498	0.9462
0.74	0.0061	0.0206	0.0537	0.2083	0.0062	0.0209	0.0542	0.2087	0.0062	0.0209	0.0548	0.2091	0.2222	0.9491	0.9486	0.9481	0.9442
0.75	0.0065	0.0214	0.0555	0.2120	0.0065	0.0218	0.0562	0.2124	0.0065	0.0219	0.0568	0.2129	0.2268	0.9474	0.9468	0.9463	0.9421
0.76	0.0069	0.0224	0.0575	0.2157	0.0069	0.0228	0.0582	0.2163	0.0069	0.0228	0.0588	0.2167	0.2316	0.9456	0.9450	0.9444	0.9399
0.77	0.0073	0.0234	0.0595	0.2195	0.0073	0.0238	0.0603	0.2201	0.0073	0.0239	0.0610	0.2207	0.2364	0.9438	0.9432	0.9425	0.9376
0.78	0.0077	0.0244	0.0616	0.2234	0.0077	0.0249	0.0624	0.2241	0.0078	0.0250	0.0632	0.2247	0.2414	0.9420	0.9413	0.9406	0.9353
0.79	0.0082	0.0255	0.0637	0.2273	0.0082	0.0261	0.0646	0.2281	0.0083	0.0261	0.0655	0.2287	0.2466	0.9401	0.9393	0.9385	0.9328
0.80	0.0087	0.0266	0.0659	0.2314	0.0087	0.0273	0.0670	0.2322	0.0088	0.0273	0.0679	0.2329	0.2519	0.9381	0.9372	0.9364	0.9302
0.81	0.0092	0.0278	0.0682	0.2354	0.0093	0.0285	0.0694	0.2363	0.0093	0.0286	0.0704	0.2372	0.2574	0.9361	0.9351	0.9342	0.9275
0.82	0.0098	0.0290	0.0706	0.2396	0.0098	0.0299	0.0718	0.2406	0.0099	0.0299	0.0730	0.2415	0.2631	0.9341	0.9330	0.9320	0.9247
0.83	0.0104	0.0303	0.0730	0.2438	0.0104	0.0313	0.0744	0.2450	0.0105	0.0314	0.0757	0.2460	0.2690	0.9319	0.9307	0.9296	0.9217
0.84	0.0110	0.0317	0.0756	0.2481	0.0111	0.0328	0.0771	0.2494	0.0112	0.0329	0.0786	0.2506	0.2752	0.9297	0.9284	0.9272	0.9186
0.85	0.0117	0.0331	0.0782	0.2526	0.0118	0.0344	0.0799	0.2540	0.0119	0.0345	0.0815	0.2553	0.2817	0.9275	0.9260	0.9246	0.9153
0.86	0.0125	0.0347	0.0809	0.2571	0.0126	0.0361	0.0829	0.2587	0.0127	0.0362	0.0846	0.2602	0.2884	0.9251	0.9235	0.9220	0.9118
0.87	0.0133	0.0363	0.0838	0.2617	0.0135	0.0379	0.0859	0.2635	0.0136	0.0381	0.0879	0.2652	0.2955	0.9227	0.9209	0.9192	0.9082
0.88	0.0142	0.0380	0.0867	0.2664	0.0144	0.0398	0.0891	0.2685	0.0146	0.0401	0.0914	0.2704	0.3031	0.9202	0.9182	0.9163	0.9042
0.89	0.0152	0.0398	0.0898	0.2713	0.0154	0.0419	0.0925	0.2736	0.0156	0.0422	0.0950	0.2758	0.3110	0.9176	0.9153	0.9132	0.9000
0.90	0.0162	0.0418	0.0930	0.2762	0.0165	0.0442	0.0960	0.2789	0.0168	0.0445	0.0989	0.2814	0.3196	0.9149	0.9124	0.9100	0.8955
0.91	0.0174	0.0438	0.0963	0.2814	0.0177	0.0467	0.0998	0.2844	0.0180	0.0470	0.1030	0.2872	0.3288	0.9121	0.9093	0.9066	0.8905
0.92	0.0186	0.0461	0.0998	0.2867	0.0191	0.0494	0.1037	0.2902	0.0195	0.0498	0.1075	0.2934	0.3387	0.9092	0.9060	0.9030	0.8852
0.93	0.0201	0.0485	0.1035	0.2921	0.0206	0.0523	0.1080	0.2962	0.0211	0.0529	0.1122	0.2999	0.3496	0.9062	0.9026	0.8991	0.8792
0.94	0.0216	0.0511	0.1074	0.2978	0.0223	0.0556	0.1125	0.3025	0.0230	0.0563	0.1175	0.3069	0.3618	0.9031	0.8989	0.8949	0.8726
0.95	0.0234	0.0539	0.1114	0.3036	0.0243	0.0593	0.1174	0.3091	0.0252	0.0603	0.1232	0.3144	0.3756	0.8998	0.8949	0.8903	0.8650
0.96	0.0255	0.0570	0.1157	0.3098	0.0266	0.0636	0.1227	0.3163	0.0278	0.0648	0.1296	0.3226	0.3917	0.8963	0.8907	0.8853	0.8562
0.97	0.0279	0.0605	0.1203	0.3162	0.0294	0.0686	0.1287	0.3240	0.0310	0.0702	0.1370	0.3317	0.4113	0.8926	0.8860	0.8795	0.8455
0.98	0.0308	0.0645	0.1252	0.3230	0.0330	0.0748	0.1354	0.3327	0.0353	0.0771	0.1459	0.3424	0.4368	0.8887	0.8807	0.8727	0.8316
0.99	0.0347	0.0693	0.1307	0.3303	0.0381	0.0831	0.1437	0.3429	0.0419	0.0870	0.1576	0.3560	0.4758	0.8844	0.8743	0.8639	0.8108
1.00	0.0428	0.0772	0.1376	0.3393	0.0518	0.1021	0.1582	0.3596	0.0632	0.1132	0.1834	0.3841	0.6116	0.8791	0.8634	0.8450	0.7434

Table A.2: *The reference table for the coarse 2+1D model.*

Appendix B

Extracts from the computer code

Extracts of the 'C' code used in the MCMC simulation is given below. The first extract is for the 1+1D Gibbs sampler.

Gibbs functions (1 + 1D model)

```
#define NUMOBS 2000   /* number of observations available */

/* helper function declarations */
int index(int i,int j);
gsl_vector * add_obs(gsl_vector_int *,gsl_vector_int *, gsl_vector *);
void read_data(char *, gsl_vector_int *,gsl_vector_int *,gsl_vector *);
double update_tau_o(double,double);
double update_tau_s(double,double,double);
gsl_vector * latent_structure(double lat,double time, double prec);
double latent_p(int ,int,double ,double );
double Foperation(int , int);
double Foperation1(int , int);

/* Global Variables */
int M,T;
gsl_vector *observations, *latent;
gsl_rng *rng;

/* Gibbs algorithm */
int main(int argc,char *argv[])
{
  gsl_vector_int *t_vec,*i_vec;
  gsl_vector *data_vec;
```

```
int i,t;
double j,k,l,o,tau_s,tau_o,update,lambda_o,lambda_s;
long iters,it;

iters=atoi(argv[1]);
M=100;    /* spatial nodes */
T=20;     /* time periods */
/*declarations*/
data_vec=gsl_vector_calloc(NUMOBS);
t_vec=gsl_vector_int_calloc(NUMOBS);
i_vec=gsl_vector_int_calloc(NUMOBS);
observations=gsl_vector_calloc(NUMOBS);
latent=gsl_vector_calloc(NUMOBS);

read_data("~/code/results/generated/1+1d/bugs/gendata1+1D.dat",t_vec,i_vec,data_vec);    /* read in data */
observations=add_obs(t_vec,i_vec,data_vec);    /* put observations into structure */
gsl_vector_int_free(i_vec);
gsl_vector_int_free(t_vec);
rng=gsl_rng_alloc(gsl_rng_mt19937);
j=1.0; k=0.00001;    /* prior for tau_s */
l=1.0; o=0.00001;    /* prior for tau_o */
tau_o=1.0; tau_s=1.0; latent=latent_structure(M,T,tau_s);    /* initialisation */
printf("iters tau_s tau_o \n");
for (it=0;it<iters;it++) {    /* main loop for MCMC simulation */
  printf("%d ",it);
  tau_o=update_tau_o(j,k);    /* calculates proposed observation precision */

  for(t=0;t<T;t++) {    /* update loop for latent structure */
    for(i=0;i<M;i++) {
      update=latent_p(i,t,tau_o,tau_s);
      gsl_vector_set(latent,index(i,t),update);
    }
  }

    tau_s=update_tau_s(j,k,tau_s);    /* update state precision */
    printf("%f %f\n",tau_s,tau_o);    /* print out MCMC parameter estimates */
  }

  gsl_vector_free(latent);    /* free memory */
  gsl_vector_free(data_vec);

  return(0);
}
```

```
/* helper functions */
/* read file data in */
void read_data(char *filename, gsl_vector_int *t_vec,gsl_vector_int *i_vec, gsl_vector *data_vec)
{
/* declarations */
  long i;
  int lat,time;
  double temp;
  FILE *s;

  s=fopen(filename,"r");
  if (s == NULL) {
    perror("failed to open file");
    exit(EXIT_FAILURE);
  }
  for (i=0;i<NUMOBS;i++) {
    fscanf(s,"%d",&lat);
      fscanf(s,"%d",&time);
    fscanf(s,"%lf",&temp);
      gsl_vector_int_set(t_vec,i,time);
      gsl_vector_int_set(i_vec,i,(lat));
      gsl_vector_set(data_vec,i,temp);
    }
    fclose(s);
}

/* adds obervations to the structure */
gsl_vector * add_obs(gsl_vector_int *t_vec,gsl_vector_int *i_vec,gsl_vector *data_vec)
{
/* declation*/
  int i;

  for (i=0;i<NUMOBS;i++) {
    gsl_vector_set(observations,index(gsl_vector_int_get(i_vec,i),gsl_vector_int_get(t_vec,i)),
                   gsl_vector_get(data_vec,i));
  }

  return(observations);
}
```

```
/* calculates proposed latent node using full conditionals */
double latent_p(int i,int t,double tau_o,double tau_s)
{
/* declarations*/
  double mu,sigma,a,part1,part2,update,sum,alpha;

  alpha=0.7;
  a=(tau_s*(1+(3*alpha*alpha/8))+tau_o);
  sigma=sqrt(1.0/a);
  sum=0.0;
  if(i==0) {
    sum=alpha*alpha*0.25*(gsl_vector_get(latent,index(i+1,t))+0.25*(gsl_vector_get(latent,index(i+2,t))));
  }
  else if(i==1) {
    sum=alpha*alpha*0.25*(gsl_vector_get(latent,index(i+1,t))+gsl_vector_get(latent,index(i-1,t))+0.25
                          *(gsl_vector_get(latent,index(i+2,t))));
  }
  else if(i==M-2) {
    sum=alpha*alpha*0.25*(gsl_vector_get(latent,index(i+1,t))+gsl_vector_get(latent,index(i-1,t))+0.25
                          *(gsl_vector_get(latent,index(i-2,t))));
  }
  else if(i==M-1) {
    sum=alpha*alpha*0.25*(gsl_vector_get(latent,index(i-1,t))+0.25*(gsl_vector_get(latent,index(i-2,t))));
  }
  else {
    sum=alpha*alpha*0.25*(gsl_vector_get(latent,index(i+1,t))+gsl_vector_get(latent,index(i-1,t))+0.25
                          *(gsl_vector_get(latent,index(i+2,t))+gsl_vector_get(latent,index(i-2,t))));
  }
  part1=tau_o* gsl_vector_get(observations,index(i,t));
  if (t==0) {
    if((i==0)||(i==M-1)) {
      part2=tau_s*(Foperation(i,t+1)-sum);
      mu=(part1+part2)/((1+5*alpha*alpha/16)*tau_s+tau_o);
      update=gsl_ran_gaussian(rng,sqrt(1.0/((1+5*alpha*alpha/16)*tau_s+tau_o)))+mu;
    }
    else {
      part2=tau_s*(Foperation(i,t+1)-sum);
      mu=(part1+part2)/a;
      update=gsl_ran_gaussian(rng,sigma)+mu;
    }
  }
  else if (t==T-1) {
    part2=tau_s*Foperation(i,t-1);
```

```
  mu=(part1+part2)/(tau_o+tau_s);
  update=gsl_ran_gaussian(rng,sqrt(1.0/(tau_o+tau_s)))+mu;
}
else if((i==0)||(i==M-1)) {
  part2=tau_s*(Foperation(i,t-1)+Foperation(i,t+1)-sum);
  mu=(part1+part2)/((1+5*alpha*alpha/16)*tau_s+tau_o);
  update=gsl_ran_gaussian(rng,sqrt(1.0/((1+5*alpha*alpha/16)*tau_s+tau_o)))+mu;
  }
  else {
    part2=tau_s*(Foperation(i,t-1)+Foperation(i,t+1)-sum);
    mu=(part1+part2)/a;
    update=gsl_ran_gaussian(rng,sigma)+mu;
  }

  return(update);
}

/* calculates gpa( theta) at time t and spatial location i for use in full conditionals */
double Foperation(int i, int t)
{
/* declaration */
  double F;

  if (t<=0) {
    F=0.0;
  }
  else if(t>T-1) {
    F=0.0;
  }
  else if (i==0) {
    F=0.7*(0.5*gsl_vector_get(latent,index(i,t))+0.25*gsl_vector_get(latent,index(i+1,t)));
  }
  else if (i==M-1) {
    F=0.7*(0.5*gsl_vector_get(latent,index(i,t))+0.25*gsl_vector_get(latent,index(i-1,t)));
  }
  else {
    F=0.7*(0.5*gsl_vector_get(latent,index(i,t))+0.25*(gsl_vector_get(latent,index(i-1,t))
          +gsl_vector_get(latent,index(i+1,t))));
  }

  return(F);
}
```

```
/* calculates proposed state precision */
double update_tau_s(double a,double b,double prec)
{
/* declarations */
  double tau_s,sumsq;
  int i,t;
  gsl_vector*structure;
  structure=gsl_vector_calloc(M*T);

  for(t=0;t<T;t++) {
    for(i=0;i<M;i++) {
      gsl_vector_set(structure,index(i,t),Foperation(i,t-1));
    }
  }

  gsl_vector_sub(structure,latent);
  gsl_blas_ddot(structure,structure,&sumsq);
  tau_s=gsl_ran_gamma(rng,a+0.5*NUMOBS,1.0/(b+0.5*sumsq));
  gsl_vector_free(structure);
  sumsq=0;

    return(tau_s);
  }

  /* calculates proposed observation precision */
  double update_tau_o(double a,double b)
  {
  /* declarations */
    double tau_o, sumsq,sum;
    gsl_vector *structure;
    structure=gsl_vector_calloc(M*T);

    sum=0;
    gsl_vector_memcpy(structure,latent);
    gsl_vector_sub (structure, observations);
    gsl_blas_ddot(structure,structure,&sumsq);
    tau_o=gsl_ran_gamma(rng,a+0.5*NUMOBS,1.0/(b+0.5*sumsq));
    sumsq=0;
    gsl_vector_free(structure);

    return((tau_o));
  }
```

```
/* calculates indentity of nodes with (i,j) coordinates */
int index(int i,int j)
{
/* declaration */
  int index;
  index=j*M+i;

  return(index);
}

/* reads in the initialisations for the latent structure */
gsl_vector * latent_structure(double lat,double time, double prec)
{
/* declaration*/
  FILE * file;
  long t,i;
  gsl_vector * latent;
  latent=gsl_vector_calloc(M*T);

  file=fopen("~/code/test/datain.dat","r");
  if (file == NULL) {
    perror("failed to open file");
    exit(EXIT_FAILURE);
  }
  gsl_vector_fscanf (file, latent);
  fclose(file);

  return(latent);
}
```

The data augmentation algorithm for the 1+1D model is given below here the latent structure is updated in a block rather than each node individually. This program uses the `GDAGsim` library to build the structure.

```
#define NUMOBS 2000   /* number of observations */

/* function prototypes */
void add_obs(Spat * spat ,gsl_vector_int *t_vec, gsl_vector_int *i_vec, gsl_vector *j_vec,
             gsl_vector *temp_vec);
void read_data(char *filename, gsl_vector_int *t_vec, gsl_vector_int *i_vec, gsl_vector *j_vec,
               gsl_vector *temp_vec);
```

```
/* data auggumentation algorithm */
int main(int argc,char *argv[])
{
/* declarations*/
  Spat *spat;
  gsl_vector_int *t_vec,*i_vec;
  gsl_vector * temp_vec, * mean,*j_vec, *scratch, *temp2,*s;
  gsl_rng *rng;
  long it,iters,pt,pt2,p,i,t;
  double a,mll,mll_p,lambda_s,lambda_s_p,lambda_o,lambda_o_p,var,mu,mu_p,x,j,k,l,o,sumsq;

  if (argc != 2) {
    fprintf(stderr,"Usage: %s <iters>\n",argv[0]);
    exit(EXIT_FAILURE);
  }
  iters=atoi(argv[1]);   /* The number of MCMC iterations */

  /*declarations*/
  spat=spat_calloc(20,100,1);   /* dimensions of latent model (time, latitude, longitude) */
  temp_vec=gsl_vector_calloc(NUMOBS);
  t_vec=gsl_vector_int_calloc(NUMOBS);
  i_vec=gsl_vector_int_calloc(NUMOBS);
  j_vec=gsl_vector_calloc(NUMOBS);
  p=2000;   /* total number of latent nodes*/

  read_data("/home/n6137483/code/gendata1+1D.dat",t_vec,i_vec,j_vec,temp_vec);  /* read data in from file */
  rng=gsl_rng_alloc(gsl_rng_mt19937);
  j=1.0; k=0.00001;  /* prior for tau_s */
  l=1.0; o=0.00001;  /* prior for tau_o */
  mu=0.0; tau_s=0.2;tau_o=0.1;   /* inits */
  scratch=gsl_vector_calloc(p);
  temp2=gsl_vector_calloc(p);

  printf("Iter tau_s tau_o \n");

  /* MCMC loop*/
  for (it=0;it<iters;it++) {
    spat_set_zero(spat);
    spat_set_latent(spat,mu,0.7, tau_s  ,tau_o);
    add_obs(spat,t_vec,i_vec,j_vec,temp_vec);
    spat_process(spat);
    s=spat_sim(rng,spat);   /* update for latent nodes */
```

```
gsl_vector_memcpy(scratch,s);    /* update for obs_p */
gsl_vector_sub(scratch,temp_vec);
gsl_blas_ddot(scratch,scratch,&sumsq);
tau_o=gdag_ran_gamma(rng,j+0.5*p,k+0.5*sumsq);

gsl_vector_set_zero(scratch);
gsl_vector_memcpy(scratch,s);
sumsq=0;

/* gpa(theta) at time t and location i*/
for (t=0 ; t < 20 ; t++) {
  for (i=0 ; i < 100 ; i++) {
   if(t==0) {
     gsl_vector_set(temp2,i,0.0);
   }
   /* two corner nodes */
   else if ( (i==0)) {
     gsl_vector_set(temp2,100*t+i,0.25*gsl_vector_get(scratch,100*(t-1)+(i+1))
                    +0.5*gsl_vector_get(scratch,100*(t-1)+(i)));
   }

   else if ( (i==99)) {
     gsl_vector_set(temp2,100*t+i,0.25*gsl_vector_get(scratch,100*(t-1)+(i-1))
                    +0.5*gsl_vector_get(scratch,100*(t-1)+(i)));
   }

   /* interior nodes */
   else {
     gsl_vector_set(temp2,100*t+i,0.25*(gsl_vector_get(scratch,100*(t-1)+(i-1))
                    +gsl_vector_get(scratch,100*(t-1)+(i+1)))+0.5*gsl_vector_get(scratch,100*(t-1)+(i)));
   }
  }
}

  gsl_vector_scale(temp2,0.7);
  gsl_vector_sub(scratch,temp2);
  gsl_blas_ddot(scratch,scratch,&sumsq);
  lambda_s=gdag_ran_gamma(rng,l+0.5*p,o+0.5*sumsq);    /* update for state_p */

  printf("%ld %f %f\n",it,tau_s,tau_o);  /* Print out MCMC parameter estimates */
}
```

```
spat_free(spat);   /*free memory*/
gsl_vector_free(temp_vec);
gsl_vector_int_free(t_vec);
gsl_vector_int_free(i_vec);
gsl_vector_free(j_vec);

return(EXIT_SUCCESS);
}

/* helper functions */
/* read in the file data */
void read_data(char *filename,gsl_vector_int *t_vec,
               gsl_vector_int *i_vec,gsl_vector*j_vec,
               gsl_vector *temp_vec)
{
/* declarations */
  long i;
  int lon,lat,year;
  double temp;
  FILE *s;

s=fopen(filename,"r");
if (s == NULL) {
  perror("failed to open file");
  exit(EXIT_FAILURE);
}

for (i=0;i<NUMOBS;i++) {
  fscanf(s,"%d",&lat);
  fscanf(s,"%d",&long);
  fscanf(s,"%d",&year);
  fscanf(s,"%lf",&temp);
  gsl_vector_int_set(t_vec,i,year+0);   /* read in spatio-temporal coordinates and value of observation */
    gsl_vector_int_set(i_vec,i,(lat));
    gsl_vector_set(j_vec,i,0);
    gsl_vector_set(temp_vec,i,temp);
  }
}
```

```
/* add the observations to the model */
void add_obs(Spat * spat,gsl_vector_int *t_vec, gsl_vector_int *i_vec,gsl_vector *j_vec,
             gsl_vector *temp_vec)
{
/* declaration */
  long i;

  for (i=0;i<NUMOBS;i++) {
    spat_add_obs(spat,
                 gsl_vector_int_get(t_vec,i),
                 gsl_vector_int_get(i_vec,i),
                 gsl_vector_get(j_vec,i),
                 gsl_vector_get(temp_vec,i));
  }
}
```

The Metropolis-Hastings model is given below with the helper functions for both the fine and coarse models. This program uses the same helper functions as the data argumentation algorithm to read in the observations from a file and to add the observations to the structure.

```
#define NUMOBS 2000   /* number of observations available */

/* helper function declarations */
void read_data(char *filename, gsl_vector_int *t_vec, gsl_vector_int *i_vec, gsl_vector_int *j_vec,
               gsl_vector *temp_vec);
void add_obs(Spat *spat, gsl_vector_int *t_vec, gsl_vector_int *i_vec, gsl_vector_int *j_vec,
             gsl_vector *temp_vec);

/* Metropolis-Hastings algorithm */
int main(int argc,char *argv[])
{
/* declarations */
  Spat *spat;
  gsl_vector_int *t_vec,*i_vec,*j_vec;
  gsl_vector *temp_vec;
  gsl_rng *rng;
  long it,t,pt,pt2,iters;
  double a,mll,mll_p,lambda_s,lambda_s_p,lambda_o,lambda_o_p,mu,mu_p,j,k,l,o,alpha;
```

```
if (argc != 2) {
  fprintf(stderr,"Usage: %s <iters>\n",argv[0]);
  exit(EXIT_FAILURE);
}

iters=atoi(argv[1]);   /* number of iterations for MCMC algorithm */
spat=spat_calloc(40,10,10);   /* dimensions of latent model (time, latitude, longitude) */
  temp_vec=gsl_vector_calloc(NUMOBS);
  t_vec=gsl_vector_int_calloc(NUMOBS);
  i_vec=gsl_vector_int_calloc(NUMOBS);
  j_vec=gsl_vector_int_calloc(NUMOBS);
  read_data("datain.dat",t_vec,i_vec,j_vec,temp_vec);   /* location of data file */
  rng=gsl_rng_alloc(gsl_rng_mt19937);
  j=0.01; k=0.01;   /* prior for lambda_s */
  l=0.01; o=0.01;   /* prior for lambda_o */
  mu=0.0; lambda_s=-1.7;lambda_o=-2.0; mll=-1e05;  alpha=0.7;   /* inits */

  printf("Iter lambda_s lambda_o\n");

/* main MCMC simulation loop */
for (it=0;it<iters;it++) {
  spat_set_zero(spat);
  mu_p = mu + gsl_ran_gaussian(rng,0.1);   /* parameter updates */
  lambda_s_p = lambda_s + gsl_ran_gaussian(rng,0.1);
  lambda_o_p = lambda_o + gsl_ran_gaussian(rng,0.1);
  spat_set_latent(spat,mu_p,alpha_p,exp(lambda_s_p),exp(lambda_o_p));
  spat_prior_process(spat);
  add_obs(spat,t_vec,i_vec,j_vec,temp_vec);   /* add observations */
  spat_process(spat);
  mll_p = spat_mloglik(spat);   /* calculate marginal log likelihood for proposed updates */

  a = j * (lambda_s_p-lambda_s) - k *(exp(lambda_s_p) - exp(lambda_s))   /* acceptance probability */
    + l * (lambda_o_p-lambda_o) - o *(exp(lambda_o_p) - exp(lambda_o))
    + mll_p - mll;

  if (gdag_accept_lp(rng,a)) {   /* keep proposed values if acceptance probability is true */
    mu=mu_p; lambda_s=lambda_s_p; lambda_o=lambda_o_p; mll=mll_p;
  }

  printf("%ld %f %f\n",it,lambda_s,lambda_o);
}
```

```
spat_free(spat);                                    /* free memory */
gsl_vector_free(temp_vec);
gsl_vector_int_free(t_vec);
gsl_vector_int_free(i_vec);
gsl_vector_int_free(j_vec);

  return(EXIT_SUCCESS);
}
```

The fine model helper functions are given below.

```
/* internal function prototypes */
size_t spat_index(Spat *self,size_t t,size_t i,size_t j);

/* function definitions */
/* defines and initialises spat (spatio-temporal) structure */
Spat *spat_calloc(size_t T,size_t n,size_t m)
{
  Spat *self;
  self=malloc(sizeof(Spat));
  self->T = T;
  self->n = n;
  self->m = m;
  self->mu = 0.0;
  self->tauS = 0.0;
  self->tau0 = 0.0;
  self->gdso = gdag_calloc(n*m*T);

  return(self);
}

/* frees memory */
void spat_free(Spat *self)
{
  gdag_free(self->gdso);
  free(self);
}

/* sets spat structure to zero */
void spat_set_zero(Spat *self)
{
  self->mu = 0.0;
  self->alpha = 0.0;
  self->tauS = 0.0;
```

```
  self->tau0 = 0.0;
  gdag_set_zero(self->gdso);
}

/* calculates index for each latent node */
size_t spat_index(Spat *self,size_t t,size_t i,size_t j)
{
  return( t*(self->n)*(self->m) + i*(self->m) + j );
}

/* adds ovbservations to spat structure */
void spat_add_obs(Spat *self,size_t t,size_t i,size_t j,double obs)
{
  gdag_usv *sp=gdag_usv_basis(spat_index(self,t,i,j));
  gdag_add_observation(self->gdso,sp,0.0,self->tau0,obs);
  gdag_usv_free(sp);
}

/* processes structure so maginal log likelihood and other functions can be used */
void spat_process(Spat *self)
{
  gdag_process(self->gdso);
}
void spat_prior_process(Spat *self)
{
  gdag_prior_process(self->gdso);
}

/* calculates marginal log likelihood for spat structure */
double spat_mloglik(Spat *self)
{
  return(gdag_mloglik(self->gdso));
}

/* defines latent structure */
void spat_set_latent(Spat *self,double mu,double alpha,double tauS,double tau0)
{
  FILE *file;
  size_t t,i,j;
  gdag_usv *sv;
  gsl_vector * edge_weight, *adj_weights;
  double b,tmp;
  int it,l;
```

```
b=(1.0-alpha)*mu;
if((file = fopen("fine_edgeweights.dat","r")) == NULL) {
  printf("cannot open edgeweight.dat \n");
}
adj_weights=gsl_vector_calloc(800);   /* full reference table for edge weights */
gsl_vector_fscanf(file,adj_weights);
fclose(file);
edge_weight=gsl_vector_calloc(8);
tmp=100.0*alpha;
l=tmp;

for(it=0;it<8;it++) {   /*  yields edge weights for given alpha */
  gsl_vector_set(edge_weight,it,gsl_vector_get(adj_weights,(8*(l-1)+it)));
}
gsl_vector_free(adj_weights);

self->mu = mu;   /* record parameters in the object for future use */
self->alpha = alpha;
self->tauS = tauS;
self->tau0 = tau0;

for (i=0;i < self->n;i++) {   /* time zero layer */
  for (j=0;j < self->m;j++) {
    gdag_add_root(self->gdso,spat_index(self,0,i,j),mu,tauS);
  }
}

for (t=1;t < self->T;t++) {   /* all other times */
  for (i=0;i < self->n;i++) {
    for (j=0;j < self->m;j++) {
      if ( (i==0)&&(j==0) ) {   /* four corners */
        sv=gdag_usv_alloc(4);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,5));
  gdag_usv_add(sv,spat_index(self,t-1,i,j+1),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j+1),gsl_vector_get(edge_weight,3));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(i-(gsl_vector_get(edge_weight,5)
                +2*gsl_vector_get(edge_weight,4)+gsl_vector_get(edge_weight,3)))*mu,
                gsl_vector_get(edge_weight,7)*tauS);
  gdag_usv_free(sv);
}
```

```
else if ( (i==0)&&(j==(self->m)-1) ) {
  sv=gdag_usv_alloc(4);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,5));
  gdag_usv_add(sv,spat_index(self,t-1,i,j-1),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j-1),gsl_vector_get(edge_weight,3));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edge_weight,5)
                +2*gsl_vector_get(edge_weight,4)+gsl_vector_get(edge_weight,3)))*mu,
                gsl_vector_get(edge_weight,7)*tauS);
  gdag_usv_free(sv);
}
else if ( (i==(self->n)-1)&&(j==0) ) {
  sv=gdag_usv_alloc(4);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,5));
  gdag_usv_add(sv,spat_index(self,t-1,i,j+1),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j+1),gsl_vector_get(edge_weight,3));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edge_weight,5)
                +2*gsl_vector_get(edge_weight,4)+gsl_vector_get(edge_weight,3)))*mu,
                gsl_vector_get(edge_weight,7)*tauS);
  gdag_usv_free(sv);
}
else if ( (i==(self->n)-1)&&(j==(self->m)-1) ) {
  sv=gdag_usv_alloc(4);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,5));
  gdag_usv_add(sv,spat_index(self,t-1,i,j-1),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j),gsl_vector_get(edge_weight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j-1),gsl_vector_get(edge_weight,3));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edge_weight,5)
                +2*gsl_vector_get(edge_weight,4)+gsl_vector_get(edge_weight,3)))*mu,
                gsl_vector_get(edge_weight,7)*tauS);
  gdag_usv_free(sv);
}
```

```
else if (i==0) {   /* four edges */
 sv=gdag_usv_alloc(6);
 gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,2));
 gdag_usv_add(sv,spat_index(self,t-1,i+1,j),gsl_vector_get(edge_weight,1));
 gdag_usv_add(sv,spat_index(self,t-1,i,j+1),gsl_vector_get(edge_weight,1));
 gdag_usv_add(sv,spat_index(self,t-1,i,j-1),gsl_vector_get(edge_weight,1));
 gdag_usv_add(sv,spat_index(self,t-1,i+1,j+1),gsl_vector_get(edge_weight,0));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j-1),gsl_vector_get(edge_weight,0));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edge_weight,2)
                +3*gsl_vector_get(edge_weight,1)+2*gsl_vector_get(edge_weight,0)))*mu,
                gsl_vector_get(edge_weight,6)*tauS);
  gdag_usv_free(sv);
}
else if (j==0) {
  sv=gdag_usv_alloc(6);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,2));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i,j+1),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j+1),gsl_vector_get(edge_weight,0));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j+1),gsl_vector_get(edge_weight,0));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edge_weight,2)
                +3*gsl_vector_get(edge_weight,1)+2*gsl_vector_get(edge_weight,0)))*mu,
                gsl_vector_get(edge_weight,6)*tauS);
  gdag_usv_free(sv);
}
else if (i==(self->n)-1) {
  sv=gdag_usv_alloc(6);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,2));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i,j+1),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i,j-1),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j+1),gsl_vector_get(edge_weight,0));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j-1),gsl_vector_get(edge_weight,0));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edge_weight,2)
                +3*gsl_vector_get(edge_weight,1)+2*gsl_vector_get(edge_weight,0)))*mu,
                gsl_vector_get(edge_weight,6)*tauS);
  gdag_usv_free(sv);
}
```

```
else if (j==(self->m)-1) {
  sv=gdag_usv_alloc(6);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edge_weight,2));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i,j-1),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j),gsl_vector_get(edge_weight,1));
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j-1),gsl_vector_get(edge_weight,0));
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j-1),gsl_vector_get(edge_weight,0));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edge_weight,2)
                +3*gsl_vector_get(edge_weight,1)+2*gsl_vector_get(edge_weight,0)))*mu,
                gsl_vector_get(edge_weight,6)*tauS);
  gdag_usv_free(sv);
}

else {   /* interior */
  sv=gdag_usv_alloc(9);
  gdag_usv_add(sv,spat_index(self,t-1,i,j),alpha*0.25);
  gdag_usv_add(sv,spat_index(self,t-1,i+1,j),alpha*0.125);
  gdag_usv_add(sv,spat_index(self,t-1,i-1,j),alpha*0.125);
  gdag_usv_add(sv,spat_index(self,t-1,i,j+1),alpha*0.125);
          gdag_usv_add(sv,spat_index(self,t-1,i,j-1),alpha*0.125);
          gdag_usv_add(sv,spat_index(self,t-1,i+1,j+1),alpha*0.0625);
          gdag_usv_add(sv,spat_index(self,t-1,i+1,j-1),alpha*0.0625);
          gdag_usv_add(sv,spat_index(self,t-1,i-1,j+1),alpha*0.0625);
          gdag_usv_add(sv,spat_index(self,t-1,i-1,j-1),alpha*0.0625);
          gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,b,tauS);
          gdag_usv_free(sv);
        }
      }
    }
  }
  gsl_vector_free(edge_weight);
}
```

The coarse model uses the same main function as the fine model however some of the helper functions need to be redefined for this model. The coarse model helper functions are given below.

```
/* function definitions */
/* defines and initialises spat structure */
Spat *spat_calloc(size_t T,size_t n,size_t m)
{
  Spat *self;
  self=malloc(sizeof(Spat));
  self->T = T;
  self->n = n/2;
  self->m = m/2;
  self->mu = 0.0;
  self->tauS = 0.0;
  self->tau0 = 0.0;
  self->gdso = gdag_calloc(n*m*T/4);
  return(self);
}

void spat_set_latent(Spat *self,double mu,double alpha,double tauS,double tau0)
{
  gsl_vector *edgweight, *adj_weights;
  FILE  *file;
  size_t t,i,j;
  gdag_usv *sv;
  double b,tmp;
  int l,it;

  t=0;
  b=(1.0-alpha)*mu;
  self->mu = mu;   /* record parameters in the object for future use */
self->alpha = alpha;
self->tauS = tauS;
self->tau0 = tau0;
if((file = fopen("~/code/test/testweight.dat","r")) == NULL) {
  printf("cannot open edges3dr.dat \n");
}

adj_weights=gsl_vector_calloc(1700);   /* full reference table for course model */
gsl_vector_fscanf(file,adj_weights);
fclose(file);
```

```
edgweight=gsl_vector_calloc(17);
tmp=100.0*alpha;
l=tmp;

for(it=0;it<17;it++) {   /* edge weight for given alpha */
  gsl_vector_set(edgweight,it,gsl_vector_get(adj_weights,(17*(l-1)+it)));
}
gsl_vector_free(adj_weights);

for (i=0;i < self->n;i++) {   /* time zero layer */
  for (j=0;j < self->m;j++) {
    gdag_add_root(self->gdso,spat_index(self,t,i,j),mu,tauS);
  }
}

for (i=0;i < self->n;i++) {   /* time one layer */
  for (j=0;j < self->m;j++) {
    sv=gdag_usv_alloc(1);
    gdag_usv_add(sv,spat_index(self,0,i,j),gsl_vector_get(edgweight,12));
    gdag_add_node(self->gdso,spat_index(self,1,i,j),sv,(1-gsl_vector_get(edgweight,12))*mu,tauS
                  *gsl_vector_get(edgweight,16));
    gdag_usv_free(sv);
  }
}

for (t=2;t < self->T;t++) {   /* all other times */
  for (i=0;i < self->n;i++) {
    for (j=0;j < self->m;j++) {
      if ( (i==0)&&(j==0) ) {   /* four corners */
        sv=gdag_usv_alloc(5);
        gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,10));
        gdag_usv_add(sv,spat_index(self,t-2,i,j+1),gsl_vector_get(edgweight,9));
        gdag_usv_add(sv,spat_index(self,t-2,i+1,j),gsl_vector_get(edgweight,9));
        gdag_usv_add(sv,spat_index(self,t-2,i+1,j+1),gsl_vector_get(edgweight,8));
        gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,11));
        gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,10)
                    +2*gsl_vector_get(edgweight,9)+gsl_vector_get(edgweight,8)+gsl_vector_get(edgweight,11)))
                    *mu,tauS*gsl_vector_get(edgweight,15));
        gdag_usv_free(sv);
      }
```

```
else if ( (i==0)&&(j==(self->m)-1) ) {
  sv=gdag_usv_alloc(5);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,10));
  gdag_usv_add(sv,spat_index(self,t-2,i,j-1),gsl_vector_get(edgweight,9));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j),gsl_vector_get(edgweight,9));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j-1),gsl_vector_get(edgweight,8));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,11));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,10)
                +2*gsl_vector_get(edgweight,9)+gsl_vector_get(edgweight,8)+gsl_vector_get(edgweight,11)))
                *mu,tauS*gsl_vector_get(edgweight,15));
  gdag_usv_free(sv);
}
else if ( (i==(self->n)-1)&&(j==0) ) {
  sv=gdag_usv_alloc(5);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,10));
  gdag_usv_add(sv,spat_index(self,t-2,i,j+1),gsl_vector_get(edgweight,9));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j),gsl_vector_get(edgweight,9));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j+1),gsl_vector_get(edgweight,8));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,11));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,10)
                +2*gsl_vector_get(edgweight,9)+gsl_vector_get(edgweight,8)+gsl_vector_get(edgweight,11)))
                *mu,tauS*gsl_vector_get(edgweight,15));
  gdag_usv_free(sv);
}
else if ( (i==(self->n)-1)&&(j==(self->m)-1) ) {
  sv=gdag_usv_alloc(5);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,10));
  gdag_usv_add(sv,spat_index(self,t-2,i,j-1),gsl_vector_get(edgweight,9));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j),gsl_vector_get(edgweight,9));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j-1),gsl_vector_get(edgweight,8));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,11));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,10)
                +2*gsl_vector_get(edgweight,9)+gsl_vector_get(edgweight,8)+gsl_vector_get(edgweight,11)))
                *mu,tauS*gsl_vector_get(edgweight,15));
  gdag_usv_free(sv);
}
```

```
else if (i==0) {   /* four edges */
  sv=gdag_usv_alloc(7);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,6));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i,j+1),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i,j-1),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j+1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j-1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,7));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,6)
                +3*gsl_vector_get(edgweight,5)+2*gsl_vector_get(edgweight,4)+gsl_vector_get(edgweight,7)))
                *mu,tauS*gsl_vector_get(edgweight,14));
  gdag_usv_free(sv);
}
else if (j==0) {
  sv=gdag_usv_alloc(7);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,6));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i,j+1),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j+1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j+1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,7));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,6)
                +3*gsl_vector_get(edgweight,5)+2*gsl_vector_get(edgweight,4)+gsl_vector_get(edgweight,7)))
                *mu,tauS*gsl_vector_get(edgweight,14));
  gdag_usv_free(sv);
}
else if (i==(self->n)-1) {
  sv=gdag_usv_alloc(7);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,6));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i,j+1),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i,j-1),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j+1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j-1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,7));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,6)
                +3*gsl_vector_get(edgweight,5)+2*gsl_vector_get(edgweight,4)+gsl_vector_get(edgweight,7)))
                *mu,tauS*gsl_vector_get(edgweight,14));
  gdag_usv_free(sv);
}
```

```
else if (j==(self->m)-1) {
  sv=gdag_usv_alloc(7);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,6));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i,j-1),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j),gsl_vector_get(edgweight,5));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j-1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j-1),gsl_vector_get(edgweight,4));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,7));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,6)
              +3*gsl_vector_get(edgweight,5)+2*gsl_vector_get(edgweight,4)+gsl_vector_get(edgweight,7)))
              *mu,tauS*gsl_vector_get(edgweight,14));
  gdag_usv_free(sv);
}

else {   /* interior */
  sv=gdag_usv_alloc(10);
  gdag_usv_add(sv,spat_index(self,t-2,i,j),gsl_vector_get(edgweight,2 ));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j),gsl_vector_get(edgweight,1 ));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j),gsl_vector_get(edgweight,1 ));
  gdag_usv_add(sv,spat_index(self,t-2,i,j+1),gsl_vector_get(edgweight,1 ));
  gdag_usv_add(sv,spat_index(self,t-2,i,j-1),gsl_vector_get(edgweight,1 ));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j+1),gsl_vector_get(edgweight,0 ));
  gdag_usv_add(sv,spat_index(self,t-2,i+1,j-1),gsl_vector_get(edgweight,0 ));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j+1),gsl_vector_get(edgweight,0 ));
  gdag_usv_add(sv,spat_index(self,t-2,i-1,j-1),gsl_vector_get(edgweight,0 ));
  gdag_usv_add(sv,spat_index(self,t-1,i,j),gsl_vector_get(edgweight,3 ));
  gdag_add_node(self->gdso,spat_index(self,t,i,j),sv,(1-(gsl_vector_get(edgweight,2)
              +4*gsl_vector_get(edgweight,1)+4*gsl_vector_get(edgweight,0)+gsl_vector_get(edgweight,3)))
              *mu,tauS*gsl_vector_get(edgweight,13));
  gdag_usv_free(sv);
 }
}
 }
 }

 gsl_vector_free(edgweight);
}
```