Bayesian Clustering of Neural Spiking Activity Using a Mixture of Dynamic Poisson Factor Analyzers

Ganchao Wei  
Department of Statistics  
University of Connecticut  
ganchao.wei@uconn.edu

Ian. H. Stevenson  
Department of Psychological Sciences  
University of Connecticut  
ian.stevenson@uconn.edu

Xiaojing Wang  
Department of Statistics  
University of Connecticut  
xiaojing.wang@uconn.edu

Abstract

Modern neural recording techniques allow neuroscientists to observe the spiking activity of many neurons simultaneously. Although previous work has illustrated how activity within and between known populations of neurons can be summarized by low-dimensional latent vectors, in many cases what determines a unique population may be unclear. Neurons differ in their anatomical location, but also, in their cell types and response properties. Moreover, multiple distinct populations may not be well described by a single low-dimensional, linear representation. To tackle these challenges, we develop a clustering method based on a mixture of dynamic Poisson factor analyzers (mixDPFA) model, with the number of clusters treated as an unknown parameter. To do the analysis of DPFA model, we propose a novel Markov chain Monte Carlo (MCMC) algorithm to efficiently sample its posterior distribution. Validating our proposed MCMC algorithm with simulations, we find that it can accurately recover the true clustering and latent states and is insensitive to the initial cluster assignments. We then apply the proposed mixDPFA model to multi-region experimental recordings, where we find that the proposed method can identify novel, reliable clusters of neurons based on their activity, and may, thus, be a useful tool for neural data analysis.

1 Introduction

With modern high-density probes [Jun et al., 2017], neuroscientists can observe the spiking activity of many neurons from many different anatomical regions simultaneously. With these expanding capabilities, new methods to analyze neural data at the population-level and at the level of multiple populations become necessary. Several recent models have been developed to extract shared latent structures from simultaneous neural recordings, assuming that neural activity can be described through low-dimensional latent states. Many existing approaches are extensions of two basic models: the linear dynamical system (LDS) model [Macke et al., 2011] and a Gaussian process factor analysis (GPFA) model [Yu et al., 2009]. The LDS model is built on the state-space model and assumes latent factors evolve with linear dynamics. On the other hand, GPFA models the latent vectors by non-parametric Gaussian processes. However, in both cases, the observation model is generalized linear. Several variants of these models have been implemented to analyze multiple neural populations and their interactions [Semedo et al., 2019; Glaser et al., 2020]. However, in many cases, the total number of distinct populations and which neurons belong to a population is unclear.
Neurons in different anatomical locations may interact with each other or receive common input from unobserved brain areas, sharing the same latent structure. On the other hand, neurons of different cell-types within the same brain area may be better described by distinct latent structures. From a functional point of view, neither the anatomical location nor cell type (Fig. 1A) indicates which neurons should be grouped into the same populations. The incorrect population assignments can lead to biased and inconsistent inference on the latent structure (Ventura, 2009). If we instead ignore multi-population structure and treat all neurons as a single population, then using linear model based methods may not describe their activity well, especially when the input is non-homogeneous. Besides, nonlinear models such as deep learning (Pandarinnath et al., 2018; Whiteway et al., 2019) and Gaussian processes (Wu et al., 2017) have been developed, but these models do not explicitly distinguish among distinct populations of neurons.

Motivated by the mixture of (Gaussian) factor analyzers (MFA, Arminger et al. 1999; Ghahramani and Hinton 1996; Fokoué and Titterington 2003), which describes globally nonlinear data by combining a number of local factor analyzers, here we group neurons based on the latent factors (Fig. 1B). A similar idea was previously implemented using a mixture of Poisson linear dynamical system (PLDS) model (mixPLDS, Buesing et al., 2014). The mixPLDS model infers the subpopulations and latent factors using deterministic variational inference (Wainwright and Jordan, 2008; Jordan et al., 1999; Emityaz Khan et al., 2013) and the model parameters are estimated by Expectation Maximization (EM). Unlike MFA, the mixPLDS can capture temporal dependencies of neural activity as well as interactions between clusters over time. However, there are several limitations for mixPLDS: 1) it requires we predetermine the number of clusters, and 2) the clustering results are often sensitive to the initial cluster assignment.

Here we cluster the neurons by a mixture of dynamic Poisson factor analyzers (mixDPFA). The DPFA model takes the advantages of both Poisson factor analysis (FA) and PLDS and includes both a population baseline and baselines for individual neurons. The number of clusters is treated as an unknown parameter in the mixDPFA, and the posteriors are sampled using Markov Chain Monte Carlo (MCMC). To sample high dimensional latent factors, we approximate the full conditional distribution of the latent state by a Gaussian, which is similar to results by sampling from exact full conditional distribution. To improve mixing in the cluster assignments, we marginalize the loading out for clustering by Poisson-Gamma conjugacy. We also discuss the constraints necessity for successful sampling of the proposed models. After validating the proposed model with simulated data, we apply it to analyze multi-region experimental recordings from behaving mice: the Visual Coding - Neuropixels Dataset from the Allen Institute for Brain Science. Overall, the proposed method provides a way to efficiently cluster neurons into populations based on their activity.

2 Methods

Here we introduce a mixture of dynamic Poisson factor analyzers (mixDPFA) to cluster neurons based on multi-population latent structure. The number of mixture components is treated as an unknown parameter and the posteriors are sampled by MCMC. In this section, we first provide the single population DPFA for a given cluster. Then, we introduce a prior on the number of clusters and describe how we use the mixture of finite mixture model (MFM) to efficiently sample the posterior of the mixDPFA.

2.1 Dynamic Poisson Factor Analyzer

Denote the observed spike count of neuron \( i \in \{1, \ldots, N\} \) at time bin \( t \in \{1, \ldots, T\} \) as \( y_{it} \) (a non-negative integer), and let \( y_i = (y_{i1}, \ldots, y_{iT})' \). Further, let \( z_i \) be the cluster indicator of neuron \( i \). Motivated by the nature of neural activity and the former PLDS model (Macke et al., 2011), we propose a new Poisson FA model by adding individual baselines \( \delta_i \). The proposed model is a combination of PLDS and Poisson FA, which includes both population baseline and individual baseline. Assume neuron \( i \) belongs to the \( j \)-th cluster (i.e., \( z_i = j \)), and its spiking activity is independently Poisson distributed, conditional on the low-dimensional latent state \( x_t^{(j)} \in \mathbb{R}^d \) and population baseline \( \mu_t^{(j)} \) as follows:

\[
y_{it} \sim \text{Poi}(\lambda_{it}), \\
\log \lambda_{it} = \delta_i + \mu_t^{(j)} + c_i'x_t^{(j)},
\]
There are multiple potential ways to define neural populations. For instance, populations could be defined by anatomical regions (left) or by cell types (right). Since the same latent structure could be shared across anatomical sites and cell types, a useful alternative may be define populations based on neural activity directly.

The main goal for the proposed method is to cluster neurons according to their activity and extract functional grouping structure, based on spike train observations. The activity of each neuron is determined by a low dimensional latent state, specific to that neuron’s cluster assignment (e.g. yellow, red, blue). Graphical representation of the mixture of finite mixtures (MFM) of dynamic Poisson factor analyzers (DPFA) generative model. Here the cluster number is treated as a random variable. The population baseline \(\mu_j(t)\) and the latent factor \(x_j(t)\) for each cluster is generated by linear dynamics, with a Gaussian noise.

\[
\begin{align*}
\mu_{t+1}^{(j)} &= g^{(j)} + h^{(j)} \mu_t^{(j)} + \epsilon_t^{(j)} , \\
x_{t+1}^{(j)} &= b^{(j)} + A^{(j)} x_t^{(j)} + \eta_t^{(j)},
\end{align*}
\]

where \(\epsilon_t^{(j)} \sim \mathcal{N}(0, \sigma^{2(j)})\) and \(\eta_t^{(j)} \sim \mathcal{N}_p(0, Q^{(j)})\).

If we denote \(\lambda_i = (\lambda_{i1}, \ldots, \lambda_{iT})'\), \(\mu^{(j)} = (\mu_{1}^{(j)}, \ldots, \mu_{T}^{(j)})'\) and \(X^{(j)} = (x_1^{(j)}, \ldots, x_T^{(j)})'\), the proposed model can be rewritten as

\[
y_i \sim \text{Poi}(\lambda_i), \\
\log \lambda_i = \delta_i 1_T + \mu^{(j)} + X^{(j)} c_i.
\]

Generally, a factor model is consistent only when \(T/N \to 0\) \cite{Johnstone2009}, but this is often not the case for most neural spike data. However, when we assume linear dynamics on \(\mu^{(j)}\) and \(X^{(j)}\), it resolves the consistency issue. As known in a FA model, when \(p > 1\), the model is only identifiable up to orthogonal rotation on \(X^{(j)}\), with \(c_i \sim \mathcal{N}(0, I_p)\). With including an individual baseline \(\delta_i 1_T\) in our proposed DPFA model (1), it further makes the model invariant to translation of \(\mu^{(j)}\) and \(X^{(j)}\). That means if \(\mu^{(j)}\) and \(X^{(j)}\) is a set solution, then \(\mu^{(j)} + a 1_T\) and \(X^{(j)} U + 1_T \otimes m'\) also satisfy the model, for any \(a\), \(m\) and orthogonal matrix \(U\). Thus, to make the model identifiable, we need to add a few constraints. Although the clustering is invariant to orthogonal rotation, how we put constraints on translation will influence the cluster assignments. Here we assume \(A^{(j)}\) and...
We find that the proposed method using a Laplace approximation is faster but performs similarly as when the population labels are known (Appendix A.1). In each iteration, we sample the model parameters assuming the known cluster indices. Approximating the intractable full conditional with a Laplace approximation of the intractable full conditional distribution of latent effects has also been used in Bayesian mixed effects binomial regression, where Berman et al. 2022 found that the approximation provided reasonable estimation accuracy with substantial computational speedups. We, thus, use a Laplace approximation for EM (Macke et al., 2011) to sample exactly from the full conditional of labeled model parameters, the latent state $X$. Once we update the latent state $X$ by the analogy of partition-based algorithm in Dirichlet process mixtures (DPM, Neal 2000), see more details of prior settings in Appendix A.1.

2.2 Clustering by Mixture of Finite Mixtures Model

When the population labels $z_i$s are unknown, we cluster the neurons by a mixture of DPFA (mixDPFA). Since the number of neural populations is finite but unknown, we need to put priors on it. To make the Bayesian computation more efficient, we utilize the idea from the mixture of finite mixtures (MFM, Miller and Harrison 2018) model, by assigning the priors for the clusters in the following way:

$$k \sim f_k,$$

$$\pi = (\pi_1, \ldots, \pi_k) \sim \text{Dir}(\gamma, \ldots, \gamma) \quad \text{given } k,$$

$$z_1, \ldots, z_N \sim \pi \quad \text{given } \pi,$$

$$\theta^{(1)}, \ldots, \theta^{(k)} \sim H \quad \text{given } k,$$

$$y_i = (y_{i1}, \ldots, y_{iT}) \sim \text{DPFA}(\delta_i, c_i, \theta^{(z_i)}) \quad \text{given } \delta_i, c_i, \theta^{(z_i)}, z_i, \forall i = 1, \ldots, N,$$

where $f_k$ is a p.m.f. on $\{1, 2, \ldots\}$, $\pi$ is a p.m.f. on $\{1, 2, \ldots, k\}$, $z_i$ is the cluster index, and $(\theta^{(1)}, \ldots, \theta^{(k)})$ are generated by the dynamic Poisson factor analyzer (DPFA) model, see more details of prior settings in Appendix A.1. To facilitate the Bayesian computation, we have to impose priors on $H$ on $\theta^{(j)}$, see more details of prior settings in Appendix A.1.

2.3 Inference

Here the posteriors of the proposed mixDPFA model are sampled by an MCMC algorithm (see Appendix A.1.1). In each iteration, we sample the model parameters assuming the known cluster indices at first, and then sample the cluster indices given the model parameters. When sampling the cluster parameters, the latent state $X^{(j)}$ and population baseline $\mu^{(j)}$ have no closed-form full conditional distributions. Although we could sample the posterior by particle MCMC directly, we could add a Metropolis-Hastings (MH) step to sample exactly from the full conditional of $X^{(j)}$ and $\mu^{(j)}$. We find that the proposed method using a Laplace approximation is faster but performs similarly as sampling from the exact joint posterior (Fig. 2, Fig. 4, Appendix A.2 and Table 1).

Once we update the latent state $X^{(j)}$ and population baseline $\mu^{(j)}$, the cluster index is then sampled by the analogy of partition-based algorithm in Dirichlet process mixtures (DPM, Neal 2000). See details in Miller and Harrison 2018 and the Appendix A.1. When doing the clustering, we need to evaluate the likelihood for neurons under each cluster. Although we can sample $c_i$ directly and evaluate the full likelihood as in MCMC for Gaussian MFA (data-augmentation/ imputation-posterior
We then compare the 10-cluster model with the simplified model ignoring the clustering structure. We first simulate 10 clusters with 5 neurons in each, with recording length \( T \) with 5 neurons in each. Running MCMC for 10,000 iterations, we find that the log-likelihood per all experiments in this paper were performed using a 3.40 GHz processor with 16 GB of RAM.

To validate and illustrate the proposed clustering method, we simulate neural data directly from the generative model (1). The labels for each neuron are assumed known and fixed at first to check convergence and model identifiability. We then infer the labels to evaluate clustering performance.

3 Simulations

To validate and illustrate the proposed clustering method, we simulate neural data directly from the generative model (1). The labels for each neuron are assumed known and fixed at first to check convergence and model identifiability. We then infer the labels to evaluate clustering performance. All experiments in this paper were performed using a 3.40 GHz processor with 16 GB of RAM.

3.1 Labeled data

We first simulate 10 clusters with 5 neurons in each, with recording length \( T = 1000 \) and \( p = 2 \) dimensional latent factors for each cluster. Individual baselines are generated by \( \delta_i \sim N(0, 0.5^2) \), and the loading for the latent states are generated by \( c_i \sim N(0, I_2) \). The population baseline \( \mu^{(j)} \) and latent vector \( X^{(j)} \) are generated by the spline interpolation on 10 to 30 evenly spaced knots. The simulations are conducted 50 times with different seeds. Here we show results for one simulation, and the performance for the rest is similar.

Since the labels are known, each whole simulation is equivalent to 10 independent simulations, with 5 neurons in each. Running MCMC for 10,000 iterations, we find that the log-likelihood per spike converges rapidly for individual clusters and overall (Fig. [2A]). Trace plots (Fig. [2B]) of the (Frobenius) norms for linear dynamics samples \( \{h^{(j)}, q^{(j)}, \sigma^{(j)}, A^{(j)}, b^{(j)}, Q^{(j)}\} \) show rapid mixing and convergence for each DPFA. The fitted mean firing rate (mean response) (Fig. [2C]), \( \mu^{(j)} \) and \( X^{(j)} \) (Fig. [2D]) match the ground truth well. The convergence is fast, especially in terms of mean response and population baseline \( \mu^{(j)} \). Together, these results demonstrate the identifiability of the DPFA with appropriate constraints.

We then compare the 10-cluster model with the simplified model ignoring the clustering structure (1-cluster model). The \( p \) for 1-cluster model is 14, chosen by 5-fold speckled cross-validation described in [Williams et al., 2020]. To evaluate the fitting performance of these two models, we
hold out 1/2 of the data and compare the held-out log-likelihood per spike. The distribution training log-likelihood is evaluated by averaging over the samples in short chains (e.g. iteration 50 to 100), which is justified by the observed fast convergence. The same procedure is replicated for 50 times. In this case, ignoring the clustering structure leads to a worse performance (Fig 2E), and the single population analysis cannot describe the data as accurately, since the input is non-homogeneous and the data is global nonlinear.

Figure 2: Bayesian inference with labeled data Here we simulate 10 clusters and assess convergence and mixing of for the DPFA. A. Traceplot of the log-likelihood per spike for all neurons and the first 3 clusters. B. The traceplot of (Frobenius) norms of linear dynamics of $\mu^{(j)}$ and $X^{(j)}$ for each cluster (showing the first 3). C. The true and the fitted mean firing rate, showing the averages over samples from iteration 5000 to 10000. D. The true (black) and the fitted (colored) population baseline and latent factor. The $X_{sl}$ denotes the $l$-th latent factor (i.e. the $l$-th column of $X$). The dashed lines show the 95% highest posterior density (HPD) interval. The cosine (the "overlap") between true values and posterior means shown besides. E. Comparison of the held-out likelihood per spike when fitting to 1/2 of the data: 1) 10-cluster model where each cluster has $p = 2$ (true value), and 2) 1-cluster model where a single DPFA describes all neurons, with $p = 14$ selected by 5-fold speckled cross-validation. Dots denote results from individual short (iterations 50-100), independent chains.

3.2 Clustering

Using the same simulation, we now infer the cluster labels. The latent factor dimension was first optimized with $p = 2$ selected by 5-fold speckled CV on small chains (100 iterations). We then compare three chains fitting with all data: two unique chains initialized using a single cluster and one chain initialized with $N = 50$ clusters (i.e. all clusters are singletons). Traceplots (Fig. 2A) of training log-likelihood and number of clusters show that all chains converge. When fit to the full data or only half, the number of clusters converges to 10, although the prior over the number of clusters is $K \sim Geometric(0.2)$. When the recording length is sufficiently long, the likelihood will dominate, and the number of clusters will not be much affected by the prior setting. The true mean firing rate for each neuron (Fig. 2B) can be well recovered, even with half data held out. To evaluate cluster membership, here we show a similarity matrix where the entry $(i, l)$ is the posterior probability that data points $i$ and $l$ belong to the same cluster. The clustering results for all 4 chains recover the true clusters, no matter what the the starting assignment is (Fig. 2C), which suggest the convergence of MCMC. The overall performance of the full model, where cluster membership is inferred alongside the latent states, is similar to the case when cluster labels are known and substantially higher than the 1-cluster model (Fig. 2D and E).
Figure 3: **Bayesian clustering** The same simulation setting as in Fig. 2 but inferring cluster labels from spike observations alone. 

A. The trace plots of training log-likelihood per spike and the number of clusters. The model is fitted by using all and half data as training with three chains shown for full data (initialized with a single cluster, C1 or \( N = 50 \) clusters, CN). 

B. The true and fitted mean firing rate, averaging from iteration 500 to 1000. 

C. The posterior similarity matrix for all chains (rows and columns ordered according to the ground truth). 

D. Traceplot of the held-out (1/2) log-likelihood for 1) the 10-cluster model with labels known, 2) the full model estimating labels and optimizing \( p \), and 3) the 1-cluster, single population model with \( p = 14 \) chosen by CV. 

E. Held-out log-likelihood per spike for each model (samples for iteration 500 to 1000). 

4 Multi-region neural spike recordings

We then apply the proposed clustering method to the Allen Institute Visual Coding Neuropixels dataset. The dataset contains spiking activity from hundreds of neurons from multiple brain regions of an awake mouse. See detailed data description in (Siegle et al., 2021). Here we investigate the clustering structure of neurons from four anatomical sites (83 neurons): 1) hippocampal CA1 (24 neurons), 2) dorsal part of the lateral geniculate complex (LGD, 36 neurons), 3) lateral posterior nucleus of the thalamus (LP, 12 neurons) and 4) primary visual cortex (VISp, 11 neurons). And we analyze responses to 20s epochs during three visual stimuli: drifting gratings, spontaneous activity, and natural movies. Only neurons with rates > 1Hz within the selected epochs are included (72% of 115 neurons) and we analyze data with 40ms bins. We use a Geometric(0.33) prior over the number of clusters, such that \( p(k \leq 4) = 0.8 \).

In responses to drifting gratings, the 5-fold speckled CV log-likelihood is optimized with \( p = 2 \), and, as in the simulations, the log-likelihood and number of clusters show rapid convergence and mixing (Fig 4A and B). Low firing rates and short recording lengths tend to cause confusions in clustering, reflecting uncertainty in cluster membership for neurons with little information. Here the average number of clusters is 16. To summarize the clustering results stored as posterior samples in MCMC, we give the single estimate for cluster indices \( \hat{z} \), by maximizing the posterior expected adjusted Rand index (maxPEAR, Fritsch and Ickstadt 2009). The maxPEAR-sorted neural activity and posterior similarity matrix are shown in Fig. 4C and D. Results sorted by Maximum a posteriori (MAP) estimate are similar and are shown in the Appendix (Fig. 5). To examine the relationship between the clustering results and anatomy, we additionally sort the neurons according anatomical labels (upper left panel in Fig. 4E). Although many identified clusters are neurons from the same anatomical area,
clusters also include neurons from different regions and neurons within a region are often clustered into separate populations (\(P(\text{neuron } i, l \mid \text{ same region}) = z_i = z_l, \{y_i\}_m, i = 1, \ldots, N) = 0.57\). Together, these results suggest that a simple assignment of populations based on anatomy may not accurately represent the latent structure.

We then evaluate the clustering patterns for different visual stimuli. We run 2 independent chains for each epoch (results from the second chain in Fig. 2D). The similarity matrices show that the pattern is consistent for the same epoch, but will change along the time even under the same experimental settings (D1 vs. D2 and S1 vs. S2). The changes in the clustering patterns may suggest long-term drift for neuron interactions. To quantify the observations, we evaluate the adjusted Rand index (ARI) of maxPEAR estimates (bottom right panel in Fig. 4E). Between-epoch comparisons tend to have lower similarity (average ARI from comparing 2 chains for each epoch) than within-epoch comparisons (different chains) for both maxPEAR and MAP (Fig. 4F).

The MAP number of clusters is largest (18) for the natural movie, suggesting this epoch has the most severe global non-linearity issue. Here we compare three models: 1) clustering model with \(p = 2\), 2) single cluster model, with \(p = 8\) selected by 5-fold speckled CV and 3) anatomical cluster model. The anatomical cluster model fits a single DPFA for each region, using the anatomical labels to define the clusters explicitly. Using cross-validation, we find that the optimized dimension for each region is \(p = (1, 11, 5, 3)\), respectively. We find that the single cluster model tends to underfit the data, while the anatomical cluster model tends to overfit the data (Fig. 4F).

Figure 4: Application in Neuropixels data. A. and B. The trace plots of log-likelihood per spike and number of clusters for drifting grating responses. All results are averages from iteration 500 to 1000. C. The observed spikes and fitted mean firing rate, sorted by the maxPEAR label. D. The posterior similarity matrix, sorted by the maxPEAR label. E. The posterior similarity matrices for 4 adjacent epochs and 1 further epoch with different visual stimuli, sorted according to maxPEAR estimate and anatomical label in the first drifting grating epoch. The last panel shows the adjusted Rand index of the maxPEAR estimates. The diagonal is the ARI between two chains for the same data, while off-diagonal values show the mean ARI of maxPEAR for the four comparisons between two chains from two different epochs. F. For the natural epoch, we hold out 1/2 data as the training, and show the histograms of training and held-out log-likelihood per spike, from iteration 500 to 1000 for three models: 1) clustering model, 2) single cluster model, and 3) anatomical cluster model.

5 Discussion

Here we introduce a Bayesian approach to cluster neural spike trains by MCMC. Previous approaches to multi-population latent variable modeling have used anatomical information to label distinct groups of neurons, but this choice is somewhat arbitrary. Brain region and cell-type, for instance, can give
contradictory population labels. The proposed method groups neurons by common latent factors, which may be useful for identifying "functional populations" of neurons. Here we use a mixDPFA model and infer the number of clusters by MFM with a partition-based algorithm similar to DPM. MFM may be more conceptually appropriate than DPM, since the number of neural "populations" is unknown but finite. Additionally, MFM produces more concentrated, evenly dispersed clusters (see Miller and Harrison 2018 for detailed discussion). The mixture modeling approach may also be appropriate in cases where neurons share non-homogeneous inputs, since it can approximate global nonlinearity with a mixture of locally linear models. Here we find that the mixture model outperforms globally linear (1-cluster) models in simulations and with experimental data.

Although the proposed method can describe data and cluster neural spiking activity successfully, there are some potential improvements. Firstly, as mentioned above, the unconstrained model does not have unique solutions. To ensure model identifiable, we put diagonal constraints on $A^{(j)}$ and $Q^{(j)}$ and constrain $\mu^{(j)}$ and $X^{(j)}$ to have mean zero. The assumption that $A^{(j)}$ and $Q^{(j)}$ are diagonal does not allow interaction between latent factors. However, these interactions could be allowed by instead constraining $X^{(j)}X^{(j)}$ to be diagonal (Krzanowski and Marriott 1994b,a; Fokoué and Titterington 2003). Such a constraint could allow unique solutions for the (P)LDS and GPFA. A second potential improvement would be to automatically infer the dimension of the latent factors (states). In this paper, we assume $p_j$ is the same for all clusters, for convenience. $p$ is a pre-selected value or can be selected by cross-validation (CV). This may limit the accuracy of the model, since populations of neurons in experimental data are likely to have different latent dimensionalities. In future work, it would also be possible to treat $p_j$ as a parameter and sample the posterior by a reversible-jump (RJ)MCMC (Lopes and West 2004), birth-death (BD)MCMC (Stephens 2000) [Fokoué and Titterington 2003], or adaptive Gibbs sampling with shrinkage prior on $X^{(j)}$ (Bhattacharya and Dunson 2011). Although RJMCMC and BD MCMC can be easily implemented, they perform poorly for high dimensional data and may be sensitive to priors. Adaptive Gibbs sampling with shrinkage, on the other hand, has been implemented with the infinite mixture of infinite factor analyzers (IMIFA, Murphy et al. 2020). The same idea may be useful here with an additional prior on linear dynamics ($A^{(j)}$, $b^{(j)}$, and $Q^{(j)}$) to encourage shrinkage in $X^{(j)}$. Finally, a deterministic approximation of MCMC, such as variational inference, may be more computationally efficient. Standard methods for fitting the PLDS could be used directly in the VI updates, and if we further use a stick-breaking representation for the MFM model, it would be straightforward to use VI for clustering as well, similar to (Blei and Jordan 2006).

As the number of neurons and brain regions that neuroscientists are able to record simultaneously continues to grow, understanding the latent structure of multiple populations will be a major statistical challenge. The Bayesian approach to clustering neural spike trains introduced here converges fast and is insensitive to the initial cluster assignments, and may, thus, be a useful tool for identifying "functional populations" of neurons.

**Acknowledgments**

This material is based upon work supported by the National Science Foundation under Grant No. 1931249.
References


Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper’s contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [No]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]

3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you’re using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]