Statistical Methods for Structured Data: Analyses of Discrete Time Series and Networks

W. Reed Palmer

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
under the Executive Committee
of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2023

Abstract

Statistical Methods for Structured Data: Analyses of Discrete Time Series and Networks

W. Reed Palmer

This dissertation addresses three problems of applied statistics involving discrete time series and network data. The three problems are (1) finding and analyzing community structure in directed networks, (2) capturing changes in dynamic count-valued time series of COVID-19 daily deaths, and (3) inferring the edges of an implicit network given noisy observations of a multivariate point process on its nodes. We use tools of spectral clustering, state-space models, Bayesian hierarchical modeling and variational inference to address these problems. Each chapter presents and discusses statistical methods for the given problem. We apply the methods to simulated and real data to both validate them and demonstrate their limitations.

In chapter 1 we consider a directed spectral method for community detection that utilizes a graph Laplacian defined for non-symmetric adjacency matrices. We give the theoretical motivation behind this directed graph Laplacian, and demonstrate its connection to an objective function that reflects a notion of how communities of nodes in directed networks should behave. Applying the method to directed networks, we compare the results to an approach using a symmetrized version of the adjacency matrices. A simulation study with a directed stochastic block model shows that directed spectral clustering can succeed where the symmetrized approach fails. And we find interesting and informative differences between the two approaches in the application to Congressional cosponsorship data.

In chapter 2 we propose a generalized non-linear state-space model for count-valued time

series of COVID-19 fatalities. To capture the dynamic changes in daily COVID-19 death counts, we specify a latent state process that involves second order differencing and an AR(1)-ARCH(1) model. These modeling choices are motivated by the application and validated by model assessment. We consider and fit a progression of Bayesian hierarchical models under this general framework. Using COVID-19 daily death counts from New York City's five boroughs, we evaluate and compare the considered models through predictive model assessment. Our findings justify the elements included in the proposed model. The proposed model is further applied to time series of COVID-19 deaths from the four most populous counties in Texas. These model fits illuminate dynamics associated with multiple dynamic phases and show the applicability of the framework to localities beyond New York City.

In Chapter 3 we consider the task of inferring the connections between noisy observations of events. In our model-based approach, we consider a generative process incorporating latent dynamics that are directed by past events and the unobserved network structure. This process is based on a leaky integrate-and-fire (LIF) model from neuroscience for aggregating input and triggering events (spikes) in neural populations. Given observation data we estimate the model parameters with a novel variational Bayesian approach, specifying a highly structured and parsimonious approximation for the conditional posterior distribution of the process's latent dynamics. This approach allows for fully interpretable inference of both the model parameters of interest and the variational parameters. Moreover, it is computationally efficient in scenarios when the observed event times are not too sparse. We apply our methods in a simulation study and to recorded neural activity in the dorsomedial frontal cortex (DMFC) of a rhesus macaque. We assess our results based on ground truth, model diagnostics, and spike prediction for held-out nodes.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I have completed this dissertation with the guidance of and in collaboration with my advisor Tian Zheng. The best parts are her work as well as mine. Chapter 2 is joint work with Tian Zheng and Richard Davis.

Thank you to my parents Bridget and Bill and my sister Caroline who supported me throughout this journey, showering me with love and encouragement in times of despair and joy.

And thank you to Alana [1] who fills my heart and whose companionship means everything to me as we navigate a life together amid our parallel doctoral journeys.

Dedication

for B&B

Introduction

Finding, understanding and taking advantage of structure in complex data is an essential part of statistics. When quantifying structure is itself a goal of the analysis, the statistician may start with few assumptions and set out to robustly learn interesting relations from the data. In other instances there may be particular structural patterns to search for, or a structural hypothesis to test. On the other hand, identifying structure can be a means to an end, for example to aid with a prediction task or to account for variation in the data. In this case, domain knowledge may inform a particular model for the data generating process that paves the way for inference and prediction. Given the data and task, a statistician may find it appropriate or necessary to impose simplifying structural assumptions like independence, linearity, or normality in their modeling. In other situations they may turn to machine learning methods that find and use structures in a black-box way, improving performance without providing interpretable insights.

This dissertation is a reflection of the varied and central role of structure in applied statics as it develops specialized methods to analyze discrete, temporal and relational data. It has three principal chapters which discuss three different applied statics problems.

Chapter 1 “Spectral Clustering for Directed Networks” [2] addresses the problem of finding and analyzing community structure in the presence of asymmetric connections. In this work we present an unsupervised learning method for clustering directed graphs. Through simulation and real data applications we show the importance of accounting for edge directionality.

In chapter 2 “Count-Valued Time Series Models for COVID-19 Daily Death Dynamics” [3] we propose and assess a progression of non-linear state-space time series models for 2020

COVID-19 mortality data in New York City. In this work, we choose second order differencing and an AR(1)-ARCH(1) model in order to accommodate dynamic structure in the data. Using a Bayesian hierarchical framework we make forecasts based on observed daily counts without explicitly accounting for any epidemiological mechanisms.

Chapter 3, "Inferring latent network edges from noisy event times with a leaky integrate-and-fire (LIF) model," considers discrete multivariate time series data with unobserved network structure. We consider the task of inferring edges while assuming a scientific form for the generative model that involves the latent network structure. We take an innovative and novel approach to the inference problem assuming a Leaky Integrate-and-Fire (LIF) model for neuronal dynamics.

Chapter 1: Spectral Clustering for Directed Networks

Community detection is a central topic in network science, where the community structure observed in many real networks is sought through the principled clustering of nodes. Spectral methods give well-established approaches to the problem in the undirected setting; however, they generally do not account for edge directionality. We consider a directed spectral method that utilizes a graph Laplacian defined for non-symmetric adjacency matrices. We give the theoretical motivation behind this directed graph Laplacian, and demonstrate its connection to an objective function that reflects a notion of how communities of nodes in directed networks should behave. Applying the method to directed networks, we compare the results to an approach using a symmetrized version of the adjacency matrices. A simulation study with a directed stochastic block model shows that directed spectral clustering can succeed where the symmetrized approach fails. And we find interesting and informative differences between the two approaches in the application to Congressional cosponsorship data.

1.1 Introduction

The goal of community detection–one of the most popular topics in statistical network analysis–is to identify groups of nodes that are more similar to each other than to other nodes in the network. Determining the number of communities in a given network and the community assignments gives key insight into the network structure, creating a natural dimensionality reduction of the data. Moreover, the existence of clusters of highly connected nodes is a feature of many empirical networks ([4], [5]). Though there is growing research for directed networks ([6], [7]), community detection is best understood and most often implemented on undirected networks. In directed networks, edge directionality is often fundamental, and communities of nodes may be characterized

by asymmetric relations. Consider, for example, citations, twitter follows and webpage hyperlinks. Properly accounting for edge directionality when analyzing such network data is very important.

Community detection is a clustering problem and requires an explicit notion of similarity between nodes. In general, clustering algorithms fall into two categories. There is model based clustering, which includes fitting procedures of a model with well-defined clusters, and there are methods motivated by what clusters of the data objects should look like. These methods specify a related objective function, and partition the data to optimize it, often approximately. For points in $\mathbb{R}^n$, Gaussian mixture modeling falls in the first category, while k-means falls in the second. The most popular community detection algorithms, including spectral clustering [8] and modularity [4], fall in the second category. However, these methods have been shown to provide consistent clustering for certain random graph models ([9], [10]).

A broadly applicable method for clustering relational data, spectral clustering requires a similarity matrix between the data objects. For graph representations of network data, the adjacency matrix of edge weights provides measures of similarity between all nodes. Thus spectral clustering is a natural choice for community detection. Spectral clustering is particularly well understood in the symmetric, undirected setting [11]. The problem is more complicated in the more general setting of weighted, undirected networks, which we consider. Building from [12] and [13], this paper presents some of the theory of spectral clustering for directed networks, as well as two applications.

Section 1.1.1 motivates spectral clustering, Section 1.1.2 presents its general framework, and Section 1.2 explains our approach to spectral clustering for directed graphs. Sections 1.3 and 1.4 delve into applications—a stochastic block model simulation study and an analysis of recent cosponsorship data from the U.S. Senate.

1.1.1 Motivation

In order to motivate the use of spectral clustering for directed networks, we consider a toy example involving points in $\mathbb{R}^2$.

Fig. 1.1a shows the points we wish separate into two spiral-shaped clusters. In Fig. 1.1b, k-

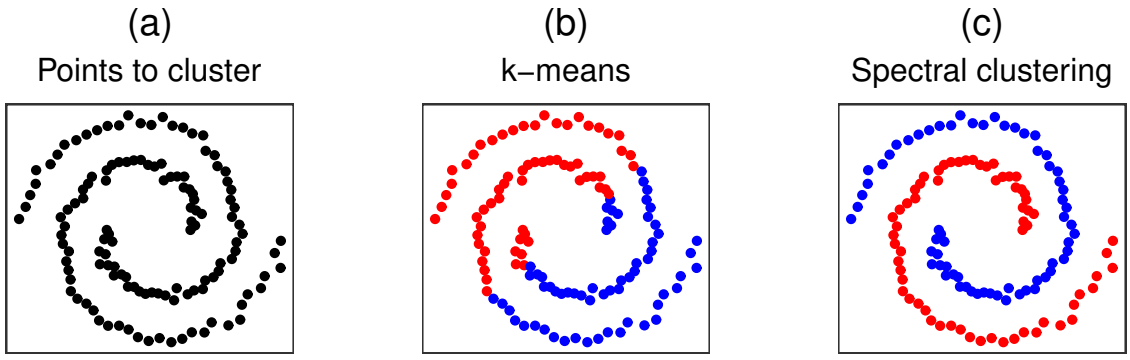

Figure 1.1: Clustering of points in $\mathbb{R}^2$ with k-means and spectral clustering.

means clustering fails to do this properly, since the clusters that we wish to capture have overlapping means. In Fig. 1.1c, spectral clustering properly separates the points.

Here we have defined a similarity matrix W_{ij} as the inverse Euclidean distance between points i and j if point j is among point i's four nearest neighbors, otherwise zero. The potential asymmetry of nearest neighbor relations means W is not symmetric, in general. Using our directed approach to spectral clustering, we are able to easily separate the points.

1.1.2 General spectral clustering algorithm

Consider the problem of partitioning n individual entities into k subsets. Generally spectral clustering ([11], [14]) proceeds in this way:

Algorithm 1 General spectral clustering

1. From data, construct an similarity matrix $W \in \mathbb{R}^{n\times n}$, where $W_{ij} \geq 0$ and $W_{ii} = 0$.
2. Compute a Laplacian $L \in \mathbb{R}^{n\times n}$ from W.
3. Compute first k eigenvectors of L, and combine into matrix X.
4. Cluster rows of X by k-means, or some other unsupervised algorithm.
5. Assign the original i^{th} entity to cluster ℓ iff the i^{th} row of X is assigned to ℓ.

Within this general framework, different approaches depend on the choice of Laplacian L, the inferring of k, manipulating of the eigenvectors in step 3 and the clustering method in step 4. Notable variations include principled eigenvector selection and clustering by Gaussian mixture modeling in [15], and Spectral Clustering On Ratios-of-Eigenvectors (SCORE) in [16], which relates to our approach, as detailed below.

For network-as-graph data, we begin with an adjacency matrix, and can skip directly to step 2. However, for directed networks, this W is not symmetric, and thus complicates the choice of Laplacian L. In the following section we motivate a graph Laplacian for directed, weighted networks, building towards it from an objective function corresponding to a notion of how communities should behave.

1.2 Spectral clustering for directed graphs

We begin with a directed, weighted graph $G = (\mathcal{V}, \mathcal{E})$ with n vertices, represented by the adjacency matrix W. For a given k, $2 \leq k \ll n$, we seek a 'best' partition $S_1, ..., S_k$ of $\mathcal{V}$, one that maximizes within-cluster similarity while minimizing between-cluster similarity. We consider a notion of similarity related to the behavior of a random walk on the vertices $\mathcal{V}$.

To introduce this random walk, we begin with a few assumptions. We assume G is strongly connected, that is, for all $i, j \in \mathcal{V}$ there exists a directed path $i \to j$. Note that breaking up a network into its connected components is a natural first step in community detection. We also assume that G is aperiodic. We define a transition probability matrix P by $P_{ij} \coloneqq W_{ij}/d_i^{\text{out}}$, where $d_i^{\text{out}} = \sum_j W_{ij}$ is the weighted out-degree of node i. Note $P = D^{-1}W$, where D is diagonal with $D_{ii} = d_i^{\text{out}}$. P is an irreducible aperiodic stochastic matrix, and thus has a unique stationary vector $\pi > 0$ satisfying $\pi^T P = \pi^T$, $\sum_i \pi_i = 1$. We define $\mathbf{\Pi}$ to be the diagonal matrix with $\mathbf{\Pi}_{ii} = \pi_i$, which we will use in the sequel.

With P and π it is natural to define a random walk $(N_t)_{t\in\mathbb{N}}$ on $\mathcal{V}$. In particular we can initialize the random walk according to π and then transition between nodes according to P. For a network with strong community structure, we expect this random walk to stay within the true communities

more often than move between them. This leads to a notion of a 'good' community S—given that the random walk is on one of its nodes, the probability that next step jumps to a different community, i.e. $\mathbb{P}(N_{t+1} \notin S | N_t \in S)$, should be relatively low. The sum of these conditional probabilities across all communities in a given k-partition provides an objective function to minimize. In particular, we wish to find community assignments that solve:

$$\min_{S_1,\ldots,S_k} \sum_{1\leq \ell \leq k} \mathbb{P}(N_{t+1} \notin S_\ell | N_t \in S_\ell). \tag{1.1}$$

It is important to note that this objective function measuring the community assignments $S_1, ..., S_k$ takes fully into account the directionality of edges in G. This follows because the random walk N_t comes from the asymmetric transition matrix $P = D^{-1}W$. This objective is equivalent to the normalized cut criterion $\text{NCut}(S_1, ..., S_k)$ for directed graphs in [12].

In (1.1) we have a discrete, non-convex optimization problem that is not readily solvable. Searching over all k-partitions is computationally intractable for even small networks. For example, there are over 580 million ways to divide 20 objects into 3 non-empty sets! Seeking an approximation solution, we proceed by rewriting the optimization problem in a form with a convex relaxation.

From G and a k-partition $S_1, ..., S_k$ of $[n]$, we define $\boldsymbol{g} = [g^1 \ldots g^k] \in \mathbb{R}^{n\times k}$ by

$$g_i^\ell = \begin{cases} \frac{\sqrt{\pi_i}}{\sqrt{\sum_{j\in S_\ell} \pi_j}} & \text{if } i \in S_\ell \\ 0 & \text{otherwise.} \end{cases}$$

This matrix encodes the node assignments of $S_1, ..., S_k$, has orthonormal columns, and can be shown (we do not go through the details here) to satisfy the equality

$$\text{Tr}(\boldsymbol{g}^T L \boldsymbol{g}) = \sum_{1\leq \ell \leq k} \mathbb{P}(N_{t+1} \notin S_\ell | N_t \in S_\ell), \tag{1.2}$$

where $L = I - \frac{\boldsymbol{\Pi}^{1/2} P \boldsymbol{\Pi}^{-1/2} + \boldsymbol{\Pi}^{-1/2} P^T \boldsymbol{\Pi}^{1/2}}{2}$ is the graph Laplacian matrix for directed networks first proposed in [13]. Thus the optimization problem (1.1) is rewritten as the minimization of the left

hand side of (1.2). While this formulation is no easier to solve exactly, it has a natural convex relaxation:

$$\min_{\substack{V \in \mathbb{R}^{n \times k} \\ V^T V = I}} \operatorname{Tr}(V^T L V).$$

Here we are minimizing a Rayleigh quotient, so that a solution is the matrix X with columns given by normalized eigenvectors corresponding to the k smallest eigenvalues of L.

What remains is to determine a clustering from these eigenvectors of L, which constitute a loose approximation to the highly structured $\boldsymbol{g}$. Hence step 4 of the spectral clustering algorithm, for which we use k-means to create a partition. Note that 0 is the smallest eigenvalue of L, corresponding to eigenvector $\sqrt{\pi}$. Now the stationary vector π describes the limiting behavior of the random walk N_t on $\mathcal{V}$ and relates to the degree distribution of G. It seems reasonable to question whether clustering should depend on the stationary distribution π, since this limiting behavior may be ancillary to existing community structure.

These considerations motivate clustering the rows of a transformed version of X, $X^* = \mathbf{\Pi}^{-1/2} X$. Here the i^{th} entry of each eigenvector is divided by $\sqrt{\pi_i}$. The first column of $\mathbf{\Pi}^{-1/2} X$ will be constant and equal to one, and therefore can be discarded. This 'dividing out' of the leading eigenvector agrees with the SCORE method for undirected networks. In [16], it is shown that the largely ancillary effects of degree heterogeneity in the Degree Corrected Stochastic Block Model are effectively removed by taking such entry-wise ratios.

In practice, when applied to various networks induced by the congressional co-sponsorship data discussed below, the values of the objective function (1.1) are consistently lower when clustering on the rows of $\mathbf{\Pi}^{-1/2} X$ as opposed to X, suggesting better resulting communities.

We now present in full our approach to spectral clustering for directed networks. We begin with a weighted, directed graph G defined by the adjacency matrix W, and a specified number of communities k. This is a modified version of Algorithm 1, above.

Algorithm 2 Spectral clustering for directed networks

1. From W, compute P, $\mathbf{\Pi}$ and $L = I - \frac{\mathbf{\Pi}^{1/2} P \mathbf{\Pi}^{-1/2} + \mathbf{\Pi}^{-1/2} P^T \mathbf{\Pi}^{1/2}}{2}$.

2. Compute the $k-1$ eigenvectors corresponding to the 2^{nd}-k^{th} smallest eigenvalues of L, and combine into matrix X.

3. Compute $X^* = \mathbf{\Pi}^{-1/2}X$, and normalize its columns.

4. Cluster rows of normalized X^* into k groups $1, ..., k$ by k-means.

5. Assign i^{th} node of G to community ℓ if and only if i^{th} row of X^* is assigned to ℓ.

The computational complexity of this algorithm comes mostly from obtaining the k leading eigenvectors of L. The simple power method can be used to find leading eigenvectors, and when the adjacency matrix is sparse, as in many network applications, this complexity is slightly larger than $O(kn^2)$ ([17], [16]).

Note that when the adjacency matrix W is symmetric, we have that $L = I - D^{-1/2}WD^{1/2}$, which is precisely the normalized Laplacian L_{sym} for symmetric similarity matrices used in [14] and highlighted in [11]. This follows since $\pi^T = (d_1^{\text{out}}, ..., d_n^{\text{out}})/\sum_i d_i^{\text{out}}$ when $W^T = W$.

The question naturally arises as to how to choose k, the number of communities. This is an important question in all clustering problems. While there may exist prior knowledge about the true number of communities in a given network, often k is unknown, unfixed and needing to be learned from the data. In general, for clustering algorithms, there are many methods for choosing k. One method devised particularly for spectral clustering is the eigengap heuristic [11]. It stipulates that we should choose k such that the first (smallest) k eigenvalues $\lambda_1, ..., \lambda_k$ are relatively small, but λ_{k+1} is relatively large. We follow the eigengap heuristic in the applications below, choosing values of k such that $\lambda_{k+1} - \lambda_k$ is relatively large.

1.3 Simulation study

We test the directed spectral clustering algorithm on networks simulated from a directed stochastic block model (SBM). Good performance on SBMs [18] is considered a necessary condition for useful community detection algorithms. However, block models do not account for complexities observed in many empirical networks, and thus do not alone provide sufficient criteria [9].

To generate a directed binary adjacency matrix $W \in \{0,1\}^{n\times n}$, we assign n nodes randomly to communities 1, 2 and 3 with probabilities .3, .3 and .4, respectively, and then simulate an independent Bernoulli edge for each directed pair (i, j), $i \neq j$ of nodes with probability $z_i^T Z z_j$, where

$$Q = \begin{bmatrix} .3 & .01 & .01 \\ .3 & .3 & .01 \\ .25 & .01 & .3 \end{bmatrix}$$

and $z_1, \ldots, z_n$ encode the community assignments.

We compare the performance of applying Algorithm 2 with W to an undirected approach in which we apply Algorithm 2 with $W_{\text{sym}} = W + W^T$. With this symmetrization, we effectively regard each directed edge as undirected. Using a naive graph transformation like W_{sym} is a common approach to community detection for directed networks ([6]). However, ignoring information about directionality can be problematic, and by using W_{sym}, we lose key information to help determine the correct k, and distinguish between communities 1 and 2.

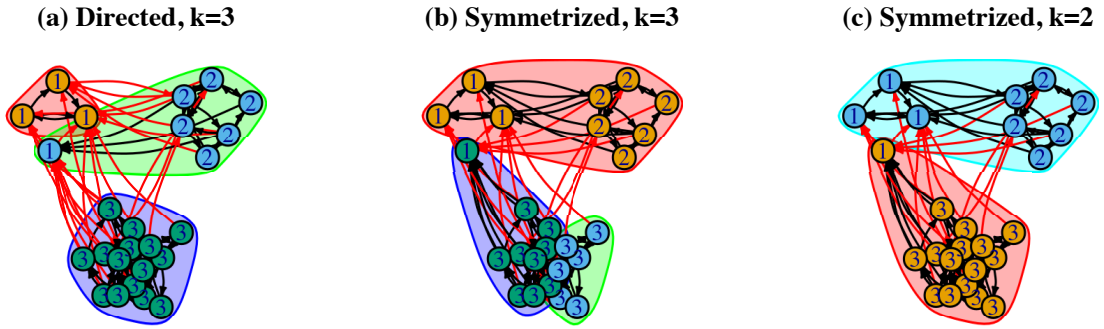

Figure 1.2: Sociograms of the simulated network of size 25.

Fig. 1.2 shows a single simulated network of size $n = 25$ along with the clustering results. Using the directed adjacency matrix and $k = 3$, Algorithm 2 nearly recovers the true communities, misclassifying just one node. On the other hand, the results of Algorithm 2 with the symmetrized adjacency matrix W_{sym} and $k = 3$ combine nodes from communities 1 and 2, and split the nodes of group 3 into two clusters. For $k = 2$, the symmetrized approach nearly recovers group 3.

Increasing the size of the simulated network to $n = 100$ tells a somewhat similar story, with improvements for the symmetrized approach. Fig. 1.3 shows the simulated network as adjacency matrix heatmaps. The block structure associated with the true groupings in Fig. 1.3a is clear. While node indices vary across the three panels, the estimated clusters in Fig. 1.3b-c are arranged to best align with the true blocks. Fig. 1.3b shows again the near recovery of the true community structure by directed spectral clustering. In Fig. 1.3c it is clear that the symmetrized approach with continues to struggle to separate the communities correctly.

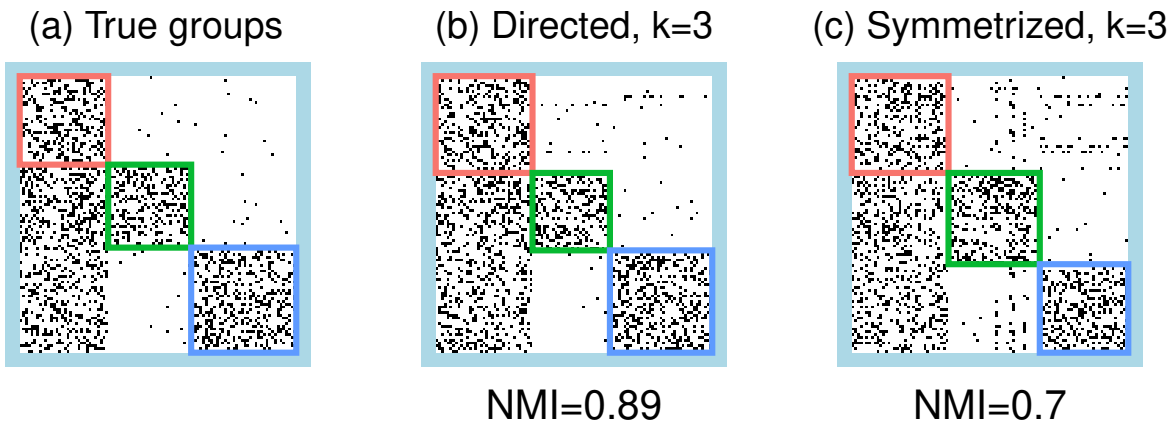

Figure 1.3: Adjacency matrix heatmaps of a simulated network of size 100.

Across the bottom of Fig. 1.3b-c is the Normalized Mutual Information (NMI) measure between the true communities and the estimated clusterings. A value of 1 indicates exact agreement up to cluster relabeling. NMI is an information theoretic measure, relating the information needed to infer one cluster from the other. NMI satisfies desirable normalization and metric properties, and is adjusted for chance [19].

Fig. 1.4 summarizes results of the spectral approaches on 100 repeated simulations of the same stochastic blockmodel, for increasing numbers of nodes. Assuming the number of communities k is unknown, we would need to infer it from the data. Fig. 1.4a shows the proportion of simulations where the eigengap heuristic correctly chooses $k = 3$ over $k = 2$. This is shown separately based on eigenvalues from the directed and symmetrized approaches. Under the directed approach, the rate at which the eigengap heuristic chooses correctly increases with the network size, reaching 100%

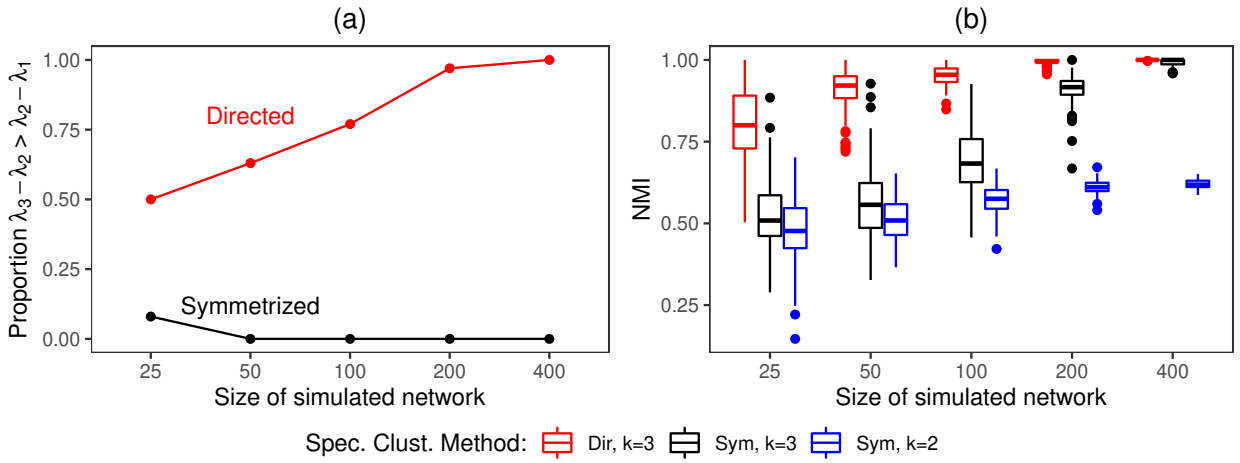

Figure 1.4: Results of spectral approaches on 100 simulations at each network size. (a): Proportion of simulations where the eigengap heuristic correctly chooses $k = 3$ over $k = 2$. (b): NMI between true grouping and estimated clusterings.

for $n = 400$. On the other hand, under the symmetrized approach, the heuristic always chooses $k = 2$ over $k = 3$, for $n \geq 50$. Thus despite the success of the symmetrized approach as n increases (as seen in Fig. 1.4b), without prior knowledge, we would choose $k = 2$ communities rather than $k = 3$. Overall, Fig. 1.4b shows the superior performance the directed approach with $k = 3$, which consistently achieves an exact recovery of the true communities for $n \geq 200$.

We found that skipping step 3 of Algorithm 2, and not 'dividing out' the first eigenvector leads to better performance on these simulations. This makes sense because there is no degree heterogeneity within communities, and, moreover, community assignment is characterized by the in- and out-degree distributions. In such cases it is better to cluster the rows of X, not X^*.

1.4 Congressional cosponsorship

Cosponsorship of bills in the U.S. Congress constitutes directed relational data. Previous network analysis of cosponsorship is found in [20]. Undirected modularity based community detection is applied to these networks in [21].

Every bill or amendment in Congress has one sponsor who introduces the measure, and may have one or more cosponsors, whose cosponsorship is generally viewed as an indication of support

[22]. We represent cosponsorship of a bill as a set of directed binary edges from cosponsor to sponsor, one for each of the bill's cosponsors, and we consider the weighted, directed graphs among members of Congress created by counting these directed binary edges across a set of bills and amendments. In this paper we analyze 21 months of Senate legislation from January 1, 2019 to September 30, 2020. This constitutes the data available at the time of writing from the 116th Congress. It includes 1,377 bills and amendments, from which we extract 7,667 cosponsorship edges.

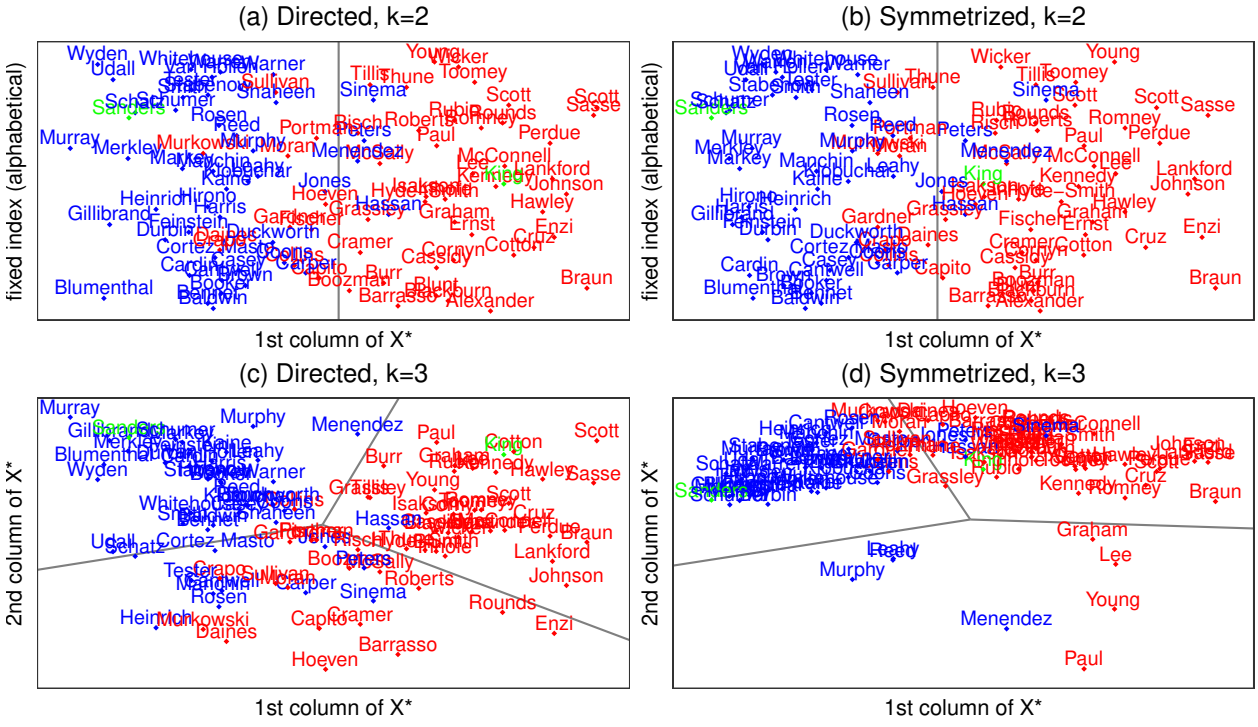

Figure 1.5: Embeddings of Senators from ratios of eigenvectors of the Laplacian, with clustering boundaries from k-means.

The largest strongly connected component of the 116th Senate cosponsorship network includes 99 Senators, and contains 4,029 directed, weighted edges. We apply Algorithm 2 with the weighted directed adjacency matrix W, as well as with the naively symmetrized matrix $W_{\text{sym}} = W + W^T$. In both approaches, the eigengap heuristic does not provide strong evidence of community structure, with the first difference dominating. However, the second and third eigengaps are larger than the rest, indicating the possibility of $k = 2$ or $k = 3$ communities. Prior knowledge of the U.S. two party

system along with current polarization points to $k = 2$; however, the persistent need for bipartisan legislation and the existence of moderate lawmakers on both sides suggests the possibility of $k > 2$.

Fig. 1.5 shows results of the two spectral clustering approaches for $k = 2$ and $k = 3$ communities. Here we plot the columns of X^* from step 3 of Algorithm 2, along with the decision boundary separating the detected communities. The colors indicate party affiliation—blue for Democrat, green for Independent, and red for Republican. The results for $k = 2$ (Fig. 1.5a-b) are similar for the directed and symmetrized approaches, with, respectively, 85 and 86 percent of Senators clustered with the majority of their party, excluding independents.

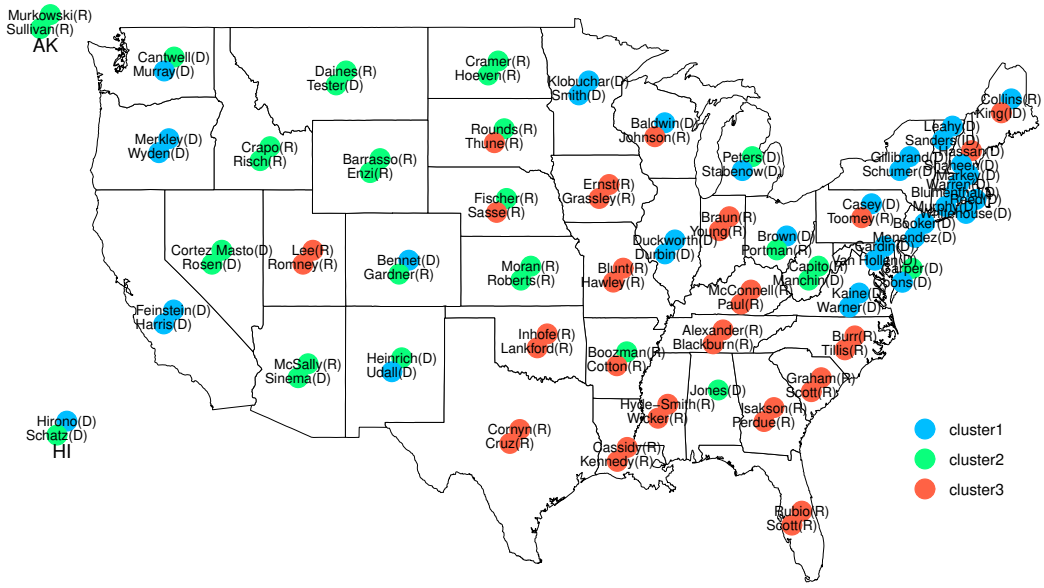

Figure 1.6: Geographic relation of directed spectral clustering results on the 116$^{\text{th}}$ Senate, for $k = 3$.

The results for $k = 3$ (Fig. 1.5c-d), however, differ greatly between the two approaches. The directed approach detects balanced and relatively well-separated communities, two of which align closely with party, and one that contains a mix of Republicans and Democrats mainly from the Plains, Mountain West, Southwest, and non-contiguous states. Fig. 1.6 shows the full geographic correspondence of the detected communities. Meanwhile, the symmetrized approach detects one diffuse and separated community of four Democrats and four Republicans, and splits the remaining Senators roughly along the same lines as in Fig. 1.5b.

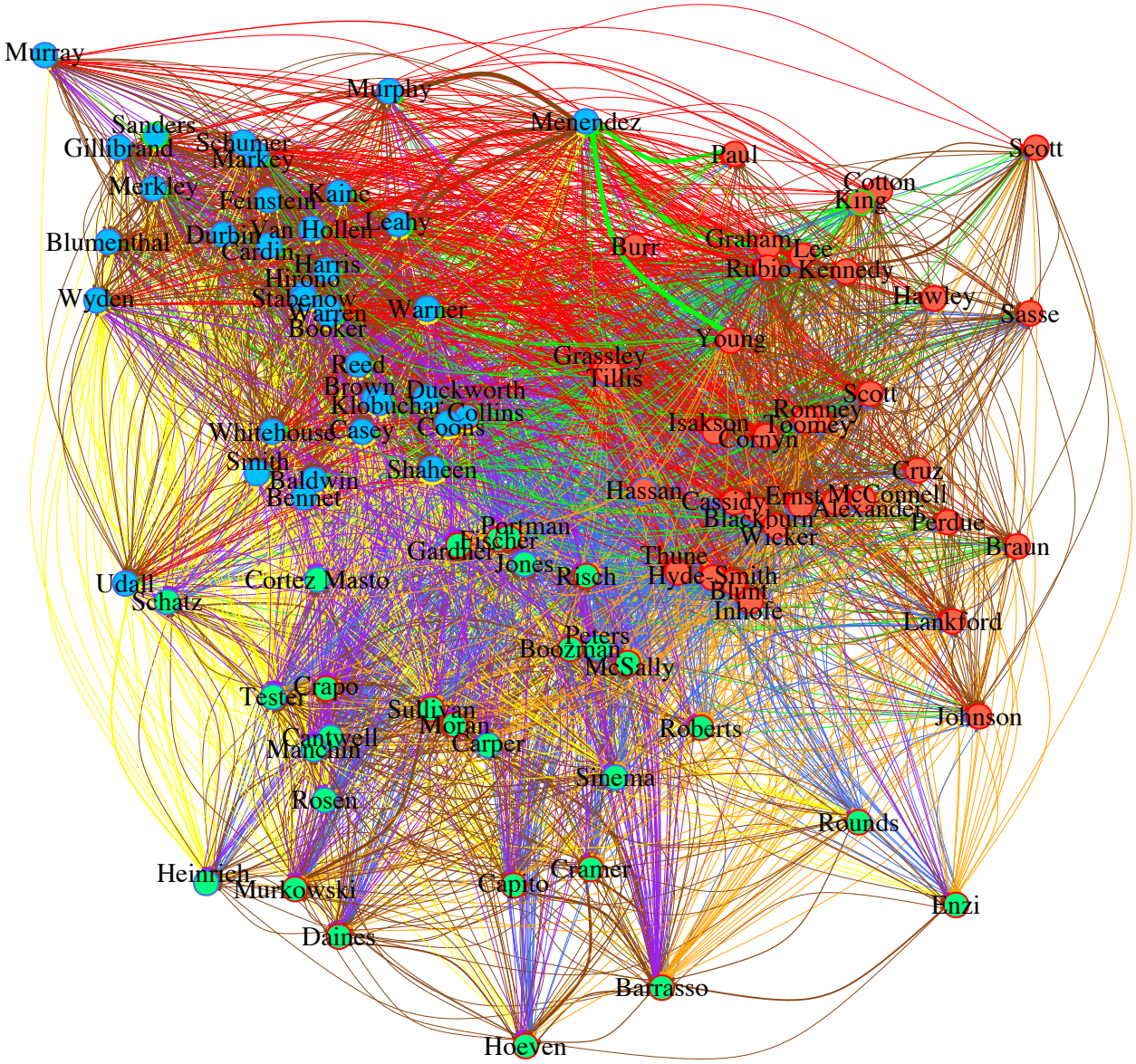

Figure 1.7: Senate cosponsorship with clustering and embedding from Algorithm 2.

In Fig. 1.7 we use the same embedding as Fig. 1.5c to lay out a sociogram of the Senate cosponsorship network. In general, since spectral clustering methods provide embeddings, we can use them for visualization. The node interior coloring corresponds to detected communities, while the node outline color indicates party. Within cluster edges are brown, while between cluster edges are colored according to the community assignments of the cosponsor and sponsor nodes.

The imbalance of flows within and between clusters is apparent. We see a higher concentration

of brown edges among the Republican core in the top right, and more inter-cluster out edges (purple and red) than inter-cluster in edges (yellow and green) for the Democrats in the top left. These patterns are borne out more clearly by the inter and intra community intensities in Fig. 1.8. Here we show the observed cosponsorship counts divided by the number of pairs of distinct legislators. The rows correspond, in order, to the blue (Democrat), green (mixed) and red (Republican) communities. The directed approach reflects inter-community asymmetries, while the symmetrized approach does not.

A notable feature of Fig. 1.7 and an exception to the patterns discussed above are four very prominent green edges from Republicans Graham, Lee, Paul and Young into Menendez (D-NJ) at the top middle, mirrored by three prominent brown edges into Menendez from Democrats Leahy, Murphy and Reed. These are precisely the eight Senators clustered together by the symmetrized approach with $k = 3$, appearing at the bottom of Fig. 1.5d. We isolate this star-like subnetwork in Fig. 1.9. Here we include three nodes for the remaining Senators of each detected community and show the combined weighted edges between the eight individual Senators and these 'remaining' clusters. The edges flowing into Menendez are blue, those flowing out from Menendez are red, and the rest are grey.

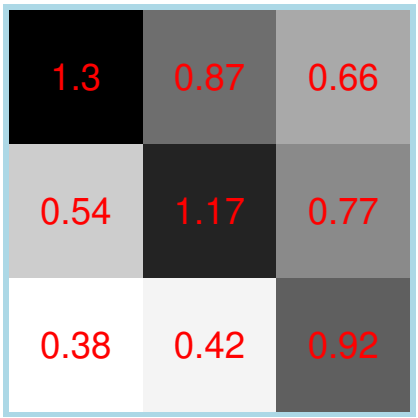

Figure 1.8: Inter and intra community intensities

Each blue edge from the Senators besides Menendez represents more than 23 cosponsorships, combining for a total of 174. Menendez is the minority ranking member of the Committee

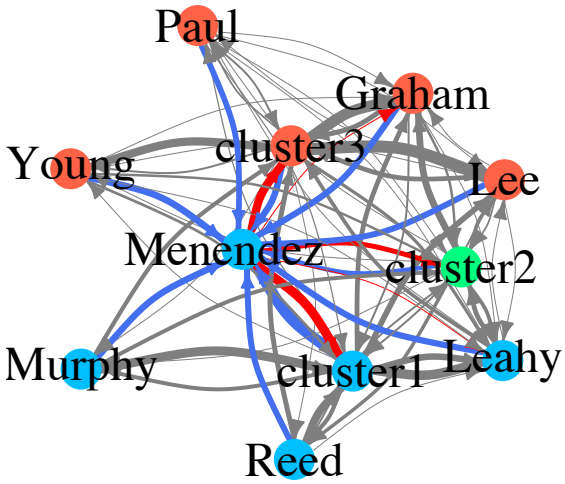

Figure 1.9: Collapsed subnetwork

on Foreign Relations, and 169 of these cosponsorships involve international affairs legislation. Menendez cosponsors just 4 bills in return, and the 'other seven' have only 18 cosponsorships among themselves. Meanwhile, all four democrats exchange heavily with the remaining Senators in cluster 1, while the Republicans exchange with those remaining in cluster 3. Considering edge directionaliy, these eight Senators do not form a natural community within the context of the entire network. The directed approach reflects this, splitting these Senators along party lines. Unable to account for the patent asymmetry, the symmetrized approach allows the high weight of the edges flowing into Menendez to pull these Senators closer together, distorting the entire embedding, as seen in Fig. 1.5d, and classifying them as a separate community.

Data Note. Bill cosponsorship data is available from the the ProPublica Congress API [23]. Amendment cosposorship is obtained directly from Congress.gov [24].

1.5 Conclusion

In this paper we presented a variation of the general spectral clustering algorithm adopted for community detection on directed networks. We described the theoretical motivation behind the directed graph Laplacian, showing its connection to an objective function that reflects a notion of how communities of nodes in directed networks should behave. We applied our algorithm to

simulated and empirical networks, and found encouraging and insightful results. When we ignore edge directionality by using a symmetrized adjacency matrix, we observe different results and worse performance on the simulated networks.

We see clear advantages to taking full account of the directionality of edges in complex networks. This is an important area of continued research, both from a theoretical and applied perspective.

1.6 Chapter 1 Appendix

This appendix provides details for the derivations in Section 1.2. Our work here follows [12], where derivations are given for case where $k = 2$. It is not entirely trivial to move to the case considered here, where $k > 2$.

For any subset $S \subset \mathcal{V}$, we define the outboundary $\partial S := \{(u, v) \in \mathcal{E} : u \in S, v \in S^c\}$. Further we define $\mathrm{Vol}(S) = \sum_{u \in S} \pi(u)$ and $\mathrm{Vol}(\partial S) = \sum_{(u,v) \in \partial S} \pi(u) p(u, v)$. In [12], the normalized cut criterion for directed graphs is defined:

$$\mathrm{NCut}(S_1, ..., S_k) = \sum_{1 \leq \ell \leq k} \frac{\mathrm{Vol}(\partial S_\ell)}{\mathrm{Vol}(S_\ell)}. \tag{1.3}$$

To see that this is equal to the sum of conditional probabilities in (1.1), we note that

$$\begin{aligned} \mathbb{P}(N_{t+1} \notin S | N_t \in S) &= \frac{\mathbb{P}(N_{t+1} \notin S, N_t \in S)}{\mathbb{P}(N_t \in S)} \\ &= \frac{\sum_{u \in S, v \in S^c} \mathbb{P}(N_{t+1} = v | N_t = u) \mathbb{P}(N_t = u)}{\sum_{u \in S} \mathbb{P}(N_t = u)} \\ &= \frac{\mathrm{Vol}(\partial S)}{\mathrm{Vol}(S)}. \end{aligned} \tag{1.4}$$

Next we will prove the equality (1.2). We can think of the columns of $\boldsymbol{g}$, defined above in Section

1.2, as mappings $\{g^\ell : \mathcal{V} \to \mathbb{R}\}$, where

$$g^\ell(u) = \begin{cases} \frac{\sqrt{\pi(u)}}{\sqrt{\mathrm{Vol}(S_\ell)}} & \text{if } u \in S_\ell \\ 0 & \text{otherwise.} \end{cases}$$

Now let's consider the functional Ω on all mappings $\{g : \mathcal{V} \to \mathbb{R}\}$ given by

$$\Omega(g) = \frac{1}{2} \sum_{(u,v)\in\mathcal{E}} \pi(u)p(u,v) \left(\frac{g(u)}{\sqrt{\pi(u)}} - \frac{g(v)}{\sqrt{\pi(v)}} \right)^2.$$

On the one hand we have for $\ell = 1, ..., k$,

$$\Omega(g^\ell) = \frac{1}{2\mathrm{Vol}(S_\ell)} \left(\sum_{(u,v)\in\partial S_\ell} \pi(u)p(u,v) + \sum_{(u,v)\in\partial S_\ell^c} \pi(u)p(u,v) \right) = \frac{\mathrm{Vol}(\partial S_\ell)}{\mathrm{Vol}(S_\ell)}. \tag{1.5}$$

Here we have used the fact that $\mathrm{Vol}(\partial S_\ell) = \mathrm{Vol}(\partial S_\ell^c)$, which follows from:

$$\begin{aligned} \sum_{(u,v)\in\partial S} \pi(u)p(u,v) &= \sum_{u\in\mathcal{V}} \sum_{v\in S^c} \pi(u)p(u,v) - \sum_{u\in S^c} \sum_{v\in S^c} \pi(u)p(u,v) \\ &= \sum_{v\in S^c} \pi(v) - \sum_{u\in S^c} \sum_{v\in S^c} \pi(u)p(u,v) \\ &= \sum_{v\in S^c} \sum_{u\in\mathcal{V}} \pi(v)p(v,u) - \sum_{u\in S^c} \sum_{v\in S^c} \pi(u)p(u,v) \\ &= \sum_{v\in S^c} \sum_{u\in S} \pi(v)p(v,u). \end{aligned}$$

On the other hand, for $g \in \{g : \mathcal{V} \to \mathbb{R}\}$, we can write $\Omega(g) = \frac{1}{4}(A(g) + B(g))$, where

$$\begin{aligned} A(g) &= \sum_{v\in\mathcal{V}} \sum_{u:(u,v)\in\mathcal{E}} \pi(u)p(u,v)\left(\frac{g(u)}{\sqrt{\pi(u)}} - \frac{g(v)}{\sqrt{\pi(v)}}\right)^2 \\ &= \sum_{v\in\mathcal{V}} \sum_{u:(u,v)\in\mathcal{E}} p(u,v)g(u)^2 - \frac{2\sqrt{\pi(u)}p(u,v)g(u)g(v)}{\sqrt{\pi(u)}} + \frac{\pi(u)p(u,v)g(v)^2}{\pi(v)} \\ &= \sum_{u\in\mathcal{V}} g(u)^2 + \sum_{v\in\mathcal{V}} \frac{\pi(v)g(v)^2}{\pi(v)} - 2\sum_{v\in\mathcal{V}}\sum_{u\in\mathcal{V}} \frac{\sqrt{\pi(u)}p(u,v)g(u)g(v)}{\sqrt{\pi(u)}} \\ &= 2g^Tg - 2g^T\mathbf{\Pi}^{1/2}P\mathbf{\Pi}^{-1/2}g, \end{aligned}$$

and analogously,

$$B(g) = \sum_{v\in\mathcal{V}} \sum_{u:(v,u)\in\mathcal{E}} \pi(v)p(v,u)\left(\frac{g(v)}{\sqrt{\pi(v)}} - \frac{g(u)}{\sqrt{\pi(u)}}\right)^2 = 2g^Tg - 2g^T\mathbf{\Pi}^{-1/2}P^T\mathbf{\Pi}^{1/2}g.$$

Thus

$$\Omega(g) = g^T\left(I - \frac{\mathbf{\Pi}^{1/2}P\mathbf{\Pi}^{-1/2} + \mathbf{\Pi}^{-1/2}P^T\mathbf{\Pi}^{1/2}}{2}\right)g.$$

Taking $L = I - \frac{\mathbf{\Pi}^{1/2}P\mathbf{\Pi}^{-1/2}+\mathbf{\Pi}^{-1/2}P^T\mathbf{\Pi}^{1/2}}{2}$, as above, and combining the previous equality with (1.3), (1.4) and (1.5), we have

$$\sum_{1\le\ell\le k} \mathbb{P}(N_{t+1} \notin S_\ell | N_t \in S_\ell) = \mathrm{NCut}(S_1, ..., S_k) = \sum_{1\le\ell\le k} \Omega(g^\ell) = \mathrm{Tr}(\boldsymbol{g}^T L\boldsymbol{g}),$$

as desired.

Chapter 2: Count-Valued Time Series Models for COVID-19 Daily Death Dynamics

We propose a generalized non-linear state-space model for count-valued time series of COVID-19 fatalities. To capture the dynamic changes in daily COVID-19 death counts, we specify a latent state process that involves second order differencing and an AR(1)-ARCH(1) model. These modeling choices are motivated by the application and validated by model assessment. We consider and fit a progression of Bayesian hierarchical models under this general framework. Using COVID-19 daily death counts from New York City's five boroughs, we evaluate and compare the considered models through predictive model assessment. Our findings justify the elements included in the proposed model. The proposed model is further applied to time series of COVID-19 deaths from the four most populous counties in Texas. These model fits illuminate dynamics associated with multiple dynamic phases and show the applicability of the framework to localities beyond New York City.

2.1 Introduction

Since early 2020, the COVID-19 pandemic has posed a sustained threat to public health across the globe. With more than 100 million confirmed cases and 2.15 million confirmed deaths across 192 territories [25][1], the infectious respiratory disease has spread, sickened and killed on an unprecedented scale.

The staggering global totals arise from local, transient outbreaks, in which disease spreading contacts, new cases and subsequent deaths are concentrated in space and time. Long range human mobility creates contacts across the globe, and a few such links can carry an epidemic between

[1] JHU dashboard on January 26, 2021

distant cities. Local contacts, however, drive the outbreaks of COVID-19, creating waves of cases and then, later on, waves of hospitalizations and deaths.

In late February and early March, following the original outbreak in China and amid the escalating situation in Europe, COVID-19 quietly took hold in New York City. At the time of the city's first confirmed case on March 1, it is estimated that there were over 10,000 infections [26]. Control measures were slow to be put in place, with bars, restaurants and schools staying open for the next two weeks, and no public face covering requirement until April 15. New York City quickly became the epicenter of the pandemic, with hospitals stretched, ICUs filling up, and sickened New Yorkers dying at an alarming rate.

In this paper we assess how a dynamic model can track the trends in observed COVID-19 mortality data. We propose a non-linear state-space time series model that is fitted to daily COVID-19 deaths in New York City, and we evaluate forecasts of held-out, look-ahead counts. Moreover, we fit and assess a succession of models within the same framework, where the progression from simpler to more complex is driven by the behavior we wish to capture—volatility in the region of high curvature and the subsequent settling following the observed peak. We use second order differencing to model these dynamics in order to de-trend the observed series and capture the smoothness in the daily deaths curvature.

Mathematical epidemiological models can generally be characterized as either mechanistic or 'phenomenological' [27]. Existing models of COVID-19 deaths fall in both categories [28], [29], [30], [31]. Our model falls into the second category, using observed historical data to make predictions, without explicitly accounting for any epidemiological mechanisms.

Good data alongside good modeling is essential for successful COVID-19 forecasting [32], [33]. We use deaths reported by the New York City Department of Health [34]. These deaths are confirmed by the City's Office of the Chief Medical Examiner and the Health Department's Bureau of Vital Statistics, a measure that can improve uniformity and reliability [35]. The counts include decedents who were NYC residents and had a positive laboratory test for the coronavirus. Figure 2.1 shows these confirmed counts in the five New York City boroughs in the spring of 2020.

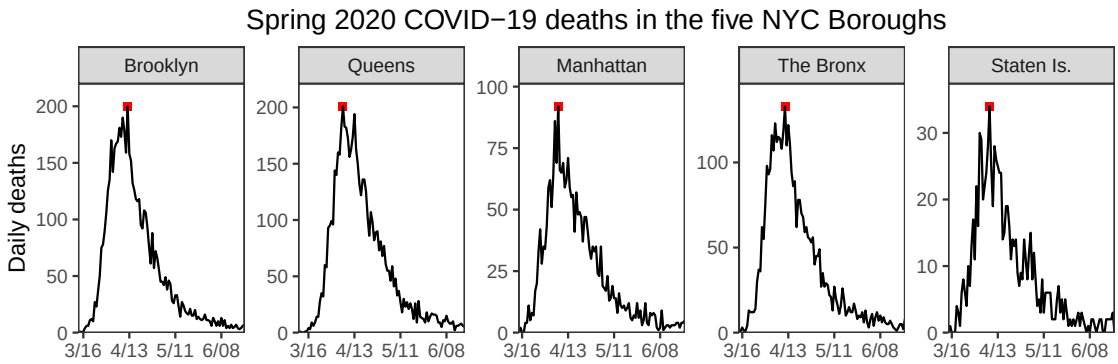

Figure 2.1: One hundred days of New York City COVID-19 mortality data. The observed peak is indicated in red.

This paper describes the statistical methods we employ and presents the results of applying these methods to observed COVID-19 death counts. Section 2.2.1 introduces the modeling framework, and Section 2.2.2 specifies the general form and progression of our considered models. Section 2.2.3 details the tools for predictive model assessment. In Section 2.3.1 we compare the progression of model fits to New York City data. Sections 2.3.2 and 2.3.3 further examine fits from the proposed model. Section 2.3.2 shows posterior estimates around the observed April peaks for all five NYC boroughs, while Section 2.3.3 shows fits to longer series of observed deaths from the four most populous Texas counties.

2.2 Methods

We take a nonlinear state-space model approach to modeling daily death counts. This approach fits nicely into a hierarchical modeling framework for which Bayesian estimation methods are accessible. Specifically, we use the Monte Carlo methods [36, 37] for fitting our model, obtaining posterior samples of the model parameters. Out-of-sample look-ahead predictions are then found by drawing from posterior predictive distributions.

2.2.1 Model framework

Let y_t be the observed daily count on day t. We assume that y_t given λ_t follows a parametric distribution for count data with mean λ_t. The dynamics of the time-varying mean λ_t is specified through a latent state process s_t, which updates according to a density that depends on the previous state s_{t-1}.

Let $p(y_t|\lambda_t, \beta)$ and $p(s_t|s_{t-1}, \beta)$ denote the conditional observational density and state update density, respectively, where β includes model parameters that do not vary with time. To complete the Bayesian specification, we assign prior distributions $p(\beta)$ and $p(s_1, ..., s_{t_0})$ to the conditioning parameters β and the initial states $s_1, ..., s_{t_0}$, respectively. For second order differencing we must initialize $t_0 = 3$ states. The model specification is:

$$\begin{aligned} y_t &\sim p(y_t|\lambda_t, \beta), && t = 1, ..., T \\ \lambda_t &\hookleftarrow s_t && t = 1, ..., T \\ s_t &\sim p(s_t|s_{t-1}, \beta) && t = t_0 + 1, ..., T \\ (s_1, ..., s_{t_0}) &\sim p(s_1, ..., s_{t_0}) \\ \beta &\sim p(\beta), \end{aligned} \tag{2.1}$$

where $\lambda_t \hookleftarrow s_t$ means that λ_t is encoded in the latent state s_t. Given T observations $y_{1:T} = (y_1, ..., y_T)$, we can fit this generative model using Monte Carlo methods and obtain estimates of the posterior distribution $p(s_{1:T}, \beta|y_{1:T})$.

We make k-day ahead forecasts based on the posterior predictive distribution

$$p(y_{T+k}|y_{1:T}) = \int p(y_{T+k}|s_{T+k}, \beta) \left[\prod_{\ell=1}^{k} p(s_{T+\ell}|s_{T+\ell-1}, \beta) \right] p(s_T, \beta|y_{1:T}) \, \mathrm{d}s_{T:T+k} \, \mathrm{d}\beta. \tag{2.2}$$

It is straightforward to sample from this distribution using the posterior samples from $p(s_T, \beta|y_{1:T})$.

In particular we can draw:

$$
\begin{aligned}
(s_T, \beta) &\sim p(s_T, \beta | y_{1:T}) \\
s_{T+\ell} &\sim p(s_{T+\ell} | s_{T+\ell-1}, \beta) \qquad \ell = 1, ..., k \\
y_{T+k} &\sim p(y_{T+k} | s_{T+k}, \beta).
\end{aligned} \tag{2.3}
$$

A set of draws $y_{T+k}^{(1)}, ..., y_{T+k}^{(B)}$ from (2.3) approximates (2.2). We make point estimates and construct intervals from these draws, and evaluate predictive performance by considering an associated loss. We assess model fit by considering held-out k-day ahead predictions relative to these posterior predictive draws.

2.2.2 Models considered

Within this generalized state-space, parameter-driven modeling framework [38], we consider and fit a sequence of models. These models vary in their observation equation $p(y_t | \lambda_t, \beta)$ and state update density $p(s_t | s_{t-1}, \beta)$. For the former, we begin with the most natural and simple choice, the Poisson distribution, and later compare with the negative binomial distribution. The state update density is specified according to an underlying process for the second order difference of $\log \lambda_t$, which we denote by δ_t. For $t > t_0$. The model is

$$
\begin{aligned}
p(s_t | s_{t-1}, \beta) &= p(\delta_t | \delta_{t-1}, \rho, \alpha_0, \alpha_1) \\
\delta_t &\coloneqq \nabla^2 \log \lambda_t = (\log \lambda_t - \log \lambda_{t-1}) - (\log \lambda_{t-1} - \log \lambda_{t-2}) \\
\delta_t &= \rho\, \delta_{t-1} + \sqrt{\alpha_0 + \alpha_1\, \delta_{t-1}^2}\, \epsilon_t \\
\epsilon_t &\overset{\text{iid}}{\sim} \text{Normal}(0, 1).
\end{aligned} \tag{2.4}
$$

Under the model, $\delta_t = \nabla^2 \log \lambda_t$ is assumed to follow an AR(1)-ARCH(1) process. This parametric form of the state equation accommodates our beliefs about how second order differences drive the dynamics of the observed daily counts. Note that since λ_t must be positive, it is natural to work with

the logarithm of the underlying mean, $\log \lambda_t$. In a wave of COVID-19 death counts, we expect the δ_t's to have high lag 1 autocorrelation and settle toward zero following the region of high curvature around the peak. Our state equation can capture this behavior—if $\rho \in (0, 1)$ and $\alpha_1 > 0$, we have δ_t positively autocorrelated, and $\lim_{t\to\infty} \mathbb{E}[\delta_t|\delta_s] = 0$. Moreover, δ_t is more concentrated around 0 when $|\delta_{t-1}|$ is small.

The δ_t update depends on δ_{t-1} whenever at least one of ρ or α_1 is nonzero. In this case, $t_0 = 3$, and for $t \geq 3$, we explicitly define s_t as the vector of length three containing $\log \lambda_t$, the first order difference $\log \lambda_t - \log \lambda_{t-1}$, and the second order difference δ_t. Because of the twice-differencing, three dimensions are necessary to satisfy the second and third lines of (2.1). If ρ and α_1 are both zero, then δ_t is white noise, $t_0 = 2$, and s_t need only contain $\log \lambda_t$ and $\log \lambda_t - \log \lambda_{t-1}$.

The update step (2.4) for δ_t has the general form of a first order autoregressive process with ARCH(1) errors. Fixing the AR(1) and ARCH(1) terms (ρ and α_1, respectively) gives rise to boundary cases and simpler models: $\alpha_1 = 0$ gives iid noise; $\rho = 1$ gives a random walk. We consider a sequence of models that vary in their treatment of ρ and α_1, and grow in complexity. Table 2.1 shows the progression of models. Note that when α_1 is set to zero, we reparameterize with $\sigma = \sqrt{\alpha_0}$ for notational ease.

Table 2.1: Progression of models fit to New York City daily COVID-19 deaths

Model	Observation dist.	δ_t update†	Parameters β to fit	t_0 ‡
P(0,0)	Poisson(λ_t)	$\sigma\,\epsilon_t$	σ	2
P(1,0)	Poisson(λ_t)	$\delta_{t-1} + \sigma\,\epsilon_t$	σ	3
P(ρ,0)	Poisson(λ_t)	$\rho\,\delta_{t-1} + \sigma\,\epsilon_t$	ρ, σ	3
P(ρ,α_1)	Poisson(λ_t)	$\rho\,\delta_{t-1} + \sqrt{\alpha_0 + \alpha_1\,\delta_{t-1}^2}\,\epsilon_t$	ρ, α_0, α_1	3
NB(ρ,α_1)	NegBinom(λ_t, ϕ)	$\rho\,\delta_{t-1} + \sqrt{\alpha_0 + \alpha_1\,\delta_{t-1}^2}\,\epsilon_t$	$\phi, \rho, \alpha_0, \alpha_1$	3
	† $\epsilon_t \overset{\text{iid}}{\sim}$ Normal(0, 1)	‡ t_0 number initialized states		

In our simplest model, P(0,0), we fix ρ and α_1 equal to zero, giving the white noise process for δ_t discussed above. Keeping $\alpha_1 = 0$, we next consider a random walk for δ_t, setting $\rho = 1$ in P(1,0). In P(ρ,0) we introduce a prior on ρ with support $[0, 1]$. Conditionally, δ_t now follows

a stationary AR(1) process. In P(ρ,α_1), we set priors for both ρ and α_1, so that conditionally δ_t follows a stationary AR(1)-ARCH(1) process. This is the most complicated state equation we consider. We take this one step further in the NB(ρ,α_1) case by introducing a negative binomial observation distribution with scale parameter ϕ, parameterized so that it has mean λ_t and variance $\lambda_t + \lambda_t^2/\phi$.

Each model in Table 2.1 has associated priors for the β parameters and the initial states $s_1, ..., s_{t_0}$. In Appendix 2.6 we provide details of the full joint distribution for NB(ρ,α_1). With the full joint specified, we can use MCMC methods to fit each model. In Appendix 2.7 we give the stan model code used to fit NB(ρ,α_1).

2.2.3 Model assessment

We perform predictive model assessment in several ways based on posterior predictive draws $y_{T+k} \sim p(y_{T+k}|y_{1:T})$ sampled according to (2.3). These evaluations allow us to meaningfully compare the models in Table 2.1.

We make point and interval forecasts by taking quantiles of posterior predictive draws. We produce forecasts and compare to observed data. As a summary measure of predictive performance, we consider the absolute relative error of 7-day ahead cumulative death predictions

$$\frac{\left|\text{Median}\left(\sum_{k=1}^{7} y_{T+k}|y_{1:T}\right) - \sum_{k=1}^{7} y_{T+k}\right|}{\sum_{k=1}^{7} y_{T+k}}. \tag{2.5}$$

The other predictive model assessments we perform evaluate model fit using the full set of posterior predictive draws.

Probability integral transform

The probability integral transform (PIT) is a useful tool generally used for assessing models for continuous data; however, it can be extended to count-valued data. We use the nonrandomized PIT histogram for count data proposed in [39] for assessing the fits of our considered models.

We fit our model to observed counts $y_{1:T}$, and make a k-day ahead prediction by sampling posterior predictive draws, $y_{T+k}^{(1)}, ..., y_{T+k}^{(B)}$ as in (2.3). We seek a probability integral transform to assess this empirical predictive distribution against future observations $y_{T+k} \sim F_{T+k}$. The empirical predictive distribution function $\tilde{F}_{B,T+k}$ is given by

$$\tilde{F}_{B,T+k}(k) = \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}_{\{y_{T+k}^{(b)} \leq k\}}. \tag{2.6}$$

If we successfully target the true predictive distribution, so that $y_{T+k}^{(1)}, ..., y_{T+k}^{(B)} \overset{\text{iid}}{\sim} F_{T+k}$, then $\lim_{B\to\infty} \tilde{F}_B(y_{T+k}) = F_{T+k}(y_{T+k})$, and it follows that the PIT using $\tilde{F}_B(y_{T+k})$ will be very close to Uniform on the unit interval for B large. For each model fit, we make $B = 4000$ post-warmup posterior draws.

Scoring rules

Scoring rules are another tool for predictive model assessment, providing a numerical score $s(P, y)$ based on a predictive distribution P and an observation y. We consider two scoring rules, the logarithm score and the ranked probability score. These scoring rules are strictly proper [39]. We average the scores over sets of look-ahead predictive distributions to compare the considered models.

The logarithm score is defined as $\log(P, y) = -\log p(y)$, where $p(x)$ is the probability of observing y under the predictive distribution P. We adapt this score to our framework for a k-day ahead forecast as $\log(P_{B,T+k}, y_{T+k}) = -\log \tilde{p}_{B,T+k}(y_{T+k})$, where

$$\tilde{p}_{B,T+k}(k) = \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}_{\{y_{T+k}^{(b)} = k\}} + \frac{1}{B^2}.$$

is an approximation of the true posterior predictive probability mass function. This is similar to (2.6) above. We add the $\frac{1}{B^2}$ term to avoid $\log(P_{B,T+k}, y_{T+k}) = -\infty$ when $y_{T+k}^{(b)} \neq y_{T+k}$ for all $b \in \{1, ..., B\}$.

The ranked probability score is defined as $\text{rps}(P, y) = \sum_{k=0}^{\infty}[P(k) - \mathbb{1}_{\{y \leq k\}}]^2$, where $P(\cdot)$ is the cumulative distribution function under P. We take $P(k) = \tilde{F}_B(y_{T+k})$, from (2.6). For both scoring rules, smaller values indicate a better fit.

2.3 Results

We fit the models in Table 2.1 to multiple subsets of the New York City data. Each daily count time series corresponds to a single borough and begins with the day of the first COVID-19 death observed in that borough. Model assessment is based on a collection of these fits and their forecasts. Comparing results across the considered models, we determine NB(ρ,α_1) to be generally best at capturing the observed dynamics in New York City's daily COVID-19 death counts over various ranges of the data. We further fit the NB(ρ,α_1) model to daily COVID-19 deaths data from the four most populous counties in Texas.

2.3.1 Model Comparison

Figures 2.2 and 2.3 illustrate how the progression of state equation specifications (δ_t updates) capture the dynamics of observed daily deaths around their peak. Here the models from Table 2.1 are fit to six ranges of Queens data. Figure 2.2 shows posterior estimates of the underlying time-varying mean λ_t, along with observed deaths, while Figure 2.3 shows posterior estimates of δ_t.

Each subfigure corresponds to a different model fit. The rows align with Table 2.1 and each column corresponds to a different training series, where T is the number of training points. We indicate where the final training day falls relative to the observed peak. For the top left subfigure, $T = 25$ days of deaths are included in the training series, starting from the first confirmed COVID-19 death in Queens on March 11 and ending with the count from April 4, two days before the observed peak. We plot approximate 90% posterior intervals and approximate posterior medians based on Monte Carlo sampling. The red shaded regions show three weeks of out-of-sample, look-ahead forecasts.

The first row of Figures 2.2 and 2.3 show estimates based on P(0,0). Under P(0,0), δ_t is

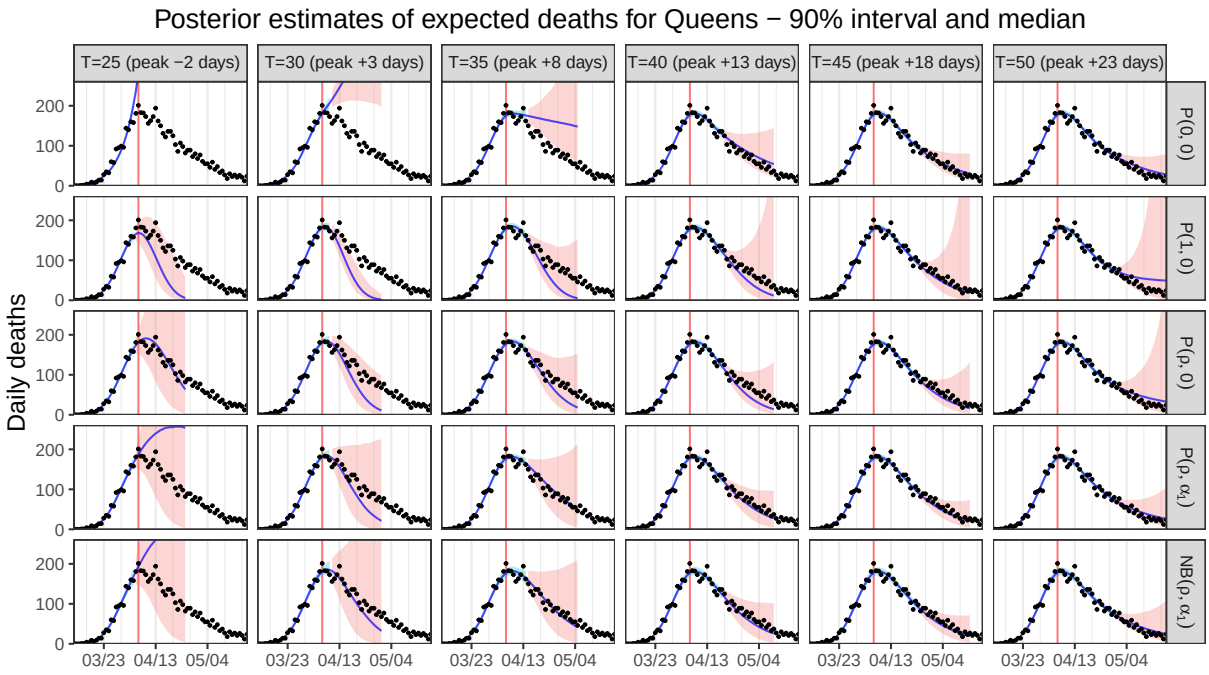

Figure 2.2: Daily COVID-19 deaths in Queens along with estimates of underlying means λ_t. Red shaded regions correspond to out-of-sample forecasts. Estimates vary by model and length of training series (T).

uncorrelated with δ_{t-1} and $\mathbb{E}[\delta_t|\delta_{t-1}] = 0$. These model features lead to poor forecasting around the observed peak, where second order differences are negative with lag 1 autocorrelation. While P(0,0) fails to capture the dynamics close to the peak, this model forecasts well when the training series extends at least 18 days past the peak.

Under P(1,0), we have that $\mathbb{E}[\delta_{T+k}|y_{1:T}] = \mathbb{E}[\delta_T|y_{1:T}]$ for $k \geq 1$. This feature is seen clearly in Figure 2.3, where the P(1,0) posterior predictive distributions for δ_{T+k} remains centered around the posterior mean of δ_T. In the first four columns, forecasts for δ_t stay negative, leading to a too quick decrease in the corresponding λ_t forecasts.

Under P(ρ,0) forecasts of δ_t increase gradually towards zero. This captures the dynamics of the observed counts much more closely than the previous two models. However, while predictive distributions for δ_{T+k} become centered around zero, there is still much uncertainty, and this seems to exaggerate the possibility of δ_t moving away from zero. Inclusion of the ARCH(1) term corrects

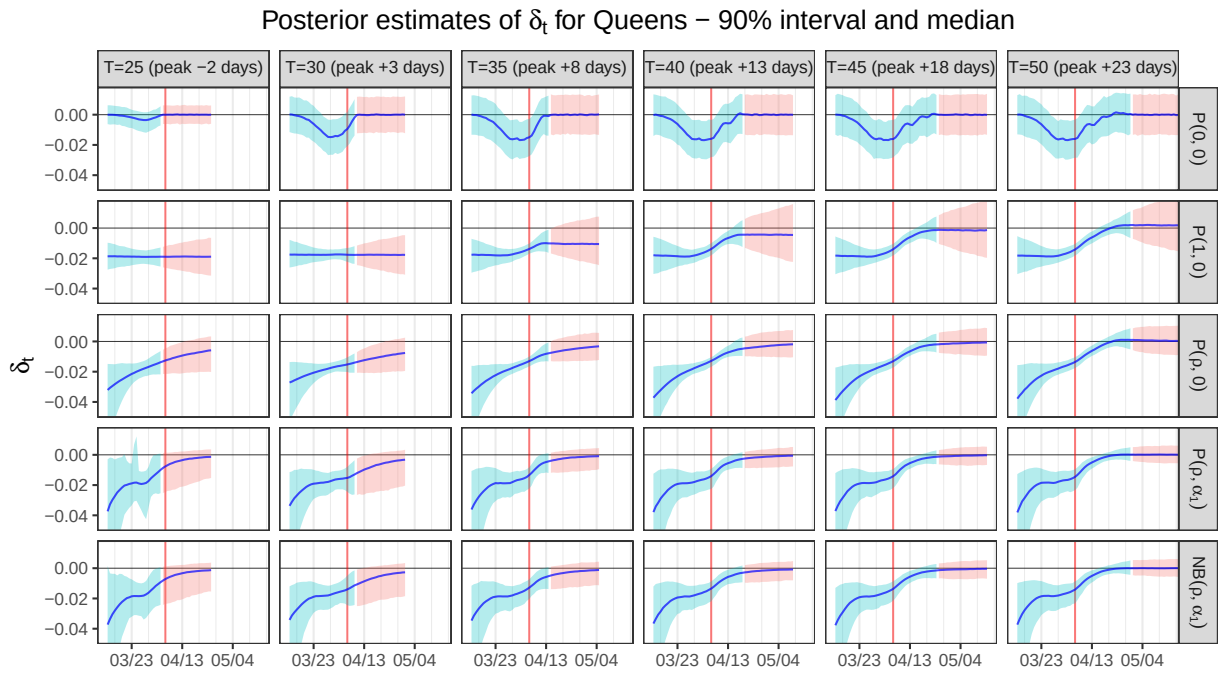

Figure 2.3: Estimates of δ_t, the second order difference of the logarithm of the underlying mean. Red shaded regions correspond to out-of-sample forecasts. Estimates vary by model and length of training series (T).

for this. In the P(ρ,α_1) and NB(ρ,α_1) models, there is more uncertainty in the second and third columns where δ_T is negative, and less in the fifth and sixth where δ_T is close to zero. The final two rows in Figures 2.2 and 2.3, corresponding to models P(ρ,α_1) and NB(ρ,α_1), are very similar. This state update model is not able to properly predict the peak in the far left column, but performs well in the period after the peak.

Table 2.2: Mean absolute relative error of 7-day ahead cumulative predictions

	Absolute Relative Error (SE)			
Model	peak–3 to peak+3†	peak+4 to peak+10†	peak+11 to peak+17†	peak+18 to peak+24†
P(0,0)	.915 (.083)	.440 (.086)	.165 (.041)	**.113** (.015)
P(1,0)	**.146** (.018)	.243 (.018)	.202 (.020)	.189 (.031)
P(ρ,0)	.208 (.042)	.177 (.015)	.145 (.013)	.159 (.027)
P(ρ,α_1)	.320 (.053)	.175 (.019)	**.099** (.010)	.139 (.023)
NB(ρ,α_1)	.357 (.058)	**.168** (.022)	**.099** (.010)	.138 (.023)
	† peak-relative week in which the last training day falls.			

Table 2.2 shows mean absolute relative errors (2.5) along with their standard errors. We average over the five NYC boroughs as well as the week-long range based on where the the final training day falls relative to the observed peak. Each cell summarizes $5 \times 7 = 35$ values.

The P(1,0) model has the lowest absolute relative error when the final training day falls within 3 days of the observed peak. This follows since absolute errors are bounded for underestimates, and the forecasts from P(1,0) decrease quickly to zero. The NB(ρ,α_1) model has the lowest error when the final training day falls 4 to 17 days after observed peak, while the P(0,0) model performs best later on.

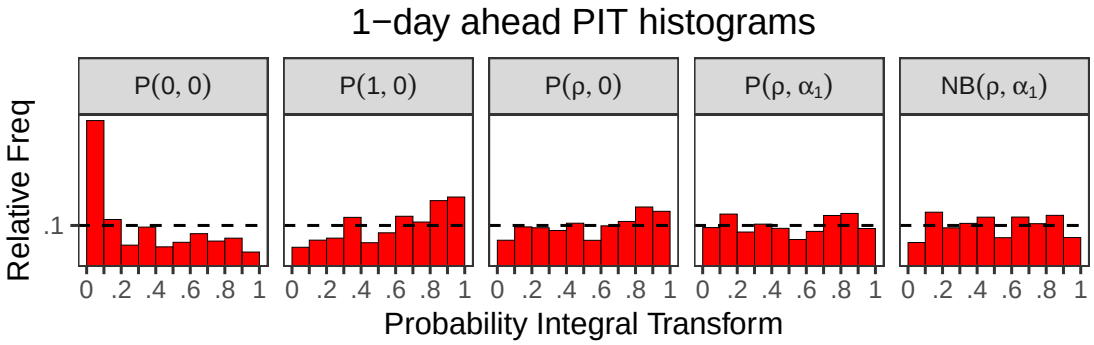

Figure 2.4: Nonrandomized PIT histograms for 1-day ahead predictive distributions.

Figure 2.4 compares the nonrandomized PIT histograms (Section 2.2.3) for 1-day ahead predictions. Each histogram is based on $5 \times 28 = 140$ model fits, with the 28 training series from each borough covering the same four peak-relative weeks in Table 2.2. The histograms corresponding to the P(0,0), P(1,0) and P(ρ,0) appear skewed. While we see some overdispersion for NB(ρ,α_1), the PITs for P(ρ,α_1) and NB(ρ,α_1) do not provide strong evidence that these models are incorrect.

Table 2.3 shows two proper scoring rules (Section 2.2.3) for the progression of models considered. These scores are averaged over the 21 days of forecasting for each model fit, the peak-relative week of the final training day, and the five boroughs. Each entry is an average of $21 \times 7 \times 5 = 735$ values.

According to these measures for predictive model assessment, the models with both the AR(1) and ARCH(1) terms dominate when the final training day is 4 to 10 days after the peak, with

Table 2.3: Scoring rules evaluated on the predictive distributions of considered models

	Ranked probability score			
Model	peak–3 to peak+3†	peak+4 to peak+10†	peak+11 to peak+17†	peak+18 to peak+24†
P(0,0)	350.88	26.41	**5.16**	3.59
P(1,0)	24.73	21.11	12.36	8.23
P(ρ,0)	**24.23**	14.73	7.55	4.58
P(ρ,α_1)	46.60	11.79	5.30	3.62
NB(ρ,α_1)	52.22	**10.98**	5.34	**3.58**
	Logarithm score			
Model	peak–3 to peak+3†	peak+4 to peak+10†	peak+11 to peak+17†	peak+18 to peak+24†
P(0,0)	13.36	6.61	**3.56**	**3.17**
P(1,0)	6.04	6.14	4.47	3.82
P(ρ,0)	**4.98**	4.90	4.00	3.45
P(ρ,α_1)	5.42	4.41	3.59	3.24
NB(ρ,α_1)	5.46	**4.39**	3.62	3.25
	† peak-relative week in which the last training day falls.			

NB(ρ,α_1) performing the better of the two. The P(ρ,α_1) and NB(ρ,α_1) models have logarithm scores close to the lowest for the other peak-relative weeks, and NB(ρ,α_1) has the lowest ranked probability score when the final training day is 18 to 24 days after the peak. We again see how P(0,0) performs well relative to the others in the later weeks, but very poorly when the final training day is close to the peak.

2.3.2 Fits around peak for all five boroughs

Figures 2.5 and 2.6 show further posterior estimates from NB(ρ,α_1). We fit the model to counts from all five New York City boroughs, with final training days concentrated around the two weeks following the observed peaks. Again we plot approximate 90% posterior intervals and approximate posterior medians for λ_t and δ_t, which are based on posterior samples.

The trajectories of the posterior estimates of δ_t show a similar pattern across the five boroughs—an increase towards zero with less posterior predictive uncertainty as more training points are included. For Brooklyn, Queens and The Bronx there is a temporary flattening out or even decrease in δ_t around two weeks before the peak. For Brooklyn and The Bronx, the peak of the posterior

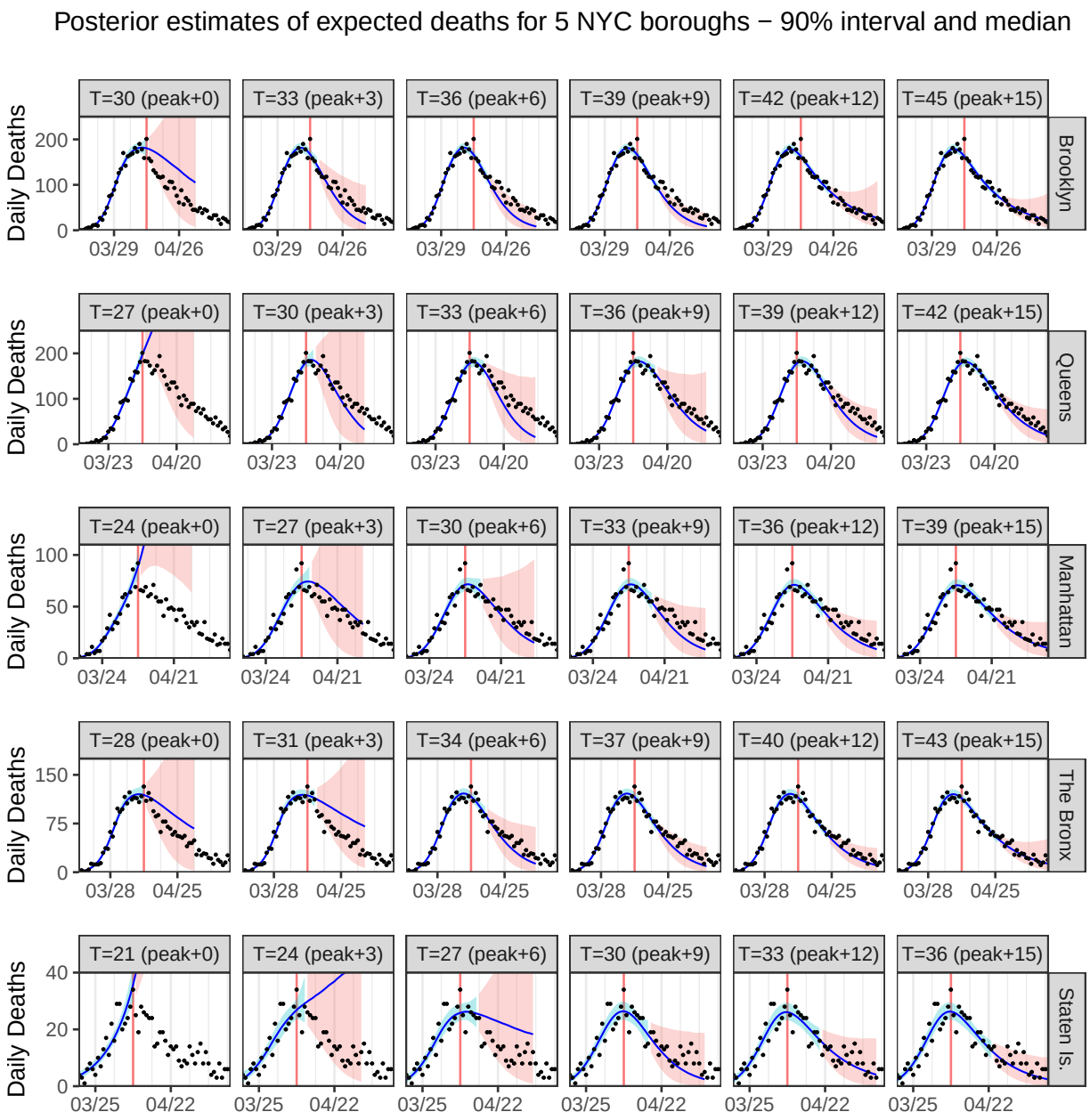

Figure 2.5: Estimates of λ_t from collection of NB(ρ,α_1) model fits. The red shaded regions show out-of-sample posterior predictive forecasting. The points are observed actual deaths.

λ_t estimates falls before the observed peak, while for Queens there appears to be a sort of double observed peak, and the maximum posterior λ_t is closer to the first.

In the far left hand column in Figure 2.5, we see a failure to recognize that the peak has been reached for Queens, Manhattan and Staten Island. Here the model incorrectly forecasts dramatic

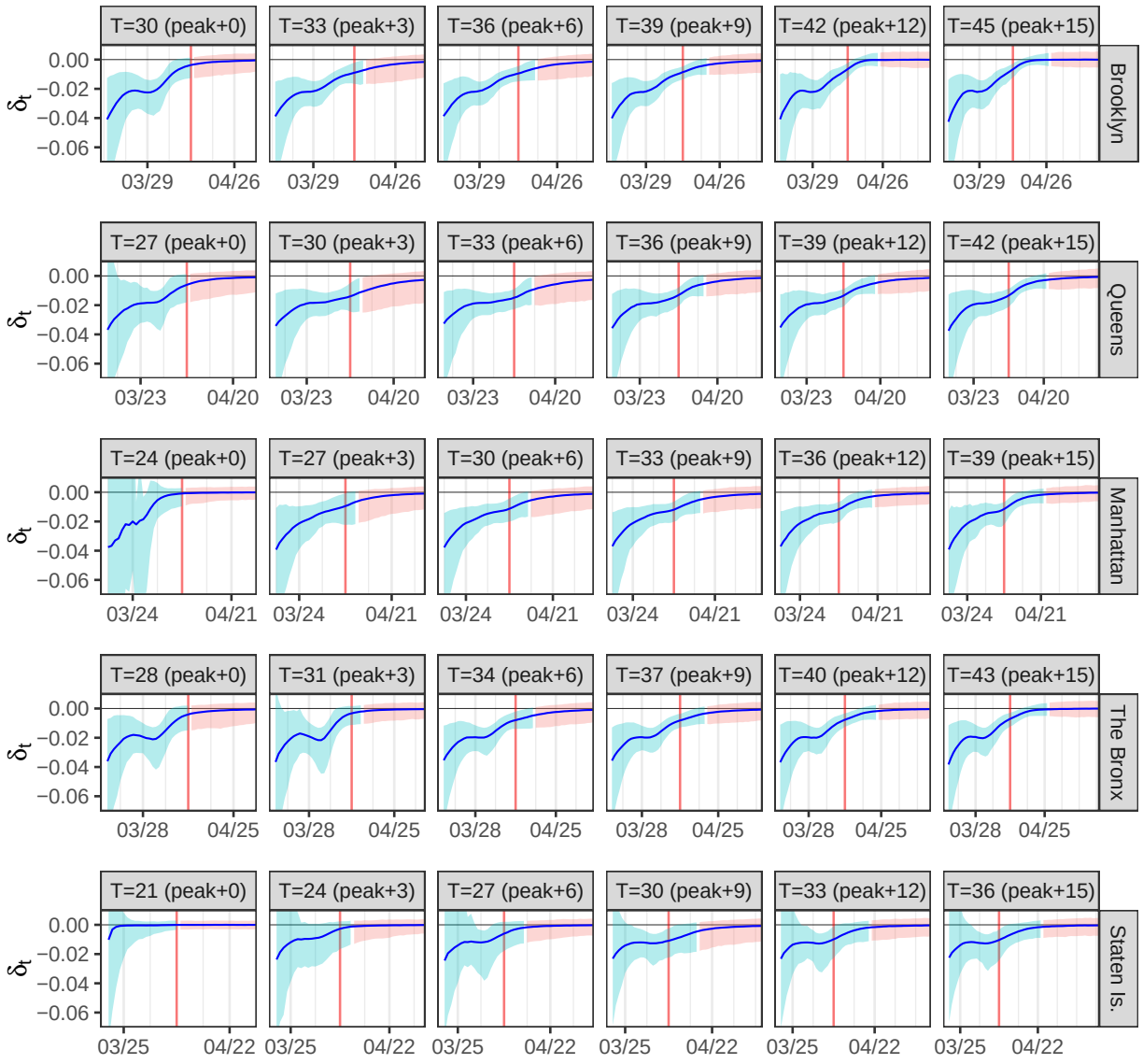

Figure 2.6: Estimates of δ_t from collection of NB(ρ,α_1) model fits. The red shaded regions show out-of-sample posterior predictive forecasting.

continued growth in the death rates for these boroughs. As more training points are added, the posterior predictive forecasts show decreasing mean processes that are better aligned with the actual deaths and have less posterior uncertainty.

Across all the boroughs, the NB(ρ,α_1) model appears to perform reasonably well in capturing

the dynamics of the observed daily counts once it has been fit with data that extends a few days past the observed peak. The fits improve as more data is included.

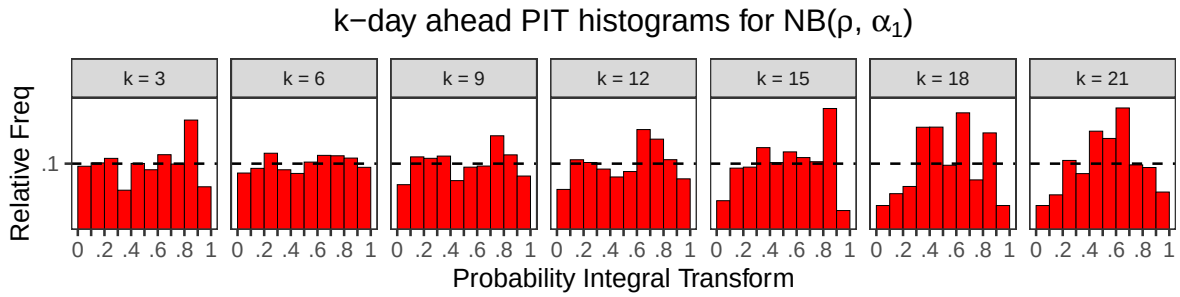

Figure 2.7: PIT histograms for k-day ahead predictive distributions, for the NB(ρ,α_1) model.

Figure 2.7 shows nonrandomized PIT histograms for k-day ahead predictions for the NB(ρ,α_1) model. The fits appear to break down around $k = 12$, with the predictive distributions becoming more and more overdispersed with increasing k.

2.3.3 Fits to four Texas counties

In further exploration, we fit the NB(ρ,α_1) model to COVID-19 mortality data from the four most populous counties in Texas [40]. These data come from the state, are subject to ongoing quality checks and updates, and include fatalities for which COVID-19 is listed as a direct cause of death on the death certificate. The four largest Texas counties are Harris County, which includes the city of Houston, Dallas and Tarrant counties, which include most of the Dallas–Fort Worth metroplex population, and Bexar County, which includes the city of San Antonio. We fit the NB(ρ,α_1) model to one long series for each county, beginning with its first recorded COVID-19 death, to the death count on December 20, 2020. The training data consists of the provisional counts available on January 12, 2021.

Figure 2.8 shows the posterior estimates from these four fits. Over these longer training series, we observe multiple dynamic phases, including pronounced peaks in mid-to-late July and waves building at the end of the year. We can see how the dynamics of the estimated δ_t's on the right are reflected in the observed training counts and estimated λ_t's on the left. The geographically adjacent

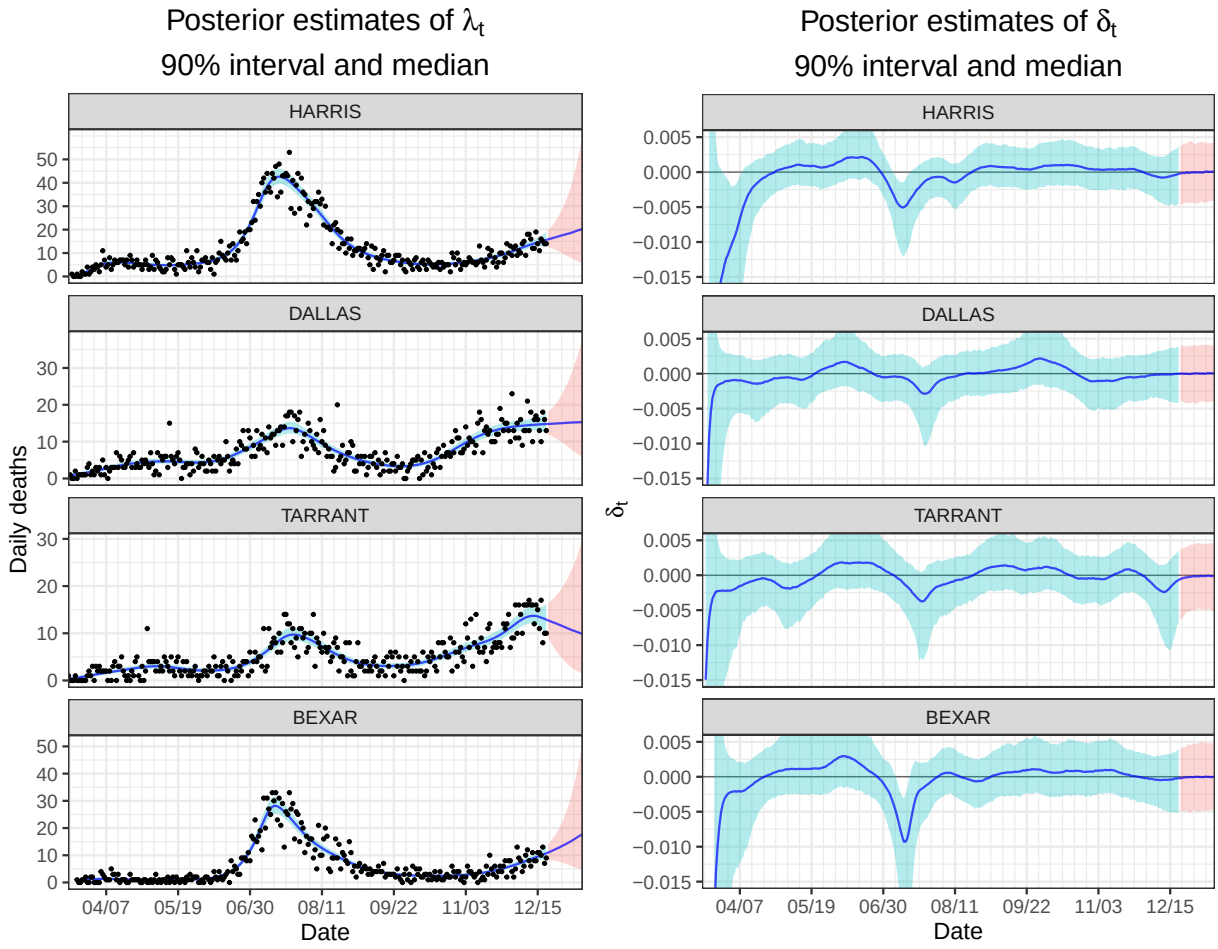

Figure 2.8: Posterior estimates of λ_t and δ_t from NB(ρ,α_1) fits to fatality data from the four largest Texas counties.

Dallas and Tarrant counties show similar patterns of a more gradual, less devastating first wave, while Harris and Bexar Counties have dramatic and pronounced peaks. According to the model forecasts, the late fall/winter peaks may not yet be reached.

2.4 Conclusion

We have presented a generalized state-space model for count-valued time series that seeks to capture the dynamics of local daily COVID-19 deaths. The model components, in particular the latent state process involving second order differencing and an AR(1)-ARCH(1) model, are motivated by supposed behavior of waves of epidemic counts. The results we see in applying our

proposed model justify these components.

2.5 Chapter 2 Appendix

2.6 Full joint distribution

In Section 2.2 we describe the salient features of our modeling approach. Here we provide details of the full joint distribution of the NB(ρ,α_1) model.

For $t \geq 3$, we define the state s_t as the three vector containing $\log \lambda_t$ along with the first and second differences of $\log \lambda_t$. The first two states s_1 and s_2 need only contain initialized variables $\log \lambda_1$ and $\log \lambda_2$. The parameters β are $(\phi, \rho, \alpha_0, \alpha_1)$, as in Table 2.1.

The negative binomial observation distribution gives

$$p(y_t|s_{1:t}, y_{1:t-1}, \beta) = p(y_t|\lambda_t, \phi) = \frac{\Gamma(y_t + \phi)}{y_t!\,\Gamma(\phi)} \left(\frac{\lambda_t}{\lambda_t + \phi}\right)^{y_t} \left(\frac{\phi}{y_t + \phi}\right)^{\phi}.$$

This satisfies conditional independence in the observation equation [38]. For $t \geq 4$, it is clear by construction that s_t is a function of s_{t-1}, β and ϵ_t, so that s_t is conditionally independent of $s_{1:t-2}$ and $y_{1:t}$, given s_{t-1} and β. For the state update density, we have

$$p(s_t|s_{1:t-1}, y_{1:t}, \beta) = p(\delta_t|\delta_{t-1}, \rho, \alpha_0, \alpha_1) = \varphi\left(\frac{\delta_t - \rho\delta_{t-1}}{\sqrt{\alpha_0 + \alpha_1\delta_{t-1}^2}}\right),$$

where $\varphi(\cdot)$ is the standard normal density. Thus conditional independence in the state equation also holds. The complete joint distribution of the model is given by:

$$p(y_{1:T}, s_{1:T}, \beta) = p(\beta)p(s_1, s_2, s_3) \prod_{t=4}^{T} p(\delta_t|\delta_{t-1}, \rho, \alpha_0, \alpha_1) \prod_{t=1}^{T} p(y_t|\lambda_t, \phi).$$

What remains is to specify the prior distributions on the initial states and parameters β. For the

initial states, we assume $p(s_1, s_2, s_3) = p(\log \lambda_1)p(\log \lambda_2|\lambda_1)p(\log \lambda_3|\lambda_1, \lambda_2)$, where

$$\begin{aligned}
\log \lambda_1 &\sim \text{Normal}\,(0, 5^2) \\
\log \lambda_2|\lambda_1 &\sim \text{Normal}\,(\log \lambda_1, 1^2) \\
\log \lambda_3|\lambda_1, \lambda_2 &\sim \text{Normal}\,(\log \lambda_2 + \nabla \log \lambda_2, .05^2).
\end{aligned}$$

The prior distributions for the parameters β are given by:

$$\begin{aligned}
\phi &\sim \text{Half Normal}\,(0, 200^2) \\
\rho &= \frac{r_0}{2} \times \left(\sqrt{\tan^2 \theta + 4} - \tan \theta\right) \\
\alpha_1 &= \rho \times \tan \theta \\
\theta &\sim \text{Uniform}\,\left(0, \frac{\pi}{2}\right) \\
r_0 &\sim \text{Beta}\,(2, 1) \\
p(\alpha_0) &\propto I(\alpha_0 \geq 0) \times \exp\left\{-\frac{1}{2}\left(\frac{\alpha_0 - .001^2}{.001^2}\right)^2\right\},
\end{aligned}$$

where ϕ, θ, r_0 and α_0 are all independent.

The priors for ϕ and α_0 are restricted normal distributions, with only positive support. These are both strong, informative priors. The Half Normal distribution prevents the Negative Binomial scale parameter ϕ from jumping to large values, which helps convergence in stan. Note that as ϕ increases, the conditional observation distribution approaches Poisson. Since we separately consider and fit the P(ρ,α_1) model, this restrictive prior does not limit our exploration.

For $\alpha_0 = \text{Var}(\delta_t|\delta_{t-1} = 0)$, setting the location and scale of the restricted normal to $.001^2$ places the prior mode away from zero, yet forces α_0 to be close to zero. This helps convergence in stan. Moreover, the former effect helps prevent us getting stuck at $\delta_t = 0$, while the latter amounts to model regularization—restricting changes in the second order difference δ_t ensures smoothness of mean process, and prevents λ_t from following the observed counts too closely.

The joint prior $p(\rho, \alpha_1)$ has support on the region $D \coloneqq \{(\rho, \alpha) \in [0, 1]^2 : \rho^2 + \alpha_1 < 1\}$.

It follows that draws of ρ and α_1 will satisfy sufficient stationarity conditions for autoregressive processes with ARCH(1) errors, as considered in [41]. While α_1 must be non-negative, we restrict $\rho > 0$ since a negative autoregressive term does not make sense in our model.

We construct $p(\rho, \alpha_1)$ so that it is close to uniform on D. A prior draw (ρ, α) is a transformation of a uniform draw on the unit quarter disk (θ, r_0), given in polar coordinates. Note that $p(\rho, \alpha_1)$ and D are given in the Cartesian coordinate system. To transform (θ, r_0) to a point in D, we preserve the polar angle θ while contracting the polar radius. In particular, we scale r_0 by $r_D(\theta)$, where $r_D(\theta)$ is the distance from the origin to the parabola $x^2 + y = 1$, along θ. Converting between coordinate systems, we have

$$r_D(\theta) \sin\theta = 1 - r_D(\theta)^2 \cos\theta.$$

This is a quadratic equation with one positive solution

$$r_D(\theta) = \frac{-\sin(\theta) + \sqrt{1 + 3\cos^2\theta}}{2\cos^2\theta}.$$

Finally, we convert the transformed draw $(\theta, r_0 \times r_D(\theta))$ to Cartesian coordinates to get

$$\rho = \frac{r_0}{2}\left(\sqrt{\tan^2\theta + 4} - \tan\theta\right)$$

and

$$\alpha_1 = \rho\ \tan\theta,$$

as above.

2.7 stan model code

Here is the stan code for the NB(ρ,α_1) model, with priors set up as described in Appendix 1.

```
data {
    // Length of training time series
  int<lower=0> N;
    // How many look-ahead predictions
  int<lower=0> N_pred;
    // Training data time series of counts
  int counts[N];
}
transformed data {
    // Constant for constraining theta
  real halfPi = pi() / 2;
}
parameters {
  real<lower=0> phi;
  real<lower=0> alpha0;
    // Params for draw on unit quarter disk
    // to be transformed into rho and alpha1
  real<lower=0> theta_raw;
  real<lower=0, upper=1> r;
    // Initialization of state space variables
  real mu_init_raw;
  real gamma_init;
  real delta_init;
    // 'Innovations'
  vector[N - 3] epsilon;
}
```

```
transformed parameters {
    // Log mean param
  vector[N] mu;
    // Differenced log mean param
  vector[N - 1] gamma;
    // Twice differenced log mean param
  vector[N - 2] delta;
  real<lower=0, upper=halfPi> theta;
  real<lower=0, upper=1> rho;
  real<lower=0, upper=1> alpha1;
    // Uniform on (0, pi/2)
  theta = exp(-theta_raw) * halfPi;
    // Transformed draw from unit qtr disk to constr region
  rho = .5 * r * (sqrt(square(tan(theta)) + 4) - tan(theta));
  alpha1 = rho * tan(theta);
    // Set initial values
  mu[1] = mu_init_raw * 5;
  gamma[1] = gamma_init;
  delta[1] = delta_init;
    // Recursive state model
  for(t in 2:N) {
      // Local level update
    mu[t] = mu[t - 1] + gamma[t - 1];
    if (t < N)
        // First order trend update
      gamma[t] = gamma[t - 1] + delta[t - 1];
    if (t < (N - 1))
```

```
        // Second order trend update with AR1-ARCH1 model
      delta[t] = rho * delta[t - 1] +
                 sqrt(.000001 * alpha0 +
                 alpha1 * square(delta[t - 1])) *
                    epsilon[t - 1];
  }
}
    // The model to be estimated
model {
  mu_init_raw ~ normal(0, 1);
  gamma_init ~ normal(0, 1);
  delta_init ~ normal(0, .05);
  epsilon ~ normal(0, 1);
  theta_untransformed ~ exponential(1);
  r ~ beta(2, 1);
    // Scaled in transformed parameters section
  alpha0 ~ normal(1, 1);
  phi ~ normal(0, 200);
  counts ~ neg_binomial_2_log(mu, phi);
}

generated quantities {
  vector<lower=0>[N] lambda;
  int countsPost[N];
  vector<lower=0>[N_pred] lambda_pred;
  vector[N_pred] epsilon_pred;
  vector[N_pred] mu_pred;
```

```
vector[N_pred] gamma_pred;
vector[N_pred] delta_pred;
int countsPost_pred[N_pred];
vector[N_pred] newCountsCum_pred;
lambda = exp(mu);
  // Within-sample posterior predictive draws
for (t in 1:N) {
    // Make sure the log mean param is not too big
  if (mu[t] < 10)
    countsPost[t] = neg_binomial_2_log_rng(mu[t], phi);
  else countsPost[t] = 20000;
}
if (N_pred > 0) {
    // Out-of-sample posterior predictive draws
  for (t in 1:N_pred)
    epsilon_pred[t] = normal_rng(0, 1);
    // 1st predictive state evolves from final fitted state
  delta_pred[1] = rho * delta[N - 2] +
    sqrt(.000001 * alpha0 +
         alpha1 * square(delta[N - 2])) * epsilon_pred[1];
  gamma_pred[1] = gamma[N - 1] + delta_pred[1];
  mu_pred[1] = mu[N] + gamma_pred[1];
    // Recursive predictive states
  for(t in 1:(N_pred - 1)) {
    delta_pred[t + 1] = rho * delta_pred[t] +
      sqrt(.000001 * alpha0 +
        alpha1 * square(delta_pred[t])) * epsilon_pred[t + 1];
```

```
      gamma_pred[t + 1] = gamma_pred[t] + delta_pred[t + 1];
      mu_pred[t + 1] = mu_pred[t] + gamma_pred[t + 1];
    }
    lambda_pred = exp(mu_pred);
      // Simulate look-ahead counts
    for (t in 1:N_pred) {
        // Avoid stan error if blown-up
      if (mu_pred[t] < 10)
        countsPost_pred[t] =
          neg_binomial_2_log_rng(mu_pred[t], phi);
      else countsPost_pred[t] = 20000;
      if (t == 1)
          // Cumulative look-ahead counts
        newCountsCum_pred[t] = countsPost_pred[t];
      else newCountsCum_pred[t] = newCountsCum_pred[t - 1] +
                                  countsPost_pred[t];
    }
  }
}
```

Chapter 3: Inferring latent network edges from noisy event times with a leaky integrate-and-fire (LIF) model

We consider the task of inferring the connections between noisy observations of events. In our model-based approach, we consider a generative process incorporating latent dynamics that are directed by past events and the unobserved network structure. This process is based on a leaky integrate-and-fire (LIF) model from neuroscience for aggregating input and triggering events (spikes) in neural populations. Given observation data we estimate the model parameters with a novel variational Bayesian approach, specifying a highly structured and parsimonious approximation for the conditional posterior distribution of the process's latent dynamics. This approach allows for fully interpretable inference of both the model parameters of interest and the variational parameters. Moreover, it is computationally efficient in scenarios when the observed event times are not too sparse. We apply our methods in a simulation study and to recorded neural activity in the dorsomedial frontal cortex (DMFC) of a rhesus macaque. We assess our results based on ground truth, model diagnostics, and spike prediction for held-out nodes.

3.1 Introduction

Let us assume we simultaneously observe the discretized times of n random events in a multivariate point process. Consider further that these events correspond to nodes in a network with unobserved edges that influence the observable process. We consider the problem of estimating the latent network connections between these nodes given their noisy emissions.

Many complex processes that we are interested in analyzing can be viewed as dynamical systems with interactions on an underlying network. Direct observation of the implicit network may be costly and difficult, even impossible. At the same time, random event data from the system may be

abundant and easy to collect. The challenge is to learn the network based on these 'noisy emissions' in a principled and effective way. In many disciplines, learning the relational structure in such multi-dimensional and partially observed systems is an important problem.

Connectivity inference from neural recording data is one important and well-studied example [42]. Cells in the nervous system form communication networks, firing action potentials that transmit information to other neurons. Electrophysiological recordings are able to partially capture neural population activity, and discrete spiking event times may be attributed among the recorded cells (spike sorting). Cell membrane voltages and the edges between neurons, however, are generally not observed.

Inferring relational structures that map to the observable data enriches our understanding of the dynamical system. With an estimated implicit network we may apply tools from network analysis [43], for example node centrality measures, community detection algorithms, and network embeddings. With inferred edges we can also access powerful machine learning methods of graph neural networks (GNNs), to gain further insight into how the objects in an observed complex system interact [44].

Current research on deep learning frameworks for complex processes offer model-free, data driven approaches for extracting latent network structure and learning underlying dynamics [45]. However, these methods assume direct measurements of nodal states. In the situation we consider, dynamic nodal states are unobserved. All we see are sparse, noisy event time 'emissions'. Without any knowledge or assumptions for how the latent network relates to the observed data, the task is nearly impossible.

In this work, we consider the problem while assuming a generative process for the discrete observations. If our only goal was to predict future dynamics, there are 'black box' tools we could use to make predictions without any interpretable inferences about the underlying structure of the data. When the goal is to understand structure, explain patterns, and ultimately make scientific discoveries, we need to take a take a more 'white box' approach. We choose to use an explainable model that allows us to perform fully interpretable, model-based inference.

The generative process we consider incorporates a soft-threshold leaky integrate-and-fire (ST-LIF) model from neuroscience specifying how events arise and how their effects transmit to other nodes. Latent dynamics are driven by past observations and the unobserved network structure.

We assume the observed event times are recorded (and discretized) in a multivariate discrete time series. Given observed data in this form we fit the generative process using variational Bayesian methods, obtaining estimates of the unknown parameters of interest. The key inference challenge is accounting for all unknown and latent model variables, including the high dimensional, evolving latent states which we are not directly interested in estimating. Our main contribution in this work is our innovative variational Bayesian approach to this problem. We propose a highly structured and parsimonious variational approximation for the conditional posterior distribution of the latent states that is carefully designed to fit the considered model, has interpretable parameters, and offers attractive computational qualities including the obviation of noisy gradient calculations.

Our work in this chapter is arranged as follows. In section 3.2 we set up the considered problem, data and model. Section 3.2.1 introduces the leaky integrate-and-fire (LIF) model, a continuous-time point process model from neuronal dynamics, section 3.2.2 presents our considered data generating process for discrete event times, and section 3.2.3 discusses the model's joint distribution.

In section 3.3 we describe our methods for model inference and statistical analysis. Sections 3.3.1, 3.3.2 and 3.3.3 present our variational Bayes approach and our approximation for the conditional posterior. Sections 3.3.4 and 3.3.5 provide the details of the stochastic gradient update steps for computing our variational approximation, and section 3.3.6 discusses our method's computational complexity. Section 3.3.7 presents our approach for inferring the existence of edges in the latent network. Section 3.3.8 discusses reconstructive prediction of the activity of a held-out node and establishes a forward-backward estimate of a posterior predictive conditional spike probability. In section 3.3.9 we discuss a signal-to-noise ratio for network effects in our considered model and show its computation for individual nodes.

We apply our methods in a simulation study (section 3.4) and to recorded neural activity from the dorsomedial frontal cortex (DMFC) of a rhesus macaque (section 3.5). In section 3.6 we discuss

our methods and approach in the context of the related literature on network inference, multivariate point processes and neuronal dynamics. Section 3.7 outlines a future research direction, and section 3.8 concludes the chapter.

Figure 3.1 provides our chapter 3 'methodological roadmap,' a schematic showing how information (observation data, probabilistic model, estimates, results) passes through and influences our applied workflow. The solid lines represent dependencies—eg observed data and our specified model are needed to perform Bayesian estimation of the latent network, and the resulting estimates, in turn, are needed to perform downstream network analysis. The dashed lines represent influencing relationships—eg the data and its context influences our beliefs about the existence of implicit network and together these affect our choice of assumed generative model.

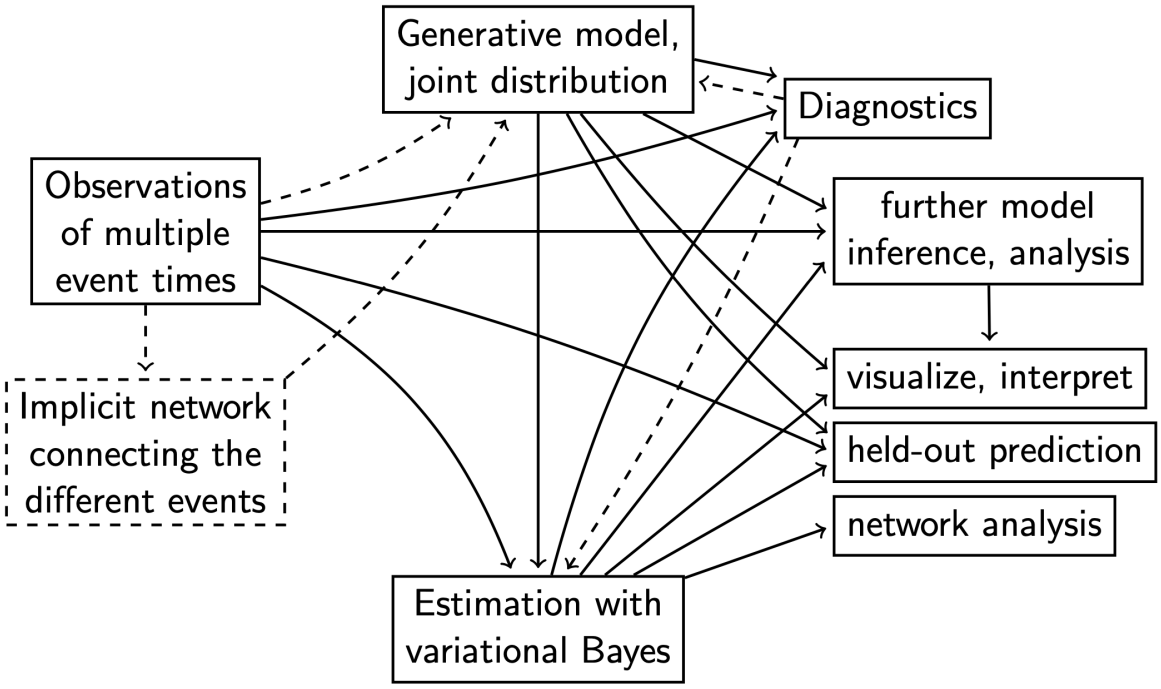

Figure 3.1: Chapter 3 'methodological roadmap' showing our applied workflow.

3.2 Problem set-up and model

In the problem we consider, T observations from a multivariate point process of size n are recorded as discretized event times in a multivariate, binary time series $\mathbf{Y} = (\mathrm{y}_{t,i}) \in \{0\,,1\}^{T\times n}$. The

columns of $\mathbf{Y}$ correspond to nodes in an implicit network $G = (\mathcal{V}, \mathcal{E})$ with signed, weighted and directed edges $\mathcal{E}$ given by the adjacency matrix $\mathbf{W} \in \mathbb{R}^{n \times n}$.

Within this general set up, we assume connections in the latent network G map past history to future observations. That is, for a node $i \in \mathcal{V}$, the probability that we observe it spike at time t is some function of the observed event history before time t of the nodes $j \in \mathcal{V}$ that are connected to i (including node i itself) and the edges $e \in \mathcal{E}$ that make up the directed paths to i in G. Expressing this loosely in mathematical notation, we write

$$\begin{matrix} \{\mathrm{y}_{s,j} : s < t,\ j \text{ and } i \text{ are connected in } G\} \\ \{\mathbf{W}_e : e \in \mathcal{E} \text{ is part of a directed path to } i\} \end{matrix} \overset{\theta, \omega}{\longmapsto} \Pr(\mathrm{y}_{t,i} = 1), \tag{3.1}$$

where θ represents other parameters involved in the mapping, and ω captures potential randomness. In this very general model framework, we can write the probability of the observed data as a likelihood function of the latent adjacency matrix $\mathbf{W}$

$$\mathrm{L}(\mathbf{W}; \mathbf{Y}) = \int_{\Omega} \prod_{t=1}^{T} \prod_{i=1}^{n} \Pr(\mathrm{y}_{t,i} = 1)^{\mathrm{y}_{t,i}} (1 - \Pr(\mathrm{y}_{t,i} = 1))^{1 - \mathrm{y}_{t,i}} \mathrm{dP}(\omega),$$

where $\Pr(\mathrm{y}_{t,i} = 1)$ depends on $\mathbf{Y}_{<t}$, $\mathbf{W}$, θ and ω. We seek to infer the edges $\mathcal{E}$ from $\mathbf{Y}$ by solving

$$\arg\max_{\mathbf{W}} \mathrm{L}(\mathbf{W}; \mathbf{Y}).$$

In order order to proceed any further with this likelihood-based approach, we need a model specifying the details of (3.1).

3.2.1 Leaky integrate-and-fire model

In our approach to this task, we assume a generative process for $\mathbf{Y}$ that incorporates scientific knowledge from neuroscience about the functional form for (3.1) when the observations are the recorded spiking activity of a neural population.

In this section we provide some background information on neuronal dynamics and details of

the adapted neuron model.

Neuroscience research on neuronal dynamics has developed a variety of models for the behavior of neurons [46]. In simple models of neurophysiology, spikes are treated as events that occur at precise moments in time, transmitting information with their presence or absence. With this understanding, the spiking activity of a group of neurons is a mulivariate point process [47]. Electrophysiological recording of neural activity (across multiple neurons simultaneously) allows for (partial) observation of these processes as spike train data.

A spiking neuron in the brain sends its action potential along its axon and through the junctions called synapses to connected cells. The spike is a communication to the postsynaptic cells, and its message is passed, not in electrical signals, but with chemical signals via neurotransmission across the minuscule space between the neurons (the synaptic cleft). The neurotransmitter in a synapse between cells is excitatory or inhibitory, either encouraging the postsynaptic cell to pass the message along with its own spike, or discouraging this further propagation. At an excitatory connection, the arriving spike increases the membrane potential of the postsynaptic cell, while at an inhibitory connection, the spike causes a decrease in the receiving cell's voltage.

In this context our motivating questions becomes: given the observed spike times of a group of simultaneously recorded neurons, what can we say about the existence of synaptic connections between the cells and the types of neurotransmission?

To answer this question with model-based inference, we consider a discrete-time adaptation of one of the simplest models describing the behavior of spiking neurons, the 'leaky integrate-and-fire' (LIF) model, which approximates neuronal dynamics as a summation process for cell membrane voltage together with a mechanism for triggering action potentials (aka spikes).

Let's consider the dynamics of one cell ($i \in \{1, \dots, n\}$) within a neural population. This neuron's current state is described by the time-dependent voltage v_t of its cell membrane, and $t^{(\mathrm{f})}$ are the times of its spikes. In the LIF model a linear differential equation describes the evolution of v_t, and spikes occur at the first passage times $t^{(\mathrm{f})}$ of v_t through a threshold voltage ϑ. After each spike, the membrane potential resets to its resting potential. The model may be expressed with the following

equations:

$$\tau_{\mathrm{m}} \frac{\mathrm{d}\mathrm{v}_t}{\mathrm{d}t} = -\mathrm{v}_t + \mathrm{R}I_t \tag{3.2}$$

$$t^{(\mathrm{f})} = \{t : \mathrm{v}_t = \vartheta\} \tag{3.3}$$

$$\lim_{\substack{\epsilon \to 0 \\ \epsilon > 0}} \mathrm{v}_{t^{(\mathrm{f})}+\epsilon} = 0, \tag{3.4}$$

where the resting potential is zero and I_t is the neuron's time-dependent input current. This model is parameterized by the cell's membrane time constant $\tau_{\mathrm{m}} > 0$ and input resistance $\mathrm{R} > 0$.

In the soft threshold leaky integrate-and-fire (SF-LIF) model, the probability of spiking at time t is a monotonically increasing and continuous function of the current voltage v_t [48]. Instead of the hard threshold spiking mechanism (3.3), we have:

$$\Pr(t^{(\mathrm{f})} \in (t - \mathrm{d}t,\, t]) = f(\mathrm{v}_t)\, \mathrm{d}t. \tag{3.5}$$

In this closely related model the input aggregation (3.2) and post-spike reset (3.4) remain the same.

Given an observed spike at $t^{(\mathrm{f})}$, we consider a time $t > t^{(\mathrm{f})}$ such that the neuron does not spike in the interval $\left(t,\, t^{(\mathrm{f})}\right)$. The solution to (3.2) for the cell membrane voltage at time t is

$$\mathrm{v}_t = \frac{\mathrm{R}}{\tau_{\mathrm{m}}} \int_0^{t-t^{(\mathrm{f})}} e^{-s/\tau_{\mathrm{m}}} I_{t-s} \mathrm{d}s. \tag{3.6}$$

We assume the time-dependent input I_t for neuron i is comprised of three components: (1) a random Gaussian white noise component with scale parameter σ, (2) a constant, deterministic component η and (3) the sum of inputs from the action potentials of other neurons in the network. This third 'network' component has the form

$$\sum_{j \neq i} \sum_{t_j^{(f)}} \mathbf{W}_{ji} \times \delta\left(t - t_j^{(\mathrm{f})}\right), \tag{3.7}$$

where $\delta(\cdot)$ is the dirac delta function. With I_t specified, (3.6) gives a continuous-time solution for

the cell membrane voltage. Assuming all the neurons in the population behave analogously, this establishes a continuous-time multivariate point process.

In practice noisy measurements capture neural activity [49, 50], and observed action potentials are assigned to individual cells and discrete event times [51]. We consider *spike-sorted* neural recording data as the observations of action potentials within fixed-length (eg 1 millisecond) intervals. To apply the continuous-time ST-LIF model in this context we adapt (3.2-3.7) to the discrete-time data.

Letting $t \in \mathbb{N}$ index time we discretize the continuous spike times $t^{(\mathrm{f})}$ (which arrive according to (3.3)) by mapping $t^{(\mathrm{f})} \mapsto t$ for $t^{(\mathrm{f})} \in (t-1,\, t]$. We define $\mathbf{y}_t \in \{0\,,1\}^n$ to be the observed spike indicators for the n neurons at each (discrete) observation time t, and we set $\mathbf{Y}$ to be the matrix of row vectors $\mathbf{y}_t$. The ith column $\mathbf{Y}_i$ of $\mathbf{Y}$ is the 'spike train' of the ith recorded neuron.

Let $t \in \mathbb{N}$ and assume $\mathrm{y}_{i,\,t-1} = 0$. A spike from neuron j at time $t-1$ charges the membrane of neuron i by $e^{-1/\tau_{\mathrm{m}}}\mathbf{W}_{ji}$ at time t (by (3.6) and (3.7)). It follows that the voltage of neuron i at time t is given by

$$\begin{aligned}\mathrm{v}_t &= \frac{\mathrm{R}}{\tau_{\mathrm{m}}}\int_0^1 e^{-s/\tau_{\mathrm{m}}} I_{t-s}\mathrm{d}s \;+\; e^{-1/\tau_{\mathrm{m}}}\mathrm{v}_{t-1} \\ &= \frac{\mathrm{R}}{\tau_{\mathrm{m}}}\mathrm{z}_t \;+\; \frac{\mathrm{R}\eta}{\tau_{\mathrm{m}}}(1-e^{-1/\tau_{\mathrm{m}}}) \;+\; \frac{\mathrm{R}}{\tau_{\mathrm{m}}}e^{-1/\tau_{\mathrm{m}}}\sum_{j=0}^{n}\mathbf{W}_{ji}\,\mathrm{y}_{t-1,\,j} \;+\; e^{-1/\tau_{\mathrm{m}}}\mathrm{v}_{t-1} \end{aligned} \tag{3.8}$$

where $\mathrm{z}_t \sim \mathrm{N}(0\,,\sigma^2)$.

Limitations of the Leaky Integrate-and-Fire Model The model described by (3.2-3.4) is highly simplified and neglects many aspects of neuronal dynamics [46]. We discuss some of its limitations, and mention extensions to more complex and realistic models.

One major simplification in the LIF model is the linear integration of input I_t, independent of the state of the postsynaptic neuron. Changing the functional form of (3.2) and (3.7) would allow for more complex interactions between nodes, with effects that could depend on the current state of postsynaptic node.

A second limiting assumption of the LIF model is the fact that it has no memory beyond the most recent spike. This means that the LIF model cannot capture adaptation, an observed behavior shown by most neurons in which they gradually adjust their firing rates to changes in stimulating current. In more complex models, this adaption behavior is mimicked with a filter for refractoriness that adds up contributions from several past spikes.

Besides the 'regularly-firing neurons' that show adaptation, there are fast-spiking neurons that show no adaptation as well as bursting and stuttering neurons which form a their own separate group. Bursting and stuttering neurons respond to constant stimulation with periodic bursting or aperiodic stuttering, behavior that may be interrupted by long intervals. In order to distinguish between neuron types [52] and capture these dynamics, a model must have memory beyond the most recent spike.

Extensions of the LIF model are broadly referred to as generalized leaky integrate-and-fire models [53, 54]. These extended models capture more of the complex dynamics of recorded neural population activity, and could be applied other other complex, interacting systems. Incorporating generalized LIF models into our approach is something we are interested in pursuing in future work.

3.2.2 Data model

We now introduce the assumed generative process for discrete event times. This is a generalized state-space model for multivariate time series of binary count data [38]. A diagram of the model is shown in Figure 3.2, and we provide a selected glossary for our data and model notation in Table 3.1.

The observations are $\mathbf{Y} = (\mathrm{y}_{t,i}) \in \{0\,, 1\}^{T \times n}$. The ith column $\mathbf{Y}_i$ are the event times of the ith node. Each node has an associated latent state that evolves with time, corresponding to the cell membrane voltage of a neuron. These variables are denoted $\mathbf{V} = (\mathrm{v}_{t,i}) \in \mathbb{R}^{T \times n}$. The unobserved connections among the nodes are given by $\mathbf{W} \in \mathbb{R}^{n \times n}$.

The observed 'event spikes' $\mathbf{y}_t = (\mathrm{y}_{t,1}, \ldots, \mathrm{y}_{t,n})$ at time t are Bernouilli draws, conditionally independent given the latent states $\mathbf{v}_t = (\mathrm{v}_{t,1}, \ldots, \mathrm{v}_{t,n})$. In particular, the event probability for node

i at time t is a logistic function of $\mathrm{v}_{t,i}$. This defines the model's observation equation.

The latent variables $\mathbf{V}$ evolve according the model's state equation. Each node's voltage resets to a 'resting level' of zero after it spikes. Voltages at time t depend on the previous voltages at time $t-1$ along with the observed spiking activity at time $t-1$ (moderated by connections encoded in $\mathbf{W}$). Previous voltages from time $t-1$ are subject to 'leakage' and decay with parameter $\delta \in [0\,,1]$. In addition, for each node i and time t, the latent state $\mathrm{v}_{t,i}$ receives constant input η_i and has an additive noise term $\mathrm{z}_{t,i}$ that captures unobserved input to or fluctuations of the cell membrane voltage. Accounting for the latent fluctuations $\mathbf{Z} = (\mathrm{z}_{t,i}) \in \mathbb{R}^{T\times n}$ is key to our inference on $\mathbf{W}$ given $\mathbf{Y}$.

We write the considered data model:

$$
\begin{aligned}
\mathrm{y}_{t,i} &\sim \text{Bernoulli}(\text{logit}^{-1}(\kappa(\mathrm{v}_{t,i}-1))) && \text{for each } i,\, t \\
\mathrm{v}_{t,i} &= \begin{cases} \delta \mathrm{v}_{t-1,i} + \eta_i + \mathrm{z}_{t,i} + \mathbf{w}_{\ast i}{}^{\intercal}\mathbf{y}_{t-1} & \text{if } \mathrm{y}_{t-1,i} = 0 \\ \eta_i + \mathrm{z}_{t,i} & \text{if } \mathrm{y}_{t-1,i} = 1 \end{cases} && \text{for each } i,\, t \\
\mathrm{z}_{t,i} &\sim \mathrm{N}\big(0\,, \sigma_i^2\big) && \text{for each } i,\, t
\end{aligned}
\tag{3.9}
$$

where $\mathbf{w}_{\ast i}$ is the ith column of $\mathbf{W}$, giving the in-edge weights for node i, and $\mathbf{y}_t$ denotes the tth row of $\mathbf{Y}$, the indicator vector of spikes at time t, so that $\mathbf{w}_{\ast i}{}^{\intercal}\mathbf{y}_t = \sum_{j=1}^{n} \mathbf{W}_{ji}\, \mathrm{y}_{t,j}$.

For the observation equation, we note that $\Pr(\mathrm{y}_{t,i} = 1 | \mathrm{v}_{t,i} = 1) = \frac{1}{2}$, and a larger κ creates a 'harder' spiking threshold around 1. This 'soft threshold' spiking mechanism ensures attractive differentiability properties to the model likelihood while allowing for behavior similar to a hard threshold mechanism (3.3) when κ is large.

If node i does not spike at time $t-1$, then at time t its previous latent voltage $\mathrm{v}_{t-1,i}$ is discounted according to δ and changes by the three input components: (1) the constant input η_i, (2) the mean zero random fluctuation $\mathrm{z}_{t,i}$ and (3) the network inputs from the nodes that did spike at time $t-1$ *and* have out-edges to i. Otherwise, if $\mathrm{y}_{t-1,i} = 1$, the voltage resets (to zero) and $\mathrm{v}_{t,i} = \eta_i + \mathrm{z}_{t,i}$.

Resting and spiking threshold membrane potentials, are typically around -70 and -50 millivolts,

respectively for cells in the nervous system. In the considered state-space model (3.9) all voltages are latent and unobserved. To make the model identifiable we fix the resting and spiking potentials at 0 and 1, respectively.

The assumptions of independence for the latent fluctuations $\mathbf{Z}$ and constant deterministic input η_i for each node i are strong and may not hold in many situations. Related events may be impacted in dependent ways by the unobserved input captured in $\mathbf{Z}$, and the expected rate η_i at which that input arrives may change over time. In section 3.7.1 we discuss a generalization of (3.9) that accommodates more complex dynamics in the latent process by eliminating these assumptions.

The generative model (3.9) is a reparameterized version of the discretized SF-LIF model (3.8). Its state update equation aligns with the recursive solution (3.8) with the following equivalences:

$$\begin{aligned} \frac{\mathrm{R}}{\tau_{\mathrm{m}}}\sigma &\longleftrightarrow \sigma_i \\ \frac{\mathrm{R}\eta}{\tau_{\mathrm{m}}}(1-e^{-1/\tau_{\mathrm{m}}}) &\longleftrightarrow \eta_i \\ \frac{\mathrm{R}}{\tau_{\mathrm{m}}}e^{-1/\tau_{\mathrm{m}}}\mathbf{w}_{\to i} &\longleftrightarrow \mathbf{w}_{\to i} \\ e^{-1/\tau_{\mathrm{m}}} &\longleftrightarrow \delta. \end{aligned} \tag{3.10}$$

Along with the latent network $\mathbf{W}$, the model parameters $\sigma_1, \ldots, \sigma_n$ and $\eta_1, \ldots, \eta_n$ are unknown, and need to be inferred. In our analyses we do not make inferences about the parameters κ and δ, instead assuming they are known or tuneable hyperparameters. We are interested in expanding our inference approach to include κ and θ in the future. In this work we have:

unknown model parameters, to infer: $\mathbf{W}$, $\theta = (\sigma_1, \ldots, \sigma_n, \eta_1, \ldots, \eta_n)$

known / tuneable hyperparemeters, not to infer: κ, δ.

To allow more concise notation going forward, we introduce θ for unknown model parameters besides the latent adjacency matrix $\mathbf{W}$.

The generative model (3.9) may be expanded by adding prior distributions for the parameters.

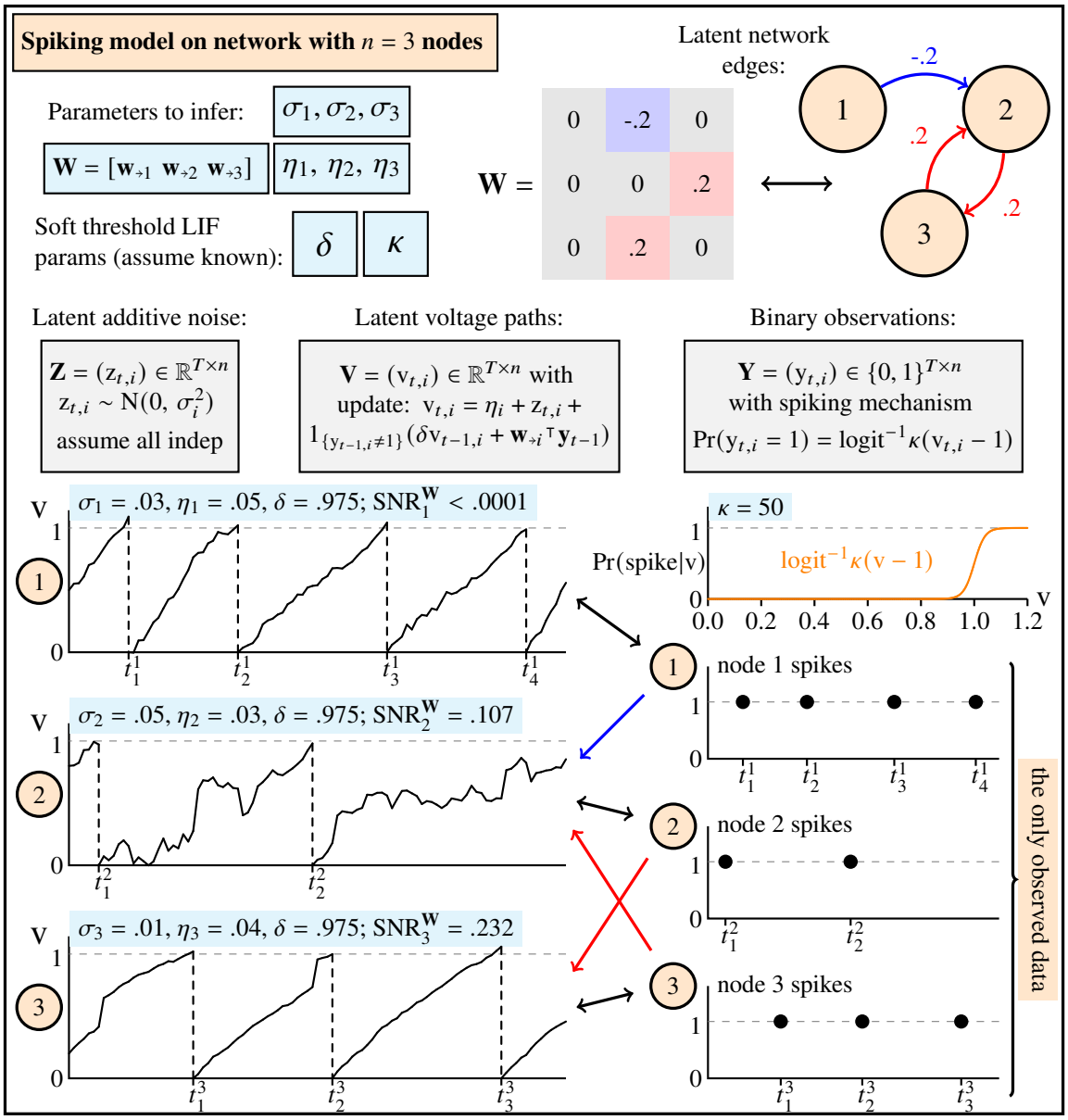

Figure 3.2: Diagram of generative model for discrete-time observations of point process data. In this figure we show the latent and observed data for a network of $n = 3$ nodes over a period of length $T = 100$. Along with the model parameters for each node we give its signal to noise ratio (see (3.45)) calculated based on $\mathbf{W}$ and 100k simulated observations.

This might be done to incorporate particular domain knowledge or to pursue a more Bayesian approach. For a prior on $\mathbf{W}$ one might use a generative network model. This is an approach taken in some related works that we discuss in section 3.6.

In our simulation study and data application (sections 3.4 and 3.5), we set and assume $\kappa = 50$.

Table 3.1: Selected notation glossary for data, model and joint likelihood

Data model notation	
n	dimension of point process / number of nodes
$\mathbf{Y} = (\mathrm{y}_{t,i}) \in \{0,1\}^{T\times n}$; $\mathbf{y}_t$	time series of discretely observed event times; tth row vector in $\mathbf{Y}$
$\mathbf{W} \in \mathbb{R}^{n\times n}$; $\mathbf{w}_{\cdot i}$	latent adjacency matrix (signed, directed, weighted), ith column of $\mathbf{W}$
$\mathbf{S} = \mathrm{sign}(\mathbf{W})$	matrix of signed, directed, unweighted edges in latent network
$\mathbf{V} = (\mathrm{v}_{t,i}) \in \mathbb{R}^{T\times n}$	dynamic latent states, the aggregated inputs (voltages) of the nodes
$\mathbf{Z} = (\mathrm{z}_{t,i}) \in \mathbb{R}^{T\times n}$	additive noise (fluctuations) in the latent aggregation process
$\theta_i = (\sigma_i, \eta_i)$	variability & constant input parameters for node i's latent aggregation
κ	scale parameter in logistic 'soft threshold' event probability function
$\delta \in (0,1)$	parameter for voltage decay / leakage
Notation for data processing and joint likelihood	
$\{t_1^i, t_2^i, \ldots, t_{m_i}^i\}$	the sorted event times for node i observed in $\mathbf{Y}$; $m_i = \sum_{t\le T} \mathrm{y}_{t,i}$
$\mathcal{X} = \{(i,k) : k \le m_i - 1\}$	indexed set of inter-spike intervals observed in $\mathbf{Y}$
$\Delta t_k^i = t_{k+1}^i - t_k^i$	length of (i,k)th inter-spike interval (node i's kth observed interval)
$I_i(s,t)$	node i's accumulated non-$\mathbf{Z}$ input in the period from s to t
$I(i,k) = I_i(t_k^i + 1,\ t_{k+1}^i)$	accumulated non-$\mathbf{Z}$ input in the (i,k)th observed inter-spike period
$\mathbf{z}_{(i,k)} = \left(\mathrm{z}_{t_k^i+1,i}, \ldots, \mathrm{z}_{t_{k+1}^i,i}\right)$	latent fluctuations for node i in its kth inter-spike period
$p(i,k)$	joint likelihood of the kth inter-spike interval for node i

The corresponding logistic function is shown in Figure 3.2 (middle-right). With this specified 'event spike' mechanism, the probability of an event is less than .007 for $\mathrm{v}_{t,i} \le .9$ and greater than .993 for $\mathrm{v}_{t,i} \ge 1.1$.

The decay parameter δ has a one-to-one correspondence with the parameter τ_m in the LIF model, an important dimensional quantity in neurophysiology called the membrane time constant [55]. Holding everything else constant, τ_m is the time (in milliseconds) it takes for a membrane potential to fall from 1 to e^{-1}, or ~36.8%.[1] It is an intrinsic membrane property that has been measured in *in vitro* studies. Estimates for τ_m range from 12ms to 30ms in studies of cells taken from guinea pigs [56], rats [57] and macaques [58].

In our simulation study we set $\delta = .975$, which corresponds to a (high) membrane time constant

[1] This can be seen in the context of our considered model by inverting (3.10)

$\tau_m \approx 39.5$ms. In section 3.5, we use the results of a relevant study to make an informed choice for δ in our analysis of neural activity recording data.

3.2.3 Joint Distribution

We consider data from the state space model (3.9) observed over a finite period; spikes are recorded for the n nodes at T consecutive times (milliseconds), and stored the binary matrix $\mathbf{Y}$. Each recorded millisecond corresponds to one data point (row in $\mathbf{Y}$). However, each observed *inter-spike period* is a more meaningful unit of observation for this model. The organization of the data into inter-spike periods is key to our analysis.

Recall that the latent state ($v_{t,i}$) of each node i resets after each of its 'spikes.' In the period before its next spike we observe all other incoming spikes (background network input), which may 'charge' the node's underlying 'voltage.' We can factorize the model's joint distribution over the observed inter-spike periods, exploiting their conditional independence.

Letting $\left\{t_1^i, t_2^i, \dots, t_{m_i}^i\right\}$ be the sorted spike times for node i, we index the set of observed *inter-spike periods* by

$$\mathcal{X} = \{(i,k) : i \in \{1,\dots,n\}, k \in \{1,\dots,m_i - 1\}\}. \tag{3.11}$$

Entries in $\mathbf{Y}$ book-ending the observed inter-spike periods for each node constitute censored data. Given values $\mathbf{W}$ for the latent network and θ for unknown model parameters, the joint distribution of the data $\mathbf{Y}$ and latent variables $\mathbf{Z}$ may be written

$$p_{\theta,\mathbf{W}}(\mathbf{Y},\mathbf{Z}) = \prod_{(i,k)\in\mathcal{X}} \underbrace{\prod_{t=t_k^i+1}^{t_{k+1}^i} p(y_{t,i},\, z_{t,i} \mid \mathcal{F}_{t-1},\, \mathbf{W})}_{=:\, p(i,k)} \times \overbrace{\prod_{\substack{(t,i)\,:\, i\in\{1,\dots,n\},\\ t\le t_1^i \text{ or } t>t_{m_i}^i}} p(y_{t,i},\, z_{t,i} \mid \mathbf{W})}^{\text{censored data contribution}}. \tag{3.12}$$

where $\mathbb{F} = (\mathcal{F}_t)_{t\ge 1}$ is the filtration of spiking and latent variable activity up to time t.

The right-most product term is the contribution of the 'censored data' to the joint likelihood. Un-

der (3.9), this term is messy to deal with *and* becomes inconsequential (relative to $\prod_{(i,k)\in\mathcal{X}} p(i,k)$) when each recorded neuron has many observed spikes (when T and $\mathcal{X}$ are large). In light of this we treat the term as equal to 1 and work with the following *approximate* joint distribution

$$p_{\theta,\mathbf{W}}(\mathbf{Y},\mathbf{Z}) = \prod_{(i,k)\in\mathcal{X}} p(i,k).$$

The term $p(i,k)$ is the contribution to the joint likelihood across the kth inter-spike interval for node i, conditioning sequentially on the other observed spiking activity in $\mathbb{F}$. For $t = t_k^i + 1, \dots, t_{k+1}^i$,

$$\begin{aligned}\log p(\mathrm{y}_{t,i}\,,\,\mathrm{z}_{t,i}\,|\,\mathcal{F}_{t-1},\,\mathbf{W}) &= \log p(\mathrm{y}_{t,i}|\mathrm{v}_{t,i}) + \log p(\mathrm{z}_{t,i}) \\ &= \mathrm{y}_{t,i}\kappa(\mathrm{v}_{t,i}-1) - \log(1+\exp(\kappa(\mathrm{v}_{t,i}-1))) + \log p(\mathrm{z}_{t,i}),\end{aligned}$$

so that the log likelihood of the (i,k)th inter-spike period may be written

$$\log p(i,k) = -\frac{\Delta t_k^i}{2}\log 2\pi\sigma_i^2 - \sum_{t=t_k^i+1}^{t_{k+1}^i}\left\{\frac{\mathrm{z}_{t,i}^2}{2\sigma_i^2} + \log(1+\exp(\kappa(\mathrm{v}_{t,i}-1)))\right\} + \kappa(\mathrm{v}_{t_{k+1}^i,i}-1), \quad (3.13)$$

where $\Delta t_k^i := t_{k+1}^i - t_k^i$ denotes the length of the inter-spike period.

We define $I_i(s,t,\mathbf{Y})$ to be the accumulated contributions from constant and background network input in the period from s to t, so that

$$I_i(s,t,\mathbf{Y}) := \eta_i \sum_{r=s}^{t}\delta^{t-r} + \mathbf{w}_{\to i}{}^{\mathsf{T}} \sum_{r=s}^{t-1}\delta^{t-r-1}\mathbf{y}_r. \quad (3.14)$$

For $t \in \{t_k^i+1, \dots, t_{k+1}^i\}$ we have $\mathrm{v}_{t,i} = I_i(t_k^i+1,\,t,\mathbf{Y}) + \sum_{s=t_k^i+1}^{t}\delta^{t-s}\mathrm{z}_{s,i}$. Note that δ^{t-s} is the exponential decay at time t of input that arrived at time s. We denote the accumulated input within the (i,k)th inter-spike period by

$$I_r(i,k,\mathbf{Y}) := I_i(t_k^i+1,\,t_k^i+r,\mathbf{Y}) \qquad \text{for } r = 1,\dots,\Delta t_k^i. \quad (3.15)$$

3.3 Methods

In this section we present our approach to finding an estimate of the latent network adjacency matrix $\mathbf{W}$ and unknown model parameters θ given the observed data $\mathbf{Y}$. We discuss methods for prediction of a partially observed node and present a signal-to-noise ratio for network effects.

3.3.1 High-level variational inference approach

We have specified a model for the observations $\mathbf{Y}$ with joint distribution $p_{\theta,\mathbf{W}}(\mathbf{Y}, \mathbf{Z})$. Our goal is to find values for the latent network $\mathbf{W}$ and unknown model parameters θ to maximize the marginal likelihood of the observations $\mathbf{Y}$

$$\mathrm{L}(\mathbf{W}; \mathbf{Y}) = p_{\theta,\mathbf{W}}(\mathbf{Y}) = \int p_{\theta,\mathbf{W}}(\mathbf{Y}, \mathbf{Z}) \, \mathrm{d}\mathbf{Z}. \tag{3.16}$$

This is an intractable integral (over all the latent fluctuations $\mathbf{Z}$), so that we cannot evaluate or differentiate the marginal likelihood. Moreover, the conditional posterior $p_{\theta,\mathbf{W}}(\mathbf{Z}|\mathbf{Y})$ is also intractable and we cannot use the EM algorithm.

Problems with similar difficulties arise often and well-known variational Bayes approaches exist [59, 60, 61]. In our related approach, we propose a highly structured variational approximation $q_{\phi|\mathbf{Y}}(\mathbf{Z})$ for $p_{\theta,\mathbf{W}}(\mathbf{Z}|\mathbf{Y})$, whose parameters ϕ we learn jointly along with the latent network $\mathbf{W}$ and unknown model parameters θ by maximizing the evidence lower bound (ELBO) of the marginal log likelihood:

$$\begin{aligned} \mathcal{L}(\theta, \mathbf{W}, \phi; \mathbf{Y}) &= \mathbb{E}_{q_\phi}[-\log q_{\phi|\mathbf{Y}}(\mathbf{Z}) + \log p_{\theta,\mathbf{W}}(\mathbf{Y}, \mathbf{Z})] \\ &\leq \mathbb{E}_{q_\phi}[-\log q_{\phi|\mathbf{Y}}(\mathbf{Z}) + \log p_{\theta,\mathbf{W}}(\mathbf{Y}, \mathbf{Z})] + \mathrm{KL}(q_\phi(\mathbf{Z}) \parallel p_{\theta,\mathbf{W}}(\mathbf{Z}|\mathbf{Y})) \\ &= \log p_{\theta,\mathbf{W}}(\mathbf{Y}). \end{aligned} \tag{3.17}$$

By maximizing $\mathcal{L}(\theta, \mathbf{W}, \phi; \mathbf{Y})$, we minimize the KL[2] divergence $\mathrm{KL}(q_\phi(\mathbf{Z}) \parallel p_{\theta,\mathbf{W}}(\mathbf{Z}|\mathbf{Y}))$ between

[2] Kullback–Leibler

our variational approximation $q_{\phi|\mathbf{Y}}(\mathbf{Z})$ and $p_{\theta,\mathbf{W}}(\mathbf{Z}|\mathbf{Y})$.

Using our proposed variational approximation $q_{\phi|\mathbf{Y}}(\mathbf{Z})$, we can take analytical gradients of a close approximation of the lower bound $\mathcal{L}(\theta, \mathbf{W}, \phi; \mathbf{Y})$ with respect to all variables $(\theta, \mathbf{W}, \phi)$. In this way we avoid noisy gradient approximations and the need for the frequently used 'reparameterization trick' [62].

Our approach allows for fully interpretable inference of both the model parameters of interest and the variational parameters. Moreover, it is computationally efficient when the observed spike trains are not too sparse. Its complexity scales linearly with length of the observation period T and number of nodes n, and quadratically in the length of the longest observed spike period (see section 3.3.6 below for further details on complexity).

3.3.2 Variational approximation $q_{\phi|\mathbf{Y}}$

We approximate $p_{\theta,\mathbf{W}}(\mathbf{Z}|\mathbf{Y})$ with a parsimonious multivariate Gaussian distribution $q_{\phi|\mathbf{Y}}(\mathbf{Z})$ that imposes conditional independence and a common structure across all the inter-spike periods. Simulations from the data model (3.9) inform our development of $q_{\phi|\mathbf{Y}}$.

Let $\mathbf{z}_{(i,\,k)}$ denote the latent variables $\left(\mathrm{z}_{t_k^i+1,\,i}, \ldots, \mathrm{z}_{t_{k+1}^i,\,i}\right)$ corresponding to the (i,k)th inter-spike period. Our approximation $q_{\phi|\mathbf{Y}}$ incorporates model parameters $(\mathbf{W}, \theta)$ and introduces variational parameters $\phi = \{\phi_i = (\vartheta_i, \tau_i, \nu_{1i}, \nu_{2i}, \beta_{1i}, \beta_{2i}) : i = 1, \ldots, n\}$. It has the form

$$q_{\phi|\mathbf{Y}}(\mathbf{Z}) = \prod_{(i,k)\in\mathcal{X}} q_{\phi_i|\mathbf{Y}}\left(\mathbf{z}_{(i,\,k)}\right) \sim \prod_{(i,k)\in\mathcal{X}} \mathrm{N}\Big(\mu_{\phi_i}(i,k,\mathbf{Y}),\ \Sigma_{\phi_i}(\Delta t_k^i)\Big). \tag{3.18}$$

Under $q_{\phi|\mathbf{Y}}$, $\mathbf{z}_{(i,\,k)}$ is normal with mean $\mu_{\phi_i}(i,k,\mathbf{Y})$ and covariance $\Sigma_{\phi_i}(\Delta t_k^i)$, and $\mathbf{z}_{(i,\,k)}$ is independent of $\mathbf{z}_{(i',\,k')}$ for all other $(i',k') \in \mathcal{X}$.

Consider $\mathrm{v}^*_{i,\,k} := \frac{1}{2}\left(\mathrm{v}_{i,\,t_{k+1}^i-1} + \mathrm{v}_{i,\,t_{k+1}^i}\right)$, the midpoint between the spiking voltage $\mathrm{v}_{i,\,t_{k+1}^i}$ and the voltage one step before $\mathrm{v}_{i,\,t_{k+1}^i-1}$ in the (i,k)th inter-spike period. We construct $\mu_{\phi_i}(i,k,\mathbf{Y})$ and

$\Sigma_{\phi_i}\left(\Delta t_k^i\right)$ so that this latent 'midpoint' quantity has a homogeneous normal distribution under $q_{\phi_i|\mathbf{Y}}$:

$$\mathrm{v}^*_{i,\,k} := \tfrac{1}{2}\left(\mathrm{v}_{i,\,t^i_{k+1}-1} + \mathrm{v}_{i,\,t^i_{k+1}}\right) \sim \mathrm{N}\left(\vartheta_i\,,\,\tau_i^2\right). \tag{3.19}$$

In simulations from (3.9), we find that these 'midpoint' values $v^*_{i,k} : (i,k) \in \mathcal{X}$ are approximately normal (see Figure 3.10). On the other hand, the simulated spiking voltages and the voltages one step before are right skewed and left skewed, respectively.

The property (3.19) will hold if and only if the following two equalities hold:

$$\mathbb{E}_{q_{\phi_i|\mathbf{Y}}}\mathbf{c}(\delta,\Delta t_k^i)^\intercal\mathbf{z}_{(i,\,k)} = \mathbf{c}(\delta,\Delta t_k^i)^\intercal\mu_{\phi_i}(i,k,\mathbf{Y}) = \vartheta_i - \tfrac{1}{2}\left(I_{\Delta t_k^i-1}(i,k,\mathbf{Y}) + I_{\Delta t_k^i}(i,k,\mathbf{Y})\right) \tag{3.20}$$

$$\mathrm{Var}_{q_{\phi_i|\mathbf{Y}}}\mathbf{c}(\delta,\Delta t_k^i)^\intercal\mathbf{z}_{(i,\,k)} = \mathbf{c}(\delta,\Delta t_k^i)^\intercal\Sigma_{\phi_i}(\Delta t_k^i)\,\mathbf{c}(\delta,\Delta t_k^i) = \tau_i^2, \tag{3.21}$$

where $\mathbf{c}(\delta,\Delta t_k^i)^\intercal\mathbf{z}_{(i,\,k)}$ is the random component of $\mathrm{v}^*_{i,\,k}$ and the vector $\mathbf{c}(\delta,\Delta t) \in \mathbb{R}^{\Delta t}$ is given by

$$\mathbf{c}(\delta,\Delta t)_t = \begin{cases} \tfrac{1}{2}\left(\delta^{\Delta t-t} + \delta^{\Delta t-1-t}\right) & \text{if } t = 1,\ldots,\Delta t-1 \\ \\ \tfrac{1}{2} & \text{if } t = \Delta t. \end{cases}$$

For our approximation $q_{\phi|\mathbf{Y}}(\mathbf{z}_{(i,\,k)})$, the mean vector $\mu_{\phi_i}(i,k,\mathbf{Y})$ is given by

$$\mu_{\phi_i}(i,k,\mathbf{Y})_s = \frac{2\left[\vartheta_i - \tfrac{1}{2}\left(I_{\Delta t_k^i-1}(i,k,\mathbf{Y}) + I_{\Delta t_k^i}(i,k,\mathbf{Y})\right)\right]\delta^{\Delta t_k^i-s+1}}{(1+\delta)\sum_{\ell=1}^{\Delta t_k^i}\delta^{2(\ell-1)} - 1} + \alpha_{\phi_i}(\Delta t_k^i)_s, \tag{3.22}$$

for $s = 1,\ldots,\Delta t_k^i$, where

$$\alpha_{\phi_i}(\Delta t)_s = \nu_{1i}\left[e^{-\nu_{2i}(\Delta t-s)} - \frac{(\delta+1)\sum_{\ell=1}^{\Delta t}(\delta e^{-\nu_{2i}})^{\ell-1} - 1}{(\delta+1)\sum_{\ell=1}^{\Delta t}\delta^{\ell-1} - 1}\right].$$

Note that $\sum_{s=1}^{\Delta t}\delta^{\Delta t-s}\alpha_{\phi_i}(\Delta t) = 0$. It is straightforward to verify that the mean vector (3.22) satisfies (3.20). The $\alpha_{\phi_i}(\Delta t)_s$ term is easier to interpret when written as the difference. For $r,s \in \{1,\ldots,\Delta t\}$

with $r < s$,

$$\alpha_{\phi_i}(\Delta t)_s - \alpha_{\phi_i}(\Delta t)_r = \nu_{1i}\left[e^{-\nu_{2i}(\Delta t - s)} - e^{-\nu_{2i}(\Delta t - r)}\right], \tag{3.23}$$

and this difference will be positive when ν_{1i} and ν_{2i} are both greater than zero.

Our approximation's covariance, $\Sigma_{\phi_i}(\Delta t)$ is specified via the Cholesky decomposition

$$\Sigma_{\phi_i}(\Delta t) = \mathbf{R}_{\phi_i}(\Delta t)\,\mathbf{R}_{\phi_i}(\Delta t)^\intercal, \tag{3.24}$$

with the entries of the (lower-triagular matrix) $\mathbf{R}_{\phi_i}(\Delta t)$ given by

$$\mathbf{R}_{\phi_i}(\Delta t)_{ts} = \begin{cases} \sigma_i\, \mathbf{r}_{\phi_i}(\Delta t)_t & \text{if } s = t \text{ (diagonal)} \\ -\sigma_i\, \varphi(\phi, \delta, \Delta t)\, \frac{\mathbf{r}_{\phi_i}(\Delta t)_s\, \mathbf{c}(\delta, \Delta t)_s}{(\Delta t - s)\, \mathbf{c}(\delta, \Delta t)_t} & \text{if } s < t \text{ (below-diagonal)} \\ 0 & \text{otherwise,} \end{cases} \tag{3.25}$$

where the vector $\mathbf{r}_{\phi_i}(\Delta t) \in \mathbb{R}^{\Delta t}$ has entries

$$\mathbf{r}_{\phi_i}(\Delta t)_t = \frac{1 - \beta_{1i}\exp\{-\beta_{2i}(\Delta t - t)/(\Delta t - 1)\}}{1 - \beta_{1i}\exp\{-\beta_{2i}\}} \qquad \text{where } \beta_{1i} \in (0,1) \text{ and } \beta_{2i} > 0,$$

and the scalar $\varphi(\phi, \delta, \Delta t)$ is set to satisfy (3.21). In order for $\varphi(\phi, \delta, \Delta t)$ to be a real number, it is sufficient that τ_i and σ_i satisfy $2\tau_i \geq \sigma_i$. We compute $\varphi(\phi, \delta, \Delta t)$ and provide the details of this constraint on τ_i and σ_i below in section 3.3.3.

Features of $q_{\phi|\mathbf{Y}}$ Our variational approximation allows for *fully interpretable, white box inference* of the data generating process. Computing the approximation to observed data $\mathbf{Y}$, we recover estimates of the model parameters of interest while also gaining insight into the dynamics of the latent fluctions $\mathbf{Z}$ and voltage paths $\mathbf{V}$ via the fitted variational parameters. Here are some key features of $q_{\phi|\mathbf{Y}}$ not yet mentioned:

- In equation (3.22) the difference $\vartheta_i - \frac{1}{2}(I_{\Delta t_k^i - 1}(i, k, \mathbf{Y}) - I_{\Delta t_k^i}(i, k, \mathbf{Y}))$ between the expected 'midpoint' voltage $\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} v_{i,k}^*$ and an average of non-random accumulated inputs is weighted by

$\delta^{\Delta t_k^i - s + 1}$. When $\delta < 1$, this difference is captured more strongly in the entries of $\mu_{\phi_i}(i, k, \mathbf{Y})$ toward the end of the inter-spike periods.

- When $\nu_{1i}, \nu_{2i} > 0$, $\alpha_{\phi_i}(\Delta t)_s$ is exponentially increasing in s, so that under $q_{\phi|\mathbf{Y}}$, larger random contributions (from $\mathbf{Z}$) to the spiking voltages are more likely to come from entries of $\mathbf{z}_{(i,k)}$ toward the end of each inter-spike interval. Here ν_{1i} controls the magnitude of the difference and ν_{2i} controls how concentrated the increased effect will be at the ends of the inter-spike periods.

- At the start of each inter-spike interval $(i, k) \in \mathcal{X}$, $\mathrm{Var}_{q_{\phi|\mathbf{Y}}}(\mathrm{z}_{t_k^i+1,i}) = \Sigma_{\phi_i}(\Delta t_k^i)_{11} = \sigma_i^2$. Under $q_{\theta_i|\mathbf{Y}}$ the variability $\Sigma_{\phi_i}(\Delta t_k^i)_{tt}$ decreases as t increases. The parameters $\beta_{1i}, \beta_{2i} > 0$ control aspects of this decrease, with β_{1i} corresponding to magnitude and β_{2i} to rate.

- The parameterization of $\Sigma_{\phi_i}(\Delta t)$ allows for negative conditional correlations among the latent fluctuations between spikes.

- In using the Cholesky decomposition to specify $\Sigma_{\phi_i}(\Delta t)$, we guarantee its positive definiteness and ensure straightforward sampling from $q_{\phi_i|\mathbf{Y}}$, since

$$\mathbf{e} \sim \mathrm{N}(\mathbf{0}, \mathbf{I}_{\Delta t}) \;\Rightarrow\; \mathbf{R}_{\phi_i}(\Delta t)\,\mathbf{e} \sim q_{\phi_i|\mathbf{Y}}$$

  In addition, the form (3.24, 3.25) of $\Sigma_{\phi_i}(\Delta t)$ allows for easy computation of the entropy $-\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log q_{\phi_i|\mathbf{Y}}(\mathbf{z}_{(i,k)})$ of $q_{\phi_i|\mathbf{Y}}$.

- The mean vectors $\mu_{\phi_i}(i, k, \mathbf{Y})$ incorporate the current network estimate $\mathbf{W}$ and model parameters η_i along with the background network input (from $\mathbf{Y}$) through $I_{\Delta t_k^i - 1}(i, k, \mathbf{Y})$ and $I_{\Delta t_k^i}(i, k, \mathbf{Y})$ (see equations (3.14) and (3.15)).

3.3.3 Further details on the variational approxiation $q_{\phi_i|\mathbf{Y}}$

In section 3.3.2 above we define $\Sigma_{\phi_i}(\Delta t)$ via its Cholesky decomposition (3.24) and the lower-triangular matrix $\mathbf{R}_{\phi_i}(\Delta t)$. We note that the scalar $\varphi(\phi, \delta, \Delta t)$ in (3.25) is set to satisfy (3.21) and

exists as a real number if $2\tau_i \geq \sigma_i$. In this section we prove this innequality and show the calculation of $\varphi(\phi, \delta, \Delta t)$.

Restating the constraint (3.21) in terms of $\mathbf{R}_{\phi_i}(\Delta t)$, we have, for the (i, k)th interval with length Δt,

$$\mathrm{Var}_{q_{\phi_i|\mathbf{Y}}} \mathrm{v}^*_{i,k} = \|\mathbf{R}_{\phi_i}(\Delta t)^\intercal \mathbf{c}(\delta, \Delta t)\|^2 = \tau_i^2.$$

Let $\mathbf{D}_{\phi_i}(\Delta t)$ be diagonal with $\mathbf{D}_{\phi_i}(\Delta t)_{tt} = \mathbf{r}_{\phi_i}(\Delta t)_t$ and $\mathbf{L}_{\phi_i}(\Delta t) = \frac{1}{\varphi(\phi,\delta,\Delta t)}[\frac{1}{\sigma_i}\mathbf{R}_{\phi_i}(\Delta t) - \mathbf{D}_{\phi_i}(\Delta t)]$. Then

$$\|\mathbf{R}_{\phi_i}(\Delta t)^\intercal \mathbf{c}(\delta, \Delta t)\|^2 = \sigma_i^2 \|\mathbf{D}_{\phi_i}\mathbf{c}(\delta, \Delta t) + \varphi(\phi, \delta, \Delta t)\mathbf{L}_{\phi_i}(\Delta t)^\intercal \mathbf{c}(\delta, \Delta t)\|^2, \tag{3.26}$$

It is clear that the right hand side is a quadratic in $\varphi(\phi, \delta, \Delta t)$. Setting this equal to τ^2, we obtain a solution

$$\varphi(\phi, \delta, \Delta t) = 1 - \frac{\sqrt{\tau_i^2/\sigma_i^2 - \mathbf{c}^2(\delta, \Delta t)_{\Delta t}\, \mathbf{r}^2_{\phi_i}(\Delta t)_{\Delta t}}}{\|\mathbf{L}_{\phi_i}(\Delta t)^\intercal \mathbf{c}(\delta, \Delta t^i_k)\|}. \tag{3.27}$$

In showing how we arrive at this solution we drop some notation so that $\varphi := \varphi(\phi, \delta, \Delta t)$, $\mathbf{L} := \mathbf{L}_{\phi_i}(\Delta t)$, $\mathbf{D} := \mathbf{D}_{\phi_i}(\Delta t)$, $\mathbf{c} := \mathbf{c}(\delta, \Delta t)$ and $\mathbf{r} := \mathbf{r}_{\phi_i}(\Delta t)$. Using the quadratic formula, we know that

$$\varphi = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}. \tag{3.28}$$

Expanding (3.26), we have: $a = \|\mathbf{L}^\intercal \mathbf{c}\|^2$, $b = 2\langle \mathbf{L}^\intercal \mathbf{c}, \mathbf{D}\mathbf{c}\rangle$ and $c = \|\mathbf{D}\mathbf{c}\|^2 - \frac{\tau^2}{\sigma^2}$. In order for φ to be real, we must have $b^2 \geq 4ac$, or equivalently,

$$\frac{\tau_i^2}{\sigma_i^2} \geq \|\mathbf{D}\mathbf{c}\|^2 - \frac{\langle \mathbf{L}^\intercal \mathbf{c}, \mathbf{D}\mathbf{c}\rangle^2}{\|\mathbf{L}^\intercal \mathbf{c}\|^2} = \mathbf{c}^2_{\Delta t}\mathbf{r}^2_{\Delta t}. \tag{3.29}$$

The equality here comes from our careful construction of $\mathbf{R}_{\phi_i}(\Delta t)$. More immediately it follows from the fact that $\mathbf{D}\mathbf{c} = -\mathbf{L}^\intercal \mathbf{c} + \mathbf{c}_{\Delta t}\mathbf{r}_{\Delta t}\mathbf{e}^{(\Delta t)}$, where $\mathbf{e}^{(\Delta t)} \in \mathbb{R}^{\Delta t}$ is the standard basis vector having one in its last entry and zeros in the rest. From the definitions of $\mathbf{c}(\delta, \Delta t)$ and $\mathbf{r}_{\phi_i}(\Delta t)$, we have that $|\mathbf{c}_{\Delta t}\mathbf{r}_{\Delta t}| \leq \frac{1}{2}$, which gives us the constraint $2\tau_i \geq \sigma_i$ presented in section 3.3.2.

Rearranging (3.29) and substituting gives

$$\frac{\sqrt{b^2 - 4ac}}{2\sqrt{a}} = \sqrt{\tau_i^2/\sigma_i^2 - \mathbf{c}_{\Delta t}^2 \mathbf{r}_{\Delta t}^2}.$$

And again using the fact that $\mathbf{Dc} = -\mathbf{L}^\intercal \mathbf{c} + \mathbf{c}_{\Delta t} \mathbf{r}_{\Delta t} \mathbf{e}^{(\Delta t)}$ and noting that $(\mathbf{L}^\intercal \mathbf{c})_{\Delta t} = 0$, we have that

$$\langle \mathbf{L}^\intercal \mathbf{c}, \mathbf{Dc} \rangle = -\|\mathbf{L}^\intercal \mathbf{c}\|^2 \Rightarrow \tfrac{-b}{2a} = 1.$$

combining these these results in (3.28) gives (3.27).

To ensure $\varphi(\phi, \delta, \Delta t) \in \mathbb{R}$ and $q_{\phi_i|\mathbf{Y}}$ is well-defined we use the following reparameterization in our gradient update steps:

$$u_{1i} = \sigma_i^2/4\tau_i^2 \in (0, 1)$$
$$u_{2i} = \tau_i > 0.$$

And here is the complete reparameterization mapping we use to allow unconstrained gradient update steps for σ_i and ϕ_i:

$$(\vartheta_i,\ \sigma_i,\ \tau_i,\ \nu_{1i},\ \nu_{2i},\ \beta_{1i},\ \beta_{2i}) \mapsto (\log \vartheta_i,\ \operatorname{logit} \frac{\sigma_i^2}{4\tau_i^2},\ \log \tau_i,\ \log \nu_{1i},\ \log \nu_{2i},\ \operatorname{logit} \beta_{1i},\ \log \beta_{2i}). \tag{3.30}$$

3.3.4 Maximizing the approximate ELBO

To compute our variational approximation $q_{\hat{\phi}|\mathbf{Y}}$, we seek values $(\hat{\theta}, \hat{\mathbf{W}}, \hat{\phi})$ that (approximately) maximize the evidence lower bound $\mathcal{L}(\theta, \mathbf{W}, \phi; \mathbf{Y})$ (3.17).

We can write this variational objective as a sum over the observed inter-spike intervals:

$$\mathcal{L}(\theta, \mathbf{W}, \phi; \mathbf{Y}) = \sum_{(i,k) \in \mathcal{X}} \left\{ -\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log q_{\phi_i|\mathbf{Y}}(\mathbf{z}_{(i,k)}) + \mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log p(i, k) \right\} \tag{3.31}$$

For the second term in the summand, we use (3.13) to write:

$$\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log p(i,k|\mathbf{W}) = -\sum_{t=t_k^i+1}^{t_{k+1}^i} \left\{ \log \sigma_i + \frac{1}{2\sigma_i^2} \mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \mathrm{z}_{t,i}^2 + \mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log(1+\exp(\kappa(\mathrm{v}_{t,i}-1))) \right\}$$
$$+ \kappa \mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \mathrm{v}_{t_{k+1}^i,i} + \text{ constant w.r.t. } \theta, \mathbf{W}, \phi.$$

The expectations $\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \mathrm{z}_{t,i}^2 = \Sigma_{\phi_i}(\Delta t_k^i)_{tt} + \mu_{\phi_i}^2(i,k,\mathbf{Y})_t$ are readily available. Computing $\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \mathrm{v}_{t_{k+1}^i,i}$ is straightforward once we note that, from (3.9) and (3.19), we can write

$$\mathrm{v}_{t_{k+1}^i,i} = \frac{1}{1+\delta}\left(2\delta\, \mathrm{v}_{i,k}^* + \eta_i + \mathrm{z}_{t_{k+1}^i,i} + \mathbf{w}_{\ast i}{}^{\mathsf{T}} \mathbf{y}_{t_{k+1}^i-1}\right).$$

On the other hand, the integral $\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log(1+\exp(\kappa(\mathrm{v}_{t,i}-1)))$ does not have a closed-form expression. To create a tractable objective, we use a piecewise polynomial approximation $h(v)$ for the function $\log(1+\exp(\kappa(v-1)))$. We provide the details of this approximation in section 3.3.5 below.

For the entropy term in (3.31), we have

$$\begin{aligned} -\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log q_{\phi_i|\mathbf{Y}}(\mathbf{z}_{(i,k)}) &= \frac{1}{2} \log\det \Sigma_{\phi_i}(\Delta t_k^i) + \text{constant w.r.t. } \theta, \mathbf{W}, \phi \\ &= \Delta t_k^i \log \sigma_i + \sum_{t=1}^{\Delta t_k^i} \log \mathbf{r}_{\phi_i}(\Delta t_k^i)_t + \text{constant w.r.t. } \theta, \mathbf{W}, \phi, \end{aligned}$$

which is straightforward to compute.

All together this gives an approximate ELBO and tractable objective $\tilde{\mathcal{L}}(\theta, \mathbf{W}, \phi; \mathbf{Y})$ to (approximately) maximize over $(\theta, \mathbf{W}, \phi)$. In practice we perform mini-batch gradient ascent with adam updates [63] to make steps $\left(\theta^{(\ell-1)}, \mathbf{W}^{(\ell-1)}, \phi^{(\ell-1)}\right) \to \left(\theta^{(\ell)}, \mathbf{W}^{(\ell)}, \phi^{(\ell)}\right)$. For a mini-batch $\mathcal{Y} \subset \mathcal{X}$

containing m observed inter-spike intervals, we write the approximate variational objective

$$\begin{aligned}\tilde{\mathcal{L}}(\theta, \mathbf{W}, \phi; \mathcal{Y}) = &\sum_{(i,k)\in\mathcal{Y}} \left\{ \Delta t_k^i \log \sigma_i + \sum_{t=1}^{\Delta t_k^i} \log \mathbf{r}_{\phi_i}(\Delta t_k^i)_t \right\} \\ &- \sum_{(i,k)\in\mathcal{Y}} \sum_{t=t_k^i+1}^{t_{k+1}^i} \left\{ \log \sigma_i + \frac{\Sigma_{\phi_i}(\Delta t_k^i)_{tt} + \mu_{\phi_i}^2(i,k,\mathbf{Y})_t}{2\sigma_i^2} + \mathbb{E}_{q_{\phi_i|\mathbf{Y}}} h(\mathrm{v}_{t,i}) \right\} \\ &+ \sum_{(i,k)\in\mathcal{Y}} \frac{\kappa}{1+\delta} \left(2\delta\vartheta_i + \eta_i + \mu_{\phi_i}(i,k,\mathbf{Y})_{\Delta t_k^i} + \mathbf{w}_{\ast i}{}^{\mathsf{T}} \mathbf{y}_{t_{k+1}^i - 1} \right).\end{aligned}$$

We can compute the gradients $\nabla_{(\theta,\mathbf{W},\phi)} \tilde{\mathcal{L}}(\theta, \mathbf{W}, \phi; \mathcal{Y})$ analytically without noisy approximation or any 'reparameterization trick' [62]. In section 3.3.5 below we show how to differentiate $\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} h(\mathrm{v}_{t,i})$ with respect to the unknown parameters $(\theta, \mathbf{W}, \phi)$.

In practice we reparameterize $(\theta, \mathbf{W}, \phi)$ with $g(\theta, \mathbf{W}, \phi)$ to allow for unconstrained gradient update steps while ensuring that the distribution $q_{\phi|\mathbf{Y}}$ is properly defined (3.30), and also to prevent estimates of model parameters $\mathbf{W}$ and $\{\eta_i : i = 1, \ldots, n\}$ from diverging wildly in early update steps. Reparameterized gradient update steps with step size γ have the form

$$\begin{aligned}\left(\theta^{(r)}, \mathbf{W}^{(r)}, \phi^{(r)}\right) \leftarrow g^{-1}\Big(&g\left(\theta^{(r-1)}, \mathbf{W}^{(r-1)}, \phi^{(r-1)}\right) \\ &+ \gamma \, \nabla_{g(\theta,\mathbf{W},\phi)} \tilde{\mathcal{L}}(\theta, \mathbf{W}, \phi; \mathcal{Y})|_{(\theta,\mathbf{W},\phi)=\left(\theta^{(r-1)}, \mathbf{W}^{(r-1)}, \phi^{(r-1)}\right)} \Big). \end{aligned} \tag{3.32}$$

3.3.5 Approximation h for $\log(1 + \exp(\kappa(v - 1)))$

Let $f(v) = \log(1 + \exp(\kappa(v - 1)))$. As we compute our variational approximation and seek updates to the unknown parameters $\mathbf{x} = (\theta, \mathbf{W}, \phi)$ we encounter quantities of the following form:

$$\mathbb{E} f(\mathrm{V}) \quad \text{and} \quad \frac{\partial}{\partial x} \mathbb{E} f(\mathrm{V}) \quad \text{where} \quad \mathrm{V} \sim \mathrm{N}(m(\mathbf{x}), s^2(\mathbf{x})). \tag{3.33}$$

These present intractable integrals, without closed form solutions. To work around this, we construct an approximation h of f which allows us to compute plug-in approximations of the these quantities

(3.33) to use in our gradient ascent update steps (3.32). The function f with $\kappa = 50$ along with our approximation h and its construction are shown in Figure 3.3.

The approximation h is piecewise polynomial, with eleven 'knots' $\xi_1 < \ldots < \xi_{11}$. For $i = 2, \ldots, 10$, we set h on $[\xi_i, \xi_{i+1})$ to be a mixture of third degree Taylor expansions $\hat{f}_3(v; \xi_i)$ and $\hat{f}_3(v; \xi_{i+1})$ of f, centered at knots ξ_i and ξ_{i+1}, respectively. Between the consecutive knots ξ_i and ξ_{i+1}, we mix these Taylor polynomials with the cubic weight function $w_{a,b}(v) = 3(x-a)^2/(b-a)^2 - 2(x-a)^3/(b-a)^3$. This function $w_{a,b}$ (shown in Figure 3.3 (b)) satisfies the following

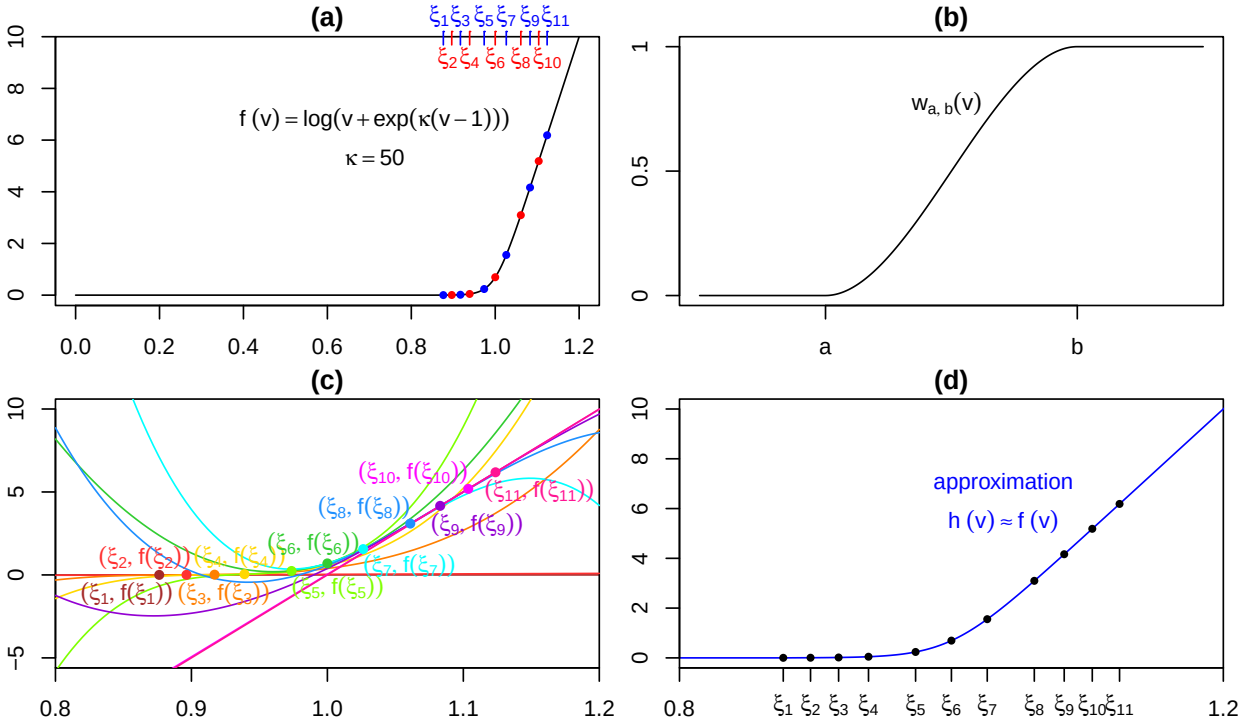

Figure 3.3: The function $f(v) = \log(1 + \exp(\kappa(v-1)))$ and the construction of its estimate h when $\kappa = 50$. Panel (a) shows f and the knots $\xi_1 < \ldots < \xi_{11}$ used for the piecewise polynomial approximation. Panel (b) shows the cubic weight function $w_{a,b}$ used to mix the Taylor expansions centered at the knots. Panel (c) shows the Taylor expansions and limiting linear functions used to construct h. Panel (d) shows the constructed approximation on the interval where f behaves non-linearly. (for $v \notin (\xi_1, \xi_{11})$, $f'''(v)$ is very close to zero.)

equalities:

$$w_{a,b}(a) = 0 \qquad w_{a,b}(b) = 1 \qquad w_{a,b}\left(\tfrac{1}{2}(a+b)\right) = 0.5$$
$$w'_{a,b}(a) = 0 \qquad w'_{a,b}(b) = 0$$

For our piecewise approximation, we set the middle knot to 1. Working outward from $\xi_6 = 1$, we set the next middlemost knots ξ_5 and ξ_7 where the absolute value of the third derivative of f is largest (where $f^{(4)}(v) = 0$). Subsequent knots above (ξ_8, ξ_9, ξ_{10}) and below (ξ_4, ξ_3, ξ_2) are set sequentially where the Taylor expansion at the previous knot stops being convex, that is, where its second derivative equals zero:

$$\xi_8 = (\xi_7 f'''(\xi_7) - f''(\xi_7))/f'''(\xi_7) \qquad \xi_4 = (\xi_5 f'''(\xi_5) - f''(\xi_5))/f'''(\xi_5).$$
$$\xi_9 = (\xi_8 f'''(\xi_8) - f''(\xi_8))/f'''(\xi_8) \qquad \xi_3 = (\xi_4 f'''(\xi_4) - f''(\xi_4))/f'''(\xi_4)$$
$$\xi_{10} = (\xi_9 f'''(\xi_9) - f''(\xi_9))/f'''(\xi_9) \qquad \xi_2 = (\xi_3 f'''(\xi_3) - f''(\xi_3))/f'''(\xi_3).$$

From the second lowest knot down to the first, h is the first degree Taylor expansion $\hat{f}_1(v; \xi_2)$ centered at ξ_2. We set ξ_1 equal to the place where this function crosses the x-axis:

$$\xi_1 = \xi_2 - f(\xi_2)/f'(\xi_2).$$

For $v < \xi_1$, $f(v)$ is very close to zero. We set $h = 0$ on $(-\infty, \xi_1]$, matching this limiting behavior.

From the second highest knot up to the highest, h is the first degree Taylor expansion $\hat{f}_1(v; \xi_{10})$ centered at ξ_{10}. We set ξ_{11} equal to the place where this function crosses the line $\kappa(v-1)$:

$$\xi_{11} = \frac{\kappa - \xi_{10} f'(\xi_{10}) + f(\xi_{10})}{\kappa - f'(\xi_{10})}$$

For $v > \xi_1$, $f(v)$ is very close to $\kappa(v-1)$. Again matching this limiting behavior with our

approximation, we set $h(v) = \kappa(v-1)$ for $v > \xi_{11}$. All together, we have:

$$h(v) = \sum_{i=1}^{10} h_i(v)\mathbf{1}_{[\xi_i,\xi_{i+1})}(v) + \kappa(v-1)\mathbf{1}_{[\xi_{11},\infty)}(v),$$

where h_i is the (at most) 6th degree polynomial given by:

$$h_i(v) = \begin{cases} \hat{f}_1(v;\xi_2) & \text{if } i = 1 \\ (1 - w_{\xi_i,\xi_{i+1}}(v)) \times \hat{f}_3(v;\xi_i) \; + \; w_{\xi_i,\xi_{i+1}}(v) \times \hat{f}_3(v;\xi_{i+1}) & \text{if } i = 2,\dots,9 \\ \hat{f}_1(v;\xi_{10}) & \text{if } i = 10. \end{cases}$$

Now we will use h to approximate the intractable quantities (3.33). First we have

$$\begin{aligned} \mathbb{E}f(\mathrm{V}) \approx \mathbb{E}h(\mathrm{V}) &= \sum_{i=1}^{10} \mathbb{E}\left[h_i(\mathrm{V})\mathbf{1}_{\mathrm{V}\in[\xi_i,\xi_{i+1})}\right] + \kappa\mathbb{E}\left[(\mathrm{V}-1)\mathbf{1}_{\mathrm{V}\in[\xi_{11},\infty)}\right] \\ &= \sum_{i=1}^{10} \int_{\xi_i}^{\xi_{i+1}} h_i(v)p(v;\mathbf{x})\mathrm{d}v + \kappa\int_{\xi_{11}}^{\infty}(v-1)p(v;\mathbf{x})\mathrm{d}v \end{aligned} \tag{3.34}$$

where

$$p(v;\mathbf{x}) = \frac{1}{\sqrt{2\pi}s(\mathbf{x})}\exp\left(\frac{-(v-m(\mathbf{x})^2)}{2s^2(\mathbf{x})}\right).$$

In (3.34) we have readily computable integrals! Note that, in general, we can efficiently compute integrals of the form $\mathbb{E}[\mathrm{X}^k\mathbf{1}_{\mathrm{X}\in[a,b)}]$, when X is normal.

For the second intractable quantity $\frac{\partial}{\partial x}\mathbb{E}f(\mathrm{V})$, we have

$$\frac{\partial}{\partial x}\mathbb{E}f(\mathrm{V}) \approx \frac{\partial}{\partial x}\mathbb{E}h(\mathrm{V}) = \sum_{i=1}^{10}\frac{\partial}{\partial x}\mathbb{E}\left[h_i(\mathrm{V})\mathbf{1}_{\mathrm{V}\in[\xi_i,\xi_{i+1})}\right] + \kappa\frac{\partial}{\partial x}\mathbb{E}\left[(\mathrm{V}-1)\mathbf{1}_{\mathrm{V}\in[\xi_{11},\infty)}\right]. \tag{3.35}$$

To see how we can compute the right hand side, first note that

$$\frac{\partial}{\partial x}p(v;\mathbf{x}) = p(v;\mathbf{x})\underbrace{\left[\frac{-\frac{\partial}{\partial x}s(\mathbf{x})}{s(\mathbf{x})} + \frac{s(\mathbf{x})(v-m(\mathbf{x}))\frac{\partial}{\partial x}m(\mathbf{x}) + (v-m(\mathbf{x}))^2\frac{\partial}{\partial x}s(\mathbf{x})}{s^3(\mathbf{x})}\right]}_{:=g_x(v;m(\mathbf{x}),s(\mathbf{x})),\text{ a 2nd degree polynomial in } v}.$$

Then if $\phi(v)$ is a kth degree polynomial on $[a, b)$, it follows that $\frac{\partial}{\partial x}(\phi(v)p(v; \mathbf{x}))$ is a $(k + 2)$th degree integrable polynomial on $[a, b)$. Using the Leibniz rule we can differentiate under the integral sign to get

$$\frac{\partial}{\partial x}\mathbb{E}\left[\phi(\mathrm{V})\mathbf{1}_{\mathrm{V}\in[a,b)}\right] = \int_a^b \phi(\mathrm{V})g_x(v;\, m(\mathbf{x}), s(\mathbf{x}))p(v; \mathbf{x})\mathrm{d}v = \mathbb{E}\left[\phi(\mathrm{V})g(v;\, m(\mathbf{x}), s(\mathbf{x}))\mathbf{1}_{\mathrm{V}\in[a,b)}\right] \tag{3.36}$$

where the final quantity is one we can efficiently compute.

3.3.6 Computational complexity

In this subsection we discuss the complexity of computing our variational approximation. In particular we consider the cost of making updates (3.32) in one full pass through the observed data $\mathbf{Y}$.

The most computationally expensive part of the gradient calculation $\nabla_{(\theta,\mathbf{W},\phi)}\tilde{\mathcal{L}}(\theta, \mathbf{W}, \phi; \mathcal{Y})$ are the gradients $\nabla_{(\theta,\mathbf{W},\phi)}\mathbb{E}_{q_{\phi_i|\mathbf{Y}}}h(\mathrm{v}_{t,i})$ of the expected approximation h for $t = t_k^i + 1, \ldots, t_{k+1}^i$ and $(i, k) \in \mathcal{Y}$. These involve computations of the form (3.35) and (3.36), whose most computationally expensive pieces are the partial derivatives of the variances $\mathrm{Var}_{q_\phi}(\mathrm{v}_{t,i})$ of the latent voltage under q_ϕ, which are taken with respect to parameters σ, τ, ν_1 and ν_2.

We consider the (i, k)th inter-spike interval with length Δt. For $t = t_k^i + r$, note that

$$\mathrm{Var}_{q_\phi}(\mathrm{v}_{t,i}) = \mathbf{c}_0(\delta, \Delta t, r)^\intercal \mathbf{R}_{\phi_i}(\Delta t)\mathbf{R}_{\phi_i}(\Delta t)^\intercal \mathbf{c}_0(\delta, \Delta t, r),$$

where the vector $\mathbf{c}_0(\delta, \Delta t, r) \in \mathbb{R}^{\Delta t}$ is given by

$$\mathbf{c}_0(\delta, \Delta t, r)_s = \begin{cases} \delta^{\Delta t - s} & \text{if } s = 1, \ldots, r \\ 0 & \text{if } s > r. \end{cases}$$

For $x \in (\sigma, \tau, \nu_1, \nu_2)$, if we set $\mathbf{R}_{\phi_i}^{(x)}(\Delta t)$ to be the matrix of partial derivatives of $\mathbf{R}_{\phi_i}(\Delta t)$ with

respect to x, then

$$\frac{\partial}{\partial x}\mathrm{Var}_{q_\phi}(\mathrm{v}_{t,i}) = \mathbf{c}_0(\delta, \Delta t, r)^\top \left[\mathbf{R}_{\phi_i}(\Delta t)\mathbf{R}^{(x)}_{\phi_i}(\Delta t)^\top + \mathbf{R}^{(x)}_{\phi_i}(\Delta t)\mathbf{R}_{\phi_i}(\Delta t)^\top\right]\mathbf{c}_0(\delta, \Delta t, r).$$

The computational complexity of computing $\frac{\partial}{\partial x}\mathrm{Var}_{q_\phi}(\mathrm{v}_{t,i})$ for each $t = t^i_k + 1, \ldots, t^i_{k+1}$ is $O(\Delta t^3)$. Recursively calculating the derivatives significantly improves runtime in our implementation, but does not change the overall complexity.

Consider the longest observed spike period in the dataset with length Δt^*. For a given node i the maximum number of Δt^*-length intervals that may be observed in the observed data is $T/\Delta t^*$. Thus the overall computation complexity of running the gradient updates across one epoch (each inter-spike interval observed once) is $O(nT(\Delta t^*)^2)$.

With the complexity scaling quadratically in Δt^*, computing our variational approximation becomes much more expensive when there is sparsity in the observed spikes and the data contains long inter-spike periods. We believe that adjustments could be made to our approach to counteract this limitation. This is an area we are interested in pursuing in future work.

3.3.7 Inferring the existence of edges $(i, j) \in \mathcal{E}$

As part of our objective to learn the latent network structure among the observed event streams, we wish to infer whether or not a latent edge exists between each given pair nodes. For the signed, directed, weighted graph $\mathcal{G}$, this amounts to estimating the signed, directed, unweighted adjacency matrix $\mathbf{S} = \mathrm{sign}(\mathbf{W})$.

Computing our variational approximation provides an estimate $\hat{\mathbf{W}}$ of the signed strength of each connection in $\mathbf{W}$ based on the observations $\mathbf{Y}$ and the generating process (3.9). However, each estimated connection $\hat{\mathbf{W}}_{ij}$ for $i \neq j$ will always be non-zero because the gradient based optimization *does not* provide sparsity. It follows that $\mathrm{sign}(\hat{\mathbf{W}})$ will be non-zero at all its off-diagonal entries and thus a very bad estimator of $\mathbf{S}$.

We propose a criterion for determining whether or not a directed pair (i, j) constitutes a

non-zero edge in $\mathcal{E}$ based on update steps from the computation of $\hat{\mathbf{W}}$. Our proposed criterion considers whether the estimate $\hat{\mathbf{W}}_{ij}$ has converged significantly away from zero *and* corresponds to a meaningful effect size.

Algorithm 3.1 provides the details of our criterion, and gives the procedure for obtaining our estimate $\hat{\mathbf{S}}$ for $\mathbf{S}$. Note K is the number of epochs run in computing our variational approximation and $\hat{\mathbf{W}}^{(k)}$ is our estimate for $\mathbf{W}$ after the kth epoch. Our criterion is based on the estimates from the last k^* steps ($k^* < K$). In sections 3.4 and 3.5 we set $k^* = 25$, $c^* = 3$ and $m^* = .05$. Figure 3.21 shows our criterion being applied to the in- and out-edges of one neuron ($i = 13$) in the DMFC_RSG dataset.

Algorithm 3.1 Estimate $\mathbf{S} = \text{sign}(\mathbf{W})$ from variational updates

Require: sequential updates $\hat{\mathbf{W}}^{(1)}, ..., \hat{\mathbf{W}}^{(K)}$
Require: $k^* \leq K$ updates to include
Require: $c^* > 0$ controlling width of interval
Require: $m^* > 0$ effect size threshold
  initialize $\hat{\mathbf{S}}_{ij} \leftarrow 0, i = 1, ..., n, j = 1, ..., n$
  for $(i, j) \in \{1, ..., n\} \times \{1, ..., n\}$ with $i \neq j$ **do**
    $m \leftarrow \frac{1}{k^*} \sum_{k=1}^{k^*} \hat{\mathbf{W}}_{ij}^{(K-k^*+k)}$
    $v^2 \leftarrow \frac{1}{k^*} \sum_{k=1}^{k^*} \left(\hat{\mathbf{W}}_{ij}^{(K-k^*+k)} - m \right)^2$
    $L \leftarrow m - c^* v$
    $U \leftarrow m + c^* v$
    if $L > 0$ **and** $U > m^*$ **then** ▷ interval above zero and reaches effect size threshold
      $\hat{\mathbf{S}}_{ij} \leftarrow 1$
    else if $U < 0$ **and** $L < -m^*$ **then** ▷ interval below zero and reaches effect size threshold
      $\hat{\mathbf{S}}_{ij} \leftarrow -1$
    end if
  end for
  return estimated edge signs $\hat{\mathbf{S}}$

Estimating $\mathbf{S}$ on top of $\mathbf{W}$ refines our understanding of the inferred latent structure and allows us to use methods for unweighted networks. It helps us visualize our inferred network by eliminating edges that we determine to be insignificant either 'statistically' or in terms of effect size, and it allows us to assess and compare the results of our approach in more ways. For example, computing our proposed $\hat{\mathbf{S}}$ provides grounds for comparison of our method to other established methods for inferring connectivity (eg transfer entropy).

3.3.8 Spike prediction for a held-out node

In situations where ground truth is available we can assess our model and inference procedure by comparing our estimates of the latent network with the known network structure. This is the case in our simulation study below, but in general the latent network connections among recorded neurons is not known. Moreover, our considered data model is an oversimplification of neuronal dynamics [64], and while we may wish to infer, in a real data application, that estimated non-zero entries of $\mathbf{W}$ to reflect real causal dependencies between neurons, we know that the spiking mechanisms and underlying process are quite different from (3.9).

Predictive performance on held-out data is a natural tool for the assessment of the model, inference procedure, and estimates. We consider the problem of predicting the event times of a held-out node in the situation where event times of all nodes is observed across a training period, but in a subsequent testing period the activity of one node is missing and needs to be predicted.

Suppose we set out to record the spiking activity of n neurons for a period of length T. The recording of neuron i^* is interrupted at some point, and we last observe it spike at time T^*. Meanwhile the recording of the other $n-1$ neurons is complete. What can we say about the spiking activity of neuron i^* in the period from T^*+1 to T?

We use the complete training data $\mathbf{Y}^{\text{train}} \in \{0,1\}^{T^* \times n}$ to compute our variational objective, obtaining model parameter estimates $(\hat{\theta}, \hat{\mathbf{W}})$. The test data $\mathbf{Y}^{\text{test}} \in \{0,1\}^{S \times n}$ is partially observed as $\mathbf{Y}^{\text{test}}_{-i^*} = (\mathrm{y}_{T^*+1,i^*}, \ldots, \mathrm{y}_{T^*+S,i^*}) \in \{0,1\}^{S \times n-1}$, where $S = T - T^*$. We wish to make reconstructive predictions of the held-out activity of neuron i^*, $\mathbf{Y}^{\text{test}}_{i^*} \in \{0,1\}^S$ given $(\hat{\theta}, \hat{\mathbf{W}})$ and $\mathbf{Y}^{\text{test}}_{-i^*}$. To this end, we estimate the conditional expectation of the unobserved spikes, $\mathbb{E}[\mathbf{Y}^{\text{test}}_{i^*} | \mathbf{Y}^{\text{train}}, \mathbf{Y}^{\text{test}}_{-i^*}]$, with the posterior predictive *conditional* expectation $\mathbb{E}[\mathbf{Y}^{\text{test}}_{i^*} | \mathbf{Y}^{\text{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}}, \mathrm{y}_{T^*,i^*} = 1]$, where

$$\mathbb{E}[\mathrm{y}_{T^*+r,i^*} | \mathbf{Y}^{\text{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}}, \mathrm{y}_{T^*,i^*} = 1] = \Pr(\mathrm{y}_{T^*+r,i^*} = 1 | \mathbf{Y}^{\text{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}}, \mathrm{y}_{T^*,i^*} = 1), \tag{3.37}$$

for $r = 1, \ldots, S$[3]. These conditional spike probabilities are point forecasts of the held-out data,

[3]Throughout this section we condition on the observed final spike $\{\mathrm{y}_{T^*,i^*} = 1\}$ for the held out neuron i^*; however, we sometimes drop this conditioned event from our expressions for ease of notation. For example we will write

which we can evaluate with cross entropy loss (see section 3.4.4.).

Computing (3.37) involves the posterior predictive distribution of v_{T^*+r,i^*}, the latent state of node i^* in the testing period, *conditional* on the observed activity of the other $n-1$ nodes. This distribution is difficult to deal with (ie simulate from) because it *anticipates* the spikes of the other nodes. Given our considered data model (3.9) with its adapted joint ditribution (3.12), it is much more natural to deal with the *non-anticipating* posterior predictive distribution of the latent states and spiking activity of the held out node, which 'sees' the spikes as they arrive. For $(\mathbf{V}^{\text{test}}_{i^*}, \mathbf{Y}^{\text{test}}_{i^*}) = (\mathrm{v}_{T^*+1,i^*}, \dots, \mathrm{v}_{T^*+S,i^*}, \mathrm{y}_{T^*+1,i^*}, \dots, \mathrm{y}_{T^*+S,i^*})$, this distribution is given by

$$
\begin{aligned}
p^{\text{na}}_{\hat{\theta}_{i^*},\hat{\mathbf{w}}_{\to i^*}|\mathbf{Y}^{\text{test}}_{-i^*},\, \mathrm{y}_{T^*,i^*}=1}(\mathbf{V}^{\text{test}}_{i^*}, \mathbf{Y}^{\text{test}}_{i^*}) &= p_{\hat{\theta}_{i^*}}(\mathrm{v}_{T^*+1,i^*}|\, \mathrm{y}_{T^*,i^*}=1)\, p(\mathrm{y}_{T^*+1,i^*}|\, \mathrm{v}_{T^*+1,i^*}) \\
&\times \prod_{r=2}^{S} p_{\hat{\theta}_{i^*},\hat{\mathbf{w}}_{\to i^*}}(\mathrm{v}_{T^*+r,i^*}|\, \mathrm{y}_{T^*+r-1,i^*}, \mathrm{v}_{T^*+r-1,i^*}, \mathbf{Y}^{\text{test}}_{r-1,-i^*})\, p(\mathrm{y}_{T^*+r,i^*}|\, \mathrm{v}_{T^*+r,i^*}).
\end{aligned} \tag{3.38}
$$

With its factorized form, this distribution is quite straightforward to sample from, and we will use it to estimate (3.37). But working with (3.38) alone will not suffice, since a 'reconstruction draw' $\mathbf{y}^{\text{na}} \in \{0,1\}^S$ of unobserved spikes from $p^{\text{na}}_{\hat{\theta}_{i^*},\hat{\mathbf{w}}_{\to i^*}|\mathbf{Y}^{\text{test}}_{-i^*},\, \mathrm{y}_{T^*,i^*}=1}$ is *not* a draw from the target distribution $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{i^*}|\mathbf{Y}^{\text{test}}_{-i^*})$. The key difference is that, in generating a sequence of latent states and spikes from (3.38), the conditioning on $\mathbf{Y}^{\text{test}}_{-i^*}$ does not remain constant and complete, but evolves sequentially. For example, the rth simulated pair $(\mathrm{v}_{T^*+r,i^*}, \mathrm{y}_{T^*+r,i^*})$ is influenced directly by network inputs only from time T^*+r-1. It is influenced indirectly, via $(\mathrm{v}_{T^*+r-1,i^*}, \mathrm{y}_{T^*+r-1,i^*})$, by spiking activity before time T^*+r-1 and is drawn independently of spikes on or after time T^*+r.

To aid our discussion of spiking activity relative to a fixed time T^*+r, we define, for each $r \in \{1,\dots,S\}$, past and current/future sets $\mathbf{Y}^{\text{test}}_{<r,-i^*} := \{\mathrm{y}_{r'j} : r' < r, j \neq i^*\}$ and $\mathbf{Y}^{\text{test}}_{\geq r,-i^*} := \{\mathrm{y}_{r'j} : r' \geq r, j \neq i^*\}$ that partition the observed spiking activity $\mathbf{Y}^{\text{test}}_{-i^*}$.

We use Monte Carlo simulations to estimate the non-anticipating posterior predictive probability $\Pr(\mathrm{y}^{\text{na}}_r = 1)$ of a spike at time T^*+r under (3.38). We can express and estimate this probability

$\Pr(\mathrm{y}_{T^*+r,i^*} = 1|\mathbf{Y}^{\text{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}})$ instead of (3.37)

recursively, with

$$\begin{aligned}\Pr(\mathrm{y}_r^{\mathrm{na}} = 1) &= \sum_{s=0}^{r-1} \Pr(\mathrm{y}_s^{\mathrm{na}} = 1, \mathrm{y}_{s+1}^{\mathrm{na}} = 0, \dots, \mathrm{y}_{r-1}^{\mathrm{na}} = 0, \mathrm{y}_r^{\mathrm{na}} = 1) \\ &= \sum_{s=1}^{r-1} \Pr(\mathrm{y}_s^{\mathrm{na}} = 1) \Pr(\mathrm{y}_{s+1}^{\mathrm{na}} = 0, \dots, \mathrm{y}_{r-1}^{\mathrm{na}} = 0, \mathrm{y}_r^{\mathrm{na}} = 1 | \, \mathrm{y}_s^{\mathrm{na}} = 1), \end{aligned} \tag{3.39}$$

where $\Pr(\mathrm{y}_0^{\mathrm{na}} = 1)$ and $\Pr(\mathrm{y}_{s+1}^{\mathrm{na}} = 0, \dots, \mathrm{y}_{r-1}^{\mathrm{na}} = 0, \mathrm{y}_r^{\mathrm{na}} = 1 | \, \mathrm{y}_s^{\mathrm{na}} = 1)$ is the probability of the next spike occurring at r given a spike at s. We compute an estimate $\tilde{p}^{\mathrm{na}}_{s \to r | s1}$ of this latter conditional probability by drawing from the non-anticipating distribution $p^{\mathrm{na}}_{\hat{\theta}_{i^*}, \hat{\mathbf{w}}_{\to i^*} | \mathbf{Y}^{\mathrm{test}}_{\geq s+1, -i^*}, \mathrm{y}_{T^*+s, i^*}=1}$, defined analogously to (3.38). Setting $\tilde{p}_0^{\mathrm{na}} = 1$, we compute recursively, for $r = 1, \dots, S$,

$$\tilde{p}_r^{\mathrm{na}} = \sum\nolimits_{s=0}^{r-1} \tilde{p}_s^{\mathrm{na}} \tilde{p}^{\mathrm{na}}_{s \to r | s1} \approx \Pr(\mathrm{y}_r^{\mathrm{na}} = 1). \tag{3.40}$$

With our estimates $\tilde{p}^{\mathrm{na}}_{ri^*}$ and $\tilde{p}^{\mathrm{na}}_{s \to r | s1}$ for $\Pr(\mathrm{y}_r^{\mathrm{na}} = 1)$ and $\Pr(\text{next spike at } r | \, \mathrm{y}_s^{\mathrm{na}} = 1)$, respectively, we compute further 'non anticipating' conditional probabilities of a spike at time r given a spike or non-spike ($\mathrm{y} \in \{0, 1\}$) at time s:

$$\tilde{p}^{\mathrm{na}}_{r | s\mathrm{y}} \approx \Pr(\mathrm{y}_r^{\mathrm{na}} = 1 | \mathrm{y}_s^{\mathrm{na}} = \mathrm{y}). \tag{3.41}$$

From our discussion above, it is clear that $\tilde{p}_r^{\mathrm{na}}$ is an unsatisfactory estimate of $\Pr(\mathrm{y}_{T^*+r, i^*} = 1 | \mathbf{Y}^{\mathrm{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}})$. To properly account for the constant and complete conditioning on *all* the observed spiking activity $\mathbf{Y}^{\mathrm{test}}_{-i^*}$, we must incorporate, in our estimates, the likelihood of the observed spiking activity given the 'reconstructed' spiking behavior of the held-out node i^*. We do this with a sort of forward-backward algorithm, computing forward spiking probabilities and backward conditional likelihoods using non-anticipating simulations and our variational approximation.

Using our partitioning of $\mathbf{Y}^{\mathrm{test}}_{-i^*}$ at r and a version of Bayes theorem, we write the posterior

predictive conditional probability (3.37) as

$$\begin{aligned}
&\Pr(\mathrm{y}_{T^*+r,i^*} = 1 |\, \mathbf{Y}^{\text{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}}) \\
&\quad = \frac{p_{\hat{\theta},\hat{\mathbf{W}}}(\mathrm{y}_{T^*+r,i^*} = 1, \mathbf{Y}^{\text{test}}_{\geq r,-i^*} |\, \mathbf{Y}^{\text{test}}_{<r,-i^*})}{p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{\geq r,-i^*} |\, \mathbf{Y}^{\text{test}}_{<r,-i^*})} \\
&\quad = \frac{\Pr(\mathrm{y}_{T^*+r,i^*} = 1 |\, \mathbf{Y}^{\text{test}}_{<r,-i^*}, \hat{\theta}, \hat{\mathbf{W}})\; p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{\geq r,-i^*} |\, \mathrm{y}_{T^*+r,i^*} = 1, \mathbf{Y}^{\text{test}}_{<r,-i^*})}{\sum_{\mathrm{y}\in\{0,1\}} \Pr(\mathrm{y}_{T^*+r,i^*} = \mathrm{y} |\, \mathbf{Y}^{\text{test}}_{<r,-i^*}, \hat{\theta}, \hat{\mathbf{W}})\; p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{\geq r,-i^*} |\, \mathrm{y}_{T^*+r,i^*} = \mathrm{y}, \mathbf{Y}^{\text{test}}_{<r,-i^*})}.
\end{aligned} \tag{3.42}$$

To compute our point estimate of the target reconstruction probability, we substitute in estimates for the components of (3.42). We have two distinct components to deal with. There is the conditional probability $\Pr(\mathrm{y}_{T^*+r,i^*} = 1 |\, \mathbf{Y}^{\text{test}}_{<r,-i^*}, \hat{\theta}, \hat{\mathbf{W}})$ of a held-out spike at time $T^* + r$ given the observed network activity *before* time $T^* + r$. This 'forward' probability is weighted by the 'backward' conditional likelihoods $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{\geq r,-i^*} |\, \mathrm{y}_{T^*+r,i^*} = \mathrm{y}, \mathbf{Y}^{\text{test}}_{<r,-i^*})$ of the observed spiking activity at and after time $T^* + r$ given a spike or non-spike at time $T^* + r$ and the observed spiking activity before time $T^* + r$.

For the first component, the 'forward' probability, we use Bayes theorem and some re-arranging to arrive at a recursion similar to (3.39). Denoting $\mathbb{P}^{\text{f}}_r := \Pr(\mathrm{y}_{T^*+r,i^*} = 1 |\, \mathbf{Y}^{\text{test}}_{<r,-i^*}, \hat{\theta}, \hat{\mathbf{W}})$, we write

$$\begin{aligned}
\mathbb{P}^{\text{f}}_r &= \sum_{s=0}^{r-1} \Pr(\mathrm{y}_{T^*+s,i^*} = 1, \mathrm{y}_{T^*+s+1,i^*} = 0, \ldots, \mathrm{y}_{T^*+r-1,i^*} = 0, \mathrm{y}_{T^*+r,i^*} = 1 |\, \mathbf{Y}^{\text{test}}_{<r,-i^*}, \hat{\theta}, \hat{\mathbf{W}}) \\
&= \sum_{s=0}^{r-1} \frac{p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{s:r,i^*} = (1, 0, \ldots, 0, 1), \mathbf{Y}^{\text{test}}_{s:(r-1),-i^*} |\, \mathbf{Y}^{\text{test}}_{<s,-i^*})}{p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*} |\, \mathbf{Y}^{\text{test}}_{<s,-i^*})} \\
&= \sum_{s=0}^{r-1} \frac{\mathbb{P}^{\text{f}}_s\, p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{(s+1):r,i^*} = (0, \ldots, 0, 1), \mathbf{Y}^{\text{test}}_{s:(r-1),-i^*} |\, \mathbf{Y}^{\text{test}}_{<s,-i^*},\; \mathbf{Y}^{\text{test}}_{s,i^*} = 1)}{p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*} |\, \mathbf{Y}^{\text{test}}_{<s,-i^*})}
\end{aligned} \tag{3.43}$$

In the second line we use Bayes theorem to remove $\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*}$, the observed spiking activity from time $T^* + s$ to $T^* + r - 1$, from the conditioning information of $\mathbb{P}^{\text{f}}_r$. This paves the way for drawing out the recursive term $\mathbb{P}^{\text{f}}_s$ in line three, where it is multiplied by the conditional non-anticipating joint likelihood of observing the next spike for the held out node i^* at time r and the observed spikes from time $T^* + s$ to $T^* + r - 1$. This joint likelihood term may be factored into its held out and

observed pieces:

$$p^{\text{na}}_{\hat{\theta}_{i^*},\hat{\mathbf{w}}_{\to i^*}|\mathbf{Y}^{\text{test}}_{(s+1):r,-i^*},\,\mathbf{Y}^{\text{test}}_{s,i^*}=1}(0,\ldots,0,1)\times p^{\text{na}}_{\hat{\theta},\hat{\mathbf{w}}|\mathbf{Y}^{\text{test}}_{(s+1):(r-1),i^*}=(0,\ldots,0)}(\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*}|\,\mathbf{Y}^{\text{test}}_{<s,-i^*},\;\mathbf{Y}^{\text{test}}_{s,i^*}=1).$$

The held-out piece on the left is straightforward to estimate with $\tilde{p}^{\text{na}}_{s\to r|s1}$ via Monte Carlo simulations as described above. Substituting the factored term back into (3.43), the right-hand term creates a likelihood ratio with the conditional likelihood $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*}|\,\mathbf{Y}^{\text{test}}_{<s,-i^*})$ in the denominator.

We estimate this likelihood ratio by simulating non-anticipating draws for each inter-spike interval that overlaps $\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*}$, that is, each indexed interval $(i,k)\in\mathcal{Y}_{s:(r-1),-i^*}$, where

$$\mathcal{Y}_{s:t,-i^*} := \{(i,k)\in\mathcal{X} : i\neq i^*,\; t^i_k < T^*+t,\; t^i_{k+1} > T^*+s\}.$$

Note that for $s<r$, and some conditioning information $\bullet$ about the held-out node i^*, we can write

$$\begin{aligned} p(\mathbf{Y}^{\text{test}}_{s:t,-i^*}|\,\mathbf{Y}^{\text{test}}_{<s,-i^*},\,\bullet) &= \prod_{(i,k)\in\mathcal{Y}_{s:t,-i^*}} p\left(\mathrm{y}_{\min\{s,\,t^i_k+1\},i},\ldots,\mathrm{y}_{\max\{t,\,t^i_{k+1}\},i}\,|\,\mathbf{Y}^{\text{test}}_{<s,-i^*},\,\bullet\right) \\ &= \prod_{\substack{(i,k)\in\mathcal{Y}_{s:t,-i^*}\\ t^i_k<T^*+s}} \frac{p\left(\mathrm{y}_{t^i_k+1,i},\ldots,\mathrm{y}_{\max\{t,\,t^i_{k+1}\},i}|\,\bullet\right)}{\Pr(t^i_{k+1}>s|\,\bullet)} \prod_{\substack{(i,k)\in\mathcal{Y}_{s:t,-i^*}\\ t^i_k\geq T^*+s}} p\left(\mathrm{y}_{t^i_k+1,i},\ldots,\mathrm{y}_{\max\{t,\,t^i_{k+1}\},i}|\,\bullet\right). \end{aligned} \tag{3.44}$$

To estimate $p(\mathbf{Y}^{\text{test}}_{s:t,-i^*}|\,\mathbf{Y}^{\text{test}}_{<s,-i^*},\,\bullet)$, we approximate the probabilities of the factored likelihood with simulations that account for the unobserved activity of node i^* given $\bullet$. Imputing $g(\bullet)\in[0,1]^S$ as the spiking activity of held-out node, we take non-anticipating draws from distributions analagous to (3.38). Note that this likelihood takes the same form as the 'backwards' conditional likelihood term $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{\geq r,-i^*}|\,\mathrm{y}_{T^*+r,i^*}=\mathrm{y},\mathbf{Y}^{\text{test}}_{<r,-i^*})$ in (3.42), which we estimate in the same way. In the

non-anticipating draws, we use the following imputations $g(\bullet)$:

$$p^{\text{na}}_{\hat{\theta},\hat{\mathbf{w}}|\mathbf{Y}^{\text{test}}_{(s+1):(r-1),i^*}=(0,\ldots,0)}(\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*}|\,\mathbf{Y}^{\text{test}}_{<s,-i^*},\,\mathbf{Y}^{\text{test}}_{s,i^*}=1) \;\Leftrightarrow\; g(\bullet)_t = \begin{cases} \tilde{p}^{\text{na}}_{t|s1} & t \leq s \\ 0 & t > s \end{cases}$$

$$p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{s:(r-1),-i^*}|\,\mathbf{Y}^{\text{test}}_{<s,-i^*}) \;\Leftrightarrow\; g(\bullet)_t = \tilde{p}^{\text{na}}_t$$

$$p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Y}^{\text{test}}_{\geq r,-i^*}|\,\mathrm{y}_{T^*+r,i^*}=\mathrm{y},\mathbf{Y}^{\text{test}}_{<r,-i^*}) \;\Leftrightarrow\; g(\bullet)_t = \tilde{p}^{\text{na}}_{t|r\mathrm{y}}.$$

We recall that $\tilde{p}^{\text{na}}_t$ and $\tilde{p}^{\text{na}}_{t|r\mathrm{y}}$ are the empirically estimated posterior predictive spike probabilities (3.40) and (3.41), and note that $\tilde{p}^{\text{na}}_{r|r\mathrm{y}} = \mathrm{y}$.

Using these imputations we 're-draw' many times along the observed inter-spike intervals that overlap with the training period, and we obtain approximate empirical estimates for the quantities in (3.44) and thus likelihoods $p(\mathbf{Y}^{\text{test}}_{s:t,-i^*}|\,\mathbf{Y}^{\text{test}}_{<s,-i^*},\,\bullet)$, which we plug into (3.43) and (3.42) to compute our forward-backward point estimate: $\hat{p}_{ri^*}$.

Computational complexity Calculating our forward-backward point estimates is quite computationally expensive. For our implementation in R[65] we vectorize our code as much as possible, which provides significant improvements in runtime while increasing the space complexity.

The computational time complexity of calculating the empirical posterior predictive spike probabilities $\tilde{p}^{\text{na}}_r$ and $\tilde{p}^{\text{na}}_{r|s\mathrm{y}}$ (all $r, s \in \{1, \ldots, S\}$) for held-out node i^* is $O(S^2N_1)$, where N_1 is the number of Monte Carlo draws used to calculate $\tilde{p}^{\text{na}}_{s\to r|s1}$. The space complexity in our implementation is $O(S^2) + O(N_1)$.

The computational time complexity of calculating our estimates $\hat{p}_{ri^*}$ on the test period for (3.42) given estimates $\tilde{p}^{\text{na}}_r$ and $\tilde{p}^{\text{na}}_{r|s\mathrm{y}}$ is $O(nS^2N_2)$, where N_2 is the number of draws used to estimate (3.44). The space complexity of these computations is $O(S\Delta t^*) + O(N_2)$, where Δt^* is the largest observed inter-spike interval observed across the testing period.

3.3.9 Signal-to-noise ratios for network effect

Under the data model (3.9), the relative levels of structured signal for the network effects (ie (3.7), $\mathbf{w}_{\cdot i}{}^{\intercal}\mathbf{y}_t$) and noise in the observable event times may vary across the n nodes. These levels depend on $\mathbf{W}$ and the model parameters θ. We propose a signal-to-noise ratio, for individual nodes, of the network effect size in their observations. This computed quantity provides an essential tool for analyzing the generative process (3.9) as well as results from fitting our variational approximation to observations $\mathbf{Y}$.

Calculating a signal-to-noise ratio for the network effect in the noisy point process data we consider is very appealing because it helps us characterize the nodes and how reliable and informative they are as message-passers within a complex system. Moreover, it is appealing for the same reason that it is difficult — it is hard to distinguish the network effect size amid multiple sources of variation in the observed data $\mathbf{Y}$.

We may interpret the observed event data as messages sent by the 'spiking' nodes to their downstream neighbors in the latent network. To us (observing just these events), node i's next recorded emission is a signal of the input it has received since its last. The amount we can learn from this signal depends on the variability in both the latent aggregation process and the spiking mechanism. When the latent fluctuations are small, the observed event provides a clearer reflection of the network edges into node i, adding structured signal to the observation. On the other hand large fluctuations pollute the signal, introducing noise.

The parameter σ_i controls the level of these latent fluctuations and is a primary driver in the network effect signal-to-noise ratio for a given node. The other model parameters play more complicated roles. Sparsity of observed events means less signal in general, while dense spiking can add more noise or more signal depending on σ_i along with the variability of input received from other nodes. Also, based on our definition, nodes with no edges will have a signal-to-noise ratio equal to zero (see, for example, node 1 in Figure 3.2), and for nodes with very few in-edges it will be small. This reflects the fact that these nodes' observed activity is not explained by the implicit network. However, their emissions still provide important information about the (lack of) network

structure.

In general, a signal-to-noise ratio (SNR) describes the tradeoff (with respect to some possibly implicit model) between structure in the data induced by the signal and variability due to the noise:

$$\text{SNR} = \frac{\sigma^2_{\text{signal}}}{\sigma^2_{\text{noise}}}.$$

This standard form does not apply in the problem we consider because the observed event times transmit both signal and noise, and there is no simple way to disentangle the two.

Our proposed SNR for individual nodes follows work in [66]. In this paper, the authors assume a point process generalized linear model (PP-GLM) for neurons spiking behavior, and compute a signal-to-noise ratio for a single neuron based on residual deviances from the PP-GLM fits. The data and context they consider aligns very closely our work, and we extend aspects of their approach to our model.

We fix $i \in \{1, \dots, n\}$ and begin by considering some logistic models for the observations from node i:

$$\left.\begin{aligned} M^i &: \text{logit}(\Pr(\mathrm{y}_{t,i} = 1)) = \kappa(\mathrm{v}_{t,i} - 1) \\ M^i_1 &: \text{logit}(\Pr(\mathrm{y}_{t,i} = 1)) = b_0 + b_1 \textstyle\sum_{s=1}^{r} \delta^{r-s} + b_2 \mathbf{w}_{\to i} \sum_{s=1}^{r} \delta^{r-s} \mathbf{y}_{t^i_k + r} \\ M^i_0 &: \text{logit}(\Pr(\mathrm{y}_{t,i} = 1)) = b_0 + b_1 \textstyle\sum_{s=1}^{r} \delta^{r-s} \end{aligned}\right\} \quad \begin{array}{c} t = t^i_k + r \leq t^i_{k+1} \\ \text{for some } (i,k) \in \mathcal{X}. \end{array}$$

Note that with the observation period partitioned by node i's recorded inter-spike periods (as described in section 3.2.3 above), we know there exists an $r > 1$ and $(i, k) \in \mathcal{X}$ such that $t = t^i_k + r$ with $r \leq t^i_{k+1}$.

The first model, M^i is the true observation model, conditional on the latent state. The only variability in M^i comes from the soft threshold spiking mechanism.

In the second model M^i_1, the log-odds of a spike at time t is conditioned on the true accumulated network input $\mathbf{w}_{\to i} \sum_{s=1}^{r} \delta^{r-s} \mathbf{y}_{t^i_k + r}$ for node i at time t. Model M^i_1 assumes knowledge of the edges $\mathbf{w}_{\to i}$ into node i and has a coefficient for constant accumulated input within the inter-spike periods.

This model contains M^i if there are no latent fluctuations (by setting $b_0 = -\kappa$, $b_1 = \kappa$ and $b_2 = \kappa\eta_i$). When $\sigma_i > 0$,

$$\mathrm{v}_{t,i} = \eta_i \sum_{s=1}^{r} \delta^{r-s} + \mathbf{w}_{\cdot i} \sum_{s=1}^{r} \delta^{r-s} \mathbf{y}_{t_k^i + r} + \text{ noise.}$$

Note that this noise 'sees' (is conditioned on) $\mathrm{y}_{t_k^i+1} = \ldots = \mathrm{y}_{t_k^i+r-1} = 0$, meaning that it cannot be treated as mean zero or independent.

We consider the following decomposition of the deviance D_1^i associated with model M_1^i into the deviance D^i of model M^i and the difference in $(D_1^i - D^i)$:

$$\begin{aligned} D_1^i &= D^i + (D_1^i - D^i) \\ &\approx \text{ [noise from observation eq.] + [noise from latent fluctuations].} \end{aligned}$$

While M^i conditions on the fluctuations $\mathrm{z}_{t_k^i+1}, \ldots, \mathrm{z}_{t_k^i+r}$, M_1^i ignores them entirely. The difference in deviance between these models provides a measure of this latent noise by telling us the cost of ignoring it.

The third model M_0^i is a null model with respect to network structure. This model only considers a coefficient for constant accumulated input and an intercept. If $\mathbf{w}_{\cdot i} = \mathbf{0}$ then model M_1^i reduces to M_0^i. The difference in deviance between these models provides a measure of the structured signal in the data associated with $\mathbf{w}_{\cdot i}$.

And so have the following definition for the signal to noise ratio for node i, as the ratio of the difference in deviance between fitted models $\hat{M}_0^i$ and $\hat{M}_1^i$, and the deviance of M_1^i, along with the approximate bias corrections for the number of parameters fit in the models:

$$\mathrm{SNR}_i^{\mathbf{W},\mathbf{Y}} := \frac{D_0^i - D_1^i + 1}{D_1^i + 3} = \frac{\log \mathbf{L}_i(\hat{M}_{1,i}) - \log \mathbf{L}_i(\hat{M}_{0,i}) + \frac{1}{2}}{-\log \mathbf{L}_i(\hat{M}_{1,i}) + \frac{3}{2}}, \tag{3.45}$$

where $\mathbf{L}_i(\hat{M})$ is the likelihood of node i's recorded observations achieved by the fitted model $\hat{M}$.

The signal-to-noise ratio is often reported in decibels (dB) as $10 \log_{10}(\mathrm{SNR})$. We adopt this convention in our simulation study and data applications (sections 3.4 and 3.5, respectively) below.

We write $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$ to emphasize that this signal-to-noise ratio is a function of the network connections $\mathbf{W}$ and the observations $\mathbf{Y}$. When $\mathbf{W}$ is unknown we approximate (3.45) with $\text{SNR}_i^{\hat{\mathbf{W}},\mathbf{Y}}$ by plugging in our estimate $\hat{\mathbf{w}}_{\to i}$ in model M_1^i.

3.4 Simulation study

We perform a simulation study, drawing multivariate point process data from the considered model (3.9) to analyze with our variational Bayes approach. We estimate the unknown model parameters $\mathbf{W}$ and θ, along with the variational parameters ϕ by running mini-batch stochastic gradient ascent with adam update steps [63].

We evaluate our model and inference approach in a number of ways: checking convergence at a high level and comparing estimates to known ground truth values, running diagnostic checks to determine how close our computed variational approximation is to its target conditional posterior distribution, and evaluating its prosterior predictive performance on data for individual held-out nodes across a testing period.

3.4.1 Simulated data and estimation procedure

We simulate from the state-space spiking model on a network of size $n = 20$. We begin by randomly generating the network adjacency matrix $\mathbf{W}$. We do this by independently drawing each (non-diagonal) edge $\mathbf{W}_{ij}$ of $\mathbf{W}$ from the following categorical distribution:

$\mathbf{W}_{ij} \sim$

w	$\Pr(\mathbf{W}_{ij} = \text{w})$
-0.1	$1/8$
0	$5/8$
0.1	$1/4$

We set the nodal parameters σ_i and η_i equal to values in (.01, .02, .03, .04, .05) and (.025, .03, .035, .04), respectively, so that each combination appears once.

We set $\delta = .975$, $\kappa = 50$ and *assume these parameters are known*. We randomly initialize the

latent voltages of the $n = 20$ nodes and run the generative model (3.9) for a short warm-up period, then we draw 52,000 realizations, storing the latent paths $\mathbf{V}$ along with the observed spikes $\mathbf{Y}$.

Table 3.2: Summary of nodes in simulation study

i	$\sum_{t<50\text{k}} y_{t,i}$	$\Delta t_i^{*\dagger}$	$\text{SNR}_i^{\mathbf{W},\mathbf{Y}\ddagger}$	σ_i	η_i	$\#\{j : \mathbf{W}_{ji} > 0\}$	$\#\{j : \mathbf{W}_{ji} < 0\}$	$\sum_{j\neq i} \mathbf{W}_{ji}$
1	1426	54	-0.79	0.01	0.030	6	1	0.5
2	1225	65	-0.81	0.01	0.025	7	3	0.4
3	1431	52	-2.78	0.01	0.035	5	1	0.4
4	1797	44	-3.13	0.01	0.040	5	0	0.5
5	1116	78	-3.55	0.02	0.025	8	3	0.5
6	921	102	-5.10	0.02	0.030	5	4	0.1
7	1440	58	-6.33	0.02	0.035	4	1	0.3
8	882	148	-7.05	0.03	0.025	6	2	0.4
9	569	228	-7.12	0.03	0.035	3	6	-0.3
10	1019	103	-8.71	0.03	0.030	5	2	0.3
11	1357	61	-8.74	0.02	0.040	2	2	0.0
12	1477	62	-8.82	0.03	0.040	4	3	0.1
13	697	212	-9.07	0.05	0.025	6	5	0.1
14	1203	123	-10.40	0.05	0.030	6	3	0.3
15	1650	65	-10.47	0.04	0.040	5	2	0.3
16	839	156	-10.61	0.04	0.030	3	2	0.1
17	1150	94	-11.14	0.04	0.035	4	4	0.0
18	1173	123	-11.98	0.05	0.035	4	3	0.1
19	548	354	-12.77	0.04	0.025	2	2	0.0
20	1638	90	-13.49	0.05	0.040	5	1	0.4

† $\Delta t_i^* = \max\{\Delta t_k^i : k = 1, ..., m_i\}$ from $T = 50\text{k}$ training set
‡ computed signal-to-noise ratio (3.45) given in decibels (dB)

Table 3.2 provides a summary of the simulated nodes. The nodes are ordered in descending order by their computed signal-to-noise ratios $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$ (3.45). For each node i we have its parameters σ_i and η_i along with a summary of its network connections — postive in-edges, negative in-edges and summed in-degree. Table 3.2 also reports two high-level summary statistics: the number of events observed at this node in $T = 50\text{k}$ observations and the maximum inter-spike interval observed in the $T = 50\text{k}$ training period.

We can see how various factors combine to affect the signal-to-noise ratio of the nodes in our simulation. The strongest effect comes from σ_i.

To see how varying amounts of observations affect the model fit, we run our analysis at five

training cutoffs: $T = 1\text{k}$, $T = 5\text{k}$, $T = 10\text{k}$, $T = 25\text{k}$ and $T = 50\text{k}$. Each successive training set contains the observations of the previous. At each cutoff we perform the following computations for our variational approximation, storing the intermediate and final updates from each epoch:

Computing the variational approximation

1. Initialize $\hat{\mathbf{W}}_{ij} = 0$ all $i \neq j$, $\sigma_i = .04$ for all i, and variational parameters ϕ_i to the same starting values for each node. We initialize η_i based on the observed event intensity of node i, setting $\eta_i = T / \sum_{t<T} y_{t,i}$, the inverse of the observed mean time per event.

2. Run stochastic batch gradient ascent, cycling through the training data $\mathbf{Y}$ for $K \geq 50$ [4] epochs, making updates (3.32) to our estimates for $(\theta, \mathbf{W}, \phi)$ using adaptively choosen, adam step sizes.

   - After each epoch, store the current updates $(\hat{\theta}^{(\ell)}, \hat{\mathbf{W}}^{(\ell)}, \hat{\phi}^{(\ell)})$ and calculate the varational objective (approximate ELBO) $\tilde{\mathcal{L}}(\hat{\theta}^{(\ell)}, \hat{\mathbf{W}}^{(\ell)}, \hat{\phi}^{(\ell)}; \mathcal{X}_T)$ across all observed interspike periods in the training data.

3. Output final parameter estimates $(\hat{\theta}, \hat{\mathbf{W}}, \hat{\phi}) \leftarrow (\hat{\theta}^{(K)}, \hat{\mathbf{W}}^{(K)}, \hat{\phi}^{(K)})$ after K epochs.

In this procedure we randomly shuffle the observed inter-spike periods $\mathcal{X}_T$ into (16 or 32) batches[5] at the start of each epoch. While we update all parameters simultaneously, the updates of node-specific variables are independent (given the mini-batch assignments). That is, updates to $(\hat{\theta}_i, \hat{\phi}_i, \hat{\mathbf{w}}_{\to i})$ do not affect the updates to $(\hat{\theta}_j, \hat{\phi}_j, \hat{\mathbf{w}}_{\to j})$ when $i \neq j$, and we can fit the model by computing variational approximations $q_{\hat{\phi}_i}$ for each node in parallel. Consequently it makes sense to evaluate the fits for each node separately. We note, however, that for the prediction of the spiking activity of a held out node (see sections 3.3.8 above and 3.4.4 below), the inferred parameters associated with all of the nodes will be used. This means that a poor fit on the training data for node j *will* affect the predictive performance for a held out node i on the test set, and vice versa.

[4] We run $K = 200$ and $K = 100$ epochs for $T = 1\text{k}$ and $T = 5\text{k}$, respectively. For $T = 10\text{k}$, $T = 25\text{k}$ and $T = 50\text{k}$ we run $K = 50$ epochs.

[5] We use 16 batches for $T = 1\text{k}$ and $T = 5\text{k}$, and 32 batches for $T = 10\text{k}$, $T = 25\text{k}$ and $T = 50\text{k}$.

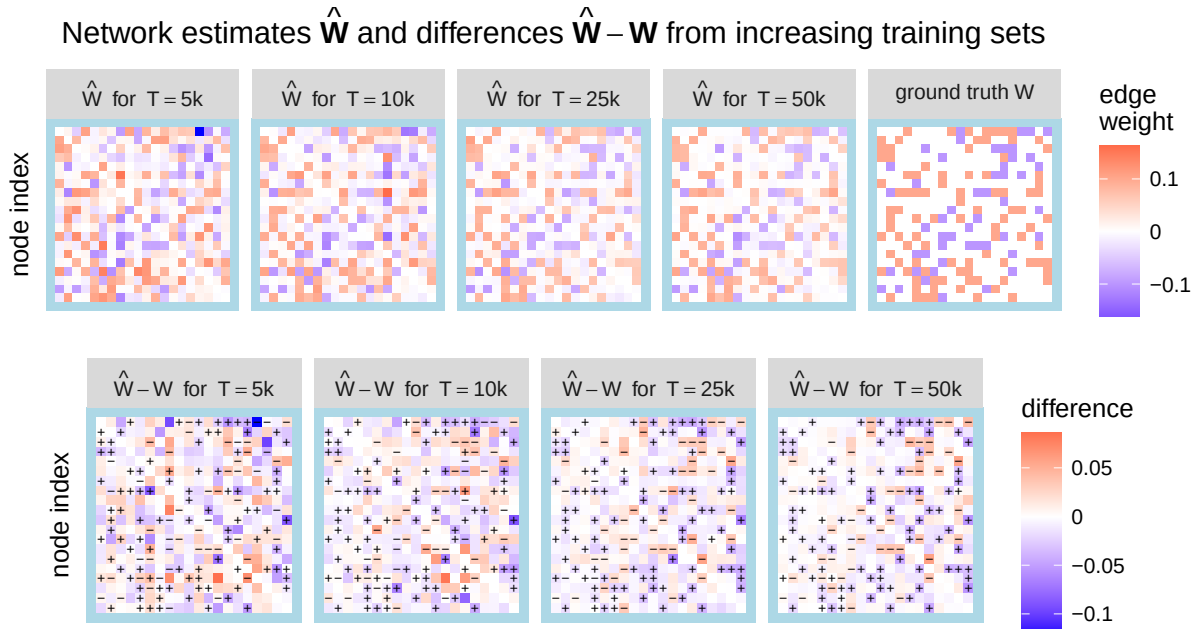

Figure 3.4: The network estimates $\hat{\mathbf{W}}$ from fitting our variational approximation to expanding sets of observations from the simulated data. The estimates are compared with the ground truth $\mathbf{W}$.

3.4.2 Inferred network results

Figure 3.4 compares the estimated adjacency matrices $\hat{\mathbf{W}}$ associated with the five expanding training periods alongside the true latent network $\mathbf{W}$. We can see that our variational approximation is able to accurately recover many of the network edges. While the estimates change and improve as more observations are included, there appears to be a degree of stability in the inferred connections.

Looking at the differences $\hat{\mathbf{W}} - \mathbf{W}$ in the second row we see underestimates for positive entries and overestimates for negative entries. We can see this similar pattern in Figure 3.6 panel (a) showing the evolving network edges for $T = 50$ by epoch and target node. It seems the many of the estimates for non-zero edges are biased toward zero.

Along with $\mathbf{W}$, Figure 3.6 shows the evolving estimates for the variational parameters ϕ and other inferred model parameters θ. Our variational approximation does a better job recovering low values of σ_i and high values of η_i Variability in the variational parameter estimates suggests to us on the one hand that our approximation may be capturing the varied behavior of the nodes. But we also see non-converging estimates, which suggests that our approximation may be too restrictive.

We discuss model diagonostics further in section 3.4.3 below.

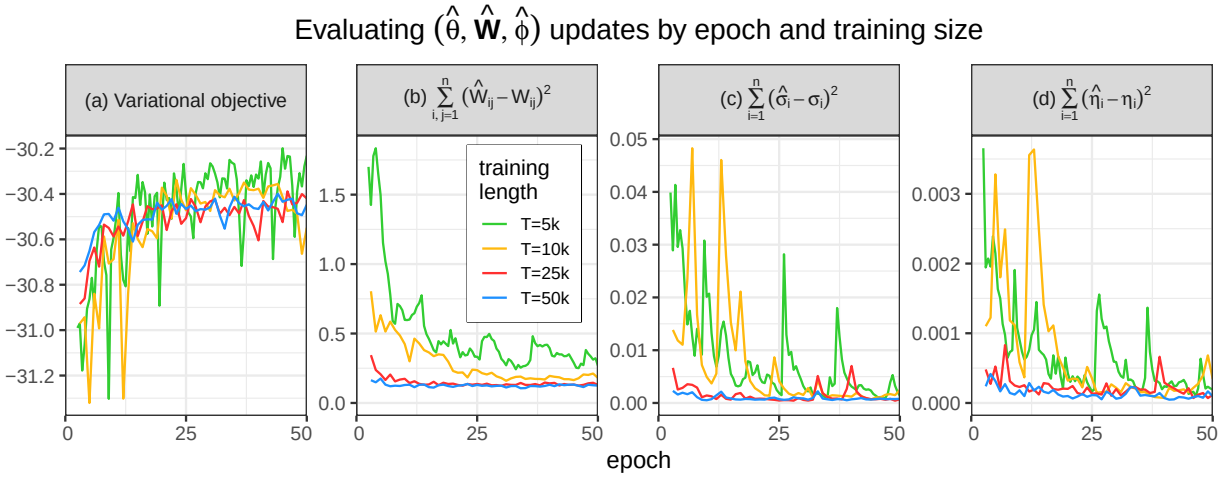

Figure 3.5: Measures evaluating our variational computations by epoch and training size: (a) shows the variational objectives $\tilde{\mathcal{L}}(\hat{\theta}^{(k)}, \hat{\mathbf{W}}^{(k)}, \hat{\phi}^{(k)}; \mathcal{X}_T)/|\mathcal{X}_T|$ averaged over the number of observed inter-spike periods included in each training set. Panels (b), (c) and (d) show sums of squared errors of the estimates for the latent network edges and model parameters.

Figure 3.5 compares four measures by epoch for the training periods of size $T = 5$k, $T = 10$k, $T = 25$k and $T = 50$k. In panel (a) we see increasing variational objectives, with much better apparent convergence for the larger training lengths. Panels (b), (c) and (d) assess the squared loss of evolving estimates of the inferred model parameters based on their known ground truth. The estimates for $T = 25$k and $T = 50$k show much better performance, with the sum of squared errors for $\hat{\mathbf{W}}$ (over all $20 \times 19 = 380$ non-diagonal edges) settling away from to zero, to around .125, and the squared error loss for the $\hat{\sigma}_i$s and $\hat{\eta}_i$s approaching zero.

We compute the node signal-to-noise ratios $\text{SNR}_i^{\hat{\mathbf{W}},\mathbf{Y}}$ based on the network estimates and data observations from the $T = 50$ fit. Figure 3.7 (a). compares these to the $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$ values computed with the true network adjacency matrix $\mathbf{W}$ (shown in Table 3.2). The computations match up quite closely, with slightly lower values for the SNRs based on $\hat{\mathbf{W}}$. Because the variational approximation does a good job of recovering $\mathbf{W}$, it makes sense that the signal to noise ratios computed based on its estimates $\hat{\mathbf{W}}$ are close to those calculated using $\mathbf{W}$.

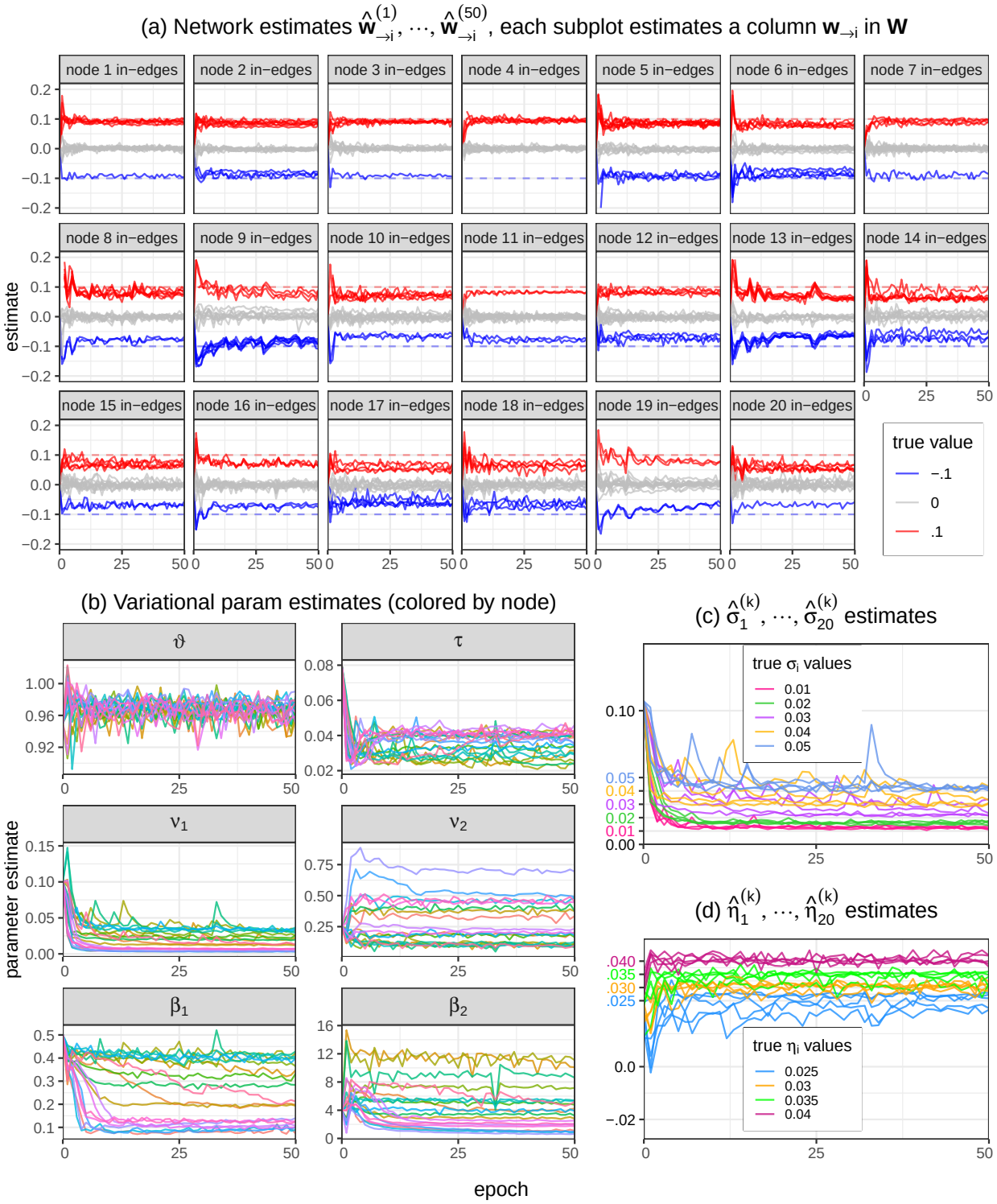

Figure 3.6: Results of computing our variational approximation q_ϕ to the simulated data with $T = 50$k observations. We show the sequential updates of the $(\hat{\theta}, \hat{\mathbf{W}}, \hat{\phi})$ estimates by epoch and node, indicating known ground truth values for $\mathbf{W}$ and θ. Panel (a) shows the evolving edge estimates by target node, ordered in descending order by their computed signal-to-noise ratios $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$.

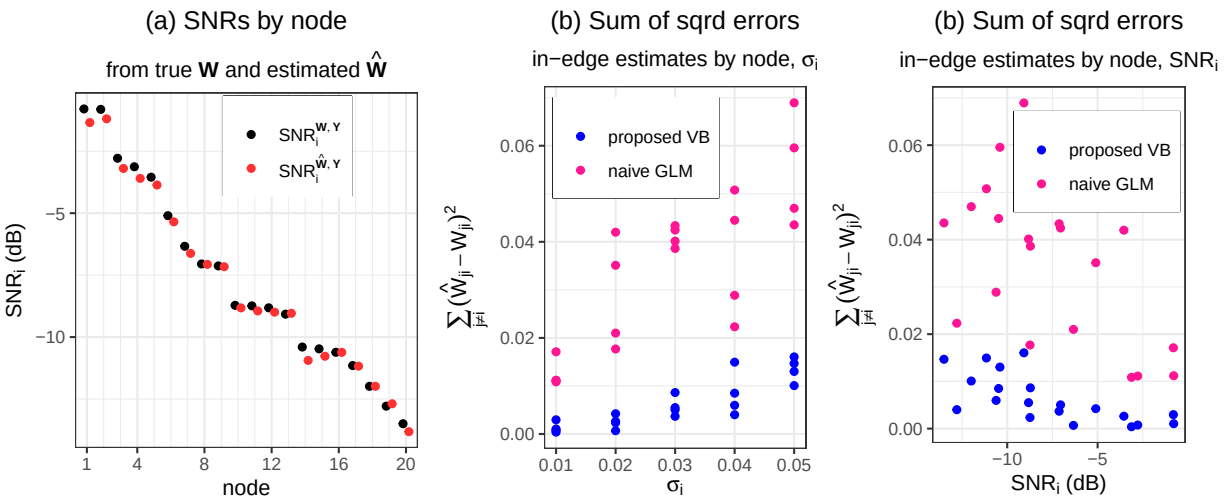

Figure 3.7: Panel (a) compares nodal signal-to-noise ratios calculated based on the true model parameters vs. the estimated parameters from our variational fit. Plots (b) and (c) compare the squared errors of in-edge estimates by node from the variational fit ('proposed VB') and from fitting M^i_{GLM} with stepwise AIC ('naive GLM'). The errors are summed by node and plotted against σ_i and SNR_i. Results are shown for training length $T = 50$k.

Comparison methods We compare our network inference results to results from two other methods: transfer entropy (TE) and a naive logistic regression model.

Transfer entropy is a popular model-free method for inferring pairwise directed links in multivariate time series data that has been successfully applied to neuronal spike train data in particular [67, 42]. TE is a unsigned causal indicator measuring the directed transfer of information between two random processes. We use [68] to estimate transfer entropy between pairs of columns $\mathbf{Y}_i$ and $\mathbf{Y}_j$ in $\mathbf{Y}$.

For the second comparison method we consider the following logistic regression model for node i's recorded activity:

$$M^i_{\text{GLM}} \;:\; \text{logit}(\Pr(\mathrm{y}_{t,i} = 1)) = \mathbf{b}^\intercal \sum_{r=s}^{t-1} \delta^{t-r-1} \mathbf{y}_{r,-i} + \mathrm{b}_1 \textstyle\sum_{s=1}^{r} \delta^{r-s} + \mathrm{b}_0$$
$$\text{where } t = t^i_k + r \le t^i_{k+1} \text{ for some } (i,k) \in \mathcal{X}.$$

This model is extends the 'null' model M^i_0 in section 3.3.9 by including terms for the accumulated

effects of spikes from the other nodes in the network. Under M^i_{GLM} we assume $\sigma_i = 0$ (hence the model is 'naive'). We would expect this model to be able to recover network structure when σ_i is close to zero, with deteriorating performance as σ_i increases.

This is borne out in Figure 3.7 (b), where the squared errors of the network estimates from M^i_{GLM} increase steeply for nodes with higher σ_i parameters. The squared error of our $\hat{\mathbf{W}}_{ij}$ estimates also increase with σ_i but less steeply and remains well below the naive GLM errors. Figure 3.7 (c) shows these squared errors against the nodal signal-to-noise ratios. We see that the estimates get worse as the nodal SNR decreases — this makes sense! Note that in plots (b) and (c) the squared errors are summed over the in-edges for each node.

Table 3.3: Variational network recovery compared to transfer entropy and naive GLM methods

	T = 1k		T = 5k		T = 10k		T = 50k		T = 50k	
Method	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR	FPR	FNR
proposed VB fit	.383	**.214**	**.089**	.062	**.038**	**.007**	**.004**	**0**	**0**	**0**
TE with α = .01	**.013**	.790	.015	.469	.019	.276	.045	.083	.079	.021
TE with α = .05	.074	.645	.051	.303	.091	.138	.143	.024	.202	.007
M^i_{GLM}+stepwise AIC	.357	.276	.243	**.021**	.251	**.007**	.315	**0**	.302	**0**
Method	ER	SSE	ER	SSE	ER	SSE	ER	SSE	ER	SSE
proposed VB fit	.318	**5.08**	**.079**	**.269**	**.026**	**.181**	**.003**	**.130**	**0**	**.125**
TE with α = .01	.309		.188		.117		.059		.057	
TE with α = .05	**.292**		.147		.109		.097		.128	
M^i_{GLM}+stepwise AIC	.326	23.2	.158	.656	.158	.685	.195	.697	.187	.696

FPR: false positive rate, FNR: false negative rate, ER: overall error rate
SSE: sum of squared errors

In Table 3.4.2 we compare the network recovery results of our variational fit with the results obtained with transfer entropy and by fitting M^i_{GLM} for the five training set sizes T = 1k, T = 5k, T = 10k, T = 25k and T = 50k.

We report the false positive (FPR), false negative (FNR) and overall error rates (ER) for inferring whether a given directed edge $(i, j) \in \mathcal{E}$ (ie $\mathbf{W}_{ij} \neq 0$). We note there are 145 nonzero entries in $\mathbf{W}$ and 235 off-diagonal zeros.

For our estimate $\hat{\mathbf{S}}$ of the existing network edges $\text{sign}(\mathbf{W}) = \mathbf{S}$ in the latent network, we use Algorithm 3.1 with $k^* = 25$, $c^* = 3$ and $m^* = .05$, applying our proposed criterion for determining

significant edges from the results of our variational computation. We fit the naive GLM model with a stepwise AIC algorithm. By performing variable selection we are able to eliminate edges from the full model M^i_{GLM}. We compute the error rates of the signed edges inferred by these two methods as well as the squared error of their estimates of $\mathbf{W}$.

With transfer entropy (TE) we only can only infer unsigned edges. We use two different significance levels $\alpha = .01$ and $\alpha = .05$ for determining whether the estimated transfer entropy between pairs of columns $\mathbf{Y}_i$ and $\mathbf{Y}_j$ indicates an edge.

The results show our method's superiority for inferring $\mathbf{W}$ in this simulation study. Based on Table 3.4.2 we note that our method falsely infers that a non-edge exists more often than it it classifies a true edge as non-existent. The same is the case for the M^i_{GLM}+stepwise AIC method, to a much greater extreme.

The opposite is true for transfer entropy, which has much lower false positive rates than false negative rates when applied to our simulated data.

3.4.3 Diagnostic checking

We evaluate of the fit of the computed variational approximations by comparing $q_{\hat{\phi}}$ with ground truth simulated data as well as with the conditional posterior predictive distribution $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$ which we would like $q_{\hat{\phi}}$ to be close to and which we can draw samples from. We also use a Pareto Smoothed Importance Sampling (PSIS) diagnostic [69] to evaluate our variational approximation and computation.

We focus on the $T = 50$k fit. The issues we note are generally more pronounced for the fits with fewer training observations. For these fits the convergence seems worse and the computed variational approximation appears further from its target.

Convergence of variational objective Figure 3.8 shows the evolving variational objectives $\tilde{\mathcal{L}}(\hat{\theta}_i^{(\ell)}, \hat{\mathbf{w}}_{*i}^{(\ell)}, \hat{\phi}_i^{(\ell)}; \{(j,k) \in \mathcal{X}^{50\text{k}} : j = i\})$ computed separately over the nodes' observed inter-spike periods in the training data after each epoch $\ell = 1, \ldots, K$. These plots indicate convergence

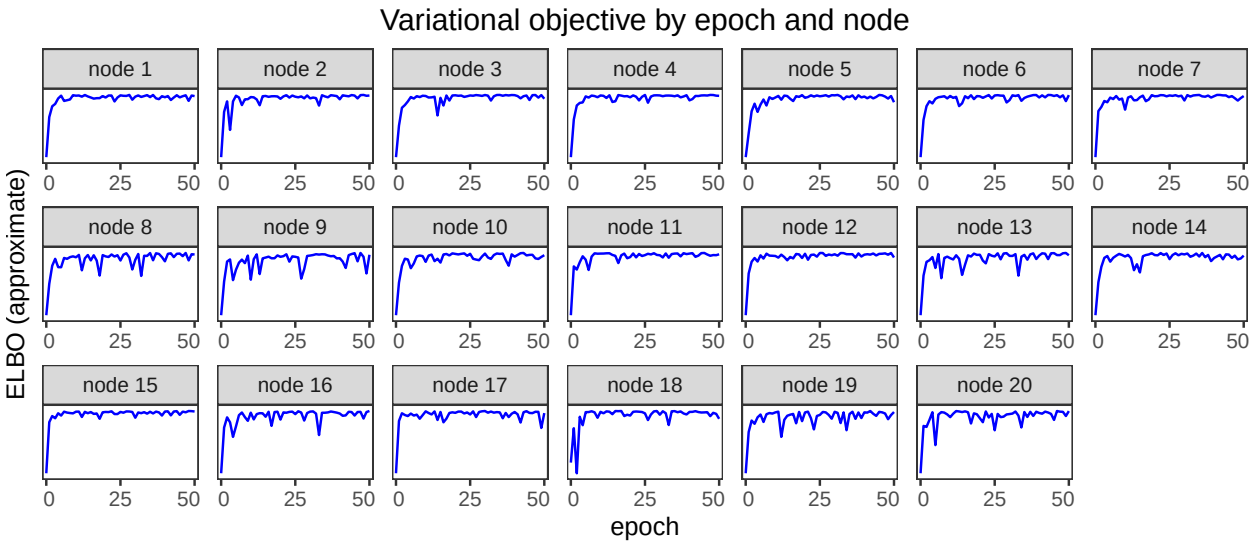

Figure 3.8: Evolving variational objectives computed over each node's observed inter-spike periods in the training data (T = 50k). Nodes are ordered by descending computed signal-to-noise ratio $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$.

for the (lower) nodes with higher computed signal-to-noise ratios $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$. In general, for higher nodes (with lower $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$), the evolving variational objectives do not settle as well, indicating worse convergence. Running the algorithm for more epochs could improve this.

PSIS diagnostic Pareto smoothed importance sampling (PSIS) is a diagnostic method that assesses the quality of an entire variational posterior [69, 70], quantifying the discrepancy between the approximate and the true distribution by the estimated continuous $\hat{k}$ value.

Figure 3.9 shows diagnostic $\hat{k}$ values by training inter-spike interval for each node. We note that $\hat{k} = .5$ and $\hat{k} = .7$ are the thresholds given in [69, 70] for determining if a variational approximation is close enough to its conditional posterior target. The subplots in Figure 3.9 show a good fit for some nodes (1-7,11) and a very poor fit for others (9,13-20), suggesting that our variational approximation $q_{\hat{\phi}}$ is not always close to its target $p(\mathbf{Z}|\mathbf{Y})$. It seems our approximation may be too parsimonious to get close to $p(\mathbf{Z}|\mathbf{Y})$ in certain cases, with not enough degrees of freedom to adapt to variation in the latent fluctuation paths $\mathbf{z}_{(i,k)}$.

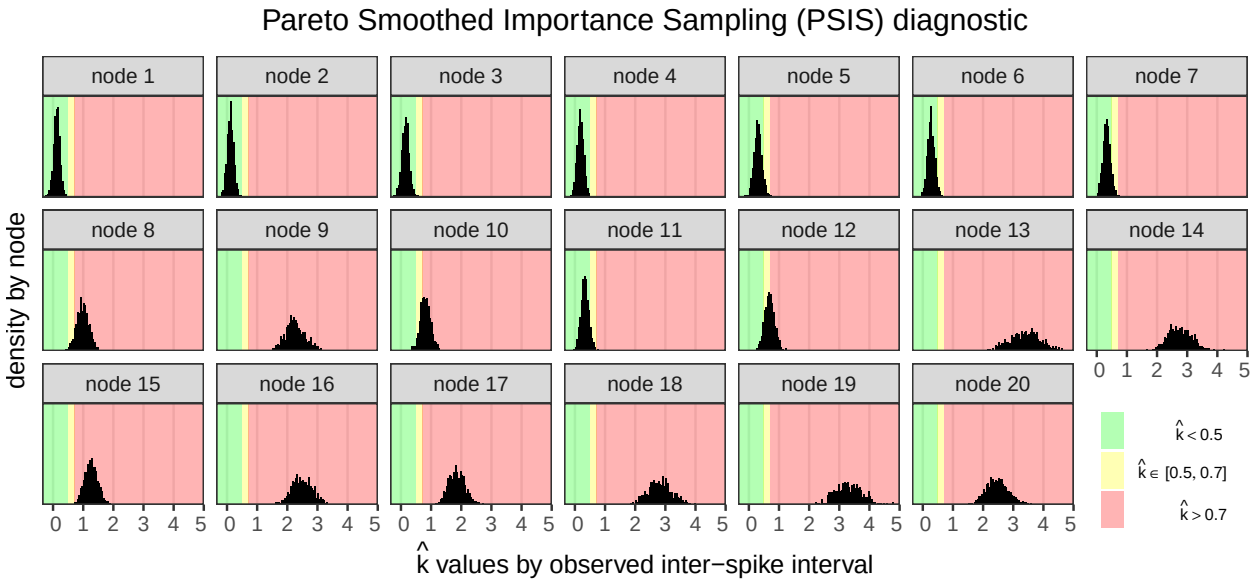

Figure 3.9: Pareto Smoothed Imporance Sampling $\hat{k}$ values for our computed variational approximation to the simulated training set of length $t = 50$k. The histograms show the diagnostic $\hat{k}$ values by training inter-spike interval for each node. $\hat{k} = .5$ and $\hat{k} = .7$ are the thresholds given in [69, 70] for determining if a variational approximation is close enough to its conditional posterior target.

Fits of key latent 'midpoint' quantity We look at the fits of the latent quantities $\mathrm{v}^*_{i,\,k} = \frac{1}{2}(\mathrm{v}_{i,t^i_{k+1}-1} + \mathrm{v}_{i,t^i_{k+1}})$, the midpoints between the spiking voltages $\mathrm{v}_{i,\,t^i_{k+1}}$ and the voltages one step before $\mathrm{v}_{i,\,t^i_{k+1}-1}$. Recall that we have constructed $\mu_{\phi_i}(i, k, \mathbf{Y})$ and $\Sigma_{\phi_i}(\Delta t^i_k)$ so that, under $q_{\phi_i|\mathbf{Y}}$, $\mathrm{v}^*_{i,\,k}$ follows a homogeneous normal distribution with mean ϑ_i and variance τ_i^2. This assumption (3.19) is a key aspect of our variational approximation q_ϕ.

Figure 3.10 shows the histograms of the latent $\mathrm{v}^*_{i,\,k}$ values from our simulated data, along with the estimated densities $\mathrm{N}(\hat{\vartheta}_i, \hat{\tau}_i^2)$ from our variational computation. Looking at the histograms by node, we note the reasonableness of our key assumption (3.19). The estimated fits roughly align with the latent values, with better agreement for lower nodes with higher computed SNRs. However, there appears to be a downward bias on our estimates of $\hat{\vartheta}_i$. The term $-\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} h(\mathrm{v}_{t,\,i}) \approx -\mathbb{E}_{q_{\phi_i|\mathbf{Y}}} \log(1 + \exp(\kappa(\mathrm{v}_{t,\,i} - 1)))$ in our variational objective heavily penalizes distributions under $q_{\phi_i|\mathbf{Y}}$ that give weight to higher (>1) values of $\mathrm{v}_{t,\,i}$ (see Figure 3.3 (a), (c)). It seems this penalization may not be properly balanced by the contributions from the expected spiking voltages in the terms

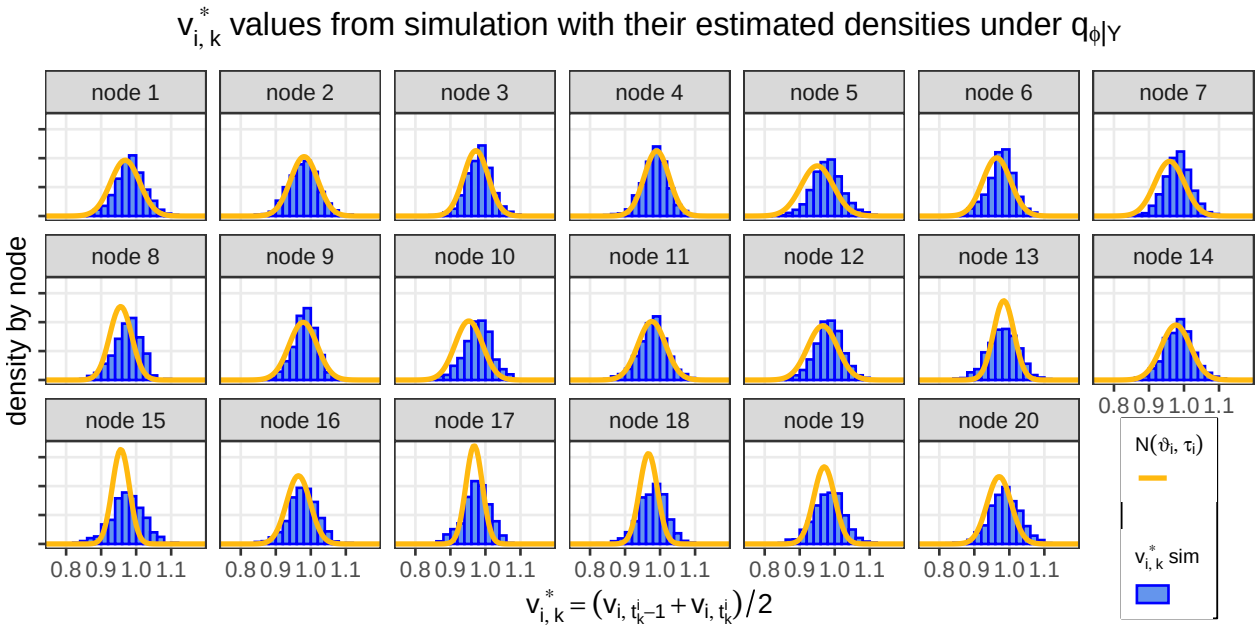

Figure 3.10: Histograms of the latent 'midpoint' quantity $v^*_{i,k}$ by node from our simulated data with cutoff $T = 50$k, shown with the associated densities $N(\hat{\vartheta}_i, \hat{\tau}_i^2)$ from our computed variational approximation $q_{\hat{\phi}}$.

$\kappa \mathbb{E}_{q_{\phi_i|\mathbf{Y}}} v_{t^i_{k+1},i}$, leading to apparent bias for $\hat{\vartheta}_i$ along with other estimates. In future work, we are interested in exploring more flexible variational approximations to reduce these biases.

Comparing $q_{\hat{\phi}|\mathbf{Y}}$ to $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$ In computing the variational approximation $q_{\hat{\phi}|\mathbf{Y}}$ we approximately minimize the KL divergence between the variational family of distributions $q_{\phi|\mathbf{Y}}$ and the target posterior conditional distribution $p_{\theta,\mathbf{W}}(\mathbf{Z}|\mathbf{Y})$. If these distributions are close then $q_{\hat{\phi}}$ will also be close to the posterior predictive distribution $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$. In the following diagnostics, we compare $q_{\hat{\phi}}$ to $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$.

Using an acceptance-rejection (AR) sampling method we obtain draws $\mathbf{Z} \sim p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$. Each inter-spike interval is sampled separately. For $(i,k) \in \mathcal{X}$, we repeat the following until the desired

number of draws are obtained:

$$\begin{aligned} \text{draw:} \quad & \mathbf{z} \sim \varphi(\mathbf{z}) = \mathrm{N}\left(0,\ \hat{\sigma}_i^2 \mathbf{I}_{\Delta t_k^i}\right) \\ \text{and} \quad & u \sim \mathrm{Unif}(0, 1] \\ \text{accept if:} \quad & u\varphi(\mathbf{z}) \leq p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{z}, \mathbf{Y}) \\ \Leftrightarrow \quad & \log u \leq \kappa \mathbf{v}_{\Delta t_k^i} - \sum_{s=1}^{\Delta t_k^i} \log\left(1 + e^{\kappa(\mathbf{v}_s - 1)}\right), \end{aligned}$$

where $\mathbf{v}$ is the corresponding latent voltage path calculated from the candidate draw $\mathbf{z}$ along with our estimates $\hat{\theta}$ and $\hat{\mathbf{W}}$.

We compare the distributions $q_{\hat{\phi}|\mathbf{Y}}$ and $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$ by calculating the Mahalanobis distances [71, 72] between accepted AR draws and our computed variational approximation. For $(i, k) \in \mathcal{X}$, the Mahalanobis distance $\mathrm{d}_M\left(\mathbf{z}, q_{\hat{\phi}_i|\mathbf{Y}}\right)$ between $\mathbf{z} \in \mathbb{R}^{\Delta t_k^i}$ and the computed variational approximation $q_{\hat{\phi}_i|\mathbf{Y}}$ is given by

$$\mathrm{d}_M\left(\mathbf{z}, q_{\hat{\phi}_i|\mathbf{Y}}\right) = \sqrt{\left(\mathbf{z} - \mu_{\phi_i}(i, k, \mathbf{Y})\right)^\intercal \Sigma_{\phi_i}^{-\mathbf{1}}(\Delta t_k^i)\left(\mathbf{z} - \mu_{\phi_i}(i, k, \mathbf{Y})\right)}.$$

These Mahalanobis distances satisfy the following property:

$$\mathbf{z} \sim q_{\hat{\phi}_i|\mathbf{Y}} \quad \Rightarrow \quad \mathrm{d}_M\left(\mathbf{z}, q_{\hat{\phi}_i|\mathbf{Y}}\right) \sim \chi^2_{\Delta t_k^i} \tag{3.46}$$

The Figure 3.11 (a) subplots show the Mahalanobis distances between draws $\mathbf{z} \sim p_{\hat{\theta},\hat{\mathbf{W}}}$ and our computed approximation $q_{\hat{\phi}|\mathbf{Y}}$ for three intervals in the $T = 50k$ training set — including the intervals with the lowest and highest PSIS $\hat{k}$ values. We see some larger distances than we would expect under the reference $\chi^2_{\Delta t}$ densities (3.46).

In the Figure 3.11 (b) and (c) subplots, we show simulated draws of the latent fluctuations and latent voltage paths, respectively, from $p_{\hat{\theta},\hat{\mathbf{W}}}$ and $q_{\hat{\phi}}$, along with the true latent values (in red). These plots show general agreement along with some interesting differences (in apparent variation)

between $q_{\hat{\phi}|\mathbf{Y}}$ and $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$. In the middle column, the true latent fluctuations $\mathbf{z}_{(i=10,k=815)}$ are 'far' from $q_{\hat{\phi}_i|\mathbf{Y}}$. However, we can see how the latent voltages $\mathbf{v}_{(i,\,k)}$ align well with our computed variational approximation at the end of the inter-spike interval (where it really counts).

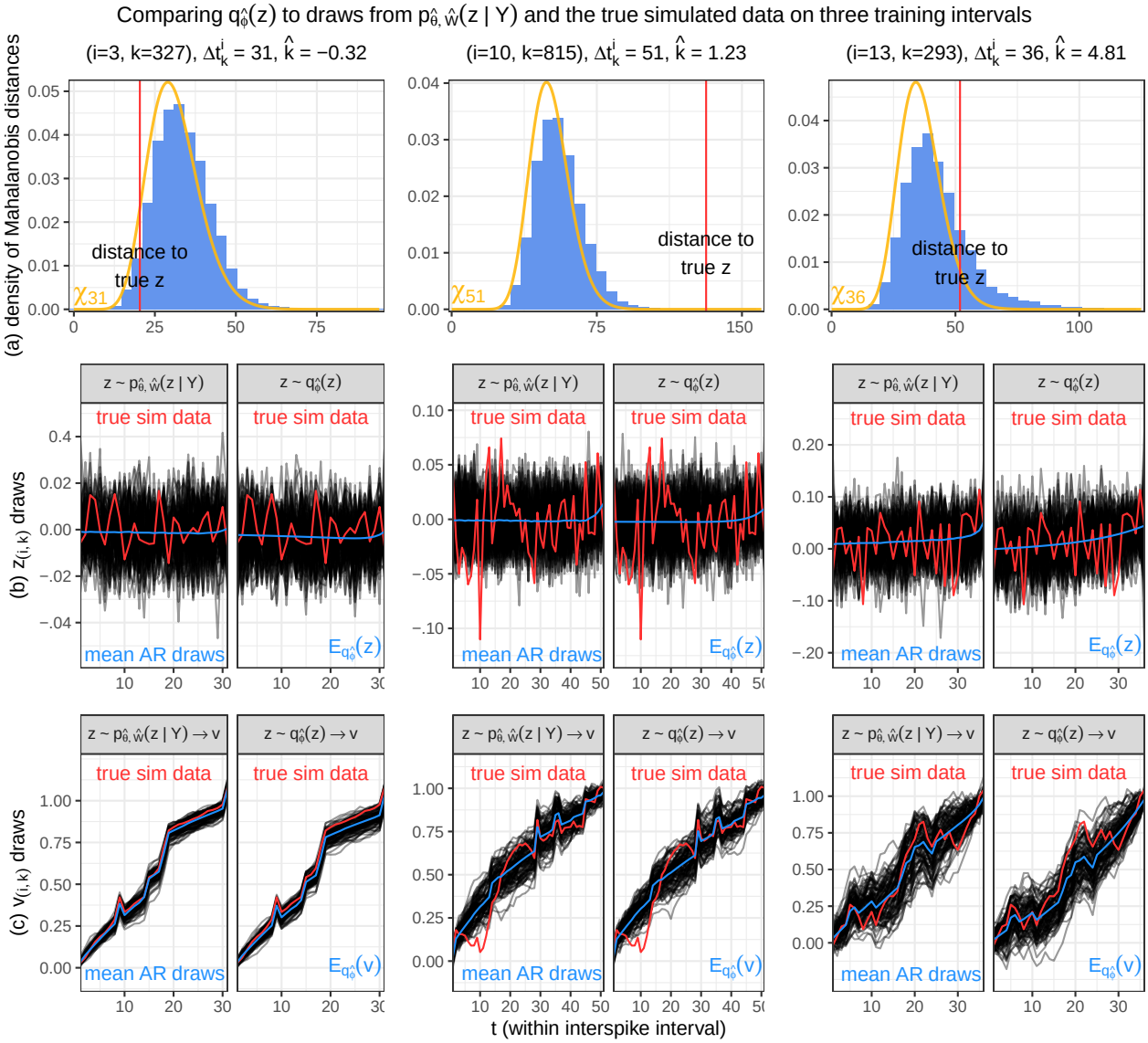

Figure 3.11: Comparing $p_{\hat{\theta},\hat{\mathbf{W}}}(\mathbf{Z}|\mathbf{Y})$ and $q_{\hat{\phi}|\mathbf{Y}}(\mathbf{Z})$ on three intervals in the $T = 50k$ training set. The top row (a) shows the Mahalanobis distances between draws $\mathbf{z} \sim p_{\hat{\theta},\hat{\mathbf{W}}}$ and $q_{\hat{\phi}|\mathbf{Y}}$, along with the reference $\chi^2_{\Delta t}$ densities (3.46). The vertical red lines show the distances involving the true latent fluctuations. Rows (b) and (c) show simulated draws of the latent fluctuations and latent voltage paths, respectively, from $p_{\hat{\theta},\hat{\mathbf{W}}}$ and $q_{\hat{\phi}}$, along with the true latent values (in red). We show the inter-spike intervals with the lowest (first column) and highest (third column) PSIS $\hat{k}$ values.

3.4.4 Reconstructing events of held-out nodes

We consider the situation described in section 3.3.8 where events from a node i^* become unobservable at some point in the observation period while the events from other nodes continue to be recorded. Manufacturing this situation in our simulated data, we apply our forward-backward point estimate procedure and see how well we recover the held-out activity.

For each of the three largest training cutoffs $T = 10\text{k}$, $T = 25\text{k}$ and $T = 50\text{k}$, we make held-out predictions for the testing periods extending to time $T + 2\text{k}$.

Bringing our notation from section 3.3.8 into this context, we have complete training data $\mathbf{Y}^{\text{train}} \in \{0\,,1\}^{T\times n}$ to which we have fit our variational approximation and obtained model estimates $(\hat{\theta}, \hat{\mathbf{W}})$. For each node $i^* \in \{1, \ldots, n\}$ we imagine it stops being observable from time $T + 1$, with its last recorded event at time $t^{i^*}_{m_{i^*}}$. The held-out testing period is from $T^* = t^{i^*}_{m_{i^*}}$ to $T + 2\text{k}$, with length $S = T + 2\text{k} - t^{i^*}_{m_{i^*}}$. In this period we observe $\mathbf{Y}^{\text{test}}_{-i^*} = (\mathrm{y}_{T^*+1,i^*}, \ldots, \mathrm{y}_{T^*+S,i^*}) \in \{0\,,1\}^{S\times n-1}$.

The problem is to infer the held-out observations $\mathbf{Y}^{\text{test}}_{i^*} \in \{0\,,1\}^S$ given the observed data $\mathbf{Y}^{\text{train}}$ and $\mathbf{Y}^{\text{test}}_{-i^*}$. We restrict our attention to the conditional expectation:

$$\mathbb{E}[\mathbf{Y}^{\text{test}}_{i^*} \mid \mathbf{Y}^{\text{train}}, \mathbf{Y}^{\text{test}}_{-i^*}] \in [0, 1]^S. \tag{3.47}$$

The entries of (3.47) are conditional spike probabilities $\Pr(\mathbf{Y}^{\text{test}}_{r,i^*} = 1 \mid \mathbf{Y}^{\text{train}}, \mathbf{Y}^{\text{test}}_{-i^*})$. We estimate these by approximating a posterior predictive conditional probability. At a high level, our approach is the following:

$$\Pr(\mathbf{Y}^{\text{test}}_{r,i^*} = 1 | \mathbf{Y}^{\text{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}}, \mathrm{y}_{T^*,i^*} = 1) \quad \text{estimates} \quad \Pr(\mathbf{Y}^{\text{test}}_{r,i^*} = 1 \mid \mathbf{Y}^{\text{train}}, \mathbf{Y}^{\text{test}}_{-i^*})$$

$$\text{and} \quad \hat{p}_{ri^*} \quad \text{approximates} \quad \Pr(\mathbf{Y}^{\text{test}}_{r,i^*} = 1 \mid \mathbf{Y}^{\text{test}}_{-i^*}, \hat{\theta}, \hat{\mathbf{W}}, \mathrm{y}_{T^*,i^*} = 1)$$

Our proposed 'forward-backward' point estimate $\hat{p}_{ri^*}$ is constructed by approximating the forward and backward components of (3.42). As part of our computations for $\hat{p}_{ri^*}$ we compute the empirically estimated posterior predictive spike probabilities $\tilde{p}^{\text{na}}_r$ (3.40). We refer to $\tilde{p}^{\text{na}}_r$ as the

'non-anticipating' probability estimate and compare these with our 'forward-backward' estimates $\hat{p}_{ri^*}$.[6]

Algorithm 3.2 Simple bootstrap estimate $\hat{\mathbf{f}}^B$ for (3.47)

Require: training inter-spike period lengths $\{\Delta t_1, ..., \Delta t_m\}$
Require: test length S
Require: number of bootstrap samples B
  Initialize $\mathbf{x}_r \leftarrow 0, r = 1, ..., S$ ▷ $\mathbf{x}$ keeps track of counts at each time in test period
  for $b \in \{1, ..., B\}$ **do**
    $R \leftarrow 0$
    while $R < S$ **do**
      $\Delta t \leftarrow \text{Unif}\{\Delta t_1, ..., \Delta t_m\}$ ▷ randomly draw training length with replacement
      $R \leftarrow R + \Delta t$
      if $R < S$ **then**
        $\mathbf{x}_R \leftarrow \mathbf{x}_R + 1$ ▷ increment stored count with 'observed spike'
      end if
    end while
  end for
  return $\frac{1}{B}\mathbf{x}$ ▷ the procedure returns the simple bootstrap estimate $\hat{\mathbf{f}}^B$

Evaluation and comparison Consider a general estimate $\hat{\mathbf{f}} \in (0, 1)^S$ for (3.47) such that

$$\hat{\mathbf{f}}_r = \hat{f}(r, \mathbf{Y}^{\text{train}}, \mathbf{Y}^{\text{test}}_{-i^*}) \text{ estimates } \Pr(\mathbf{Y}^{\text{test}}_{r,i^*} \mid \mathbf{Y}^{\text{train}}, \mathbf{Y}^{\text{test}}_{-i^*}).$$

We may evaluate $\hat{\mathbf{f}}$ by its cross entropy loss computed on the true held out data $\mathbf{Y}^{\text{test}}_{i^*}$:

$$\text{L}^{\text{CE}}(\mathbf{Y}^{\text{test}}_{i^*}, \hat{\mathbf{f}}) = \sum_{r=1}^{S} \mathbf{Y}^{\text{test}}_{r,i^*} \log \hat{\mathbf{f}}_r + (1 - \mathbf{Y}^{\text{test}}_{r,i^*}) \log(1 - \hat{\mathbf{f}}_r).$$

We compute this cross entropy loss for our forward backward estimates as well as for our non-anticipating probability estimates. We compare these to the loss of the actual held-out probabilities $\mathbf{p}^{\text{true}}$ given by

$$\mathbf{p}^{\text{true}}_r = \Pr(\mathbf{Y}^{\text{test}}_{r,i^*} = 1 \mid \mathbf{V}^{\text{test}}_{r,i^*}) = \text{logit}^{-1}(\kappa(\mathbf{V}^{\text{test}}_{r,i^*} - 1)),$$

as well a very simple nonparametric bootstrap estimate $\hat{\mathbf{f}}^B$ based on just the observed inter-spike

[6] We use N_1 =2,000 and N_2 =1,000 Monte Carlo draws in our computations when sampling from (3.39) and (3.44), respectively

intervals in the training data. The procedure for obtaining $\hat{\mathbf{f}}^B$ is given in Algorithm 3.2.

The cross entropy loss of the true probabilities acts as a reference ceiling while $\mathrm{L}^{\mathrm{CE}}(\mathbf{Y}^{\mathrm{test}}_{i^*}, \hat{\mathbf{f}})$ provides us a reference floor, a minimally acceptable standard to beat.

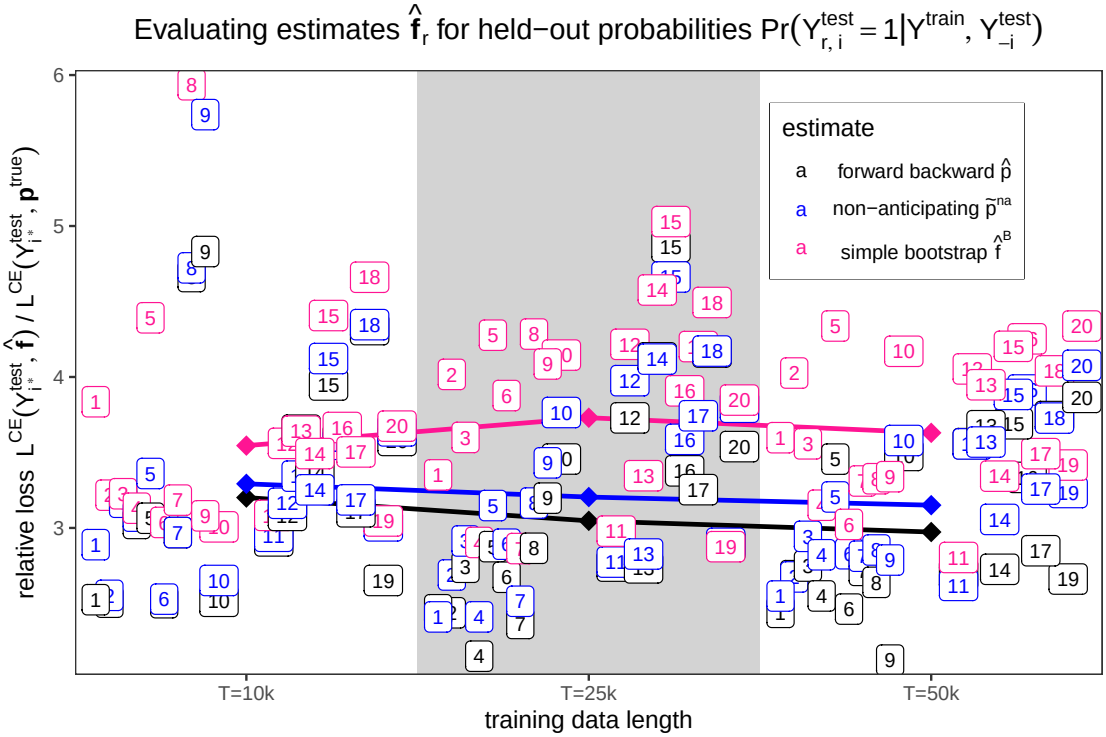

Figure 3.12: Relative cross entropy loss of our proposed forward backward and comparison estimates for $\Pr(\mathbf{Y}^{\mathrm{test}}_{r,i^*} = 1 \mid \mathbf{Y}^{\mathrm{train}}, \mathbf{Y}^{\mathrm{test}}_{-i^*})$, shown by held-out node and training length. The lines show the relative loss across all $n = 20$ nodes for a given training period.

Figure 3.12 provides a high-level summary of the results of the held-out predictions by held-out node and training period. This figure shows the computed cross entropy losses of the three estimates we have discussed relative to the cross entropy loss of the true probabilities:

$$\text{(i.)} \quad \frac{\mathrm{L}^{\mathrm{CE}}(\mathbf{Y}^{\mathrm{test}}_{i^*}, \hat{\mathbf{f}})}{\mathrm{L}^{\mathrm{CE}}(\mathbf{Y}^{\mathrm{test}}_{i^*}, \mathbf{p}^{\mathrm{true}})} \qquad \text{(ii.)} \quad \frac{\sum_{i^*=1}^{n} \mathrm{L}^{\mathrm{CE}}(\mathbf{Y}^{\mathrm{test}}_{i^*}, \hat{\mathbf{f}})}{\sum_{i^*=1}^{n} \mathrm{L}^{\mathrm{CE}}(\mathbf{Y}^{\mathrm{test}}_{i^*}, \mathbf{p}^{\mathrm{true}})}.$$

The labelled points show ratio (i.) for a single held-out node, while the darker lines show ratio (ii.) of the losses calculated over all possible held-out nodes.

Our forward backward estimates (shown in black) consistently outperform the other two es-

timates for individual nodes (75% of the time) as well as overall across the three training period lengths. Here is a table showing how often each method outperforms the others in cross entropy loss for single held-out nodes by training length T:

# top performances	T = 10k	T = 25k	T = 50k
forward backward $\hat{p}_{ri^*}$	15	16	14
non-anticipating $\tilde{p}_r^{\text{na}}$	4	3	6
simple bootstrap $\hat{\mathbf{f}}^B$	1	1	0

For each training length T there is a diagonal-upward trend in relative loss attributable to our labeling of nodes based on the reverse order of their signal-to-noise ratios $\text{SNR}_1^{\mathbf{W},\mathbf{Y}} > \ldots > \text{SNR}_{20}^{\mathbf{W},\mathbf{Y}}$.

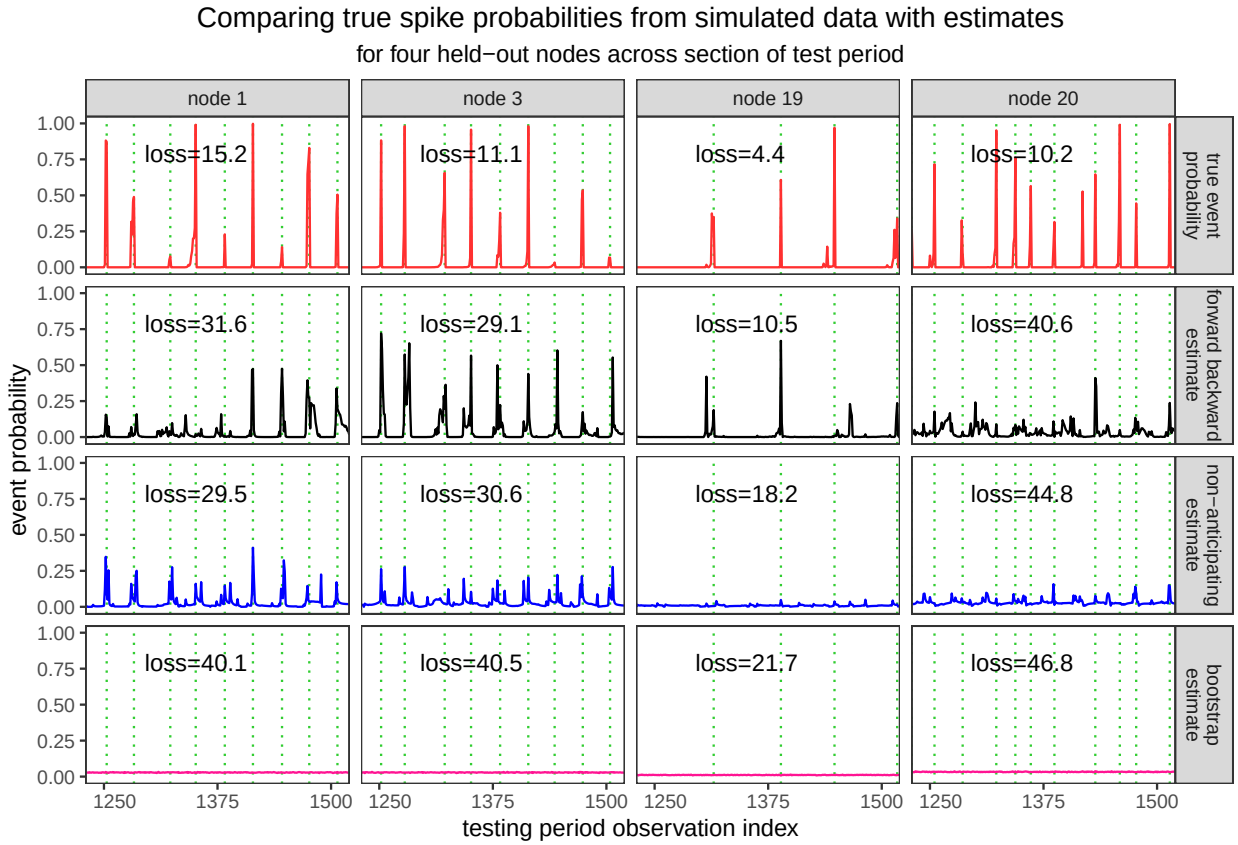

Figure 3.13: Predictive event probability estimates for held-out nodes 1, 3, 19 and 20 across a common test period segment. The true event probabilities in the top row are compared against our proposed forward-backward estimates $\hat{p}_{ri^*}$, the non-anticipating probability estimates $\tilde{p}_r^{\text{na}}$ and the simple bootstrap estimates $\hat{\mathbf{f}}_r$, with the cross entropy loss of each estimate on the plotted segment.

In Figure 3.13 we show the actual estimates $\hat{\mathbf{f}}_r$ for nodes 1, 3, 19 and 20 along with the true probabilities $\mathbf{p}^{\text{true}}$ on a segment of the test period.

3.5 Application to neural activity recording data

We fit our variational approximation to spike-sorted neural activity from the dorsomedial frontal cortex (DMFC) of a rhesus macaque performing a time-interval reproduction task. We infer network connections among 20 neurons. We present the results of our analysis and discuss the shortcomings of the considered model and our variational approach to this data.

3.5.1 Dataset and motivation

The data we analyze is made available as part of the Neural Latents Benchmark (NLB) [73], a project started in 2021 with the aim of coordinating latent variable modeling efforts of neural population activity data. The NLB initiative focuses on unsupervised latent variable modeling that is not directly conditioned on measured external variables. This is in line with our work and in contrast to the more widespread and mature study of supervised learning for task-related neural activity or the neural response to external stimulus in experimental studies.

We chose to look at the DMFC_RSG dataset in particular because the dorsomedial frontal cortex (DMFC) seems to us like a particularly interesting brain region to infer connections in and to analyze in general with interpretable methods like ours. We surmise that its neuronal dynamics hold insights into cognition that go beyond relationships with a subject's observable behavior or experimental treatment.

The DMFC is contained in the dorsal frontal cortex, an area of the primate brain that is studied in both humans and macaques. The human dorsal frontal cortex is associated with the most sophisticated aspects of cognition, including those that are thought to be especially refined in humans. Research links key psychological and behavioral aspects of 'mentalizing' with DMFC functions, including executive inhibition, distinction between self and others, prediction under uncertainty, and perception of intentions [74]. Studies find that the DMFC plays an important role

in social cognition in monkeys as well.

In [75], the authors infer and compare the organization of dorsal frontal cortex in humans and macaques using diffusion-weighted magnetic resonance imaging (DW-MRI) and functional MRI (fMRI). Their results suggest important similarities in frontal cortex organization, even for regions thought to carry out uniquely human functions.

The data we analyze comes from a 2018 study of *in vivo* recordings of neural activity in the DMFC while rhesus macaques performed a time-interval reproduction task [76]. The realeased dataset contains spike sorted data for one monkey from a single recording session, along with 'external' information about the repeated trials. The data contains 4,809,332 binary observations of 54 neurons, with time measured in milliseconds. In this work we look at a small piece of this. Figure 3.14 shows the data we analyze, the first 52,000 observations, corresponding to 52 seconds of recorded neural activity.

Within the considered period there is no missing data. From the experiment information data, we know when each of the repeated time-interval reproduction tasks begins and ends, indicated with green and red lines, respectively in Figure 3.14. There are short breaks between each trial. The rows correspond to individual neurons and each thin vertical line represents an observed (binary) spike.

There is a lot of variation across and within these 54 spike trains. Some neurons have very sparse spiking activity. Other cells seem to have distinct spiking patterns with respect to trial periods. For example neurons 30 and 32 tend to emit increased bursts of spikes around the middle-to-end of the trials.

We find that to compute our variational approximation and fit our considered model we need a certain level of event activity in the input spike trains (columns of $\mathbf{Y}$). In particular our method does not deal well with long inter-spike periods. This has to due with the computational complexity limitations of our inference algorithm (see section 3.3.6) as well the limitations of the leaky integrate-and-fire (LIF) model we assume (see section 3.2.1) which does not accommodate stuttering or bursting neurons. Moreover, it is apparent from looking at Figure 3.14 that a number of our model's assumptions are violated by the data, for example the assumption of a time-independent baseline

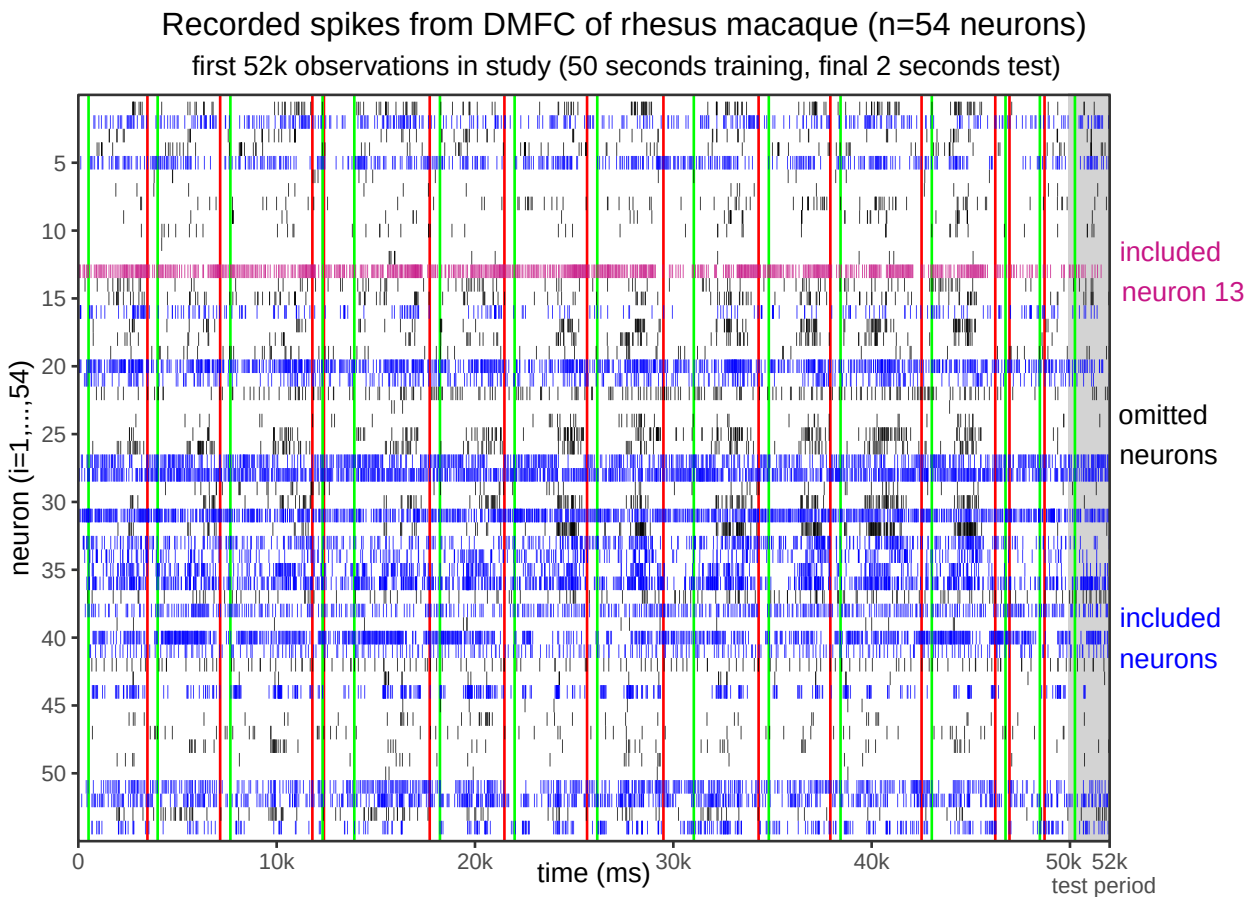

Figure 3.14: Fifty-two seconds of spike sorted neural recording data from the dorsomedial frontal cortex (DMFC) of a rhesus macaque. The green and red vertical lines correspond to the beginning and end, respectively, of single experimental trials. We indicate in blue and magenta the ($n = 20$) spike trains included in our variational computation and analysis. We perform additional analyses involving the highlighted neuron 13, discussed in section 3.5.3 below.

input η_i for each neuron. Right away we can see the need for a more flexible and complex model to properly account for the data's complex dynamics.

In order to apply our method we must filter out some of problematic data and nodes. In particular, we include only the neurons with at least 100 spikes in the first 25,000 milliseconds, omitting entirely any neurons with less. This leaves the $n = 20$ rows indicated in blue and magenta. In addition, for each epoch in our fitting procedure, we assign to mini-batches only the spike intervals with length greater than 1 and less than or equal to 500 ($1 < \Delta t \leq 500$). We make this omission *without* changing any entries of $\mathbf{Y}$. We note that both cases ('double spikes' and very long intervals) are

uncommon among the included neurons.

3.5.2 Results of our methods applied to the DMFC_RSG dataset

To compute our variational approximation and obtain an estimated network among the $n = 20$ included neurons, we must specify values for the parameters κ and δ which we do not fit from the data. We set $\kappa = 50$, as in our simulation study. As mentioned in section 3.2.2, this corresponds to a spiking mechanism that behaves somewhat similarly to a hard threshold at 1. Our particular choice of 50 is, however, arbitrary.

We set δ based on [58], a study of age-related hippocampal dysfunction using *in vitro* recordings of dentate granule cells from young and old rhesus macaques. Along with other intrinsic membrane properties, the paper reports estimates for the membrane time constant τ_m in the range of 20 to 30 ms, without a statistically significant difference between the two age groups. Based on these estimates and the fact that we do not know the age of the monkey recorded in the DMFC_RSG dataset, we set $\delta = (\exp(-1/20) + \exp(-1/30))/2 \approx .960$

With these paramters set, we compute our variational approximation in the same way as in the simulation study.

Figure 3.15 shows the main network inference results from the DMFC_RSG dataset. We note the apparent convergence, in panel (a), of the variational objectives from fitting our variational approximation to $T = 25$k and $T = 50$k observations from the dataset. And we note the apparent agreement between the final estimated adjacency matrices $\hat{\mathbf{W}}$ from the two fits, shown in panel (b).

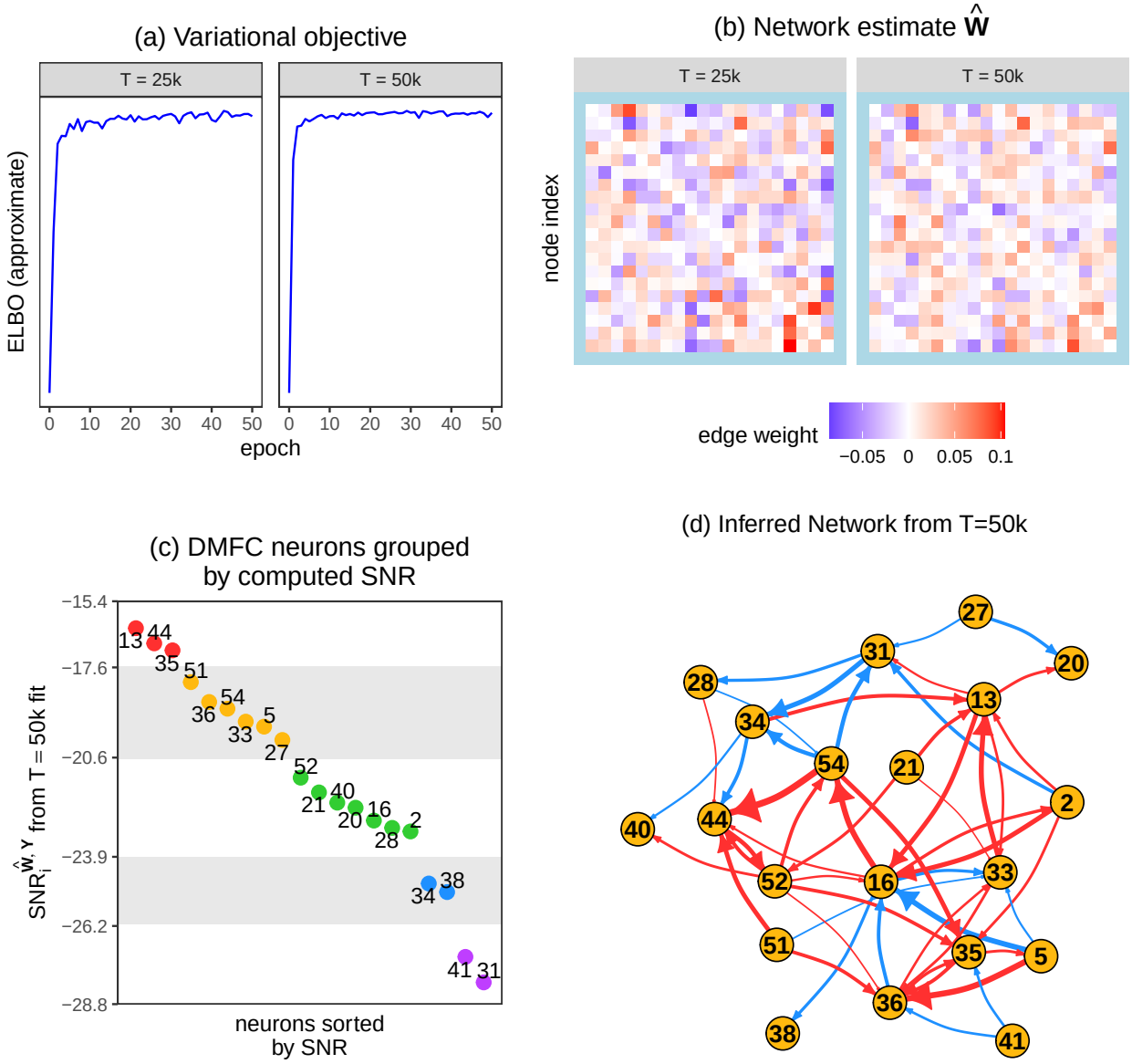

Figure 3.15: **Main network inference results from DMFC_RSG dataset**: panel (a) shows the approximate ELBO $\tilde{\mathcal{L}}(\hat{\theta}^{(k)}, \hat{\mathbf{W}}^{(k)}, \hat{\phi}^{(k)}; \mathcal{X}_T)$ by epoch from fitting our variational approximation to $T = 25$k and $T = 50$k observations from the DMFC_RSG dataset. Panel (b) shows the final estimated adjacency matrices $\hat{\mathbf{W}}$ from the two fits. Panel (c) shows our estimates $\text{SNR}_i^{\hat{\mathbf{W}},\mathbf{Y}}$ of the neurons' signal-to-noise ratios for network effect in descending order, split at the four biggest differences. Panel (d) shows the inferred network among the 20 included neurons, based on our estimates $\hat{\mathbf{W}}$ and $\hat{\mathbf{S}}$. We use our estimate $\hat{\mathbf{S}}$ for $\mathbf{S} = \text{sign}(\mathbf{W})$ to choose which estimated (weighted) edges from $\hat{\mathbf{W}}$ to plot.

Table 3.4 shows a summary of the neurons included in our analysis. Comparing the quanities to Table 3.2 we see much more variability in the number of observed events, with fewer spike overall and longer inter-spike periods. Our computed signal-to-noise ratios based on our estimates $\mathbf{W}$ are much lower than those we computed in the simulation study. Interestingly, the range of $\text{SNR}_i^{\hat{\mathbf{W}},\mathbf{Y}}$ values (-28.1 dB to -13.6 dB) we report here for macaque DMFC neurons is in (uncannily close) alignment with the range of signal-to-noise ratio estimates (-28 dB to -14 dB) for stimulus effects reported in [66] for 13 macaque hippocampal neurons.[7]

Table 3.4: Summary of the 20 neurons from DMFC_RSG dataset included in our analysis

i	$\sum_{t<50\text{k}} y_{t,i}$	$\Delta t_i^{*\dagger}$	$\text{SNR}_i^{\hat{\mathbf{W}},\mathbf{Y}\ddagger}$	$\hat{\sigma}_i$	$\hat{\eta}_i$	$\#\{j : \hat{\mathbf{S}}_{ji} > 0\}$	$\#\{j : \hat{\mathbf{S}}_{ji} < 0\}$	$\sum_{j\neq i} \hat{\mathbf{W}}_{ji}$
2	360	496	-23.06	0.067	0.021	1	0	0.106
5	452	498	-19.57	0.067	0.025	1	0	-0.139
13	774	457	-16.30	0.053	0.032	4	0	0.138
16	217	488	-22.69	0.115	0.002	3	2	0.002
20	898	383	-22.27	0.108	0.020	1	1	0.213
21	304	494	-21.76	0.080	0.016	0	0	0.375
27	613	499	-20.01	0.074	0.027	0	0	-0.062
28	996	281	-22.94	0.085	0.030	0	1	-0.111
31	1064	267	-28.08	0.165	0.014	1	3	-0.243
33	519	478	-19.40	0.118	0.005	3	3	0.139
34	385	496	-24.79	0.131	0.001	0	2	-0.045
35	458	500	-17.03	0.140	0.001	4	1	0.274
36	925	445	-18.75	0.158	0.003	5	1	0.303
38	453	489	-25.07	0.061	0.025	0	1	-0.044
40	850	488	-22.10	0.091	0.030	1	1	0.106
41	260	500	-27.23	0.069	0.018	0	0	-0.080
44	387	498	-16.80	0.143	0.002	5	1	0.156
51	498	458	-18.09	0.070	0.023	0	0	0.177
52	598	477	-21.27	0.119	0.009	2	0	0.156
54	295	452	-18.97	0.114	0.010	2	1	0.052

† $\Delta t_i^* = \max\{\Delta t_k^i : k = 1, ..., m_i\}$ from first $T = 50\text{k}$ observations in dataset

‡ computed signal-to-noise ratio (3.45) based on estimates $\hat{\mathbf{W}}$, given in decibels (dB)

[7] We further note that the range of $\text{SNR}_i^{\mathbf{W},\mathbf{Y}}$ values (-13 to -1 dB) in Table 3.2 roughly aligns with the range (-10 to -3) found for guinea pig auditory cortex neurons in [66].

Figure 3.16: Results of computing our variational approximation q_ϕ to the spike trains of $n = 20$ neurons in the DMFC_RSG dataset based on the first $T = 50$k observations. We show the sequential updates of the $(\hat{\theta}^{(k)}, \hat{\mathbf{W}}^{(k)}, \hat{\phi}^{(k)})$ estimates by epoch and node. The colors correspond to ranges of $\text{SNR}_i^{\hat{\mathbf{W}},\mathbf{Y}}$ vases that we have grouped the nodes into (see Figure 3.15 (c)).

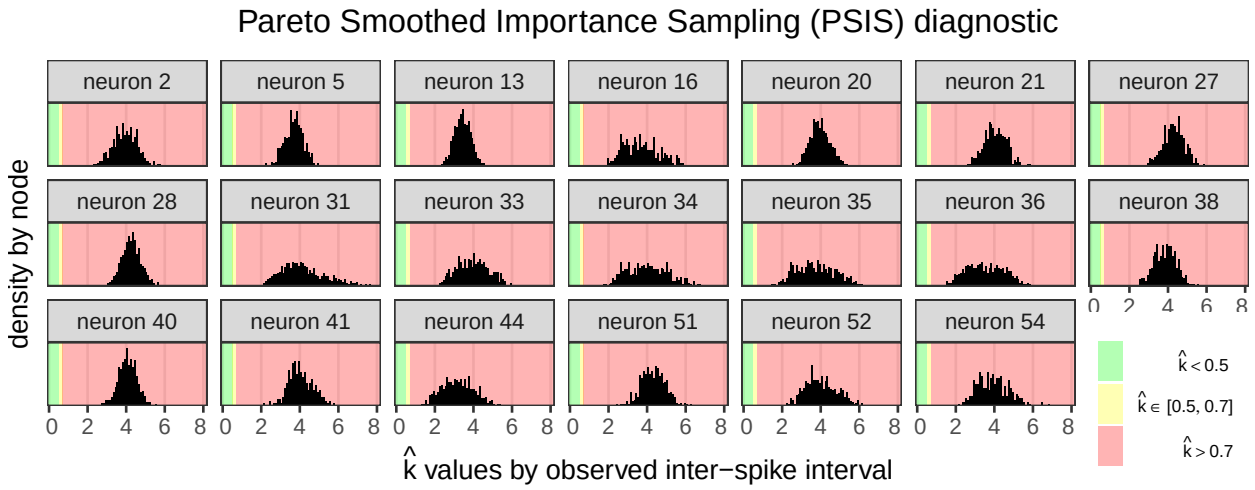

Figure 3.17: PSIS $\hat{k}$ values for our computed variational approximation to the first $T = 50$k observations in the DMFC_RSG dataset. The histograms show the diagnostic $\hat{k}$ values by inter-spike interval for each of the 20 fitted neurons.

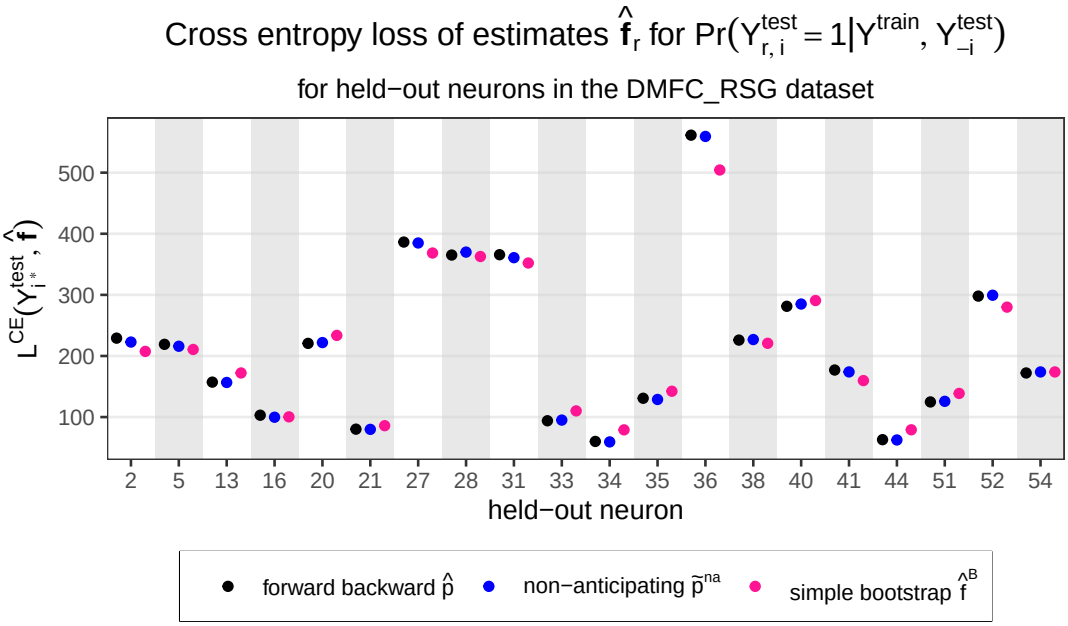

Figure 3.18: Cross entropy loss of our proposed forward backward and comparison estimates for $\Pr(\mathbf{Y}^{\text{test}}_{r,i^*} = 1 \mid \mathbf{Y}^{\text{train}}, \mathbf{Y}^{\text{test}}_{-i^*})$, shown by held-out node. For $T = 50$k training set and $S = 2$k test set (see Figure 3.14)

Figure 3.16 shows the evolving estimates for the latent network $\mathbf{W}$, model parameters θ, and the variational parameters ϕ. The estimates for certain neurons are strongly influenced by the constraints $\eta_i > 0$ and $2\tau_i \geq \sigma_i$ (see the subplot of $\hat{\sigma}_i/\hat{\tau}_i$ ratios). It is clear that the considered model and our proposed variational approximation cannot adequately capture the observed spiking behavior of this subgroup of recorded neurons.

Figure 3.17 shows the Pareto smoothed importance sampling (PSIS) diagnostic $\hat{k}$ values by training inter-spike interval for each neuron. These show a poor fit for all neurons and intervals, suggesting that our variational approximation $q_{\hat{\phi}|\mathbf{Y}}$ cannot get close to its target $p(\mathbf{Z}|\mathbf{Y})$.

For the training cutoff $T = 50$k we make held-out predictions for the testing period extending to time $T + S = 52$k. Figure 3.18 plots the results of the held-out predictions by held-out node, showing the computed cross entropy losses of the three estimates discussed in section 3.4.4. Our forward-backward estimate (shown in black) outperforms the simple bootstrap estimate for only 10

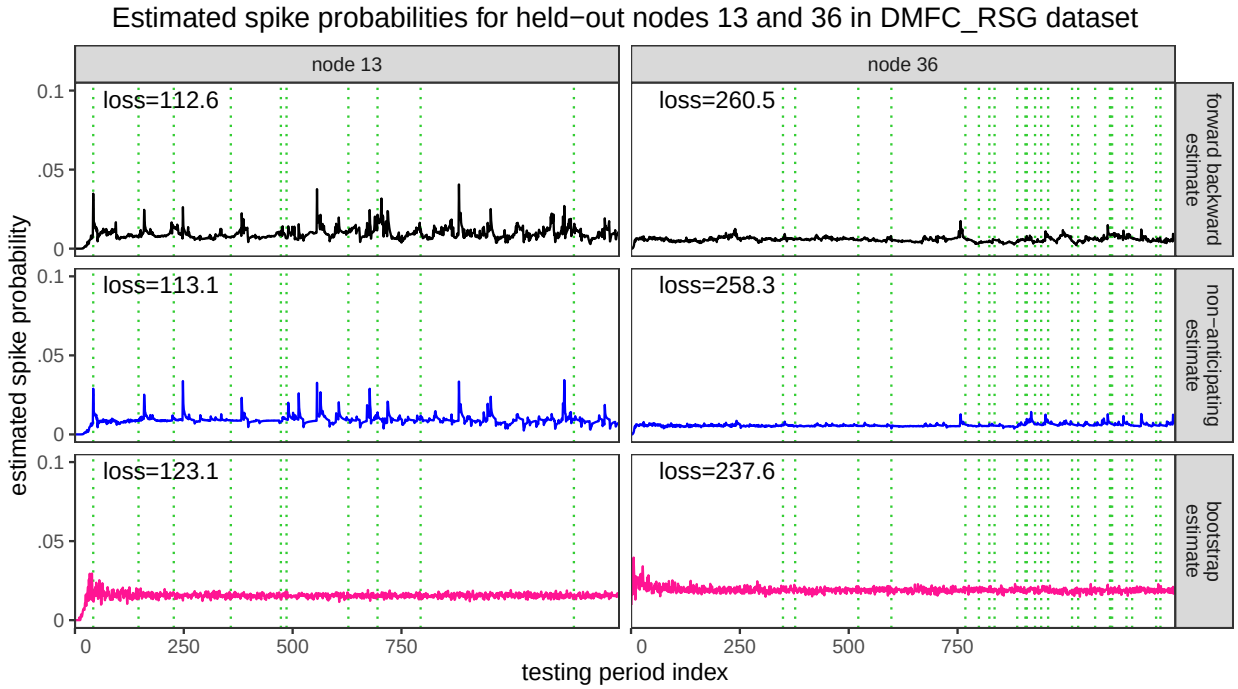

Figure 3.19: Predictive event probability estimates for held-out neurons 13 and 39 on the test period, comparing our proposed forward-backward estimates $\hat{p}_{ri^*}$, the non-anticipating probability estimates $\tilde{p}_r^{\text{na}}$ and the simple bootstrap estimates $\hat{\mathbf{f}}_r$.

(half) of the 20 neurons.

Figure 3.18 shows the predictive event probability estimates for held-out neurons 13 and 39. These appear to be the best and worse fits, respectively.

3.5.3 Shuffling neuron 13's spikes

The results and diagnostics presented and discussed above reveal limitations of our approach when applied to the DMFC_RSG dataset. On the whole they demonstrate the need for expansion and more flexibility in both our assumed model (3.9) and in our variational approach. We find that our considered model and proposed inference approach capture the behavior of certain neurons in the DMFC_RSG dataset better than others. This is reflected in the computed signal-to-noise ratios, the evolving estimates shown in Figure 3.16, the PSIS diagnostic $\hat{k}$ values in Figure 3.17 and the held-out prediction results.

Neuron 13 stands out as the neuron whose observed spiking behavior in the $T = 50$k training set appears *best* captured by the assumed model and our variational approach. We note that its observed spikes (highlighted in magenta in Figure 3.14) exhibit relatively less stuttering or bursting behavior, and a more consistent firing rate across the training period (compared to the other nodes). In applying our methods to the DMFC_RSG dataset, we find that neuron 13 has the highest computed signal-to-noise ratio, its estimated model and variational parameters appear more reasonable and not heavily influenced by the constraints $2\tau_i \geq \sigma_i$ and $\eta_i > 0$, its PSIS $\hat{k}$ values are lower, and we see relatively better performance for neuron 13 in our held-out predictions (Figure 3.18).

We consider the question of whether the estimated network connections involving this 'best fitting' neuron 13 reflect meaningful underlying structure in the DMFC_RSG dataset, or if its highest computed SNR and its four 'significant' inferred in-edges are artifacts of our applied methods. We investigate this question by shuffling neuron 13's observed spikes and re-applying our methods to the perturbed data.

In particular, we generate a random mapping $\pi^* : \{1, \dots, m_{13}\} \mapsto \{1, \dots, m_{13}\}$ and construct

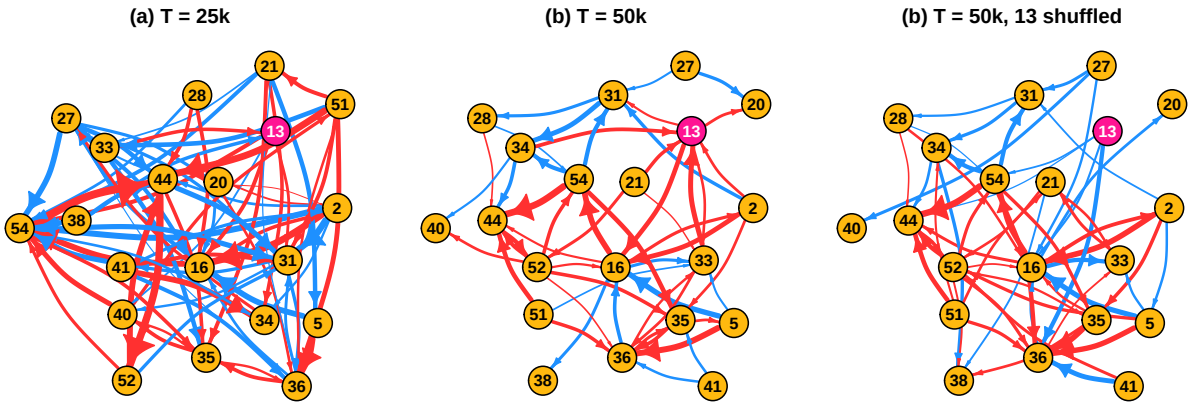

Figure 3.20: Subplots (a) and (b) show the inferred networks based on our estimates $\hat{\mathbf{W}}$ and $\hat{\mathbf{S}}$ from $T = 25$k and $T = 50$k observations, repsectively. Subplot (c) shows the inferred network estimates $\hat{\mathbf{W}}^*$ and $\hat{\mathbf{S}}^*$ following our shuffling of the order of the observed spikes for the highlighted neuron 13.

the following $T = 50$k training set $\mathbf{Y}^*$ based on the observed data $\mathbf{Y}$ and the random permutation π^*:

$$\mathbf{Y}^*_{ti} = \begin{cases} \mathbf{Y}_{ti} & \text{if } i \neq 13 \\ 1 & \text{if } i = 13 \text{ and there exists } \ell \in \{1, \dots, m_{13}\} \text{ such that } t = \sum_{k=1}^{\ell} \Delta t_{13}^{\pi^*(k)} \\ 0 & \text{otherwise} \end{cases}$$

With this perturbed data we re-compute our variational approximation $q_{\hat{\phi}^*|\mathbf{Y}^*}$, outputting the inferred network estimates $\hat{\mathbf{W}}^*$ and $\hat{\mathbf{S}}^*$, along with the re-computed signal-to-noise ratios $\text{SNR}_i^{\hat{\mathbf{W}}^*,\mathbf{Y}^*}$. Figure 3.20 shows the inferred networks based on the original and shuffled data. Comparing subplots (b) and (c), we see significant changes in the inferred connections, particularly for neuron 13. After shuffling the order of its spikes, neuron 13 loses its four 'significant' inferred in-edges, gaining none. Its four estimated positive out-edges also disappear, while four spurious negative out-edges are inferred based on the perturbed data.

Figure 3.21 shows the estimated in-edge and out-edges for neuron 13 based on the original spike train data $\mathbf{Y}$ and the perturbed data $\mathbf{Y}^*$. Figure 3.21 also shows the criterion intervals (Algorithm 3.1) for inferring whether an edge exists for a each pair of nodes involving neuron 13. These are used to construct $\hat{\mathbf{S}}$.

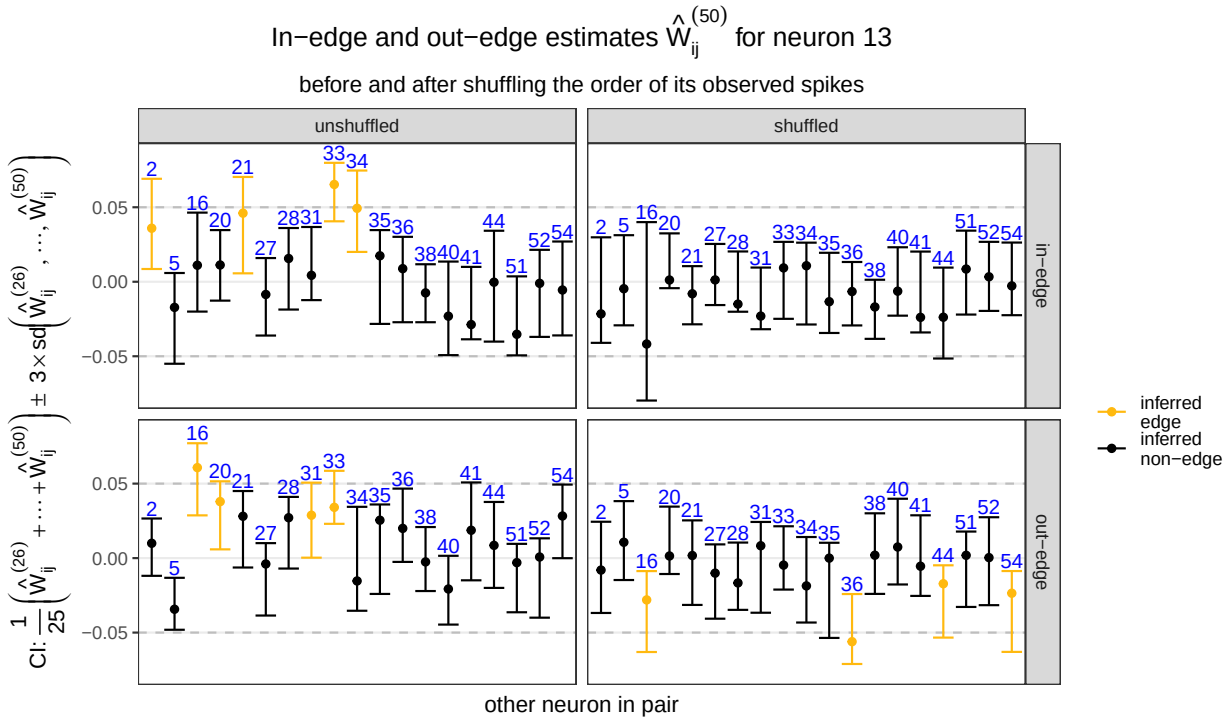

Figure 3.21: In-edge and out-edge estimates for neuron 13 before and after shuffling its spikes. The points are the final estimates after 50 epochs. The 'confidence intervals' are based on estimates after epochs 26-50. This plot shows our criterion (3.1) for inferring whether an edge exists for a given pair of nodes.

There are other changes in the inferred network not involving neuron 13. The disappearance of the negative edge between neurons 27 and 20 could have been caused by the shuffling of node 13's edges. On the other hand the appearances of the negative edge between neurons 2 and 5 and the positive edge between neurons 36 and 38 appear unrelated and spurious.

Much of the structure and connections in the network not involving neuron 13 remains the same. For example, the edges into neuron 44 do not appear significantly changed. Note that neuron 44 has the second highest computed SNR based on $\mathbf{Y}$ and $\hat{\mathbf{W}}$ and is one of the best fitting neurons besides neuron 13. This remains the case based on the perturbed data.

Figure 3.22 shows the computed signal-to-noise ratios $\text{SNR}_i^{\hat{\mathbf{W}},\mathbf{Y}}$ and $\text{SNR}_i^{\hat{\mathbf{W}}^*,\mathbf{Y}^*}$ using the original and shuffled data, respectively. The highlighted row corresponds to the neuron 13. Shuffling its spikes causes its SNR to plummet—$\text{SNR}_{13}^{\hat{\mathbf{W}}^*,\mathbf{Y}^*}$ is by far the lowest computed SNR value. For the

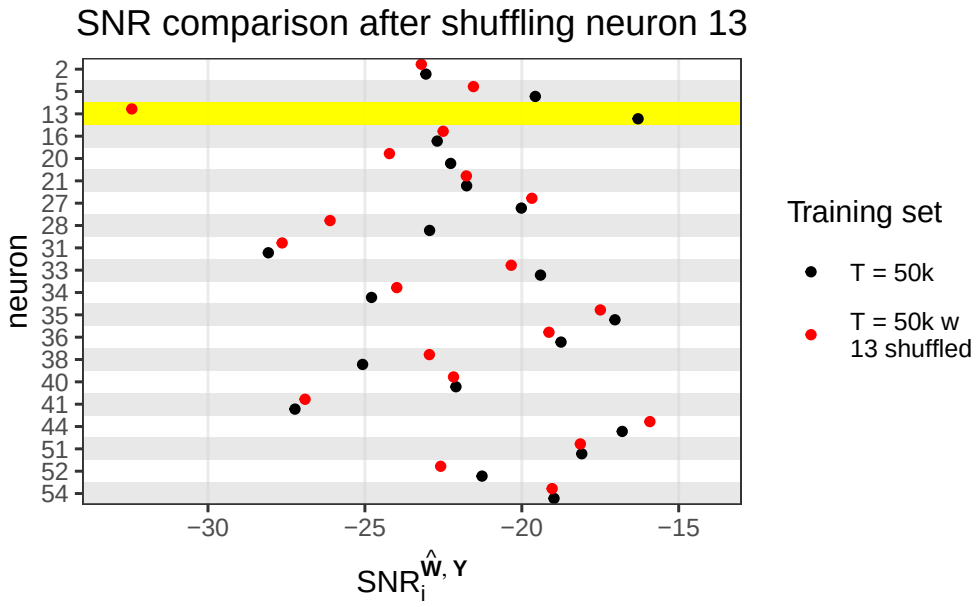

Figure 3.22: The computed SNR values before and after shuffling the order of the observed spikes for the highlighted neuron 13.

most part, the other SNR values do not change drastically based on the perturbed data.

3.6 Discussion

In this section we discuss our model and inference approach in the context of the literature on network inference, multivariate point processes and neuronal dynamics.

Network Reconstruction Related to our work and the implicit network inference problem we consider is the field of network reconstruction, which comprises methods for inferring unknown structures of a network given partial knowledge of its properties [77]. Of particular interest to us are frameworks for inferring edges from noisy, incomplete or unreliable data. In the doubly stochastic approach of [78], inferences are based on a network model specifying how edges are generated along with a data model specifying how the network structure maps onto the observations.

General notions of network reconstruction may include the problem we consider; in practice, however, network reconstruction methods involve explicitly relational data observations [79]. We note that network reconstruction naturally arises in interaction networks with relational point process

data [80], a particularly relevant context to our work. An important feature distinguishing our approach from network reconstruction methods is the scientific data generating process we consider. Incorporating a prior distribution for $\mathbf{W}$ in our approach is a topic for future work, and for this prior we are interested in considering generative network models as in [78, 79, 80, 81].

Unobserved Networks and Multivaratiate Times Series We briefly mention related work on network inference for more general multivariate time series data. Causal network reconstruction from time series [82, 83, 67] has the goal of distinguishing direct from indirect dependencies and common drivers among multiple individual time series. To accommodate dependencies at multiple lags, an infinite time series graph is introduced, whose edges are inferred using model-free tests for conditional independence, including Granger causality and tranfer entropy, along with more complicated algorithms [84].

Other works (not concerned with causality) propose mapping multivariate time series data to a multilayer network in order to more effectively extract information from a high dimensional dynamical system [85]. In order to leverage the tools of graph neural networks (GNNs) in situations where there is unknown network structure among multiple time series, [86] proposes simultaneously learning the structure and the fitting GNN, using a generative graph model parameterized by a neural network.

While the problem we consider applies generally to situations where latent network connections are inferred from point process data 'on' a set of nodes [81], we are motivated in particular by the context of neuroscience. Connectivity inference from neural activity recording data is a major research area in neuroscience [42], and its methods can be broadly classified as descriptive, model-free approaches and model-based approaches that assume a data generating process.

Descriptive methods include pairwise correlation and information-theoretic measures that may be calculated based on the observed activity of each pair of nodes. Recall that we compare our method to transfer entropy (TE) in our simulation study (section 3.4).

The approaches to unknown network structure we mention for general multivariate time series

along with these model-free methods for connectivity inference are not equipped to incorporate application-specific scientific knowledge or models. This differentiates them from our approach and makes them poor candidates for the problem we consider.

Multivariate Point Processes The Poisson process is the canonical point process, and with its independent increments, a Poisson process is not able capture interactions between events. Generalizing the Poisson process, the Hawkes process [87, 88] is an important model for event streams in which a past events can affect the probability of future events. A mutually-exciting multivariate Hawkes process can capture interactions between events, assuming that the effect is positive, additive over the past events, and exponentially decaying with time. Such models have been applied to event data on networks when edges are observed [89] as well as when they are being inferred [81].

As in the Hawkes Process, a spiking node in the model we consider directly influences the behavior of its out-neighbors. Moreover this effect decays (at the rate of δ) following its initial arrival. There are, however, key differences. The influences in our adapted LIF model may be inhibitory in addition to excitatory. And crucially, once a downstream neighbor spikes, any inputs previously received are forgotten as its latent voltage resets.

The standard Hawkes process assumptions on its allowable interactions limit the model's applicability in general [90] and to neural data in particular. Variations of the Hawkes process allow for applications beyond the mutually exciting and linear case. Introducing Markov modulation of the baseline intensity [91] accommodates heterogeneity in interevent waiting times. And the neurally self-modulated multivariate point process proposed in [92] generalizes the Hawkes Process in order to allow past events to influence the future in more complex and realistic ways. In this approach the event intensities are modeled by the hidden state of a recurrent neural network.

Model-based Approaches to Neural Population Dynamics The basic paradigm of a model-based approach for neural activity is to first assume a generative process for the observed data, and then to estimate the model parameters based on recorded neural activity. Assumed models can

range from more realistic, mechanistic models based on biological principals of neurons' behavior to more mathematically abstract, 'phenomenological' models describing the observable behavior (phenomena) of the recorded neurons in a way that is not derived from scientific theory.

The Hodgkin–Huxley model [93] is the paradigmatic mechanistic neuron model. It is a detailed conductance-based model consisting of a set of nonlinear differential equations describing the relationship between the flow of ionic currents across the neuronal cell membrane and the membrane voltage of the cell.

Hodgkin–Huxley-type models have been shown to accurately capture complex neuronal dynamics, but their intrinsic complexity makes them difficult to analyze and computationally expensive to implement [53]. And given that much simpler integrate-and-fire-type models have been shown to be more useful for exploring recorded neural populations, there are questions of the general viability of these types of models in applied settings [94].

At the opposite end of the spectrum are unsatisfactorily simple autoregressive generalized linear models [42] and inhomogeneous Poisson process models for generating spikes. These approaches are computationally tractable but fail to capture neuronal dynamics. Methods incorporating leaky integrate-and-fire (LIF) mechanisms provide an appealing compromise between oversimplified, tractable models and the more realistic but intractable mechanistic models for spike generation. In [95] the authors show the computational tractability of such a model while also demonstrating that the spike history dependence introduced by LIF mechanisms allow their model to emulate many of the spiking behaviors observed in recorded neural activity. Related work in [48] demonstrates efficient computation of the maximum a posteriori path and parameter estimation for SF-LIF and more general state-space models for the observed spiking activity of single neurons.

The leaky integrate-and-fire model we incorporate in the generative process (3.9) describes how individual neurons integrate input (3.2), spike (3.5), reset (3.4) and how their action potentials transmit to other nodes (3.7). This offers a complete (if unrealistically simple) description of the dynamical system. We discuss limitations and extensions of the LIF model at the end of section 3.2.1.

In the literature there are not many existing methods for or examples of using integrate-and-fire models to infer network connections. In [96] the authors infer 'directed information' connections for a limited network motif using a model that incorporates LIF mechanisms. We did not find any applications of generalized leaky integrate-and-fire models to connectivity inference in our literature review.

A more abstract class of methods for analyzing recorded neural population activity considers a common latent process for all the recorded neurons (as opposed to modeling individual membrane voltages or latent states). Some very popular and widely adopted methods take this approach by modeling recorded neural population activity as noisy emissions from an underlying neural trajectory, a low-dimensional latent process that is shared across the spiking cells [97, 98, 99].

Extracting smoothly varying trajectories that quantify the activity (or response) of recorded neurons within a single experimental trial is a key part of the analysis of many neurophysiological studies [100, 101]. Extracted trajectories may be used to understand the relationships between stimuli or observed behavior and neural population activity, or to test specific scientific hypotheses.

Earlier works averaged noisy spiking activity of recorded neurons across multiple trials, smoothing over trial-to-trial variability and providing more continuous firing rates for subsequent analysis while discarding key single-trial dynamics. Over time researchers in the field have discarded this approach, finding it imperative to analyze neural data on a trial-by-trial basis.

In [97] the authors propose Gaussian-process factor analysis (GPFA) for simultaneously smoothing noisy recordings and performing dimensionality reduction within a single probabilistic framework, modeling square-rooted observed counts as approximately normal. Variational latent Gaussian processes (vLGP) [99] extends GPFA by incorporating an autoregressive filter and a discrete observation model for the observed point process data. Under this model a neuron's activity depends on its observed self-history along along with the common latent process. Latent factor analysis via dynamical systems (LFADS) [98] is a machine-learning method based on recurrent neural networks (RNNs) that models recorded neural population activity as a latent dynamical system with Poisson-distributed spiking variability.

In these three approaches, connections and associations within the observed populations are implicitly encoded in the low-dimensional latent neural states and its mapping to individual neurons' behavior. Based on the models proposed for GPFA and vLGP, it seems these connections could be decoded. For LFADS, the model is much less interpretable. Connectivity inference, however, is not a goal of these works and is not mentioned.

3.7 Future research

3.7.1 A more general model for the latent dynamics

In this section we introduce and briefly discuss a generalization of (3.9) that accommodates more complex dynamics in the latent process generating the observed events. This model allows for correlations among the $z_{t,i}$'s for a fixed t and allows the deterministic input to for each node $\eta_{t,i}$ to vary with time. Generalizing our inference approach to this expanded model is a subject of future work.

The assumptions in (3.9) of independence for the latent fluctuations $\mathbf{Z}$ and constant deterministic input η_i for each node i are strong and may not hold in many situations. Groups of associated spiking neurons may be impacted in dependent ways by the unobserved input captured in $\mathbf{Z}$, and the expected rate η_i at which that input arrives may change over time.

Making these assumptions greatly facilitates our inference while still allowing for interesting applications and results. However, they are quite limiting, and worth working to eliminate or relax.

We present the following expanded model for latent dynamics that involve the unobserved network $\mathbf{W}$. In this formulation we allow that the distribution $p(\mathrm{y}_{t,i} \mid \mathrm{v}_{t,i})$ may be the same as in

(3.9) or take on some other form:

$$
\begin{aligned}
&\mathrm{y}_{t,i} \sim p(\mathrm{y}_{t,i} \mid \mathrm{v}_{t,i}) && && \text{for each } i,\, t \\
&\mathrm{v}_{t,i} = \begin{cases} \delta \mathrm{v}_{t-1,i} + \eta_{t,i} + \mathrm{z}_{t,i} + \mathbf{w}_{\to i}{}^{\top} \mathbf{y}_{t-1} & \text{if } \mathrm{y}_{t-1,i} = 0 \\ \eta_{t,i} + \mathrm{z}_{t,\,i} & \text{if } \mathrm{y}_{t-1,i} = 1 \end{cases} && && \text{for each } i,\, t \\
&\mathbf{z}_t \sim \mathrm{N}(0,\, \Sigma) && && \text{for each } i,\, t.
\end{aligned}
$$

In this model the random vector $\mathbf{z}_t = (\mathrm{z}_{t,1}, \ldots, \mathrm{z}_{t,n})$ has covariance given by Σ, and the $\eta_{t,i}$'s are not constant for each node over time, but instead follow trajectories that need to be estimated. This full model reduces to (3.9) when $\Sigma = \mathrm{diag}(\sigma_1^2, \ldots, \sigma_n^2)$ and $\eta_{t,i} = \eta_i$ for all t. Looking at the joint distribution of this model,

$$
p(\mathbf{Y}, \mathbf{Z}) = \prod_{t=1}^{T} p(\mathbf{y}_t, \mathbf{z}_t | \mathcal{F}_{t-1}) = \prod_{t=1}^{T} \left\{ p(\mathbf{z}_t) \prod_{i=1}^{n} p(\mathrm{y}_{t,i} | \mathrm{v}_{t,i}) \right\}.
$$

we can see that it does not factor by node and inter-spike period as in (3.12). Thus a variational approximation q_ϕ of the form (3.18) will not work.

We are interested in adopting methods from the Variational Gaussian Process [102] to construct and compute a variational approximation for this model. Roughly speaking, we plan to consider a q_ϕ of the form:

$$
q_\phi(\mathbf{Z}) = \int \left\{ \prod_{(i,k) \in \mathcal{X}} q_{\phi_i | \mathbf{Y}} \left(\mathbf{z}_{(i,k)} \middle| \lambda_{(i,k)} \right) \right\} q_{\phi | \mathbf{Y}}(\lambda)\, \mathrm{d}\lambda
$$

where $\lambda_{(i,k)} = (\lambda_{t_k^i + 1, i}, \ldots, \lambda_{t_{k+1}^i, i})$, and

$$
q_{\phi_i | \mathbf{Y}} \left(\mathbf{z}_{(i,k)} | \lambda_{(i,k)} \right) \sim \mathrm{N}\Big(\mu_{\phi_i, \lambda_{(i,k)}}(i,k),\, \Sigma_{\phi_i}(\Delta t_k^i) \Big).
$$

Here the approximate posterior distribution for the latent variables $\mathbf{Z}$ is a hierarchical model. The

latent parameters $\lambda \in \mathbb{R}^{T \times n}$ have the distribution:

$$q_{\phi|\mathbf{Y}}(\lambda) = \prod_{t=1}^{T} q_{\phi|\mathbf{Y}}(\lambda_{t,1}, \ldots, \lambda_{t,n}) \sim \prod_{t=1}^{T} \mathrm{N}(0, \Gamma).$$

This is a subject of future work.

3.8 Conclusion

In this work we have considered the task of inferring the connections between noisy observations of events. In our model-based approach, we assumed a generative process incorporating latent dynamics that are directed by the unobserved network structure, along with a mechanistic model from neuroscience for aggregating input and triggering observations.

We developed a novel variational Bayesian approach for estimating the model parameters as well as 'downstream' methods for further analysis of the the model and data (see Figure 3.1).

From our applications to simulated and real data we see the success and potential of our method, as well as its limitations.

Conclusion

In its three chapters this dissertation examines, develops and applies explainable models and methods for challenging structured data. Through applications we have seen how interpetable, 'white box' methods are able to identify where the data is most complicated and difficult to explain.

Faced with the task of developing explainable methods for difficult problems, the applied statistician may lament an apparent paradox—while 'black box' approaches that achieve reductions in prediction error falsely present an aura of mastery, even explainability to the data they consider, interpretable methods shine a bright light where they fail and can leave us with mixed feelings about the difficulty of the considered problem.

To me this part of doing research in applied statistics. Properly accounting for structure in the data we analyze provides the indispensable context for the complexity of the considered problem.

References

[1] A. Menendez *et al.*, "Strong dynamics in tidal marsh doc export in response to natural cycles and episodic events from continuous monitoring," *Journal of Geophysical Research: Biogeosciences*, vol. 127, no. 7, 2022.

[2] W. R. Palmer and T. Zheng, "Spectral clustering for directed networks," in *Complex Networks & Their Applications IX: Volume 1, Proceedings of the Ninth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2020*, Springer, 2021, pp. 87–99.

[3] W. R. Palmer, R. A. Davis, and T. Zheng, "Count-valued time series models for covid-19 daily death dynamics," *Stat*, vol. 10, no. 1, e369, 2021.

[4] M. Girvan and M. E. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci.*, vol. 99, no. 12, pp. 7823–7826, 2002.

[5] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Statistical properties of community structure in large social and information networks," *Proc. of WWW*, pp. 695–704, 2008.

[6] F. Malliaros and M. Vazirgiannis, "Clustering and community detection in directed networks," *Phys. Rep.*, vol. 533, no. 4, pp. 95–142, 2013.

[7] K. Rohe, S. Chatterjee, and B. Yu, "Co-clustering directed graphs to discover asymmetries and directional communities," *Proceedings of the National Academy of Sciences of the USA*, vol. 113, no. 45, pp. 12 679–12 684, 2016.

[8] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[9] K. Rohe, S. Chatterjee, and B. Yu, "Spectral clustering and the high-dimensional stochastic blockmodel," *Ann. Stat.*, vol. 39, no. 4, pp. 1878–1915, 2011.

[10] P. J. Bickel and A. Chen, "A nonparametric view of network models and newman–girvan and other modularities," *Proc. Nat. Acad. Sci.*, vol. 106, no. 50, pp. 21 068–21 073, 2009.

[11] U. von Luxburg, "A tutorial on spectral clustering," *Stat. Comput.*, vol. 17, pp. 395–416, 2007.

[12] D. Zhou, J. Huang, and B. Schoelkopf, "Learning from labeled and unlabeled data on a directed graph," *Proceedings of the 22nd International Conference on Machine Learning*, 2005.

[13] F. Chung, "Laplacians and the cheeger inequality for directed graphs," *Ann. Comb.*, vol. 9, no. 1, pp. 1–19, 2005.

[14] M. I. J. Andrew Y. Ng and Y. Weiss, "On spectral clustering: Analysis and an algorithm," *Neural Inf. Process. Syst.*, vol. 14, pp. 849–856, 2001.

[15] T. Xiang and S. Gong, "Spectral clustering with eigenvector selection," *Pattern Recognit.*, vol. 41, no. 3, pp. 1012–1029, 2008.

[16] J. Jin, "Fast community detection by score," *Ann. Stat.*, vol. 43, no. 1, pp. 57–89, 2015.

[17] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences of the USA*, vol. 103, no. 23, pp. 8577–8582, 2006.

[18] P. W.Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social Networks*, vol. 5, no. 2, pp. 109–137, 1983.

[19] N. X. Vinh, J. R. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *J. Mach. Learn. Res*, vol. 11, pp. 2837–2854, 2010.

[20] J. H. Fowler, "Connecting the congress: A study of cosponsorship networks," *Political Analysis*, vol. 14, no. 4, pp. 456–487, 2006.

[21] Y. Zhang, A. Friend, A. L. Traud, M. A. Porter, J. H. Fowler, and P. J. Mucha, "Community structure in congressional cosponsorship networks," *Physica*, vol. 387, no. 7, pp. 1705–1712, 2008.

[22] M. J. Oleszek, *Sponsorship and cosponsorship of house bills*, Congressional Research Service Report RS22477, 2019.

[23] ProPublica, *Congress api*, 2020.

[24] Congress.gov, *Legislative search results*.

[25] Center for Systems Science and Engineering (CSSE) at Johns Hopkins University, *COVID-19 Dashboard*, Available from: `https://coronavirus.jhu.edu/map.html` [last accessed January 2021], 2020.

[26] B. Carey and J. Glanz, *Hidden outbreaks spread through U.S. cities far earlier than americans knew, estimates say*, 2020.

[27] G. Chowell, L. Sattenspiel, S. Bansal, and C. Viboud, "Mathematical models to characterize early epidemic growth: A review," *Physics of Life Reviews*, vol. 28, pp. 66–97, 2016.

[28] I. Cooper, A. Mondal, and C. G. Antonopoulos, "A SIR model assumption for the spread of COVID-19 in different communities," *Chaos, Solitons & Fractals*, vol. 139, p. 110 057, 2020.

[29] S. He, Y. Peng, and K. Sun, "SEIR modeling of the COVID-19 and its dynamics," *Nonlinear Dynamics*, vol. 101, no. 3, pp. 1667–1680, 2020.

[30] N. Altieri *et al.*, "Curating a COVID-19 data repository and forecasting county-level death counts in the United States," *Harvard Data Science Review*, 2020.

[31] A. Harvey and P. Kattuman, "Time Series Models Based on Growth Curves with Applications to Forecasting Coronavirus," *Harvard Data Science Review*, 2020.

[32] N. Jewell, J. Lewnard, and B. Jewell, "Predictive Mathematical Models of the COVID-19 Pandemic: Underlying Principles and Value of Projections.," *JAMA*, vol. 323, no. 19, pp. 1893–1894, 2020.

[33] V. Chin *et al.*, "A case study in model failure? COVID-19 daily deaths and ICU bed utilisation predictions in New York state," *European Journal of Epidemiology*, vol. 35, no. 8, pp. 733–742, 2020.

[34] New York City Department of Health and Mental Hygiene, *NYC Coronavirus Disease 2019 (COVID-19) Data*, Available from: `https://github.com/nychealth/coronavirus-data` [last accessed January 1, 2021], 2020.

[35] S. Pappas, *How COVID-19 deaths are counted*, 2020.

[36] B. Carpenter *et al.*, "Stan: A Probabilistic Programming Language," *Journal of Statistical Software, Articles*, vol. 76, no. 1, pp. 1–32, 2017.

[37] Stan Development Team, *Stan User's Guide*, 2020.

[38] R. A. Davis, Y. Wang, and W. T. M. Dunsmuir, "Modelling time series of count data," in *Asymptotics, Nonparametric & Time Series*, New York City, NY, USA: CRC Press, 1999, pp. 63–114.

[39] C. Czado, T. Gneiting, and L. Held, "Predictive Model Assessment for Count Data," *Biometrics*, vol. 65, no. 4, pp. 1254–1261, 2009.

[40] Texas Department of State Health Services, *Texas COVID-19 Data: Fatalities over Time by County*, Available from: `https://dshs.texas.gov/coronavirus/additionaldata/` [last accessed January 12, 2021], 2020.

[41] M. Borkovec and C. Klüppelberg, "The Tail of the Stationary Distribution of an Autoregressive Process with Arch(1) Errors," *Annals of Applied Probability*, vol. 11, no. 4, pp. 1220–1241, 2001.

[42] I. Magrans de Abril, J. Yoshimoto, and K. Doya, "Connectivity inference from neural recording data: Challenges, mathematical bases and research directions," *Neural Networks*, vol. 102, pp. 120–137, 2018.

[43] E. Kolaczyk, *Statistical Analysis of Network Data: Methods and Models*. Springer New York, NY, 2009, ISBN: 978-0-387-88145-4.

[44] P. W. Battaglia, R. Pascanu, M. Lai, D. J. Rezende, and K. Kavukcuoglu, "Interaction networks for learning about objects, relations and physics," *CoRR*, 2016.

[45] Z. Zhang *et al.*, "A general deep learning framework for network reconstruction and dynamics learning," *Applied Network Science*, vol. 4, no. 1, p. 110, 2019.

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

[47] P. Lewis, "Multivariate point processes," in *Proceedings of the Berkeley Symposium on Mathematical Statistics and Probability*, University of California Press, vol. 1, 1972, p. 401.

[48] S. Koyama and L. Paninski, "Efficient computation of the maximum a posteriori path and parameter estimation in integrate-and-fire and more general state-space models," *J. Comput. Neurosci.*, vol. 29, pp. 89–105, 2010.

[49] C. Grienberger and A. Konnerth, "Imaging calcium in neurons," *Neuron*, vol. 73, no. 5, pp. 862–885, 2012.

[50] M. E. Spira and A. Hai, "Multi-electrode array technologies for neuroscience and cardiology," *Nature nanotechnology*, vol. 8, no. 2, pp. 83–94, 2013.

[51] M. S. Lewicki, "A review of methods for spike sorting: The detection and classification of neural action potentials," *Network: Computation in Neural Systems*, vol. 9, no. 4, R53, 1998.

[52] C. Teeter *et al.*, "Generalized leaky integrate-and-fire models classify multiple neuron types," *Nature Communications*, vol. 9, no. 1, p. 709, 2018.

[53] R. Jolivet, T. J. Lewis, and W. Gerstner, "Generalized integrate-and-fire models of neuronal activity approximate spike trains of a detailed model to a high degree of accuracy," *Journal of Neurophysiology*, vol. 92, no. 2, pp. 959–976, 2004.

[54] Y.-H. Liu and X.-J. Wang, "Spike-frequency adaptation of a generalized leaky integrate-and-fire model neuron," *Journal of Computational Neuroscience*, vol. 10, no. 1, pp. 25–45, 2001.

[55] W. Rall, "Time constants and electrotonic length of membrane cylinders and neurons," *Biophysical journal*, vol. 9, no. 12, 1483–1508, 1969.

[56] D. A. McCormick, B. W. Connors, J. W. Lighthall, and D. A. Prince, "Comparative electrophysiology of pyramidal and sparsely spiny stellate neurons of the neocortex," *Journal of Neurophysiology*, vol. 54, no. 4, pp. 782–806, 1985, PMID: 2999347.

[57] P Jonas, G Major, and B Sakmann, "Quantal components of unitary epscs at the mossy fibre synapse on ca3 pyramidal cells of rat hippocampus," *The Journal of Physiology*, vol. 472, no. 1, pp. 615–663, 1993.

[58] J. I. Luebke and D. L. Rosene, "Aging alters dendritic morphology, input resistance, and inhibitory signaling in dentate granule cells of the rhesus monkey," *Journal of Comparative Neurology*, vol. 460, no. 4, pp. 573–584, 2003.

[59] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine Learning*, vol. 37, no. 2, pp. 183–233, 1999.

[60] D. P. Kingma and M. Welling, *Auto-encoding variational Bayes*, 2013.

[61] E. Archer, I. M. Park, L. Buesing, J. Cunningham, and L. Paninski, *Black box variational inference for state space models*, 2015.

[62] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Advances in Neural Information Processing Systems*, vol. 28, Curran Associates, Inc., 2015.

[63] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2014.

[64] E. Izhikevich, "Simple model of spiking neurons," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1569–1572, 2003.

[65] R Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, 2023.

[66] G. Czanner *et al.*, "Measuring the signal-to-noise ratio of a neuron," *Proceedings of the National Academy of Sciences*, vol. 112, no. 23, pp. 7141–7146, 2015.

[67] L. Cui and J. M. Moore, "Causal network reconstruction from nonlinear time series: A comparative study," *International Journal of Modern Physics C*, vol. 32, no. 04, p. 2 150 049, 2021.

[68] B. Simon, D. Thomas, P. Franziska J., and Z. David J., "Rtransferentropy — quantifying information flow between different time series using effective transfer entropy," *SoftwareX*, vol. 10, no. 100265, pp. 1–9, 2019.

[69] Y. Yao, A. Vehtari, D. Simpson, and A. Gelman, "Yes, but did it work?: Evaluating variational inference," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, PMLR, 2018, pp. 5581–5590.

[70] A. Vehtari, D. Simpson, A. Gelman, Y. Yao, and J. Gabry, "Pareto smoothed importance sampling," *arXiv preprint arXiv:1507.02646*, 2015.

[71] P. Mahalanobis, "On the generalised distance in statistics," in *Proceedings National Institute of Science, India*, 1936, pp. 49–55.

[72] G. McLachlan, "Mahalanobis distance," *Resonance*, vol. 4, no. 6, pp. 20–26, 1999.

[73] F. Pei *et al.*, "Neural latents benchmark '21: Evaluating latent variable models of neural population activity," in *Advances in Neural Information Processing Systems (NeurIPS), Track on Datasets and Benchmarks*, 2021.

[74] M. Isoda and A. Noritake, "What makes the dorsomedial frontal cortex active during reading the mental states of others?" *Frontiers in neuroscience*, vol. 7, p. 232, 2013.

[75] J. Sallet *et al.*, "The organization of dorsal frontal cortex in humans and macaques," *Journal of Neuroscience*, vol. 33, no. 30, pp. 12 255–12 274, 2013.

[76] H. Sohn, D. Narain, N. Meirhaeghe, and M. Jazayeri, "Bayesian computation through cortical latent dynamics," *bioRxiv*, 2018.

[77] G. Cimini, R. Mastrandrea, and T. Squartini, *Reconstructing Networks*. Cambridge University Press, 2021.

[78] M. E. J. Newman, "Estimating network structure from unreliable measurements," *Phys. Rev. E*, vol. 98, 6 2018.

[79] M. S. Handcock and K. J. Gile, "Modeling social networks from sampled data," *The Annals of Applied Statistics*, vol. 4, no. 1, 2010.

[80] G. A. Zagatti, S.-K. Ng, and S. Bressan, "Latent relational point process: Network reconstruction from discrete event data," in *Database and Expert Systems Applications*, Springer International Publishing, 2022, pp. 32–46.

[81] S. W. Linderman and R. P. Adams, "Discovering latent network structure in point process data," in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, 2014, pp. 1413–1421.

[82] J. Runge, "Causal network reconstruction from time series: From theoretical assumptions to practical estimation," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, no. 7, 2018.

[83] M. Eichler, "Causal inference with multiple time series: Principles and problems," *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, vol. 371, 2013.

[84] H. Ma, K. Aihara, and L. Chen, "Detecting causality from nonlinear dynamics with short-term time series," *Scientific reports*, vol. 4, no. 1, pp. 1–10, 2014.

[85] L. Lacasa, V. Nicosia, and V. Latora, "Network structure of multivariate time series," *Scientific Reports*, vol. 5, no. 1, 2015.

[86] C. Shang, J. Chen, and J. Bi, "Discrete graph structure learning for forecasting multiple time series," *CoRR*, 2021.

[87] A. G. Hawkes, "Spectra of some self-exciting and mutually exciting point processes," *Biometrika*, vol. 58, no. 1, pp. 83–90, 1971.

[88] A. G. Hawkes and D. Oakes, "A cluster process representation of a self-exciting process," *Journal of Applied Probability*, vol. 11, no. 3, pp. 493–503, 1974.

[89] J. Etesami, N. Kiyavash, K. Zhang, and K. Singhal, *Learning network of multivariate Hawkes processes: A time series approach*, 2016.

[90] S. Chen, A. Shojaie, E. Shea-Brown, and D. Witten, *The multivariate Hawkes process in high dimensions: Beyond mutual excitation*, 2019. arXiv: 1707.04928 [stat.ME].

[91] J. Wu, O. G. Ward, J. Curley, and T. Zheng, "Markov-modulated Hawkes processes for modeling sporadic and bursty event occurrences in social interactions," *The Annals of Applied Statistics*, vol. 16, no. 2, pp. 1171 –1190, 2022.

[92] H. Mei and J. M. Eisner, "The neural Hawkes process: A neurally self-modulating multivariate point process," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.

[93] A. Hodgkin and A. Huxley, "A quantitative description of membrane current and its application to conduction and excitation in nerve," *Bulletin of mathematical biology*, vol. 52, no. 1-2, pp. 25–71, 1990.

[94] C. Meunier and I. Segev, "Playing the Devil's advocate: Is the Hodgkin–Huxley model useful?" *Trends in Neurosciences*, vol. 25, no. 11, pp. 558–563, 2002.

[95] L. Paninski, J. W. Pillow, and E. P. Simoncelli, "Maximum likelihood estimation of a stochastic integrate-and-fire neural encoding model," *Neural Computation*, vol. 16, no. 12, pp. 2533–2561, 2004.

[96] N. Soltani and A. J. Goldsmith, "Directed information between connected leaky integrate-and-fire neurons," *IEEE Transactions on Information Theory*, vol. 63, no. 9, pp. 5954–5967, 2017.

[97] B. M. Yu, J. P. Cunningham, G. Santhanam, S. Ryu, K. V. Shenoy, and M. Sahani, "Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity," in *Advances in Neural Information Processing Systems*, vol. 21, Curran Associates, Inc., 2008.

[98] C. Pandarinath *et al.*, "Inferring single-trial neural population dynamics using sequential auto-encoders," *Nature Methods*, vol. 15, no. 10, pp. 805–815, 2018.

[99] Y. Zhao and I. M. Park, "Variational latent Gaussian process for recovering single-trial dynamics from population spike trains," *Neural Computation*, vol. 29, no. 5, pp. 1293–1316, 2017.

[100] M. M. Churchland *et al.*, "Neural population dynamics during reaching," *Nature*, vol. 487, no. 7405, pp. 51–56, 2012.

[101] J. P. Cunningham and B. M. Yu, "Dimensionality reduction for large-scale neural recordings," *Nature Neuroscience*, vol. 17, no. 11, pp. 1500–1509, 2014.

[102] D. Tran, R. Ranganath, and D. M. Blei, *The variational Gaussian process*, 2016.