Efficient Bayesian Computation with Applications in Neuroscience and Meteorology

by

Meixi Chen

A thesis
presented to the University of Waterloo
in fulfilment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Statistics

Waterloo, Ontario, Canada, 2024

© Meixi Chen 2024

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner:	Babak Shahbaba Professor, Department of Statistics, University of California, Irvine
Supervisor(s):	Martin Lysy Associate Professor, Department of Statistics and Actuarial Science, University of Waterloo
	Reza Ramezan Associate Professor, Teaching Stream, Department of Statistics and Actuarial Science, University of Waterloo
Internal Member:	Alex Stringer Assistant Professor, Department of Statistics and Actuarial Science, University of Waterloo
	Paul Marriott Professor, Department of Statistics and Actuarial Science, University of Waterloo
Internal-External Member:	Britt Anderson Associate Professor, Department of Psychology, University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

I am the sole author of Chapters 1 and 5.

I am the first author of the peer-reviewed article and software package associated with Chapter 2:

- Chen, M., Ramezan, R. and Lysy, M. (2024+). Fast and Scalable Approximate Inference for Spatial Extreme Value Models. *Canadian Journal of Statistics* (to appear).
- Chen, M., Lysy, M. and Ramezan, R. (2024). SpatialGEV: An efficient computational method for fitting spatial extreme value models. https://github.com/meixichen/SpatialGEV.

Chapter 3 has been prepared for submission to a peer-reviewed journal. I am the first author of the journal article, the associated software, and its preliminary version published as a conference proceeding:

- Chen, M., Lysy, M, Moorman, D. and Ramezan, R. (2023). Functional Connectivity: Continuous-Time Latent Factor Models for Neural Spike Trains. *Proceedings of 2023 Conference on Cognitive Computational Neuroscience.*
- Chen, M., Lysy, M. and Ramezan, R. (2024). fastr: Factor Analysis of Spike Trains in R. https://github.com/meixichen/fastr.

The experimental data in Chapter 3 is provided by Dr. Beata Kaminska and Dr. David Moorman from the University of Massachusetts Amherst.

Chapter 4 is a joint work with Yun Ling, Martin Lysy and Reza Ramezan. I have contributed to the superfast solver in R, the Gibbs sampler for Gaussian Process Factor Analysis, the extended Gaussian Process Factor Analysis for Poisson observations, and the simulation studies.

Abstract

Hierarchical models are important tools for analyzing data in many disciplines. Efficiency and scalability in model inference have become increasingly important areas of research due to the rapid growth of data. Traditionally, parameters in a hierarchical model are estimated by deriving closed-form estimates or Monte Carlo sampling. Since the former approach is only possible for simpler models with conjugate priors, the latter, Markov Chain Monte Carlo (MCMC) methods in particular, has become the standard approach for inference without a closed form. However, MCMC requires substantial computational resources when sampling from hierarchical models with complex structures, highlighting the need for more computationally efficient inference methods. In this thesis, we study the design of Bayesian inference to improve computational efficiency, with a focus on a class of hierarchical models known as *latent Gaussian models*. The background of hierarchical modelling and Bayesian inference is introduced in Chapter 1.

In Chapter 2, we present a fast and scalable approximate inference method for a widely used model in meteorological data analysis. The model features a likelihood layer of the generalized extreme value (GEV) distribution and a latent layer integrating spatial information via Gaussian process (GP) priors on the GEV parameters, hence the name GEV-GP model. The computational bottleneck is caused by the high number of spatial locations being studied, which corresponds to the dimensionality of the GPs. We presented an inference procedure based on the Laplace approximation to the likelihood followed by a Normal approximation to the posterior of interest. By combining the above approach with a sparsity-inducing spatial covariance approximation technique, we demonstrate through simulations that it accurately estimates the Bayesian predictive distribution of extreme weather events, scales to several thousand spatial locations, and is several orders of magnitude faster than MCMC. We also present a case study in forecasting extreme snowfall across Canada.

Building on the approximate inference scheme discussed in Chapter 2, Chapter 3 introduces a new modelling framework for capturing the correlation structure in high-dimensional neuronal data, known as *spike trains*. We propose a novel continuous-time multi-neuron latent factor model based on the biological mechanism of spike generation, where the underlying neuronal activities are represented by a multivariate Markov process. To the best of our knowledge, this is the first multivariate spike-train model in a continuous-time setting to study interactions between neural spike trains. A computationally tractable Bayesian inference procedure is proposed to address the challenges in estimating high-dimensional latent parameters. We show that the proposed model and inference method can accurately recover underlying neuronal interactions when applied to

a variety of simulated datasets. Application of our model on experimental data reveals that the correlation structure of spike trains in rats' orbitofrontal cortex predicts outcomes following different cues.

While Chapter 3 restricts modelling to Markov processes for the latent dynamics of spike trains, Chapter 4 presents an efficient inference method for non-Markov stationary GPs with noisy observations. While computations for such models typically scale as $\mathcal{O}(n^3)$ in the number of observations n, our method utilizes a "superfast" Toeplitz system solver which reduces computational complexity to $\mathcal{O}(n \log^2 n)$. We demonstrate that our method dramatically improves the scalability of Gaussian Process Factor Analysis (GPFA), which is commonly used for extracting low-dimensional representation for high-dimensional neural data. We extend GPFA to accommodate Poisson count observations and design a superfast MCMC inference algorithm for the extended GPFA. The accuracy and speed of our inference algorithms are illustrated through simulation studies.

Acknowledgements

First and foremost, I am most grateful to my supervisors Dr. Martin Lysy and Dr. Reza Ramezan for their guidance and support throughout my graduate study. I have learned so much from them, both in research and in life. It has really been an honour.

I would like to offer my gratitude to my committee members, Dr. Britt Anderson, Dr. Paul Marriott, Dr. Babak Shahbaba, and Dr. Alex Stringer, for their insight on this thesis. In particular, I would like to thank Alex for the helpful discussion on Chapter 2 of this thesis.

I appreciate all the help I received from the staff in the department, especially Mary Lou.

My time at UW would not have been this enjoyable without my friends and comrades in M3. I would like to thank Angela for the longest and emotional phone calls, Augustine for being my first friend at UW, Connor for being my Covid climbing buddy, Frieda (and Matt) for the most wholesome time, Jiayue, Yuanyuan and Zhanyi for the good hikes followed by spicy food, Jie and Jingyue for being the best office mates, Yingxi and Yue for the long walks and breakfast dates. Special thanks to the morning climbing crew (Connie, Jim, Laura, Peter, Sigrid, et al.) at Grand River Rocks Kitchener for making the most fun memories on the walls.

The conditional expectation of my life quality given Michael is far greater than the marginal version of it. Your love and support mean everything to me.

The person I would like to thank the most is my mother, the strongest and coolest person in the world, without whom I would not have been where I am now.

Dedication

This thesis is dedicated to my mother.

Table of Contents

Examining Committee Membership ii

Author's Declaration iii

Statement of Contributions iv

Abstract v

Acknowledgements vii

Dedication viii

List of Figures xiii

List of Tables xix

1 Introduction 1

1.1 Background 4

1.1.1 Scalable Covariance Matrix Computation 4

1.1.2 Approximate Bayesian Inference 6

2 Fast and Scalable Inference for Spatial Extreme Value Models 8

2.1 Introduction 8

2.2 Methodology . . . 10
2.2.1 Scalable Likelihood Evaluations . . . 12
2.2.2 Estimation of Hyperparameters . . . 13
2.2.3 Estimation of Random Effects . . . 16
2.2.4 Estimation of Return Levels . . . 17
2.2.5 Implementation . . . 18
2.3 Simulation Study . . . 18
2.3.1 Small-Scale Study on Smooth Surfaces . . . 20
2.3.2 Large-Scale Study on Rough Surfaces . . . 28
2.4 Case Study . . . 30
2.4.1 Model Selection . . . 32
2.4.2 Parameter Inference . . . 33
2.5 Discussion . . . 36

3 Continuous-Time Multi-Neuron Latent Factor Model for Spike Trains 39
3.1 Introduction . . . 39
3.2 Model . . . 42
3.2.1 Single-Neuron Model . . . 44
3.2.2 Multi-Neuron Factor Model for Neuron Dependence . . . 45
3.2.3 Parameter Identifiability . . . 46
3.3 Inference . . . 46
3.3.1 Time Discretization for Computations . . . 46
3.3.2 Parameter Priors . . . 48
3.3.3 Integration using Laplace Approximation . . . 48
3.3.4 Two-Step Estimation with Fixed Marginal Parameters . . . 50
3.3.5 Additional Computational Considerations . . . 51
3.3.6 Determining of the Number of Factors . . . 52
3.4 Model Checking . . . 53

3.4.1 Leave-Neuron-Out Posterior Predictive Check . . . 54
3.4.2 Diagnostic Statistic . . . 55
3.5 Simulation Studies . . . 56
3.5.1 Simulated Data from the Proposed Model . . . 58
3.5.2 Data Simulated from NEURON Software . . . 64
3.6 Neuronal Functional Connectivity in Rat OFC . . . 71
3.6.1 Data Collection . . . 71
3.6.2 Metrics for Comparing Neuronal Activities . . . 72
3.6.3 Leave-Neuron-Out GOF Assessment . . . 76
3.6.4 Results . . . 78
3.7 Discussion . . . 79
3.7.1 Future Work . . . 82

4 Superfast Inference for Gaussian Process Factor Analysis **85**
4.1 Introduction . . . 85
4.2 Methods . . . 87
4.2.1 Superfast Toeplitz System Solver . . . 87
4.2.2 Gaussian Process Factor Analysis . . . 88
4.2.3 Superfast GPFA Inference Algorithm . . . 90
4.2.4 Extended GPFA for Poisson Observations . . . 92
4.3 Simulation Studies . . . 93
4.3.1 Gaussian-GPFA Model . . . 94
4.3.2 Poisson-GPFA Model . . . 95
4.4 Discussion . . . 96

5 Conclusion **101**

References **103**

APPENDICES **124**

A Appendix for Chapter 2 **125**

A.1 Estimation of Functions of Random Effects and Hyperparameters 125

A.2 Estimation Errors for the Random Effects in the Small-Scale Simulation Study 126

A.3 Bayesian Posterior Prediction at Arbitrary Locations 127

B Appendix for Chapter 3 **129**

B.1 Comparison Between Two-Step Estimation and Joint Estimation 129

B.2 Leave-Neuron-Out Posterior Predictive Test 132

B.2.1 Example of LNO-PP Diagnostic for Unsatisfactory Model fit 132

B.2.2 Model Comparison using LNO-PP Test 133

B.3 Exploratory Factor Analysis for Choosing the Number of Factors 134

B.3.1 Analysis of Data Simulated From NEURON 137

B.3.2 Analysis of Real Data of Rats 138

B.4 Sensitivity Analysis and Runtimes for Different Numbers of Factors 138

B.5 Summary Statistics for Comparing Neuronal Interactions Before and After Cue 138

C Appendix for Chapter 4 **142**

C.1 Derivation of the Latent States Update for the GPFA Gibbs Sampler . . . 142

C.2 Metropolis-Within-Gibbs Sampler for the Extended GPFA Model with Poisson Likelihood 143

List of Figures

2.1 The true $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$ and $s(\boldsymbol{x}_i)$ for the small-scale simulation study plotted on regular lattices. 20

2.2 Posterior distributions of the hyperparameters $\boldsymbol{\theta}$ in the small-scale simulation study. 22

2.3 True GEV location parameters $a(\boldsymbol{x})$, scale parameters $b(\boldsymbol{x})$, shape parameters $s(\boldsymbol{x})$, and 10-year return levels $z_{10}(\boldsymbol{x})$ plotted against their estimates obtained using MCMC, Max-and-Smooth, and Laplace. 24

2.4 Upper panels: Standard deviations of posterior samples of $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$, $s(\boldsymbol{x}_i)$ and $z_{10}(\boldsymbol{x})$ from Laplace-MQ plotted on the regular lattice. Lower panels: Ratios between the posterior standard deviations obtained from Laplace-MQ and those from HMC-NUTS. 26

2.5 The true $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$ and $s(\boldsymbol{x}_i)$ for the large-scale simulation study simulated from three Gaussian random fields. 28

2.6 (a)-(d): True GEV location parameters $a(\boldsymbol{x})$, scale parameters $b(\boldsymbol{x})$, shape parameters $s(\boldsymbol{x})$, and 10-year return levels $z_{10}(\boldsymbol{x})$ plotted against their estimates obtained using the Laplace method when the true parameter surfaces are unsmooth. (e)-(f): Q-Q plots for the Z-scores corresponding to $a(\boldsymbol{x})$, $b(\boldsymbol{x})$, $s(\boldsymbol{x})$, and $z_{10}(\boldsymbol{x})$. 29

2.7 (a): All 3833 locations at which data was collected from 1987 to 2021. (b): 509 locations at which at least 10 years of data was recorded after gridding. 31

2.8 PDF (left) and CDF (right) of all $n = 509$ extreme value observations pooled across locations and time. 32

2.9 Posterior predicted versus empirical test statistics. Each dot corresponds to one of the $n = 509$ spatial locations. Upper panel: Model with both $a(\boldsymbol{x})$ and $b(\boldsymbol{x})$ as spatial random effects. Lower panel: Model with only $a(\boldsymbol{x})$ random. Left panel: Predicted medians versus sample medians. Right panel: Predictive upper 10% quantiles versus sample upper 10% quantiles. 34

2.10 Posterior means and standard deviations of $a(\boldsymbol{x})$, $b(\boldsymbol{x})$, and $z_{10}(\boldsymbol{x})$. 35

3.1 (a) Evolution of the membrane potential of a neuron during a spike (adapted from p31 of Kalat, 2018). (b) The raster plot of spike trains with each dot indicating the occurrence of a spike at the corresponding time. 40

3.2 Illustration of the spike-generating mechanism for a single neuron. 43

3.3 LNO-PP graphical diagnostic using the cumulative spike count as the statistic of interest for a held-out neuron in a 12-neuron simulated dataset, with each grey line representing one sample from the LNO-PP check algorithm. 56

3.4 LNO-PP graphical diagnostic using spike time as a statistic for a held-out neuron in a 12-neuron simulated dataset, with each histogram representing samples of the n-th spike time from the LNO-PP distribution and the vertical dashed line corresponding to the observed n-th spike time. 57

3.5 Rasterplot of 20-neuron spike trains simulated from the proposed model. . 59

3.6 Model checking results using the LNO-PP check. Close agreement is observed between the predicted and the observed spike counts as the data is simulated from the model. Each corresponds to 100 draws from the posterior predictive distribution, leaving neurons 4, 8, 12, and 16 out for prediction, respectively. 59

3.7 (a): Estimated firing rates $\widehat{\alpha_i/k_i}$ plotted against their true values. Each dot corresponds to one neuron. The vertical error bar corresponds to the 95% confidence interval. (b): Box plots of the errors of the covariance matrix $\widehat{\boldsymbol{\Sigma}} = \widehat{\boldsymbol{\Lambda}}\widehat{\boldsymbol{\Lambda}}^T + \widehat{\boldsymbol{\Psi}}$ for models with different number of factors. (c): True factor loading matrix $\boldsymbol{\Lambda}$. (d): Varimax-transformed estimated factor loading matrix $\widehat{\boldsymbol{\Lambda}}_{\text{Varimax}}$ when $d = d^* = 4$. (e): Varimax-transformed estimated factor loading matrix when $d = 2 < d^*$. (f): Varimax-transformed estimated factor loading matrix when $d = 6 > d^*$. For (c)-(f), the columns correspond to the factors and the rows correspond to the neurons. 61

3.8 (a) Simulated spike train convolved with an exponentially decaying kernel. (b) Cross-correlation between pairs of convolved spike trains at lag 0 accompanied by the dendrogram based on hierarchical clustering. The non-significant correlation coefficients are set to zero. (c) Cross-correlation based on loading matrix estimates from our model. 62

3.9 Total variance explained by the factors as a function of the number of factors (d) chosen. 63

3.10 (a) Blue rectangle: a group of 125 Poisson spike generators. Green circle: a cluster of 1000 excitatory neurons. Orange circle: all 1000 inhibitory neurons. Blue arrow: stimuli from a group of Poisson spike generators to a cluster of excitatory neurons. Green arrow: excitatory inputs from a cluster of excitatory neurons to the inhibitory neurons. Orange arrow: inhibitory inputs from inhibitory neurons to a cluster of excitatory neurons. (b) Raster plot of the 46 sampled neurons simultaneously recorded for 10s. The red horizontal dashed lines divide the neuron clusters. The neurons are ordered by the true cluster membership. Each horizontal dotted line corresponds to the spike train of one neuron. The top rows with the most spikes correspond the cluster of inhibitory neurons, while the rest of the four clusters are all excitatory neurons. 65

3.11 Correlation matrix plots and dendrograms obtained from (a) convolved data and (b) loading matrix estimates from the model, using data simulated from NEURON. 67

3.12 (a) Estimated factor loading matrix $\widehat{\mathbf{\Lambda}}$ with rows ordered by hierarchical clustering. (b) Estimated $\widehat{\mathbf{\Lambda}}$ with rows ordered by the true underlying clustering shown in Figure 3.10a, with the true clusters divided by the dashed lines. The columns correspond to the factors and the rows correspond to the neurons. Varimax rotation has been applied for both (a) and (b). . . . 68

3.13 Visualization of the cluster membership matrices obtained using different methods. The (i, j)-th entry of the matrix is 1 (colour-coded by cluster membership) if neuron i and j are assigned to the same cluster, and 0 (white) otherwise. (b) and (e): Cluster memberships obtained by applying K-means on $\widehat{\mathbf{\Lambda}}$ estimated using our model. (c) and (f): Cluster memberships given by direct K-means applied on convolved spike trains. (d): Cluster memberships given by the Humphries (2011) spike-train community detection algorithm. (g)-(l): Cluster memberships obtained by applying K-means on $\widehat{\mathbf{\Lambda}}$ estimated using the GPFA with different bin sizes. 70

3.14 Explanation of the experiment and the data. 73

3.15 Examples of estimated quantities from one session consisting of 22 neurons for rat 1. Neurons are classified into projection (PR) neurons and wide high (WH) neurons based on the waveform of their raw signals prior to spike sorting. (a)-(c): Estimated firing rates in the baseline and following ethanol, sucrose and quinine cues, respectively. The error bars represent the posterior standard deviations. (d)-(f): Estimated correlation matrices in two baseline windows and post ethanol cue. Correlation coefficients that are not significantly different from zero are set to zero in the plots. 75

3.16 LNO-PP graphical diagnostic results based on cumulative spike counts for two different held-out neurons (PR1 and PR5) of animal 1 in two time windows (baseline and post-cue). Each grey line represents one sample from the LNO-PP check algorithm and the red line corresponds to the average of all grey lines. 77

3.17 Plots of the summary statistic Δ for comparing FC between neurons in different cue conditions. (a) compares the change of FC in baselines and that after the alcohol cue. Each red dot represent a value of $\Delta^{\text{bl}}_{\text{alcohol,a,s,i}}$ for the corresponding animal and session in the alcohol cue condition. Each vertical streak of red dots come from one session. The blue triangles represent $\Delta^{\text{obs}}_{\text{alcohol,a,s}}$ for the corresponding animal and session. The grey dashed line corresponds to zero. (b) pools the baseline values $\Delta^{\text{bl}}_{c,a,s,i}$ from all sessions of an animal, represented by the red dots. The observed statistics $\Delta^{\text{obs}}_{c,a,s}$ for each cue condition are also pooled from all sessions, with green, blue and purple dots representing alcohol, sucrose and quinine cues, respectively. . . 80

3.18 Plots of the summary statistic Δ for comparing FC between neurons. (a) and (b) compare the change of FC in baselines and that after the sucrose and quinine cues, repectively. Each red dot represent a value of $\Delta^{\text{bl}}_{c,a,s,i}$. Each vertical streak of red dots comes from one session. The blue triangles represent $\Delta^{\text{obs}}_{c,a,s}$ for the corresponding animal and session. 81

4.1 Posterior estimates for the Gaussian-GPFA model. (a) Estimated posterior distribution densities of θ^w_k, $k = 1, 2, 3$. The red dashed lines correspond to the respective true values used for simulating data. (b) and (c) Estimated posterior mean versus truth of α_d, σ^2_d and $\beta_{k,d}$, $d = 1, \ldots, 12$, $k = 1, 2, 3$. . . 95

4.2 Posterior mean of the latent factor processes $\boldsymbol{x}_{:,k}, k = 1, 2, 3$ plotted with the true factor processes for the Gaussian-GPFA model. 96

4.3 Estimated posterior distributions for the Poisson-GPFA model parameters. Posterior distributions are shown for half of the elements of $\boldsymbol{\beta}$. The vertical red dashed lines correspond to the true values. 99

4.4 Posterior mean of the latent factor processes $\boldsymbol{x}_{:,k}, k = 1, 2, 3$ plotted with the true factor processes for the Poisson-GPFA model. 100

A.1 Estimation errors for $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$, $s(\boldsymbol{x}_i)$ and $z_{10}(\boldsymbol{x}_i)$ in the small-scale simulation study plotted on regular lattices. 126

B.1 Point estimates of parameters of interest from two-step estimation plotted against the point estimates from joint estimation. The blue dashed lines correspond to the 45 (and the -45) degree lines. 130

B.2 Left: Posterior standard deviation of $\hat{\ell}^{\star}_{ij}$ from two-step estimation plotted against the those from joint estimation. The dashed line corresponds to the 45 degree line. Middle: Box plot of the ratio between the posterior standard deviations from two-step estimation and joint estimation, respectively. Right: A zoomed-in version of the box plot from the middle panel. 131

B.3 Examples of unsatisfactory LNO-PP test results obtained by plotting the predicted cumulative spike counts from a trial i against the observed cumulative spike count from trial j for $j \neq i$. 132

B.4 LNO-PP test results for our model and for GPFA with different bin lengths. Each grey line, representing one sample from the posterior predictive distribution of spike times, plots the predicted cumulative spike counts on the Y-axis against the observed for the corresponding left-out neuron on the X-axis. The red line represents the average of all grey lines in the plot. The green dashed line corresponds to the 45-degree diagonal reference line. . . . 135

B.5 EFA results for simulated data from Section 3.5.1. Left: Cumulative proportion of variance explained by the d factors $cumvar(d)$ for different choices of d. Middle: Increase in the cumulative proportion of variance explained, $cumvar(d+1) - cumvar(d)$, for different choices of d. Right: The minimum of the max loading in each column of $\widehat{\boldsymbol{\Lambda}}_d$ for different values of d. The green dashed line corresponds to the chosen d. 136

B.6 EFA results for NEURON simulated data from Section 3.5.2. Left: Cumulative proportion of variance explained by the d factors $cumvar(d)$ for different values of d. Middle: Increase in the cumulative proportion of variance explained, $cumvar(d+1) - cumvar(d)$, for different values of d. Right: The minimum of the max loading in each column of $\widehat{\mathbf{\Lambda}}_d$ for different values of d. The green dashed line corresponds to the chosen d. 137

B.7 EFA results for real data of rats from Chapter 3.6. Plotted on the y-axis is the minimum of the max loading in each column of $\widehat{\mathbf{\Lambda}}_d$. The x-axis is different choices of d. The green dashed line corresponds to the chosen d. (a)-(c) correspond to ethanol, quinine, and sucrose experimental paradigms, respectively. 139

B.8 Runtimes for different choices of d in the simulation study using data simulated from the model. 140

B.9 Varimax-transformed estimated factor loading matrix $\widehat{\mathbf{\Lambda}}$ for different choices of d. 141

List of Tables

2.1 Comparison between HMC-NUTS, Laplace (MQ/MCMC), and Max-and-Smooth (MQ/MCMC). The MAE statistic is computed as $\text{KS}\left(\widehat{u}(\boldsymbol{x})\right) = \frac{\sum_{i=1}^{n} |u(\boldsymbol{x}_i) - \widehat{u}(\boldsymbol{x}_i)|}{n}$, where $u \in \{a, b, s\}$. $\text{KS}\left(\widehat{z}_{10}(\boldsymbol{x})\right)$ is the value of the two-sample KS test statistic between the HMC-NUTS and each other posterior for $z_{10}(\boldsymbol{x}_i)$ averaged over all spatial locations $i = 1, \ldots, 400$. 27

3.1 Run times for fitting the proposed model to simulated data under different inference settings. 60

Chapter 1

Introduction

The advancement of data collection technology across various fields has led to a large quantity of data available for statistical analysis, crucial for informing scientific and business decisions. A fundamental requirement for analyzing such "big data" is the availability of a flexible and efficient modelling framework, particularly for handling high-dimensional data with complex structures. In this regard, a popular approach is hierarchical modelling, which combines different structured sources of information in the dataset to form a unified modelling framework. The simplest form of a hierarchical model can be written as

$$\begin{aligned} y_i &\overset{\text{ind}}{\sim} p(y_i \mid \theta_i), \quad i = 1, \ldots, n \\ \boldsymbol{\theta} &\sim p(\boldsymbol{\theta} \mid \boldsymbol{\psi}), \end{aligned}$$

where $\boldsymbol{y} = (y_1, \ldots, y_n)$ represent the observed data, $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)$ are the *latent variables* or *random effects*, and $\boldsymbol{\psi} = (\psi_1, \ldots, \psi_m)$ are termed *hyperparameters* or *fixed effects*. Typically, the elements of $\boldsymbol{\theta}$ are independently and identically distributed (iid) given $\boldsymbol{\psi}$. That is, $p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) = \prod_{i=1}^{n} p(\theta_i \mid \boldsymbol{\psi})$, whose evaluation scales as $\mathcal{O}(n)$.

Despite their flexibility in model specification, hierarchical models also present computational challenges during model inference. These challenges manifest in two primary forms. First, the high dimensionality of data results in computationally expensive evaluations of probability distributions, particularly when $\boldsymbol{\theta}$ is not iid. For instance, the evaluation of $p(\boldsymbol{\theta} \mid \boldsymbol{\psi})$ scales as $\mathcal{O}(n^3)$ when it is a multivariate Normal distribution, which is the case in many applications, including those discussed in this thesis. Second, since the likelihood $p(\boldsymbol{y} \mid \boldsymbol{\psi}) = \int p(\boldsymbol{y}, \boldsymbol{\theta} \mid \boldsymbol{\psi}) p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) \, \mathrm{d}\boldsymbol{\theta}$ involves integration over $\boldsymbol{\theta}$, more sophisticated models typically lack closed-form solutions for inference. In this case, it is common to carry out Bayesian inference, with a prior $\pi(\boldsymbol{\psi})$ on $\boldsymbol{\psi}$, to estimate the joint posterior distribution

$p(\boldsymbol{\theta}, \boldsymbol{\psi} \mid \boldsymbol{y}) \propto p(\boldsymbol{y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{\psi}) \pi(\boldsymbol{\psi})$, which is conventionally sampled from using computationally intensive techniques such as Markov Chain Monte Carlo (MCMC) methods. As a result, the iterative nature of evaluations exacerbates the computational costs imposed by the high dimensionality of the data. Therefore, developing computational methods to tackle these two challenges is a crucial area of research, for which some background is given Section 1.1.

Motivated by practical applications in meteorology and neuroscience involving high-dimensional data, this thesis devises statistical models and inference methods to address the aforementioned challenges. Specifically, our scope is focused on a subset of hierarchical models with Gaussian process latent layers, known as *latent Gaussian models*, which have a wide range of applications. Throughout this thesis, we present model inference in a Bayesian setting for interpretability.

We begin by introducing a fast and accurate inference approach in Chapter 2 for the GEV-GP model (Smith and Naylor, 1987; Coles and Powell, 1996; Cooley et al., 2007; Davison et al., 2012), a class of models combining the generalized extreme value (GEV) distribution and Gaussian process (GP) priors on the GEV parameters. This model is particularly relevant for analyzing spatial data characterized by heavy-tailed distributions, such as extreme weather phenomena (e.g., Dyrrdal et al., 2015; Tye and Cooley, 2015; García et al., 2018). However, closed-form expressions are not available for the posterior distributions of the model parameters. Moreover, the high-dimensional nature of spatial data renders conventional MCMC methods prohibitively expensive. Our proposed inference method circumvents MCMC by employing a Laplace approximation to the marginal likelihood to derive an approximate posterior distribution. Computational efficiency is further enhanced by approximate GP evaluations using a sparsity-inducing kernel function. In our simulation studies, we show that our Laplace-based inference method is much faster than MCMC with comparable accuracy, and more accurate than another efficient MCMC alternative. The applicability of the inference approach is demonstrated in a real data analysis of extreme snowfalls.

In Chapter 3, we introduce a novel hierarchical factor analysis model for analyzing high-dimensional neural spike train recordings, along with an efficient inference procedure built upon the approximate inference methods in Chapter 2. The aim of this chapter lies in uncovering statistically meaningful interactions between neurons in the brain (Stevenson et al., 2008; Ostojic et al., 2009). Our proposed model extends existing factor analysis frameworks by incorporating latent layers that capture underlying neuronal activities. Given the high dimensionality of spike trains, existing neuron models often resort to time discretization for scalability (Yu et al., 2009; Macke et al., 2011; Wu et al., 2017; Wei et al., 2022), but the arbitrary choice of bin size can be detrimental to making scientifically re-

liable conclusions based on the model, which is demonstrated in a study using simulated spike trains. Our proposed model fills in the gap in the literature for a multivariate spike-train model to study neuronal interactions without coarse spike train binning. The model parameters are learned using an efficient two-step inference procedure, which leverages closed-form solutions for neuron-specific parameters and approximate inference techniques for the correlation parameters. Simulation studies highlight the efficacy of our modelling and inference approach in uncovering meaningful neuronal interactions. Experimental data analysis using our model reveals that the correlation patterns in rats' OFC brain region are predictive of outcomes delivered to rats following different cues.

Chapter 4 extends the computational framework of this thesis by presenting an exact and efficient inference method for stationary Gaussian processes. Leveraging an efficient Toeplitz system solver in the literature (Ammar and Gragg, 1987; Ling and Lysy, 2020; Ling, 2019), this method enables the evaluation of stationary GPs in "superfast" time, operating in $\mathcal{O}(n \log^2 n)$ complexity compared to the conventional $\mathcal{O}(n^3)$ (Golub and Van Loan, 1996), and outperforming the well-established "fast" time of $\mathcal{O}(n^2)$ (Levinson, 1946; Durbin, 1960; Trench, 1964; Zohar, 1969). We demonstrate the use of this method through the development of "superfast" MCMC algorithms tailored for the *Gaussian Process Factor Analysis* (GPFA) models for neural spike trains (Yu et al., 2009), including extensions to handle Poisson observations where closed-form inference is not readily available (Keeley et al., 2019). The results in this chapter provide essential computational tools for developing latent Gaussian models with more flexible GP structures. Notably, the "superfast" inference method offers a promising avenue for extending the neuron model introduced in Chapter 3 to capture non-Markovian temporal patterns in the latent processes.

In summary, this thesis contributes to the literature of Bayesian computational methods, improving computational efficiency for both existing and novel models while addressing complex challenges in high-dimensional data analysis. Chapter 2 introduces an approximate inference method, serving as a more efficient alternative to MCMC for the established GEV-GP models in analyzing spatial meteorological extremes. Chapter 3 proposes a novel model alongside a tailored efficient inference procedure, designed specifically for studying high-dimensional neural spike train data. Chapter 4 extends an existing rapid GP inference approach to efficient MCMC algorithms, with potential applications to models extending the work in Chapter 3. To illustrate efficacy and potential advantages of the proposed methodologies, they are assessed by comparisons with existing approaches wherever feasible. Furthermore, all computational methods are accompanied by efficient implementations in R packages for reproducibility and contribution to the statistical software community.

1.1 Background

In the following, we provide some background information related to the computational challenges in hierarchical modelling and approximate solutions for them.

1.1.1 Scalable Covariance Matrix Computation

A *Gaussian process* (GP) is a stochastic process characterizing the behaviour of a function $f(x)$ on an infinite-dimensional domain. It is often denoted as $f(x) \sim \mathcal{GP}(\mu(x), k(x, x'))$, where $\mu(x)$ is the mean function and $k(x, x') = \mathrm{Cov}(f(x), f(x'))$ is the covariance kernel function. For example, $f(x)$ could represent the temperature at a given location and time x. The covariance between function values evaluated at two times is represented by the kernel function $k(x, x')$, which fully defines the GP when the process is assumed to have a zero mean. When the kernel function depends solely on the distance between x and x', such as the radial basis kernel $k(x, x') = \sigma^2 \exp(-(x - x')^2/(2\ell^2))$, it leads to the special case of *stationary* GPs. For more details on GPs, refer to Rasmussen and Williams (2006).

Notably, n equally spaced realizations of a GP are jointly distributed as an n-dimensional multivariate Normal (MVN) random variable, with the covariance matrix parameterized by the kernel function. Evaluating n-dimensional MVN distributions poses a computationally expensive problem scaling as $\mathcal{O}(n^3)$, due to the need to invert the $n \times n$ covariance matrix. For a symmetric and positive definite covariance matrix $\boldsymbol{\Sigma}$, the numerically stable Cholesky decomposition is the standard routine for solving the system $\boldsymbol{\Sigma x} = \boldsymbol{b}$ for $\boldsymbol{x}$ and for calculating the log-determinant of $\boldsymbol{\Sigma}$. The cost of this procedure is dominated by the $n^3/6$ operations of producing the Cholesky factor $\boldsymbol{L}$ such that $\boldsymbol{\Sigma} = \boldsymbol{L}\boldsymbol{L}^T$. Both exact and approximate solutions have been proposed to alleviate the computational burden associated with covariance matrix computation. Exact solutions leverage the special structure of the covariance matrix, while approximate solutions seeks faster computation using an approximate representation of the covariance matrix.

The covariance matrix resulting from a stationary Gaussian process is known to have the diagonal-constant property, wherein elements along each diagonal from upper left to lower right are identical in the form of

$$\boldsymbol{V} = \begin{bmatrix} v_0 & v_{-1} & v_{-2} & \cdots & v_{-(N-1)} \\ v_1 & v_0 & v_{-1} & \cdots & v_{-(N-2)} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ v_{N-1} & v_{N-2} & v_{N-3} & \cdots & v_0 \end{bmatrix}$$

This is referred to as a *Toeplitz matrix*. For solving Toeplitz systems $\boldsymbol{V} \cdot \boldsymbol{x} = \boldsymbol{b}$, there exist "fast" algorithms which require $\mathcal{O}(n^2)$ operations rather than $\mathcal{O}(n^3)$. Notable examples are the Levinson-Durbin (Levinson, 1946; Durbin, 1960) and Trench-Zohar algorithms (Trench, 1964; Zohar, 1969). As n grows, the quadratic time complexity still poses significant computational challenges. In Chapter 4, we show a "superfast" $\mathcal{O}(n \log^2 n)$ algorithm for Toeplitz matrix inversion and log-determinant calculation.

When n is in the order of tens of thousands and the MVN evaluation is performed repeatedly in an iterative manner, approximate computation of the covariance matrix offers substantial improvement in computational speed. One class of approximation methods involves low-rank representations of the form $\boldsymbol{V} = \boldsymbol{L}\boldsymbol{L}^T$ with a rank-m ($m \ll n$) matrix $\boldsymbol{L}$, which reduces the complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(m^2 n)$ (e.g., Williams and Seeger, 2000; Smola and Schölkopf, 2000; Fine and Scheinberg, 2002). These methods are also connected to factor analysis which summarizes high-dimensional inputs via lower-dimensional factors – a dimension reduction technique explored in Chapter 3. Another method scaling as $\mathcal{O}(m^2 n)$ is the sparse GP framework. It learns a small set of pseudo-inputs or inducing points of size m through optimization to approximate the full GP (Snelson and Ghahramani, 2006a,b; Quiñonero-Candela and Rasmussen, 2005). More comprehensive reviews of GP approximations are given in Rasmussen and Williams (2006) and Liu et al. (2020). While these methods offers promising computational enhancement for GP models, their representations of the GPs must be inferred through iterative algorithms. For complicated latent Gaussian models with non-Gaussian likelihoods, their applications are often limited due to difficulties in implementing nested iterative procedures or optimizations.

An alternative approach to approximating the covariance matrix is by employing a set of stochastic partial differential equations (SPDEs) to represent the covariance kernel function. The SPDE approximation to the Matérn kernel introduced in Lindgren et al. (2011) has gained significant popularity, especially when combined with the use of the integrated Laplace (INLA) approximation framework proposed by Rue et al. (2009). For an overview, refer to the review by Rue et al. (2017). Unlike the sparse GP approximations discussed above, a notable advantage of the SPDE approach is its analytic form. Specifically, it does not rely on iterative algorithms to find the approximate representation, thereby simplifying inference for complex models. Hence, in Chapter 2, we employ the SPDE approximation to greatly speed up computation. Furthermore, it has been noted in Miller et al. (2020) that this SPDE technique is equivalent to the basis-penalty smoother (Wood, 2011) within the generalized additive model (GAM) framework (Wood, 2017), a widely adopted approach for frequentist inference.

Scalable methods for covariance matrix computation are fundamental components for efficient inference. In the Bayesian paradigm, the primary focus of inference is often the

posterior distribution $p(\theta \mid y)$ or an expectation derived from it. In cases where analytic derivation of the posterior distribution is not feasible, alternative inferential procedures are required to approximate the distribution or expectation. Below, we give a brief overview of approximate Bayesian inference.

1.1.2 Approximate Bayesian Inference

When an analytic form for the posterior distribution of interest is unavailable, Markov Chain Monte Carlo (MCMC) algorithms (Metropolis et al., 1953; Hastings, 1970; Geman and Geman, 1984; Gelfand and Smith, 1990) are conventionally employed to sample from it. With a sufficiently large number of samples, the Monte Carlo approximation to the true posterior can be made arbitrarily accurate, assuming convergence to the true distribution. Since the distance between the approximation and the truth (without an analytic form) is not quantifiable in practice, the algorithm is usually run iteratively with multiple parallel sampling chains until convergence diagnostic criteria are met, such as an R-hat metric around 1 (Gelman and Rubin, 1992; Vehtari et al., 2021) and an estimated effective sample size over 400 (e.g., Gelman et al., 2013; Gong and Flegal, 2016; Vats et al., 2019). Therefore, MCMC approximated posteriors are often viewed as gold standards in terms of inferential results for models without closed-form solutions. The implementation of MCMC for complex models has been facilitated by popular statistical software packages like BUGS (Spiegelhalter et al., 1996), JAGS (Plummer, 2003), Nimble (de Valpine et al., 2017), Stan (Stan Development Team, 2020), etc. In Chapter 2, the Stan implementation of the No-U-Turn variant (Hoffman and Gelman, 2014) of the Hamiltonian Monte Carlo sampler (Neal, 2011) is used as a benchmark for evaluating our proposed inference method. Though off-the-shelf MCMC implementations are helpful for generic problems, more sophisticated models often still require users' own implementation. In Chapter 4, we develop specialized MCMC samplers using efficient matrix operations for problems where the off-the-shelf MCMC implementations are too slow.

Unlike MCMC algorithms, which approximate the exact posterior, approximate inference methods instead construct an approximate posterior, e.g., $q(\theta \mid y) \approx p(\theta \mid y)$, often through deterministic procedures. These methods commonly replace sampling with optimization. Variational inference is a prominent alternative to MCMC for approximating posterior distributions. It seeks a surrogate for the true posterior known as the variational distribution $q(\theta \mid y)$, which is either constrained to belong to a parametric family of distributions, typically Normal, or approximated via sampling (Zhang et al., 2016). The appropriate $q(\theta \mid y)$ is found by minimizing its Kullback-Leibler (KL) divergence from the true posterior. For computational tractability, multivariate posterior is often assumed to

factorize into $q(\boldsymbol{\theta} \mid y) = \prod_i^k q_i(\theta_i \mid y)$ (e.g., see section 5 of Wainwright and Jordan, 2008), and $q_i(\theta_i \mid y)$ is commonly taken to be Normal. Review of variational methods are given in Jordan et al. (1999) and Blei et al. (2016).

A simpler and perhaps the most common approach for approximating posterior densities is the Normal or mode-quadrature approximation. This method approximates the posterior using a Normal distribution with its mean centred at the posterior mode and its variance derived from the inverse Hessian matrix at the posterior mode. This simplifies inference to a mode-finding optimization problem using techniques such as the Expectation-Maximization algorithm (Dempster et al., 1977) or the Laplace approximation (Tierney and Kadane, 1986) to the marginal likelihood. Notably, the latter is a key component to the popular INLA methodology (Rue et al., 2009) and the `R-INLA` software (Lindgren and Rue, 2015), which is a widely adopted MCMC alternative for spatial and temporal mixed-effects modelling (Bakka et al., 2018). We employ the Normal approximation and the Laplace approximation in both Chapters 2 and 3.

Chapter 2

Fast and Scalable Inference for Spatial Extreme Value Models

2.1 Introduction

Statistical modelling of extreme weather data has important applications including efficient wind energy management and damage prevention for floods and hurricanes. Extreme value theory (EVT) provides the tools for analyzing such data, covering a wide range of applications from climatology (Casson and Coles, 1999; Fawcett and Walshaw, 2006; Blanchet and Davison, 2011) to insurance and finance (Embrechts et al., 1997; Gilli and Këllezi, 2006). Developed under the EVT assumptions, the generalized extreme value (GEV) distribution is commonly used for analyzing a sequence of maxima within non-overlapping time periods. Hence, the GEV distribution is also known as the block maxima model. Coles (2001) provided an overview of the properties and applications of the GEV distribution with an emphasis on the analysis of hydrological and meteorological data.

Weather extremes are distinguished from other types of extremes due to the potential presence of spatial patterns. To incorporate the additional spatial information such as longitude and latitude into the analysis, Bayesian hierarchical modelling is a natural approach. The work of Smith and Naylor (1987) was the first to show the advantages of Bayesian analysis over likelihood-based methods in some cases of extreme value modelling. Coles and Powell (1996) reviewed the early studies of Bayesian models of extreme values, and discussed the use of Bayesian methods to pool spatial information when data are sparse in the spatial domain. Later, Coles and Casson (1998) presented a simulation study of hurricane wind speeds where each GEV parameter was linked to the spatial covariates via

a linear combination of a regression function and a Gaussian process. Cooley et al. (2007) modelled precipitation extremes as conditionally independent given the GEV model parameters, which are assumed to follow spatial Gaussian processes. Adding a copula model in the data layer, Sang and Gelfand (2010) dropped the conditional independence assumption, while retaining the spatial Gaussian process layer for the parameters. Reich and Shaby (2012) and Stephenson et al. (2016) proposed spatial max-stable models whose marginal distribution is GEV, and thus can be viewed as infinite-dimensional generalizations of the GEV model. Davison et al. (2012) provide a comprehensive review of Bayesian hierarchical models of spatial extremes, discussing their development in terms of methodology and theoretical framework. Specifically, they compare three classes of models: latent variable models, copulas, and max-stable models. While latent variable models have the flexibility to specify a host of different latent variable structures, they have been criticized for the unrealistic assumption of conditional independence between nearby locations. In contrast, copula and max-stable models tend to capture the dependency between these locations more efficiently. That being said, Davison et al. (2012) argue that latent variable models are more useful for estimating marginal (location-specific) quantities such as the expected return level, an important metric for extreme weather analyses (see Section 2.2.4). This chapter, therefore, focuses on latent variable GEV models with parameters modelled via spatial Gaussian processes, which we refer to as GEV-GP models.

There have been numerous applications of the GEV-GP framework to modelling weather extremes in different regions and countries (e.g., Dyrrdal et al., 2015; Tye and Cooley, 2015; Jalbert et al., 2017; García et al., 2018; Jóhannesson et al., 2022; Youngman, 2022). However, inference for these complex models can be computationally challenging. This is because the computational complexity of likelihood evaluations for the latent Gaussian process scales as $\mathcal{O}(n^3)$, where n is the number of sites being studied. Therefore, computations for large-scale spatial analyses involving numerous sites can quickly become prohibitively expensive. This is exacerbated by the fact that the n latent variables corresponding to each location cannot be analytically integrated out. Instead, Markov Chain Monte Carlo (MCMC) methods are often used in the context of Bayesian inference to sample from the joint space of latent variables and model parameters. However, MCMC algorithms can take days or even weeks to converge when the number of spatial random effects is moderate to large.

As an alternative to MCMC, in this chapter we develop a Bayesian inference method for GEV-GP models based on the Laplace approximation (e.g., Tierney and Kadane, 1986; Barndorff-Nielsen and Cox, 1989; Breslow and Lin, 1995; Skaug and Fournier, 2006), a technique for converting an intractable integration problem into an optimization problem which is much easier to solve. The Laplace approximation is not new to spatio-temporal

modelling. Indeed, a refinement of it known as the integrated nested Laplace approximation (INLA) (Rue et al., 2009, 2017) is typically more accurate, and has been recently used to model weather extremes (e.g. Opitz et al., 2018; Castro-Camilo et al., 2019, 2020). However, the computational complexity of INLA increases exponentially with the number of hyperparameters in the model. In practice, this limitation is so severe that the flagship INLA implementation via the `R-INLA` package (Lindgren and Rue, 2015) can only handle multiple random effects if they enter linearly into the likelihood function, which is not the case for GEV-GP models. In contrast, the original Laplace approximation imposes no such restriction. We use the Laplace approximation here in combination with a sparsity-inducing approximation of certain spatial Gaussian processes as Gaussian Markov random fields (Lindgren et al., 2011). Our simulation studies indicate that the proposed approach accurately approximates Bayesian posterior predictive return levels for the full range of GEV-GP models, for several thousand spatial locations and several orders of magnitude faster than state-of-the-art MCMC.

The remainder of this chapter is structured as follows. Section 2.2 discusses the details of the GEV-GP model, the Bayesian Laplace approximation, and the sparsity-inducing spatial covariance matrix approximation. Section 2.3 presents simulation studies comparing our Laplace approximation to both a state-of-the-art MCMC algorithm, and to another approximate inference method for GEV-GP models known as "Max-and-Smooth" (Hrafnkelsson et al., 2021), which is conceptually similar but typically much faster than Laplace. Section 2.4 presents a case study analyzing extreme monthly total snowfall in Canada. In particular, it highlights the importance of having multiple spatial random effects in the GEV model for accurately predicting extreme weather events, a modelling strategy which cannot be achieved in practice using INLA. Concluding remarks are offered in Section 2.5. Efficient implementations of our methods are provided in the R/C++ library `SpatialGEV`.

2.2 Methodology

The GEV-GP is a hierarchical model consisting of a data layer and a spatial random effects layer. Let $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n \in \mathbb{R}^2$ denote the geographical coordinates of n locations, and let y_{ik} denote the kth extreme value measurement at location i, for $k = 1, \ldots, n_i$. The data layer specifies that each observation y_{ik} has a GEV distribution, denoted by $y \sim \text{GEV}(a, b_o, s_o)$,

for which the CDF is given by

$$F(y \mid a, b_o, s_o) = \begin{cases} \exp\left\{-\left(1 + s_o \frac{y-a}{b_o}\right)^{-\frac{1}{s_o}}\right\} & s_o \neq 0, \\ \exp\left\{-\exp\left(-\frac{y-a}{b_o}\right)\right\} & s_o = 0, \end{cases} \tag{2.1}$$

where $a \in \mathbb{R}$, $b_o > 0$, and $s_o \in \mathbb{R}$ are location, scale, and shape parameters, respectively. The support of the GEV distribution depends on the parameter values: y is bounded below by $a - b_o/s_o$ when $s_o > 0$, bounded above by $a - b_o/s_o$ when $s_o < 0$. To avoid imposing an upper bound on extreme weather events, we assume that $s_o > 0$.

In order to capture the spatial dependence in the data, we let the GEV parameters $a(\boldsymbol{x})$, $b_o(\boldsymbol{x})$, and $s_o(\boldsymbol{x})$ be location-dependent and model some or all of them as spatially varying random effects. In the most general setting, we follow Cooley et al. (2007) and use independent latent Gaussian processes for each GEV parameter $a(\boldsymbol{x})$, $b(\boldsymbol{x}) = \log(b_o(\boldsymbol{x}))$, and $s(\boldsymbol{x}) = \log(s_o(\boldsymbol{x}))$. A Gaussian process

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')) \tag{2.2}$$

is fully characterized by its mean function $m(\boldsymbol{x}) = E[f(\boldsymbol{x})]$ and its kernel function $k(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}(f(\boldsymbol{x}), f(\boldsymbol{x}'))$, the latter of which captures the strength of the spatial correlation between locations. Here, we give the mean function $m(\boldsymbol{x}) = \boldsymbol{c}(\boldsymbol{x})^T\boldsymbol{\beta}$ a regression structure, where $\boldsymbol{c}(\boldsymbol{x})$ is a (known) vector-valued function of covariates at location $\boldsymbol{x}$ and $\boldsymbol{\beta}$ is the coefficient vector. For the kernel function, we choose the Matérn kernel (Handcock and Stein, 1993) given by

$$k(\boldsymbol{x}, \boldsymbol{x}' \mid \sigma^2, \kappa, \nu) = \sigma^2 \frac{2^{1-\nu}}{\Gamma(\nu)} (\kappa||\boldsymbol{x} - \boldsymbol{x}'||)^\nu K_\nu(\kappa||\boldsymbol{x} - \boldsymbol{x}'||), \tag{2.3}$$

where $K_v(\cdot)$ is the modified Bessel function of the second kind (Abramowitz and Stegun, 1972), σ^2 is the variance parameter, $1/\kappa > 0$ is the range parameter, and $\nu > 0$ is the shape parameter that is typically fixed. Throughout this chapter, we let $\nu = 1$ (Lindgren et al., 2011). Thus, $\boldsymbol{\beta}$, σ^2 and κ are hyperparameters of the Gaussian process.

Given the spatial locations, the data are assumed to follow independent GEV distributions each with their own parameters. The complete GEV-GP hierarchical model is thus

$$\begin{aligned} y_{ik} \mid a(\boldsymbol{x}_i), b(\boldsymbol{x}_i), s(\boldsymbol{x}_i) &\overset{\text{ind}}{\sim} \text{GEV}\left(a(\boldsymbol{x}_i), \exp(b(\boldsymbol{x}_i)), \exp(s(\boldsymbol{x}_i))\right) \\ a(\boldsymbol{x}) \mid \boldsymbol{\beta}_a, \sigma_a, \kappa_a &\sim \mathcal{GP}\left(\boldsymbol{c}_a(\boldsymbol{x})^T\boldsymbol{\beta}_a,\ k(\boldsymbol{x}, \boldsymbol{x}' \mid \sigma_a^2, \kappa_a)\right) \\ b(\boldsymbol{x}) \mid \boldsymbol{\beta}_b, \sigma_b, \kappa_b &\sim \mathcal{GP}\left(\boldsymbol{c}_b(\boldsymbol{x})^T\boldsymbol{\beta}_b,\ k(\boldsymbol{x}, \boldsymbol{x}' \mid \sigma_b^2, \kappa_b)\right) \\ s(\boldsymbol{x}) \mid \boldsymbol{\beta}_s, \sigma_s, \kappa_s &\sim \mathcal{GP}\left(\boldsymbol{c}_s(\boldsymbol{x})^T\boldsymbol{\beta}_s,\ k(\boldsymbol{x}, \boldsymbol{x}' \mid \sigma_s^2, \kappa_s)\right). \end{aligned} \tag{2.4}$$

The hyperparameters of the model are

$$\boldsymbol{\theta} = (\boldsymbol{\beta}_a, \log(\sigma_a^2), \log(\kappa_a), \boldsymbol{\beta}_b, \log(\sigma_b^2), \log(\kappa_b), \boldsymbol{\beta}_s, \log(\sigma_s^2), \log(\kappa_s)).$$

2.2.1 Scalable Likelihood Evaluations

Let $\boldsymbol{y}_i = (y_{i1}, \ldots, y_{in_i})$ denote the extreme value observations at location $\boldsymbol{x}_i$, and let $a_i = a(\boldsymbol{x}_i)$, $b_i = b(\boldsymbol{x}_i)$, and $s_i = s(\boldsymbol{x}_i)$ denote the corresponding random effects. Let $\boldsymbol{y} = (\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n)^T$, $\boldsymbol{a} = (a_1, \ldots, a_n)^T$, $\boldsymbol{b} = (b_1, \ldots, b_n)^T$, $\boldsymbol{s} = (s_1, \ldots, s_n)^T$, and $\boldsymbol{u} = (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{s})$. Then the joint distribution of data and random effects is

$$\begin{aligned} p(\boldsymbol{y}, \boldsymbol{u} \mid \boldsymbol{\theta}) = \prod_{i=1}^{n} \prod_{k=1}^{n_i} \left\{ \frac{1}{\exp(b_i)} \left(1 + \exp(s_i) w_{ik}\right)^{-\frac{1+\exp(s_i)}{\exp(s_i)}} \exp\left[-\left(1 + \exp(s_i) w_{ik}\right)^{-\frac{1}{\exp(s_i)}} \right] \right\} \\ \times \frac{1}{\sqrt{(2\pi)^n \mid \boldsymbol{\Sigma}_a \mid}} \exp\left[-\frac{1}{2} (\boldsymbol{a} - \boldsymbol{C}_a \boldsymbol{\beta}_a)^T \boldsymbol{\Sigma}_a^{-1} (\boldsymbol{a} - \boldsymbol{C}_a \boldsymbol{\beta}_a) \right] \\ \times \frac{1}{\sqrt{(2\pi)^n \mid \boldsymbol{\Sigma}_b \mid}} \exp\left[-\frac{1}{2} (\boldsymbol{b} - \boldsymbol{C}_b \boldsymbol{\beta}_b)^T \boldsymbol{\Sigma}_b^{-1} (\boldsymbol{b} - \boldsymbol{C}_b \boldsymbol{\beta}_b) \right] \\ \times \frac{1}{\sqrt{(2\pi)^n \mid \boldsymbol{\Sigma}_s \mid}} \exp\left[-\frac{1}{2} (\boldsymbol{s} - \boldsymbol{C}_s \boldsymbol{\beta}_s)^T \boldsymbol{\Sigma}_s^{-1} (\boldsymbol{s} - \boldsymbol{C}_s \boldsymbol{\beta}_s) \right], \end{aligned} \tag{2.5}$$

where $w_{ik} = (y_{ik} - a(\boldsymbol{x}_i))/\exp(b(\boldsymbol{x}_i))$, $\boldsymbol{C}_r = (\boldsymbol{c}_r(\boldsymbol{x}_1) \cdots \boldsymbol{c}_r(\boldsymbol{x}_n))^T$, and $\boldsymbol{\Sigma}_r = [k(\boldsymbol{x}_i, \boldsymbol{x}_j \mid \sigma_r^2, \kappa_r)]_{1 \le i,j \le n}$ for $r \in \{a, b, s\}$.

Since the matrix inversions $\boldsymbol{\Sigma}_a^{-1}$, $\boldsymbol{\Sigma}_b^{-1}$ and $\boldsymbol{\Sigma}_s^{-1}$ and corresponding determinants in (2.5) are $\mathcal{O}(n^3)$, each evaluation of $p(\boldsymbol{y}, \boldsymbol{u} \mid \boldsymbol{\theta})$ quickly becomes extremely expensive as the number of spatial locations n increases. One approach to this problem (e.g., Sang and Gelfand, 2008; Schliep et al., 2010; Cooley and Sain, 2010) is to employ a conditional autoregressive (CAR) model (Besag, 1974; Besag et al., 1991) for the random effects, for which the underlying assumption of conditional independence leads to computationally efficient factorizations of the GP contributions to (2.5). However, the CAR model is only defined at a predetermined set of spatial locations. Alternatively, Lindgren et al. (2011) developed an efficient approximation to the GP terms in (2.5) using a stochastic partial differential equation (SPDE) for the Matérn covariance kernel (see also Opitz, 2017; Miller et al., 2020). Specifically, a GP $f(\boldsymbol{x})$ with the Matérn kernel defined in (2.3) can be written as the solution to the SPDE

$$(\kappa^2 - \Delta)^{\alpha/2} f(\boldsymbol{x}) = \mathcal{W}(\boldsymbol{x}), \quad \alpha = \nu + n/2, \tag{2.6}$$

where $(\kappa^2 - \Delta)^{\alpha/2}$ is the differential operator with $\Delta = \sum_{i=1}^{2} \partial^2/\partial^2 x_i^2$, and $\mathcal{W}(\boldsymbol{x})$ is a Gaussian white noise process. Lindgren et al. (2011) showed how to construct a finite element representation of the solution to (2.6), which in turn allows one to approximate $\boldsymbol{f} = (f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n))$ as a Gaussian Markov random field (GMRF), for which covariance matrix inversions and determinant calculations are only $\mathcal{O}(n^{3/2})$ (Rue and Held, 2005). Moreover, the SPDE approximation readily lends itself to making predictions at new spatial locations whereas the CAR approach does not.

Yet another means of reducing the computational burden is to approximate $f(\boldsymbol{x})$ via a basis representation, in the framework of generalized additive models (GAMs) (Wood, 2011; Wood et al., 2016). This approach has been implemented for GEV models in the R packages `mgcv` (Wood, 2017, 2022) and `evgam` (Youngman, 2022). While GAMs typically treat the spatially-varying parameter function $f(\boldsymbol{x})$ as deterministic, there exists a basis expansion for the Matérn GP family with fixed range parameter $1/\kappa$, formulated along similar lines as the SPDE approximation above (Lindgren and Rue, 2015). The random effects of the model are thus reduced to the coefficients of the basis expansion. As the number of these is user-specified and independent of the number of spatial locations, basis expansions provide an invaluable means of scaling approximate GEV-GP inference to massive spatial datasets. However, basis function expansions are notoriously less accurate than keeping all n spatial random effects when the underlying spatial process is not smooth and/or exhibits pockets of short-range correlation (e.g., Wood, 2020; Lindgren et al., 2022). In this chapter, we therefore focus on the SPDE approximation to achieve scalable evaluations of the joint likelihood function (2.5).

2.2.2 Estimation of Hyperparameters

We proceed in a Bayesian context by specifying a prior $\pi(\boldsymbol{\theta})$ on the hyperparameters of the model in (2.4). The choice of priors for hyperparameters in spatial models has been widely studied. For example, Banerjee et al. (2003) suggested using informative priors on the variance and range parameters. However, such informative priors typically require domain knowledge from an expert. Simpson et al. (2017) introduced the so-called PC prior, which penalizes complex models by shrinking the range parameter to infinity and the variance to zero. In this chapter, we employ a combination of uninformative and weakly informative priors (described fully in Sections 2.3 and 2.4), obtaining good results on real and simulated data. Since the focus of this chapter is on computations, an in-depth study of different priors is omitted. However, we note that the choice of prior does not limit the applicability of the proposed methods.

Given the prior $\pi(\boldsymbol{\theta})$, Bayesian inference is typically accomplished using MCMC to sample from the joint posterior distribution of hyperparameters and random effects,

$$p(\boldsymbol{u}, \boldsymbol{\theta} \mid \boldsymbol{y}) \propto p(\boldsymbol{y}, \boldsymbol{u} \mid \boldsymbol{\theta}) \pi(\boldsymbol{\theta}). \tag{2.7}$$

However, the mixing time of MCMC algorithms for the posterior distribution (2.7) grows quickly as a function of the number of spatial locations n. As an alternative, we now present the Laplace approximation to the marginal hyperparameter distribution

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{y}) \pi(\boldsymbol{\theta}), \tag{2.8}$$

where

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta} \mid \boldsymbol{y}) &= \int p(\boldsymbol{y}, \boldsymbol{u} \mid \boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{u} \\ &= \int \exp\big\{ - G(\boldsymbol{u}; \boldsymbol{\theta}) \big\} \, d\boldsymbol{u}. \end{aligned} \tag{2.9}$$

For the GEV-GP model, this integral is intractable, suggesting the use of MCMC on the joint posterior distribution (2.7). Instead, the Laplace approximation converts the intractable integral into a tractable optimization problem. It does this by noting that for fixed $\boldsymbol{\theta}$, a second order Taylor expansion of $-G(\boldsymbol{u}; \boldsymbol{\theta}) = \log p(\boldsymbol{y} \mid \boldsymbol{\theta}) + \log p(\boldsymbol{u} \mid \boldsymbol{y}, \boldsymbol{\theta})$ about its mode in $\boldsymbol{u}$ approximates the conditional distribution of the random effects given both data and hyperparameters as multivariate normal, which in turn produces a tractable approximation to the marginal distribution $p(\boldsymbol{y} \mid \boldsymbol{\theta})$. Specifically, for given $\boldsymbol{\theta}$ let

$$\boldsymbol{u}_{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{u}} G(\boldsymbol{u}; \boldsymbol{\theta}), \tag{2.10}$$

and let us approximate $G(\boldsymbol{u}; \boldsymbol{\theta})$ by its second-order Taylor expansion about $\boldsymbol{u}_{\boldsymbol{\theta}}$:

$$G(\boldsymbol{u}; \boldsymbol{\theta}) \approx G(\boldsymbol{u}_{\boldsymbol{\theta}}; \boldsymbol{\theta}) + \frac{1}{2} (\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{\theta}})^T \boldsymbol{H}_{\boldsymbol{\theta}} (\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{\theta}}), \tag{2.11}$$

where

$$\boldsymbol{H}_{\boldsymbol{\theta}} = \frac{\partial^2}{\partial \boldsymbol{u} \partial \boldsymbol{u}^T} G(\boldsymbol{u}; \boldsymbol{\theta})|_{\boldsymbol{u} = \boldsymbol{u}_{\boldsymbol{\theta}}} := \frac{\partial^2}{\partial \boldsymbol{u} \partial \boldsymbol{u}^T} G(\boldsymbol{u}_{\boldsymbol{\theta}}; \boldsymbol{\theta}), \tag{2.12}$$

and the first-order term $\partial / \partial \boldsymbol{u} \; G(\boldsymbol{u}_{\boldsymbol{\theta}}; \boldsymbol{\theta})^T (\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{\theta}})$ has vanished since the gradient of $G(\boldsymbol{u}; \boldsymbol{\theta})$ equals zero at the mode $\boldsymbol{u}_{\boldsymbol{\theta}}$. Let $\phi(\boldsymbol{z} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ denote the PDF of $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. Substituting the Taylor expansion (2.11) into (2.9) gives the Laplace approximation to the marginal

likelihood:

$$\begin{aligned}\widehat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{y}) &= \int \exp\left\{-G(\boldsymbol{u}_{\boldsymbol{\theta}}; \boldsymbol{\theta}) - \frac{1}{2}(\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{\theta}})^T \boldsymbol{H}_{\boldsymbol{\theta}}(\boldsymbol{u} - \boldsymbol{u}_{\boldsymbol{\theta}})\right\} \mathrm{d}\boldsymbol{u} \\ &\propto \exp\left\{-G(\boldsymbol{u}_{\boldsymbol{\theta}}; \boldsymbol{\theta}) - \frac{1}{2}\log|\boldsymbol{H}_{\boldsymbol{\theta}}|\right\} \cdot \int \phi(\boldsymbol{u}|\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{H}_{\boldsymbol{\theta}}^{-1}) \, \mathrm{d}\boldsymbol{u} \\ &= \exp\left\{-G(\boldsymbol{u}_{\boldsymbol{\theta}}; \boldsymbol{\theta}) - \frac{1}{2}\log|\,\boldsymbol{H}_{\boldsymbol{\theta}}|\right\}.\end{aligned} \tag{2.13}$$

The Laplace approximation (2.13) leads to the marginal posterior approximation

$$\widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto \widehat{\mathcal{L}}(\boldsymbol{\theta} \mid \boldsymbol{y})\pi(\boldsymbol{\theta}). \tag{2.14}$$

In contrast to the joint posterior (2.7) on both hyperparameters and random effects, $\widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$ can be explored via MCMC targeting the much smaller set of hyperparameters alone. The Laplace approximation can also be used to forego MCMC altogether by constructing a Normal approximation to the hyperparameter posterior distribution,

$$\boldsymbol{\theta} \mid \boldsymbol{y} \approx \mathcal{N}(\widehat{\boldsymbol{\theta}}, \ \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}}), \tag{2.15}$$

where $\widehat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log \widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$ and $\widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} = -\left[\partial^2/\partial\boldsymbol{\theta}^2 \ \log \widehat{p}(\widehat{\boldsymbol{\theta}} \mid \boldsymbol{y})\right]^{-1}$.

The approximation error of the Laplace approximation (2.13) has been investigated by e.g., Shun and McCullagh (1995); Rue et al. (2009); Ogden (2021). Assuming, for simplicity, that there is an equal number of observations $n_i \equiv n_0$ at each spatial location, the Bayesian normal approximation (2.15) converges to the true posterior as $n_0 \to \infty$ with $n/n_0 \to 0$, where n is the number of spatial locations, which includes the possibility of $n \to \infty$ as well (Rue et al., 2009).

Remark. Our treatment of the GP mean coefficient hyperparameters $\boldsymbol{\beta}_a$, $\boldsymbol{\beta}_b$, and $\boldsymbol{\beta}_s$ differs from INLA in that – provided we use a Gaussian prior for these variables as we have done in Section 2.3 – we do not group them together with the random effect parameters $\boldsymbol{u}$ to integrate them out of the model. Since the computational cost of INLA increases exponentially with the number of hyperparameters, it is advantageous for it to integrate out as many of these as possible. Since the Laplace approximation has no such limitation, we opted to use it only to integrate over the random effects, thereby obtaining an approximate marginal likelihood for all the hyperparameters of the model. We could also have used Laplace to integrate over the mean coefficient hyperparameters. While a systematic comparison between the two approaches in terms of speed and accuracy is beyond

the scope of this chapter, our preliminary investigations indicate that their performance is fairly similar. That being said, with a Gaussian prior, the mean coefficient hyperparameters can be analytically integrated out of the random-effects model, thus leading to a more accurate Laplace approximation than either of the aforementioned alternatives. However, performing this integration results in a model for which the random-effects precision matrix is no longer sparse, such that the computational benefits of the SPDE approximation are destroyed.

2.2.3 Estimation of Random Effects

As noted in Section 2.2.2, the Laplace marginal likelihood is obtained by approximating the conditional random-effects distribution $p(\boldsymbol{u} \mid \boldsymbol{y}, \boldsymbol{\theta})$ as

$$\boldsymbol{u} \mid \boldsymbol{y}, \boldsymbol{\theta} \approx \mathcal{N}(\boldsymbol{u}_{\boldsymbol{\theta}}, \widehat{\boldsymbol{V}}_{\boldsymbol{u}}(\boldsymbol{\theta})), \tag{2.16}$$

where $\widehat{\boldsymbol{V}}_{\boldsymbol{u}}(\boldsymbol{\theta}) = \boldsymbol{H}_{\boldsymbol{\theta}}^{-1}$, and $\boldsymbol{u}_{\boldsymbol{\theta}}$ and $\boldsymbol{H}_{\boldsymbol{\theta}}$ are given by (2.10) and (2.12). This suggests a two-step sampling scheme for a Laplace approximation to the random effects posterior distribution $p(\boldsymbol{u} \mid \boldsymbol{y}) = \int p(\boldsymbol{u} \mid \boldsymbol{y}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{\theta}$, by first drawing a hyperparameter $\boldsymbol{\theta}_i \sim \mathcal{N}(\widehat{\boldsymbol{\theta}}, \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}})$ from the Normal hyperparameter posterior approximation (2.15), then drawing $\boldsymbol{u}_i \sim \mathcal{N}(\boldsymbol{u}_{\boldsymbol{\theta}_i}, \widehat{\boldsymbol{V}}_{\boldsymbol{u}}(\boldsymbol{\theta}_i))$. However, this procedure is computationally intensive because a separate numerical optimization must be performed for each $\boldsymbol{\theta}_i$ to find the mean and covariance in (2.16). Alternatively, we propose a much faster method to approximate $p(\boldsymbol{u} \mid \boldsymbol{y})$ as follows.

Given the conditional Laplace approximation (2.16), we apply a first-order Taylor expansion of $\boldsymbol{u}_{\boldsymbol{\theta}}$ and a zeroth-order Taylor expansion of $\widehat{\boldsymbol{V}}_u(\boldsymbol{\theta})$ about $\boldsymbol{\theta} = \widehat{\boldsymbol{\theta}}$, such that these two functions are approximated by

$$\boldsymbol{u}_{\boldsymbol{\theta}} \approx \boldsymbol{u}_{\widehat{\boldsymbol{\theta}}} + \boldsymbol{J}_{\boldsymbol{u}}(\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}), \qquad \widehat{\boldsymbol{V}}_u(\boldsymbol{\theta}) \approx \widehat{\boldsymbol{V}}_u(\widehat{\boldsymbol{\theta}}), \tag{2.17}$$

where the Jacobian is

$$\boldsymbol{J}_{\boldsymbol{u}} = \frac{\partial}{\partial \boldsymbol{\theta}} \boldsymbol{u}_{\boldsymbol{\theta}} \bigg|_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}} = -\left[\left(\frac{\partial^2}{\partial \boldsymbol{u} \partial \boldsymbol{u}^T} G(\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \right)^{-1} \left(\frac{\partial^2}{\partial \boldsymbol{u} \partial \boldsymbol{\theta}^T} G(\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \right) \right] \bigg|_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}}. \tag{2.18}$$

Substituting the Taylor approximations in (2.17) into the Laplace conditional distribution (2.16), and combining it with the Normal hyperparameter marginal approximation (2.15), one obtains the jointly Normal Laplace approximation

$$\boldsymbol{u}, \boldsymbol{\theta} \mid \boldsymbol{y} \approx \mathcal{N}\left(\begin{pmatrix} \boldsymbol{u}_{\widehat{\boldsymbol{\theta}}} \\ \widehat{\boldsymbol{\theta}} \end{pmatrix}, \begin{pmatrix} \widehat{\boldsymbol{V}}_u(\widehat{\boldsymbol{\theta}}) + \boldsymbol{J}_{\boldsymbol{u}} \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \boldsymbol{J}_{\boldsymbol{u}}^T & \boldsymbol{J}_{\boldsymbol{u}} \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \\ \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \boldsymbol{J}_{\boldsymbol{u}}^T & \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \end{pmatrix} \right). \tag{2.19}$$

which we refer to henceforth as $\widetilde{p}(\boldsymbol{\theta}, \boldsymbol{u} \mid \boldsymbol{y})$. The marginal random effects posterior $\widetilde{p}(\boldsymbol{u} \mid \boldsymbol{y})$ arising from (2.19) is

$$\boldsymbol{u} \mid \boldsymbol{y} \approx \mathcal{N}\big(\boldsymbol{u}_{\widehat{\boldsymbol{\theta}}},\ \widehat{\boldsymbol{V}}_u(\widehat{\boldsymbol{\theta}}) + \boldsymbol{J}_{\boldsymbol{u}} \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \boldsymbol{J}_{\boldsymbol{u}}^T\big), \tag{2.20}$$

which coincides with the parametric empirical Bayes (PEB) approximation of Kass and Steffey (1989). In this manner, the computationally intensive two-step sampling scheme above can be replaced by the much faster method of sampling from the jointly Normal approximate posterior of hyperparameters and random effects (2.19).

2.2.4 Estimation of Return Levels

An important application of extreme weather modelling is to estimate the $p \times 100\%$ upper quantile of the extreme value distribution at a given location $\boldsymbol{x}$ (Coles, 2001). For a GEV model with parameters depending on $\boldsymbol{x}$, we denote this quantity by

$$\begin{aligned} z_p(\boldsymbol{x}) &= z_p\big(a(\boldsymbol{x}), b(\boldsymbol{x}), s(\boldsymbol{x})\big) \\ &= F^{-1}(1-p \mid a(\boldsymbol{x}), \exp(b(\boldsymbol{x})), \exp(s(\boldsymbol{x}))) \\ &= a(\boldsymbol{x}) - \frac{\exp(b(\boldsymbol{x}))}{\exp(s(\boldsymbol{x}))} \left\{1 - [-\log(1-p)]^{-\exp(s(\boldsymbol{x}))}\right\}, \end{aligned} \tag{2.21}$$

where $F^{-1}(\cdot \mid a, b_o, s_o)$ is the quantile function of the GEV distribution (2.1). With extreme annual rainfalls as an example, $z_p(\boldsymbol{x})$ is interpreted as the value above which the maximum precipitation level in a given year at location $\boldsymbol{x}$ occurs with probability p. Once such an event occurs, the expected time for the maximum precipitation to return to this level is $1/p$ years. For this reason, $z_p(\boldsymbol{x})$ is also called the *expected return level* for $1/p$ years. It is an important indicator of how extreme an event might be at a given location when p is chosen to be a small number.

In the context of the GEV-GP model, suppose we now wish to estimate $z_p(\boldsymbol{x}_i)$, $i = 1, \ldots, n$, for each of the n spatial locations in a given dataset. For problems where n is small to moderate, approximately sampling from each posterior return level distribution $p(z_p(\boldsymbol{x}_i) \mid \boldsymbol{y})$ can be achieved by first sampling from the approximate random effects posterior $\widetilde{p}(\boldsymbol{u} \mid \boldsymbol{y})$ in (2.20), then applying the transformation (2.21) to each resulting draw of $(a(\boldsymbol{x}_i), b(\boldsymbol{x}_i), s(\boldsymbol{x}_i))$. While this strategy works well when n is relatively small, for large n, sampling from $\widetilde{p}(\boldsymbol{u} \mid \boldsymbol{y})$ requires n^2 locations in memory to store the covariance matrix of the marginal posterior in (2.20). We will briefly discuss how to handle this problem below (Section 2.2.5).

2.2.5 Implementation

The calculation of the posterior mode $\widehat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \log \widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$ of the Laplace marginal approximation (2.14) is a nested optimization problem, with the inner optimization $\boldsymbol{u}_{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{u}} G(\boldsymbol{u}; \boldsymbol{\theta})$ being performed at each step of the outer optimization of $\widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$. Moreover, each step of the outer optimization problem requires the calculation of $\log |\boldsymbol{H}_{\boldsymbol{\theta}}|$, where $\boldsymbol{H}_{\boldsymbol{\theta}} = \partial^2 / (\partial \boldsymbol{u} \partial \boldsymbol{u}^T)\ G(\boldsymbol{u}_{\boldsymbol{\theta}}; \boldsymbol{\theta})$. While, in principle, this can be accomplished with any number of automatic differentiation (AD) programs, careful implementation is required for spatial random effects models due to the large size of $\boldsymbol{H}_{\boldsymbol{\theta}}$. Such an implementation is provided by the R/C++ library `TMB` (Kristensen et al., 2016).

Not only does `TMB` use the state-of-the-art sparse Cholesky methods of Chen et al. (2008) to calculate $\log |\boldsymbol{H}_{\boldsymbol{\theta}}|$ with low memory overhead, it also provides an efficient computation of $\partial / \partial \boldsymbol{\theta}\ \log \widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$, such that $\widehat{\boldsymbol{\theta}}$ can be efficiently obtained using gradient-based algorithms. Furthermore, `TMB` provides a Delta method calculation (described in Appendix A.1) to obtain element-wise posterior means and standard deviations of arbitrary parameter transformations $\boldsymbol{\eta}(\boldsymbol{\theta}, \boldsymbol{u}) = \big(\eta_1(\boldsymbol{\theta}, \boldsymbol{u}), \ldots, \eta_m(\boldsymbol{\theta}, \boldsymbol{u})\big)$, without having to store the $m \times m$ posterior covariance matrix of $\boldsymbol{\eta}(\boldsymbol{\theta}, \boldsymbol{u})$ in memory. This is particularly useful for estimating posterior return levels at all n spatial locations when n is large. We provide methods for fitting GEV-GP models via `TMB` in our R/C++ package `SpatialGEV`. Our package uses an implementation of the SPDE approximation discussed in Section 2.2.1 provided by the `R-INLA` package (Lindgren and Rue, 2015).

2.3 Simulation Study

In this section, two simulation studies – consisting of 400 and 6400 spatial locations, respectively – are carried out to demonstrate the speed and accuracy of our Laplace approximation. We compare the proposed approximation to two competing Bayesian inference methods. The first is MCMC on the joint posterior distribution of hyperparameters and spatial random effects, of which the covariance matrix is obtained from the SPDE approximation (2.6). We shall refer to this henceforth as the "true" posterior distribution. Since this quantity is unavailable in closed form, the accuracy of the Laplace approximation is benchmarked using an MCMC algorithm run until convergence.

Early works on GEV-GP models conduct Bayesian inference using Gibbs sampling on a handful of posterior random variables at a time (e.g., Cooley et al., 2007; Schliep et al., 2010; Reich and Shaby, 2012), but these methods tend to be inefficient when the latent variables are strongly intercorrelated. In this regard, more efficient block Gibbs samplers have

been developed (e.g., Knorr-Held and Rue, 2002; Geirsson et al., 2014, 2020). In particular, Jalbert et al. (2022) note that for GMRFs, conditionally independent spatial locations can be updated in parallel (e.g., Besag, 1974; Jalbert et al., 2017), thus leading to substantial computational accelerations when multiple cores with low communication overhead are available. However, these samplers still alternate between updating random effects and hyperparameters, and thus suffer from poor mixing when these quantities are highly correlated. In such situations, joint MCMC updates on all posterior random variables such as Hamiltonian Monte Carlo (HMC) (Duane et al., 1987) and elliptical slice sampling (Murray et al., 2010; Murray and Adams, 2010) typically offer faster convergence. In this chapter, the MCMC algorithm we employ is the No-U-Turn sampling (NUTS) variant of HMC (Hoffman and Gelman, 2014) as implemented in the R/C++ library `rstan` (Stan Development Team, 2020). This implementation is heavily optimized for speed and features adaptive hands-free tuning of the NUTS/HMC parameters. It thus serves to benchmark the Laplace approximation both in terms of accuracy against the true posterior, and in terms of speed compared to a state-of-the-art off-the-shelf MCMC sampler as provided by `rstan`.

The other Bayesian inference method to which we compare Laplace is the Max-and-Smooth method of Hrafnkelsson et al. (2021). Broadly speaking, this method replaces the log-likelihood term of each spatial location by a quadratic approximation – in the simplest case, using the MLE and Fisher information from only the observations at the given location. Inference for the random effects model is thus reduced to the much simpler "Normal-Normal" model, for which the random effects can be analytically integrated out. This principle, seemingly first noted in Daniels and Kass (1998), is commonly employed for pooling estimates in meta-analytic studies (e.g., Sutton and Abrams, 2001; Nam et al., 2003; Dias et al., 2018), and also appears in various applied studies involving hierarchical models (e.g. Daniels et al., 2000; Dominici et al., 2000; Coull et al., 2003; Behseta et al., 2005; Lysy et al., 2016). The Max-and-Smooth methodology provides a unified framework for these approaches with a focus on high-dimensional and complex random-effects structures.

In contrast to the Laplace approximation, Max-and-Smooth performs a single mode-quadrature calculation which can be parallelized across spatial locations, as opposed to making repeated (non-parallelizable) mode-quadrature calculations at each evaluation of the hyperparameter posterior (2.14). Thus, Max-and-Smooth stands to be much faster than Laplace. It has also been shown to be highly accurate for estimating spatial extreme value model parameters when the number of observations per location is sufficiently large (Hazra et al., 2021; Jóhannesson et al., 2022).

2.3.1 Small-Scale Study on Smooth Surfaces

This study consists of spatial locations on a 20×20 regular lattice on $[0, 10] \times [0, 10] \subset \mathbb{R}^2$, resulting in a total of $n = 400$ locations. The corresponding $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$ and $s(\boldsymbol{x}_i)$ are set deterministically via the functions

$$a(\boldsymbol{x}) = -0.5 \cdot \log\{\det(2\pi\boldsymbol{\Sigma}_0)\} - 0.5 \cdot (\boldsymbol{x} - \boldsymbol{\mu}_0)^T \boldsymbol{\Sigma}_0^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_0) + 80, \tag{2.22}$$

$$\begin{aligned} b(\boldsymbol{x}) =& 0.083 \cdot \log\left\{0.4 \cdot \det(2\pi\boldsymbol{\Sigma}_1)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_1)^T \boldsymbol{\Sigma}_1^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_1)\right)\right. \\ &\left. + 0.6 \cdot \det(\boldsymbol{\Sigma}_2)^{-\frac{1}{2}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_2)^T \boldsymbol{\Sigma}_2^{-1} (\boldsymbol{x} - \boldsymbol{\Sigma}_2)\right)\right\} + 3.33, \end{aligned} \tag{2.23}$$

$$s(\boldsymbol{x}) = -0.1 \cdot \log\{\det(2\pi\boldsymbol{\Sigma}_3)\} - 0.1 \cdot (\boldsymbol{x} - \boldsymbol{\mu}_3)^T \boldsymbol{\Sigma}_3^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_3), \tag{2.24}$$

where

$$\boldsymbol{\mu}_0 = \begin{pmatrix} 3 \\ 3 \end{pmatrix}, \ \boldsymbol{\Sigma}_0 = \begin{pmatrix} 3 & 0 \\ 0 & 3 \end{pmatrix}, \ \boldsymbol{\mu}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \ \boldsymbol{\Sigma}_1 = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix},$$
$$\boldsymbol{\mu}_2 = \begin{pmatrix} 8 \\ 7 \end{pmatrix}, \ \boldsymbol{\Sigma}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \ \boldsymbol{\mu}_3 = \begin{pmatrix} 4 \\ 4 \end{pmatrix}, \ \boldsymbol{\Sigma}_3 = \begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}.$$

Heat maps for $a(\boldsymbol{x})$, $b(\boldsymbol{x})$ and $s(\boldsymbol{x})$ on the spatial domain are displayed in Figure 2.1. Values of the GEV parameters are chosen so that the simulated observations mimic extreme

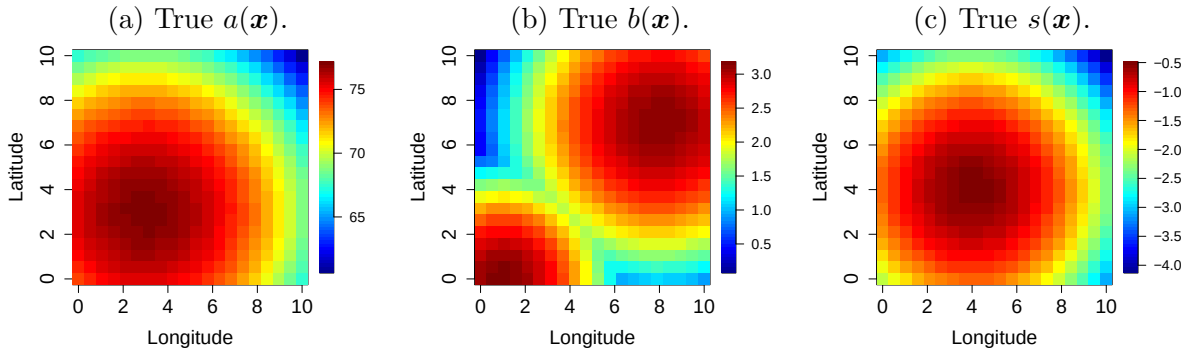


Figure 2.1: The true $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$ and $s(\boldsymbol{x}_i)$ for the small-scale simulation study plotted on regular lattices.

daily precipitation levels in millimetres (Cooley and Sain, 2010). In particular, the range of the shape parameters is $(0.016, 0.6)$, which guarantees that the GEV distribution at each location has a finite mean. At each location $\boldsymbol{x}_i$, n_i observations are simulated from

$\text{GEV}(a(\boldsymbol{x}_i), e^{b(\boldsymbol{x}_i)}, e^{s(\boldsymbol{x}_i)})$ with n_i drawn from a discrete Uniform$(20, 50)$. No covariates are included in the mean functions of the spatial Gaussian processes, so each GP has one fixed intercept parameter in the mean function: β_a, β_b, and β_s. We specify weakly informative Normal priors on the GP mean parameters: $\beta_a \sim \mathcal{N}(0, 100^2)$, $\beta_b \sim \mathcal{N}(0, 50^2)$, and $\beta_s \sim \mathcal{N}(0, 20^2)$. Flat priors are assumed for the hyperparameters of the GP covariance functions. We also tried the PC prior of Simpson et al. (2017) for these but found the difference in parameter estimation to be negligible.

This simulation study compares five different Bayesian estimators: HMC-NUTS on the joint posterior, and two different estimators for each of Laplace and Max-and-Smooth. For Laplace, the first of these is the jointly Normal approximation $\widetilde{p}(\boldsymbol{\theta}, \boldsymbol{u} \mid \boldsymbol{y})$ in (2.19), which is based on the mode and quadrature of its marginal hyperparameter distribution $\widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$ in (2.14). A jointly Normal mode-quadrature approximation is constructed analogously for Max-and-Smooth. For Laplace, we also consider an estimator which samples from its exact posterior distribution $\widehat{p}(\boldsymbol{\theta}, \boldsymbol{u} \mid \boldsymbol{y})$, i.e., without the jointly Normal approximation. To do this, we first sample from $\widehat{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$ in (2.14) using MCMC. Then to each resulting draw of $\boldsymbol{\theta}_i$, we combine a draw $\boldsymbol{u}_i$ from the Laplace conditional Normal distribution $\widetilde{p}(\boldsymbol{u} \mid \boldsymbol{\theta}, \boldsymbol{y})$ in (2.16), following the two-stage sampling approach described in Section 2.2.3. Once again, the analogous exact posterior estimator is constructed for Max-and-Smooth. We shall refer to these five estimators as HMC-NUTS, Laplace-MQ, Laplace-MCMC, MaxSmooth-MQ, and MaxSmooth-MCMC.

For the HMC-NUTS method, six Markov chains were run in parallel for 4000 iterations per chain, after an initial 4000 "warm-up" steps per chain to determine the algorithm's tuning parameters. The effective sample sizes for all fixed and random effects range between 629 and $38,223$, with a mean of $18,065$. We checked that all chains have mixed well based on the $\widehat{R} < 1.01$ convergence metric advocated in Vehtari et al. (2021).

For the Max-and-Smooth methods, following Jóhannesson et al. (2022), the Max step at each location was performed using a generalized likelihood estimator (Martins and Stedinger, 2000). That is, we maximize the log-posterior of the GEV likelihood defined by (2.1), in terms of our restricted parameterization $(a, b = \log b_o, s = \log s_o)$, combined with a prior $s \sim \mathcal{N}(0, 100^2)$ that prevents the mode from being on boundary of the parameter space $s_o = 0$.

For the Laplace-MCMC and MaxSmooth-MCMC methods, the MCMC algorithms used to explore the marginal hyperparameter posteriors were multivariate Normal random walks with variances proportional to the marginal variance of the corresponding mode-quadrature approximation. For each of these estimators, six chains were run for 40,000 iterations each, following 5,000 steps of burn-in.

Posterior distributions of the hyperparameters for each of the five Bayesian estimators are shown in Figure 2.2. Figures 2.2(a) and 2.2(d) show that the two mode-quadrature

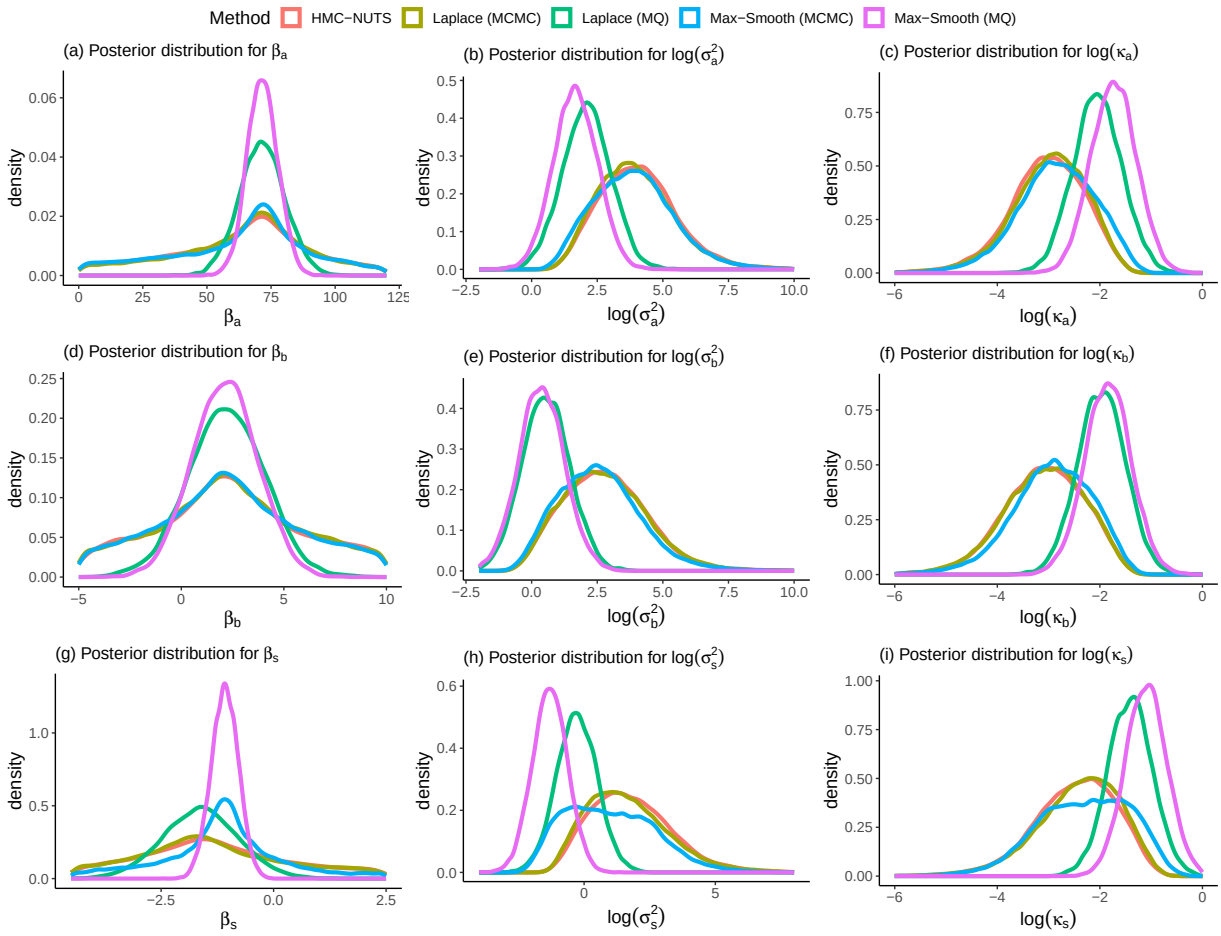


Figure 2.2: Posterior distributions of the hyperparameters $\boldsymbol{\theta}$ in the small-scale simulation study.

(MQ) methods Laplace-MQ and MaxSmooth-MQ have the same mode of β_a and β_b as the true posterior obtained from HMC-NUTS. The middle and right columns of Figure 2.2 show that the MQ methods underestimate the means of the variance parameters $(\sigma_a^2, \sigma_b^2, \sigma_s^2)$ and overestimate the means of the inverse range parameters $(\kappa_a, \kappa_b, \kappa_s)$ compared to HMC-NUTS. Moreover, the MQ posteriors for all hyperparameters are much narrower than the corresponding true posteriors. In contrast, the Laplace-MCMC and MaxSmooth-MCMC estimators are extremely accurate for all hyperparameters except β_s.

While the Laplace-MQ and MaxSmooth-MQ posteriors are generally quite similar, the former is always closer to the true posterior. The difference between the two is most pronounced for β_s, the overall mean of the shape parameter, for which Laplace-MQ is markedly superior. In fact, Laplace-MQ is considerably closer to the true posterior than even MaxSmooth-MCMC in this case.

Figure 2.3 displays the posterior means of the random-effects parameters for each of the five Bayesian estimators against their true values displayed in Figure 2.1. The Laplace-MQ method underestimates the location parameter and correspondingly overestimates the shape parameter at some locations. Further investigations (presented in Appendix A.2) reveal that the locations at which the point estimates of $a(\boldsymbol{x})$ and $s(\boldsymbol{x})$ deviate most from the true values are those lying on the boundary of the spatial domain, and where true boundary values differ substantially from neighbouring values closer to the middle of the domain. The estimation errors are much worse for the shape parameter than for the location parameter, with the former known to be particularly difficult to estimate (e.g., Schliep et al., 2010; Cooley and Sain, 2010). That being said, the Laplace-MQ posterior mean estimates of the GEV parameters are almost identical to those of the true posterior distribution obtained by HMC-NUTS, exhibiting no more than a 1-5% difference almost invisible in Figure 2.3.

The MaxSmooth-MQ estimates of the location and shape parameters exhibit a similar spatial pattern to those of Laplace-MQ, with the largest error occurring around the boundary and corners of the map. However, the estimates of the shape parameters are substantially worse than those of Laplace-MQ, and are not much improved with the exact posterior estimator MaxSmooth-MCMC.

We now consider estimating the return levels $z_p(\boldsymbol{x})$, an important meteorological quantity of interest. At each of the $n = 400$ spatial locations, the 10-year return level $z_{10}(\boldsymbol{x})$ is estimated by its posterior mean, which is computed using the simulation-based method described in Section 2.2.4. For the Laplace-MQ and MaxSmooth-MQ methods, this is achieved by first drawing $n_{\text{sim}} = 10,000$ samples from their respective multivariate Normal approximations to $p(\boldsymbol{u} \mid \boldsymbol{y})$, then transforming each draw to the corresponding return level via (2.21). For the three MCMC estimators, the procedure is identical except using the MCMC draws of $\boldsymbol{u}$.

The rightmost column of Figure 2.3 displays the estimated return levels against their true values. The HMC-NUTS estimator and the two Laplace estimators all exhibit very similar results. Despite overestimating the shape parameter $s(\boldsymbol{x})$ at its smallest values, the estimates of $z_{10}(\boldsymbol{x})$ by the three aforementioned methods are generally quite accurate. This suggests that some degree of bias in estimating the shape parameters has only a modest

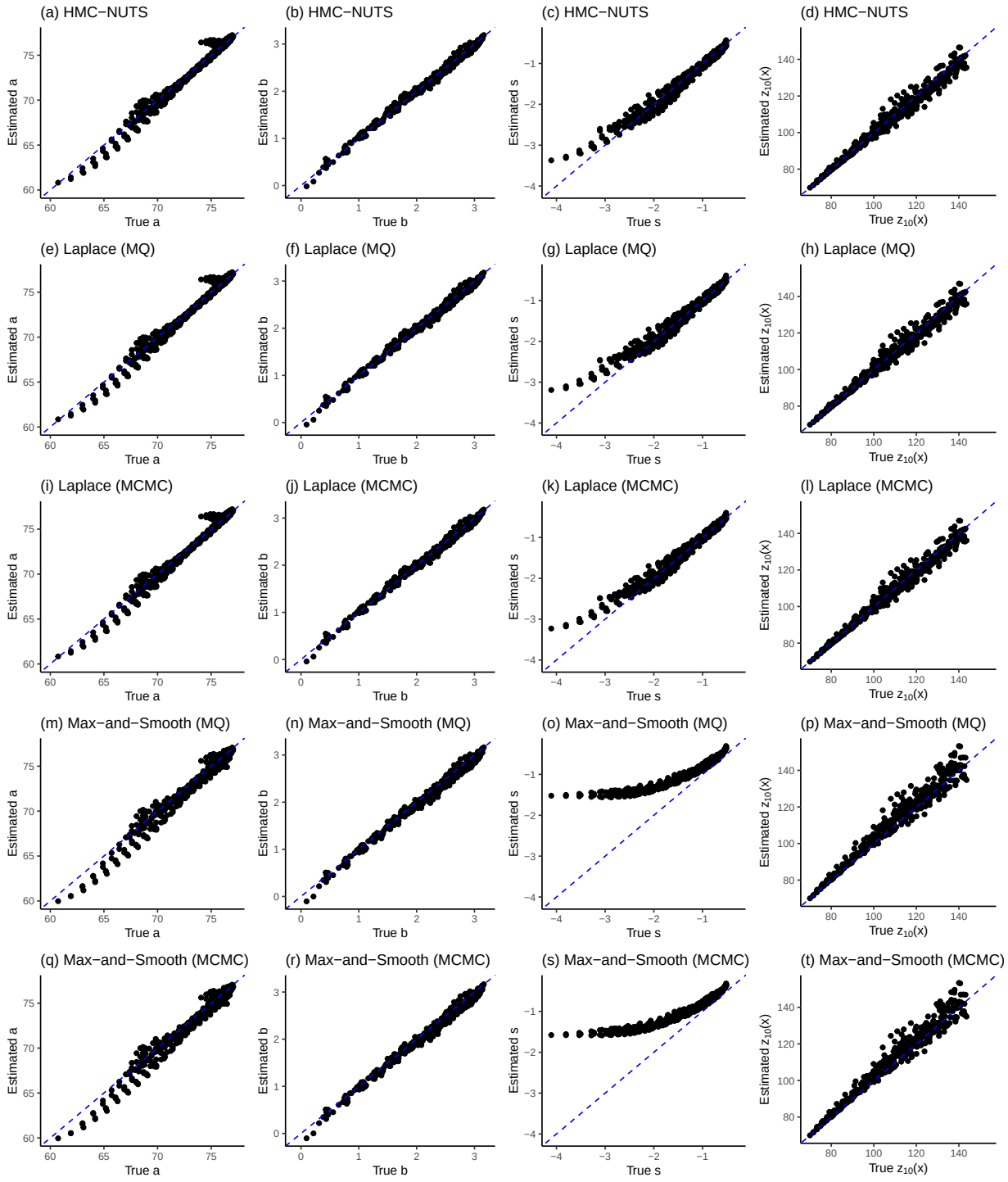


Figure 2.3: True GEV location parameters $a(\boldsymbol{x})$, scale parameters $b(\boldsymbol{x})$, shape parameters $s(\boldsymbol{x})$, and 10-year return levels $z_{10}(\boldsymbol{x})$ plotted against their estimates obtained using MCMC, Max-and-Smooth, and Laplace.

effect on the marginal quantile estimates when the true shape parameter is close to zero on the original scale.

The return level estimates of the two Max-and-Smooth estimators are also quite accurate, though somewhat less so than the other three estimators at larger return levels. This is due to the shape parameter being substantially overestimated in the top right corner of Figure 2.1, which is also where larger values of the scale parameter $b(\boldsymbol{x})$ occur, and thus also of the true return levels $z_{10}(\boldsymbol{x})$.

Table 2.1 provides a numerical summary of these results, reporting the mean absolute errors (MAEs) for $a(\boldsymbol{x})$, $b(\boldsymbol{x})$, $s(\boldsymbol{x})$ and $z_{10}(\boldsymbol{x})$ at the $n = 400$ spatial locations. The MAEs for the two Laplace methods differ by no more than 10% from those of HMC-NUTS. In contrast, the MAEs of MaxSmooth-MQ and MaxSmooth-MCMC are 1.4 to 3.9 times larger than those of HMC-NUTS.

While the Laplace and Max-and-Smooth approximations have the same order of error in terms of the number of observations per spatial location (Daniels and Kass, 1998), it is reasonable to expect that the former is more accurate. This is because the Laplace approximation employs a mode-quadrature approximation to $p(\boldsymbol{u} \mid \boldsymbol{\theta}, \boldsymbol{y})$ individually calibrated to each value of $\boldsymbol{\theta}$, instead of a single mode-quadrature approximation calibrated to only the data-level likelihood.

A well-known shortcoming of the Laplace approximation (e.g., Kass and Steffey, 1989) is that, while it typically gives good estimates of the posterior mean of the random effects, it often underestimates their posterior variance. To assess this, Figure 2.4 examines the uncertainty estimates provided by Laplace-MQ. The top row of Figure 2.4 shows the posterior standard deviations of the three GEV parameters at each point on the regular lattice. More uncertainty is observed on the boundary – especially at the corners – of the spatial domain. The bottom row of Figure 2.4 displays the ratio between the posterior standard deviation of Laplace-MQ and that of HMC-NUTS. These plots indicate that Laplace-MQ accurately captures the posterior uncertainty in this simulation setting, with all ratios being in the range of 0.95-1.05.

A summary measure of the accuracy of the posterior distributions for $z_{10}(\boldsymbol{x})$ for each of the Laplace and Max-and-Smooth estimators is given by the $\mathrm{KS}\left(\widehat{z}_{10}(\boldsymbol{x})\right)$ metric in Table 2.1. This metric is obtained by first computing the two-sample Kolmogorov-Smirnov test statistic (KS) between the posterior draws of $z_{10}(\boldsymbol{x}_i)$ from HMC-NUTS and those of the given estimator at each location $i = 1, \ldots, 400$; then, we average these $n = 400$ test statistics for each estimator to get $\mathrm{KS}\left(\widehat{z}_{10}(\boldsymbol{x})\right)$. The more similar are the posterior distributions of $z_{10}(\boldsymbol{x}_i)$, the smaller the values of $\mathrm{KS}\left(\widehat{z}_{10}(\boldsymbol{x})\right)$. By this metric, the Laplace posteriors for $z_{10}(\boldsymbol{x})$ are much more similar to the true posteriors produced by HMC-NUTS

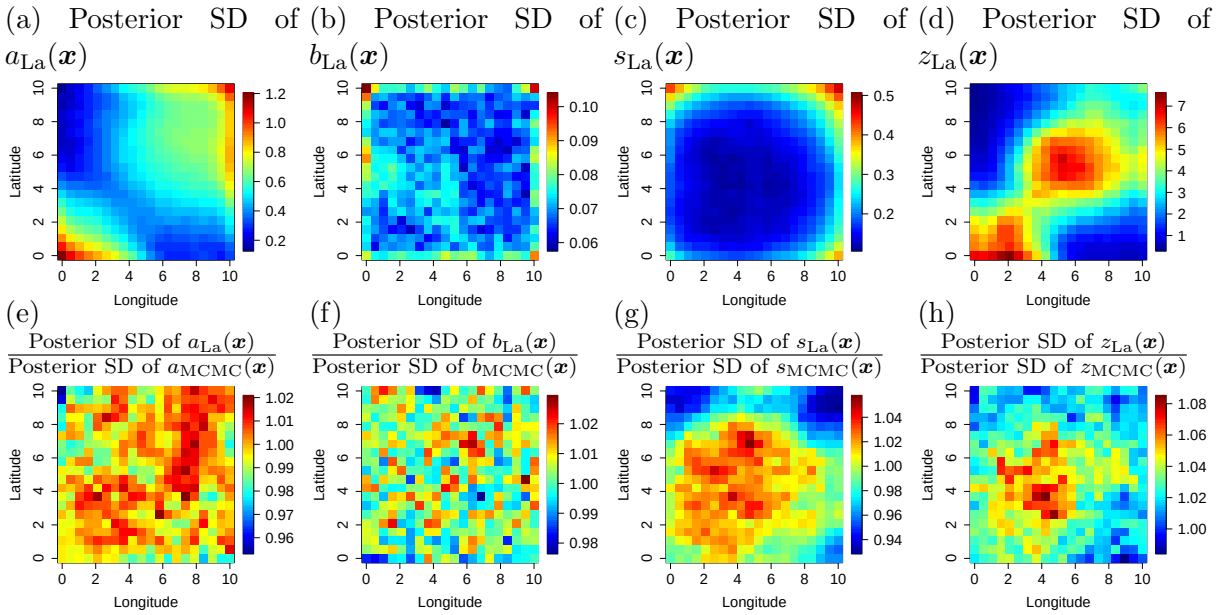


Figure 2.4: Upper panels: Standard deviations of posterior samples of $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$, $s(\boldsymbol{x}_i)$ and $z_{10}(\boldsymbol{x})$ from Laplace-MQ plotted on the regular lattice. Lower panels: Ratios between the posterior standard deviations obtained from Laplace-MQ and those from HMC-NUTS.

than those of Max-and-Smooth.

The final metric in Table 2.1 is the mean computation time for each estimator and its standard error. The latter is calculated from 20 replications of the full estimation procedure for Laplace-MQ and MaxSmooth-MQ, and from the six parallel chains run for each of Laplace-MCMC, MaxSmooth-MCMC, and HMC-NUTS. All computations were carried out using `R` version 3.6.2 on a computer cluster with six 2.7GHz Xeon(R) CPUs (E5-4650) and 1GB memory per CPU. The Max step of Max-and-Smooth was parallelized over the six cores, whereas the Smooth step of MaxSmooth-MQ, as well as the entirety of the Laplace-MQ calculations, were computed on a single core. The runtimes in Table 2.1 indicate that MCMC on the approximate marginal posteriors of Laplace or Max-and-Smooth is considerably faster than on the full joint posterior targeted by HMC-NUTS. MaxSmooth-MCMC is twice as fast as Laplace-MCMC, and would thus be preferable in a setting with enough observations at each spatial location to reliably compute the Max step. However, both MaxSmooth-MCMC and Laplace-MCMC are orders of magnitude slower than their MQ counterparts. While the MQ approximations produce biased estimates of the hyperparameters posteriors, Laplace-MQ estimates the random effects and return level posterior distributions with extremely high accuracy.

		Laplace		Max-and-Smooth	
	HMC-NUTS	MQ	MCMC	MQ	MCMC
MAE $\left(\widehat{a}(\boldsymbol{x})\right)$	0.384	0.384	0.383	0.603	0.602
MAE $\left(\widehat{b}(\boldsymbol{x})\right)$	0.048	0.051	0.050	0.076	0.076
MAE $\left(\widehat{s}(\boldsymbol{x})\right)$	0.102	0.111	0.107	0.487	0.483
MAE $\left(\widehat{z}_{10}(\boldsymbol{x})\right)$	2.262	2.192	2.186	3.136	3.067
KS $\left(\widehat{z}_{10}(\boldsymbol{x})\right)$	0	0.031	0.027	0.341	0.340
Mean (SD)	-	(0.013)	(0.012)	(0.180)	(0.173)
Runtime	225,091s	18s	70,532s	14s	36,011s
Mean (SD)	(9110s)	(0.2s)	(114s)	(0.1s)	(116s)

Table 2.1: Comparison between HMC-NUTS, Laplace (MQ/MCMC), and Max-and-Smooth (MQ/MCMC). The MAE statistic is computed as $\text{KS}\left(\widehat{u}(\boldsymbol{x})\right) = \frac{\sum_{i=1}^{n}|u(\boldsymbol{x}_i)-\widehat{u}(\boldsymbol{x}_i)|}{n}$, where $u \in \{a, b, s\}$. $\text{KS}\left(\widehat{z}_{10}(\boldsymbol{x})\right)$ is the value of the two-sample KS test statistic between the HMC-NUTS and each other posterior for $z_{10}(\boldsymbol{x}_i)$ averaged over all spatial locations $i = 1, \ldots, 400$.

2.3.2 Large-Scale Study on Rough Surfaces

The simulation study above has examined the performance of the proposed Laplace method when the GEV parameters are all simulated from relatively smooth surfaces. Here, we consider a more difficult case in which the parameter surfaces are more "patchy" (as shown in Figure 2.5) and where the number of locations is an order of magnitude larger than the previous simulation study. The parameters $a(\boldsymbol{x})$, $b(\boldsymbol{x})$, and $s(\boldsymbol{x})$ are simulated from Gaussian random fields with an exponential covariance kernel function on an 80×80 regular lattice on $[0, 60] \times [0, 60] \subset \mathbb{R}^2$:

$$a(\boldsymbol{x}) \sim \mathcal{GP}\big(70,\ k(\boldsymbol{x}, \boldsymbol{x}' \mid 8, 20)\big) \tag{2.25}$$

$$b(\boldsymbol{x}) \sim \mathcal{GP}\big(a(\boldsymbol{x})/10 - 4,\ k(\boldsymbol{x}, \boldsymbol{x}' \mid 0.01, 20)\big) \tag{2.26}$$

$$s(\boldsymbol{x}) \sim \mathcal{GP}\big(-a(\boldsymbol{x})/10 + 5,\ k(\boldsymbol{x}, \boldsymbol{x}' \mid 0.01, 30)\big), \tag{2.27}$$

where $k(\boldsymbol{x}, \boldsymbol{x}' \mid \sigma, \lambda)$ is the Matérn covariance kernel (2.3) with shape parameter $\nu = 1$ and scale and range parameters σ and $\lambda = 1/\kappa$. Simulation is carried out using the `rgp()` function in the `SpatialExtremes` package (Ribatet et al., 2022). The other aspects of the simulation study – e.g., data generation mechanism, prior specification, and other modelling setups – are the same as those in Section 2.3.1.

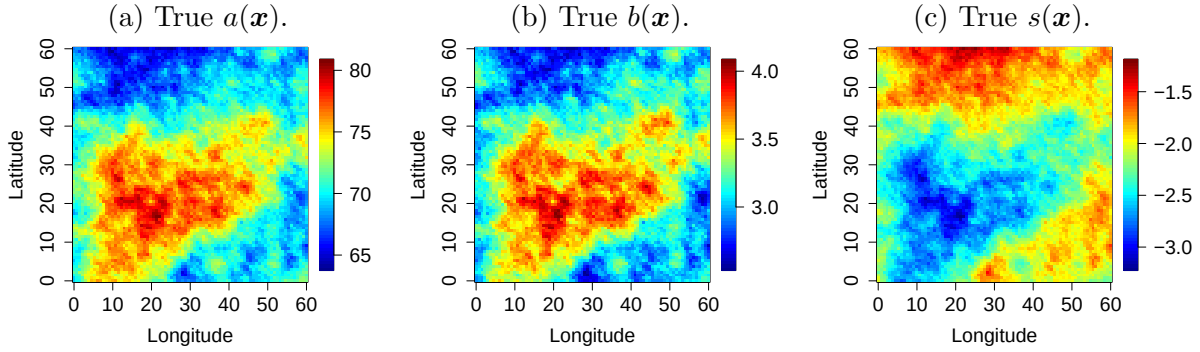


Figure 2.5: The true $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$ and $s(\boldsymbol{x}_i)$ for the large-scale simulation study simulated from three Gaussian random fields.

In this simulation, $\boldsymbol{u}$ consists of three random effects at each of the $n = 6400$ spatial locations, resulting in a total of 19200 random effects. Here we only consider the Laplace-MQ estimator, which took 22 hours to fit on the same computer cluster as used in Section 2.2.5 but with 90Gb of memory. As it is computationally infeasible to store and

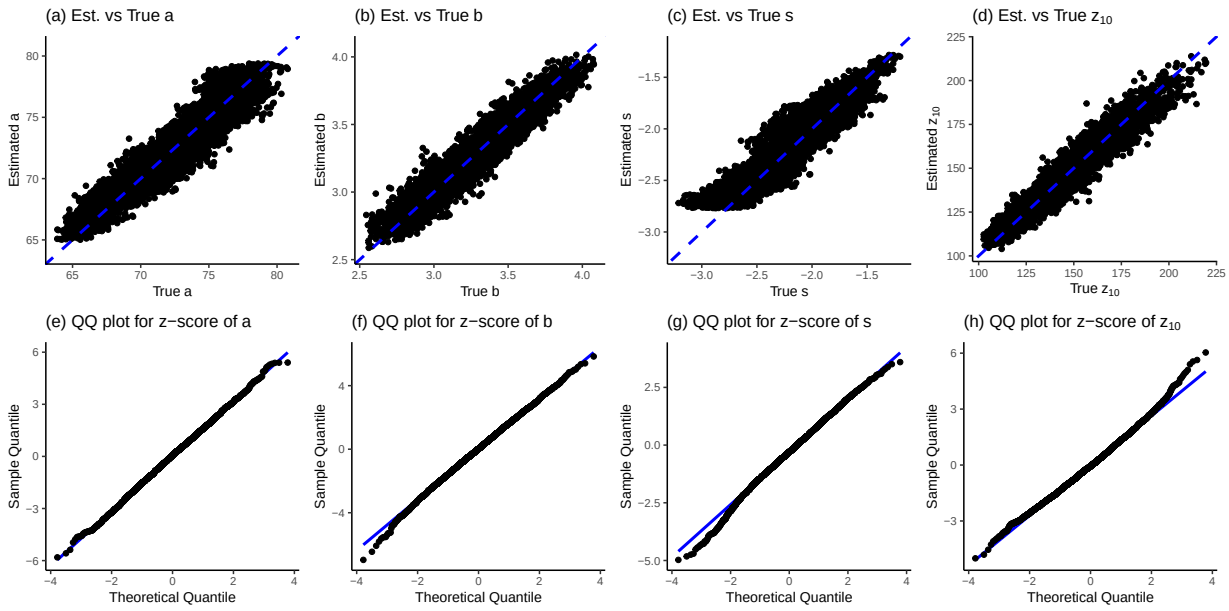


Figure 2.6: (a)-(d): True GEV location parameters $a(\boldsymbol{x})$, scale parameters $b(\boldsymbol{x})$, shape parameters $s(\boldsymbol{x})$, and 10-year return levels $z_{10}(\boldsymbol{x})$ plotted against their estimates obtained using the Laplace method when the true parameter surfaces are unsmooth. (e)-(f): Q-Q plots for the Z-scores corresponding to $a(\boldsymbol{x})$, $b(\boldsymbol{x})$, $s(\boldsymbol{x})$, and $z_{10}(\boldsymbol{x})$.

invert the 19200×19200 dimensional covariance matrix to sample from $p(\boldsymbol{u} \mid \boldsymbol{y})$, the Delta method is used to compute only the posterior mean and standard deviation of $z_{10}(\boldsymbol{x}_i)$ at each spatial location, as discussed in Section 2.2.5 and Appendix A.1. As a point of reference, obtaining 100 iterations per parallel chain for HMC-NUTS– not nearly enough to converge to the true posterior distribution – took over a week. While we have not obtained a runtime estimate for Laplace-MCMC, the timings in Table 2.1 suggest it too would be infeasible to run till convergence with the $n = 6400$ locations considered here.

Figure 2.6 (a)-(d) compare the posterior mean estimates of all GEV parameters and the 10-year return levels from the Laplace method to their true values. The overall trends of both the location parameter $a(\boldsymbol{x})$ and scale parameter $b(\boldsymbol{x})$ are captured well by the corresponding posterior mean estimates. The shape parameter tends to be slightly overestimated when the true value is small, which has been also observed in Figure 2.3(c). However, as in the small-scale simulation study, this does not have much of an effect on the estimates of $z_{10}(\boldsymbol{x})$, which align well with the true values at the majority of the locations.

We now evaluate the accuracy of the uncertainty estimates of Laplace-MQ. Since we do not have access to the true posterior distribution for the large-scale simulation, accuracy is assessed by computing at each location i a z-score of the form

$$\mathcal{Z}(\gamma_i) = \frac{\gamma_i - \hat{\gamma}_i}{\widehat{\text{sd}}(\gamma_i)}, \tag{2.28}$$

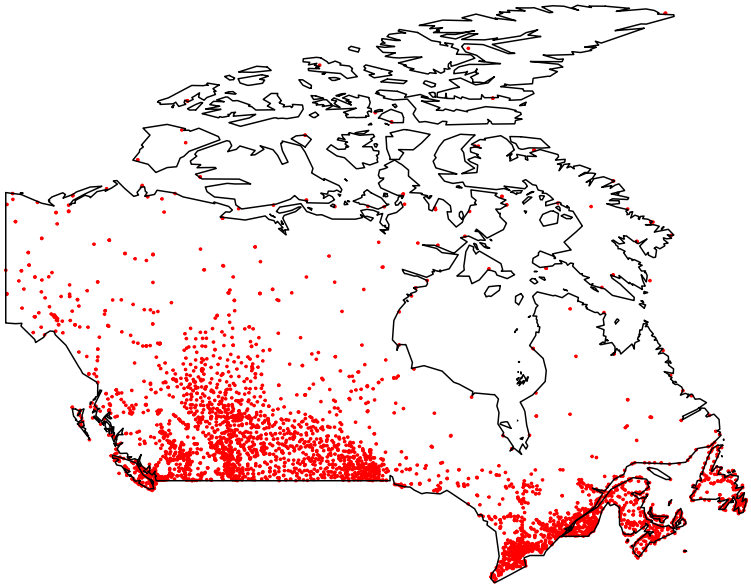
where γ_i is the true value at location i of the quantity $\gamma \in \{a, b, s, z_{10}\}$, and $\hat{\gamma}_i$ and $\widehat{\text{sd}}(\gamma_i)$ are the Laplace-MQ estimates of its posterior mean and standard deviation. If the posterior uncertainty is accurately estimated by Laplace-MQ, then these z-scores should be close to the 45° line on a QQ plot. Such QQ plots are displayed in Figure 2.6 (e)-(h). The true values of b and s are further than expected in the lower tail of the Laplace-MQ posterior, and those of z_{10} are further than expected in the upper tail. However, the corresponding departures from the 45° line only happen for z-scores of magnitude more than two, meaning that the Laplace-MQ posteriors cover about 95% of the true underlying values. These results demonstrate both the computational feasibility of the Laplace-MQ method and its ability to accurately estimate GEV parameters and return levels at several thousand spatial locations.

2.4 Case Study

Here, we study a real data set of monthly total snowfall in Canada from January 1987 to December 2021, which is publicly available through Environment and Climate Change Canada (2022). The monthly total snowfall at a location is the amount of frozen precipitation (in cm), including snow and ice pellets, observed throughout a month. The goal of this study is to produce a cross-country map of 10-year return levels $z_{10}(\boldsymbol{x})$ of extreme monthly total snowfall at the observed locations.

The dataset includes 3833 weather stations plotted in Figure 2.7(a). In Figure 2.7(b), the data are gridded into $1° \times 1°$ cells. We only include cells in which there are at least 10 years of observations, and in line with extreme value theory, use the maximum yearly records of monthly snowfall in each cell as the response values. This results $n = 509$ spatial locations (grid cells) each with at least 10 extreme value observations. The marginal distribution of these extreme value observations at $n = 509$ locations is plotted in Figure 2.8. We see that this distribution is heavy-tailed to the right, and thus a good candidate for modelling via the GEV distribution with positive shape parameter. No additional covariates are included in the model. The triangulated mesh for SPDE approximation is superimposed

(a) All locations



(b) Gridded locations on the mesh

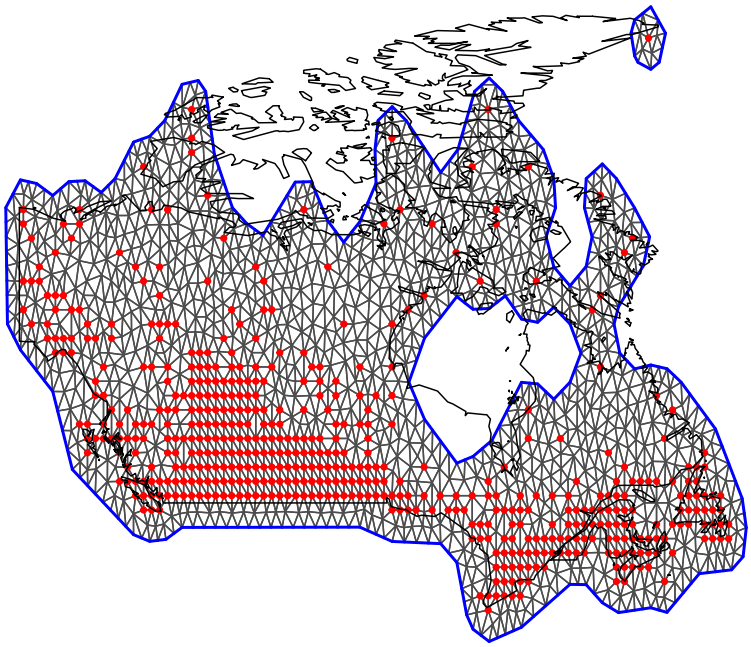


Figure 2.7: (a): All 3833 locations at which data was collected from 1987 to 2021. (b): 509 locations at which at least 10 years of data was recorded after gridding.

on the map in Figure 2.7(b). A non-convex boundary is built for the mesh, leaving out a hole in the region of Hudson Bay due to the physical boundary it creates in the spatial domain. This results in $n' = 1565$ nodes, each of which will have separate estimated GEV random effects. Prediction at unobserved locations is straightforward using the Gaussian process methodology (Rasmussen and Williams, 2006) described in Appendix A.3.

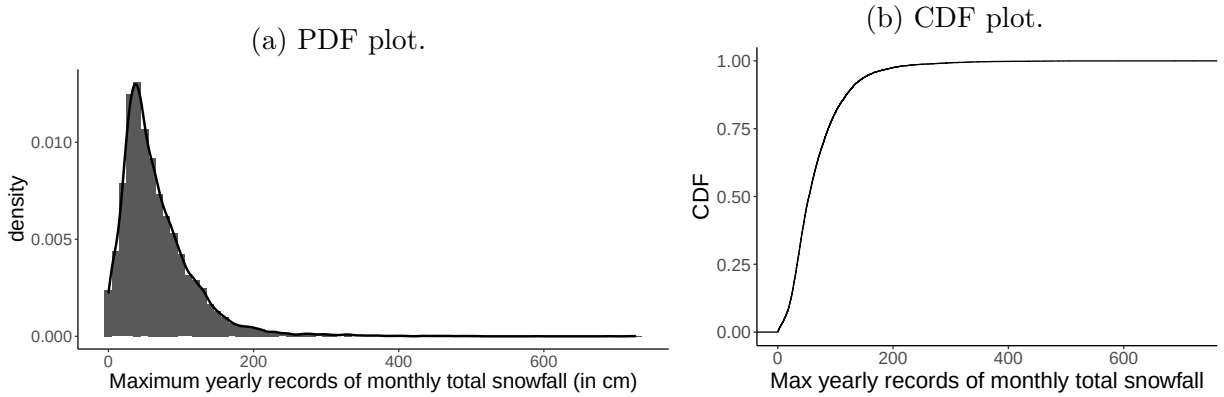


Figure 2.8: PDF (left) and CDF (right) of all $n = 509$ extreme value observations pooled across locations and time.

2.4.1 Model Selection

The Laplace-MQ method is first used to fit a GEV-GP model with all GEV parameters as spatial random effects, which is denoted by M_{abs}. The point estimates $\widehat{s}(\boldsymbol{x})$ at all locations have a small range from -8.1 to -8.0, suggesting that the model with spatially varying $s(\boldsymbol{x})$ might not be needed. Therefore, we reduce the model complexity by having the shape parameter as a fixed effect s across space, and denote this model by M_{ab}. A Normal prior $\mathcal{N}(-5, 5)$ is imposed on s. Furthermore, we compare model M_{ab} to the simplest spatial extreme model M_{a} with only $a(\boldsymbol{x})$ spatially varying. The model with random scale parameter and fixed location and shape parameters, i.e., M_{b}, is not considered here as the spatial variation of $a(\boldsymbol{x})$ contributes greatly to the observation variation across different locations.

The fit of models M_{ab} and M_{a} to the snowfall dataset is evaluated using a model checking procedure described below. The posterior predictive distribution of maximum

monthly total snowfall $y_\star$ at spatial location $\boldsymbol{x}_\star$ is given by

$$p(y_\star \mid \boldsymbol{y}) = \int p\big(y_\star \mid a(\boldsymbol{x}_\star), b(\boldsymbol{x}_\star), s(\boldsymbol{x}_\star)\big) \times p\big(a(\boldsymbol{x}_\star), b(\boldsymbol{x}_\star), s(\boldsymbol{x}_\star) \mid \boldsymbol{y}\big)\, \mathrm{d}a(\boldsymbol{x}_\star)\, \mathrm{d}b(\boldsymbol{x}_\star)\, \mathrm{d}s(\boldsymbol{x}_\star). \tag{2.29}$$

Agreement between model and data can be measured by computing a chosen test statistic $T_{\text{rep}} = T(y_{\text{rep}})$, where $y_{\text{rep}} \sim p(y_\star \mid \boldsymbol{y})$ are replicated data drawn from the posterior predictive distribution (2.29), and comparing it to its empirical counterpart $T_{\text{obs}} = T(y_{\text{obs}})$ computed from the observed data $y_\star = y_{\text{obs}}$. If the model is correctly specified, then $T_{\text{rep}} \approx T_{\text{obs}}$. The calculations of the posterior predictive distribution (2.29) and T_{rep} are detailed in Appendix A.3.

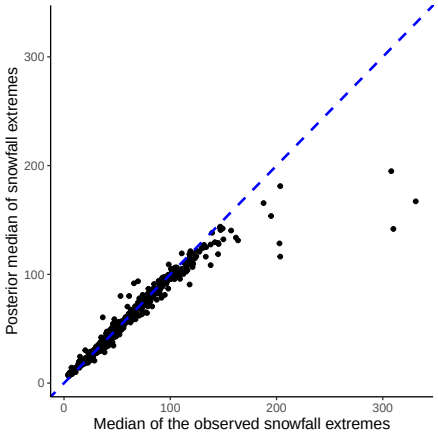
The test statistics chosen here are the median and the 10% upper quantile, i.e. the 10-year return rate. Figure 2.9 thus plots the median and 10% upper quantiles of the posterior predictive distribution (2.29) (T_{rep}) for both models M_{ab} and M_{a} versus their sample values in the observed data at each of the $n = 509$ locations (T_{obs}). Lack of fit is indicated by departures from the 45° line. Reading the plots, it is clear that model M_{ab} outperforms M_a. This observation underscores the importance of having the ability to model more than just the location parameter of the GEV distribution as spatially varying, which cannot be accomplished using existing INLA implementations.

The posterior predictive checks in Figure 2.9 can also be used to evaluate the goodness-of-fit of the GEV-GP model and its Laplace approximation. We see excellent agreement in the median and upper 10% quantiles between the actual snowfall values and the corresponding predictions of model M_{ab}, except at the 10-20 locations with the largest snowfall, where it is systematically underestimated. Consequently, the remaining analyses in this section will be performed with model M_{ab}.

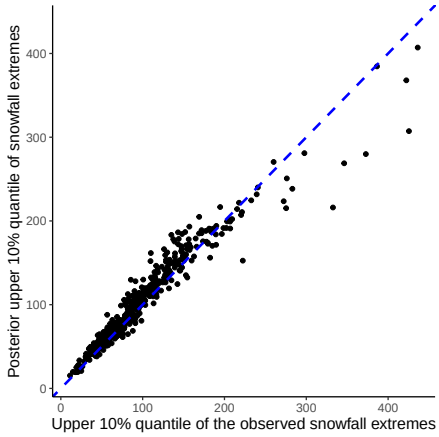
2.4.2 Parameter Inference

The posterior mean estimate of s is -7.7 with a posterior standard deviation of 1.54. While there is some evidence in the simulation studies that the Laplace approximation tends to overestimate s at small values (Figures 2.3(g) and 2.5c), this only suggests that the true posterior estimate of s could be even smaller. Hence, it would appear that the conditional (i.e., spatially-varying) GEV distribution fitted to these data is in fact light-tailed, thus essentially reducing to a Gumbel distribution (e.g., Youngman, 2022). This stands in contrast to the heavy-tailed marginal distribution of the extreme value observations shown in Figure 2.8.

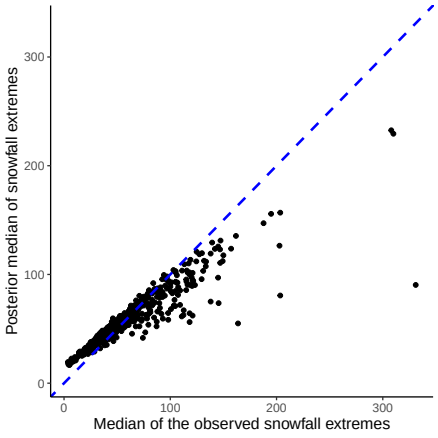
(a) Model M_{ab}: random $a(\boldsymbol{x})$ and $b(\boldsymbol{x})$ Test quantity: median.



(b) Model M_{ab}: random $a(\boldsymbol{x})$ and $b(\boldsymbol{x})$. Test quantity: upper 10% quantile



(c) Model M_{a}: random $a(\boldsymbol{x})$, fixed b. Test quantity: median



(d) Model M_{a}: random $a(\boldsymbol{x})$, fixed b. Test quantity: upper 10% quantile

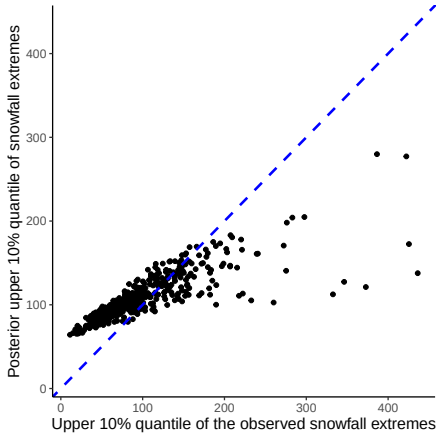
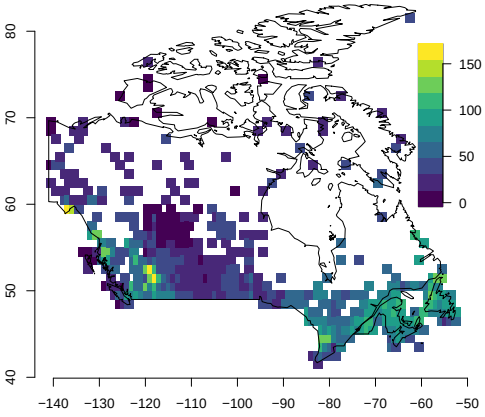
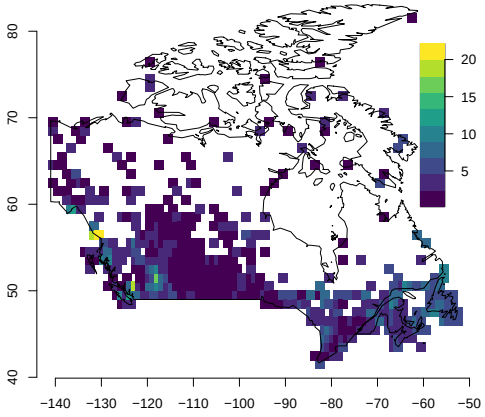


Figure 2.9: Posterior predicted versus empirical test statistics. Each dot corresponds to one of the $n = 509$ spatial locations. Upper panel: Model with both $a(\boldsymbol{x})$ and $b(\boldsymbol{x})$ as spatial random effects. Lower panel: Model with only $a(\boldsymbol{x})$ random. Left panel: Predicted medians versus sample medians. Right panel: Predictive upper 10% quantiles versus sample upper 10% quantiles.

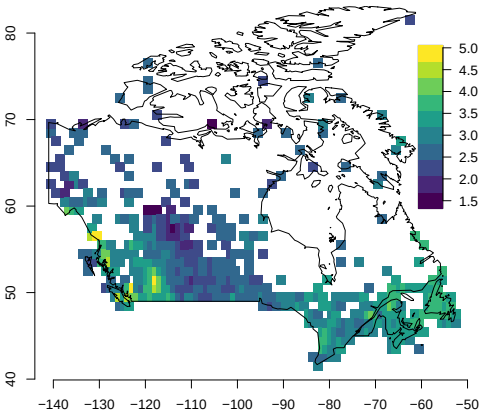
(a) Posterior mean of $a(\boldsymbol{x})$.



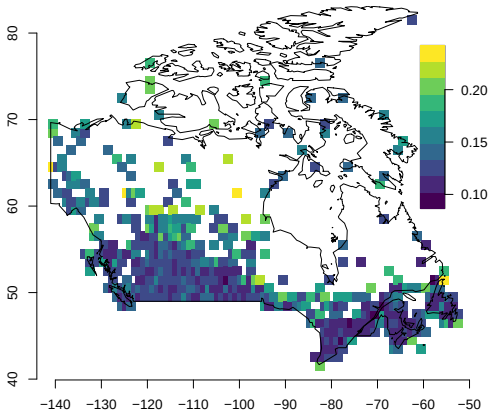
(b) Posterior standard deviation of $a(\boldsymbol{x})$.



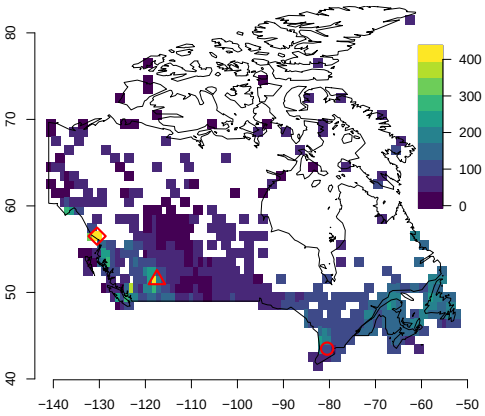
(c) Posterior mean of $b(\boldsymbol{x})$.



(d) Posterior standard deviation of $b(\boldsymbol{x})$.



(e) Posterior mean of 10-year return levels $z_{10}(\boldsymbol{x})$.



(f) Posterior standard deviation of $z_{10}(\boldsymbol{x})$.

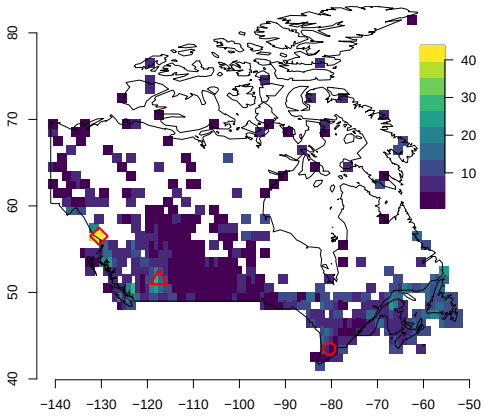


Figure 2.10: Posterior means and standard deviations of $a(\boldsymbol{x})$, $b(\boldsymbol{x})$, and $z_{10}(\boldsymbol{x})$.

The posterior means and standard deviations of model parameters at each location are plotted in Figure 2.10a, 2.10b, 2.10c, 2.10d. Spatial patterns are observed for the values of $\widehat{a}(\boldsymbol{x}) = E[a(\boldsymbol{x}) \mid \boldsymbol{y}]$ and $\widehat{b}(\boldsymbol{x}) = E[b(\boldsymbol{x}) \mid \boldsymbol{y}]$ across Canada, with higher values of $\widehat{a}(\boldsymbol{x})$ and $\widehat{b}(\boldsymbol{x})$ in the Rocky mountains region in British Columbia and south of Alberta. We note that the uncertainty is higher for $b(\boldsymbol{x})$ in southern Nunavut and the southeast of Northwest Territories, where data is most sparse. Plotted in Figure 2.10e and 2.10f are the posterior 10-year return level estimates $\widehat{z}_{10}(\boldsymbol{x}) = E[z_{10}(\boldsymbol{x}) \mid \boldsymbol{y}]$ and the corresponding standard deviations. These two maps provide a practical guide to interpreting the risk of extreme snowfall across Canada. For example, the posterior mean estimate of 10-year return level at the coordinates $\boldsymbol{x} = (-80.5°, 43.5°)$ in Waterloo, ON (marked by the red circle in Figure 2.10e and 2.10f) being 129.12cm means that it takes an average of 10 years before one observes a monthly total snowfall as extreme as 129.12cm at this location. The 10-year return level can also be interpreted as a 10% chance of observing such extreme monthly total snowfalls in any year at a given location. The same claim about coordinates $\boldsymbol{x} = (-117.5°, 51.5°)$ in Glacier National Park in BC (marked by the red triangle) would be true for a monthly total snowfall as extreme as 377.40 cm. Reading Figure 2.10f, the estimate of the former location is relatively more precise compared to the latter one. From the uncertainty maps, we can also identify locations where more observations may be needed to attain a practical level of precision, e.g., at $\boldsymbol{x} = (-130.5°, 56.5°)$ near Granduc, BC (marked by the red diamond).

2.5 Discussion

In this chapter we develop a computationally efficient method of Bayesian inference for fitting GEV-GP models, which have a wide range of applications in studying weather extremes. The proposed method, Laplace-MQ, applies the Laplace approximation for calculating the marginal likelihood and a Normal approximation based on mode and quadrature to obtain the joint posterior distribution of hyperparameters and random effects. Scalability to many thousand spatial observations is achieved using a sparsity-inducing approximation to the spatial precision matrix via GRMFs. An efficient implementation of our method is provided in the R/ C++ package `SpatialGEV`. Our simulation studies indicate that the Laplace-MQ method can produce biased estimates of the model hyperparameters. In situations where it is important to accurately estimate these, exact Bayesian inference methods such as MCMC may be preferable. However, when primary scientific interest lies in estimating the spatially-varying GEV parameters – e.g., for making return-level predictions, or to perform model selection and goodness-of-fit assessments as we have done in

Section 2.4.1 – then our simulations show that Laplace-MQ is highly accurate, and significantly faster than a state-of-the-art MCMC algorithm on the joint posterior distribution commonly used to fit hierarchical models such as the GEV-GP.

The INLA method is a popular alternative to MCMC which has been used successfully in many analyses of weather extremes (e.g., Opitz et al., 2018; Castro-Camilo et al., 2019, 2020). However, existing implementations cannot be used for GEV-GP models with two or more spatial random effects. Such flexibility can be crucial for accurate prediction of weather extreme values, as demonstrated in the analysis of extreme snowfall in Section 2.4. Another alternative to MCMC on the full posterior distribution is the Max-and-Smooth marginal likelihood approximation, which is similar in spirit to that of Laplace, but can be much faster. However, our simulations show that it is less accurate than Laplace for estimating the shape parameter of the GEV-GP model when the number of observations per location is not sufficiently large.

We adopted a simple Taylor approximation for the conditional posterior distribution of the random effects $p(\boldsymbol{u} \mid \boldsymbol{\theta}, \boldsymbol{y})$ given the fixed parameters $\boldsymbol{\theta}$ of the GEV-GP model. In a sense, we have assumed that it is sufficient for estimation accuracy and precision to condition on the posterior mode of the hyperparameters. This approach might underestimate the random effects uncertainty when the posterior distribution of the hyperparameters is wide. Alternatively, a second-order Laplace approximation (e.g., Tierney et al., 1989; Bianconcini and Cagnone, 2012) can be applied to potentially increase the precision, but the cost entailed in computing the second-order term demands a trade-off between speed and precision. Numerical quadrature techniques can also be used to integrate over multiple representative values of the hyperparameters, resulting in a more precise estimation of the posterior distribution of the random effects (e.g., Bianconcini, 2014). Applying such methods requires careful choice of the quadrature nodes and weights and other computational considerations. Stringer (2021) has implemented an adaptive Gauss-Hermite quadrature technique on a flexible class of latent Gaussian models. It is of interest to investigate its applicability to the GEV-GP model.

A fast inference method is important for the modelling and analysis of datasets containing a large number of spatial locations, especially when numerical procedures are required to work with an intractable form of the posterior distribution. Our method can work with both dense covariance matrices in the latent Gaussian processes, which scale poorly to large datasets due to the $\mathcal{O}(n^3)$ cost of covariance matrix inversions, and sparse matrices that approximate the dense covariance matrix. Indeed, computational complexity can be improved by either introducing sparsity in the covariance matrices or by reducing their dimensionality. Methods that adopt the former approach typically involve assumptions about the dependence structure of the random effects, such as the SPDE approximation

for the covariance matrix employed in this chapter and in the `R-INLA` implementation. An example of the latter approach, which is another line of research, is the inducing point method within the variational inference framework (Quiñonero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2006a; Titsias and Lawrence, 2010) which has been widely used in Gaussian process regression with tens of thousands of observations, and has also been applied on some versions of the latent Gaussian model with non-Gaussian likelihoods (e.g. Gal et al., 2015; Bonilla et al., 2019). These methods in the Gaussian process literature are appealing from a theoretical point of view, but challenges remain in finding the optimal variational distribution that approximates the true posterior well. Extending these methods to the GEV-GP setting is a promising direction for future work.

Chapter 3

Continuous-Time Multi-Neuron Latent Factor Model for Spike Trains

3.1 Introduction

An integral part of neuroscience research is understanding the interactions between different parts of the brain, which provides insight on how information traverses the nervous system and how behavioural and neurophysiological phenomena are modulated by this information (Okatan et al., 2005; Lynall et al., 2010; Rosenberg et al., 2020). Such interactions can be quantified by inferring the functional connectivity (FC) in the brain, which corresponds to the temporal dependence between neural activities (Sporns et al., 2005; Ostojic et al., 2009; Rubinov and Sporns, 2010; Sporns, 2013; Bastos and Schoffelen, 2016). In order to assess FC patterns on relatively short timescales, the data of choice consists of high-resolution electrochemical signals, known as *spikes*, simultaneously recorded from multiple neurons in various brain areas of interest.

For each neuron, a sequence of such consecutive signals is called a *spike train*. During the spike-generating process for a neuron, the aggregation of its received excitatory and inhibitory inputs leads to an increase in its membrane potential (Kalat, 2018). When the membrane potential exceeds its excitation threshold, a spike occurs and the potential quickly resets to its resting state (Penit-Soria et al., 1987), a process taking only milliseconds (Rosenkranz and Grace, 2002; Tierney et al., 2004) (see Figure 3.1a). Since the spike time is orders of magnitude faster than the time between spikes, spike trains are typically treated as point process data, as illustrated in the raster plot in Figure 3.1b. Recent advances in neural data collection technology, notably the NeuroPixels probes (Jun

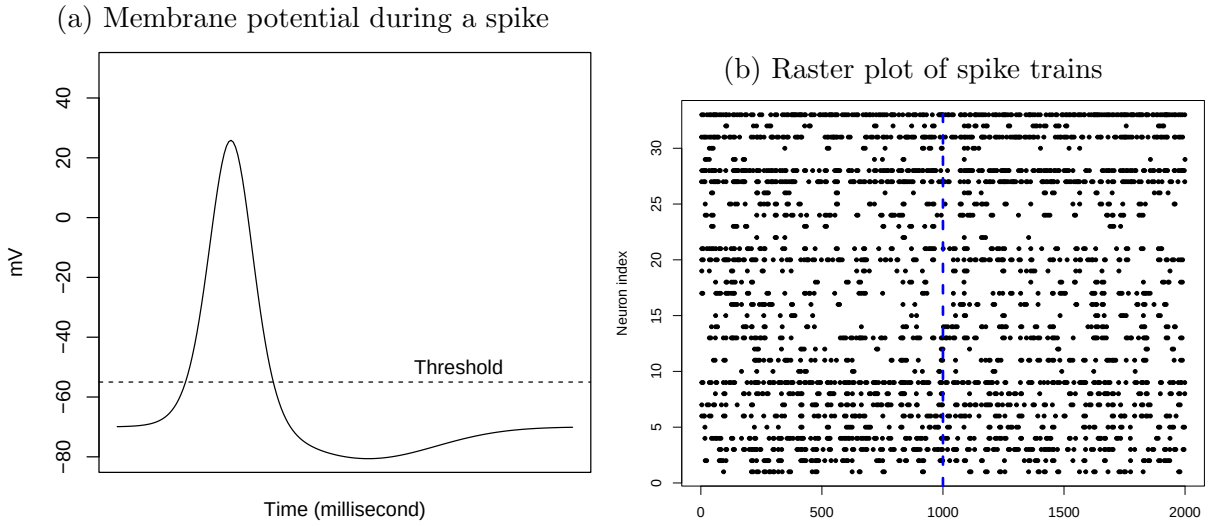


Figure 3.1: (a) Evolution of the membrane potential of a neuron during a spike (adapted from p31 of Kalat, 2018). (b) The raster plot of spike trains with each dot indicating the occurrence of a spike at the corresponding time.

et al., 2017; Steinmetz et al., 2020), has enabled the unprecedented recording of hundreds of neurons at once. These multivariate spike trains provide invaluable information about population dynamics and functional connectivity between neurons (deCharms and Merzenich, 1996; Petersen et al., 2001; Averbeck et al., 2006; Stevenson et al., 2008; Chelaru and Dragoi, 2008), which in turn can be studied to better understand the interactions formed within a larger part of the brain.

Both model-free and model-based spike-train analysis tools have been developed to study neuron population dynamics. The simplest forms of model-free analysis consider descriptive statistics such as the joint peri-stimulus time histogram (Aertsen et al., 1989; Cardoso de Oliveira et al., 2001; Ventura et al., 2005) and the cross-correlogram (Perkel et al., 1967; Fujisawa et al., 2008; Benchenane et al., 2010; Diba et al., 2014). Both of these tools allow for pairwise comparisons between spike trains via visual inspection or hypothesis testing (Harrison et al., 2013). However, such pairwise comparisons do not scale well to moderate or large populations of neurons. For such populations, similarity-based algorithms are a more efficient model-free neuron clustering approach. These algorithms typically define a distance measure representing the (dis)similarity between the spike trains of neurons, which is subsequently used to find neuron clusters formed as a result of synchro-

nized activities (see, e.g., Fellous et al., 2004; Humphries, 2011; Lopes-dos-Santos et al., 2013; Billeh et al., 2014; Herzog et al., 2021). Despite their computational efficiency, model-free methods lack uncertainty quantification and often fail to uncover more complex underlying neuronal relationships beyond correlation at the level of observed data.

In contrast, latent variable models (LVMs) for multiple neurons are able to discover patterns which model-free algorithms cannot, thanks to their ability to specify different structures in the latent layers for summarizing the covariation across neuronal activities. The remarkable success of spike-train LVMs in predictive tasks (Yu et al., 2009; Gao et al., 2016; Wu et al., 2017; Zhao and Park, 2017; Pandarinath et al., 2018) and neuronal clustering (Macke et al., 2011; Buesing et al., 2014; Lin et al., 2018; Wei et al., 2022) motivates the use of LVMs in this work to study the functional connectivity between neurons. As all the LVMs in the aforementioned works are specified in discrete time, they require binning of the experimental time to obtain spike counts, which are then typically modelled using a Poisson likelihood. Two popular modelling frameworks are the Poisson linear dynamic system (LDS) and the Poisson generalized linear model (GLM). For Poisson-LDS models, the latent neuronal dynamics is often assumed to be Markov and linear (Macke et al., 2011; Buesing et al., 2014; Wei et al., 2022). Generalizations beyond these assumptions have been achieved using the fLDS model (Gao et al., 2016) and latent deep neural networks (Archer et al., 2015; Sussillo et al., 2016; Pandarinath et al., 2018). In the Poisson-GLM framework, interactions between a given neuron and other neurons enter the model as linear components of the log conditional intensity function for the underlying Poisson process (Truccolo et al., 2005; Okatan et al., 2005; Pillow et al., 2008; Zoltowski and Pillow, 2018). When used for FC analysis or clustering, the inclusion of pairwise interaction terms results in poor scalability when dealing with a large number of neurons.

However, a major concern about the aforementioned Poisson models is that the size of time bins is often chosen arbitrarily – despite it having a considerable impact on parameter estimation (Nelson, 2002; Kass and Ventura, 2006; Ramezan et al., 2014). Although there are Poisson process models set in continuous time without binning (e.g., Kass and Ventura, 2001; Duncker and Sahani, 2018; Williams et al., 2020), these models either cannot handle multiple neurons, or are not designed to study the neuronal interactions. There exist non-Poisson continuous-time models (Zhou et al., 2016, 2022) which allow for inferring dynamic FC varying over time, but these models quickly become computationally infeasible as the number of neurons grows. Therefore, simultaneously achieving scalability to hundreds of neurons and robustness to binning is a critical challenge in multi-neuron modelling.

In this work, we present a novel multivariate spike-train latent-factor model for FC analysis. To the best of our knowledge, this is the first continuous-time model developed

for this purpose. We propose an efficient and accurate method of approximate Bayesian inference for the model – i.e., scaling linearly in both the duration of the data and the number of neurons – via a Laplace approximation. While the computations do require time discretization, the time bins can be made so small as to incur negligible bias. The proposed model is biologically justifiable as it explicitly connects the underlying latent processes with the dynamics of membrane potentials in the neuronal system. Furthermore, we provide fast and accurate approximation techniques to (i) determine the number of latent factors and (ii) sample from leave-one-out neuron predictive distributions. The latter is particularly useful for model comparisons and goodness-of-fit assessments.

We utilize our model to analyze a large set of experimental data from rats' orbitofrontal cortex (OFC). Our analyses investigate whether the patterns of FC reflects how rats' OFC predicts stimuli or cues, a process known as *value coding* (Louie and Glimcher, 2012; Eickhoff and Müller, 2015; Pischedda et al., 2020). The experiments involve rats trained to recognize three types of cues corresponding to sucrose, quinine and alcohol. During data recording sessions, the cues are presented to the rats to check their neural responses. Applying the proposed model to the recorded spike trains, we devise a statistical protocol which allows us to detect statistically and scientifically meaningful changes in FC between neurons following cues. This finding is consistent with the neurophysiological theory that the OFC plays an integral role in motivated behaviours and value-coding in rats' brain. We note that our protocol detects a signal across multiple rats and sessions, despite the presence of substantial inter-animal and inter-session variability.

The remainder of this chapter is structured as follows. We introduce our proposed model in Section 3.2, followed by an efficient Bayesian inference procedure based on the Laplace approximation and a goodness-of-fit diagnostic measure in Section 3.3. Model performance is demonstrated through simulation studies in Section 3.5. Section 3.6 presents the real data analysis described above on rats' OFC. Concluding remarks and discussion are offered in Section 3.7.

3.2 Model

Assuming that q neurons are sampled (recorded) from the population of interest, the net effects of the underlying neural integration of excitatory and inhibitory inputs are represented using a q-dimensional multivariate Wiener process $\boldsymbol{x}(t) = (x_1(t), \ldots, x_q(t))$ whose covariance matrix $\boldsymbol{\Sigma}$ encodes neuronal interactions. In other words, $\boldsymbol{x}(t)$ can be thought of as proxies for the evolving membrane potentials of the neurons. To infer the

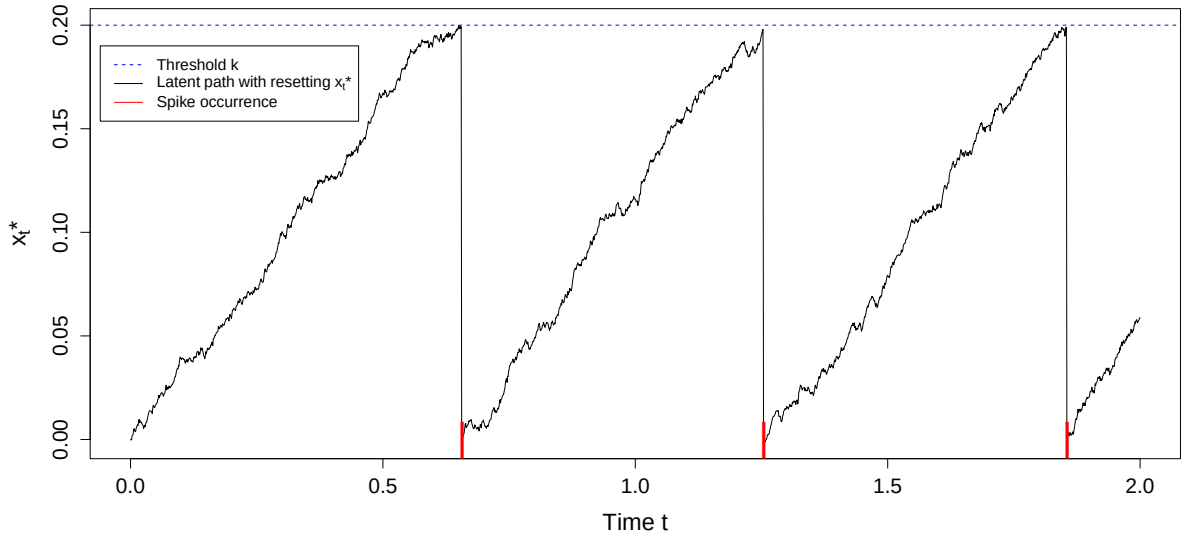


Figure 3.2: Illustration of the spike-generating mechanism for a single neuron.

functional connectivity among neurons, a core part of model inference lies in estimating $\boldsymbol{\Sigma}$.

Marginally for each neuron i, when the excitatory inputs dominate the inhibitory ones, the corresponding univariate process $x_i(t)$ drifts upwards, and vice versa. To represent the spike-generating machinery described in Section 3.1, we assume that the time of a spike occurrence for neuron i is the stopping time defined by $\tau(k_i) := \inf\{t : x_i(t) = k_i\}$, where k_i is the neuron-specific threshold of excitation. Parallel to the physiological process of spike generation, the process immediately resets to its initial value $x_i(0)$ following threshold crossing. A graphical illustration of this process is given in Figure 3.2. Biologically, the neuron then goes through a refractory period (lasting a few milliseconds) of no subsequent spikes, which in turn translates to a time discretization scale in model inference (see Section 3.3). The marginal single-neuron model is similar to the leaky integrate-and-fire model (Gerstner et al., 2014) which first appeared in Lapicque (1907). The ideas of threshold-crossing and resetting are motivated by the Skellam process with resetting model for neural spike trains in Ramezan et al. (2016) and our model generalizes the limiting case of its multivariate version in Ramezan et al. (2022). The link between our point process model and the underlying spike-generating mechanism offers biologically plausible interpretation for downstream analyses.

3.2.1 Single-Neuron Model

We begin by introducing the single-neuron (univariate) version of the proposed model. Suppose the observed data consists of spike trains from q neurons simultaneously recorded during the time interval $(0, T]$. For time $t \in (0, T]$, let $y_i(t) \in \{0, 1\}$ be the spike indicator for neuron i, $n_i(t)$ the total number of spikes up until time t, and $x_i(t)$ the dynamic latent path function which summarizes all external inputs received by neuron i as well as its own baseline activities. In fact, $x_i(t)$ role-plays the cross-membrane potential of neuron i at time t. Motivated by the spike generating mechanism introduced in Section 3.1, we assume that neuron i fires a spike at time $t^\star$ if its latent path $x_i(t)$ crosses some neuron-specific threshold k_i at time $t^\star$, after which the $x_i(t)$ resets to its baseline level $x_i(0)$, which is analogous to the resting potential of the neuron. Thus, the continuous-time model for each neuron $i = 1, \ldots, q$ is written as

$$\begin{aligned} y_i(t) &= \mathbb{1}\big(x_i^\star(t) \geq k_i\big), \\ x_i^\star(t) &= x_i(t) - k_i n_i(t), \\ x_i(t) &= x_i(0) + \alpha_i t + \sigma_i w(t), \end{aligned} \tag{3.1}$$

where $x_i^\star(t)$ is the dynamic latent path with resetting, α_i and σ_i correspond to the drift and diffusion parameters of $x_i(t)$, and $w(t)$ is a Wiener process. Furthermore, we assume the drift $\alpha_i > 0$ for all $i = 1, \ldots, q$. The positive drift guarantees that $P(Z_i < \infty) = 1$ where Z_i is the first passage time of threshold k_i for $x_i(t)$. For simplicity, we assume the initial values $x_i(0) = 0$, and, consequently, $k_i > 0$ for $i = 1, ..., q$. When the experiment includes multiple repeated trials under the same experimental conditions, we assume that the trials are independent realizations of the same latent process described by $x_i(t)$ in (3.1).

When only a single neuron is considered, the model defined by (3.1) corresponds to a homogeneous inverse Gaussian (IG) model for the interspike intervals (ISIs), i.e., the time between two consecutive spikes. For a univariate process $x_i(t)$ with positive drift α_i and variance σ_i^2, the first passage time for crossing the threshold k_i, i.e. the ISI, follows an IG distribution (Folks and Chhikara, 1978) with its probability density function given by

$$f(t \mid k_i, \alpha_i, \sigma_i^2) = \frac{k_i}{\sigma_i \sqrt{2\pi t^3}} \exp\left\{ -\frac{(k_i - \alpha_i t)^2}{2\sigma_i^2 t} \right\}. \tag{3.2}$$

Hence, the parameters $(k_i, \alpha_i, \sigma_i)$ characterize the spiking activity of a single neuron i. The mean and variance of the ISI distribution are $E_i(t) = k_i/\alpha_i$ and $\mathrm{Var}_i(t) = k_i\sigma_i^2/\alpha_i^3$. The inverse of the mean interspike interval $1/E_i(t) = \alpha_i/k_i$ is the firing rate of neuron i, which is a helpful summary statistic in downstream analyses.

3.2.2 Multi-Neuron Factor Model for Neuron Dependence

The multi-neuron model is obtained by including multiple latent processes, one per neuron, hence extending (3.1) to a multivariate form. Let $\boldsymbol{y}(t) = (y_1(t), \ldots, y_q(t))$ denote the multivariate spike train, $\boldsymbol{x}^*(t) = (x_1^*(t), \ldots, x_q^*(t))$ the multivariate dynamic latent path with resetting, and $\boldsymbol{x}(t) = \big(x_i(t), \ldots, x_q(t)\big)$ the multivariate dynamic latent path without resetting. The functional connectivity between neurons is summarized using the covariance matrix for the process $\boldsymbol{x}(t)$. Hence, we can write the multivariate version of (3.1) as:

$$\begin{aligned} \boldsymbol{y}(t) &= \Big(\mathbb{1}\big(x_1^\star(t) \geq k_1\big), \ldots, \ \mathbb{1}\big(x_q^\star(t) \geq k_q\big)\Big), \\ \boldsymbol{x}^*(t) &= \big(x_1(t) - k_1 n_1(t), \ldots, \ x_q(t) - k_q n_q(t)\big), \\ \boldsymbol{x}(t) &= \boldsymbol{x}_0 + \boldsymbol{\alpha} t + \boldsymbol{\Sigma}^{\frac{1}{2}} \boldsymbol{w}(t), \end{aligned} \tag{3.3}$$

where $\boldsymbol{x}_0 = (x_1(0), \ldots, x_q(0))$ consists of all initial states, $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_q)$ is the drift vector, $\boldsymbol{w}(t) = \big(w_1(t), \ldots, w_q(t)\big)$ is a vector of mutually independent Wiener processes, and the covariance matrix $\boldsymbol{\Sigma}$ encodes the correlation between the latent paths, hence neurons. Inheriting from the univariate model, it is assumed that repeated trials consisting of the same group of simultaneously recorded neurons are independent realizations of $\boldsymbol{x}(t)$ in (3.3).

To better understand the functional connectivity between neurons, a core but challenging problem in model inference is the estimation of $\boldsymbol{\Sigma}$. The factor analysis framework is a popular way to represent a correlation matrix in terms of a smaller number of simple components. We adopt this framework to model the neuron dependence for its advantages in both interpretability and computational efficiency.

Suppose that the latent dynamic path can be represented using d independent common processes $\boldsymbol{f}(t) = (f_1(t), \ldots, f_d(t)) \in \mathbb{R}^d$, where d is typically much smaller than the number of neurons q. One can interpret the functional connectivity between neurons as driven by a set of common inputs $\boldsymbol{f}(t)$. Namely,

$$\boldsymbol{x}(t) = \boldsymbol{x}_0 + \boldsymbol{\alpha} t + \boldsymbol{\Lambda} \boldsymbol{f}(t) + \boldsymbol{\Psi}^{1/2} \boldsymbol{\epsilon}(t), \tag{3.4}$$

where $\boldsymbol{\Lambda} \in \mathbb{R}^{q \times d}$ $(d < q)$, $\boldsymbol{\Psi} = \mathrm{diag}(\psi_1^2, \ldots, \psi_q^2)$, and $\boldsymbol{f}(t)$ and $\boldsymbol{\epsilon}(t)$ are d- and q-dimensional vectors of iid Wiener processes, respectively. In fact, (3.4) produces a process of the form of $\boldsymbol{x}(t)$ in (3.3) with $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}$. Relating (3.4) to the factor analysis (FA) framework, it is worth noting that $\boldsymbol{\Lambda}$ corresponds to the loading matrix in FA. Then each component of $\boldsymbol{\Psi}$ represents the uniqueness of the underlying path of a neuron. Thus, $\boldsymbol{\Lambda}$ can be used for downstream analyses after model fitting, such as clustering neurons, as we will see later in Section 3.5 and 3.6.

3.2.3 Parameter Identifiability

Here we address three identifiability issues about the proposed model. The first one is the joint identification of the marginal parameters of each neuron i: k_i, α_i, and σ_i^2, the ith diagonal elements of $\mathbf{\Sigma}$. When estimating the correlation between the underlying paths of two neurons is not of interest, the marginal parameters can be estimated by fitting an IG model to the interspike interval data of each individual neuron. A common reparameterization of the IG distribution (3.5) is to let $\mu = E(t) = k/\alpha$ and $\lambda = k^2/\sigma^2$ (Tweedie, 1957), where μ is the inverse firing rate of the neuron as described in Section 3.2.1. This leads to the following density function:

$$f(t \mid \mu, \lambda) = \left(\frac{\lambda}{2\pi t^3}\right)^{\frac{1}{2}} \exp\left\{-\frac{\lambda(t-\mu)^2}{2\mu^2 t}\right\}. \tag{3.5}$$

Given the recorded interspike intervals for a neuron, only μ and λ are identifiable from the IG model, whereas k, α, and σ^2 cannot be recovered simultaneously (Brown, 2005). Thus, we fix $\sigma_i^2 = 1$ for $i = 1, \ldots, q$, and let α_i and k_i be estimated freely.

The second identifiability issue is associated with the loading matrix $\mathbf{\Lambda}$, which is identified up to a rotation. For any orthogonal matrix $\boldsymbol{C} \in \mathbb{R}^{d\times d}$, one can define $\mathbf{\Lambda}^\star = \mathbf{\Lambda}\boldsymbol{C}$ and $\boldsymbol{f}(t)^\star = \boldsymbol{C}^T\boldsymbol{f}(t)$ so that $\mathbf{\Lambda}^\star$ and $\boldsymbol{f}(t)^\star$ still satisfy equation (3.4). In order to identify a unique $\mathbf{\Lambda}$, we refer to the conditions in Anderson (2003) and require that its first d rows are lower triangular. Since the diagonal elements of $\mathbf{\Sigma}$ are $\sigma_1^2 = \sigma_2^2 = \cdots = \sigma_q^2 = 1$, $\mathbf{\Psi}$ is identified once $\mathbf{\Lambda}$ is.

The last identifiability problem concerns the choice of the number of independent factors d. Recalling the covariance matrix $\mathbf{\Sigma} = \mathbf{\Lambda}\mathbf{\Lambda}^T + \mathbf{\Psi}$, the left-hand side has $q(q-1)/2$ free parameters, while the right-hand side has $dq - d(d-1)/2$ ones. For $\mathbf{\Lambda}$ and $\mathbf{\Psi}$ to be identifiable, the following inequality needs to hold in addition to the constraint $d < q$ defined for the factor model (3.4):

$$q(q-1)/2 \geq dq - d(d-1)/2. \tag{3.6}$$

For example, when the dataset consists of $q = 6$ neurons, solving (3.6) gives $d \leq 3$.

3.3 Inference

3.3.1 Time Discretization for Computations

To construct computational algorithms, we discretize the experimental time window into bins of length δ such that the probability of observing more than one spike within each bin

is negligible, resulting in $n = \lceil T/\delta \rceil$ bins. This choice is also consistent with the biological phenomenon of *refractory period*. After the neuron fires a spike, it enters a refractory period, typically between 0.5-10ms depending on the neuron type (Hodgkin and Huxley, 1952; Kass et al., 2014; Kalat, 2018), during which the neuron is usually unable to fire another spike.

For a given matrix $\boldsymbol{A} = [a_{i,j}]_{1 \le i \le m, 1 \le j \le n} \in \mathbb{R}^{m \times n}$, let $\boldsymbol{a}_{i,:} = (a_{i,1}, ..., a_{i,n})$ and $\boldsymbol{a}_{:,j} = (a_{1,j}, ..., a_{m,j})$. In each time bin $j = 0, \ldots, n$, let $y_{i,j} \in \{0, 1\}$ indicate spike occurrence and $x_{i,j} = x_i(\delta j)$ the value of the latent path (without resetting) at the endpoint of bin j for neuron $i = 1, \ldots, q$, and define $f_{i,j}$ and $\epsilon_{i,j}$ similarly for the factors and noise, respectively. Thus, $\boldsymbol{x}_{:,j}$ denotes a q-dimensional vector corresponding to the latent states of all neurons in time bin j, and similarly for $\boldsymbol{f}_{:,j}$ and $\boldsymbol{\epsilon}_{:,j}$. We assume that $\boldsymbol{x}_{:,0} = \boldsymbol{0}$ and that no spikes are observed at the initial states, i.e., $\boldsymbol{y}_{:,0} = 0$.

To obtain the resetting process $x^\star_{i,j}$, we reset $x_{i,j}$ to its initial state 0 whenever it crosses the threshold k_i, i.e., when a spike occurs. Since $x^\star_{i,j}$ is only an artificial proxy for the membrane voltage, the threshold k_i does not correspond to the actual biological threshold for spike generation (typically at around -55mV). Instead, k_i is estimated from the data. To facilitate a more efficient gradient-based inference algorithm, which will be introduced in Section 3.3.3, we rewrite the spike generation process in terms of soft-thresholding. For $i = 1, \ldots, q$ and $j = 1, \ldots, n$, the time-discretized version of model (3.3) is given by,

$$\begin{aligned} y_{i,j} &\sim \text{Bernoulli}\big(g_s(x^\star_{i,j} - k_i)\big), \\ x^\star_{i,j} &= x_{i,j} - k_i \sum_{t=0}^{j-1} y_{i,t}, \\ \boldsymbol{x}_{:,j} &= \boldsymbol{x}_{:,j-1} + \boldsymbol{\alpha}\delta + \boldsymbol{\Lambda}\Delta\boldsymbol{f}_{:,j} + \boldsymbol{\Psi}^{1/2}\Delta\boldsymbol{\epsilon}_{:,j}, \end{aligned} \tag{3.7}$$

where $\Delta\boldsymbol{f}_{:,j}$ and $\Delta\boldsymbol{\epsilon}_{:,j}$ are iid normals with mean zero and variance proportional to the identity matrix times δ. In (3.7), soft-thresholding for spike generation is achieved using a sigmoid function $g_s(x) = 1/(1 + e^{-sx})$ with a scaling factor $s \in \mathbb{R}^+$ controlling the steepness fo $g_s(x)$ around zero. We specify $s = 15$ to avoid hard thresholding so that the likelihood is differentiable at the spike times (i.e., not too steep to cause numerical instability for differentiation), while ensuring a near one probability of spiking when $x^\star_{i,j} \ge k_i$. Throughout all experiments, we set the bin length $\delta = 5$ms, the sigmoid scaling factor $s = 15$.

3.3.2 Parameter Priors

For the neuron-specific parameters $\log(k_1), \log(\alpha_1), \ldots, \log(k_q), \log(\alpha_q)$, we choose a flat prior on each of them after observing its good performance on both simulated and experimental data (see Section 3.5 and 3.6). This is because the neuron-specific parameters can be estimated accurately given a moderately large number of spike train data without borrowing information from a prior distribution. For the non-zero entries of $\boldsymbol{\Lambda}$, $(\ell_{11}, \ell_{21}, \ldots, \ell_{qd}) \in \mathbb{R}^{dq-d(d-1)/2}$, a convenient prior on each element is the Normal prior (Lopez and West, 2004). However, as a consequence of the constraint of $\sigma_i^2 = 1$ imposed in Section 3.2.3, the elements of $\boldsymbol{\Lambda}$ are also constrained such that the sum of squares for each row of $\boldsymbol{\Lambda}$ must satisfy $\sum_{u=1}^{d} \ell_{iu}^2 < 1$ for $i = 1, \ldots, q$. Due to the challenge of specifying priors on constrained parameters, we define an unconstrained computational basis $(\ell_{11}^*, \ell_{21}^*, \ldots, \ell_{qd}^*) \in \mathbb{R}^{dq-d(d-1)/2}$. The (i, j) element of $\boldsymbol{\Lambda}$ and the uniqueness component of the ith neuron are given by

$$\ell_{ij} = \frac{\ell_{ij}^\star}{\sqrt{\sum_{u=1}^{d} \ell_{iu}^{\star 2} + 1}}, \quad \psi_i^2 = \frac{1}{\sqrt{\sum_{u=1}^{d} \ell_{iu}^{\star 2} + 1}},$$

respectively. One can verify that $\sum_{u=1}^{d} \ell_{iu}^2 < 1$ for all i and that $\sigma_i^2 = \sum_{u=1}^{d} \ell_{iu}^2 + \psi_i^2 = 1$. We then specify independent Normal priors $\mathcal{N}(0, \tau^{-2})$ on the elements of ℓ_{ij}^* instead. It is noteworthy that the Normal prior is in fact equivalent to $L2$ regularization, so τ also acts as a penalization term on the loading values to prevent extremely large loading estimates when a neuron is mainly driven by one factor. We set $\tau = 1$ after observing it provides sufficient regularizing effect to prevent estimates from exploding.

3.3.3 Integration using Laplace Approximation

To simplify the notation, let $\boldsymbol{Y} \in \mathbb{R}^{qn}$ denote the data from all neurons, $\boldsymbol{X} \in \mathbb{R}^{qn}$ denote all latent states, and $\boldsymbol{\theta}$ summarize the parameters of interest, including the neuron-specific parameters and the non-zero entries of $\boldsymbol{\Lambda}$. The posterior distribution of interest $p(\boldsymbol{\theta} \mid \boldsymbol{Y})$ can be written as the proportion relation:

$$p(\boldsymbol{\theta} \mid \boldsymbol{Y}) \propto p(\boldsymbol{\theta}) p(\boldsymbol{Y} \mid \boldsymbol{\theta}), \tag{3.8}$$

where $p(\boldsymbol{Y} \mid \boldsymbol{\theta}) = \int p(\boldsymbol{Y}, \boldsymbol{X} \mid \boldsymbol{\theta})\, d\boldsymbol{X}$ and the factors $\boldsymbol{f}$ are integrated out analytically. The lack of analytic form for this integral suggests means of inference via MCMC methods. However, the complex hierarchical model structure and the high dimension of $\boldsymbol{X}$ cause prohibitively slow convergence for MCMC algorithms. Here, we describe the Laplace method

(Tierney and Kadane, 1986) to approximate the intractable integration by a tractable optimization procedure.

The desired marginal likelihood can be written as

$$p(\boldsymbol{Y} \mid \boldsymbol{\theta}) = \frac{p(\boldsymbol{Y}, \boldsymbol{X} \mid \boldsymbol{\theta})}{p(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta})}, \tag{3.9}$$

which holds for any value of $\boldsymbol{X}$. The Laplace method approximates the denominator $p(\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta})$ by a Normal distribution with mode-quadrature matching, which is given by

$$\boldsymbol{X} \mid \boldsymbol{Y}, \boldsymbol{\theta} \approx \mathcal{N}\big(\widehat{\boldsymbol{X}}(\boldsymbol{\theta}),\ (-\boldsymbol{H}(\boldsymbol{\theta}))^{-1}\big), \tag{3.10}$$

where $\widehat{\boldsymbol{X}}(\boldsymbol{\theta})$ is the mode of $\log p(\boldsymbol{Y}, \boldsymbol{X} \mid \boldsymbol{\theta})$ and $\boldsymbol{H}(\boldsymbol{\theta})$ is its Hessian at $\boldsymbol{X} = \widehat{\boldsymbol{X}}(\boldsymbol{\theta})$. Plugging (3.10) into (3.9) gives the Laplace approximation to the marginal likelihood: $p_{\text{LA}}(\boldsymbol{Y} \mid \boldsymbol{\theta})$. More details of the Laplace approximation are given in Section 2.2.2.

Given the Laplace-approximated marginal likelihood, we construct a Normal approximation to the posterior of interest. This results in the approximate marginal posterior distribution

$$\boldsymbol{\theta} \mid \boldsymbol{Y} \approx \mathcal{N}(\widehat{\boldsymbol{\theta}},\ \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}}), \tag{3.11}$$

where

$$\widehat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \big(\log p_{\text{LA}}(\boldsymbol{Y} \mid \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}) \big),$$
$$\widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} = - \left[\big(\frac{\partial^2}{\partial \boldsymbol{\theta}^2} \log p_{\text{LA}}(\boldsymbol{Y} \mid \widehat{\boldsymbol{\theta}}) + \log p(\widehat{\boldsymbol{\theta}}) \big) \right]^{-1},$$

corresponding to the mode and quadrature of the Laplace-approximated right-hand side of (3.8).

Note that the inference model described in (3.7) is closely connected to a binary Gaussian process classification (GPC) model (Rasmussen and Williams, 2006), for which the Laplace method is a widely used approximate inference approach (Williams and Barber, 1998; Kuss and Rasmussen, 2005; Nickisch and Rasmussen, 2008). Error of the Laplace approximation for high-dimensional integrals is studied in Shun and McCullagh (1995); Rue et al. (2009); Ogden (2021). While investigating the approximation error of $p(\boldsymbol{Y} \mid \boldsymbol{\theta}) \approx p_{\text{LA}}(\boldsymbol{Y} \mid \boldsymbol{\theta})$ for the proposed model is beyond the scope of this chapter, Section 3.5.1 studies the overall inference performance, which is of more interest since the Laplace approximation is ultimately used for inference purposes.

3.3.4 Two-Step Estimation with Fixed Marginal Parameters

While the Laplace method already provides substantial computational gain in comparison to exact Bayesian inference techniques such as MCMC, it can be further sped up using a two-step procedure. By defining $\boldsymbol{\theta}_1 = (\log(k_1), \log(\alpha_1), \ldots, \log(k_q), \log(\alpha_q))$ and letting $\boldsymbol{\theta}_2$ be the non-zero entries of $\boldsymbol{\Lambda}$, our proposed two-step inference estimates $p(\boldsymbol{\theta}_1 \mid \boldsymbol{Y})$ and $p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y})$ sequentially rather than jointly estimating $p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \mid \boldsymbol{Y})$. Specifically, the first step estimates the posterior for the marginal parameters $p(\boldsymbol{\theta}_1 \mid \boldsymbol{Y})$ by considering each single-neuron model independently, and the second step approximates the posterior for the dependence parameters $p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y})$ by the means of the Laplace approximation.

The first step constructs a likelihood function under a purposefully incorrect working assumption of independence of neurons ($\boldsymbol{\Sigma} = \boldsymbol{I}$), resulting in likelihood separability:

$$\mathcal{L}_{\text{ind}}(\boldsymbol{\theta}_1 \mid \boldsymbol{Y}) = \prod_{i=1}^{q} \mathcal{L}(\boldsymbol{\theta}_{1,i} \mid \boldsymbol{y}_{i,:}), \tag{3.12}$$

where $\boldsymbol{\theta}_{1,i} = (\log(\alpha_i), \log(k_i))$ are the marginal parameters and $\boldsymbol{y}_{i,:}$ is the observable data for neuron i. This likelihood can be viewed as a misspecified model and is also known as a particular case of the composite likelihood (Lindsay, 1988; Varin et al., 2011). Recall in Section 3.2.1 that, given $\boldsymbol{\theta}_{1,i}$, the interspike intervals (ISIs) for neuron i follows an inverse Gaussian distribution. Using the inverse Gaussian parametrization in (3.5), the MLEs of μ_i and λ_i for each neuron $i = 1, \ldots, q$ can be computed, by maximizing the ith likelihood component $\mathcal{L}(\boldsymbol{\theta}_{1,i} \mid \boldsymbol{y}_{i,:})$, in closed forms as

$$\widehat{\mu}_i = \bar{t}_i, \quad \widehat{\lambda}_i = n_i \Big/ \sum_{j=1}^{n_i} (1/t_{i,j} - 1/\bar{t}_i), \tag{3.13}$$

where $\{t_{i,1}, \ldots, t_{i,n_i}\}$ is a sample of ISIs for neuron i and n_i is the sample size. The variance estimate of $(\widehat{\mu}_i, \widehat{\lambda}_i)$ is given by the corresponding inverse negative Hessian $\widehat{\boldsymbol{H}}_i^{-1}$ evaluated at $(\widehat{\mu}_i, \widehat{\lambda}_i)$, where

$$\widehat{\boldsymbol{H}}_i = \begin{pmatrix} -\frac{n_i \widehat{\lambda}_i}{\widehat{\mu}_i^3} & 0 \\ 0 & -\frac{n_i}{2\widehat{\lambda}_i^2} \end{pmatrix}.$$

Applying a Normal approximation of $p(\boldsymbol{\theta}_1 \mid \boldsymbol{Y})$ about its mode leads to

$$\boldsymbol{\theta}_1 \mid \boldsymbol{Y} \approx \mathcal{N}(\widehat{\boldsymbol{\theta}}_1, \widehat{\boldsymbol{V}}_1). \tag{3.14}$$

In the second step, $p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y})$ is given by

$$p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y}) = \int p(\boldsymbol{\theta}_2 \mid \boldsymbol{\theta}_1, \boldsymbol{Y}) p(\boldsymbol{\theta}_1 \mid \boldsymbol{Y}) \, \mathrm{d}\boldsymbol{\theta}_1. \tag{3.15}$$

An approximation to (3.15) is using the posterior mode $\widehat{\boldsymbol{\theta}}_1$ as a plug-in, which assumes that all of the probability mass of $p(\boldsymbol{\theta}_1 \mid \boldsymbol{Y})$ is at $\boldsymbol{\theta}_1 = \widehat{\boldsymbol{\theta}}_1$. That is,

$$p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y}) \approx \widehat{p}(\boldsymbol{\theta}_2 \mid \boldsymbol{Y}) = p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y}; \widehat{\boldsymbol{\theta}}_1). \tag{3.16}$$

This allows us to use the Laplace approximation described in Section 3.3.3 to estimate $p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y}; \widehat{\boldsymbol{\theta}}_1)$ with $\boldsymbol{\theta}_1$ treated as known, thereby reducing the dimension of parameter space in the corresponding numerical optimizations. As a result of the Laplace approximation, a Normal approximation can be constructed for $p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y})$ such that

$$\boldsymbol{\theta}_2 \mid \boldsymbol{Y} \approx \mathcal{N}(\widehat{\boldsymbol{\theta}}_2, \widehat{\boldsymbol{V}}_2), \tag{3.17}$$

where $\widehat{\boldsymbol{\theta}}_2 = \operatorname{argmax}_{\boldsymbol{\theta}_2} \log \widehat{\mathcal{L}}_{\text{Laplace}}(\boldsymbol{\theta}_2 \mid \boldsymbol{Y}; \widehat{\boldsymbol{\theta}}_1)$ and $\widehat{\boldsymbol{V}}_2$ is the corresponding inverse Hessian matrix. In summary, the two-step procedure first approximates $p(\boldsymbol{\theta}_1 \mid \boldsymbol{Y})$ via (3.14) and then $p(\boldsymbol{\theta}_2 \mid \boldsymbol{Y})$ via (3.17) with $\boldsymbol{\theta}_1 = \widehat{\boldsymbol{\theta}}_1$ plugged in.

The two-step estimation makes two important simplifications compared to the joint estimation. First, under the working independence assumption of (3.12), the neuron-specific marginal parameters $\boldsymbol{\theta}_{1,i}$, $i = 1, \ldots, q$, are estimated separately via maximizing the corresponding likelihood component $\mathcal{L}(\boldsymbol{\theta}_{1,i} \mid \boldsymbol{y}_{i,:})$. This can be justified if, when the data of neuron i is provided, the contribution of the data from neurons j for $j \neq i$ is relatively negligible to estimate $\boldsymbol{\theta}_{1,i}$. The second simplification involves using the plug-in $\boldsymbol{\theta}_1 = \widehat{\boldsymbol{\theta}}_1$ instead of integrating over the uncertainty of $\boldsymbol{\theta}_1$ in (3.15). In practice, it is expected that $p(\boldsymbol{\theta}_1 \mid \boldsymbol{Y})$ becomes more and more centred at $\boldsymbol{\theta}_1$ as n_i, the number of ISIs, grows for all neurons. Toy simulation examples in Appendix B.1 compare the accuracy and precision of two-step estimation versus joint estimation of $p(\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \mid \boldsymbol{Y})$ via the Laplace approximation (3.11), finding the two to give similar results. Therefore, we believe that the use of plug-in can be reasonably justified with a sufficiently large number of observed spikes. Satisfactory performance of the proposed inference on artificial data is demonstrated in the simulation studies in Section 3.5.

3.3.5 Additional Computational Considerations

It is worth noting that expressing the latent process $\boldsymbol{X}$ in (3.3) in the FA framework in (3.4) provides not only interpertability but also computational gain. When a large number of

simultaneously recorded neurons are being analyzed, the $\mathcal{O}(q^3)$ computational complexity of evaluating the multivariate Wiener process in (3.3) imposes significant computational challenges on model fitting. Moreover, the matrix $\boldsymbol{\Sigma}$ contains $q(q-1)/2$ free parameters to estimate, which grows quadratically with q. In the FA framework, the number of free parameters in $\boldsymbol{\Lambda}$ is linear with q when the number of factors d is fixed. Furthermore, calculations of $\boldsymbol{\Sigma}^{-1}$ and the log determinant $\log|\boldsymbol{\Sigma}|$ in each evaluation of (3.3) are also more efficient writing $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}$. Inversion of $\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}$ can be performed more efficiently using the Woodbury matrix identity (Section 3.7 of Banerjee and Roy, 2014):

$$(\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi})^{-1} = \boldsymbol{\Psi}^{-1} - \boldsymbol{\Psi}^{-1}\boldsymbol{\Lambda}\boldsymbol{\Omega}^{-1}\boldsymbol{\Lambda}^T\boldsymbol{\Psi}^{-1}, \tag{3.18}$$

where $\boldsymbol{\Omega} = \boldsymbol{I} + \boldsymbol{\Lambda}^T\boldsymbol{\Psi}^{-1}\boldsymbol{\Lambda} \in \mathbb{R}^{d\times d}$. As a result, inverting $\boldsymbol{\Omega}$ is $\mathcal{O}(d^3)$ as opposed to $\mathcal{O}(q^3)$, where typically $d \ll q$ in factor analysis, and inverting the diagonal $\boldsymbol{\Psi}$ is trivial. Similarly, a faster way of calculating $\log|\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}|$ is given by (Theorem 10.10 Banerjee and Roy, 2014)

$$\log|\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}| = \log|\boldsymbol{\Omega}| + \log|\boldsymbol{\Psi}|, \tag{3.19}$$

whose computational cost is mainly that of computing $\log|\boldsymbol{\Omega}|$.

Implementation of the Laplace approximation in Section 3.3.3 is also carefully considered. Finding the mode $\widehat{\boldsymbol{\theta}}$ for the Normal-approximated posterior (3.11) is a nested optimization problem, where an inner optimization $\arg\max_{\boldsymbol{X}} \log p(\boldsymbol{Y}, \boldsymbol{X} \mid \boldsymbol{\theta})$ is performed at each step of the outer optimization of $\arg\max_{\boldsymbol{\theta}}(\log p_{\text{LA}}(\boldsymbol{Y} \mid \boldsymbol{\theta}) + \log p(\boldsymbol{\theta}))$. Additionally, the calculation of $\log|\boldsymbol{H}(\boldsymbol{\theta})|$ is computationally demanding due to the large size of $\boldsymbol{H}(\boldsymbol{\theta})$. To avoid memory overflow and to speed up the numerical procedures, we implement the Laplace method using the R/C++ library `TMB` (Kristensen et al., 2016), whose advantages have been discussed in Section 2.2.5.

Leveraging `TMB` and the other aforementioned computational techniques, we implement the proposed model and inference procedure in the R/C++ library `fastr` (**F**actor **A**nalysis of **S**pike **T**rains in **R**) hosted at https://github.com/meixichen/fastr.

3.3.6 Determining of the Number of Factors

The true number of factors d^* is typically unknown in real data analysis. Large computational resources are needed for standard model selection criteria such as the AIC or cross-validation, as they require fitting the model multiple times for different values of d. We employ an exploratory factor analysis to choose a suitable d. For each neuron i, a single-neuron model, defined by (3.1) with $q = 1$, is fitted to its spike train to give the

latent path estimate $\widehat{\boldsymbol{x}}_i$. Note that fitting q single-neuron models is orders of magnitude faster than fitting a joint q-dimensional multi-neuron factor model. The latent path increment estimates are given by $d\widehat{\boldsymbol{x}}_i = (\hat{x}_{i,1} - x_{i,0}, \ldots, \hat{x}_{i,n} - \hat{x}_{i,n-1}),\ i = 1, \ldots, q$. For each $d \in \{d_1, \ldots, d_m\}$, which is a set of candidate values for the number of factors, we treat the matrix $\widehat{\boldsymbol{X}} = (d\widehat{\boldsymbol{x}}_1, \ldots, d\widehat{\boldsymbol{x}}_q)^T$ as an observed data matrix and apply the standard factor analysis procedure on it (using the `factanal()` function in R).

The results, including the estimated loading matrix and the variance explained by the factors, are then used as guides for choosing d. Two approaches for selecting d based on these results are given in Appendix B.3. For instance, by plotting the estimated cumulative proportion of variance explained by d factors against the candidate values of d, an optimal d can be found by looking for the "elbow" of the plot. The absence of an elbow in the plot potentially suggests that the data cannot be sufficiently summarized by a small number of factors.

3.4 Model Checking

A goodness-of-fit test is integral for the assessment of model fit. For Bayesian model checking, it is common to perform the posterior predictive approach described in Rubin (1984) and Gelman et al. (2013), which calculates the discrepancy between observed data and replicated data from the posterior predictive distribution $p(\boldsymbol{Y}^{\text{rep}} \mid \boldsymbol{Y}^{\text{obs}}) = p(\boldsymbol{Y}^{\text{rep}} \mid \boldsymbol{X}, \boldsymbol{\theta}) p(\boldsymbol{X}, \boldsymbol{\theta} \mid \boldsymbol{Y}^{\text{obs}})\, \mathrm{d}X\, \mathrm{d}\boldsymbol{\theta}$. For our model, it is particularly important to evaluate whether the inferred functional connectivity ($\widehat{\boldsymbol{\Lambda}}$) is representative of the coordinated activities of neurons. The posterior predictive approach does not explicitly assess this aspect of the model, since each individual neural activity Y_i^{rep} is largely determined by the marginal parameters (α_i, k_i). A cross-validatory model checking is better suited for assessment of inferred functional connectivity between neurons, such as the methods described in Marshall and Spiegelhalter (2007). Such diagnostic methods set aside the data of one unit and subsequently use the data of other units to predict the held-out unit. Leave-neuron-out (LNO) prediction is a well known approach for evaluating neural population model fit (Pei et al., 2021; Yu et al., 2009). In the training set, data from all neurons are fit to infer the model parameters, which are then used in the test set to predict the neural activity of a held-out neuron based on the activities of the held-in neurons. Predictability of held-out neuron indicates whether the model sufficiently characterizes the association between neural activities. When the computational cost of model fitting is high, leave-neuron-out prediction can be expensive due to having to fit the model repeatedly.

3.4.1 Leave-Neuron-Out Posterior Predictive Check

We propose a model checking procedure called *leave-neuron-out posterior predictive (LNO-PP) check*, a combination of leave-neuron-out prediction for neural latent variable models and Bayesian posterior predictive check. Suppose neuron i is held out for prediction, the LNO-PP check is described in Algorithm 1. Briefly, it constructs a LNO-PP distribution

$$\begin{aligned} p(\boldsymbol{Y}_i^{\text{pred}} \mid \boldsymbol{Y}^{\text{obs}}) &= \int p(\boldsymbol{Y}_i^{\text{pred}} \mid \boldsymbol{Y}^{\text{obs}}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{Y}^{\text{obs}}) \, \mathrm{d}\boldsymbol{\theta} \\ &\approx \int p(\boldsymbol{Y}_i^{\text{pred}} \mid \boldsymbol{Y}_{-i}^{\text{obs}}, \boldsymbol{\theta}) p(\boldsymbol{\theta} \mid \boldsymbol{Y}^{\text{obs}}) \, \mathrm{d}\boldsymbol{\theta} \end{aligned} \tag{3.20}$$

where $\boldsymbol{Y}_i^{\text{pred}}$ denotes the predicted spike trains of the left-out neuron i, $\boldsymbol{Y}_{-i}^{\text{obs}}$ the observed spike trains from all but the i-th neuron, and $\boldsymbol{Y}^{\text{obs}}$ the observed spike trains from all neurons. Specifically, $p(\boldsymbol{Y}_i^{\text{pred}} \mid \boldsymbol{Y}_{-i}^{\text{obs}}, \boldsymbol{\theta})$ in (3.20) corresponds to leave-neuron-out prediction using data excluding neuron i and parameters estimated from all data. We note that $p(\boldsymbol{\theta} \mid \boldsymbol{Y}^{\text{obs}})$, which is based on all data, cannot be replaced by $p(\boldsymbol{\theta} \mid \boldsymbol{Y}_{-i}^{\text{obs}})$ because $\boldsymbol{Y}_{-i}$ provides no information about neuron i. In practice, the LNO-PP distribution (3.20) is approximated by a large number of M samples. Given the samples $\boldsymbol{Y}_i^{\text{pred},1}, \ldots, \boldsymbol{Y}_i^{\text{pred},M}$ and a chosen test statistic $T(y)$, the LNO-PP check compares $T(\boldsymbol{Y}_i^{\text{obs}})$ to $\big(T(\boldsymbol{Y}_i^{\text{pred},1}), \ldots, T(\boldsymbol{Y}_i^{\text{pred},M})\big)$.

Algorithm 1 Leave-neuron-out posterior predictive (LNO-PP) check

Input: M, $p(\boldsymbol{\theta} \mid \boldsymbol{Y})$, $p(\boldsymbol{X}_i \mid \boldsymbol{\theta}, \boldsymbol{Y}_{-i})$ ▷ $\boldsymbol{Y}_{-i}$ excludes data from neuron i
Output: $T(\boldsymbol{Y})_i^{\text{pred},1}, \ldots, T(\boldsymbol{Y})_i^{\text{pred},M}$ ▷ Predicted test statistic for neuron i

for $m = 1, \ldots, M$ **do**

1. Simulate $\boldsymbol{\theta}^{\text{rep},m} \sim p(\boldsymbol{\theta} \mid \boldsymbol{Y})$.
2. Simulate $\boldsymbol{X}_i^{\text{pred},m} \sim p(\boldsymbol{X}_i \mid \boldsymbol{\theta}^{\text{rep},m}, \boldsymbol{Y}_{-i})$.
3. Determine $\boldsymbol{Y}_i^{\text{pred},m}$ from $\boldsymbol{X}_i^{\text{pred},m}$ and $\boldsymbol{\theta}_i^{\text{rep},m}$ via 3.3.
4. Compute test statistic $T(\boldsymbol{Y}_i^{\text{pred},m})$.

end for

Compare the observed test statistic $T(\boldsymbol{Y}_i^{\text{obs}})$ to the approximated posterior predictive distribution of the test statistic $\big(T(\boldsymbol{Y}_i^{\text{pred},1}), \ldots, T(\boldsymbol{Y}_i^{\text{pred},M})\big)$.

The LNO-PP check is equivalent to the Rubin (1984) posterior predictive check when replacing $\boldsymbol{Y}_{-i}$ by $\boldsymbol{Y}$ in step 2 of Algorithm 1. On the other hand, if $\boldsymbol{Y}$ is replaced by $\boldsymbol{Y}_{-i}$ in step 1, the LNO-PP check becomes the Marshall and Spiegelhalter (2007) cross-validatory posterior check, which cannot be applied to our case as $p(\boldsymbol{\theta} \mid \boldsymbol{Y}_{-i})$ provides no

information on predicting $\boldsymbol{Y}_i$. In practice, we replace the posterior in step 1 by its Normal approximation in (3.11) or (3.17), depending on whether joint or two-step estimation is used. For step 2, the posterior is also approximated by a Normal distribution with mean at $\widehat{E}(\boldsymbol{X}_i \mid \boldsymbol{\theta}^{\text{rep},m})$ and variance $\widehat{\text{Var}}(\boldsymbol{X}_i \mid \boldsymbol{\theta}^{\text{rep},m})$, both of which are found via gradient-based optimization over the Laplace approximated marginal likelihood described in Section 3.3.3.

Repeated calculation of $p(\boldsymbol{X}_i \mid \boldsymbol{\theta}, \boldsymbol{Y}_{-i})$ or its Normal approximation is expensive for large M, as optimization needs to be carried out in each iteration. Therefore, we propose a simplified LNO-PP (sLNO-PP) check which combines steps 1 and 2 and only performs optimization once at the beginning. The steps are given in Algorithm 2. The method for obtaining the joint approximate posterior $\widetilde{p}(\boldsymbol{\theta}, \boldsymbol{X}_i \mid \boldsymbol{Y}_{-i})$ is similar to the joint approximation of fixed and random effects (2.19) described in Chapter 2. Unless specified otherwise, model checking procedures in the paper are all performed using the sLNO-PP check.

Algorithm 2 Simplified leave-neuron-out posterior predictive (sLNO-PP) check

Input: M, $\widehat{\boldsymbol{\theta}}$, $\widehat{\boldsymbol{V}}_{\boldsymbol{\theta}}$, $p(\boldsymbol{X}_i, \boldsymbol{\theta} \mid \boldsymbol{Y}_{-i})$ ▷ $\boldsymbol{Y}_{-i}$ excludes data from neuron i
Output: $T(\boldsymbol{Y})_i^{\text{pred},1}, \ldots, T(\boldsymbol{Y})_i^{\text{pred},M}$ ▷ Predicted test statistic for neuron i

0. Approximate the joint posterior $p(\boldsymbol{\theta}, \boldsymbol{X}_i \mid \boldsymbol{Y}_{-i}) \approx \widetilde{p}(\boldsymbol{\theta}, \boldsymbol{X}_i \mid \boldsymbol{Y}_{-i})$ by a Normal.

for $m = 1, \ldots, M$ **do**

1. Simulate $\boldsymbol{\theta}^{\text{rep},m}, \boldsymbol{X}_i^{\text{pred},m} \sim \widetilde{p}(\boldsymbol{\theta}, \boldsymbol{X}_i \mid \boldsymbol{Y}_{-i})$.
2. Determine $\boldsymbol{Y}_i^{\text{pred},m}$ from $\boldsymbol{X}_i^{\text{pred},m}$ and $\boldsymbol{\theta}_i^{\text{rep},m}$ via (3.3).
3. Compute test statistic $T(\boldsymbol{Y}_i^{\text{pred},m})$.

end for

Compare the observed test statistic $T(\boldsymbol{Y}_i^{\text{obs}})$ to the approximated posterior predictive distribution of the test statistic $\big(T(\boldsymbol{Y}_i^{\text{pred},1}), \ldots, T(\boldsymbol{Y}_i^{\text{pred},M})\big)$.

3.4.2 Diagnostic Statistic

We present two statistics here, one being the cumulative spike count across time and the other being the spike times. For the first diagnostic, we plot the observed and the predicted cumulative spike count for the left-out neuron (from a single trial) across time against each other, where a good fit would result in the values lying approximately along the 45-degree line. This can be repeated M times to represent the M samples from Algorithm 1 or 2. Figure 3.3 shows examples of the graphical LNO-PP check results for a held-out neuron in a simulated dataset of 12 neurons simulated over the time window of length 5s (with 3 factors, 1000 time steps and 3 trials). Each grey line corresponds to one sample of LNO-PP

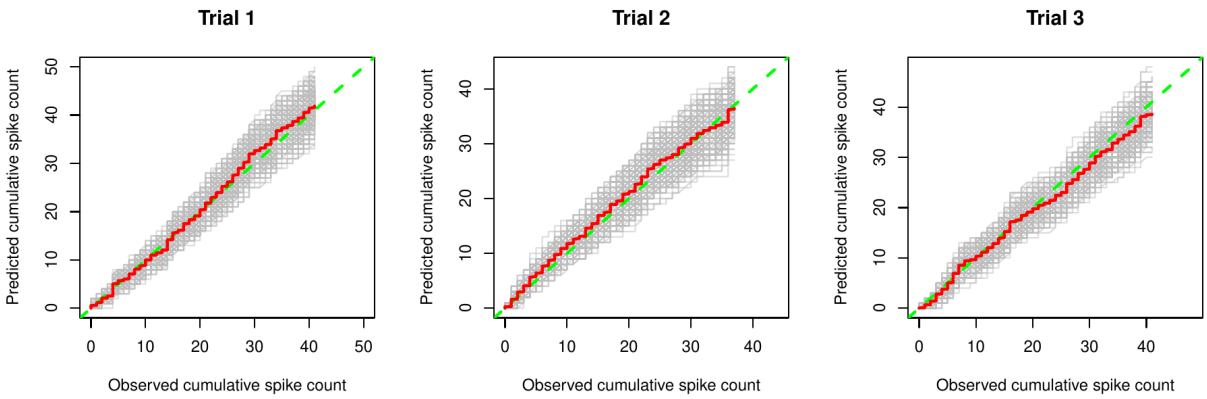


Figure 3.3: LNO-PP graphical diagnostic using the cumulative spike count as the statistic of interest for a held-out neuron in a 12-neuron simulated dataset, with each grey line representing one sample from the LNO-PP check algorithm.

predicted counts versus the observed counts across time. The red lines correspond to the average of all $M = 1000$ grey lines. The green dashed line corresponds to the 45-degree diagonal reference line. An example of LNO-PP diagostic for unsatisfactory model fit is given in Appendix B.2.1.

For the second diagnostic, we compare the spike time T_n^{obs} of the n-th spike from $\boldsymbol{Y}_i^{\text{obs}}$ to the distribution of the spike time T_n^{pred} of the n-th spike from $\boldsymbol{Y}_i^{\text{pred}}$. The comparison can be repeated for a number of randomly selected spike indices. Figure 3.4 illustrates such comparison using 5 randomly chosen spike indices. With the vertical dashed line indicating the observed spike time T_n^{obs} and each histogram showing 1000 samples of T_n^{pred} from the LNO-PP distribution, these plots show reasonable agreement between the assumed model and the observed data.

3.5 Simulation Studies

Two simulation studies were conducted to evaluate the performance of the proposed methods. The first study used data generated from the model itself to recover the model parameters, effectively showcasing the accuracy and efficiency of our inference procedure. Additionally, we explored the effect of the selected number of factors on the results. The second study leveraged the biologically plausible NEURON simulation environment (Hines and Carnevale, 1997) to simulate artificial neurons whose true connections are known. By

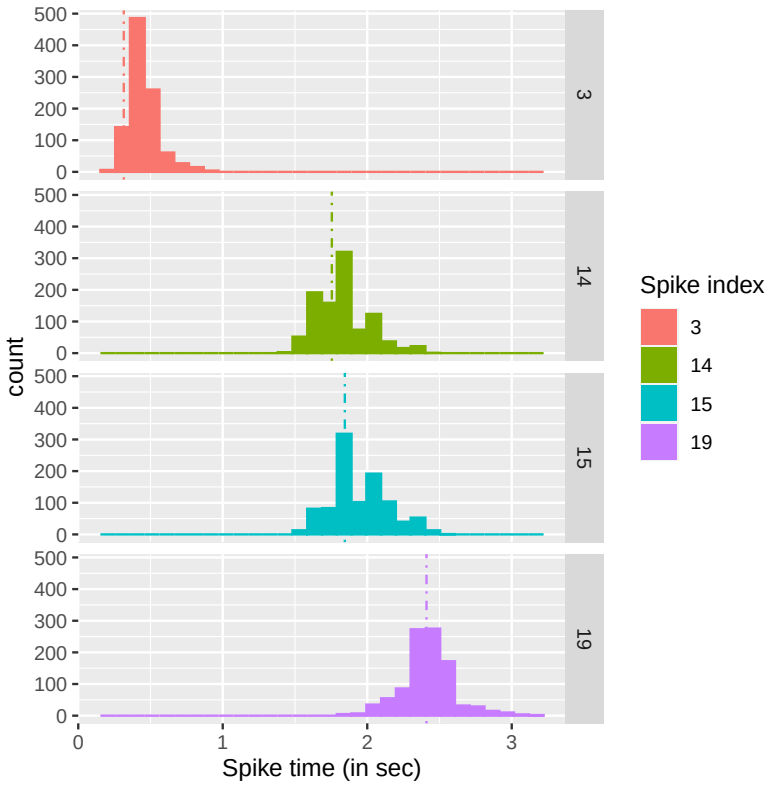


Figure 3.4: LNO-PP graphical diagnostic using spike time as a statistic for a held-out neuron in a 12-neuron simulated dataset, with each histogram representing samples of the n-th spike time from the LNO-PP distribution and the vertical dashed line corresponding to the observed n-th spike time.

fitting our model to the resulting spike trains, we are able to assess the model's performance in explaining the underlying neural connectivities. Furthermore, we demonstrate that neuronal clustering based on the estimated factor loadings from the model provides better cluster separation and more robust results than competing methods.

3.5.1 Simulated Data from the Proposed Model

We simulate $T = 10$ seconds of synthetic spike trains using $d = 4$ common factors for $q = 20$ neurons with bin size $\delta = 5$ms, a length consistent with common refractory periods for neurons. The drift parameters $\boldsymbol{\alpha}$, threshold parameters $\boldsymbol{k}$, and factor loadings are simulated from Uniform distributions as follows:

$$\alpha_i \overset{iid}{\sim} U(2, 5.5),\ i = 1, \ldots, 20, \tag{3.21}$$

$$k_i \overset{iid}{\sim} U(0.1, 0.6),\ i = 1, \ldots, 20, \tag{3.22}$$

$$\boldsymbol{\Lambda} \sim \begin{pmatrix} \boldsymbol{U(0.7, 0.95)} & U(0.1, 0.25) & U(0.1, 0.25) & U(0.1, 0.25) \\ \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{U(0.7, 0.95)} & U(0.1, 0.25) & U(0.1, 0.25) & U(0.1, 0.25) \\ U(0.1, 0.25) & \boldsymbol{U(0.7, 0.95)} & U(0.1, 0.25) & U(0.1, 0.25) \\ \vdots & \vdots & \vdots & \vdots \\ U(0.1, 0.25) & \boldsymbol{U(0.7, 0.95)} & U(0.1, 0.25) & U(0.1, 0.25) \\ U(0.1, 0.25) & U(0.1, 0.25) & \boldsymbol{U(0.7, 0.95)} & U(0.1, 0.25) \\ \vdots & \vdots & \vdots & \vdots \\ U(0.1, 0.25) & U(0.1, 0.25) & \boldsymbol{U(0.7, 0.95)} & U(0.1, 0.25) \\ U(0.1, 0.25) & U(0.1, 0.25) & U(0.1, 0.25) & \boldsymbol{U(0.7, 0.95)} \\ \vdots & \vdots & \vdots & \vdots \\ U(0.1, 0.25) & U(0.1, 0.25) & U(0.1, 0.25) & \boldsymbol{U(0.7, 0.95)} \end{pmatrix}. \tag{3.23}$$

The parameters are chosen such that the firing rates are between 3 to 33Hz. The structure of $\boldsymbol{\Lambda}$ in (3.23) groups the 20 neurons into 4 clusters each containing 5 neurons. To simulate spike trains from the time-discretized model described in (3.7), we first simulate the multivariate latent path increments $d\boldsymbol{x}_j \mid \boldsymbol{\alpha}, \boldsymbol{\Lambda} \overset{\text{ind}}{\sim} \mathcal{N}(\delta\boldsymbol{\alpha}, \delta(\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}))$, where $\boldsymbol{\Psi}$ is a diagonal matrix with entries defined by the relation $\text{diag}(\boldsymbol{\Lambda}\boldsymbol{\Lambda}^T + \boldsymbol{\Psi}) = \mathbf{1}$. Next, spikes are generated based on threshold crossing of the simulated latent paths $\boldsymbol{X}$ in each bin as $y_{ij} = \mathbb{1}\big(x_{ij} \geq k_i(1 + \sum_{r=0}^{j-1} y_{ir})\big)$, where $x_{ij} = \sum_{r=1}^{j} dx_{ir}$ is the latent path state of neuron i in bin j and the initial spike indicator $y_{i0} = 0$. The sigmoid scaling factor s is set to 15

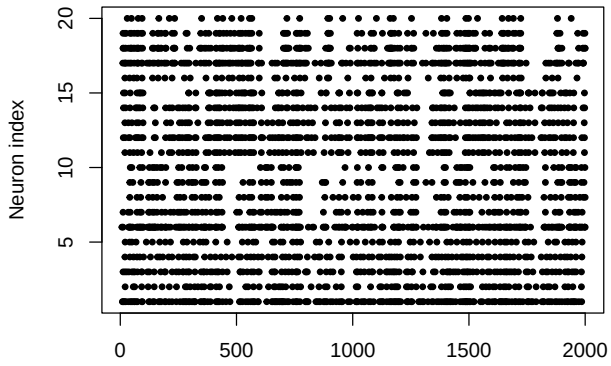


Figure 3.5: Rasterplot of 20-neuron spike trains simulated from the proposed model.

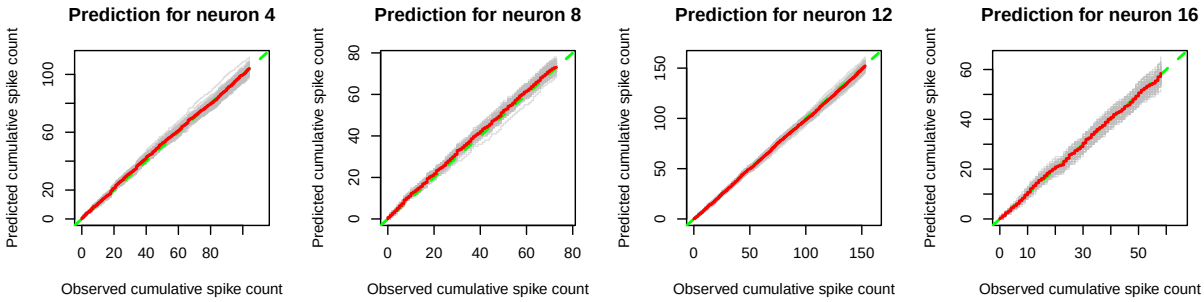


Figure 3.6: Model checking results using the LNO-PP check. Close agreement is observed between the predicted and the observed spike counts as the data is simulated from the model. Each corresponds to 100 draws from the posterior predictive distribution, leaving neurons 4, 8, 12, and 16 out for prediction, respectively.

(see equation (3.7)). A rasterplot of the simulated spike trains is given in Figure 3.5. With the simulated data at hand, we fit the model to the simulated spike trains $\boldsymbol{Y}$ assuming that the number of factors d is known. We will explain later the process of selecting d. The LNO-PP check described in Section 3.4 is employed for model checking in Figure 3.6. Table 3.1 summarizes the computation times for joint estimation (3.11) with $d = 4$ and two-step estimation (3.17) with $d = 2, 4, 6$ on an Ubuntu 22.04 machine with i7-9700 CPU and 16Gb of RAM.

Figure 3.7a shows that the estimated firing rates $\widehat{\mu^{-1}} = \widehat{\alpha/k}$ align closely with their true values for each neuron. Superimposed on each dot is the 95% credible interval. For the true loading matrix $\boldsymbol{\Lambda}$ used for simulation, its corresponding estimate $\widehat{\boldsymbol{\Lambda}}$ can be brought to

	Joint Estimation	Two-Step Estimation		
	d=4	d=2	d=4	d=6
Run time in minutes	51	6	22	42

Table 3.1: Run times for fitting the proposed model to simulated data under different inference settings.

a simpler structure by applying the Varimax transformation. This transformation seeks an orthogonal rotation on $\widehat{\mathbf{\Lambda}}$ such that the sum of the variances of the squared loadings between factors is maximized, leading to more near-zero entries and hence a more interpretable matrix structure in $\widehat{\mathbf{\Lambda}}_{\text{Varimax}}$ (Kaiser, 1958). Note that the order of the factors, namely the column permutation of $\widehat{\mathbf{\Lambda}}$, are not preserved after Varimax transformation. This is not a concern for our analysis since we later use $\widehat{\mathbf{\Lambda}}$ for neuron clustering instead of interpreting the meaning of each factor. The true $\mathbf{\Lambda}$ and Varimax-transformed estimate $\widehat{\mathbf{\Lambda}}_{\text{Varimax}}$ are plotted in Figure 3.7c and 3.7d, respectively. We see that $\widehat{\mathbf{\Lambda}}$ recovers $\mathbf{\Lambda}$ well (up to column permutations) and clearly reveals the 4 clusters in the simulated spike trains.

It is natural to wonder whether the same clustering pattern can be observed at the data level. To check this, we first smooth the spike trains by convolving the spike train for neuron i, $\{y_{ij}\}_{j=1,\dots,n}$, with an exponentially decaying kernel (see Figure 3.8a), a common smoothing approach in the literature (see, e.g., Gerstein et al., 1985; van Rossum, 2001; Fellous et al., 2004; Paiva et al., 2007; Jurjuţ et al., 2019). The convolved data $y^c_{i,j}$ is given by

$$y^c_{i,j} = \begin{cases} y^c_{i,j-1} + 1, & \text{if } \ y_{i,j} = 1, \\ e^{-\frac{1}{5}} \cdot y^c_{i,j-1}, & \text{otherwise,} \end{cases} \tag{3.24}$$

Then we compute the cross-correlation between each pair of neurons using the convolved spike train, with all the non-significant values set to 0 before being plotted in Figure 3.8b. The correlation matrix plot is ordered by hierarchical clustering and accompanied by a dendrogram to visualize the clusters. Overall, a higher value of cross-correlation is observed for neurons belonging to the same cluster. The correlation at the data level is less prominent for neurons 1-5, whereas the factor model can clearly identify their connections as a common input source. Using our proposed factor model, a correlation matrix estimate can be approximated by $\widehat{\mathbf{\Sigma}} = \widehat{\mathbf{\Lambda}}\widehat{\mathbf{\Lambda}}^T + \widehat{\mathbf{\Psi}}$. A dendrogram is plotted in Figure 3.8c for head-to-head comparison with the dendrogram simply based on convolved spike trains, showing much clearer cluster separation based on factor loading estimates. This demonstrates that our proposed model provides a better means for estimating the correlations between

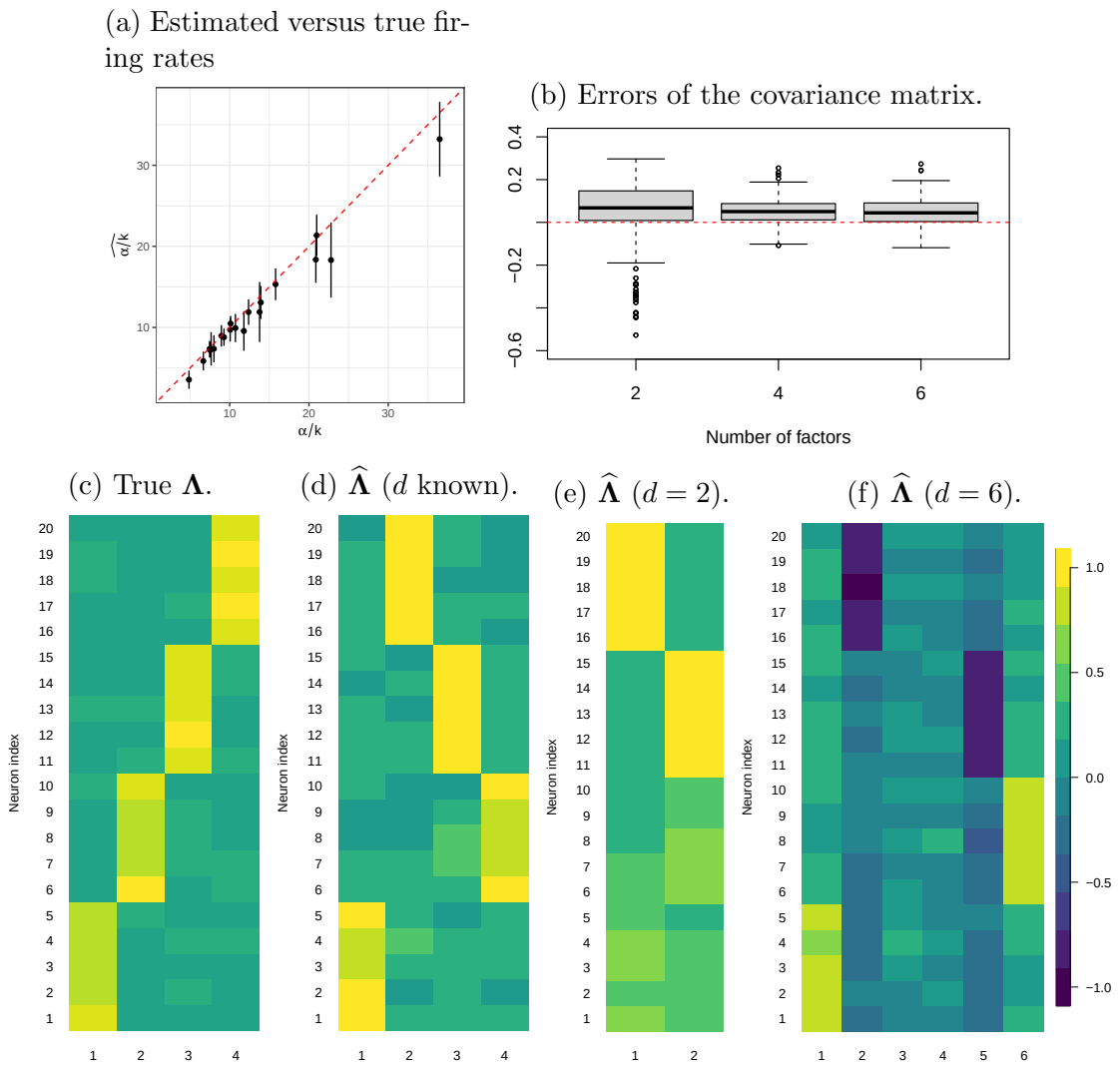


Figure 3.7: (a): Estimated firing rates $\widehat{\alpha_i/k_i}$ plotted against their true values. Each dot corresponds to one neuron. The vertical error bar corresponds to the 95% confidence interval. (b): Box plots of the errors of the covariance matrix $\widehat{\mathbf{\Sigma}} = \widehat{\mathbf{\Lambda}}\widehat{\mathbf{\Lambda}}^T + \widehat{\mathbf{\Psi}}$ for models with different number of factors. (c): True factor loading matrix $\mathbf{\Lambda}$. (d): Varimax-transformed estimated factor loading matrix $\widehat{\mathbf{\Lambda}}_{\text{Varimax}}$ when $d = d^* = 4$. (e): Varimax-transformed estimated factor loading matrix when $d = 2 < d^*$. (f): Varimax-transformed estimated factor loading matrix when $d = 6 > d^*$. For (c)-(f), the columns correspond to the factors and the rows correspond to the neurons.

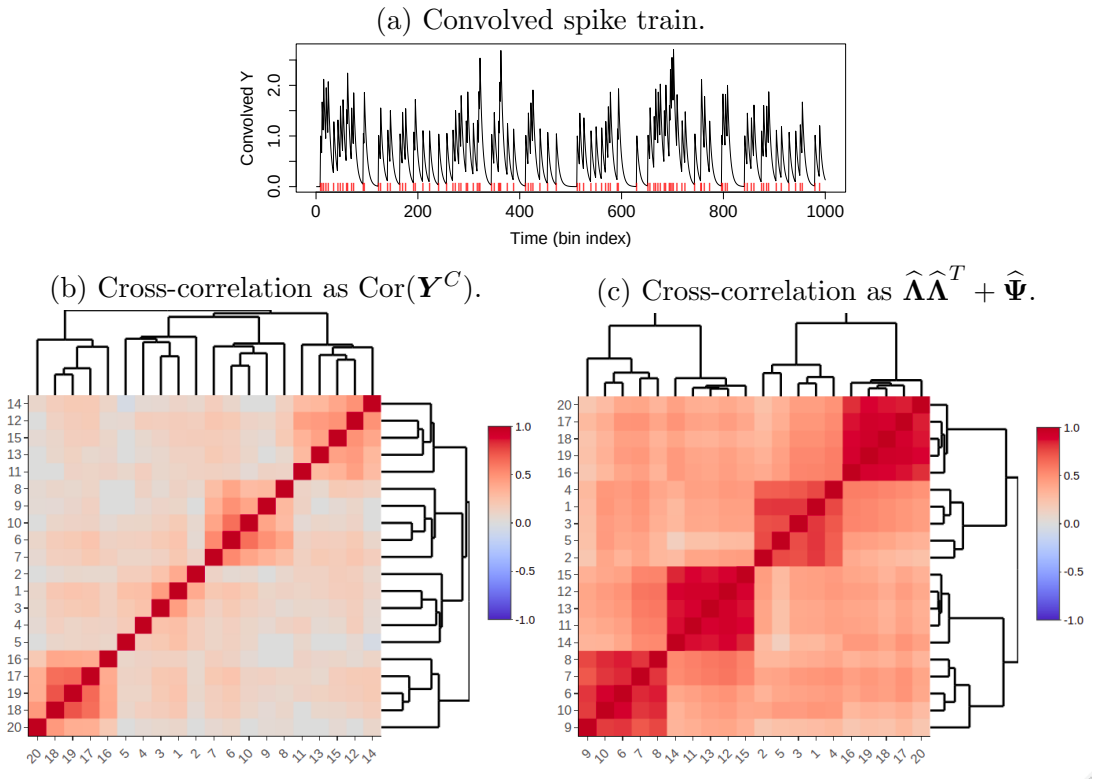


Figure 3.8: (a) Simulated spike train convolved with an exponentially decaying kernel. (b) Cross-correlation between pairs of convolved spike trains at lag 0 accompanied by the dendrogram based on hierarchical clustering. The non-significant correlation coefficients are set to zero. (c) Cross-correlation based on loading matrix estimates from our model.

neurons.

The exploratory factor analysis method described in Section 3.3.6 is employed to choose the number of factors d for model inference. We examine the estimate of the cumulative proportion of variance explained by all d factors. After repeating this for a sequence of values $d = \{1, \ldots, 10\}$, we plot the cumulative variances explained against each choice of d in Figure 3.9. The optimal value of d is at the "elbow" of this plot, which for Figure 3.9 occurs at $d = 4$. This number coincides with the true number of factors d^* used for simulating spike trains. More details on the selection of d are given in Appendix B.3.

To examine how the factor model performs when d is misspecified, we consider two scenarios: one with $d = 2$ and the other with $d = 6$. The estimated loading matrices

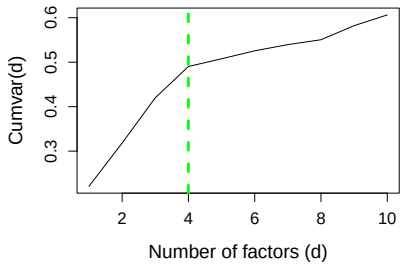


Figure 3.9: Total variance explained by the factors as a function of the number of factors (d) chosen.

(after Varimax transformation) are presented in Figure 3.7e and 3.7f. When $d = 2$, two clusters, neurons 11-15 and neurons 16-20, can be clearly identified as they load heavily on factors 2 and 1, respectively. Neurons 1-10 load lightly on both factors, leading to no clear separation of clusters by visual inspection of the loading matrix. The true and estimated idiosyncratic components, i.e., diagonal elements of $\mathbf{\Psi}$, corresponding to neurons 1-10 are given below:

$$\mathbf{\Psi}_{1:10} = (0.16, 0.32, 0.32, 0.30, 0.28, 0.01, 0.32, 0.36, 0.36, 0.16)$$
$$\widehat{\mathbf{\Psi}}_{1:10} = (0.50, 0.70, 0.55, 0.51, 0.73, 0.57, 0.50, 0.55, 0.73, 0.67).$$

The idiosyncratic values being closer to 1 than to 0 indicate that the two present factors cannot sufficiently summarize their activities and that more factors are needed. When $d = 6$, factors 3 and 4 both have very small loadings and can be considered extra. The other four factors are representative of the activities of the four neural clusters. Similar observations of extra factors are also made for the cases of $d = 8$ and $d = 10$ (see Appendix B.4). Recall that the covariance matrix is estimated by $\widehat{\mathbf{\Sigma}} = \widehat{\mathbf{\Lambda}}\widehat{\mathbf{\Lambda}}^T + \widehat{\mathbf{\Psi}}$ and the errors of the covariance matrix is defined as $\hat{\sigma}_{uv} - \sigma_{uv}$, where σ_{uv} is the (u, v)-th element of $\mathbf{\Sigma}$. Inspecting the errors displayed in Figure 3.7b, the model with $d > d^*$ (right) performs as well as the model with the true number of factors (middle). In conclusion, using $d < d^*$ can lead to a loading matrix that obscures part of the clustering structure of the neurons, whereas using $d > d^*$ does not undermine the model's ability to detect such structure. However, Figure 3.7b also suggests that the model tends to slightly overestimate the correlation between neurons. As for the run time (summarized in Table 3.1), while fitting the model with $d = 2$ took 6 minutes, fitting $d = 6$ took 42 minutes, which is twice as long as it took for fitting $d = 4$. This suggests the consideration of speed and accuracy trade-off when choosing d.

Besides the use for approximating the correlation matrix in data analysis (e.g. see real data analysis in Section 3.6), the estimated factor loadings offer a more meaningful biological interpretation. For example, if a group of neurons load heavily on two distinct factors, these neurons might be connected to two different driving forces for spiking, such as two sources of stimuli or two different communities of other neurons. This will be further discussed in the second simulation study in Section 3.5.2. Importantly, the factor model provides a framework for analyzing the association between neurons with multiple possibilities for downstream analysis such as clustering and correlation analysis.

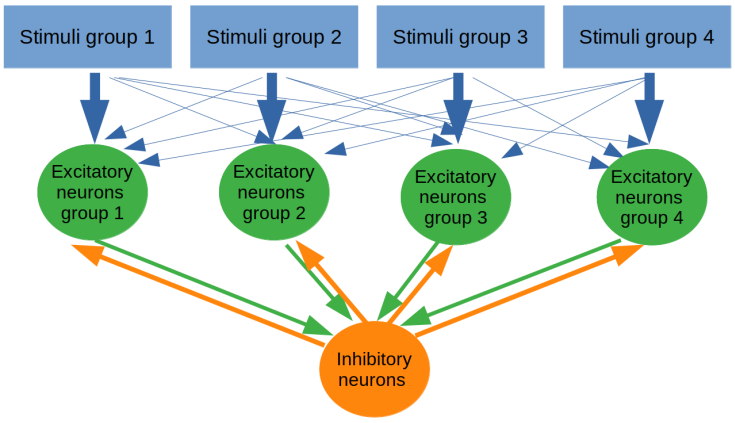
3.5.2 Data Simulated from NEURON Software

NEURON (Hines and Carnevale, 1997) is a software which has been widely adopted in the neuroscience community to simulate artificial spiking neurons. Using the leaky integrate-and-fire model implemented in NEURON, we simulate a large neuronal network consisting of 4000 excitatory neurons and 1000 inhibitory neurons. Excitatory neurons send excitatory inputs to other neurons, and inhibitory neurons transmit inhibitory inputs. Connections are also formed within the excitatory population and inhibitory population. The network is stimulated by 500 Poisson spike generators (with 50Hz rates) connecting to the excitatory neurons. We artificially create 4 clusters each containing 1000 excitatory neurons by letting neurons in the same cluster receive the majority of stimuli from a common group of spike generators. The network is illustrated in Figure 3.10a. Python code for simulating the network is adapted from that of Case Study 1 in Tikidji-Hamburyan et al. (2017).

From the simulated neuronal network, we randomly sample 46 neurons and record their spiking activities for 10s. Figure 3.10b shows the raster plot of the neurons. The proposed model is fit to the spike trains of the 46 neurons in the sample. The number of factors is chosen to be 6 based on the exploratory factor analysis detailed in Appendix B.3. Since the connections between neurons in the network are known, we can examine whether the proposed model can reveal the true neural connections. Since the focus of this simulation study is on inferring connections between neurons, the estimation of firing rates is omitted in this section.

Figure 3.11a and 3.11b present two correlation plots with dendrograms using convolved data obtained by (3.24) and $\widehat{\mathbf{\Sigma}} = \widehat{\mathbf{\Lambda}}\widehat{\mathbf{\Lambda}}^T + \widehat{\mathbf{\Psi}}$, respectively. We see that hierarchical clustering based on data could not separate between neuron groups except for the excitatory neurons indexed 37-46. On the other hand, the dendrogram based on our model-estimated $\widehat{\mathbf{\Sigma}}$ shows better, though still not very clear, cluster separation. In the following, we show that the loading matrix estimate provides a more meaningful interpretation on the structural

(a) Illustration of the neuronal network



(b) Raster plot

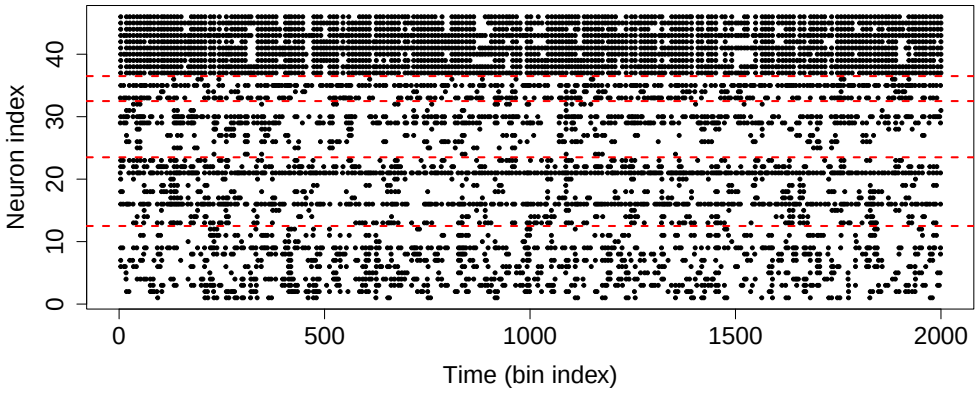


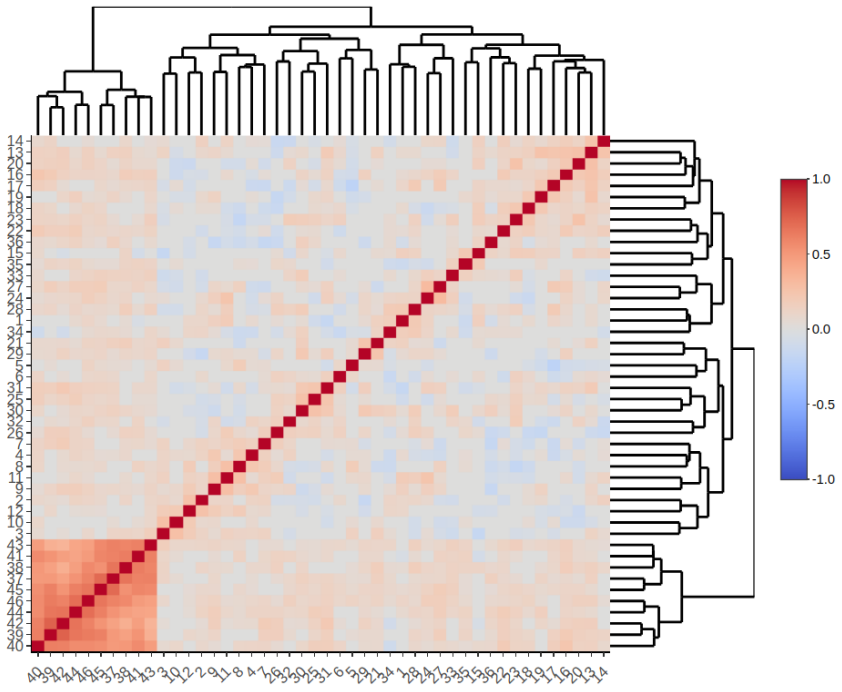
Figure 3.10: (a) Blue rectangle: a group of 125 Poisson spike generators. Green circle: a cluster of 1000 excitatory neurons. Orange circle: all 1000 inhibitory neurons. Blue arrow: stimuli from a group of Poisson spike generators to a cluster of excitatory neurons. Green arrow: excitatory inputs from a cluster of excitatory neurons to the inhibitory neurons. Orange arrow: inhibitory inputs from inhibitory neurons to a cluster of excitatory neurons. (b) Raster plot of the 46 sampled neurons simultaneously recorded for 10s. The red horizontal dashed lines divide the neuron clusters. The neurons are ordered by the true cluster membership. Each horizontal dotted line corresponds to the spike train of one neuron. The top rows with the most spikes correspond the cluster of inhibitory neurons, while the rest of the four clusters are all excitatory neurons.

association between neurons.

The estimated loading matrix $\widehat{\mathbf{\Lambda}}$ with Varimax rotation applied is presented in Figure 3.12a, where the neurons are ordered by hierarchical clustering, namely the ordering of neurons labels in Figure 3.11b. To check whether the ordering of neurons in Figure 3.12a reflects the true underlying clustering of the neurons, presented in Figure 3.12b is a re-ordered $\widehat{\mathbf{\Lambda}}$ with the neurons ordered by the true cluster memberships and with black dashed lines dividing the true clusters. It is found that the ordering of neurons in Figure 3.12b is mostly preserved in Figure 3.12a. In Figure 3.12b, neurons in clusters 1-4 correspond to the four groups of excitatory neurons, whereas neurons in cluster 5 are the inhibitory neurons. Neurons in cluster 1 load most heavily on factor 1 except neuron 6, while those in cluster 2 load more heavily on factor 3. Most of the neurons in cluster 3 have heavier loadings on factor 4, and cluster 4 seems to load on both factor 4 and 6. Thus, factors 1, 3 and 4 can be thought of as representations of the three stimuli groups that provide the most stimulation to clusters 1, 2 and 3. The small loadings on factor 2 suggest that this factor might be extra, which is consistent with the artificially created network structure of five clusters. Interestingly, $\widehat{\mathbf{\Lambda}}$ shows that the inhibitory neurons, all of which are in cluster 5, are mainly associated with factors 1, 3, 4 and 6 with moderate to large loadings. This reveals the connections of the inhibitory neurons with the four groups of excitatory neurons, as depicted in Figure 3.10a. Therefore, $\widehat{\mathbf{\Lambda}}$ provides a sensible structural interpretation of the neuronal network.

Next, we apply our model to the neuron clustering task, a particularly important component of the downstream analysis to be facilitated by our spike train model. For clustering, we apply the K-means algorithm to $\widehat{\mathbf{\Lambda}}$ representing the lower dimensional latent space. We refer to this clustering procedure as the K-means-$\mathbf{\Lambda}$ method. Three other clustering algorithms are presented here for comparison. The first one is the direct K-means method which applies the generic K-means algorithm on smoothed spike trains by first convolving them with a decaying exponential kernel (van Rossum, 2001; Fellous et al., 2004; Paiva et al., 2007). The second one is the spike-train community detection method introduced by Humphries (2011), which is a clustering algorithm specific to neural spike trains with a MATLAB implementation provided on the author's GitHub page. Its results are based on the spike times for each neuron instead of the binned binary sequences. The third one is clustering based on the inferred loading matrix from the popular Gaussian process factor analysis (GPFA) by Yu et al. (2009), which similarly projects the high-dimensional spike train onto a low-dimensional latent space via a factor loading matrix $\mathbf{\Lambda}_{\text{GPFA}}$. Notably, GPFA employs a Gaussian likelihood on the square-rooted spike counts in consecutive time bins, so it is considered a discrete-time model as opposed to our continuous-time setting. The clustering results via GPFA are obtained similarly by applying K-means to $\widehat{\mathbf{\Lambda}}_{\text{GPFA}}$,

(a) Cross-correlation as $\mathrm{Cor}(\boldsymbol{Y}^c)$



(b) Cross-correlation as $\widehat{\boldsymbol{\Lambda}}\widehat{\boldsymbol{\Lambda}}^T + \widehat{\boldsymbol{\Psi}}$

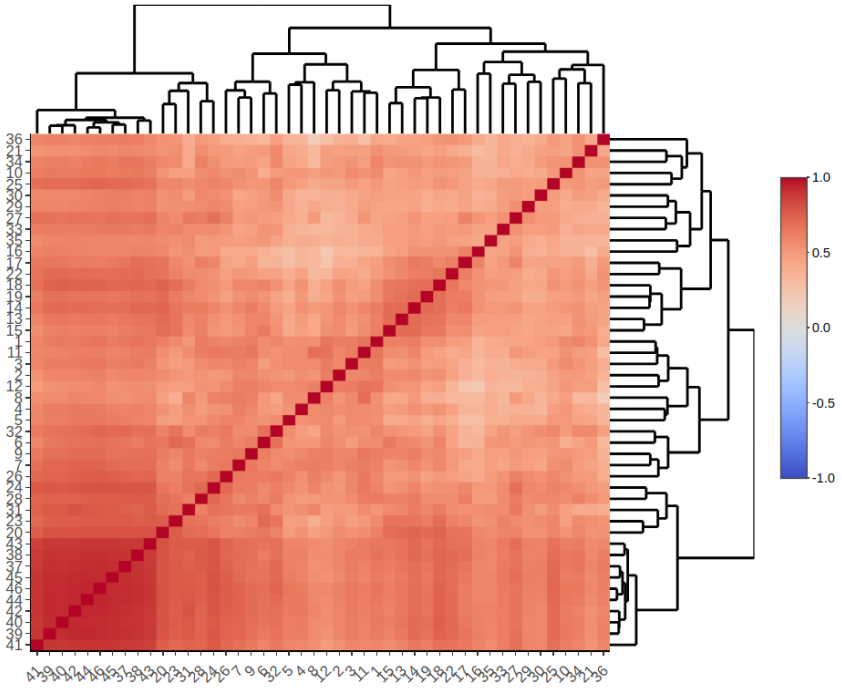


Figure 3.11: Correlation matrix plots and dendrograms obtained from (a) convolved data and (b) loading matrix estimates from the model, using data simulated from NEURON.

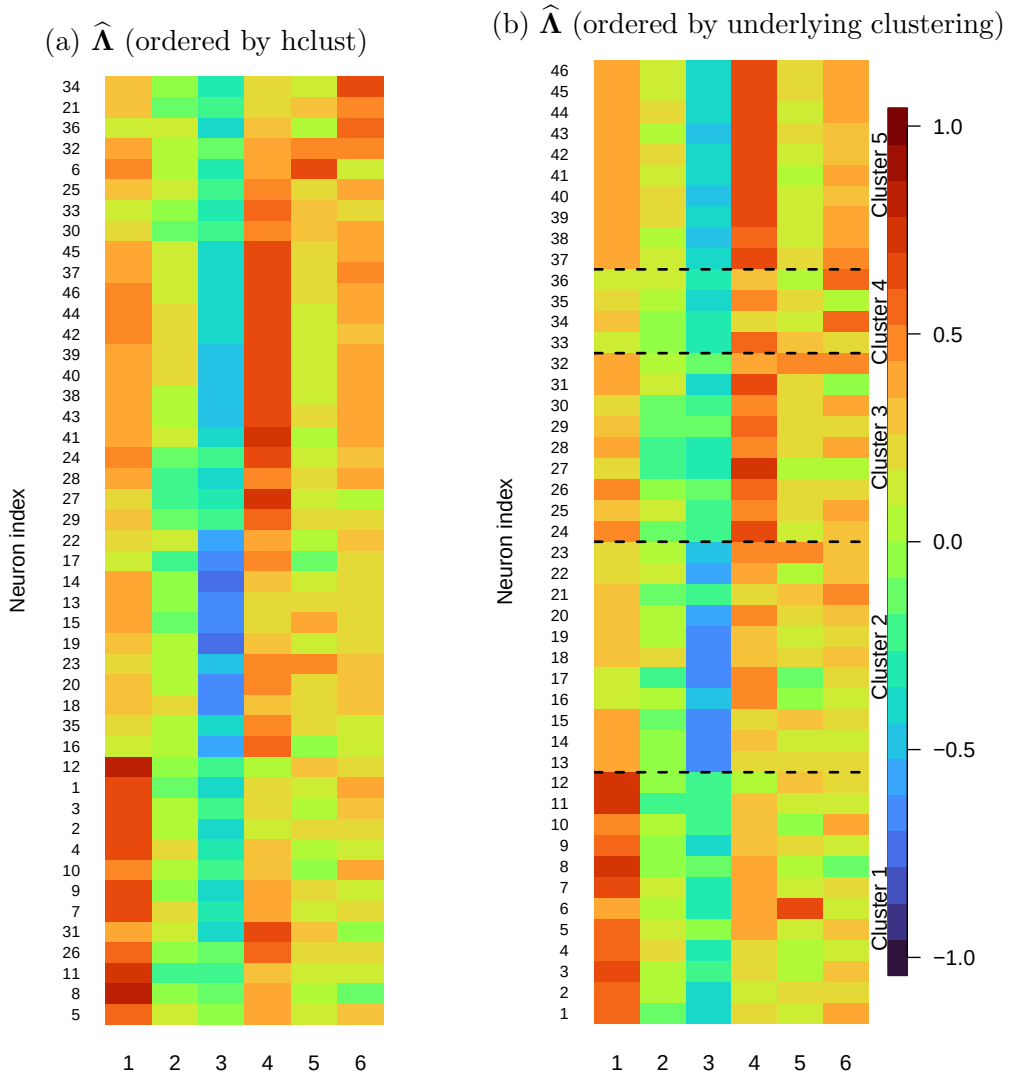


Figure 3.12: (a) Estimated factor loading matrix $\widehat{\mathbf{\Lambda}}$ with rows ordered by hierarchical clustering. (b) Estimated $\widehat{\mathbf{\Lambda}}$ with rows ordered by the true underlying clustering shown in Figure 3.10a, with the true clusters divided by the dashed lines. The columns correspond to the factors and the rows correspond to the neurons. Varimax rotation has been applied for both (a) and (b).

which is referred to as the K-means-GPFA method. Since GPFA requires time binning, three different choices of bin widths are presented.

The performance of each method is evaluated using a cluster membership matrix, which is visualized in Figure 3.13 with coloured/white entries of the heatmap indicating that neurons are in/not in the same cluster. For K-means-$\mathbf{\Lambda}$, K-means-GPFA and K-means-Direct, the number of neuronal clusters, K, needs to be specified beforehand. This choice is embedded in the Humphries algorithm, which picks $K = 7$ based on a benefit function. To ensure a fair comparison, we repeat each method except for the community detection algorithm for $K = 5$ and $K = 7$, as 5 is the true number of clusters. While K-means-$\mathbf{\Lambda}$ took over an hour to complete on the same machine described in Section 3.5.1, K-means-Direct was instantaneous, and both GPFA and the community detection algorithm took less than a minute.

Inspecting Figure 3.13(b) and (e), we find that K-means-$\mathbf{\Lambda}$ can identify the true clusters of neurons to a reasonable accuracy. Both K-means-Direct and the community detection algorithm tend to incorrectly group more neurons belonging to different clusters together. Consequently, the clusters are not well-separated. Neuron clusters produced by K-means GPFA under different bin widths are substantially different, underlining the impact of bin size on model parameter estimation for discrete-time models (Nelson, 2002; Kass and Ventura, 2006; Ramezan et al., 2014). In contrast, our continuouFs-time model does not demand an arbitrary choice of bin size and thus provides more robust clustering results. Additional comparisons between our model and GPFA in terms of goodness-of-fit are described in Appendix B.2.2.

Though more computational resources are required for K-means-$\mathbf{\Lambda}$, not only does it provide better separation of the neuronal clusters, but it also provides interpretable quantification of the association between neurons in the form of $\widehat{\mathbf{\Lambda}}$ in Figure 3.12b. Such results can potentially illuminate the biological properties of a large neuron ensemble, which is typically lacking in methods that are designed only for clustering based on observed neural activities.

The two simulation studies have exhibited several desired properties of the proposed model. When fit to data simulated from the model, the individual firing rates can be estimated accurately with a high precision provided the number of spikes is adequately large for each neuron. This is because the estimation is carried out marginally for each neuron using its interspike intervals, hence justified by the central limit theorem. The loadings are recovered accurately up to the permutation of the columns if the number of factors $d = d^*$ is correctly specified. Since d is rarely known beforehand, our sensitivity analysis shows that the interpretation of $\widehat{\mathbf{\Lambda}}$ remains true when using $d > d^*$. Therefore,

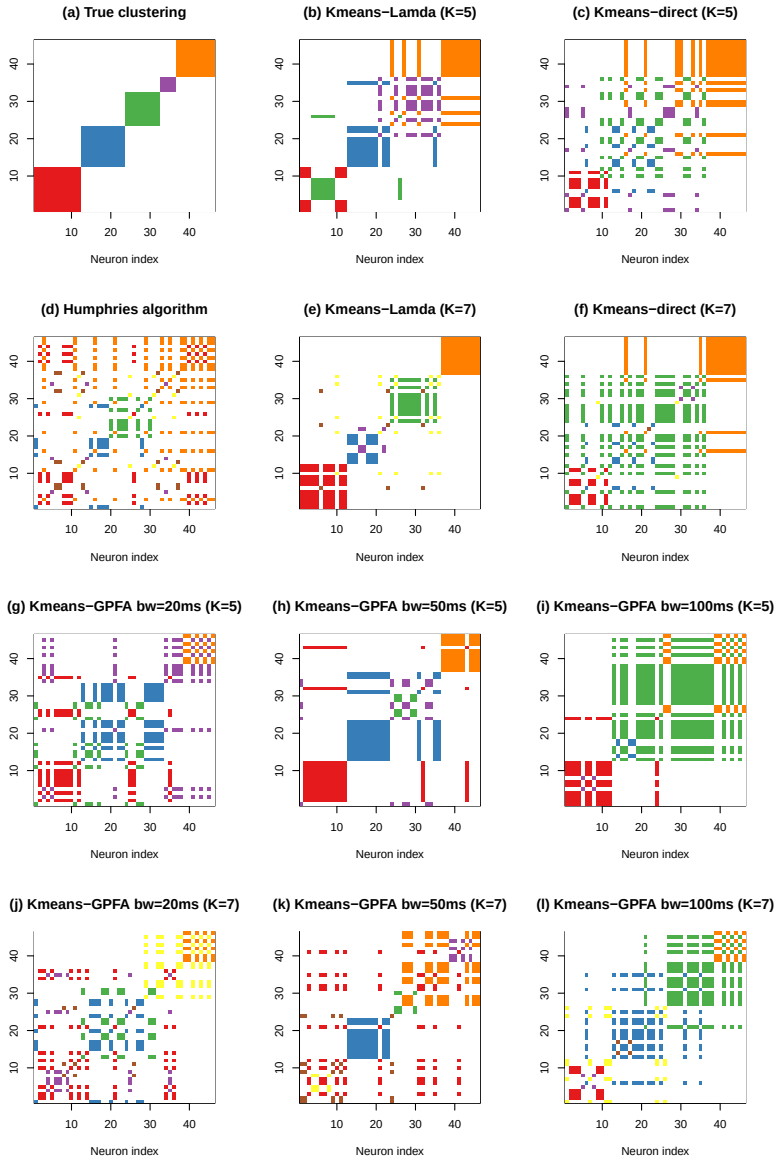


Figure 3.13: Visualization of the cluster membership matrices obtained using different methods. The (i, j)-th entry of the matrix is 1 (colour-coded by cluster membership) if neuron i and j are assigned to the same cluster, and 0 (white) otherwise. (b) and (e): Cluster memberships obtained by applying K-means on $\widehat{\mathbf{\Lambda}}$ estimated using our model. (c) and (f): Cluster memberships given by direct K-means applied on convolved spike trains. (d): Cluster memberships given by the Humphries (2011) spike-train community detection algorithm. (g)-(l): Cluster memberships obtained by applying K-means on $\widehat{\mathbf{\Lambda}}$ estimated using the GPFA with different bin sizes.

it is recommended that one make an initial guess for d, denoted by d_0. Then using the exploratory factor analysis algorithm described in Appendix B.3, select a larger value $d > d_0$ for which the available computing resources allow. When applied to data generated from a different computational (but biologically plausible) model, the proposed model can uncover the underlying associations between neurons. This provides evidence of the validity of the model output, which is crucial as it supports the results of the real data analysis in Section 3.6 and functional connectivity. Finally, model comparisons in this section demonstrate more superior performance of our model in a clustering task. This highlights that the factor loading estimates from our model are helpful for downstream analyses.

3.6 Neuronal Functional Connectivity in Rat OFC

Both human and animal studies show evidence of the orbitofrontal cortex (OFC) encoding value-related cues, representing either positive or negative outcome such as rewards/punishments (Rolls and Grabenhorst, 2008; Mainen and Kepecs, 2009; Sul et al., 2010). Recent studies (Moorman, 2018; Hernandez and Moorman, 2020) have found that OFC activities are associated with alcohol-seeking behaviours and alcohol cues, whose predictive value cannot be clearly categorized as positive or negative. In this section, we examine experimental data collected from rats' OFC to explore how their brain activities respond to three types of cues: surcrose cues, quinine cues, and ethanol cues. While sucrose is generally considered an appetitive outcome and quinine an aversive one, the role of ethanol perceived by rats remains unclear. However, human studies have found that alcohol abuse results in a strong neural response to alcohol-related cues (Heinz et al., 2004; Myrick et al., 2004). Utilizing the proposed model, we examine how cues are associated with group firing rates and FC of neurons in rats' OFC.

3.6.1 Data Collection

The subjects of this study are rats that have previously been trained to predict outcomes based on the cue they received. An intraoral catheter attached to the rat's mouth directly delivered a drop of liquid to its mouth following separate cues, as illustrated in Figure 3.14a. The cued outcomes are sucrose, quinine, or ethanol, each associated with a different ringtone (cue) played to them about 1.7 seconds before the delivery. Details about the training and the experiment setup can be found in Kaminska (2020).

Our data analysis focuses on spike trains simultaneously recorded from the OFC of 6 rats. Each rat underwent several sessions of experiments. A session consists of repeated trials of each of the three cues. A trial refers to the time window including 2 seconds before and 2 seconds after a sucrose/quinine/alcohol cue is played to the rat. All trials within the same session recorded the same group of neurons simultaneously through a surgically implanted array of tetrodes, whereas a different group of neurons were sampled from the OFC in another session. Figure 3.14b summarizes the number of sessions recorded for each animal and the number of neurons per session. An illustration of the data structure is given in Figure 3.14c.

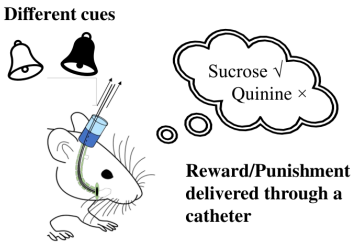
3.6.2 Metrics for Comparing Neuronal Activities

The primary objective of the analysis is to assess the change of group firing rates and FC between neurons in response to a cue. However, the presence of potentially unobserved effects from the environment and the inherent stochastic nature of neuronal activities mean that neuronal behaviour is constantly fluctuating. To determine whether the estimated changes are meaningful consequences of the cues, it is crucial to compare the cue-induced changes to baseline neuronal behaviour without the influence of a cue. Additionally, the heterogeneity between animals and sessions presents significant challenges in aggregating results. Therefore, the analysis can greatly benefit from interpretable and easily calculable metrics to summarize the model estimates. In the following, we describe a statistical analysis protocol which compares the cue-induced change to the baseline variations in neuronal activities, while accounting for between-session and between-animal variability.

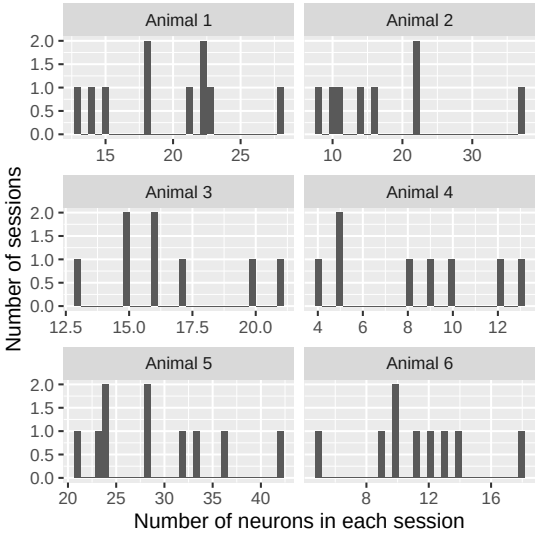
We consider three types of spike trains, each 2 seconds in length: from $t+2$ to $t+4$ seconds before the cue (baseline 1), from t to $t+2$ seconds before the cue (baseline 2), and from 0 to 2 seconds after cue (post-cue), where $t \geq 0$. The first two time windows serve as baselines, and the third is the experimental window during which neurons might react to the cue. A pair of baseline 1 and baseline 2 spike trains, adjacent in time, is considered a baseline pair. In contrast, an experiment pair consists of the baseline 2 spike train immediately before the cue and the post-cue spike train. See Figure 3.14d for an illustration of this design. For each animal and session combination, where the same group of neurons is studied under different cues, we extract 6 baseline pairs, and 3 experiment pairs each corresponding to one of the 3 cue conditions. We fit our model to the spike train in each of these time windows. The number of factors was chosen based on exploratory factor analyses (see Section 3.3.6 and Appendix B.3).

To quantify the difference between post-tone versus baseline neuronal activities, we

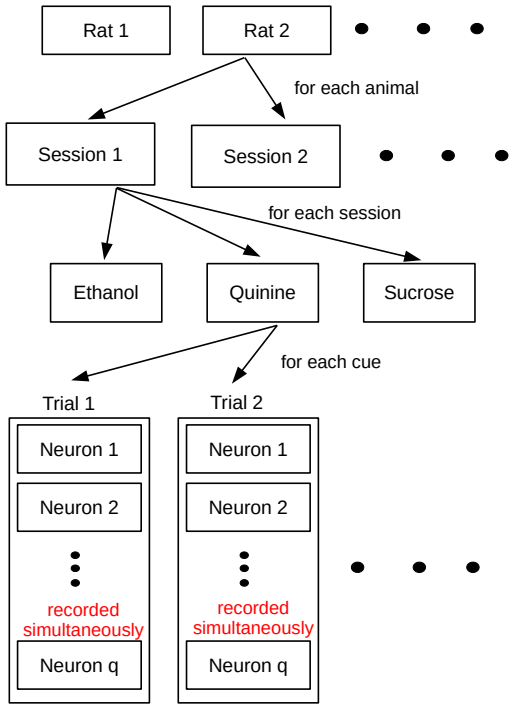
(a) Experimental setup on the rats.



(b) Counts of neurons in each session.



(c) Structure of the dataset.



(d) Illustration of statistical analysis using baseline recordings.

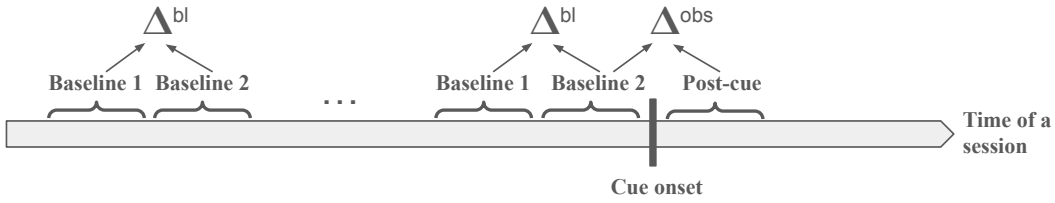


Figure 3.14: Explanation of the experiment and the data.

examine the estimated firing rates $\widehat{\alpha_i/k_i}$ and the neuronal associations represented by the squared loading matrix $\widehat{\mathbf{\Sigma}} = \widehat{\mathbf{\Lambda}}\widehat{\mathbf{\Lambda}}^T$ with nonsignificant entries of $\widehat{\mathbf{\Sigma}}$ set to zero. We adopted $\widehat{\mathbf{\Sigma}}$ instead of $\widehat{\mathbf{\Lambda}}$ for comparing baseline and post-tone activities to avoid random permutation of the columns of $\widehat{\mathbf{\Lambda}}$. Note that here we do not add $\widehat{\mathbf{\Psi}}$ to $\widehat{\mathbf{\Lambda}}\widehat{\mathbf{\Lambda}}^T$ to make the diagonal elements of $\widehat{\mathbf{\Sigma}}$ all equal to 1, as this allows us to account for the change in the uniqueness of each neuron.

As an example, Figure 3.15 plots the estimated metrics for one session of rat 1. Note that the same group of neurons were recorded for all three cue conditions. Compared to quinine, the estimated firing rates of two neurons significantly spear up after both alcohol and sucrose cues, indicating that OFC neuron firing rates might encode information about the cue. Indeed, similar observations in OFC have been made in the literature about value-encoding cues modulating OFC neurons (e.g., Critchley and Rolls, 1996; Padoa-Schioppa and Assad, 2006). Plotted $\widehat{\mathbf{\Sigma}}$ matrices in Figure 3.15d, 3.15e and 3.15f show coordinated spiking activities between OFC neurons in both baselines but less neuronal association after the alcohol cue. The change in the correlation structure before and after the cue confirms that the OFC is coding the information in the cue, which, in the case of sucrose, is consistent with the literature (e.g., Critchley and Rolls, 1996; Padoa-Schioppa and Assad, 2006). Since the coding capabilities of OFC for ethanol cues are not studied, to the best of our knowledge, our finding is the first to confirm the ethanol coding function for OFC.

To investigate whether this finding is consistent across sessions and animals, a summary statistic is needed to summarize the change in the correlation matrix from before cue to after cue. After carefully considering several candidate statistics (see Appendix B.5), we adopted the test statistic in Srivastava and Yanagihara (2010), which was proposed for testing the equality of covariance matrices denoted by Δ. We chose this statistic for its convenience in implementation, the ability to indicate in which direction the overall correlation changes, and its tendency to centre around zero in the absence of notable change. It is defined as

$$\Delta = \mathrm{tr}(\mathbf{\Sigma}_1\mathbf{\Sigma}_1^{\mathrm{T}})/\mathrm{tr}(\mathbf{\Sigma}_1)^2 - \mathrm{tr}(\mathbf{\Sigma}_2\mathbf{\Sigma}_2^{\mathrm{T}})/\mathrm{tr}(\mathbf{\Sigma}_2)^2, \tag{3.25}$$

where $\mathbf{\Sigma}_1$ in our case represents the pre-cue neuronal correlations and $\mathbf{\Sigma}_2$ the post-cue ones. In order to construct what is similar to a reference distribution of Δ, we first calculate the baseline $\Delta^{\mathrm{bl}}_{c,a,s}$ using $\mathbf{\Sigma}_1$ from baseline 1 and $\mathbf{\Sigma}_2$ from baseline 2 for each cue c, animal a and session s. Note that baseline 2 refers to the 2-second window following the corresponding baseline 1 and we have 6 pairs of baselines for each combination of animal and session. We do not pool statistics across animals because of the relatively large animal-to-animal variability. Instead, for each combination of cue, animal, and session, we obtain

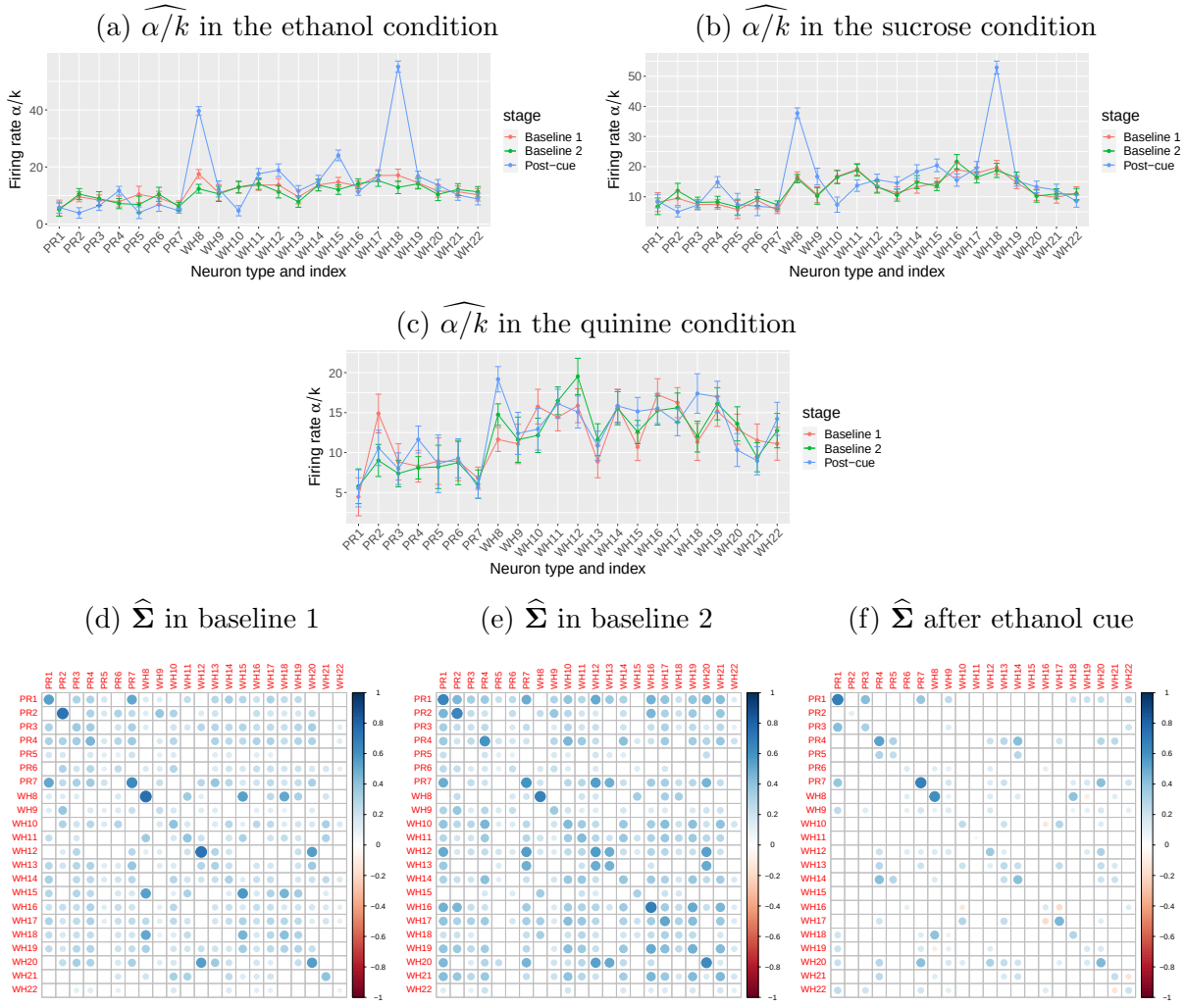


Figure 3.15: Examples of estimated quantities from one session consisting of 22 neurons for rat 1. Neurons are classified into projection (PR) neurons and wide high (WH) neurons based on the waveform of their raw signals prior to spike sorting. (a)-(c): Estimated firing rates in the baseline and following ethanol, sucrose and quinine cues, respectively. The error bars represent the posterior standard deviations. (d)-(f): Estimated correlation matrices in two baseline windows and post ethanol cue. Correlation coefficients that are not significantly different from zero are set to zero in the plots.

a reference sequence consisting of the baseline statistics $\Delta^{\text{bl}}_{c,a,s,1}, \Delta^{\text{bl}}_{c,a,s,2}, \ldots, \Delta^{\text{bl}}_{c,a,s,6}$. Then we determine what is similar to the "observed test statistic" $\Delta^{\text{obs}}_{c,a,s}$ with $\boldsymbol{\Sigma}_1$ from baseline 2 and $\boldsymbol{\Sigma}_2$ from the post cue condition. To quantify the change in overall FC between neurons attributed to the cue, we check where $\Delta^{\text{obs}}_{c,a,s}$ lies relative to the reference sequence $\{\Delta^{\text{bl}}_{c,a,s,i},\ i = 1, 2, \ldots, 6\}$.

3.6.3 Leave-Neuron-Out GOF Assessment

The LNO-PP algorithm proposed in Section 3.4 can be used to evaluate the agreement between the proposed model and the data. Each combination of animal, session, experimental condition (ethanol, sucrose, quinine), and tone status (baseline, baseline 2, post-tone) requires a separate model fit, making the LNO-PP algorithm computationally expensive to run for all combinations. By randomly sampling fit models for GOF assessment using cumulative spike counts (ses Section 3.4.2), we found that the GOF is associated with the presence of sufficient signal in the data to predict the left-out neuron, and that our model provides satisfactory fit to the data when signals of interactions exist.

As an example, we present the estimated $\widehat{\boldsymbol{\Sigma}}$ and GOF results of 1000 LNO-PP samples for one session of animal 1 in Figure 3.16. The estimated neuronal association $\widehat{\boldsymbol{\Sigma}}$ is plotted for a baseline period and the 2s post-cue period in Figure 3.16a and 3.16b, respectively. We performed three LNO-PP checks with the left-out neuron chosen to be neurons PR1 and PR5 in the baseline period and PR5 in the post-cue period. PR1 is an example of a neuron which has large idiosyncrasy, as indicated by the small values of the elements in $\widehat{\boldsymbol{\Sigma}}$ corresponding to PR1 in both baseline and post-cue windows. The poor GOF diagnostic result in Figure 3.16c confirms that the activities of PR1 cannot be predicted well based on information from other neurons. For neuron PR5, its correlations with other neurons are higher in the baseline period but decrease after the cue. This is reflected in Figure 3.16d and 3.16e, showing less satisfactory prediction for PR5 after the cue.

These observations suggest that our model can capture the neuronal associations well when such associations exist. Conversely, for neurons with negligible interactions with others, our model can only make an informed guess at predicting their activities. In this case, instead of concluding that the model does not fit the data well, it is more likely that the neuron being predicted does not belong to the group of sampled neurons in the data.

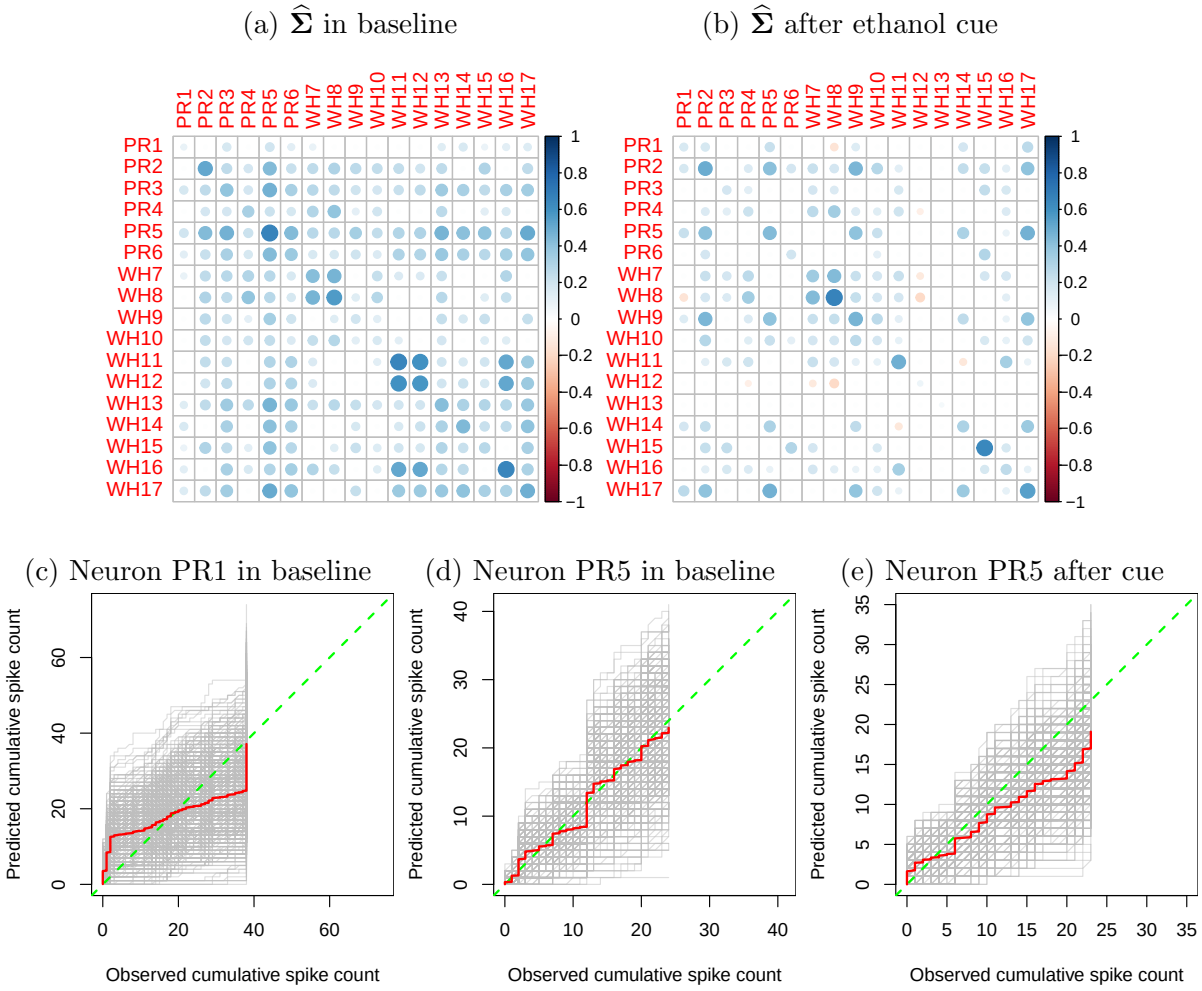


Figure 3.16: LNO-PP graphical diagnostic results based on cumulative spike counts for two different held-out neurons (PR1 and PR5) of animal 1 in two time windows (baseline and post-cue). Each grey line represents one sample from the LNO-PP check algorithm and the red line corresponds to the average of all grey lines.

3.6.4 Results

We first consider the change of FC between neurons after the ethanol tone is played. The reference sequence $\{\Delta^{\text{bl}}_{c,a,s,i},\ i = 1, 2, \ldots, 6\}$ for baseline changes are plotted for each session and animal in Figure 3.17a, where each vertical streak of red dots represent the baseline changes for one session. The observed statistic $\Delta^{\text{obs}}_{\text{alcohol,a,s}}$ (blue triangles) is greater than the values in the reference sequence for most sessions. Note that a large Δ indicates a decrease in overall FC. This observation is consistent across animals except for rat 5 and the reduction of FC is most prominent for rat 6. It is noteworthy that the variability of Δ differs for different rats, though the level of variability is similar for both baseline and different cue conditions within the same animal. This suggests avoiding pooling across animals in our analyses due to heteroskedasticity. The high variability in baseline can be caused by many factors, including the animals' states of rest or movement during baseline recording.

Figure 3.17b summarizes the baseline and cue-conditioned Δ by pooling across sessions for each animal and cue combination. The baseline Δ values (indicated by red dots) at the bottom of each panel are all centred around zero (indicated by the grey dashed line), suggesting that there is on average little change in FC under the baseline condition. It is easy to see that more cue-conditioned Δ values (indicated by purple, blue and green dots) lie on the positive half of the x-axis. We applied the one-sided Wilcoxon rank test to compare baseline and alcohol-cued Δ for all animals. Rats 1, 3, 4 and 6 are significant at 5% level. The same tests comparing baseline and sucrose-cued Δ found rats 1, 4 and 6 significant at 5% level. Comparison between baseline and quinine-cued Δ found rats 1, 4 and 6 significant at 5% level and rat 5 significant at 10% level. The session-level plots for sucrose and quinine cues are presented in Figure 3.18a and 3.18b. Further examination of these plots show a large reduction of FC post quinine cue in some sessions from rats 4 and 6.

Although we did not correct for multiple testing at this point, our exploratory findings show evidence of asynchronous activities following the cues of all three types. Importantly, we emphasize that this observation is consistent across at least half of the animals considered in the analysis. Furthermore, we found that the majority of OFC neurons in the samples did not have a significant increase in firing rates after the cue, while very few of them fired up drastically such as the examples in Figure 3.15a and 3.15b. While it is possible that increased firing rates manifest a higher correlation among neurons, we observed the opposite, which is a decrease in overall correlation. An explanation for this phenomenon is that the cues might disrupt the coordinated activities between neurons at the baseline level, causing them to desynchronize. We believe our results on rats' OFC

neurons bring scientifically interesting insight about their functional connectivity into the neuroscience community.

3.7 Discussion

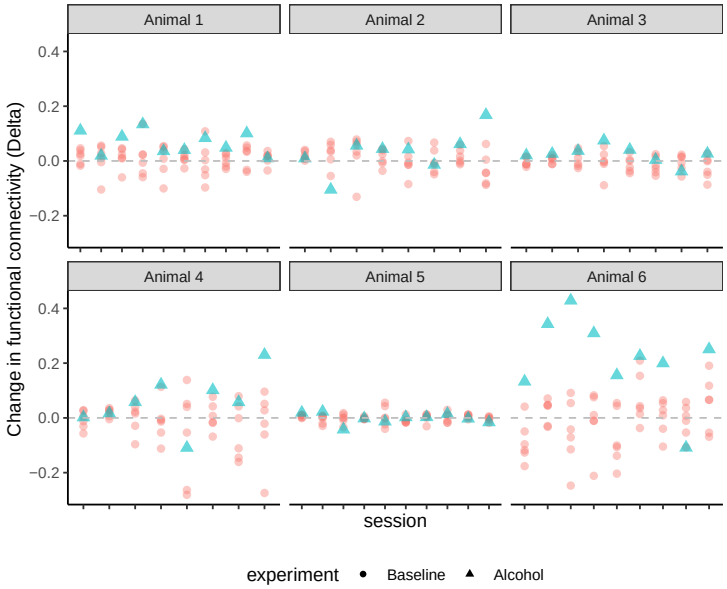
We have introduced a novel continuous-time latent factor model for analyzing spike trains simultaneously recorded from multiple neurons. The focus of downstream analyses using our model results is inferring functional connectivity between neurons. The main scientific contribution of this work is the finding of a consistent association between the collective neural activity in rat OFC and cues. The impact of cues manifests itself in both the functional connectivity between neurons and their firing intensities. The change of functional connectivity and firing rates vary for the OFC neurons, which are known to be associated with both alcohol use and reward/punishment coding in the brain. In particular, the OFC neurons show higher firing intensities and less coordinated activities after the ethanol cue. Our findings have brought scientifically interesting and meaningful insight into the roles of OFC in value encoding and alcohol use.

The proposed model is presented in a continuous-time setting to avoid arbitrary time binning. Importantly, the model is both biologically plausible and scalable. In the latent layer, linked to the observed spike trains are the dynamic resetting paths which are proxies for the membrane potentials of the neurons. Biological plausibility is essential for a computational model as the goal of the model is to help scientists understand the underlying physiological phenomena, and, potentially, contribute to the study of the nervous system.

The scalability of the model is achieved in two key areas. First, instead of modelling the full covariance matrix of all latent paths, we assume that a small set of latent factors governs their joint trajectories. The factor analysis approach reduces both the number of parameters and the computational complexity involved in the covariance matrix inversion. The factor loading matrix provides not only an approximation to the covariance matrix, but also a different angle from which the functional connectivity among neurons can be interpreted, as illustrated in Section 3.5.2. This highlights the superior interpretability of latent state models for clustering and predicting neuroscientific phenomena (Tresch et al., 2006; Lunardini et al., 2015; Sani et al., 2021).

Second, we proposed an efficient inference procedure, which includes a method for selecting the number of factors, a variant of the Laplace approximation for posterior distribution estimation, and an algorithm for sampling from an approximate posterior predictive distribution to assess goodness-of-fit. Compared to the MCMC methods traditionally used

(a) Alcohol cue



(b) Summary of all cues

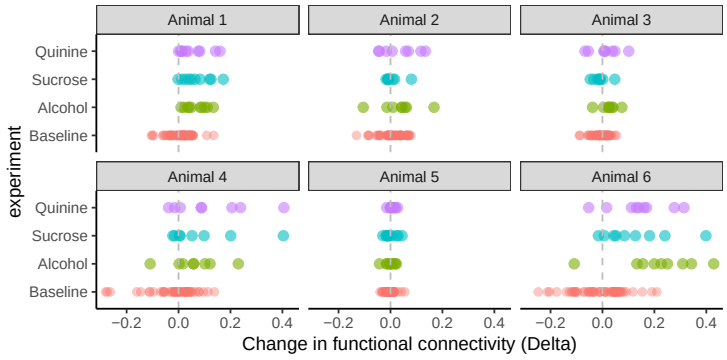
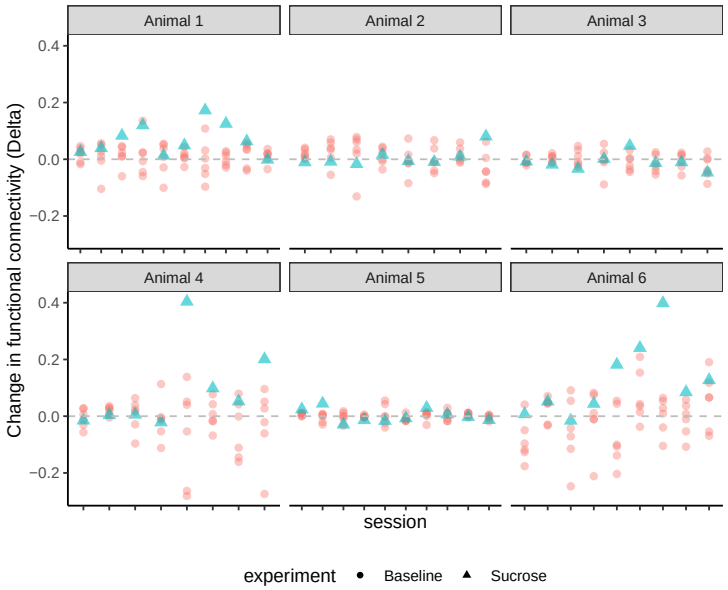


Figure 3.17: Plots of the summary statistic Δ for comparing FC between neurons in different cue conditions. (a) compares the change of FC in baselines and that after the alcohol cue. Each red dot represent a value of $\Delta^{\text{bl}}_{\text{alcohol,a,s,i}}$ for the corresponding animal and session in the alcohol cue condition. Each vertical streak of red dots come from one session. The blue triangles represent $\Delta^{\text{obs}}_{\text{alcohol,a,s}}$ for the corresponding animal and session. The grey dashed line corresponds to zero. (b) pools the baseline values $\Delta^{\text{bl}}_{c,a,s,i}$ from all sessions of an animal, represented by the red dots. The observed statistics $\Delta^{\text{obs}}_{c,a,s}$ for each cue condition are also pooled from all sessions, with green, blue and purple dots representing alcohol, sucrose and quinine cues, respectively.

(a) Sucrose cue



(b) Quinine cue

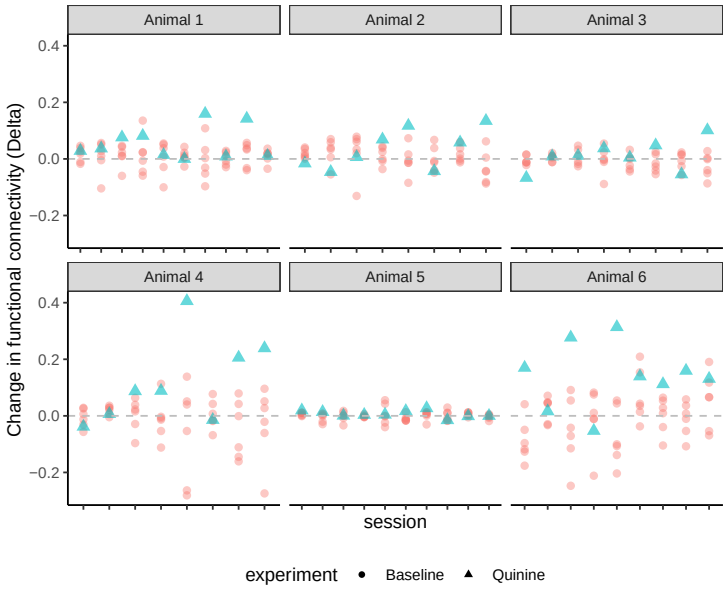


Figure 3.18: Plots of the summary statistic Δ for comparing FC between neurons. (a) and (b) compare the change of FC in baselines and that after the sucrose and quinine cues, repectively. Each red dot represent a value of $\Delta^{\text{bl}}_{c,a,s,i}$. Each vertical streak of red dots comes from one session. The blue triangles represent $\Delta^{\text{obs}}_{c,a,s}$ for the corresponding animal and session.

for Bayesian inference, the Laplace method has been shown to scale to a much larger number of latent variables (see, e.g. Chen et al., 2023). This is critical for developing computational models as state-of-the-art recording techniques are quickly advancing the number of simultaneously recorded neurons (Jun et al., 2017; Steinmetz et al., 2020). In the proposed model, the number of latent variables grows with the number of neurons, so the standard asymptotics of Laplace approximations for fixed q and $n \to \infty$ does not apply here (Shun and McCullagh, 1995; Ogden, 2021). Williams and Barber (1998) justified the use of Laplace approximation for Gaussian Process Classification (GPC) by considering infill asymptotics (Cressie, 1993) in which the number of observations increases in a bounded region. When compared to other methods which, theoretically, provide more accurate approximations to the marginal likelihood, Laplace approximation is still often preferred due to its computational efficiency and reasonable inference results observed in empirical studies (Nickisch and Rasmussen, 2008; Chen et al., 2023). Furthermore, our GOF diagnostic algorithm leverages this approximation technique to construct a leave-neuron-out posterior predictive distribution, whose samples can be compared to the observed data to check the agreement between the model and the data.

Our simulation studies demonstrated the accuracy of the proposed inference procedure using data generated from the model itself. Moreover, we also applied the proposed model on a simulated neuronal network consisting of artificial neurons. Our model outputs a loading matrix which represents the inherent functional connectivity of the network in a quantitative manner. The loading matrix offers insight about how heavily each neuronal cluster depends on difference sources of stimuli. Such information is valuable for inferring the network properties but is unavailable in most hard classification or clustering algorithms. To this end, we showed the advantages of our proposed model in a clustering task in comparison with two other popular neuron clustering methods.

3.7.1 Future Work

Future directions of this work are two-fold: improvement on our existing modelling framework and extensions of the model. The current modelling approach can benefit from a more systematic strategy to choose the number of factors, the inclusion of a trial-level random effect, and a way to account for refractory periods in neuronal spiking. While we have discussed an exploratory factor analysis procedure in Section 3.5.1 and Appendix B.3, it requires the analyst to make subjective decisions based on the results. Some existing methods for choosing the number of factors include MCMC algorithms, which consider it as a model parameter (Lopez and West, 2004; Ghosh and Dunson, 2009), or assume an infinite number of factors with a shrinkage prior imposed on each factor loading column

(Schiavon et al., 2022). However, such methods are incompatible with our current inference framework and are computationally prohibitive.

Regarding the trial-level effect, our current model assumes that all trials of an animal can be treated as independent and identical repetitions of the same experiment. However, trial-to-trial variability has been observed in many studies (Arieli et al., 1996; Ventura et al., 2005; Churchland et al., 2011). Including a trial-level random effect could account for this variability. Similarly, an animal-level random effect might be considered if a single model is to be fit to all animals, though the large inter-animal variability might obscure the signals of interest.

Refractory period refers to a short period immediately after a spike where the neuron cannot generate a second spike. As discussed in Section 3.3.1, it provides biological justification for discretizing the time using a sufficiently short bin length which is compatible with the duration of the refractory period. The model can accommodate potentially longer refractory periods by introducing a random effect $u_{it} \geq 0$ such that the latent state x_{it} after each spike is reset to $x_{i0} - u_{it}$ instead of x_{i0}. Since now the latent path needs to traverse an additional distance of u_{it} before reaching the threshold, this effectively introduces a refractory period that delays the second spike.

Potential extensions of the proposed model are focused on incorporating dynamic functional connectivity and more complex non-linear structure in the latent layer. A critical assumption of the proposed model is the time-invariance of model parameters, which typically does not hold when analyzing long or highly irregular/non-stationary spike trains. We address this by focusing on shorter experimental durations, where we can reasonably assume stationarity for the experimental data. The choice of dynamic models over non-dynamic ones inevitably trades computational complexity for temporally varying estimates. A computationally tractable model is necessary given the scale of our data with a large number of neurons and trials. The proposed model can be applied to a longer observation window if the spike trains are broken down into shorter segments, during which the data-generating process can be assumed both homogeneous and stationary. This would require identifying change points before model fitting. To incorporate change point detection of firing intensities and correlation structure in the model, the latent paths need a different modelling assumption. A history-dependent trend term can be added to the stochastic process describing the latent path. The covariance matrix $\mathbf{\Sigma} = \mathbf{\Lambda}\mathbf{\Lambda}^T + \mathbf{\Psi}$ can be made dynamic by allowing a time-dependent loading matrix, i.e., $\mathbf{\Lambda} = \mathbf{\Lambda}_t$, which was originally proposed in Aguilar and West (2000). The challenge is that, without additional assumptions on its structure, $\mathbf{\Lambda}_t$ will become increasingly complex as the number of neurons grows (Zhou et al., 2016; Opschoor et al., 2020). Therefore, a natural next step is to investigate a computationally feasible and biologically reasonable dependence structure for the dynamic

factor model.

Another direction of extending the latent variable model is to specify the latent layer of the model using an artificial neural network (ANN) architecture. The ANNs can be used as a learning model to infer the complex neuronal dynamics, rather than representing the biological components of the network (Saxena and Cunningham, 2019; Sinz et al., 2019; Yang and Wang, 2020). The proposed model makes the simplifying assumption that the current state of the latent path depends linearly on its previous state and the states of other neuronal paths. Multi-layer neural networks are a popular way for modelling nonlinearities in the data generation mechanism (see, e.g., Pandarinath et al., 2018). It can potentially capture more complex correlation structures between neurons and the dependence on history.

Computational neuroscience is a fascinating field where researchers from different disciplines collaboratively study the brain using experiments and quantitative tools. As the experimental technology develops to accommodate large-scale neuron recordings, computation needs to scale up, too. Our proposed model and inference method can be utilized to explore the collective and interactive activities of double-digit neurons in the time scale of seconds. Further development of the model will emphasize on flexible, interpretable and scalable modelling strategies.

Chapter 4

Superfast Inference for Gaussian Process Factor Analysis

4.1 Introduction

Gaussian Processes (GPs) are widely used in applications such as meteorology, finance, biological systems (e.g., Cressie, 1993; Banerjee et al., 2003; Wilson and Ghahramani, 2011; Yu et al., 2009; Rasmussen and Williams, 2006), providing a flexible framework for modelling correlation in the data. Particularly in neuroscience, GPs are often employed to model neural data characterized by hidden dynamics with temporal correlation (Yu et al., 2009; Wu et al., 2017; Zhao and Park, 2017). The Gaussian Process Factor Analysis (GPFA) proposed by Yu et al. (2009) is a commonly used framework for projecting the observed neural spike counts into a low-dimensional latent space consisting of GPs referred to as factors. Conditional on a set of factors $\boldsymbol{x}_t$ where each element x_t follows an independent GP, the observed square-rooted spike counts $\boldsymbol{y}_t$ of all neurons in time interval of index t is assumed to depend linearly on $\boldsymbol{x}_t$ with Gaussian noises, i.e., $\boldsymbol{y}_t \sim \mathcal{N}(\boldsymbol{\beta}\boldsymbol{x}_t, \boldsymbol{\Sigma})$. Owing to the analytic tractability from the Normal-Normal conjugacy, the GPFA is a fast and flexible tool for extracting long-memory latent dynamics driving neuronal activities.

Despite its superior computational scalability, we have shown in Chapter 3 that the GPFA model produces estimates less robust to the bin size. In contrast, our proposed model in Chapter 3 circumvents the choice of bin size by assuming that the spikes are generated from threshold-crossing of some proxies for neuronal membrane potentials modelled by latent Wiener processes, which themselves depend on a smaller set of factors. While this model has advantages over the GPFA in biological interpretability and avoids choosing a

bin size, its Markov assumption on the latent factor processes, adopted for computational convenience, does not accommodate for potential long-term memories in neural dynamics that can be observed in neuronal processes such as long-term potentiation (LTP) and long-term depression (LTD) (Morris, 2003; Bliss and Cooke, 2011). In brain areas where long memories are known to exist, it might be desirable to incorporate a more general covariance structure on the latent processes for assessing the importance of long-memory effects. When the binning effect is assumed to be negligible, such as when the neural dynamics are changing slowly, the GPFA model with a general GP covariance kernel is preferable for its flexibility in capturing long-memory latent dynamics. However, this leads to a complexity of $\mathcal{O}(N^3)$ for evaluating a general GP, as opposed to $\mathcal{O}(N)$ for the Markovian case. Specifically, N equally spaced realizations of a GP are distributed as an N-dimensional multivariate Normal random variable. As a result, routine log-likelihood evaluations for GPs scale as $\mathcal{O}(N^3)$, primarily due to the inversion of the covariance matrix and the calculation of its log-determinant.

In this chapter, we present an efficient MCMC inference method tailored for the more general GPFA framework, and extend it to a Poisson-GPFA model accommodating Poisson observations. The proposed method relies on a "superfast" algorithm scaling as $\mathcal{O}(N \log^2 N)$ for the GP log-likelihood evaluation (Ling, 2019). For stationary GPs wherein the covariance kernels depend only on the distance between two time steps, their covariance matrices possess a special structure known as *Toeplitz matrices* (Golub and Van Loan, 1996). A Toeplitz matrix has identical elements on each diagonal from upper left to lower right, taking the form

$$\boldsymbol{V}_{\boldsymbol{\theta}} = \begin{bmatrix} v_{\boldsymbol{\theta}}(0) & v_{\boldsymbol{\theta}}(-1) & v_{\boldsymbol{\theta}}(-2) & \cdots & v_{\boldsymbol{\theta}}(-(N-1)) \\ v_{\boldsymbol{\theta}}(1) & v_{\boldsymbol{\theta}}(0) & v_{\boldsymbol{\theta}}(-1) & \cdots & v_{\boldsymbol{\theta}}(-(N-2)) \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ v_{\boldsymbol{\theta}}(N-1) & v_{\boldsymbol{\theta}}(N-2) & v_{\boldsymbol{\theta}}(N-3) & \cdots & v_{\boldsymbol{\theta}}(0) \end{bmatrix}, \tag{4.1}$$

where $\boldsymbol{\theta}$ parameterizes the GP covariance kernel $k(x, x' \mid \boldsymbol{\theta})$. This property can be exploited to devise more efficient algorithms for solving systems such as $\boldsymbol{V}_{\boldsymbol{\theta}} \cdot \boldsymbol{y} = \boldsymbol{b}$ for $\boldsymbol{y}$, with $\boldsymbol{b} = \boldsymbol{I}$ representing a special case of inverting $\boldsymbol{V}_{\boldsymbol{\theta}}$. In contrast to routine solvers with a complexity of $\mathcal{O}(N^3)$, "fast" Toeplitz system solvers are available which scale as $\mathcal{O}(N^2)$, which is the case for the popular Levinson-Durbin and Trench-Zohar algorithms (Levinson, 1946; Durbin, 1960; Trench, 1964; Zohar, 1969). The computational cost of inference for stationary GPs can be significantly reduced by adopting more efficient Toeplitz system solvers scaling as $\mathcal{O}(N \log^2 N)$, termed as "superfast". Ammar and Gragg (1987, 1988) demonstrated that the generalized Schur algorithm, combined with careful use of fast Fourier transform, enables solving Toeplitz systems of size $N = 2^k + 1$ and computing

the log-determinant in superfast time. Recently, Ling (2019) extended the Ammar-Gragg algorithm to an arbitrary size N, along with providing an efficient implementation of their algorithms in the R/C++ package `superGauss` (Ling and Lysy, 2020). This software allows users to implement superfast inference for a wide range of stationary GP models.

Using the algorithm in Ling (2019) and its implementation in `superGauss`, we design two superfast MCMC algorithms for the GPFA model and the Poisson-GPFA model, respectively. While inference for the former can be achieved in closed form due to its conjugate prior, the original inference method in Yu et al. (2009) using the Expectation-Maximization algorithm lacks uncertainty estimates for the parameters. We present a superfast Gibbs sampling algorithm facilitating uncertainty quantification. On the other hand, due to the absence of conjugacy, existing inference methods for the Poisson-GPFA model typically employ approximations for inference such as variational methods (Wu et al., 2017; Zhao and Park, 2017). Here, we introduce a superfast Metropolis-within-Gibbs algorithm enabling exact inference, subject to Monte Carlo errors.

In the remainder of this chapter, we briefly outline the superfast Toeplitz system solver by Ling (2019) in Section 4.2.1. In Sections 4.2.2-4.2.4, we present superfast MCMC inference algorithms for two classes of multi-neuron spike-train models whose latent dynamic processes are described by stationary GPs. The speed and accuracy of the proposed inference methods are demonstrated via simulation studies in Section 4.3. Concluding remarks and discussion are offered in Section 4.4.

4.2 Methods

4.2.1 Superfast Toeplitz System Solver

For an $N \times N$ positive definite Toeplitz covariance matrix $\boldsymbol{V}$, it is clear from observing the structure in (4.1) that $\boldsymbol{V}$ can be represented by its first row or column. Gohberg and Semencul (Gohberg and Semencul, 1972) showed that this is also true for its inverse $\boldsymbol{V}^{-1}$. That is, both $\boldsymbol{V}$ and $\boldsymbol{V}^{-1}$ can be stored as a vector in $\mathcal{O}(N)$ rather than a dense matrix in $\mathcal{O}(N^2)$. Let $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_N)$ denote the first column of $\boldsymbol{V}^{-1}$, the Gohber-Semencul formula claims that $\boldsymbol{V}^{-1}$ can be written as

$$\boldsymbol{V}^{-1} = \frac{1}{\delta_1}(\boldsymbol{L}_1\boldsymbol{L}_1^T - \boldsymbol{L}_2\boldsymbol{L}_2^T), \tag{4.2}$$

where $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ are lower-triangular Toeplitz matrices given by

$$\boldsymbol{L}_1 = \begin{bmatrix} \delta_1 & & & \\ \delta_2 & \delta_1 & & \\ \vdots & \ddots & \ddots & \\ \delta_N & \cdots & \delta_2 & \delta_1 \end{bmatrix}, \quad \boldsymbol{L}_2 = \begin{bmatrix} 0 & & & \\ \delta_N & 0 & & \\ \vdots & \ddots & \ddots & \\ \delta_2 & \cdots & \delta_N & 0 \end{bmatrix}. \tag{4.3}$$

Thus, to solve a Toeplitz system $\boldsymbol{V}\boldsymbol{y} = \boldsymbol{b}$, it suffices to find $\boldsymbol{\delta}$ giving $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$. This is followed by successive multiplications of a Toeplitz matrix and a vector. Importantly, by embedding the Toeplitz matrix into a $2N \times 2N$ circulant matrix, this multiplication is attainable in $\mathcal{O}(N \log N)$ steps using the fast Fourier transform rather than the normal $\mathcal{O}(N^2)$ steps. Details of the $\mathcal{O}(N \log N)$ algorithm for Toeplitz matrix-vector multiplication are given in Sections 5.3.3 of Kailath and Sayed (1987).

It was Ammar and Gragg (1987) who pointed out that $\boldsymbol{\delta}$ can be produced by applying $N - 1$ steps of Schur's algorithm (Schur, 1917) on polynomials constructed using the first column of $\boldsymbol{V}$. The determinant $|\boldsymbol{V}|$ is obtained as a by-product in this process. We refer to Ammar and Gragg (1987) and Ling (2019) for the technical details on computing $\boldsymbol{\delta}$. While the original Schur algorithm takes $\mathcal{O}(N^2)$ to sequentially execute $N - 1$ steps, Ammar and Gragg (1987) proposed a generalized Schur algorithm (GSA) which avoids most intermediary steps and therefore only costs $\mathcal{O}(N \log^2 N)$ for $N = 2^k + 1$. Given that any positive integer can be written as a sum of powers of 2, Ling (2019) proposed the extended GSA for arbitrary N using a procedure to merge pieces obtained from the original GSA. Furthermore, Ling (2019) presented how the extended GSA, together with other carefully designed technical accelerations, lead to the superfast computation of gradient and hessian for stationary GP likelihood evaluations. Both theoretical analyses in Ammar and Gragg (1989) and numerical experiments in Ling (2019) have demonstrated that the superfast solver surpasses Levinson's solver around $N = 260$. Consequently, it is expected that the superfast solver will substantially speed up inference for stationary GPs comprising thousands of time points, as we will illustrate with the models described below.

4.2.2 Gaussian Process Factor Analysis

Summarizing a high-dimensional data set with low-dimensional embeddings is a common approach for exploring data structure. The Gaussian process factor analysis (GPFA) framework, introduced by Yu et al. (2009), is frequently used to extract informative low-dimensional representations of high-dimensional neural spike-train data in the brain. It accomplishes both dimension reduction and latent space smoothing. In this section, we

outline the GPFA model and propose a superfast Gibbs sampler for its inference using the extended GSA proposed by Ling (2019).

Consider a neuronal ensemble consisting of D neurons whose individual spike counts were simultaneously recorded over time intervals indexed by $n = 1, \ldots, N$. As discussed in Chapter 3, the choice of the interval length has a substantial impact on inference results. Nonetheless, some guidelines on the bin length choice can be found in Shimazaki and Shinomoto (2006) and Omi and Shinomoto (2011). For a given matrix $\boldsymbol{A} = [a_{ij}]_{1\leq i\leq m, 1\leq j\leq n} \in \mathbb{R}^{m\times n}$, let $\boldsymbol{a}_{i,:} = (a_{i,1}, ..., a_{i,n})$ and $\boldsymbol{a}_{:,j} = (a_{1,j}, ..., a_{m,j})$. We define y_{nd} as the square-rooted spike count in time interval n for neuron d, and let $\boldsymbol{Y} = [y_{nd}]_{1\leq n\leq N, 1\leq d\leq D}$ denote all observations across time intervals and neurons. The GPFA framework assumes that the observed spiking activities are driven by some latent variables, with x_{nk} denoting the latent state of the k-th latent process in time interval n. Namely, it estimates a lower-dimensional summary of the D-dimensional spiking activities $\boldsymbol{y}_{n,:}$ in time interval n by extracting K-dimensional latent states $\boldsymbol{x}_{n,:}$, where K ($K < D$) denotes the number of latent factors. Assuming a linear relationship between the latent and observed states and Gaussian noises on the observations, the GPFA likelihood is given by

$$\boldsymbol{y}_{n,:} \mid \boldsymbol{x}_{n,:} \sim \mathcal{N}(\boldsymbol{x}_{n,:}\boldsymbol{\beta} + \boldsymbol{\alpha}, \boldsymbol{\Sigma}),$$

where $\boldsymbol{\beta} \in \mathbb{R}^{K\times D}$ represents the coefficients for the latent factors (referred to as the loading matrix), $\boldsymbol{\alpha} \in \mathbb{R}^{1\times D}$ denotes the drift parameters, and $\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2, \ldots, \sigma_D^2)$ is a $D \times D$ diagonal matrix capturing observation noise or the idiosyncrasy of each neuron. Each of the K latent factor processes is assumed to be a zero-mean Gaussian process, such that

$$\boldsymbol{x}_{:,k} \overset{\text{ind}}{\sim} \mathcal{N}\big(\boldsymbol{0}, \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\big),$$

where $\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} \in \mathbb{R}^{N\times N}$ denotes the covariance matrix parameterized by a kernel function $f_{\boldsymbol{\theta}_k}(\cdot)$ with parameters $\boldsymbol{\theta}_k$ such that $[\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}]_{i,j} = f_{\boldsymbol{\theta}_k}(|i-j|)$.

Thus, we can summarize the GPFA model as follows:

$$\begin{aligned} \boldsymbol{y}_{n,:} \mid \boldsymbol{x}_{n,:} &\sim \mathcal{N}(\boldsymbol{x}_{n,:}\boldsymbol{\beta} + \boldsymbol{\alpha}, \boldsymbol{\Sigma}), \quad n = 1, \ldots, N, \\ \boldsymbol{x}_{:,k} &\overset{\text{ind}}{\sim} \mathcal{N}\big(\boldsymbol{0}, \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\big), \quad k = 1, \ldots, K, \\ \boldsymbol{\alpha} &\sim \pi(\boldsymbol{\alpha}), \quad \boldsymbol{\Sigma} \sim \pi(\boldsymbol{\Sigma}), \quad \boldsymbol{\beta} \sim \pi(\boldsymbol{\beta}), \quad \boldsymbol{\theta} \sim \pi(\boldsymbol{\theta}), \end{aligned} \tag{4.4}$$

where priors are imposed on the parameters. The parameters of interest are the factor loading matrix $\boldsymbol{\beta}$, the drift parameters $\boldsymbol{\alpha}$, the observation noise covariance matrix $\boldsymbol{\Sigma}$, and the kernel parameters encoding the temporal correlation for each latent factor process $\boldsymbol{\theta}_k$, $k = 1, \ldots, K$.

Inferences for the GPFA model in Yu et al. (2009) were carried out using the Expectation-Maximization (EM) algorithm, where the E-step requires evaluating the conditional distribution of $\boldsymbol{x}_{:,k}$ for $k = 1, \ldots, K$. Let $\otimes$ denote the Kronecker product and vec() an operator to transform a matrix into a column vector by stacking the columns of the matrix vertically. The E-step in Yu et al. (2009) is then given by

$$\begin{aligned} \boldsymbol{x}_{:,k} \mid \boldsymbol{Y} \sim \mathcal{N}\Big(& (\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \boldsymbol{\Omega}^{-1} \operatorname{vec}(\boldsymbol{Y} - \boldsymbol{A}), \\ & \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)^{-1}} - (\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \boldsymbol{\Omega}^{-1} (\boldsymbol{\beta}_{k,:}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \Big), \end{aligned} \tag{4.5}$$

where $\boldsymbol{\Omega} = \sum_{k=1}^{K} \boldsymbol{\beta}_{k,:}^T \boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} + \boldsymbol{\Sigma} \otimes \mathrm{I}_N \in \mathbb{R}^{dN \times dN}$, and $\boldsymbol{A} = [\boldsymbol{\alpha}^T, \ldots, \boldsymbol{\alpha}^T]^T$ contains N copies of $\boldsymbol{\alpha}$. Since the inversion of $\boldsymbol{\Omega}$ is $\mathcal{O}(d^3 N^3)$, it is computationally expensive to repeatedly evaluate the conditional expectation $E_{\boldsymbol{X}|\boldsymbol{Y}}\left[\ell(\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Sigma}, \boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y})\right]$ in the E-step. Moreover, the EM algorithm provides only point estimates for the parameters of interest, lacking uncertainty quantification.

4.2.3 Superfast GPFA Inference Algorithm

To improve the efficiency of parameter inference and to provide uncertainty quantification on parameter estimates, we introduce a superfast Gibbs sampler where each iteration of sampling incurs a cost of $\mathcal{O}(N \log^2 N)$. That is, the inference algorithm is superfast. The sampler iteratively updates the components in (4.4) via the following sampling scheme:

$$\boldsymbol{x}_{:,k} \mid \boldsymbol{Y}, \boldsymbol{x}_{:,-k}, \boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\Sigma} \sim \pi(\boldsymbol{x} \mid \boldsymbol{Y}, \boldsymbol{x}_{:,-k}, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{\Sigma}), \tag{4.6}$$

$$\boldsymbol{\Sigma} \mid \boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\theta} \sim \pi(\boldsymbol{\Sigma} \mid \boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{\theta}), \tag{4.7}$$

$$\boldsymbol{\beta} \mid \boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\alpha}, \boldsymbol{\theta}, \boldsymbol{\Sigma} \sim \pi(\boldsymbol{\beta} \mid \boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\theta}, \boldsymbol{\Sigma}), \tag{4.8}$$

$$\boldsymbol{\alpha} \mid \boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{\theta}, \boldsymbol{\Sigma} \sim \pi(\boldsymbol{\alpha} \mid \boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{\Sigma}), \tag{4.9}$$

$$\boldsymbol{\theta} \mid \boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\Sigma} \sim \pi(\boldsymbol{\theta} \mid \boldsymbol{X}). \tag{4.10}$$

For conjugate priors

$$\sigma_d^2 \sim \text{Inv-Gamma}(\alpha_d, \beta_d), \quad \alpha_d \sim \mathcal{N}(\nu_d, \tau_d^2), \quad \boldsymbol{\beta}_{:,d} \sim \mathcal{N}(\boldsymbol{\Psi}_d, \boldsymbol{S}_d), \quad d = 1, 2, \ldots, D,$$

the updates (4.7), (4.9) and (4.8) have the following analytic forms:

$$\begin{aligned} \sigma_d^2 \mid \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\beta} &\sim \text{Inv-Gamma}\left(\alpha_d^\star, \beta_d^\star\right) \\ \alpha_d \mid \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\beta}, \boldsymbol{\Sigma} &\sim \mathcal{N}(\nu_d^\star, \tau_d^{2\star}) \\ \boldsymbol{\beta}_{:d} \mid \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{\Sigma} &\sim \mathcal{N}\left(\boldsymbol{\Psi}_d^\star, \boldsymbol{S}_d^\star\right) \end{aligned} \tag{4.11}$$

where

$$
\begin{aligned}
\alpha_d^\star &= \alpha_d + \frac{N}{2} \\
\beta_d^\star &= \beta_d + \frac{(\boldsymbol{y}_d - \boldsymbol{x}\boldsymbol{\beta}_{:d} - \alpha_d)^T(\boldsymbol{y}_d - \boldsymbol{x}\boldsymbol{\beta}_{:d} - \alpha_d)}{2} \\
\nu_d^\star &= \tau_d^{2\star}\left(\frac{\nu_d}{\tau_d^2} + \frac{\sum_{n=1}^N (y_{n,d} - x_{n,:}\beta_{:,d})}{\sigma_d^2}\right) \\
\tau_d^{2\star} &= \left(\frac{1}{\tau_d^2} + \frac{N}{\sigma_d^2}\right)^{-1} \\
\boldsymbol{\Psi}_d^\star &= \boldsymbol{S}_d^\star\left[\boldsymbol{S}_d^{-1}\boldsymbol{\Psi}_d + \frac{\boldsymbol{x}'(\boldsymbol{y}_d - \alpha_d \cdot \boldsymbol{1})}{\sigma_d^2}\right] \\
\boldsymbol{S}_d^\star &= \left[\boldsymbol{S}_d^{-1} + \frac{\boldsymbol{x}'\boldsymbol{x}}{\sigma_d^2}\right]^{-1}
\end{aligned}
\tag{4.12}
$$

For the GP hyperparameters $\boldsymbol{\theta}_k,\ k = 1, \ldots, K$, sampling from (4.10) requires a Metropolis-Hastings (MH) step due to the lack of analytic form for its full conditional distribution. Note that

$$
\pi(\boldsymbol{\theta} \mid \boldsymbol{X}) = \prod_{k=1}^K \pi(\boldsymbol{\theta}_k \mid \boldsymbol{x}_{:,k}) \propto \prod_{k=1}^K L(\boldsymbol{\theta}_k \mid \boldsymbol{x}_{:,k})\pi(\boldsymbol{\theta}_k), \tag{4.13}
$$

where $L(\boldsymbol{\theta}_k \mid \boldsymbol{x}_k)$ is the likelihood in (4.4):

$$
L(\boldsymbol{\theta}_k \mid \boldsymbol{x}_{:,k}) = \frac{\exp\left(-\frac{1}{2}\boldsymbol{x}_{:,k}{\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}}^{-1}\boldsymbol{x}_{:,k}\right)}{\sqrt{|2\pi\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}|}}. \tag{4.14}
$$

Therefore, it is easy to sample $\boldsymbol{\theta}$ via an MH random walk with the log acceptance probability calculated as the difference between the log of (4.13) evaluated at new and old samples of $\boldsymbol{\theta}$, respectively. Using the superfast algorithm in Ling (2019), both the log-determinant and the inverse of $\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}$ in the log of (4.14) can be evaluated in superfast time. Hence, sampling from (4.10) is superfast.

It is worth noting that the dimension of each of the conditional distributions (4.7)-(4.10) for the parameters is much smaller than that of (4.6) for the latent states $\boldsymbol{x}_{:,k}$. Therefore, the computational cost of each sampling iteration is primarily dominated by evaluating (4.6). However, the analytic form of the latent states update (4.6) is non-trivial. In the following, we describe the procedure to sample from (4.6) efficiently.

Define

$$\boldsymbol{Y}_k^\star := \boldsymbol{Y} - \sum_{i \neq k} \boldsymbol{x}_{:,i}\boldsymbol{\beta}_{i,:} - \boldsymbol{A} \equiv \boldsymbol{x}_{:,k}\boldsymbol{\beta}_{k,:} + \boldsymbol{\epsilon},$$

where $\boldsymbol{\epsilon}_{n,:} \overset{iid}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. We find that $\boldsymbol{x}_{:,k} \mid \boldsymbol{Y}_k^\star$ is multivariate Normal:

$$\boldsymbol{x}_{:,k} \mid \boldsymbol{Y}_k^\star \sim \mathcal{N}\left((\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\boldsymbol{\Omega}_k^{-1} \operatorname{vec}(\boldsymbol{Y}_k^\star), \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)^{-1}} - (\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\boldsymbol{\Omega}_k^{-1}(\boldsymbol{\beta}_{k,:}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\right) \tag{4.15}$$

where $\boldsymbol{\Omega}_k = \boldsymbol{\beta}_{k,:}^T\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} + \boldsymbol{\Sigma} \otimes \mathrm{I}_N$ is a $dN \times dN$ matrix. Unlike the computation of $p(\boldsymbol{x}_{:,k} \mid \boldsymbol{Y})$ (4.5) which also involves inverting a $dN \times dN$ matrix $\boldsymbol{\Omega}$, the inversion of $\boldsymbol{\Omega}_k$ can be greatly simplified using the Woodbury matrix identity (Higham, 2002)

$$\boldsymbol{\Omega}_k^{-1} = \boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N - (\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{k,:}^T\boldsymbol{\beta}_{k,:}\boldsymbol{\Sigma}^{-1}) \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1}$$

where $c = \boldsymbol{\beta}_{k,:}\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{k,:}^T$ is a scalar and $\boldsymbol{Q} = c \cdot \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} + \mathrm{I}_N$ is an $N \times N$ Toeplitz matrix. Plugging $\boldsymbol{\Omega}_k^{-1}$ into (4.15) gives

$$\boldsymbol{x}_{:,k} \mid \boldsymbol{Y}_k^\star \sim \mathcal{N}\left(\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1}\boldsymbol{Y}_k^\star\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{k,:}^T, \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1}\right), \tag{4.16}$$

whose simulation can be achieved efficiently by generating

$$\boldsymbol{x}_{:,k} = \boldsymbol{Q}^{-1}\boldsymbol{\varepsilon}_1 + \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1}\boldsymbol{\varepsilon}_2,$$

where

$$\boldsymbol{\varepsilon}_1 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}), \qquad \boldsymbol{\varepsilon}_2 \sim \mathcal{N}(\boldsymbol{Y}_k^\star\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{k,:}^T, c \cdot \mathrm{I}_N).$$

Details of this derivation is given in Appendix C.1. Note that sampling from a Normal distribution with a Toeplitz covariance matrix and Toeplitz matrix-vector multiplication are both superfast using the algorithm in Ling (2019). Since both $\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}$ and $\boldsymbol{Q}$ are $N \times N$ Toeplitz matrices, sampling from (4.16) is superfast.

4.2.4 Extended GPFA for Poisson Observations

Many have argued against the use of Gaussian likelihood for modelling spike counts (Macke et al., 2011; Keeley et al., 2019), given their inherently discrete nature and tendency to exhibit non-Gaussian distributions. Therefore, a straightforward adaption of the original GPFA model in (4.4) to a discrete count model involves substituting the joint Gaussian

observation likelihood in (4.4) with the product of D Poisson likelihoods, each with a log-rate of $\boldsymbol{x}_{n,:}\boldsymbol{\beta}_{:,d}$, $d = 1, \ldots, D$. Assuming the same stationary GP prior in (4.4) on each factor process $\boldsymbol{x}_{:,k}$, the Poisson-GPFA model is given by

$$\begin{aligned} \boldsymbol{y}_{n,d} \mid \boldsymbol{x}_{n,:} &\overset{\text{ind}}{\sim} \text{Poisson}\left(\exp(\boldsymbol{x}_{n,:}\boldsymbol{\beta}_{:,d} + \alpha_d)\right), \quad n = 1, \ldots, N, \quad d = 1, \ldots, D \\ \boldsymbol{x}_{:,k} &\overset{\text{ind}}{\sim} \mathcal{N}(\boldsymbol{0}, \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}), \quad k = 1, \ldots, K \\ \boldsymbol{\alpha} &\sim \pi(\boldsymbol{\alpha}), \quad \boldsymbol{\beta} \sim \pi(\boldsymbol{\beta}), \quad \boldsymbol{\theta} \sim \pi(\boldsymbol{\theta}). \end{aligned} \tag{4.17}$$

In contrast to the Gaussian-GPFA model, the GP priors on the factor processes are not conjugate to the Poisson likelihood. Therefore, closed-form full conditional updates are no longer available for Gibbs sampling on the model. Different variants of the Poisson-GPFA model resort to approximations for inference, such as variational methods (Wu et al., 2017; Zhao and Park, 2017), for computational efficiency. Instead, we introduce a superfast MCMC algorithm by replacing each full conditional in (4.6)-(4.9) by an MH step. Note that the MH update for $\boldsymbol{\theta}$ in (4.10) can be reused here. In comparison to approximate inference methods, the superfast MH-within-Gibbs algorithm is exact up to Monte Carlo errors. We outline the iterative updating scheme in Algorithm 3 in Appendix C.2. Calculations of the acceptance probabilities for $\boldsymbol{x}_{:,k}$ and $\boldsymbol{\theta}_k$ both require evaluating N-dimensional GPs, which can be obtained in superfast time. The variance parameters of the proposal distributions, $\boldsymbol{\Sigma}_{\boldsymbol{x}}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}}, \sigma_{\alpha}^2, \boldsymbol{\Sigma}_{\boldsymbol{\theta}}$, can be tuned to obtain a reasonable acceptance rate, such as the optimal acceptance rate 0.234 advocated in Roberts et al. (1997), or between 0.1 and 0.4, as discussed in Roberts and Rosenthal (2001).

4.3 Simulation Studies

Two simulation studies are conducted to assess the proposed superfast MCMC inference algorithms for the Gaussian-GPFA and Poisson-GPFA models. For both models, the radial basis kernel $k(x, x' \mid \boldsymbol{\theta}) = \theta^w \exp(-|x - x'|^2/\theta^l)$, with lengthscale θ^l and widthscale θ^w, is used for all GPs. Parameter values are given in each simulation study. For inference, each MCMC sampler is run with 4 parallel chains for 10,000 iterations, including 5,000 burn-ins. Initial values are set to their respective true values. All computation is carried out using R version 4.3.1 on a Linux server with 2 Intel Xeon Gold 6434 8-core 3.7GHz CPUs and 256GB memory.

4.3.1 Gaussian-GPFA Model

In this study, $D = 12$ observed processes are simulated conditional on $K = 3$ latent factor processes (GPs) on 2000 time bins of length 0.1s. Parameters are generated by sampling from the following

$$\alpha_d \sim \mathcal{N}(5, 1), \quad \sigma_d \sim \mathcal{N}(1, 0.5), \quad d = 1, \ldots, 12,$$
$$\boldsymbol{\beta} \sim \begin{pmatrix} \boldsymbol{U(3,5)} & U(1,2) & U(1,2) \\ U(1,2) & \boldsymbol{U(3,5)} & U(1,2) \\ U(1,2) & U(1,2) & \boldsymbol{U(3,5)} \end{pmatrix},$$

and setting the GP lengthscale parameters to $\boldsymbol{\theta}^l = (0.005, 0.01, 0.015)$. Since the latent GP can be scaled by scaling $\boldsymbol{\beta}$, the GP widthscale parameters are seen as redundant (Yu et al., 2009). Hence, they are fixed to be constant $\theta_k^w = 0.5$ for $k = 1, 2, 3$. The following priors are specified on the parameters:

$$\begin{aligned} \alpha_d &\sim \mathcal{N}(0, 10), \quad d = 1, \ldots, 12, \\ \sigma_d^2 &\sim \text{Inv-Gamma}(1, 1), \quad d = 1, \ldots, 12, \\ \beta_{k,d} &\sim \mathcal{N}(1, 5), \quad d = 1, \ldots, 12, \quad k = 1, \ldots, 3, \\ \theta_k^l &\sim U(-\infty, \infty), \quad k = 1, \ldots, 3. \end{aligned} \tag{4.18}$$

The Gibbs sampling took 8.4 minutes. We also implemented the same model in the Stan language (Stan Development Team, 2020), which provides the highly optimized No-U-Turn Hamiltonian Monte Carlo inference (Neal, 2011; Hoffman and Gelman, 2014) in R. The Stan model uses the routine $\mathcal{O}(N^3)$ matrix inversion method. In comparison, Stan took 19 minutes to sample 500 times from each of the 4 parallel chains. Therefore, it would have taken hours and large memory to obtain a reasonable number of samples using Stan. In contrast, our inference algorithm can be performed orders of magnitude faster on a personal computer.

Posterior distributions of θ_k^w, $d = 1, \ldots, 3$, are given in Figure 4.1a, where the mode of each density plot aligns closely with the true value of θ_k^w indicated by the red dashed line. Posterior means of α_d and σ_d^2 for $d = 1, \ldots, 12$ are illustrated in Figure 4.1b, 4.1c and 4.1d, respectively, with vertical bars corresponding to the 95% credible intervals. It is evident from the plots that the Gibbs sampler provides accurate estimates for both $\boldsymbol{\alpha}$ and $\boldsymbol{\Sigma}$. For the factor loadings $\boldsymbol{\beta}$, the smaller loadings are estimated with relatively larger uncertainty, while the larger loadings are lightly underestimated. Nonetheless, the 95% credible intervals still provide reasonable coverage. Figure 4.2 shows a satisfactory agreement between the posterior means of the latent factor processes and their respective true values.

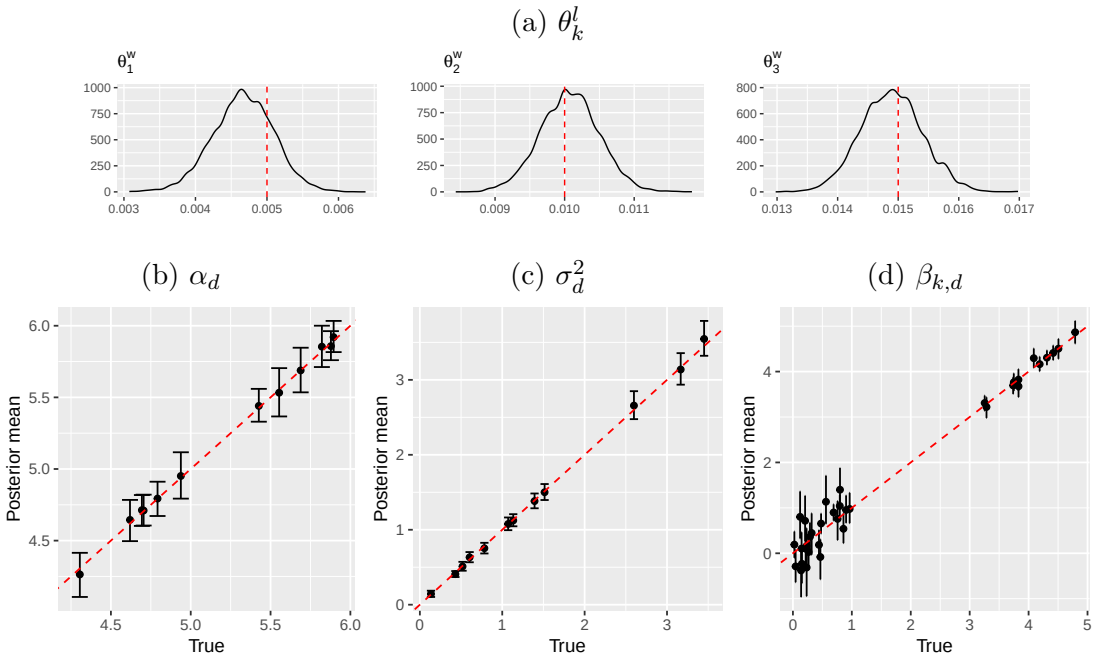


Figure 4.1: Posterior estimates for the Gaussian-GPFA model. (a) Estimated posterior distribution densities of θ_k^w, $k = 1, 2, 3$. The red dashed lines correspond to the respective true values used for simulating data. (b) and (c) Estimated posterior mean versus truth of α_d, σ_d^2 and $\beta_{k,d}$, $d = 1, \ldots, 12$, $k = 1, 2, 3$.

4.3.2 Poisson-GPFA Model

For the Poisson-GPFA model, we simulate 10 observed processes conditional on 2 latent GPs on 2000 time bins of length 0.1s. The parameters are simulated from

$$\alpha_d \sim \mathcal{N}(1, 0.5), \quad d = 1, \ldots, 10,$$
$$\boldsymbol{\beta} \sim \begin{pmatrix} \boldsymbol{U(1,2)} & U(0, 0.5) \\ U(0, 0.5) & \boldsymbol{U(1,2)} \end{pmatrix},$$

with the GP lengthscale parameters set to $\boldsymbol{\theta}^l = (0.01, 0.0065)$. Similar to the previous simulation, we fix all GP widthscale parameters to 0.25 to avoid model redundancy. Here, the parameter values are chosen so that the simulated Poisson count in each bin mimics the value of spike count in a bin, with most spike counts between 1 and 6 in each 0.1s bin. Priors on $\boldsymbol{\theta}$, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the same as those in (4.18). MCMC sampling took 8.46 minutes.

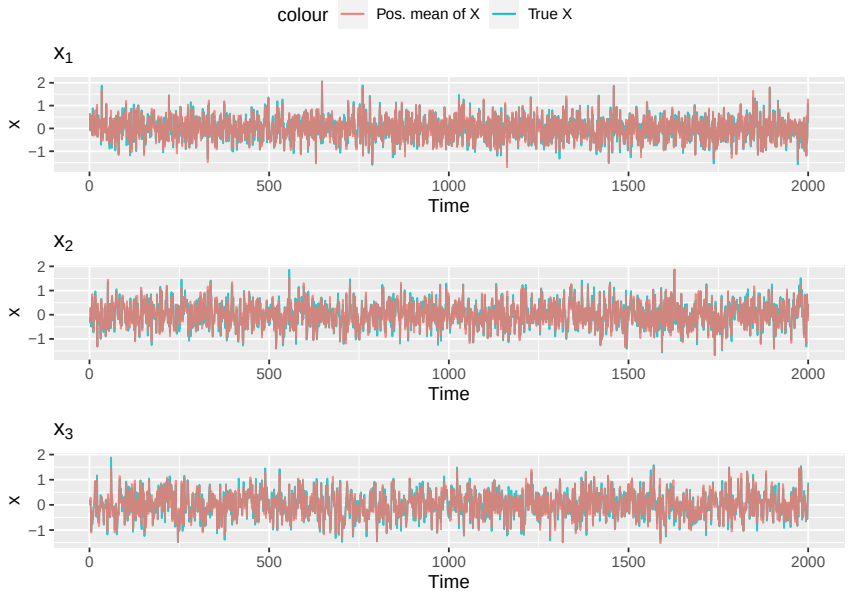


Figure 4.2: Posterior mean of the latent factor processes $\boldsymbol{x}_{:,k}, k = 1, 2, 3$ plotted with the true factor processes for the Gaussian-GPFA model.

Posterior estimates of the parameters are depicted in Figure 4.3. For conciseness, posterior distributions are displayed only for half of the elements of $\boldsymbol{\beta}$. Despite slight bias observed for some of the parameters, the proposed MCMC algorithm provides satisfactory estimation accuracy overall. Posterior means of the latent GPs are given in Figure 4.4, showing agreement with the true GPs.

4.4 Discussion

Our work leverages the superfast Toeplitz system solver in Ling (2019) and its efficient R/C++ implementation (Ling and Lysy, 2020) to design efficient MCMC inference algorithms for high-dimension neural dynamics models with latent Gaussian processes. For latent GPs of dimension N, each step of the superfast MCMC sampling scales as $\mathcal{O}(N \log^2 N)$, contrasting with the $\mathcal{O}(N^3)$ routine GP evaluations. We have demonstrated the applicability of our proposed samplers to the Gaussian Process Factor Analysis (GPFA) model and its variants with non-Gaussian likelihoods. Both the GPFA and non-Gaussian GPFA models are widely used tools for analyzing multi-neuron spike trains. GPFA inference

is typically done using optimization-based methods like the EM algorithm, which, while computationally scalable, lacks uncertainty quantification. For non-Gaussian GPFA models without closed-form posterior for the latent GPs, the proposed MCMC sampler offers an exact inference approach as an alternative to the variational approximation methods commonly used for this type of model. Although automated general-purpose MCMC software such as JAGS (Plummer, 2003) and Stan (Stan Development Team, 2020) can also be employed for inference, they are not tailored for handling high-dimensional GPs in the latent space and therefore can be extremely slow.

Our simulation studies demonstrate that both inference algorithms for GPFA with either Gaussian or Poisson likelihood provide satisfactory estimation results when applied to small simulated datasets. Inference accuracy and efficiency can be further improved in several ways. First, better proposal distributions can be designed to achieve more efficient sampling with less correlated samples. Adaptive methods, for instance, can be considered to devise a more optimal scheme for proposing samples (Roberts and Rosenthal, 2009). Second, it is worth exploring more appropriate prior choices for the GP lengthscale, such as the penalized complexity prior (Simpson et al., 2017). This is particularly crucial in real data analysis where prior knowledge about the GP behaviour is limited. One potential concern to be addressed in future work is the numerical stability of the Toeplitz solver for long-memory GPs. Models with long-memory GPs might lead to ill-conditioned covariance matrix (Chen et al., 2006), causing numerical errors for the generalized Schur algorithm. Therefore, inference algorithms should be designed with caution to handle these scenarios.

Verifying the performance of the proposed algorithms on real spike-train data is a critical part of future work. The computational benefits of our proposed algorithms are best demonstrated on long recordings with a large number of time bins, which makes the advantage of $\mathcal{O}(N \log^2 N)$ complexity over $\mathcal{O}(N^3)$ complexity more pronounced. Since the GPFA model does not accommodate non-stationarity in the data, the real data analysis should focus on long spike trains that are close to stationarity to align with the GPFA assumptions.

An important extension of this work is to use it as a foundation to modify the model in Chapter 3 to a multi-neuron spike-train model with memory, specifically temporal correlation. To avoid choosing a bin size for spike counts, the extended GPFA in Section 4.2.4 can be transformed into a logistic-GPFA model, where the spiking probability is determined by combining the latent factor processes. Some computationally intensive update blocks of the MCMC sampler can be replaced by sampling from an approximate conditional distribution (see, e.g., Wei et al., 2022). Alternatively, if the exact inference is preferred, the sampling efficiency of the MCMC algorithm can be improved by carefully designing a proposal distribution for $\boldsymbol{x}_k$ which targets sample space with higher acceptance

probability. A surrogate model can be constructed such that the likelihood is replaced by a mode-quadrature Normal approximation, whose mode is the mean log firing rate and whose variance is the first-order Taylor expansion at the mode. This results in the original Gaussian-GPFA model where the full conditional distribution $p(\boldsymbol{x}_k \mid \boldsymbol{Y})$ is available in closed form in (4.15). Therefore, an appropriate proposal distribution is $p(\boldsymbol{x}_k \mid \boldsymbol{Y})$ based on the surrogate model. However, it remains a challenge to derive an efficient method to simulate from $p(\boldsymbol{x}_k \mid \boldsymbol{Y})$.

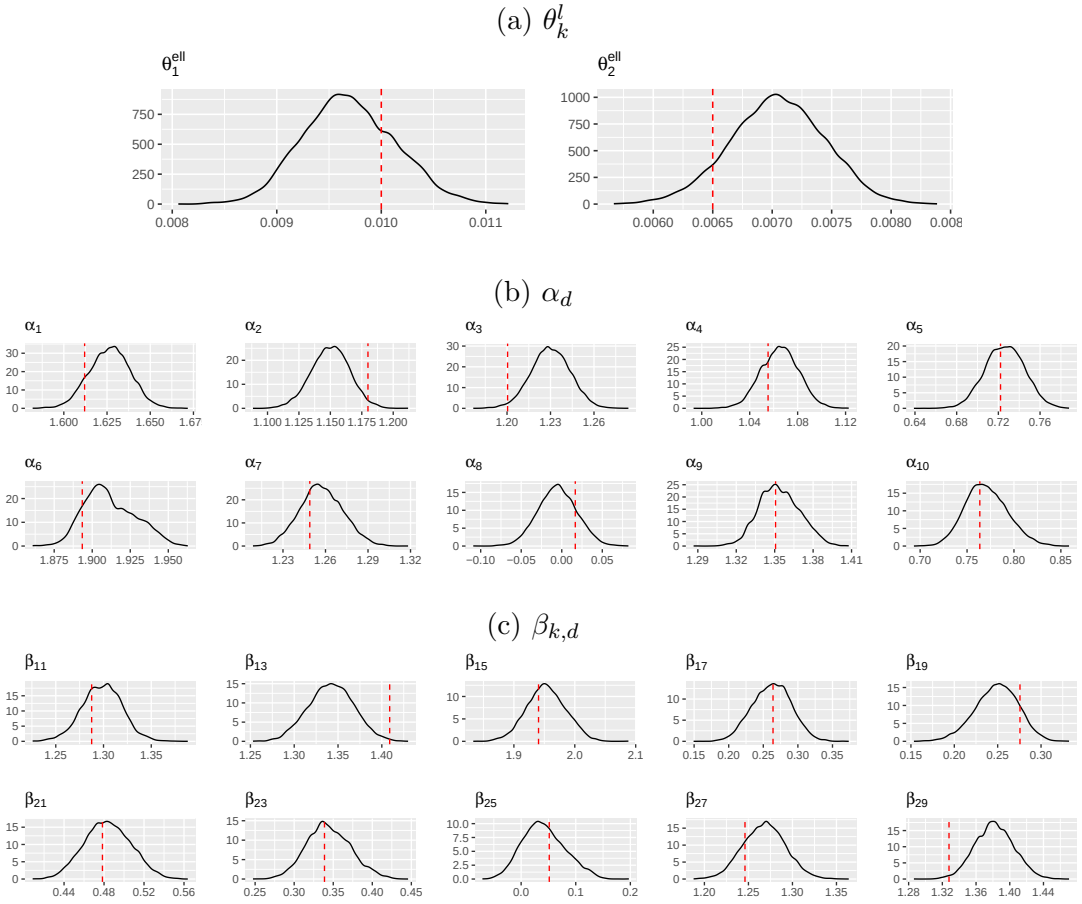


Figure 4.3: Estimated posterior distributions for the Poisson-GPFA model parameters. Posterior distributions are shown for half of the elements of $\boldsymbol{\beta}$. The vertical red dashed lines correspond to the true values.

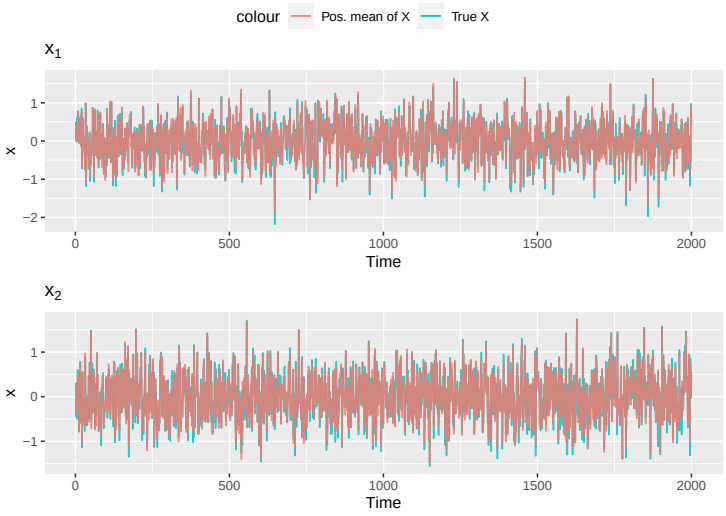


Figure 4.4: Posterior mean of the latent factor processes $\boldsymbol{x}_{:,k}, k = 1, 2, 3$ plotted with the true factor processes for the Poisson-GPFA model.

Chapter 5

Conclusion

The main contribution of this thesis lies in scalable Bayesian computation and statistical modelling across different real-world applications. It presents several methodologies for improving the computational efficiency for fitting Bayesian hierarchical models containing high-dimensional latent variables, with a particular focus on those described by Gaussian processes (GPs). A wide range of models fall under the scope of latent GP models, all benefiting from various techniques for computational accelerations. These techniques are approached from multiple angles, including model specification and model estimation. From the model specification perspective, latent processes can be replaced by an approximate representation such as a low-rank representation. From the model estimation perspective, faster evaluation routines for probability distribution densities can be leveraged to reduce the cost of model fitting, leading to either exact or approximate density. Throughout this thesis, we tackle the high-dimensional statistical challenges from these two perspectives.

Chapter 2 presented a fast inference method tailored for an existing model, specifically the GEV-GP model, which features generalized extreme value likelihoods and GP priors over a two-dimensional spatial domain for the GEV parameters. This chapter employed both an SPDE-approximated representation of the GPs and an approximation to the posterior distribution through the Laplace method. It demonstrated that the proposed inference method is orders of magnitude faster than a state-of-the-art MCMC method while maintaining comparable estimation accuracy. In Chapter 3, we proposed a novel latent factor multi-neuron spike-train model, filling a gap in the literature by providing a continuous-time multivariate point process model for studying the correlation between neural spike trains. Inference for this model is notably challenging due to both the large number of neurons and the high-dimensional temporal processes involved. These challenges are addressed through an approximate inference algorithm designed to compute the posterior

of interest via a series of approximation steps. Finally, Chapter 4 explored efficient and exact inference for latent GP models using a superfast GP evaluation method scaling as $\mathcal{O}(N \log^2 N)$ rather than the conventional $\mathcal{O}(N^3)$ time. This chapter presented two superfast MCMC inference algorithms, alongside simulation studies for two popular classes of spike-train models, one with closed-form inference and one without. An important implication of the proposed inference algorithms is the potential extension of the multi-neuron spike-train model introduced in Chapter 3.

The work in this thesis suggests several avenues for future research. While various sparse covariance matrix representations have been proposed in the literature, such as the one employed in Chapter 2, they are predominantly used for univariate GPs with scalar observations at each time point. Extending these representations to multivariate settings may potentially reduce the computational cost for the multivariate GPs discussed in Chapter 3. One well-known technique in this regard is the Karhunen–Loève expansion for multivariate Wiener processes, although our simulations using data in Chapter 3 did not show improved inference speed with its use. Nevertheless, exploring the applicability of other more sophisticated sparse multivariate GP representations remains an area of interest. Additionally, the methods presented in Chapter 4 offer an alternative sampling-based inference technique for extending the model in Chapter 3 to incorporate temporal memory. By adapting blocks of the superfast MCMC sampler via approximations, more efficient and flexible models for multi-neuron spike trains can be developed.

References

Abramowitz, M. and I. Stegun (1972). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. United States Government Printing Office: Washington, DC.

Aertsen, A., G. L. Gerstein, M. K. Habib, and G. Palm (1989). Dynamics of neuronal firing correlation: modulation of "effective connectivity". *Journal of neurophysiology 61*(5), 900–917.

Aguilar, O. and M. West (2000). Bayesian dynamic factor models and portfolio allocation. *Journal of Business and Economic Statistics 18*(3), 338–357.

Ammar, G. S. and W. B. Gragg (1987). The generalized Schur algorithm for the superfast solution of Toeplitz systems. In *Rational approximation and its applications in mathematics and physics*, pp. 315–330. Springer.

Ammar, G. S. and W. B. Gragg (1988). Superfast solution of real positive definite Toeplitz systems. *SIAM Journal on Matrix Analysis and Applications 9*(1), 61–76.

Ammar, G. S. and W. B. Gragg (1989). Numerical experience with a superfast real Toeplitz solver. *Linear Algebra and its Applications 121*, 185–206.

Anderson, T. W. (2003). *An Introduction to Multivariate Statistical Analysis* (3rd Edition ed.). John Wiley & Sons.

Archer, E., I. M. Park, L. Buesing, J. Cunningham, and L. Paninski (2015). Black box variational inference for state space models.

Arieli, A., A. Sterkin, A. Grinvald, and A. Aertsen (1996). Dynamics of ongoing activity: explanation of the large variability in evoked cortical responses. *Science 273*(5283), 1868–1871.

Averbeck, B. B., P. E. Latham, and A. Pouget (2006). Neural correlations, population coding and computation. *Nature Reviews Neuroscience 7*, 358–366.

Bakka, H., H. Rue, G.-A. Fuglstad, A. Riebler, D. Bolin, E. T. Krainski, D. P. Simpson, and F. Lindgren (2018). Spatial modelling with R-INLA: A review. *Wiley Interdisciplinary Reviews 10*(6).

Banerjee, S., B. P. Carlin, and A. E. Gelfand (2003). *Hierarchical modeling and analysis for spatial data* (1st Edition ed.). Chapman and Hall/CRC.

Banerjee, S. and A. Roy (2014). *Linear Algebra and Matrix Analysis for Statistics* (1st Edition ed.). Chapman and Hall/CRC.

Barndorff-Nielsen, O. E. and D. R. Cox (1989). *Asymptotic techniques for use in statistics*, Volume 11. Springer.

Bastos, A. M. and J.-M. Schoffelen (2016). A tutorial review of functional connectivity analysis methods and their interpretational pitfalls. *Frontiers in Systems Neuroscience 9*.

Behseta, S., R. E. Kass, and G. L. Wallstrom (2005). Hierarchical models for assessing variability among functions. *Biometrika 92*(2), 419–434.

Benchenane, K., A. Peyrache, M. Khamassi, P. L. Tierney, Y. Gioanni, F. P. Battaglia, and S. I. Wiener (2010). Coherent theta oscillations and reorganization of spike timing in the hippocampal- prefrontal network upon learning. *Neuron 66*, 921–936.

Besag, J. (1974). Spatial interaction and the statistical analysis of lattice systems (with discussion). *Journal of the Royal Statistical Society. Series B (Statistical Methodology) 36*, 192–236.

Besag, J., J. York, and A. Mollié (1991). Bayesian image restoration with applications in spatial statistics (with discussion). *Annals of the Institute of Statistical Mathematics 43*, 19–59.

Bianconcini, S. (2014). Asymptotic properties of adaptive maximum likelihood estimators in latent variable models. *Bernoulli 20*(3).

Bianconcini, S. and S. Cagnone (2012). Estimation of generalized linear latent variable models via fully exponential Laplace approximation. *Journal of Multivariate Analysis 112*, 183–193.

Billeh, Y. N., M. T. Schaub, A. C. A., B. M., and K. C. (2014). Scalable and accurate method for neuronal ensemble detection in spiking neural networks. *Journal of Neuroscience Methods 236*, 92–106.

Blanchet, J. and A. C. Davison (2011). Spatial modelling of extreme snow depth. *Annals of Applied Statistics 5*, 1699–1725.

Blei, D. M., A. Kucukelbir, and J. D. McAuliffe (2016). Variational inference: A review for statisticians. *Journal of the American Statistical Association 112*, 859 – 877.

Bliss, T. V. P. and S. F. Cooke (2011). Long-term potentiation and long-term depression: a clinical perspective. *Clinics 66*, 3–17.

Bonilla, E. V., K. Krauth, and A. Dezfouli (2019). Generic inference in latent Gaussian process models. *Journal of Machine Learning Research 20*, 1–63.

Breslow, N. E. and X. Lin (1995). Bias correction in generalised linear mixed models with a single component of dispersion. *Biometrika 82*(1), 81–91.

Brown, E. N. (2005). Theory of point processes for neural systems. In *Methods and Models in Neurophysics*, Chapter 14, pp. 691–727. Elsevier.

Buesing, L., T. A. Machado, J. P. Cunningham, and L. Paninski (2014). Clustered factor analysis of multineuronal spike data. In *Advances in Neural Information Processing Systems*, Volume 27.

Cardoso de Oliveira, S., A. Gribova, O. Donchin, H. Bergman, and E. Vaadia (2001). Neural interactions between motor cortical hemispheres during bimanual and unimanual arm movements. *European Journal of Neuroscience 14*(11), 1881–1896.

Casson, E. and S. Coles (1999). Spatial regression models for extremes. *Extremes 1*, 449–468.

Castro-Camilo, D., R. Huser, and H. Rue (2019). A spliced gamma-generalized Pareto model for short-term extreme wind speed probabilistic forecasting. *Journal of Agricultural, Biological, and Environmental Statistics 24*, 517–534.

Castro-Camilo, D., L. Mhalla, and T. Opitz (2020). Bayesian space-time gap filling for inference on extreme hot-spots: an application to red sea surface temperatures. *Extremes 24*, 105–128.

Chelaru, M. and V. Dragoi (2008). Efficient coding in heterogeneous neuronal populations. *Proceedings of the National Academy of Sciences 105*, 16344–16349.

Chen, M., R. Ramezan, and M. Lysy (2023). Fast and scalable approximate inference for spatial extreme value models.

Chen, W. W., C. M. Hurvich, and Y. Lu (2006). On the correlation matrix of the discrete fourier transform and the fast solution of large Toeplitz systems for long-memory time series. *Journal of the American Statistical Association 101*(474), 812–822.

Chen, Y., T. A. Davis, W. W. Hager, and S. Rajamanickam (2008). Algorithm 887: Cholmod, supernodal sparse Cholesky factorization and update/downdate. *ACM Trans. Math. Softw. 35*(3), 1–14.

Churchland, A. K., R. Kiani, R. Chaudhuri, X.-J. Wang, A. Pouget, and M. N. Shadlen (2011). Variance as a signature of neural computations during decision making. *Neuron 69*(4), 818–831.

Coles, S. and E. Powell (1996). Bayesian methods in extreme value modeling: A review and new developments. *International Statistical Review 64*(1), 119–136.

Coles, S. G. (2001). *An Introduction to Statistical Modeling of Extreme Values*. Springer.

Coles, S. G. and E. Casson (1998). Extreme value modelling of hurricane wind speeds. *Structural Safety 20*, 283–296.

Cooley, D., D. Nychka, and P. Naveau (2007). Bayesian spatial modeling of extreme precipitation return levels. *Journal of the American Statistical Association 102*, 824–840.

Cooley, D. and S. Sain (2010). Spatial hierarchical modeling of precipitation extremes from a regional climate model. *Journal of Agricultural, Biological, and Environmental Statistics 15*(3), 381–402.

Coull, B. A., M. Mezzetti, and L. M. Ryan (2003). A bayesian hierarchical model for risk assessment of methylmercury. *Journal of Agricultural, Biological, and Environmental Statistics 8*, 253–270.

Cressie, N. A. C. (1993). *Statistics for Spatial Data*. John Wiley & Sons, Inc.

Critchley, H. D. and E. T. Rolls (1996). Hunger and satiety modify the responses of olfactory and visual neurons in the primate orbitofrontal cortex. *Journal of neurophysiology 75 4*, 1673–86.

Daniels, M. J., F. Dominici, J. M. Samet, and S. L. Zeger (2000). Estimating particulate matter-mortality dose-response curves and threshold levels: an analysis of daily time-series for the 20 largest US cities. *American journal of epidemiology 152*(5), 397–406.

Daniels, M. J. and R. E. Kass (1998). A note on first-stage approximation in two-stage hierarchical models. *Sankhyā: The Indian Journal of Statistics, Series B 60*, 19–30.

Davison, A., S. A. Padoan, and M. Ribatet (2012). Statistical modeling of spatial extremes. *Statistical Sciences 27*, 161–186.

de Valpine, P., D. Turek, C. Paciorek, C. Anderson-Bergman, D. Temple Lang, and R. Bodik (2017). Programming with models: writing statistical algorithms for general model structures with NIMBLE. *Journal of Computational and Graphical Statistics 26*, 403–417.

deCharms, R. C. and M. M. Merzenich (1996, Jun 13). Primary cortical representation of sounds by the coordination of action-potential timing. *Nature 381*, 610–613.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological) 39*(1), 1–22.

Dias, S., A. E. Ades, N. J. Welton, J. P. Jansen, and A. J. Sutton (2018). *Network Meta-Analysis for Decision-Making* (1st Edition ed.). John Wiley & Sons Ltd.

Diba, K., A. Amarasingham, K. Mizuseki, and G. Buzsáki (2014). Millisecond timescale synchrony among hippocampal neurons. *The Journal of Neuroscience 34*, 14984–14994.

Dominici, F., J. M. Samet, and S. L. Zeger (2000). Combining evidence on air pollution and daily mortality from the 20 largest US cities: a hierarchical modelling strategy. *Journal of the Royal Statistical Society Series A: Statistics in Society 163*(3), 263–302.

Duane, S., A. D. Kennedy, B. J. Pendleton, and D. Roweth (1987). Hybrid Monte Carlo. *Physics Letters B 195*, 216–222.

Duncker, L. and M. Sahani (2018). Temporal alignment and latent Gaussian process factor inference in population spike trains. In *Advances in Neural Information Processing Systems*.

Durbin, J. (1960). The fitting of time-series models. *Revue de l'Institut International de Statistique 28*, 233–244.

Dyrrdal, A. V., A. Lenkoski, T. L. Thorarinsdottir, and F. Stordal (2015). Bayesian hierarchical modeling of extreme hourly precipitation in Norway. *Environmetrics 26*, 89–106.

Eickhoff, S. B. and V. I. Müller (2015). Functional connectivity. In A. W. Toga (Ed.), *Brain Mapping*, pp. 187–201. Waltham: Academic Press.

Embrechts, P., C. Klüppelberg, and T. Mikosch (1997). *Modelling Extreme Events: for Insurance and Finance*. Springer.

Environment and Climate Change Canada (2022). Monthly climate summaries. https://climate-change.canada.ca/climate-data/#/monthly-climate-summaries.

Fawcett, L. and D. Walshaw (2006). A hierarchical model for extreme wind speeds. *Applied Statistics 55*, 631–646.

Fellous, J.-M., P. H. E. Tiesinga, P. J. Thomas, and T. J. Sejnowski (2004). Discovering spike patterns in neuronal responses. *The Journal of Neuroscience 24*, 2989–3001.

Fine, S. and K. Scheinberg (2002). Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research 2*, 243–264.

Folks, J. L. and R. S. Chhikara (1978). The inverse gaussian distribution and its statistical application - a review. *Journal of Royal Statistical Society, Series B 40*(3), 263–289.

Fujisawa, S., A. Amarasingham, M. T. Harrison, and G. Buzsáki (2008). Behavior-dependent short-term assembly dynamics in the medial prefrontal cortex. *Nature Neuroscience 11*, 823–833.

Gal, Y., Y. Chen, and Z. Ghahramani (2015). Latent Gaussian processes for distribution estimation of multivariate categorical data.

Gao, Y., E. Archer, L. Paninski, and J. P. Cunningham (2016). Linear dynamical neural population models through nonlinear embeddings. In *30th Conference on Neural Information Processing Systems*.

García, J. A., J. Martín, L. Naranjo, and F. J. Acero (2018). A Bayesian hierarchical spatio-temporal model for extreme rainfall in Extremadura (Spain). *Hydrological Sciences Journal 63*, 878–894.

Geirsson, Ó. P., B. Hrafnkelsson, and D. P. Simpson (2014). Computationally efficient spatial modeling of annual maximum 24-h precipitation on a fine grid. *Environmetrics 26*, 339–353.

Geirsson, Ó. P., B. Hrafnkelsson, D. P. Simpson, and H. Sigurdarson (2020). LGM split sampler: An efficient MCMC sampling scheme for latent Gaussian models. *Statistical Science 35*, 218–233.

Gelfand, A. E. and A. F. M. Smith (1990). Sampling-based approaches to calculating marginal densities. *Journal of the American Statistical Association 85*(410), 398–409.

Gelman, A., J. B. Carlin, H. S. Starn, D. B. Dunson, A. Vehtari, and D. B. Rubin (2013). *Bayesian Data Analysis* (3rd Edition ed.). Chapman and Hall/CRC.

Gelman, A. and D. B. Rubin (1992). Inference from iterative simulation using multiple sequences. *Statistical Science 7*(4), 457–472.

Geman, S. and D. Geman (1984). Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence 6*, 721–741.

Gerstein, G. L., D. H. Perkel, and J. E. Dayhoff (1985). Cooperative firing activity in simultaneously recorded populations of neurons: detection and measurement. *Journal of Neuroscience 94*, 1513–1528.

Gerstner, W., W. M. Kistler, R. Naud, and L. Paninski (2014). *Neuronal Dynamics: From Single Neurons to Networks and Models of Cognition*. Cambridge University Press.

Ghosh, J. and D. B. Dunson (2009). Default prior distributions and efficient posterior computation in Bayesian factor analysis. *Journal of Computational and Graphical Statistics 18*(2), 306–320.

Gilli, M. and E. Këllezi (2006). An application of extreme value theory for measuring financial risk. *Computational Economics 27*, 1–23.

Gohberg, I. and A. Semencul (1972). On the inversion of finite Toeplitz matrices and their continuous analogs. *Mat. issled 2*, 201–233.

Golub, G. H. and C. F. Van Loan (1996). *Matrix Computations* (Third ed.). The Johns Hopkins University Press.

Gong, L. and J. M. Flegal (2016). A practical sequential stopping rule for high-dimensional Markov Chain Monte Carlo. *Journal of Computational and Graphical Statistics 25*(3), 684–700.

Handcock, M. and M. L. Stein (1993). A Bayesian analysis of kriging. *Technometrics 35*(4), 403–410.

Harrison, M., A. Amarasingham, and R. Kass (2013). Statistical identification of synchronous spiking. In P. D. Lorenzo and J. Victor (Eds.), *Spike Timing: Mechanisms and Function*, pp. 77–120. Taylor & Francis.

Hastings, W. K. (1970). Monte Carlo sampling methods using Markov chains and their applications. *Biometrika 57*(1), 97–109.

Hazra, A. K., R. Huser, and Á. V. Jóhannesson (2021). Latent Gaussian models for high-dimensional spatial extremes.

Heinz, A., T. Siessmeier, J. Wrase, D. Hermann, S. Klein, S. M. Grüsser, H. Flor, D. F. Braus, H.-G. Buchholz, G. Gründer, M. Schreckenberger, M. N. Smolka, F. Rösch, K. F. Mann, and P. Bartenstein (2004). Correlation between dopamine d(2) receptors in the ventral striatum and central processing of alcohol cues and craving. *The American Journal of Psychiatry 161*(10), 1783–1789.

Hernandez, J. S. and D. E. Moorman (2020). Orbitofrontal cortex encodes preference for alcohol. *eNeuro 7*.

Herzog, R., A. Morales, S. Mora, J. Araya, M.-J. Escobar, A. G. Palacios, and R. Cofré (2021). Scalable and accurate method for neuronal ensemble detection in spiking neural networks. *PLoS ONE 16*.

Higham, N. J. (2002). *Accuracy and stability of numerical algorithms*. SIAM.

Hines, M. L. and N. T. Carnevale (1997). The neuron simulation environment. *Neural Computation 9*, 1179–1209.

Hodgkin, A. L. and A. F. Huxley (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology 117*, 500–544.

Hoffman, M. D. and A. Gelman (2014). The No-U-Turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research 15*, 1593–1623.

Hrafnkelsson, B., S. Siegert, R. Huser, H. Bakka, and Á. V. Jóhannesson (2021). Max-and-smooth: A two-step approach for approximate Bayesian inference in latent gaussian models. *Bayesian Analysis 16*(2), 611–638.

Humphries, M. D. (2011). Spike-train communities: Finding groups of similar spike trains. *Journal of Neuroscience 31*(6), 2321–2336.

Jalbert, J., A.-C. Favre, C. Bélisle, and J.-F. Angers (2017). A spatiotemporal model for extreme precipitation simulated by a climate model, with an application to assessing changes in return levels over North America. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 66*, 941–962.

Jalbert, J., C. Genest, and L. Perreault (2022). Interpolation of precipitation extremes on a large domain toward IDF curve construction at unmonitored locations. *Journal of Agricultural, Biological and Environmental Statistics 27*, 461–486.

Jóhannesson, Á. V., S. Siegert, R. Huser, H. Bakka, and B. Hrafnkelsson (2022). Approximate Bayesian inference for analysis of spatiotemporal flood frequency data. *The Annals of Applied Statistics 16*(2), 905–935.

Jordan, M. I., Z. Ghahramani, T. Jaakkola, and L. K. Saul (1999). An introduction to variational methods for graphical models. *Machine Learning 37*, 183–233.

Jun, J. J., N. A. Steinmetz, J. H. Siegle, D. J. Denman, M. Bauza, B. Barbarits, A. K. Lee, C. Anastassiou, A. Andrei, C. Aydin, M. Barbic, T. J. Blanche, V. Bonin, J. Couto, B. Dutta, S. L. Gratiy, D. A. Gutnisky, M. Häusser, B. Karsh, P. Ledochowitsch, C. M. Lopez, C. C. Mitelut, S. Musa, M. Okun, M. Pachitariu, J. Putzeys, P. D. Rich, C. Rossant, W. lung Sun, K. Svoboda, M. Carandini, K. D. Harris, C. Koch, J. O'Keefe, and T. D. Harris (2017). Fully integrated silicon probes for high-density recording of neural activity. *Nature 551*, 232–236.

Jurjuţ, O. F., M. D. Gheorghiu, W. Singer, D. Nikolić, and R. C. Muresan (2019). Hold your methods! how multineuronal firing ensembles can be studied using classical spike-train analysis techniques. *Frontiers in Systems Neuroscience 13*.

Kailath, T. and A. H. Sayed (1987). *Fast reliable algorithms for matrices with structure*. SIAM.

Kaiser, H. F. (1958). The Varimax criterion for analytic rotation in factor analysis. *Psychometrika 23*(3), 187–200.

Kalat, J. W. (2018). *Biological Psychology* (13th Edition ed.). Wadsworth Publishing.

Kaminska, B. (2020). *Value and Action Encoding for Natural Outcomes and Ethanol in the Medial Prefrontal and Orbitofrontal Cortex*. Phd thesis, University of Massachusetts Amherst. Available at https://scholarworks.umass.edu/dissertations_2/2034/.

Kass, R. E., U. T. Eden, and E. N. Brown (2014). *Analysis of Neural Data*. Springer.

Kass, R. E. and D. Steffey (1989). Approximate Bayesian inference in conditionally independent hierarchical models (parametric empirical Bayes model). *Journal of the American Statistical Association 84*, 717–726.

Kass, R. E. and V. Ventura (2001). A spike-train probability model. *Neural Computation 13*, 1713–1720.

Kass, R. E. and V. Ventura (2006). Spike count correlation increases with length of time interval in the presence of trial-to-trial variation. *Neural Computation 18*(11), 2583–2591.

Keeley, S. L., D. M. Zoltowski, Y. Yu, J. L. Yates, S. L. Smith, and J. W. Pillow (2019). Efficient non-conjugate Gaussian process factor models for spike count data using polynomial approximations. In *International Conference on Machine Learning*.

Knorr-Held, L. and H. Rue (2002). On block updating in markov random field models for disease mapping. *Scandinavian Journal of Statistics 29*, 597–614.

Kristensen, K., A. Nielsen, C. W. Berg, H. Skaug, and B. M. Bell (2016). TMB: Automatic differentiation and Laplace approximation. *Journal of Statistical Software 70*(5), 1–21.

Kuss, M. and C. E. Rasmussen (2005). Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research 6*, 1679–1704.

Lapicque, L. (1907). Recherches quantitatives sur l'excitation e´lectrique des nerfs traite´e comme une polarization. *Journal de Physiologie et de Pathologie Générale 9*, 620–635.

Levinson, N. (1946). The Wiener (root mean square) error criterion in filter design and prediction. *Journal of Mathematics and Physics 25*(1), 261–278.

Li, J. and S. X. Chen (2012). Two sample tests for high-dimensional covariance matrices. *The Annals of Statistics 40*, 908–940.

Lin, A., Y. Zhang, J. Heng, S. A. Allsop, K. M. Tye, P. E. Jacob, and D. E. Ba (2018). Clustering time series with nonlinear dynamics: A Bayesian non-parametric and particle-based approach. In *International Conference on Artificial Intelligence and Statistics*.

Lindgren, F., D. Bolin, and H. Rue (2022). The SPDE approach for Gaussian and non-Gaussian fields: 10 years and still running. *Spatial Statistics 50*, 100599.

Lindgren, F. K. and H. Rue (2015). Bayesian spatial modelling with R-INLA. *Journal of Statistical Software 63*, 1–25.

Lindgren, F. K., H. Rue, and J. Lindström (2011). An explicit link between Gaussian fields and Gaussian Markov random fields: The stochastic partial differential equation approach (with discussion). *Journal of the Royal Statistical Society. Series B (Statistical Methodology) 73*, 423–498.

Lindsay, B. G. (1988). Composite likelihood methods. *Contemporary Mathematics 60*, 221–239.

Ling, Y. (2019, December). *Superfast Inference for Stationary Gaussian Processes in Particle Tracking Microrheology*. PhD thesis, University of Waterloo, Waterloo, ON. Available at http://hdl.handle.net/10012/15338.

Ling, Y. and M. Lysy (2020). *SuperGauss: Superfast Likelihood Inference for Stationary Gaussian Time Series*. R package version 1.0.2.

Liu, H., Y.-S. Ong, X. Shen, and J. Cai (2020). When Gaussian process meets big data: A review of scalable GPs. In *IEEE transactions on neural networks and learning systems*.

Lopes-dos-Santos, V., S. Ribeiro, and A. B. L. Tort (2013). Detecting cell assemblies in large neuronal populations. *Journal of Neuroscience Methods 220*, 149–166.

Lopez, H. F. and M. West (2004). Bayesian model assessment in factor analysis. *Statistica Sinica 14*, 41–67.

Louie, K. and P. W. Glimcher (2012). Efficient coding and the neural representation of value. *Annals of the New York Academy of Sciences 1251*, 13–32.

Lunardini, F., C. Casellato, M. Bertucco, T. D. Sanger, and A. L. G. Pedrocchi (2015). Muscle synergies in children with dystonia capture "healthy" patterns regardless the altered motor performance. *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC) 2015*, 2099–2102.

Lynall, M.-E., D. S. Bassett, R. W. Kerwin, P. J. McKenna, M. G. Kitzbichler, U. Muller, and E. T. Bullmore (2010). Functional connectivity and brain networks in schizophrenia. *The Journal of Neuroscience 30*, 9477–9487.

Lysy, M., N. S. Pillai, D. B. Hill, M. G. Forest, J. W. Mellnik, P. A. Vasquez, and S. A. McKinley (2016). Model comparison and assessment for single particle tracking in biological fluids. *Journal of the American Statistical Association 111*(516), 1413–1426.

Macke, J. H., L. Buesing, J. P. Cunningham, B. M. Yu, K. V. Shenoy, and M. Sahani (2011). Empirical models of spiking in neural populations. In *Advances in Neural Information Processing Systems*, Volume 24.

Mainen, Z. F. and A. Kepecs (2009). Neural representation of behavioral outcomes in the orbitofrontal cortex. *Current Opinion in Neurobiology 19*, 84–91.

Marshall, E. C. and D. J. Spiegelhalter (2007). Identifying outliers in Bayesian hierarchical models: a simulation-based approach. *Bayesian Analysis 2*, 409–444.

Martins, E. S. and J. R. Stedinger (2000). Generalized maximum-likelihood generalized extreme-value quantile estimators for hydrologic data. *Water Resources Research 36*(3), 737–744.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics 21*(6), 1087–1092.

Miller, D. L., R. Glennie, and A. E. Seaton (2020). Understanding the stochastic partial differential equation approach to smoothing. *Journal of Agricultural, Biological and Environmental Statistics 25*, 1–16.

Moorman, D. E. (2018). The role of the orbitofrontal cortex in alcohol use, abuse, and dependence. *Progress in Neuro-Psychopharmacology and Biological Psychiatry 87*, 85–107.

Morris, R. G. M. (2003). Long-term potentiation and memory. *Philosophical transactions of the Royal Society of London. Series B, Biological sciences 358*, 643–647.

Murray, I. and R. P. Adams (2010). Slice sampling covariance hyperparameters of latent Gaussian models. In *Advances in Neural Information Processing Systems*.

Murray, I., R. P. Adams, and D. J. C. MacKay (2010). Elliptical slice sampling. In *the 13th International Conference on Artificial Intelligence and Statistics*.

Myrick, H., R. F. Anton, X. Li, S. Henderson, D. J. Drobes, K. E. Voronin, and M. S. George (2004). Differential brain activity in alcoholics and social drinkers to alcohol cues: Relationship to craving. *Neuropsychopharmacology 29*, 393–402.

Nam, I.-S., K. Mengersen, and P. Garthwaite (2003). Multivariate meta-analysis. *Statistics in Medicine 22*, 2309–2333.

Neal, R. M. (2011). MCMC using Hamiltonian dynamics. In *The Handbook of Markov Chain Monte Carlo*, Chapter 5, pp. 113–162. Chapman & Hall / CRC Press.

Nelson, M. E. (2002). Multiscale spike train variability in primary electrosensory afferents. *Journal of Physiology-Paris 96*(5), 507–516.

Nickisch, H. and C. E. Rasmussen (2008). Approximations for binary Gaussian process classification. *Journal of Machine Learning Research 9*, 2035–2078.

Ogden, H. (2021). On the error in Laplace approximations of high-dimensional integrals. *Stat 10*, e380.

Okatan, M., M. Wilson, and E. Brown (2005). Analyzing functional connectivity using a network likelihood model of ensemble neural spiking activity. *Neural Computation 17*, 1927–1961.

Omi, T. and S. Shinomoto (2011). Optimizing time histograms for non-Poissonian spike trains. *Neural Computation 23*(12), 3125–3144.

Opitz, T. (2017). Latent Gaussian modeling and INLA: A review with focus on space-time applications. *Journal of the French Statistical Society 158*, 62–85.

Opitz, T., R. Huser, H. Bakka, and H. Rue (2018). INLA goes extreme: Bayesian tail regression for the estimation of high spatio-temporal quantiles. *Extremes 21*, 441–462.

Opschoor, A., A. Lucas, I. Barra, and D. J. C. van Dijk (2020). Closed-form multi-factor copula models with observation-driven dynamic factor loadings. *Journal of Business & Economic Statistics 39*, 1066–1079.

Ostojic, S., N. J.-B. Brunel, and V. Hakim (2009). How connectivity, background activity, and synaptic properties shape the cross-correlation between spike trains. *The Journal of Neuroscience 29*, 10234–10253.

Padoa-Schioppa, C. and J. A. Assad (2006). Neurons in the orbitofrontal cortex encode economic value. *Nature 441*, 223–226.

Paiva, A. R. C., S. Rao, I. Park, and J. C. Príncipe (2007). Spectral clustering of synchronous spike trains. In *2007 International Joint Conference on Neural Networks*, pp. 1831–1835.

Pandarinath, C., D. J. O'Shea, J. Collins, R. Józefowicz, S. D. Stavisky, J. C. Kao, E. M. Trautmann, M. T. Kaufman, S. I. Ryu, L. R. Hochberg, J. M. Henderson, K. V. Shenoy, L. F. Abbott, and D. Sussillo (2018). Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods 15*(10), 805–815.

Pei, F., J. Ye, D. M. Zoltowski, A. Wu, R. H. Chowdhury, H. Sohn, J. E. O'Doherty, K. V. Shenoy, M. T. Kaufman, M. Churchland, M. Jazayeri, L. E. Miller, J. Pillow, I. M. Park, E. L. Dyer, and C. Pandarinath (2021). Neural latents benchmark '21: Evaluating latent variable models of neural population activity. In *Advances in Neural Information Processing Systems*.

Penit-Soria, J., E. Audinat, and F. Crepel (1987). Excitation of rat prefrontal cortical neurons by dopamine: an in vitro electrophysiological study. *Brain research 425*(2), 263–274.

Perkel, D. H., G. L. Gerstein, and G. P. Moore (1967). Neuronal spike trains and stochastic point processes. II. simultaneous spike trains. *Biophysical journal 7*(5), 419–40.

Petersen, R. S., S. Panzeri, and M. E. Diamond (2001). Population coding of stimulus location in rat somatosensory cortex. *Neuron 32*, 503–514.

Pillow, J. W., J. Shlens, L. Paninski, A. Sher, A. M. Litke, E. J. Chichilnisky, and E. P. Simoncelli (2008). Spatio-temporal correlations and visual signalling in a complete neuronal population. *Nature 454*(7207), 995–999.

Pischedda, D., S. Palminteri, and G. Coricelli (2020). The effect of counterfactual information on outcome value coding in medial prefrontal and cingulate cortex: From an absolute to a relative neural code. *The Journal of Neuroscience 40*, 3268–3277.

Plummer, M. (2003). JAGS: Just another Gibbs sampler. https://mcmc-jags.sourceforge.io/.

Quiñonero-Candela, J. and C. E. Rasmussen (2005). A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research 6*, 1939–1959.

Ramezan, R., M. Chen, M. Lysy, and P. Marriott (2022). A multivariate point process model for simultaneously recorded neural spike trains.

Ramezan, R., P. Marriott, and S. Chenouri (2014). Multiscale analysis of neural spike trains. *Statistics in Medicine 33*, 238–256.

Ramezan, R., P. Marriott, and S. Chenouri (2016). Skellam process with resetting: a neural spike train model. *Statistics in Medicine 35*, 5717–5729.

Rasmussen, C. E. and C. K. I. Williams (2006). *Gaussian Processes for Machine Learning*. The MIT Press.

Reich, B. J. and B. A. Shaby (2012). A hierarchical max-stable spatial model for extreme precipitation. *The Annals of Applied Statistics 6*, 1430–1451.

Ribatet, M., R. Singleton, and R Core team (2022). *`SpatialExtremes`: Modelling Spatial Extremes*.

Roberts, G. O., A. Gelman, and W. R. Gilks (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *Annals of Applied Probability 7*, 110–120.

Roberts, G. O. and J. S. Rosenthal (2001). Optimal scaling for various Metropolis-Hastings algorithms. *Statistical Science 16*, 351–367.

Roberts, G. O. and J. S. Rosenthal (2009). Examples of adaptive MCMC. *Journal of Computational and Graphical Statistics 18*, 349–367.

Rolls, E. T. and F. Grabenhorst (2008). The orbitofrontal cortex and beyond: From affect to decision-making. *Progress in Neurobiology 86*, 216–244.

Rosenberg, M. D., M. D. Rosenberg, D. Scheinost, A. S. Greene, E. W. Avery, Y. H. Kwon, E. S. Finn, R. Ramani, M. Qiu, R. T. Constable, and M. M. Chun (2020). Functional connectivity predicts changes in attention over minutes, days, and months. *Proceedings of the National Academy of Sciences 117*, 3797–3807.

Rosenkranz, J. A. and A. A. Grace (2002). Cellular mechanisms of infralimbic and prelimbic prefrontal cortical inhibition and dopaminergic modulation of basolateral amygdala neurons in vivo. *Journal of Neuroscience 22*(1), 324–337.

Rubin, D. B. (1984). Bayesian justifiable and relevant frequency calculations for the applied statistician. *Annals of Statistics 12*, 1151–1172.

Rubinov, M. and O. Sporns (2010). Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage 52*, 1059–1069.

Rue, H. and L. Held (2005). *Gaussian Markov Random Fields: Theory and Applications* (1st Edition ed.). Chapman and Hall/CRC.

Rue, H., S. Martino, and N. Chopin (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of the Royal Statistical Society. Series B (Statistical Methodology) 71*, 319–392.

Rue, H., A. Riebler, S. H. Sørbye, J. B. Illian, D. P. Simpson, and F. K. Lindgren (2017). Bayesian computing with INLA: A review. *Annual Review of Statistics and Its Application 4*, 395–421.

Sang, H. and A. E. Gelfand (2008). Hierarchical modeling for extreme values observed over space and time. *Environmental and Ecological Statistics 16*, 407–426.

Sang, H. and A. E. Gelfand (2010). Continuous spatial process models for spatial extreme values. *Journal of Agricultural, Biological, and Environmental Statistics 15*, 49–56.

Sani, O. G., H. Abbaspourazad, Y. T. Wong, B. Pesaran, and S. M. M. (2021). Modeling behaviorally relevant neural dynamics enabled by preferential subspace identification. *Nature Neuroscience 4*, 140–149.

Saxena, S. and J. P. Cunningham (2019). Towards the neural population doctrine. *Current Opinion in Neurobiology 55*, 103–111.

Schiavon, L., A. Canale, and D. B. Dunson (2022). Generalized infinite factorization models. *Biometrika 109*(3), 817–835.

Schliep, E. M., D. Cooley, S. Sain, and J. Hoeting (2010). A comparison study of extreme precipitation from six different regional climate models via spatial hierarchical modeling. *Extremes 13*, 219–239.

Schott, J. R. (2007). A test for the equality of covariance matrices when the dimension is large relative to the sample sizes. *Computational Statistics and Data Analysis 51*, 6535–6542.

Schur, J. (1917). Über potenzreihen, die im innern des einheitskreises beschränkt sind. *Journal für die reine und angewandte Mathematik 147*, 205–232.

Shimazaki, H. and S. Shinomoto (2006). A recipe for optimizing a time-histogram. In *Advances in Neural Information Processing Systems*, pp. 1289–1296.

Shun, Z. and P. McCullagh (1995). Laplace approximation of high dimensional integrals. *Journal of the Royal Statistical Society. Series B (Statistical Methodology) 57*, 749–760.

Simpson, D., H. Rue, A. Riebler, T. G. Martins, and S. H. Sørbye (2017). Penalising model component complexity: A principled, practical approach to constructing priors. *Statistical Science 32*, 1–28.

Sinz, F. H., X. Pitkow, J. Reimer, M. Bethge, and A. S. Tolias (2019). Engineering a less artificial intelligence. *Neuron 103*, 967–979.

Skaug, H. J. and D. A. Fournier (2006). Automatic approximation of the marginal likelihood in non-Gaussian hierarchical models. *Computational Statistics & Data Analysis 51*(2), 699–709.

Smith, R. and J. Naylor (1987). A comparison of maximum likelihood and Bayesian estimators for the three parameter Weibull distribution. *Applied Statistics 36*, 358–369.

Smola, A. and B. Schölkopf (2000). Sparse greedy matrix approximation for machine learning. In *17th International Conference on Machine Learning*, pp. 911–918.

Snelson, E. and Z. Ghahramani (2006a). Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18*, pp. 1257–1264. MIT Press.

Snelson, E. and Z. Ghahramani (2006b). Variable noise and dimensionality reduction for sparse Gaussian processes. In *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*, pp. 461–468.

Spiegelhalter, D., A. Thomas, N. Best, and W. Gilks (1996). BUGS: Bayesian inference using Gibbs sampling. https://www.mrc-bsu.cam.ac.uk/software/bugs/.

Sporns, O. (2013). Structure and function of complex brain networks. *Dialogues in Clinical Neuroscience 15*, 247–262.

Sporns, O., G. Tononi, and R. Kötter (2005). The human connectome: A structural description of the human brain. *PLoS Computational Biology 1*, 245–251.

Srivastava, M. S. and H. Yanagihara (2010). Testing the equality of several covariance matrices with fewer observations than the dimension. *Journal of Multivariate Analysis 101*, 1319–1329.

Stan Development Team (2020). RStan: the R interface to Stan. Version 2.21.2. http://mc-stan.org/.

Steinmetz, N. A., C. Aydin, A. Lebedeva, M. Okun, M. Pachitariu, M. Bauza, M. Beau, J. Bhagat, C. Böhm, M. Broux, S. Chen, J. Colonell, R. J. Gardner, B. Karsh, D. Kostadinov, C. Mora-Lopez, J. Park, J. Putzeys, B. A. Sauerbrei, R. J. J. van Daal, A. Z. Vollan, M. Welkenhuysen, Z. Ye, J. T. Dudman, B. Dutta, A. W. Hantman, K. D. Harris, A. K. Lee, E. I. Moser, J. O'Keefe, A. Renart, K. Svoboda, M. Häusser, S. Haesler, M. Carandini, and T. D. Harris (2020). Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science 372*, eabf4588.

Stephenson, A. G., E. A. Lehmann, and A. Phatak (2016). A max-stable process model for rainfall extremes at different accumulation durations. *Weather and Climate Extremes 13*, 44–53.

Stevenson, I. H., J. M. Rebesco, L. E. Miller, and K. P. Kording (2008). Inferring functional connections between neurons. *Current Opinion in Neurobiology 18*, 582–588.

Stringer, A. (2021). Implementing approximate Bayesian inference using adaptive quadrature: the aghq package.

Sul, J. H., H. Kim, N. Huh, D. Lee, and M. W. Jung (2010). Distinct roles of rodent orbitofrontal and medial prefrontal cortex in decision making. *Neuron 66*, 449–460.

Sussillo, D., R. Jozefowicz, L. F. Abbott, and C. Pandarinath (2016). LFADS - latent factor analysis via dynamical systems.

Sutton, A. J. and K. R. Abrams (2001). Bayesian methods in meta-analysis and evidence synthesis. *Statistical Methods in Medical Research 10*, 277–303.

Tierney, L. and J. Kadane (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association 81*, 82–86.

Tierney, L., R. E. Kass, and J. B. Kadane (1989). Fully exponential Laplace approximations to expectations and variances of nonpositive functions. *Journal of the American Statistical Association 84*(407), 710–716.

Tierney, P. L., E. Dégenètais, A.-M. Thierry, J. Glowinski, and Y. Gioanni (2004). Influence of the hippocampus on interneurons of the rat prefrontal cortex. *European Journal of Neuroscience 20*(2), 514–524.

Tikidji-Hamburyan, R. A., V. K. Narayana, Z. Bozkus, and T. A. El-Ghazawi (2017). Software for brain network simulations: A comparative study. *Frontiers in Neuroinformatics 11*.

Titsias, M. K. and N. D. Lawrence (2010). Bayesian Gaussian process latent variable model. In *the 13th International Conference on Artificial Intelligence and Statistics*, Volume 9, pp. 844–851.

Trench, W. F. (1964). An algorithm for the inversion of finite Toeplitz matrices. *Journal of the Society for Industrial and Applied Mathematics 12*(3), 515–522.

Tresch, M. C., V. C. K. Cheung, and A. d'Avella (2006). Matrix factorization algorithms for the identification of muscle synergies: evaluation on simulated and experimental data sets. *Journal of neurophysiology 95*, 2199–2212.

Truccolo, W., U. T. Eden, M. R. Fellow, J. P. Donoghue, and E. N. Brown (2005). A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. *Journal of Neurophysiology 93*(2), 1074–1089.

Tweedie, M. C. K. (1957). Statistical properties of inverse Gaussian distributions I. *Annals of Mathematical Statistics 28*(3), 696–705.

Tye, M. R. and D. Cooley (2015). A spatial model to examine rainfall extremes in Colorado's front range. *Journal of Hydrology 530*, 15–23.

van Rossum, M. C. (2001). A noval spike distance. *Neural Computation 13*(4), 751–763.

Varin, C., N. Reid, and D. Firth (2011). An overview of composite likelihood methods. *Statistica Sinica 21*, 5–42.

Vats, D., J. M. Flegal, and G. L. Jones (2019). Multivariate output analysis for Markov chain Monte Carlo. *Biometrika 106*(2), 321–337.

Vehtari, A., A. Gelman, D. Simpson, B. Carpener, and P. Bükner (2021). Rank-normalization, folding, and localization: An improved $\widehat{R}$ for assessing convergence of MCMC. *Bayesian Analysis -1*, 1–28.

Ventura, V., C. Cai, and R. E. Kass (2005). Trial-to-trial variability and its effect on time-varying dependency between two neurons. *Journal of neurophysiology 94*(4), 2928–2939.

Wainwright, M. J. and M. I. Jordan (2008). Graphical models, exponential families, and variational inference. *Found. Trends Mach. Learn. 1*, 1–305.

Wei, G., I. H. Stevenson, and X. Wang (2022). Bayesian clustering of neural activity with a mixture of dynamic Poisson factor analyzers.

Williams, A. H., A. Degleris, Y. Wang, and S. W. Linderman (2020). Point process models for sequence detection in high-dimensional neural spike trains. In *Advances in Neural Information Processing Systems*.

Williams, C. and M. Seeger (2000). Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems*, Volume 13.

Williams, C. K. I. and D. Barber (1998). Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 20*(12), 1342–1351.

Wilson, A. G. and Z. Ghahramani (2011). Generalised wishart processes. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 736–744.

Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73*, 3–36.

Wood, S. N. (2017). *Generalized Additive Models: An Introduction with R* (2 ed.). Chapman and Hall/CRC.

Wood, S. N. (2020). Inference and computation with generalized additive models and their extensions. *Test 29*, 307–339.

Wood, S. N. (2022). *`mgcv`: Mixed GAM Computation Vehicle with Automatic Smoothness Estimation*.

Wood, S. N., N. Pya, and B. Säfken (2016). Smoothing parameter and model selection for general smooth models. *Journal of the American Statistical Association 111*, 1548–1563.

Wu, A., N. A. Roy, S. Keeley, and J. W. Pillow (2017). Gaussian process based nonlinear latent structure discovery in multivariate spike train data. In *Advances in Neural Information Processing Systems*, Volume 30.

Yang, G. R. and X.-J. Wang (2020). Artificial neural networks for neuroscientists: A primer. *Neuron 107*, 1048–1070.

Youngman, B. D. (2022). evgam: An R package for generalized additive extreme value models. *Journal of Statistical Software 103*, 1–26.

Yu, B. M., J. P. Cunningham, G. Santhanam, S. I. Ryu, K. V. Shenoy, and M. Sahani (2009). Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Journal of Neurophysiology 102*(1), 614–635.

Zhang, C., B. Shahbaba, and H. Zhao (2016). Variational Hamiltonian Monte Carlo via score matching. *Bayesian Analysis 13*, 485–506.

Zhao, Y. and I.-S. Park (2017). Variational latent gaussian process for recovering single-trial dynamics from population spike trains. *Neural Computation 29*, 1293–1316.

Zhou, B., D. E. Moorman, S. Behseta, H. Ombao, and B. Shahbaba (2016). A dynamic Bayesian model for characterizing cross-neuronal interactions during decision-making. *Journal of the American Statistical Association 111*(514), 459–471.

Zhou, F., Q. Kong, Z. Deng, J. Kan, Y. Zhang, C. Feng, and J. Zhu (2022). Efficient inference for dynamic flexible interactions of neural populations. *Journal of Machine Learning Research 23*(211), 1–49.

Zohar, S. (1969). Toeplitz matrix inversion: The algorithm of WF Trench. *Journal of the ACM (JACM) 16*(4), 592–601.

Zoltowski, D. M. and J. W. Pillow (2018). Scaling the Poisson GLM to massive neural datasets through polynomial approximations. *Advances in neural information processing systems 31*, 3517–3527.

APPENDICES

Appendix A

Appendix for Chapter 2

A.1 Estimation of Functions of Random Effects and Hyperparameters

Consider any non-linear vector-valued function $\boldsymbol{\eta}(\boldsymbol{u}, \boldsymbol{\theta}) : \mathbb{R}^{\dim(\boldsymbol{u})+\dim(\boldsymbol{\theta})} \to \mathbb{R}^m$. Once samples have been drawn from $\widetilde{p}(\boldsymbol{\theta} \mid \boldsymbol{y})$ and $\widetilde{p}(\boldsymbol{u} \mid \boldsymbol{y})$, which are computed using the method in Section 2.2, samples of $p(\boldsymbol{\eta}(\boldsymbol{u}, \boldsymbol{\theta}) \mid \boldsymbol{y})$ are obtained by taking non-linear transformations of samples $\boldsymbol{u}^s$ and $\boldsymbol{\theta}^s$. However, when the number of locations n is large, it is extremely costly to store in memory the covariance matrix of $\widetilde{p}(\boldsymbol{u} \mid \boldsymbol{y})$ whose dimension is a multiple of n, and to sample repeatedly from a multivariate normal distribution of the same dimension. Applying the Delta method,

$$\boldsymbol{\eta}(\boldsymbol{u}, \boldsymbol{\theta}) \approx \boldsymbol{\eta}(\boldsymbol{u}_{\widehat{\boldsymbol{\theta}}}, \widehat{\boldsymbol{\theta}}) + \boldsymbol{J}_{\boldsymbol{\eta}\circ\boldsymbol{u}} \cdot (\boldsymbol{\theta} - \widehat{\boldsymbol{\theta}}), \tag{A.1}$$

$$\widehat{\boldsymbol{V}}_{\boldsymbol{\eta}}(\boldsymbol{\theta}) \approx \boldsymbol{J}_{\boldsymbol{\eta}} \widehat{\boldsymbol{V}}_{\boldsymbol{u},\boldsymbol{\theta}}(\widehat{\boldsymbol{\theta}}) \boldsymbol{J}_{\boldsymbol{\eta}}^T, \tag{A.2}$$

where, with $\boldsymbol{u}_{\widehat{\boldsymbol{\theta}}} = \arg\min_{\boldsymbol{u}} G(\boldsymbol{u}; \widehat{\boldsymbol{\theta}})$ and $\widehat{\boldsymbol{V}}_{\boldsymbol{u}}(\widehat{\boldsymbol{\theta}}) = \left[\frac{\partial^2}{\partial \boldsymbol{u} \partial \boldsymbol{u}^T} G(\boldsymbol{u}_{\widehat{\boldsymbol{\theta}}}; \widehat{\boldsymbol{\theta}})\right]^{-1}$, we have

$$\boldsymbol{J}_{\boldsymbol{\eta}\circ\boldsymbol{u}} = \boldsymbol{J}_{\boldsymbol{\eta}} \cdot \begin{bmatrix} \boldsymbol{J}_u \\ \boldsymbol{I} \end{bmatrix}, \qquad \boldsymbol{J}_{\boldsymbol{u}} = \frac{\partial}{\partial \boldsymbol{\theta}} \boldsymbol{u}_{\boldsymbol{\theta}} \Big|_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}} = -\widehat{\boldsymbol{V}}_{\boldsymbol{u}}(\widehat{\boldsymbol{\theta}}) \left(\frac{\partial^2}{\partial \boldsymbol{u} \partial \boldsymbol{\theta}^T} G(\boldsymbol{u}_{\boldsymbol{\theta}}, \boldsymbol{\theta}) \Big|_{\boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}} \right)$$

$$\boldsymbol{J}_{\boldsymbol{\eta}} = \left[\tfrac{\partial}{\partial \boldsymbol{u}} \boldsymbol{\eta}(\boldsymbol{u}, \boldsymbol{\theta}) \quad \tfrac{\partial}{\partial \boldsymbol{\theta}} \boldsymbol{\eta}(\boldsymbol{u}, \boldsymbol{\theta}) \right] \Big|_{\boldsymbol{u}=\boldsymbol{u}_{\widehat{\boldsymbol{\theta}}}, \boldsymbol{\theta}=\widehat{\boldsymbol{\theta}}}, \qquad \widehat{\boldsymbol{V}}_{\boldsymbol{u},\boldsymbol{\theta}}(\widehat{\boldsymbol{\theta}}) = \begin{bmatrix} \widehat{\boldsymbol{V}}_{\boldsymbol{u}}(\widehat{\boldsymbol{\theta}}) + \boldsymbol{J}_{\boldsymbol{u}} \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \boldsymbol{J}_{\boldsymbol{u}}^T & \boldsymbol{J}_{\boldsymbol{u}} \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \\ \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \boldsymbol{J}_{\boldsymbol{u}}^T & \widehat{\boldsymbol{V}}_{\boldsymbol{\theta}} \end{bmatrix}. \tag{A.3}$$

In the special case of $\boldsymbol{\eta}(\boldsymbol{u}, \boldsymbol{\theta}) = \boldsymbol{u}$, (A.1) and (A.2) are, respectively, the mean and covariance matrix of the approximate marginal posterior of $\boldsymbol{u}$ in (2.20).

A.2 Estimation Errors for the Random Effects in the Small-Scale Simulation Study

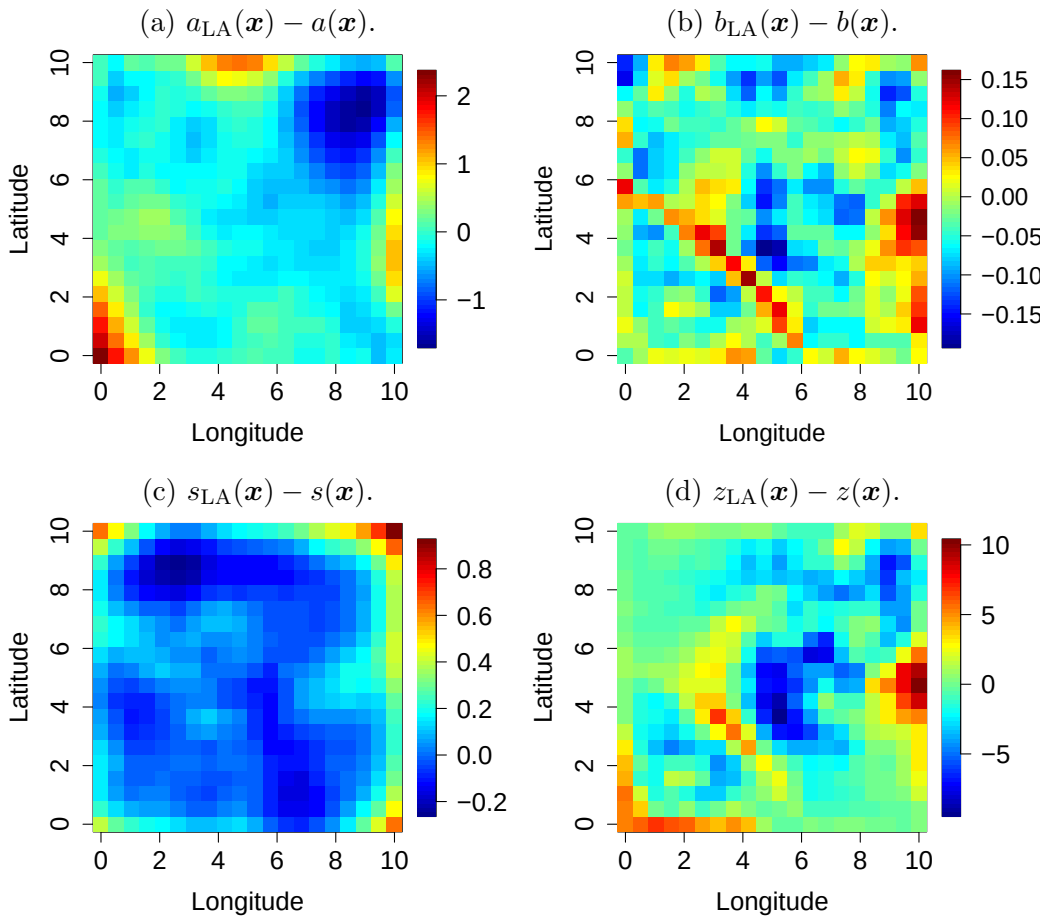


Figure A.1: Estimation errors for $a(\boldsymbol{x}_i)$, $b(\boldsymbol{x}_i)$, $s(\boldsymbol{x}_i)$ and $z_{10}(\boldsymbol{x}_i)$ in the small-scale simulation study plotted on regular lattices.

A.3 Bayesian Posterior Prediction at Arbitrary Locations

Let $y_\star$ denote the extreme value at a given spatial location $\boldsymbol{x}_\star$, and $\boldsymbol{u}_\star = \big(a(\boldsymbol{x}_\star), b(\boldsymbol{x}_\star), s(\boldsymbol{x}_\star)\big)$ denote the corresponding random effect parameters. Let us first assume that $\boldsymbol{x}_\star$ is a new location, i.e., not included in the locations $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^T$ used to fit the GEV-GP model.

Recall that $\boldsymbol{u} = (\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{s})$ is the vector of random effects at the observed locations and $\boldsymbol{\theta}$ is the vector of hyperparameters. The posterior predictive distribution at the new location is given by

$$
\begin{aligned}
p(y_\star \mid \boldsymbol{y}) &= \int p\big(y_\star \mid a(\boldsymbol{x}_\star), b(\boldsymbol{x}_\star), s(\boldsymbol{x}_\star)\big) p(\boldsymbol{u}_\star, \boldsymbol{\theta} \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{u}_\star \, \mathrm{d}\boldsymbol{\theta} \\
&= \int p(y_\star \mid a(\boldsymbol{x}_\star), b(\boldsymbol{x}_\star), s(\boldsymbol{x}_\star)) \left(\int p(\boldsymbol{u}_\star \mid \boldsymbol{u}, \boldsymbol{\theta}) p(\boldsymbol{u}, \boldsymbol{\theta} \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{u} \right) \mathrm{d}\boldsymbol{u}_\star \, \mathrm{d}\boldsymbol{\theta} \\
&= \int p(y_\star \mid a(\boldsymbol{x}_\star), b(\boldsymbol{x}_\star), s(\boldsymbol{x}_\star)) \times \\
&\qquad \left(\int p\big(a(\boldsymbol{x}_\star) \mid \boldsymbol{u}, \boldsymbol{\theta}\big) p\big(b(\boldsymbol{x}_\star) \mid \boldsymbol{u}, \boldsymbol{\theta}\big) p\big(s(\boldsymbol{x}_\star) \mid \boldsymbol{u}, \boldsymbol{\theta}\big) p(\boldsymbol{u}, \boldsymbol{\theta} \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{u} \right) \mathrm{d}\boldsymbol{u}_\star \, \mathrm{d}\boldsymbol{\theta} \\
&= \int p(y_\star \mid a(\boldsymbol{x}_\star), b(\boldsymbol{x}_\star), s(\boldsymbol{x}_\star)) \times \\
&\qquad \left(\int p\big(a(\boldsymbol{x}_\star) \mid \boldsymbol{a}, \boldsymbol{\theta}\big) p\big(b(\boldsymbol{x}_\star) \mid \boldsymbol{b}, \boldsymbol{\theta}\big) p\big(s(\boldsymbol{x}_\star) \mid \boldsymbol{u}, \boldsymbol{\theta}\big) p(\boldsymbol{u}, \boldsymbol{\theta} \mid \boldsymbol{y}) \, \mathrm{d}\boldsymbol{u} \right) \mathrm{d}\boldsymbol{u}_\star \, \mathrm{d}\boldsymbol{\theta}.
\end{aligned}
\tag{A.4}
$$

Note that assuming a Gaussian distribution on the location parameter $\boldsymbol{a}$, we have

$$
\begin{bmatrix} a(\boldsymbol{x}_\star) \\ \boldsymbol{a} \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} c(\boldsymbol{x}_\star)'\boldsymbol{\beta} \\ c(\boldsymbol{X})'\boldsymbol{\beta} \end{bmatrix}, \begin{bmatrix} \sigma_a^2 & \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}_\star, \boldsymbol{X}) \\ \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{X}, \boldsymbol{x}_\star) & \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{X}, \boldsymbol{X}) \end{bmatrix} \right). \tag{A.5}
$$

where $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^T$, $c(\boldsymbol{x}_\star)$ is the covariate vector at new location $\boldsymbol{x}_\star$, $c(\boldsymbol{X})$ is the design matrix of the covariates at observed locations, and $[\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{X}, \boldsymbol{X})]_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j' \mid \sigma^2, \kappa)$ using (2.3).

Hence, we can obtain $p(a(\boldsymbol{x}_\star) \mid \boldsymbol{u}, \boldsymbol{\theta}) = p(a(\boldsymbol{x}_\star) \mid \boldsymbol{a}, \boldsymbol{\theta})$ via

$$
a(\boldsymbol{x}_\star) \mid \boldsymbol{a}, \boldsymbol{\theta} \sim \mathcal{N}(m_{\text{new}}(\boldsymbol{x}_\star), \sigma_{\text{new}}(\boldsymbol{x}_\star)) \tag{A.6}
$$

where

$$m_{\text{new}}(\boldsymbol{x}_\star) = c(\boldsymbol{x}_\star)'\boldsymbol{\beta} + \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}_\star, \boldsymbol{X})\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{X}, \boldsymbol{X})^{-1}(\boldsymbol{a} - c(\boldsymbol{X})'\boldsymbol{\beta}), \tag{A.7}$$

$$\sigma_{\text{new}}(\boldsymbol{x}_\star) = \sigma_a^2 - \boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{x}_\star, \boldsymbol{X})\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{X}, \boldsymbol{X})^{-1}\boldsymbol{K}_{\boldsymbol{\theta}}(\boldsymbol{X}, \boldsymbol{x}_\star). \tag{A.8}$$

We can obtain $p(b(\boldsymbol{x}_\star) \mid \boldsymbol{u}, \boldsymbol{\theta})$ and $p(s(\boldsymbol{x}_\star) \mid \boldsymbol{u}, \boldsymbol{\theta})$ in a similar way as above.

The posterior predictive distribution $p(y_\star \mid \boldsymbol{y})$ can be approximated by approximating integrals via Riemann sums and performing a multi-stage sampling scheme as follows.

1. Sample $(\boldsymbol{a}^j, \boldsymbol{b}^j, \boldsymbol{s}^j, \boldsymbol{\theta}^j)$ from $p(\boldsymbol{u}, \boldsymbol{\theta} \mid \boldsymbol{y})$, $j = 1, 2, \ldots, m$.

2. For each $(\boldsymbol{a}^j, \boldsymbol{b}^j, \boldsymbol{s}^j, \boldsymbol{\theta}^j)$, sample $a^j(\boldsymbol{x}_\star)$, $b^j(\boldsymbol{x}_\star)$ and $s^j(\boldsymbol{x}_\star)$ from $p(a(\boldsymbol{x}_\star) \mid \boldsymbol{a}^j, \boldsymbol{\theta}^j)$, $p(b(\boldsymbol{x}_\star) \mid \boldsymbol{b}^j, \boldsymbol{\theta}^j)$ and $p(s(\boldsymbol{x}_\star) \mid \boldsymbol{s}^j, \boldsymbol{\theta}^j)$.

3. For each $\big(a^j(\boldsymbol{x}_\star), b^j(\boldsymbol{x}_\star), s^j(\boldsymbol{x}_\star)\big)$, sample $y_\star^j$ from $p(y_\star \mid a^j(\boldsymbol{x}_\star), b^j(\boldsymbol{x}_\star), s^j(\boldsymbol{x}_\star))$. We end up with m draws of $y_\star$.

4. The posterior mean of $y_\star$ can be approximated by $\frac{1}{m}\sum_{j=1}^m y_\star^j$, and any quantile of the posterior predictive distribution $p(y_\star \mid \boldsymbol{y})$ can be approximated by the corresponding sample quantile of $y_\star^1, \ldots, y_\star^m$.

Sampling from $p(y_\star \mid \boldsymbol{y})$ when $x_\star = \boldsymbol{x}_i \in \boldsymbol{X}$ is the ith observed location is similar but much easier, in that step 2 above simply consists of setting $a^j(\boldsymbol{x}_\star)$, $b^j(\boldsymbol{x}_\star)$, and $s^j(\boldsymbol{x}_\star)$ to the ith values of $\boldsymbol{a}^j$, $\boldsymbol{b}^j$, and $\boldsymbol{s}^j$. In either case, sampling from the posterior predictive distribution of any test statistic $p(T(y_\star) \mid \boldsymbol{y})$ follows immediately by setting $T^j = T(y_\star^j)$.

Appendix B

Appendix for Chapter 3

B.1 Comparison Between Two-Step Estimation and Joint Estimation

This section compares the results from the two-step estimation and the joint estimation described in Section 3.3.4 using the simulated data from Section 3.5.1. Figure B.1 examines the point estimates (posterior means) of the parameters of interest, which include the log drift $\log(\alpha_i)$, the log threshold $\log(k_i)$, the firing rate α_i/k_i, and the unconstrained loadings $\ell^\star_{ij},\ i = 1, \ldots, q,\ j = 1, \ldots, d$. Despite that both $\log(\alpha_i)$ and $\log(k_i)$ are estimated to be smaller in the two-step estimation compared to the joint estimation (Figure B.1 (a) and (b)), the rate estimates, which are of main interest in practice, are approximately the same using both procedures (Figure B.1 (c)). The differences in $\widehat{\log(\alpha_i)}$ and $\widehat{\log(k_i)}$ between two-step and joint are likely due to approximating the step function using the sigmoid function in the joint estimation. Figure B.1 (d) shows that some estimates of $\ell^\star_{ij}$ from the two-step estimation are flipped in sign compared to the estimates from the joint estimation, despite their absolute values being very close. This is expected as the columns of the loading matrix can only be estimated up to flipping signs. Hence, it can be concluded that both the two-step procedure and the joint procedure produce similar point estimates for this simulated data set.

Next, we compare the posterior standard deviations of $\hat{\ell}^\star_{ij},\ i = 1, \ldots, q,\ j = 1, \ldots, d$, obtained from the two-step estimation and the joint estimation, respectively. The left panel of Figure B.2 shows that some estimates of $\ell^\star_{ij}$ have large posterior standard deviations using both estimating procedures. A closer examination of the estimated loading matrices

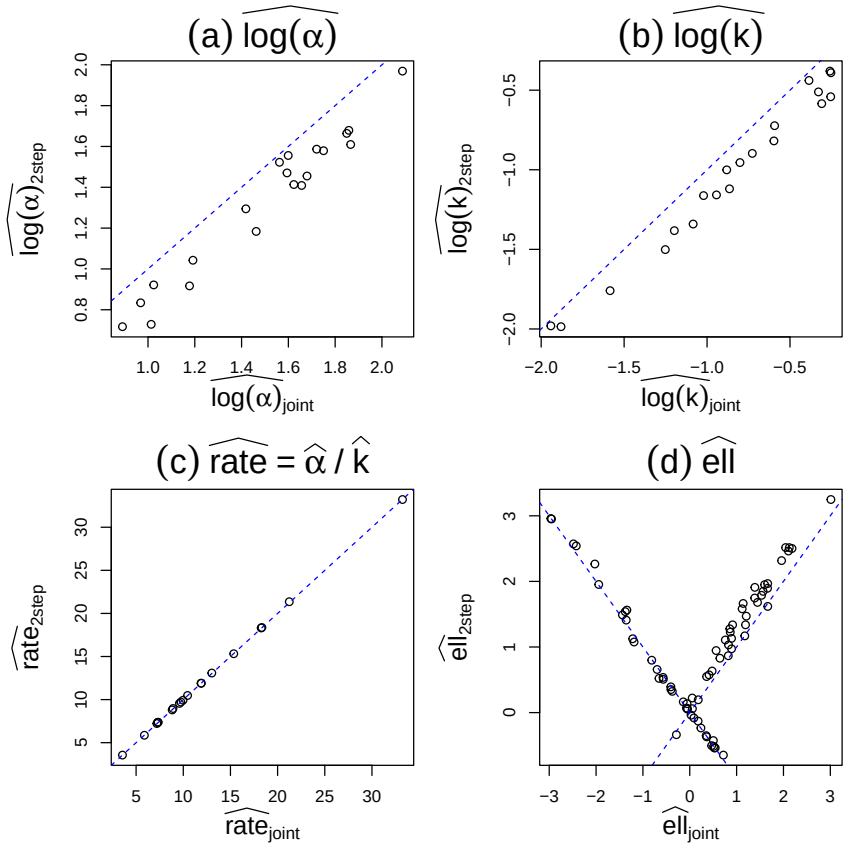


Figure B.1: Point estimates of parameters of interest from two-step estimation plotted against the point estimates from joint estimation. The blue dashed lines correspond to the 45 (and the -45) degree lines.

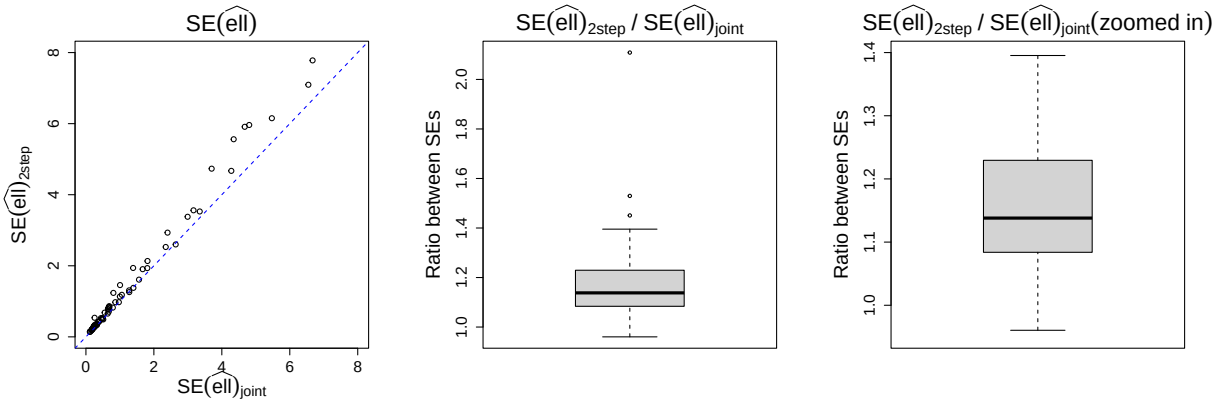


Figure B.2: Left: Posterior standard deviation of $\hat{\ell}_{ij}^{\star}$ from two-step estimation plotted against the those from joint estimation. The dashed line corresponds to the 45 degree line. Middle: Box plot of the ratio between the posterior standard deviations from two-step estimation and joint estimation, respectively. Right: A zoomed-in version of the box plot from the middle panel.

show that the large SEs occur for loadings that are smaller in values when compared to other loadings on the same row. It is possible that the approximate marginal likelihood $\widetilde{\mathcal{L}}(\boldsymbol{\theta}_2 \mid \widehat{\boldsymbol{\theta}}_1, \boldsymbol{Y})$ has very flat curvature at the corresponding dimensions of the parameter space. Further investigation is required to understand how the large SEs arise. The middle panel of Figure B.2 shows a box plot of the ratio between the two SEs from the two-step procedure and the joint procedure, respectively, for each $\hat{\ell}_{ij}^{\star}$. For better visibility, plotted in the right panel of Figure B.2 is a zoomed-in version of the box plot focusing on the range between 0.9 and 1.4. It is found that the majority of the ratio between the two paired SEs are close to 1, with a median of 1.14. This suggests that two-step estimation results in systematically slightly higher SEs for $\hat{\ell}_{ij}^{\star}$. This corresponds to more conservative uncertainty quantification. It is of interest to repeat this simulation study for different sets of simulated data with varying duration of experimental time, number of neurons, and number of factors.

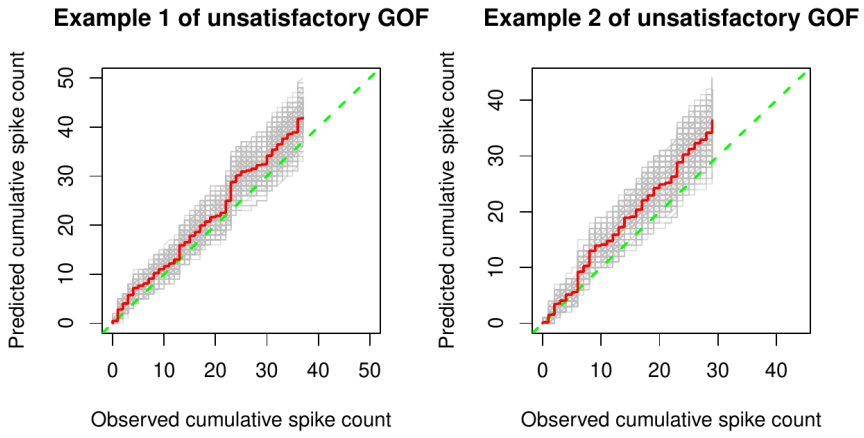


Figure B.3: Examples of unsatisfactory LNO-PP test results obtained by plotting the predicted cumulative spike counts from a trial i against the observed cumulative spike count from trial j for $j \neq i$.

B.2 Leave-Neuron-Out Posterior Predictive Test

B.2.1 Example of LNO-PP Diagnostic for Unsatisfactory Model fit

Figure B.3 demonstrates two examples of unsatisfactory agreement between model prediction and observed data using prediction results from Algorithm 2. The test statistic is the graphical line plotting the predicted cumulative spike counts for the left-out neuron against its observed cumulative spike counts, as described in Section 3.4. To generate these "bad" examples, we simulated multiple repeated trials for 12 neurons. We plot the predicted cumulative spike count for a left-out neuron in trial i against the observed cumulative spike count for the same neuron in a different trial j, $j \neq i$. Since the different trials were simulated using the same model parameters, their respective firing rates are expected to be similar. However, the exact spike times of the held-in neurons in trial i do not predict the spike times of the left-out neuron in j. Therefore, the samples of test statistic (grey lines) are expected to deviate from the 45-degree line.

B.2.2 Model Comparison using LNO-PP Test

The LNO-PP test can also be used for comparing different latent variable spike-train models given that spike times can be simulated from the model. We adopt this procedure to compare the prediction performance of our model to the Gaussian Process Factor Analysis (GPFA) model (Yu et al., 2009), another popular latent variable spike-train model. The GPFA model parameters are estimated using the MATLAB code provided at `https://users.ece.cmu.edu/~byronyu/software.shtml`. We refer to Yu et al. (2009) for details of the GPFA model. Since the GPFA method works on binned spike counts, we use the following procedure to simulate posterior predictive spike times from a fitted GPFA model. Suppose neuron $i = 1$ is the left-out neuron for leave-neuron-out prediction.

1. Keeping neuron i in, estimate the GPFA model parameters: loading matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{q \times d}$, the bias term $\boldsymbol{d}$, and the observation noise matrix $\boldsymbol{R}$.

2. Leaving neuron i out, estimate the multivariate latent trajectory $\widehat{\boldsymbol{X}} \in \mathbb{R}^{d \times n}$.

3. Simulate square-rooted spike count for neuron i in bin j from the leave-neuron-out posterior predictive distribution

$$z_{i,j} \sim N(\widehat{\boldsymbol{\Lambda}}_{i,:}\widehat{\boldsymbol{X}}_{:,j} + \widehat{\boldsymbol{d}}_i,\ \widehat{\boldsymbol{R}}_{i,i}),$$

   where $\widehat{\boldsymbol{\Lambda}}_{i,:}$ is the i-th row of the estimated loading matrix, $\widehat{\boldsymbol{X}}_{:,j} \in \mathbb{R}^d$ is the j-th column of the estimated multivariate latent trajectory, $\widehat{\boldsymbol{d}}_i$ is the i-th element of the estimated bias term, and $\widehat{\boldsymbol{R}}_{i,i}$ is the i-th diagonal element of the estimated $d \times d$ noise matrix.

4. Assuming a constant firing rate of $z_{i,j}^2$ throughout bin j, we simulate spike times from a Poisson process with rate $z_{i,j}^2$.

5. Repeating steps 3 and 4 for all bins gives the posterior predictive spike times for the left-out neuron using the GPFA method.

We have simulated 10 seconds of spike trains for 14 neurons sampled from the artificial neuronal network described in Section 3.5.2 using the NEURON software. The total spike count for each neuron is 86, 95, 235, 61, 270, 113, 72, 59, 52, 198, 222, 230, 172, 209, respectively. 6 factors were used for model fitting for both GPFA and our model. We perform the graphical LNO-PP tests with neuron 3 left out using the cumulative spike count described in Section 3.4 as the test statistic. 100 samples are drawn from the posterior

predictive distribution for each method. For GPFA, we use the aforementioned procedure with different bin lengths: 20ms, 50ms,75ms, 100ms. For our model, we employ the sLNO-PP test in Algorithm 2. The results are shown in Figure B.4. Compared to our model, the leave-neuron-out prediction of GPFA produce more biased and uncertain results with all bin lengths. This is suggested by the misalignment between the red and green lines, as well as the wider (compared to Figure B.4(a)-(d)) bandwidth of the region occupied by the grey lines in Figures B.4(e)-(p). Nonetheless, we note that the original GPFA framework described in Yu et al. (2009) generates binned spike counts instead of spike times. Thus, predicting spike times is not what GPFA is designed for, and its performance can be undermined by the spike time generating mechanism we propose above.

B.3 Exploratory Factor Analysis for Choosing the Number of Factors

To choose the number of factors d in the model, we employ an exploratory factor analysis (EFA) procedure described as follows.

1. For each neuron $i = 1, \ldots, q$,
   - 1.1 fit a single-neuron model (3.1) to the spike train $\boldsymbol{y}_i \in \mathbb{R}^n$ and obtain the corresponding latent path estimate $\widehat{\boldsymbol{x}}_i \in \mathbb{R}^n$, and
   - 1.2 compute the latent path increment estimates $d\widehat{\boldsymbol{x}}_i = (\hat{x}_{i,1} - \hat{x}_{i,0}, \ldots, \hat{x}_{i,n} - \hat{x}_{i,n-1})$.
2. Collect the latent path increment estimates for all q neurons in $\widehat{\boldsymbol{X}}_{\text{EFA}} = (d\widehat{\boldsymbol{x}}_1, \ldots, d\widehat{\boldsymbol{x}}_q)^T$. Treat it as an observed data matrix.
3. For each candidate value $d \in \{d_1, \ldots, d_m\}$,
   - 3.1 fit a factor model on $\widehat{\boldsymbol{X}}_{\text{EFA}}$ with d factors, and
   - 3.2 obtain the estimated loading matrix $\widehat{\boldsymbol{\Lambda}}_d$, and
   - 3.3 compute the cumulative proportion of variance explained by all d factors, denoted by $cumvar(d)$.

Based on the resulted $\widehat{\boldsymbol{\Lambda}}_d$ and $cumvar(d)$ for $d \in \{d_1, \ldots, d_m\}$, two methods for choosing d are introduced. One way to check that d factors are sufficient for modelling purpose is to

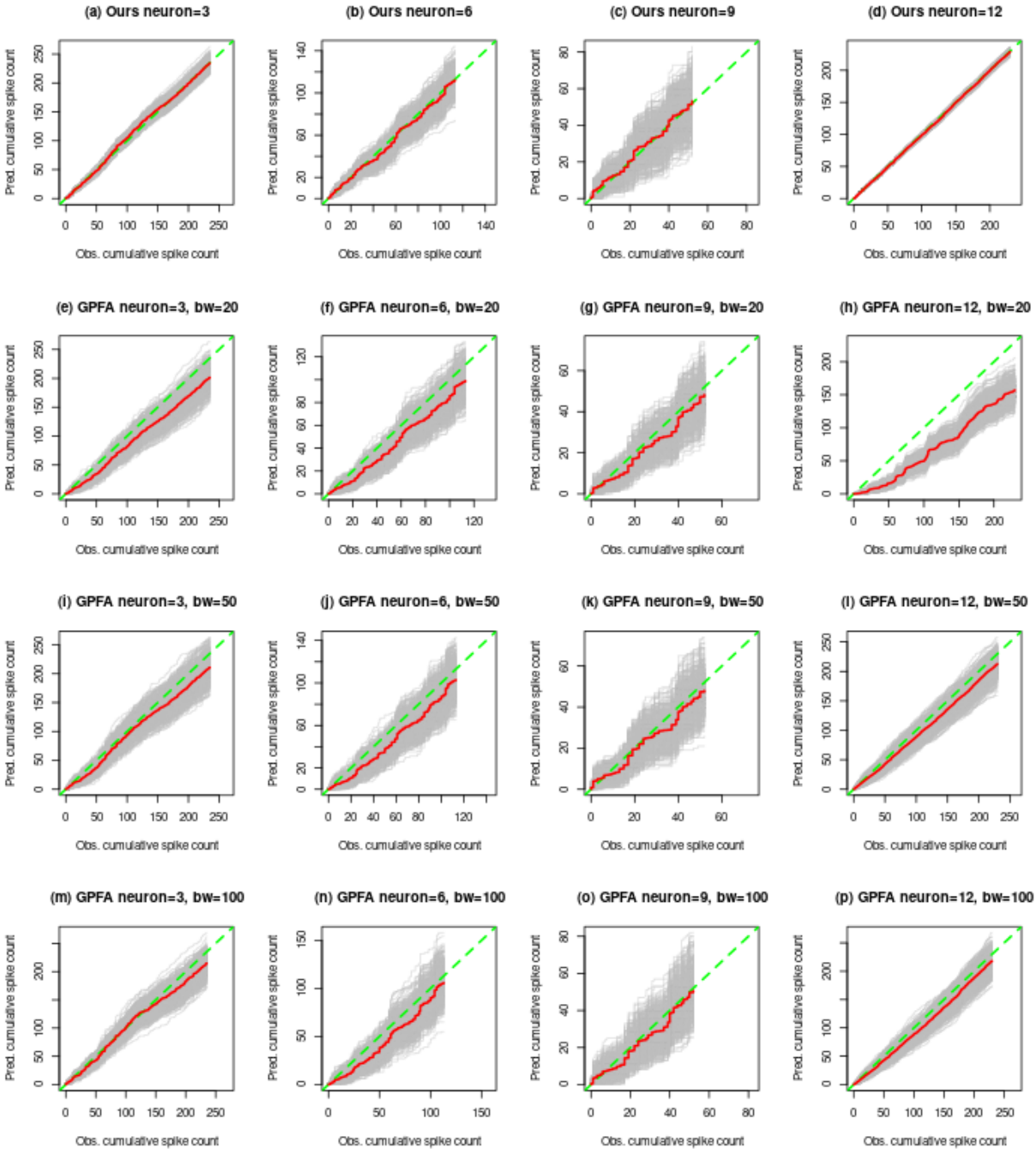


Figure B.4: LNO-PP test results for our model and for GPFA with different bin lengths. Each grey line, representing one sample from the posterior predictive distribution of spike times, plots the predicted cumulative spike counts on the Y-axis against the observed for the corresponding left-out neuron on the X-axis. The red line represents the average of all grey lines in the plot. The green dashed line corresponds to the 45-degree diagonal reference line.

examine how much variance in $\widehat{\boldsymbol{X}}_{\text{EFA}}$ is explained by the d factors. We refer to this method as the cumulative variance method. To this end, one can plot $cumvar(d)$ as a function of d, and look for the "elbow". The elbow occurs when adding one more factor to the model does not lead to much increase in variance explained. When the purpose of using a factor model is dimension reduction, the elbow is a helpful indicator for when to stop increasing the latent dimension. Using the simulated spike trains from 20 neurons in Section 3.5.1, we plot $cumvar(d)$ versus d as well as $cumvar(d+1) - cumvar(d)$ versus d in the left and middle panel of Figure B.5. We see that for $d > 4$, a unit increase in d only leads to an increase in variance proportion explained around 2%. It is clear that the elbow occurs at $d = 4$.

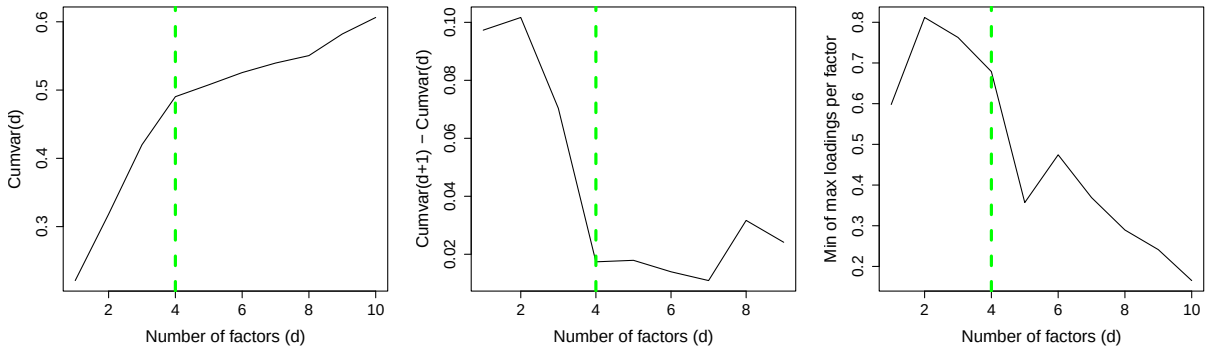


Figure B.5: EFA results for simulated data from Section 3.5.1. Left: Cumulative proportion of variance explained by the d factors $cumvar(d)$ for different choices of d. Middle: Increase in the cumulative proportion of variance explained, $cumvar(d+1) - cumvar(d)$, for different choices of d. Right: The minimum of the max loading in each column of $\widehat{\boldsymbol{\Lambda}}_d$ for different values of d. The green dashed line corresponds to the chosen d.

Alternatively, one can choose d based on the loadings. We refer to this method as the minimax loading method. Intuitively, when a factor can explain some variance in at least one dimension of the observed data, the max loading on that factor should be moderate to large. On the flip side, if the factor model uses more factors than the true number of factors, the extra factors would have relatively small loadings on them. For each estimated $\widehat{\boldsymbol{\Lambda}}_d$, we can collect the max loading on each factor, denoted by $\ell_1^{\max}, \ldots, \ell_d^{\max}$, respectively. If the minimum of them, $\min(\ell_1^{\max}, \ldots, \ell_d^{\max})$, is large, this could mean that all d factors contribute greatly to explaining the commonality in the observed data; if the minimum is small, some factors might be viewed as extra. Using the simulated data from Section 3.5.1,

we plot $\min(\ell_1^{\max}, \ldots, \ell_d^{\max})$ for different choices of d in the right panel of Figure B.5. It is found that for $d > 4$, the min of max loading drops to below 0.5. Both the cumulative variance method and the minimax loading method, though arguably subjective, provide some evidence for choosing $d = 4$.

B.3.1 Analysis of Data Simulated From NEURON

We have applied both EFA methods on the data simulated from the NEURON software in Section 3.5.2. The plots of $cumvar(d)$ and minimax loadings versus d are given in Figure B.6. The left panel of Figure B.6 shows that $cumvar(d)$ increases almost linearly as d grows. This might suggest that the estimated latent paths from the single-neuron models exhibit more idiosyncrasy than commonality, and that each additional factor helps explain more idiosyncratic variance. Inspecting the middle panel of Figure B.6, we see that, for $d > 2$, the increase in $cumvar(d)$ for each unit increase of d fluctuates between 1% to 2.3%. Thus, adding more factors does not lead to a notable increase in $cumvar(d)$. As for the right panel of Figure B.6, no meaningful conclusion can be drawn from it about the choice of d. Judging from the information provided by the three plots, we decided to choose $d = 6$, on the grounds that not much gain is made in terms of $cumvar(d)$ or minimax loading for $d > 3$.

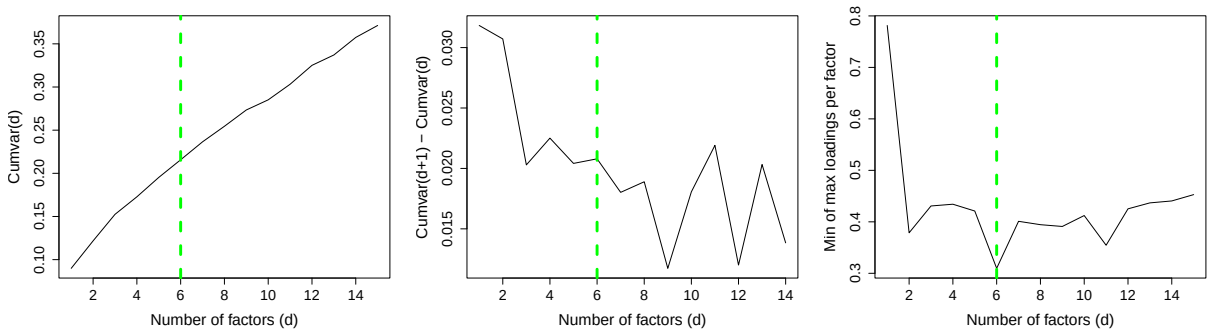


Figure B.6: EFA results for NEURON simulated data from Section 3.5.2. Left: Cumulative proportion of variance explained by the d factors $cumvar(d)$ for different values of d. Middle: Increase in the cumulative proportion of variance explained, $cumvar(d+1) - cumvar(d)$, for different values of d. Right: The minimum of the max loading in each column of $\widehat{\mathbf{\Lambda}}_d$ for different values of d. The green dashed line corresponds to the chosen d.

B.3.2 Analysis of Real Data of Rats

Note that the real data analysis in Chapter 3.6 involves fitting 2 models, one pre-cue and one post-cue, for each of the 3 experimental conditions. Therefore, we perform the EFA procedure 6 times. Since none of the plots for $cumvar(d)$ has an elbow shape, we only look at the minimax loadings for different choices of d for this analysis. The plots in Figure B.7 show that, across for all three paradigms and for both pre-cue and post-cue, the minimax loading decreases as the number of factors increases, and plateaus after $d = 6$. Therefore, we decide to choose $d = 6$.

B.4 Sensitivity Analysis and Runtimes for Different Numbers of Factors

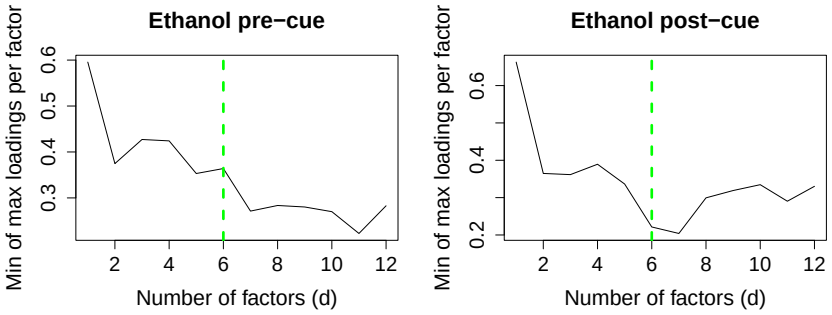
Figure B.9 shows the estimated loading matrix $\widehat{\mathbf{\Lambda}}$ after Varimax rotation for $d = 2, 4, 6, 8, 10$ for the simulation study using data simulated from the proposed model in Section 3.5.1. It is found that, for $d > 4$, $\widehat{\mathbf{\Lambda}}$ only has four factors with large loadings and the rest can be viewed as extra. Plotted in Figure B.8 are the runtimes (in minutes) for fitting the factor model with different choices of d using `R` 3.6.2 on a computer cluster with three 2.7GHz Xeon(R) CPUs (E5-4650) and 20GB memory allocated to each CPU.

B.5 Summary Statistics for Comparing Neuronal Interactions Before and After Cue

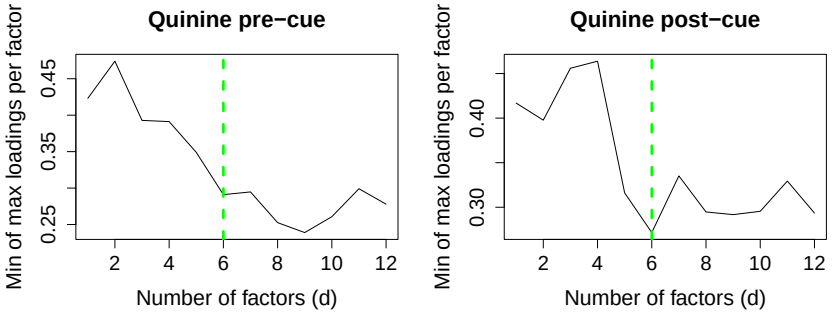
For summarizing the change in functional connectivity between neurons before and after the cue in the real data analysis, we have considered the following statistics.

1. **Smallest angle between principal components.** We first calculated the first K principal components $PC_1^{\text{bl}}, \ldots, PC_K^{\text{bl}}$ of the baseline estimate $\widehat{\mathbf{\Sigma}}^{\text{bl}}$ and those for the post-cue estimate $PC_1^{\text{post}}, \ldots, PC_K^{\text{post}}$. Then we calculate the angle between each pair of $(PC_i^{\text{bl}}, PC_j^{\text{post}})$ for $i, j = 1, \ldots, K$. The summary statistic is taken to be the smallest angle, which is viewed as representing the max similarity between $\widehat{\mathbf{\Sigma}}^{\text{bl}}$ and $\widehat{\mathbf{\Sigma}}^{\text{post}}$ in some low-dimensional latent space.

(a) Ethanol paradigm



(b) Quinine paradigm



(c) Sucrose paradigm

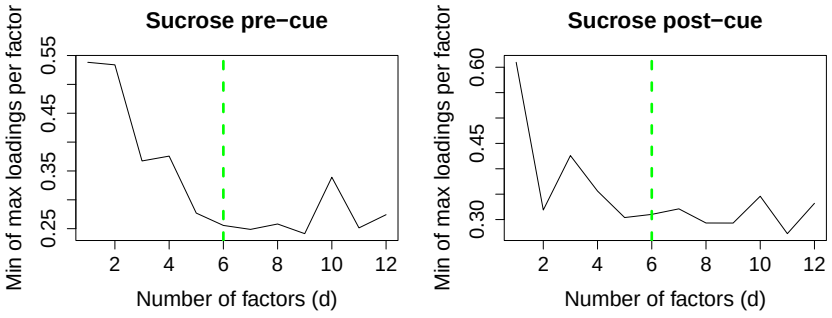


Figure B.7: EFA results for real data of rats from Chapter 3.6. Plotted on the y-axis is the minimum of the max loading in each column of $\widehat{\mathbf{\Lambda}}_d$. The x-axis is different choices of d. The green dashed line corresponds to the chosen d. (a)-(c) correspond to ethanol, quinine, and sucrose experimental paradigms, respectively.

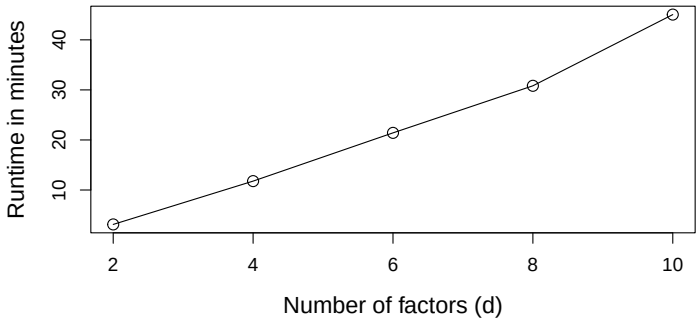


Figure B.8: Runtimes for different choices of d in the simulation study using data simulated from the model.

2. **Maximum correlation between principal components.** Similar to 1, but replacing the pairwise angle by pairwise correlation. The summary statistic is then taken to be the maximum correlation.

3. **Canonical cosine similarity between principal components.** We calculate the first K principal components for $\widehat{\boldsymbol{\Sigma}}^{\text{bl}}$ and $\widehat{\boldsymbol{\Sigma}}^{\text{post}}$ similar to 1. Then we arrange the principal components into $q \times K$ matrices $\widehat{\boldsymbol{P}}^{\text{bl}}$ and $\widehat{\boldsymbol{P}}^{\text{post}}$, whose k-th column corresponds to the k-th principal component. Find length K vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ such that the cosine similarity $\frac{(\widehat{\boldsymbol{P}}^{\text{bl}}\boldsymbol{a})\cdot(\widehat{\boldsymbol{P}}^{\text{post}}\boldsymbol{b})}{\|\widehat{\boldsymbol{P}}^{\text{bl}}\boldsymbol{a}\|\cdot\|\widehat{\boldsymbol{P}}^{\text{post}}\boldsymbol{b}\|}$ is maximized. Then the max cosine similarity, canonical cosine similarity, is taken as the summary statistic.

4. **Frobenius form of matrix difference.** Consider a different test statistic for testing the equality of covariance matrices based on the Frobenius matrix norm: $\text{tr}\big((\boldsymbol{\Sigma}_1 - \boldsymbol{\Sigma}_2)(\boldsymbol{\Sigma}_1 - \boldsymbol{\Sigma}_2)^{\text{T}}\big)$ (Schott, 2007; Li and Chen, 2012).

None of the aforementioned statistics indicates which direction the overall functional connectivity shifts to as they are all positive-valued. That is, we cannot know whether the post-cue functional connectivity is overall higher or lower than the baseline connectivity. On the other hand, our adopted statistic Δ informs this change in direction and is centred around zero if the shift is small. That said, in the case where the functional connectivity shift from highly positively correlated to highly negative correlated, Δ might be not informative of this drastic change. However, highly negatively correlated $\widehat{\boldsymbol{\Sigma}}$ is uncommon.

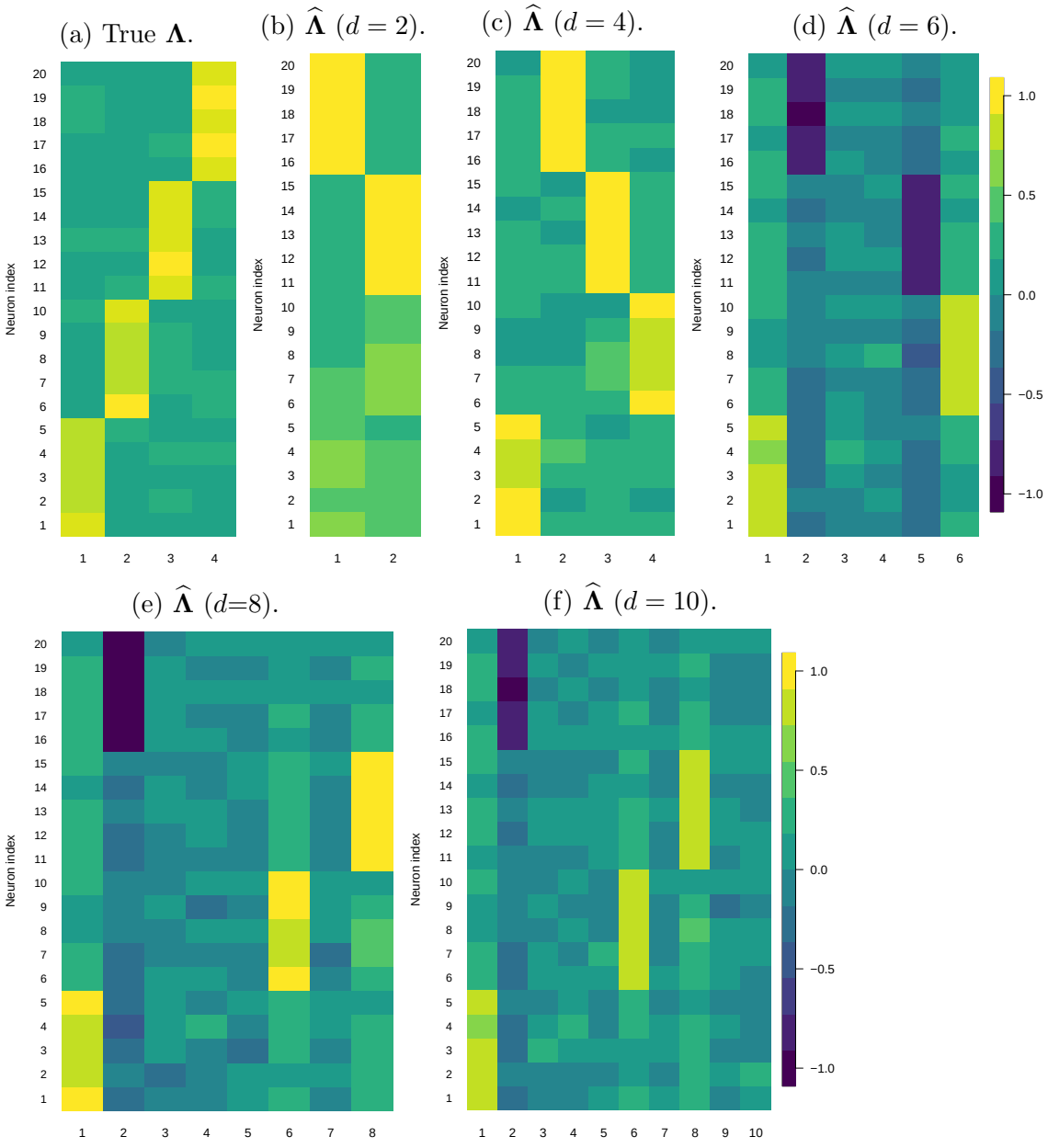


Figure B.9: Varimax-transformed estimated factor loading matrix $\widehat{\mathbf{\Lambda}}$ for different choices of d.

Appendix C

Appendix for Chapter 4

C.1 Derivation of the Latent States Update for the GPFA Gibbs Sampler

Here we show how to efficiently evaluate $\boldsymbol{\Omega}_k^{-1} = \boldsymbol{\Sigma} \otimes \mathrm{I}_N + \boldsymbol{\beta}_{k,:}^T \boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}$ in the conditional update $\boldsymbol{x}_{:,k} \mid \boldsymbol{Y}_k^\star$ in (4.15). Note that $\boldsymbol{\Omega}_k$ can be written as

$$\boldsymbol{\Omega}_k = \boldsymbol{\Sigma} \otimes \mathrm{I}_N + (\boldsymbol{\beta}_{k,:}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \mathrm{I}_N (\boldsymbol{\beta}_{k,:} \otimes \mathrm{I}_N).$$

Using the Woodbury identity, its inverse is given by

$$\begin{aligned}
\boldsymbol{\Omega}_k^{-1} &= \boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N - \\
&\quad (\boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N)(\boldsymbol{\beta}_{:,k}^T \otimes \boldsymbol{V}_k^{(k)}) \big[\mathrm{I}_N + (\boldsymbol{\beta}_{:,k} \otimes \mathrm{I}_N)(\boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N)(\boldsymbol{\beta}_{:,k}^T \otimes \boldsymbol{V}_k^{(k)})\big]^{-1} \cdot \\
&\quad (\boldsymbol{\beta}_{:,k} \otimes \mathrm{I}_N)(\boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N) \\
&= \boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N - (\boldsymbol{\Sigma}^{-1} \boldsymbol{\beta}_{:,k}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \big[\mathrm{I}_N + (\boldsymbol{\beta}_{:,k} \boldsymbol{\Sigma}^{-1} \boldsymbol{\beta}_{:,k}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\big]^{-1} (\boldsymbol{\beta}_{:,k} \boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N) \\
&= \boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N - (\boldsymbol{\Sigma}^{-1} \boldsymbol{\beta}_{:,k}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \big(1 \otimes \big[\mathrm{I}_N + (\boldsymbol{\beta}_{:,k} \boldsymbol{\Sigma}^{-1} \boldsymbol{\beta}_{:,k}^T \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\big]^{-1}\big) (\boldsymbol{\beta}_{:,k} \boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N) \\
&= \boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N - (\boldsymbol{\Sigma}^{-1} \boldsymbol{\beta}_{:,k}^T \boldsymbol{\beta}_{:,k} \boldsymbol{\Sigma}^{-1}) \otimes (\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} \boldsymbol{Q}^{-1})
\end{aligned} \tag{C.1}$$

where $\boldsymbol{Q} = [\mathrm{I}_N + c \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}]^{-1}$ and $c = \boldsymbol{\beta}_{:,k} \boldsymbol{\Sigma}^{-1} \boldsymbol{\beta}_{:,k}^T$ is a scalar.

Plugging (C.1) into the conditional update (4.15), the mean of this Normal distribution

is expressed as

$$
\begin{aligned}
&(\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\boldsymbol{\Omega}_k^{-1} \operatorname{vec}(\boldsymbol{Y}_k^{\star}) \\
&= (\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\Big(\boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N - (\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{:,k}^T\boldsymbol{\beta}_{:,k}\boldsymbol{\Sigma}^{-1}) \otimes (\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1})\Big) \operatorname{vec}(\boldsymbol{Y}_k^{\star}) \\
&= \big((\boldsymbol{\beta}_{k,:}\boldsymbol{\Sigma}^{-1}) \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\big) \operatorname{vec}(\boldsymbol{Y}^{\star}) - \big((c\boldsymbol{\beta}_{k,:}\boldsymbol{\Sigma}^{-1}) \otimes (\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1})\big) \operatorname{vec}(\boldsymbol{Y}_k^{\star}) \\
&= \boldsymbol{V}_k^{(k)}\boldsymbol{Y}_k^{\star}\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}^T - c\boldsymbol{V}_k^{(k)}\boldsymbol{V}_k^{(k)}\boldsymbol{Q}^{-1}\boldsymbol{Y}_k^{\star}\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}^T \\
&= \boldsymbol{V}_k^{(k)}(\mathrm{I}_N - c\boldsymbol{V}_k^{(k)}\boldsymbol{Q}^{-1})\boldsymbol{Y}_k^{\star}\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{k,:}^T \\
&= \boldsymbol{V}_k^{(k)}\boldsymbol{Q}^{-1}\boldsymbol{Y}_k^{\star}\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{k,:}^T
\end{aligned}
$$

and the covariance matrix is

$$
\begin{aligned}
&\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} - (\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\boldsymbol{\Omega}_k^{-1}(\boldsymbol{\beta}_{k,:}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \\
&= \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} - (\boldsymbol{\beta}_{k,:} \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)})\cdot \\
&\qquad\qquad \Big(\boldsymbol{\Sigma}^{-1} \otimes \mathrm{I}_N - (\boldsymbol{\Sigma}^{-1}\boldsymbol{\beta}_{:,k}^T\boldsymbol{\beta}_{:,k}\boldsymbol{\Sigma}^{-1}) \otimes (\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1})\Big)(\boldsymbol{\beta}_{k,:}^T \otimes \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \\
&= \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} - \Big(c\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} - c^2\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1}\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\Big) \\
&= \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} - c\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}(\mathrm{I}_N - c\boldsymbol{Q}^{-1}\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}) \\
&= \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)} - c\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1} \\
&= \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}(\mathrm{I}_N - c\boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1}) \\
&= \boldsymbol{V}_{\boldsymbol{\theta}}^{(k)}\boldsymbol{Q}^{-1}
\end{aligned}
$$

C.2 Metropolis-Within-Gibbs Sampler for the Extended GPFA Model with Poisson Likelihood

Here we present Algorithm 3 for sampling from the Poisson-GPFA model using a Metropolis-Hastings-Within-Gibbs block updating scheme.

Algorithm 3 Superfast Metropolis-Hastings-Within-Gibbs Sampler

Input: Initial states of the parameters $\boldsymbol{X}^0$, $\boldsymbol{\beta}^0$, $\boldsymbol{\alpha}^0$, $\boldsymbol{\theta}^0$, number of iterations M
Output: Samples $\{\boldsymbol{X}\}_{m=1}^M$, $\{\boldsymbol{\beta}\}_{m=1}^M$, $\{\boldsymbol{\alpha}\}_{m=1}^M$, $\{\boldsymbol{\theta}\}_{m=1}^M$

for $m = 1, \ldots, M$: **do**

for $k = 1, \ldots, K$: **do**
Propose $\boldsymbol{x}_{:,k}^\star \sim \mathcal{N}(\boldsymbol{x}_{:,k}^{m-1}, \boldsymbol{\Sigma}_{\boldsymbol{x}})$. Let $\boldsymbol{x}_{:,k}^m = \boldsymbol{x}_{:,k}^\star$ with probability

$$\min\left\{1, \frac{p(\boldsymbol{Y} \mid \boldsymbol{x}_{:,-k}^{m-1}, \boldsymbol{x}_{:,k}^\star, \boldsymbol{\beta}^{m-1}, \boldsymbol{\alpha}^{m-1}, \boldsymbol{\theta}^{m-1}) p(\boldsymbol{x}_{:,k}^\star \mid \boldsymbol{\theta}^{m-1})}{p(\boldsymbol{Y} \mid \boldsymbol{x}_{:,-k}^{m-1}, \boldsymbol{x}_{:,k}^{m-1}, \boldsymbol{\beta}^{m-1}, \boldsymbol{\alpha}^{m-1}, \boldsymbol{\theta}^{m-1}) p(\boldsymbol{x}_{:,k}^{m-1} \mid \boldsymbol{\theta}^{m-1})}\right\}.$$

end for

for $d = 1, \ldots, D$: **do**
Propose $\boldsymbol{\beta}_{:,d}^\star \sim \mathcal{N}(\boldsymbol{\beta}_{:,d}^{m-1}, \boldsymbol{\Sigma}_{\boldsymbol{\beta}})$. Let $\boldsymbol{\beta}_{:,d}^m = \boldsymbol{\beta}_{:,d}^\star$ with probability

$$\min\left\{1, \frac{p(\boldsymbol{y}_d \mid \boldsymbol{X}^m, \boldsymbol{\beta}_d^\star, \boldsymbol{\alpha}^{m-1}, \boldsymbol{\theta}^{m-1}) \pi(\boldsymbol{\beta}_d^\star)}{p(\boldsymbol{y}_d \mid \boldsymbol{X}^m, \boldsymbol{\beta}_d^{m-1}, \boldsymbol{\alpha}^{m-1}, \boldsymbol{\theta}^{m-1}) \pi(\boldsymbol{\beta}_d^{m-1})}\right\}.$$

end for

for $d = 1, \ldots, D$: **do**
Propose $\alpha_d^\star \sim \mathcal{N}(\alpha_d^{m-1}, \sigma_\alpha^2)$. Let $\alpha_d^m = \alpha_d^\star$ with probability

$$\min\left\{1, \frac{p(\boldsymbol{y}_d \mid \boldsymbol{X}^m, \boldsymbol{\beta}_d^m, \alpha_d^\star, \boldsymbol{\theta}^{m-1}) \pi(\alpha_d^\star)}{p(\boldsymbol{y}_d \mid \boldsymbol{X}^m, \boldsymbol{\beta}_d^m, \alpha_d^{m-1}, \boldsymbol{\theta}^{m-1}) \pi(\alpha^{m-1})}\right\}.$$

end for

for $k = 1, \ldots, K$: **do**
Propose $\boldsymbol{\theta}_k^\star \sim \mathcal{N}(\boldsymbol{\theta}_k^{m-1}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$. Let $\boldsymbol{\theta}_k^m = \boldsymbol{\theta}_k^\star$ with probability

$$\min\left\{1, \frac{p(\boldsymbol{x}_{:,k} \mid \boldsymbol{\theta}_k^\star) \pi(\boldsymbol{\theta}_k^\star)}{p(\boldsymbol{x}_{:,k} \mid \boldsymbol{\theta}_k^{m-1}) \pi(\boldsymbol{\theta}_k^{m-1})}\right\}.$$

end for
end for