Bayesian Clustering of Neural Activity with a Mixture of Dynamic Poisson Factor Analyzers

Ganchao Wei*
Department of Statistics, University of Connecticut

Ian H. Stevenson
Department of Psychological Sciences, University of Connecticut

Xiaojing Wang
Department of Statistics, University of Connecticut

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 (DPFA) model, with the number of clusters and dimension of latent states treated as unknown parameters. To analyze DPFA model, we propose a 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 is insensitive to the initial cluster assignments. We then apply the proposed mixture of DPFA 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...

*Corresponding author: ganchao.wei@uconn.edu
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 [Pandarinath 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, Emtiyaz 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: it requires we predetermine 1) the number of clusters and 2) dimension of latent vectors for each cluster, and 3) the clustering results are often sensitive to the initial cluster assignment.

Here we cluster the neurons by a mixture of dynamic Poisson factor analyzers (DPFA). 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. Both the number of clusters and latent factor dimensions are treated as unknown parameters in the mixture of DPFA, and the posteriors are sampled using Markov Chain Monte Carlo (MCMC). To sample the latent factors, we develop an efficient Metropolis-Hasting algorithm with the Polya-Gamma data augmentation technique [Polson et al., 2013]. The number of latent factors for each cluster is sampled by a birth-and-death MCMC (BDMCMC) algorithm [Stephens, 2000]. The cluster indices are sampled by a partition-based algorithm developed in mixture of finite mixtures (MFM) model [Miller and Harrison, 2018]. To improve mixing, we evaluate the approximated marginal likelihood for both BDMCMC and clustering, by Poisson-Gamma conjugacy. 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 (DPFA) to cluster neurons based on multi-population latent structure. Both the number of mixture components and the number of latent factors in each cluster are treated as unknown parameters and the posteriors are sampled by MCMC. In this section, we first provide the single population DPFA model 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 mixture of DPFA. We finally provide methods for sampling parameters, with more details in Appendix A

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 dynamic 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 $x_{t}^{(j)} \in \mathbb{R}^p$ and population baseline $\mu_t^{(j)}$ as follows:

$$
y_{it} \sim \text{Poi}(\lambda_{it}),$$

$$
\log \lambda_{it} = \delta_i + \mu_{t}^{(j)} + c_i'x_{t}^{(j)},
$$
Figure 1: Model overview. A. 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 to define populations based on neural activity directly. B. 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). C. 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.

with \( c_i \sim N(0, I_p) \). The neuron-specific baseline \( \delta_i \) is a constant across time for the i\textsuperscript{th} neuron and unrelated to the cluster assignment. Further, we assume the population baseline \( \mu_i(j) \) and the latent state \( x_i(j) \) evolve linearly over time with Gaussian noise as following

\[
\mu_{i+1}(j) = g(j) + h(j) \mu_i(j) + \epsilon_i(j), \\
x_{i+1}(j) = b(j) + A(j) x_i(j) + \eta_i(j),
\]

where \( \epsilon_i(j) \sim N(0, \sigma^2(j)) \) and \( \eta_i(j) \sim N_{p_j}(0, Q(j)) \).

If we denote \( \lambda_i = (\lambda_{i1}, \ldots, \lambda_{iT})' \), \( \mu^{(j)} = (\mu^{(j)}_1, \ldots, \mu^{(j)}_T)' \) and \( X^{(j)} = (x^{(j)}_1, \ldots, x^{(j)}_T)' \), the proposed model can be rewritten as

\[
y_i \sim Pois(\lambda_i), \\
\log \lambda_i = \delta_i 1_T + \mu^{(j)} + X^{(j)} c_i.
\]
Generally, a factor model is consistent only when $T/N \to 0$ [Johnstone and Lu, 2009], 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 $p_j > 1$, the model is only identifiable up to orthogonal rotation on $X^{(j)}$, with $c_i \sim N(0, I_{p_j})$. 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 $X^{(j)}$. To make the model identifiable and encourage clustering based on the trajectories of latent factors, we assume $A^{(j)}$ and $Q^{(j)}$ are diagonal [Peña and Poncela, 2004, Lopes et al., 2008], 

\[ \sum_{t=1}^{T} \mu^{(j)}_t = 0 \]

and \( \sum_{t=1}^{T} X^{(j)}_t = 0 \).

Given the parameters of the $j$-th cluster $\theta^{(j)} = \{ \mu^{(j)}, X^{(j)}, h^{(j)}, g^{(j)}, \sigma^2^{(j)}, A^{(j)}, b^{(j)}, Q^{(j)} \}$, the spike counts of neuron $i$ are generated by the dynamic Poisson factor analyzer (DPFA) model as 

\[ y_i \mid z_i = j \sim DPFA(\delta_i, c_i, \theta^{(z_i)}) \]

To facilitate the Bayesian computation, we have to impose priors $H$ on $\theta^{(j)}$ except $\mu^{(j)}$ and $X^{(j)}$, see more details of prior settings in Appendix A.

### 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. 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:

\[ k \sim f_k, \quad f_k \text{ is a p.m.f. on } \{1, 2, \ldots \}, \]

\[ \pi = (\pi_1, \ldots, \pi_k) \sim Dir_k(\gamma_1, \ldots, \gamma_k) \quad \text{given } k, \]

\[ z_1, \ldots, z_N \overset{i.i.d.}{\sim} \pi \quad \text{given } \pi, \]

\[ \theta^{(1)}, \ldots, \theta^{(k)} \overset{i.i.d.}{\sim} H \quad \text{given } k, \]

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

where p.m.f denotes the probability mass function. By using the MFM, we can integrate the field knowledge about the number of neural populations into our analysis. In the analysis of this paper, we assume $k$ follows a geometric distribution, i.e., $k \sim \text{Geometric}(\alpha)$ with its density defined as $f_k(k|\alpha) = (1 - \alpha)^{k-1}\alpha$ for $k = 1, 2, \ldots$, and let $\gamma = 1$. The complete generative model is summarized in a graphical form shown in Fig. 1C.

### 2.3 Inference

Here the posteriors of the proposed mixture of DPFA model are sampled by an MCMC algorithm (see details in the Appendix A). In each iteration, we sequentially sample 1) the model parameters assuming the known cluster indices, 2) the cluster indices given the model parameters, and 3) the dimension of latent states for each cluster.
When sampling the (labeled) model parameters, the latent state $X^{(j)}$ and population baseline $\mu^{(j)}$ have no closed-form full conditional distributions. Here, we sample the posteriors by a Pólya-Gamma (PG) data augmentation approach [Windle et al., 2013, Linderman et al., 2017, 2016, Polson et al., 2013] with an additional Metropolis-Hastings (MH) step [Metropolis et al., 1953, Hastings, 1970]. Although the Poisson observations don’t follow the PG augmentation scheme directly, we can approximate the Poisson distribution by a negative binomial (NB) distribution, i.e. $\lim_{r \to \infty} \text{NB}(r, \sigma(\psi - \log r)) = \text{Poisson}(e^\psi)$, where $\sigma(\psi) = e^\psi/(1 + e^\psi)$ and $\text{NB}(r, p)$ denotes the NB distribution with $rp/(1 - p)$ as its expectation. By approximating Poisson with NB, we can sample $X^{(j)}$ and $\mu^{(j)}$ by PG data augmentation technique with forward-filtering-backward-sampling (FFBS) algorithm. To ensure the samples are from the exact posteriors, we use samples from FFBS algorithm as a proposal and employ a Metropolis-Hastings (MH) step to reject or accept the proposal. In this step, the dispersion parameter $r$ in NB distribution becomes a tuning parameter, to balance acceptance rate and autocorrelation in MH.

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). 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 algorithm, Fokoué and Titterington [2003]), the chain has poor mixing and stops after a few iterations, because of the high dimensionality. The heavy dependency on the starting point when fitting the mixture of PLDS (mixPLDS, Buesing et al. [2014]) model may suggest a similar problem. To resolve this, we evaluate the marginal likelihood by integrating out the neuron-specific $c_i$, i.e., the marginal likelihood of neuron $i$ in cluster $j$ is computed by

$$M_{\theta^{(j)}}(y_i) = P(y_i|\theta^{(j)}, \delta_i) = \int P(y_i|\theta^{(j)}, \delta_i, c_i) P(c_i) dc_i. \quad (3)$$

However, this marginal likelihood has no closed form. Though we may evaluate it by a Laplace approximation, but iterating over all potential clusters for each neuron is computationally intensive. To make faster clustering, we approximate the marginal likelihood by utilizing a Poisson-Gamma conjugacy. This approach has been previously utilized to approximate posteriors [El-Sayyad, 1973] and predictive distributions [Chan and Vasconcelos, 2009]. In our situation, since $c_i \sim N(0, I_\nu)$, we have $\lambda_{it} = \exp(\delta_i + \mu^{(j)} + c_i'x_t^{(j)}) \sim \text{lognormal}(\delta_i + \mu^{(j)}, x_t^{(j)}, x_t^{(j)})$, and then we can approximate this lognormal distribution by a gamma distribution, i.e., assume $\lambda_{it}$ follows $\text{Gamma}(a_{it}, b_{it})$ with $a_{it} = (x_t^{(j)}x_t^{(j)})^{-1}$ and $b_{it} = x_t^{(j)}x_t^{(j)}e^{\delta_i + \mu^{(j)}}$. Then, by the conjugate property with Poisson and Gamma random variables, we have

$$P(y_{it}|\theta^{(j)}, \delta_i) = \int P(y_{it}|\lambda_{it}) P(\lambda_{it}) d\lambda_{it} \approx NB(y_{it}|\nu_{it}, p_{it}),$$
with $\nu_{it} = a_{it}$ and $p_{it} = 1/(1 + b_{it})$. Further, noticing that we have the conditional independence assumption for $P(y_{i} | \theta^{(j)}, \delta_{i})$, that is $P(y_{i} | \theta^{(j)}, \delta_{i}) = \prod_{t=1}^{T} P(y_{it} | \theta^{(j)}, \delta_{i})$, we then have a closed-form for Equation (3). Another possible idea is to approximate the log-likelihood by second-order polynomials, with coefficients determined by Chebyshev polynomial approximation [Keeley et al., 2019]. However, we find that this approximation doesn’t work well in practice when spike counts have a wide range.

To sample the number of latent factors $p_{j}$ in each cluster, we implement the birth-and-death MCMC (BDMCMC) as in Stephens [2000], Fokoué and Titterington [2003]. To efficiently simulating a birth-death Markov point process to estimate $p_{j}$ for high dimensional (large $T$) DPFA model, we again evaluate the approximated marginal likelihood by integrating out the neuron-specific $c_{i}$ instead of the full likelihood.

See details of MCMC sampling in the Appendix A. The model is implemented in MATLAB and the code is available at https://github.com/weigcdsb/MFM_DPFA_clean.

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 inference on $p_{j}$. We then infer the labels to evaluate clustering performance.

3.1 Labeled data

We first simulate 5 neurons sharing a common $p_{j} = 2$ dimensional latent factors with $T = 1000$, i.e. 5 neurons within one cluster. The 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 14 and 32 evenly spaced knots. The label is assumed to be known to check fitting performance of DPFA.

Running MCMC for 10,000 iterations, we find the chain converges fast by checking the trace plots for $\delta_{i}$ and $||\mu^{(j)}||_2$. The dimension of latent factors $p_{j}$ can also be estimated by BDMCMC successfully. The posterior mean of $\mu^{(j)}$ matches the ground truth well, although the uncertainty is higher if we don’t assume $p_{j}$ is known. Since the $p_{j}$ is not fixed, showing the fitting result of $X^{(j)}$ is not easy. However, if we fix $p_{j}$ as the ground truth, we can successfully recover the ground truth of $X^{(j)}$. This suggests that the model is identifiable.

3.2 clustering

We then simulate 10 clusters with 5 neurons in each, with recording length $T = 1000$ and $p_{j} = 2$ dimensional latent factors for each cluster. The DPFA
Figure 2: Bayesian inference with labeled data. Here we simulate one cluster with 5 neurons and assess convergence and mixing of the DPFA. 

A. Trace plot of $\delta_i$. B. The traceplot of $||\mu^{(j)}||_2$, the dashed red line shows the true value. C. Histogram of posterior samples of $p_j$, discarding the first 2500 iterations as burn-in. The ground truth is marked as red dashed line. D. The true (black) and the fitted (colored) population baseline and latent factor, when $p_j$ is known or unknown. The results with fixed $p_j$ are shown in orange, while the results with unknown $p_j$ are shown in blue. The dashed lines show the 95% highest posterior density (HPD) interval. The cosine (“overlap”) between true values and posterior means shown besides.

parameters for each cluster are generated as the one population in the previous simulation. In other words, $\delta_i \sim N(0, 0.5^2), c_i \sim N(0, I_2), \mu^{(j)}$ and $X^{(j)}$ are generated by the spline interpolation on 10 to 35 evenly spaced knots. The label is unknown to check the performance of clustering.

Here, we compare the clustering performance for 4 chains with 10,000 iterations: 1 chain with $p_j = 2$ is known, while 3 chains $p_j$ is known. For chains with unknown $p_j$, 2 of them are initialized using a single cluster, and 1 are initialized with $N = 50$ clusters. Trace plots of cluster numbers show that all chains converge to ground truth (10 clusters), with prior $K \sim Geometric(0.2)$. To evaluate cluster membership, we further evaluate similarity matrices where the entry $(i, l)$ is the posterior probability that data points $i$ and $l$ belong to the same cluster. All four matrices show that these chains can recover the true assignment, with mild confusions between cluster 2 and 6, and cluster 3 and 7. These results show that our method is not sensitive to the initial assignment and whether $p_j$ for each cluster is known or not will not change the clustering results.
Figure 3: **Bayesian clustering** Simulate 10 clusters with 5 neurons in each, and infer cluster labels from spike observations alone. **A.** The trace plots of cluster numbers for four chains, with $p_j$ known or not and with different initial assignments. **B.** The similarity matrices for four chains. Discard the first 2500 iterations as burn-in.

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 in 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 50s epochs ($T = 500$ with bin size = 0.1s) during three visual stimuli: drifting gratings, spontaneous activity, and natural movies. Only neurons with rates $\geq 1$Hz within the selected epochs are included and we analyze data with 40ms bins. We use a $\text{Geometric}(0.33)$ prior over the number of clusters, such that $p(k \leq 4) = 0.8$.

In responses to drifting gratings, the trace plots of cluster numbers show convergence and mixing for a chain with 10,000 iterations. Low firing rates tend to cause confusions in clustering, and the average number of clusters is 9. To summarize the clustering results stored as posterior samples in MCMC, we give the single estimate for cluster indices $\hat{z}_i$ by maximizing the posterior expected adjusted Rand index (maxPEAR, Fritsch and Ickstadt 2009). The maxPEAR-sorted posterior similarity matrix and neural activity are shown in Fig. XX. Results sorted by Maximum a posteriori (MAP) estimate are similar and are shown in the Appendix B. To examine the relationship between the clustering
results and anatomy, we additionally sort the neurons according anatomical labels (upper left panel in Fig. 4D). 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 ($P(\text{neuron}_i, l \text{ in the same region} | z_i = z_l, \{y_i\}_{i=1}^N) = 0.51$). 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. 5D). 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. 3E). 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. 5C).

Figure 4: Application in Neuropixels data. A. The trace plot and histogram of cluster numbers, in the first drifting grating epoch (Drifting 1, D1). The first 2500 iterations are discarded as burn-in when plotting the histogram. B. The posterior similarity matrix of the first chain in D1 epoch, sorted by the maxPEAR label. C. The observed spikes and fitted mean firing rate, sorted by the maxPEAR label. D. 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 D1 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.
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 mixture of DPFA for clustering, with cluster number inferred by a partition-based algorithm for MFM and dimension of latent states sampled by BDMCMC. Conceptually, MFM may be more 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. Besides the BDMCMC, there are some other methods to infer the number of latent states in Gaussian factor analysis (FA) model, such as putting a multiplicative Gamma process (MGP) prior [Bhattacharya and Dunson, 2011], multiplicative exponential process (MEP) prior [Wang et al., 2016], Beta process (BP) prior [Paisley and Carin, 2009, Chen et al., 2010] or Indian Buffet process (IBP) [Knowles and Ghahramani, 2007, 2011, Roćková and George, 2016] prior on the loading matrix. Although these methods may be better than BDMCMC, they cannot be implemented in our case easily. Since putting prior on $c_i$ makes evaluation of marginal likelihood difficult, while putting prior on $X^{(j)}$ will break the assumed linear dynamics. Motivated by these methods, it may be possible to put prior on linear dynamics ($A^{(j)}$, $b^{(j)}$ and $Q^{(j)}$) to encourage shrinkage in $X^{(j)}$.

Although the proposed method can describe data and cluster neural spiking activity successfully, there are some potential improvements. Firstly, to make 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, 1994a,b, Fokoué and Titterington, 2003]. Such a constraint could allow unique solutions for the (P)LDS and GPFA. Secondly, 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

G. Arminger, P. Stein, and J. Wittenberg. Mixtures of conditional mean-and
covariance-structure models. *Psychometrika*, 64(4):475–494, 1999. ISSN
00333123. doi: 10.1007/BF02294568. URL /record/2000-13246-005.

A. Bhattacharya and D. B. Dunson. Sparse Bayesian infinite factor models.
*Biometrika*, 98(2):291, 2011. ISSN 00063444. doi: 10.1093/BIOMET/ASR013.

D. M. Blei and M. I. Jordan. Variational Inference for Dirichlet Process Mixtures.
*Bayesian Analysis*, 1(1):121–144, 2006. URL http://www.cs.berkeley.edu/
$\sim$blei/.

L. Buesing, T. A. Machado, J. P. Cunningham, and L. Paninski. Clustered factor analysis of multineuronal spike data. In Z. Ghahramani,
M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL
https://proceedings.neurips.cc/paper/2014/file/e8dfff4676a47048d6f0c4ef899593dd-paper.pdf.

C. K. Carter and R. Kohn. On Gibbs sampling for state space models. *Biometrika*,
81(3):541–553, 1994. ISSN 0006-3444. doi: 10.1093/biomet/81.3.541. URL
https://doi.org/10.1093/biomet/81.3.541.

A. B. Chan and N. Vasconcelos. Bayesian poisson regression for crowd counting.
*Proceedings of the IEEE International Conference on Computer Vision*, pages
545–551, 2009. doi: 10.1109/ICCV.2009.5459191.

M. Chen, J. Silva, J. Paisley, C. Wang, D. Dunson, and L. Carin. Compressive
sensing on manifolds using a nonparametric mixture of factor analyzers:
Algorithm and performance bounds. *IEEE Transactions on Signal Processing*,
58(12):6140–6155, 2010. doi: 10.1109/TSP.2010.2070796.

G. M. El-Sayyad. Bayesian and Classical Analysis of Poisson Regression. *Journal of the Royal Statistical Society: Series B (Methodological)*, 35(3):445–451, 1973. ISSN
2517-6161. doi: 10.1111/J.2517-6161.1973.TB00972.X.

M. Emtiyaz Khan, A. Aravkin, M. Friedlander, and M. Seeger. Fast dual vari-
ational inference for non-conjugate latent gaussian models. In S. Dasgupta
E. Fokoué and D. M. Titterington. Mixtures of factor analysers. Bayesian estimation and inference by stochastic simulation. *Machine Learning*, 50(1-2): 73–94, 2003. ISSN 0885-6125. doi: 10.1023/A:1020297828025.

A. Fritsch and K. Ickstadt. Improved criteria for clustering based on the posterior similarity matrix. *Bayesian Analysis*, 4(2):367–391, 2009. doi: 10.1214/09-BA414. URL https://doi.org/10.1214/09-BA414

S. Frühwirth-Schnatter. Data augmentation and dynamic linear models. *Journal of Time Series Analysis*, 15:183–202, 1994.

Z. Ghahramani and G. E. Hinton. The EM Algorithm for Mixtures of Factor Analyzers. 1996.

J. Glaser, M. Whiteway, J. P. Cunningham, L. Paninski, and S. Linderman. Recurrent Switching Dynamical Systems Models for Multiple Interacting Neural Populations. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 14867–14878. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/aa1f5f73327ba40d47ebce155e785aaf-Paper.pdf.

W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. ISSN 0006-3444. doi: 10.1093/biomet/57.1.97. URL https://doi.org/10.1093/biomet/57.1.97.

S. Jain and R. M. Neal. A split-merge markov chain monte carlo procedure for the dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13(1):158–182, 2004. doi: 10.1198/1061860043001. URL https://doi.org/10.1198/1061860043001.

S. Jain and R. M. Neal. Splitting and merging components of a nonconjugate Dirichlet process mixture model. *Bayesian Analysis*, 2(3):445 – 472, 2007. doi: 10.1214/07-BA219. URL https://doi.org/10.1214/07-BA219.

I. M. Johnstone and A. Y. Lu. On Consistency and Sparsity for Principal Components Analysis in High Dimensions. *Journal of the American Statistical Association*, 104(486):682, 2009. ISSN 01621459. doi: 10.1198/JASA.2009.0121.

M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An Introduction to Variational Methods for Graphical Models. *Machine Learning* 1999 37:2, 37(2):183–233, 1999. ISSN 1573-0565. doi: 10.1023/A:1007665907178. URL https://link.springer.com/article/10.1023/A:1007665907178.
J. J. Jun, N. A. Steinmetz, J. H. Siegle, et al. Fully integrated silicon probes for high-density recording of neural activity. *Nature*, 551(7679):232–236, 2017. ISSN 1476-4687. doi: 10.1038/nature24636. URL https://www.nature.com/articles/nature24636.

S. L. Keeley, D. M. Zoltowski, Y. Yu, J. L. Yates, S. L. Smith, and J. W. Pillow. Efficient non-conjugate Gaussian process factor models for spike count data using polynomial approximations. *37th International Conference on Machine Learning, ICML 2020*, PartF16814:5133–5142, 2019.

D. Knowles and Z. Ghahramani. Infinite sparse factor analysis and infinite independent components analysis. In M. E. Davies, C. J. James, S. A. Abdallah, and M. D. Plumbley, editors, *Independent Component Analysis and Signal Separation*, pages 381–388, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-74494-8.

D. Knowles and Z. Ghahramani. Nonparametric Bayesian sparse factor models with application to gene expression modeling. *The Annals of Applied Statistics*, 5(2B):1534 – 1552, 2011. doi: 10.1214/10-AOAS435. URL https://doi.org/10.1214/10-AOAS435.

W. J. Krzanowski and F. H. C. Marriott. Multivariate analysis. Part I. Distributions, ordination and inference. 1994a.

W. J. Krzanowski and F. H. C. Marriott. Multivariate analysis. Part 2: Classification, covariance structures and repeated measurements. 1994b.

S. Linderman, R. P. Adams, and J. W. Pillow. Bayesian latent structure discovery from multi-neuron recordings. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper/2016/file/708f3cf8100d5e71834b1db77dafa15d6-Paper.pdf.

S. Linderman, M. Johnson, A. Miller, R. Adams, D. Blei, and L. Paninski. Bayesian Learning and Inference in Recurrent Switching Linear Dynamical Systems. In A. Singh and J. Zhu, editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 914–922. PMLR, 2017. URL https://proceedings.mlr.press/v54/linderman17a.html.

H. F. Lopes, E. Salazar, and D. Gamerman. Spatial dynamic factor analysis. *Bayesian Analysis*, 3(4):759–792, 2008. ISSN 19360975. doi: 10.1214/08-BA329.

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

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equation of state calculations by fast computing machines. *The Journal*
J. W. Miller and M. T. Harrison. Mixture models with a prior on the number of components. *Journal of the American Statistical Association*, 113(521):340, 2018. doi: 10.1080/01621459.2016.1255636.

R. M. Neal. Markov Chain Sampling Methods for Dirichlet Process Mixture Models. *Journal of Computational and Graphical Statistics*, 9(2):249–265, 2000.

J. Paisley and L. Carin. Nonparametric factor analysis with beta process priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML ’09, page 777–784, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585161. doi: 10.1145/1553374.1553474. URL https://doi.org/10.1145/1553374.1553474.

C. Pandarinath, D. J. O’Shea, J. Collins, R. Jozefowicz, S. D. Stavisky, J. C. Kao, E. M. Trautmann, M. T. Kaufman, S. I. Ryu, L. R. Hochberg, J. M. Henderson, K. V. Shenoy, L. F. Abbott, and D. Sussillo. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10):805–815, 2018. ISSN 1548-7105. doi: 10.1038/s41592-018-0109-9. URL https://www.nature.com/articles/s41592-018-0109-9.

D. Peña and P. Poncela. Forecasting with nonstationary dynamic factor models. *Journal of Econometrics*, 119(2):291–321, 2004. ISSN 0304-4076. doi: 10.1016/S0304-4076(03)00198-2.

N. G. Polson, J. G. Scott, and J. Windle. Bayesian inference for logistic models using pólya-gamma latent variables. *Journal of the American Statistical Association*, 108(504):1339–1349, 2013. doi: 10.1080/01621459.2013.829001. URL https://doi.org/10.1080/01621459.2013.829001.

V. Ročková and E. I. George. Fast bayesian factor analysis via automatic rotations to sparsity. *Journal of the American Statistical Association*, 111(516):1608–1622, 2016. doi: 10.1080/01621459.2015.1100620. URL https://doi.org/10.1080/01621459.2015.1100620.

J. D. Semedo, A. Zandvakili, C. K. Machens, B. M. Yu, and A. Kohn. Cortical areas interact through a communication subspace. *Neuron*, 102(1):249–259.e4, 2019. ISSN 0896-6273. doi: https://doi.org/10.1016/j.neuron.2019.01.026. URL https://www.sciencedirect.com/science/article/pii/S0896627319300534.

D. Sen, S. Patra, and D. Dunson. Constrained inference through posterior projections, 2018. URL https://arxiv.org/abs/1812.05741.

J. H. Siegle, X. Jia, S. Durand, et al. Survey of spiking in the mouse visual system reveals functional hierarchy. *Nature*, 592(7852):86–92, 2021. ISSN 1476-4687. doi: 10.1038/s41586-020-03171-x. URL https://www.nature.com/articles/s41586-020-03171-x.
M. Stephens. Bayesian analysis of mixture models with an unknown number of components—an alternative to reversible jump methods. The Annals of Statistics, 28(1):40 – 74, 2000. doi: 10.1214/aos/1016120364. URL https://doi.org/10.1214/aos/1016120364.

V. Ventura. Traditional waveform based spike sorting yields biased rate code estimates. Proceedings of the National Academy of Sciences of the United States of America, 106(17):6921–6926, 2009. ISSN 00278424. doi: 10.1073/PNAS.0901771106/SUPPL-FILE/SI-APPENDIX.PDF. URL www.pnas.org/cgi/content/full.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. Foundations and Trends® in Machine Learning, 1(1–2):1–305, 2008. ISSN 1935-8237. doi: 10.1561/2200000001. URL http://dx.doi.org/10.1561/2200000001.

Y. Wang, A. Canale, and D. Dunson. Scalable geometric density estimation. In A. Gretton and C. C. Robert, editors, Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, volume 51 of Proceedings of Machine Learning Research, pages 857–865, Cadiz, Spain, 2016. PMLR. URL https://proceedings.mlr.press/v51/wang16e.html.

M. R. Whiteway, K. Socha, V. Bonin, and D. A. Butts. Characterizing the nonlinear structure of shared variability in cortical neuron populations using latent variable models. Neurons, behavior, data analysis and theory, 3(1), 2019. doi: 10.6080/K0NC5Z4X. URL /pmc/articles/PMC6779168/PMC6779168/?report=abstracthttps://www.ncbi.nlm.nih.gov/pmc/articles/PMC6779168/.

J. Windle, C. M. Carvalho, J. G. Scott, and L. Sun. Efficient Data Augmentation in Dynamic Models for Binary and Count Data. 2013. URL http://arxiv.org/abs/1308.0774.

A. Wu, N. A. Roy, S. Keeley, and J. W. Pillow. Gaussian process based nonlinear latent structure discovery in multivariate spike train data. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/b3b4d2dbedc99fe843f33dedb0f086f-Paper.pdf.

B. M. Yu, J. P. Cunningham, G. Santhanam, S. I. Ryu, K. V. Shenoy, and M. Sahani. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. Journal of Neurophysiology, 102(1):614–635, 2009. ISSN 15221598. doi: 10.1152/JN.90941.2008/ASSET/IMAGES/LARGE/Z9K007099508A002.JPEG. URL https://journals.physiology.org/doi/abs/10.1152/jn.90941.2008.
Appendices

A MCMC updates

The posteriors are sampled by a Gibbs sampler. In each iteration, the sampling scheme has three main steps: (1) sample the model parameters assuming known labels $z_i$ and fixed number of latent factors $p_j$, (2) sample the cluster indices $z_i$ given model parameters and (3) sample the number of latent factors $p_j$. Sampling of the model parameters (step 1) is conducted without considering constraints for $\mu_t^{(j)}$ and $x_t^{(j)}$ at first, and then we project the samples onto the constraint space for $\sum_{t=1}^{T} \mu_t^{(j)} = 0$ and $\sum_{t=1}^{T} x_t^{(j)} = 0$ as used in [Sen et al., 2018].

A.1 Update population baseline and latent factors

In this section, we provide details of sampling algorithms to draw the latent states $X^{(j)}$ and population baseline $\mu^{(j)}$ from the full conditional distribution. In order to explain the algorithm much clearer, we just focus on sampling in one cluster, and thus in this section we suspend the superscript $(j)$ or $j$ as the cluster index in our notations.

The priors for initial population baseline and latent factor are:

$$
\mu_1 \sim \mathcal{N}(0, 1),
$$

$$
x_1 \sim \mathcal{N}(0, I_p).
$$

Assume there are $n$ neurons in the given cluster. Then, $y_{it} \sim \text{Poi}(\lambda_{it})$, $\lambda_{it} = \delta_i + \tilde{c}_i \tilde{x}_t$ for $i = 1, \ldots, n$ and $t = 1, \ldots, T$. Here, $\tilde{c}_i = (1, c'_i)'$ and $\tilde{x}_t = (\mu_t, x'_t)'$.

Further define $\tilde{A}$, $\tilde{b}$ and $\tilde{Q}$, such that $\tilde{x}_t$ follows linear dynamics $\tilde{x}_{t+1} | \tilde{x}_t \sim \mathcal{N}(\tilde{A} \tilde{x}_t + \tilde{b}, \tilde{Q})$. The corresponding prior for $\tilde{x}_1$ is $\tilde{x}_1 \sim \mathcal{N}(m_0, V_0)$, where $m_0 = 0$ and $V_0 = I_{p+1}$. Under this scheme and notation, sampling $\tilde{X} = (\mu, X)$ from full conditional distribution is equivalent to sample parameters for dynamic Poisson GLM. In the following, we present the sampling idea with two major parts followed by the summarized algorithm.

A.1.1 Pólya-Gamma Augmentation

Although the mixture of the DPFA model does not directly follow the PG augmentation scheme [Polson et al., 2013], we can approximate the Poisson distribution by a negative binomial (NB) distribution, i.e., $\lim_{r \to \infty} \text{NB}(r, \sigma(\psi - \log r)) = \text{Poisson}(e^\psi)$, where $\sigma(\psi) = e^\psi / (1 + e^\psi)$ and $\text{NB}(r, p)$ denotes the NB distribution with $rp/(1 - p)$ as its expectation. We approximate the proposed DPFA model using a NB, then we can instead sample a mixture of NB factor analyzers using the PG scheme [Windle et al., 2013]. Further, we use the forward-filtering-backward-sampling (FFBS) algorithm [Carter and Kohn, 1994,
Frühwirth-Schnatter, 1994] to update the latent states $X$ and population baseline $\mu$. These two steps are summarized in the step 1 and step 2 of Algorithm 1 for updating $\tilde{X}$.

A.1.2 Metropolis-Hastings Step

In the next step, we use the samples of $\tilde{X}$ yielded from FFBS algorithm as a proposal, where we employ a Metropolis-Hastings (MH) step to reject or accept the proposal. In this step, the dispersion parameter $r$ in NB distribution becomes a tuning parameter, to balance acceptance rate and autocorrelation in MH. When $r$ is large, the approximation to Poisson observation is accurate and the MH performs similar to the Gibbs sampler. Here, we allow neurons at different time points to have unique tuning parameters $r_{it}$. The MH step is summarized in step 3 of Algorithm 1.

A.1.3 Algorithm

Here, we put these steps together and provide details as follows in Algorithm 1.

A.2 Update neuron-specific baseline and loading

We specify the prior for neuron-specific baseline $\delta_i$ as $\delta_i \sim \mathcal{N}(0, 1)$ and we have assumed the loading $c_i \sim \mathcal{N}(0, I_p)$. Then, from the matrix representation of DPFA in (1), i.e., $\log \lambda_i = \delta_i 1_T + \mu^{(j)} + X^{(j)} c_i$, it is easy to see that given $\mu^{(j)}$ and $X^{(j)}$ are known, the update of $\delta_i$ and $c_i$ is just a regular Bayesian Poisson regression problem. Thus, we can sample the full conditional distribution of $\delta_i$ and $c_i$ by a Hamilton Monte-Carlo (HMC) within the Gibbs sampler.

A.3 Update parameters of latent state

The parameters for linear dynamics are $h^{(j)}$, $g^{(j)}$, $\sigma^{2(j)}$, $A^{(j)}$, $b^{(j)}$ and $Q^{(j)}$. To make the model identifiable, we simply assume $A^{(j)} = \text{diag}(a_1^{(j)}, \ldots, a_p^{(j)})$ and $Q^{(j)} = \text{diag}(q_1^{(j)}, \ldots, q_p^{(j)})$. Therefore, we can update $A^{(j)}$, $b^{(j)}$ and $Q^{(j)}$ for each diagonal element separately, as the update in $h^{(j)}$, $g^{(j)}$ and $\sigma^{2(j)}$. Here, we update $h^{(j)}$, $g^{(j)}$ and $\sigma^{2(j)}$ as follows.

First, we specify the priors for $\sigma^{2(j)}$ following $\text{IG}(\nu_0/2, \nu_0 \sigma_0^2/2)$ and $(g^{(j)}, h^{(j)})' \sim \mathcal{N}(\tau_0, \sigma^{2(j)} \Lambda_0^{-1})$, with $\nu_0 = 1$, $\sigma_0 = 0.01$, $\tau_0 = (0, 1)'$ and $\Lambda_0 = I_2$. Here, the "IG" denotes the inverse-gamma distribution.

Denote $\mu_{2:T}^{(j)} = \left(\mu_2^{(j)}, \ldots, \mu_T^{(j)}\right)'$ and $\tilde{\mu}_{1:(T-1)}^{(j)} = \left(1_{T-1}, \mu_{1:(T-1)}^{(j)}\right)'$, with $\mu_{1:(T-1)}^{(j)} = \left(\mu_1^{(j)}, \ldots, \mu_{T-1}^{(j)}\right)'$. The full conditional distributions for $\sigma^{2(j)}$ and $(g^{(j)}, h^{(j)})'$. 

18
are:

$$\sigma^2(j)\{\mu^{(j)}_t\}_{t=1}^T \sim IG\left(\frac{\nu_0 + T - 1}{2}, \frac{\nu_0 \sigma_0^2 + \mu^{(j)}_{2:T} \mu^{(j)}_{1:T} + \tau_0' \Lambda_0 \tau_0 - \tau_0' \Lambda_0 \tau_0}{2}\right),$$

$$\left(g^{(j)}, h^{(j)}\right)' \{\mu^{(j)}_t\}_{t=1}^T \sim \mathcal{N}(\tau_n, \sigma^2(j) \Lambda_n^{-1}),$$

with $$\Lambda_n = \mu^{(j)}_{1:(T-1)} \mu^{(j)}_{1:(T-1)} + \Lambda_0$$, and $$\tau_n = \Lambda_n^{-1} \left(\mu^{(j)}_{1:(T-1)} \mu^{(j)}_{2:T} + \Lambda_0 \tau_0\right).$$

### A.4 Update labels

To update the cluster labels $$z_i$$, we use a partition based algorithm for MFM, similarly as described in Miller and Harrison 2018. Because of the non-conjugacy, the marginal likelihood in terms of all cluster parameters $$\theta(z)$$ can not be easily computed. This issue can be solved by introducing auxiliary variables, as in "Algorithm 8" in Neal 2000 for DPM. A similar algorithm can be used to infer the MFM.

Let $$C$$ denote a partition of neurons, and $$C\setminus i$$ denote the partition obtained by removing neuron $$i$$ from $$C$$.

1. Initialize $$C$$ and $$\{\theta(c) : c \in C\}$$ (e.g. one cluster).

2. Repeat the following steps $$G$$ times to obtain $$G$$ samples. For $$i = 1, \ldots, N$$:

   (a) in $$c \in C\setminus i$$ with probability $$\propto (|c| + \gamma)M_{\theta(c)}(y_i)$$, where $$\gamma$$ is defined in the MFM model in the main text (Equation 2) and $$M_{\theta(c)}(y_i)$$ denotes the marginal likelihood of neuron $$i$$ in cluster $$c$$, when integrating the loading $$c_i$$ out.

   (b) in a new cluster $$c^*$$ with probability $$\propto \gamma V_n(t+1)/V_n(t) M_{\theta(c^*)}(y_i)$$, where $$t$$ is the number of partitions obtained by removing the neuron $$i$$ and $$V_n(t) = \sum_{j=1}^{\infty} \left(\frac{j}{\gamma}\right)^{|c|} f_k(j)$$, with $$x^{(m)} = x(x+1) \cdots (x+m-1)$$, $$x^{(0)} = 1$$ and $$x^{(0)} = 1$$.

The update is an adaptation of partition-based algorithm for DPM [Neal, 2000], but with two substitutions: 1) replace $$|c_i|$$ by $$|c_i| + \gamma$$ and 2) replace $$\alpha$$ by $$\gamma V_n(t+1)/V_n(t)$$. See more details and discussions in [Miller and Harrison, 2018].

When evaluating the likelihood, we marginalize the cluster-independent loading $$c_i$$ out. This is necessary for the high dimensional situation, otherwise the chain will stop moving.

One issue with incremental Gibbs samplers such as Algorithm 3 and 8 in Neal [2000], when applied to DPM, is that mixing can be somewhat slow. To further improve the mixing, we may intersperse the "split-merge" Metropolis-Hasting updates [Jain and Neal, 2007, 2004] between Gibbs sweeps, as in [Miller and Harrison, 2018].
A.5 Update number of latent factors

The number of latent factors for each cluster $p_j$ is sampled by a birth-death MCMC method (BDMCMC), similar to Fokoué and Titterington [2003]. We put a truncated Poisson prior for $p_j$ with hyperparameter $\alpha = 2$, i.e. $P(p_j) \propto \alpha^{p_j}/p_j!$, for $j = 1, \ldots$ Set the birth rate as $\beta = 0.5$ and duration time as $\rho = 1$. Define $C = X^{(j)}_{*,1}, X^{(j)}_{*,2}, \ldots$, where $X^{(j)}_{*,l}$ denotes the $l$-th column of $X^{(j)}$. Let $C \cup \{X^{(j)}_{*,l}\}$ and $C \setminus \{X^{(j)}_{*,l}\}$ denote the addition and deletion of column $X^{(j)}_{*,l}$ from the current configuration $C$. Further let $M(\cdot)$ denote the marginal likelihood by integrating $c_i \sim N(0, I_{p_j})$ out. The resulting sampling scheme is summarized in the Algorithm 2.

B Supplementary Results for Neuropixels Application

This section shows supplementary results when applying the proposed model to the Neuropixels dataset (Multi-region neural spike recordings). Here, we show 1) results sorted by maximum a posteriori probability (MAP) estimates and 2) clustering results in another independent chain (Fig. 4).

Figure 5: Supplementary Results for Neuropixels Application. A. Posterior similarity matrices for the second chains. Neurons are sorted as in the first panel of Fig. 4D. B. The posterior similarity matrix sorted by MAP, for chain 1 when the mouse is exposed to drifting gratings as in Fig. 3. C. ARI of MAP estimates. The diagonal is ARI between 2 chains, and the off-diagonal is mean ARI of MAP for 4 combinations.
Algorithm 1: Pólya-Gamma-Metropolis-Hastings Algorithm (PG-MH) for Poisson Dynamic Model

Given the sample from the \((G-1)\)-th iteration \(\bar{x}^{(G-1)}_t\) and \(U = \{\bar{c}_i, \delta_i, A, b, Q\}\).

1. Sample \(\omega_{it}\) from PG distribution and calculate \(\bar{y}_{it}\), which follows \(\mathcal{N}(\bar{c}_i^T \bar{x}^{(G-1)}_t, \omega_{it}^{-1})\)

\[
\begin{align*}
\text{for } t = 1, \ldots, T & \text{ do} \\
\text{for } i = 1, \ldots, n & \text{ do} \\
& \text{sample } \omega_{it} \sim P_{\text{PG}}(r_{it} + y_{it}, \delta_i + \bar{c}_i^T \bar{x}^{(G-1)}_t - \log r_{it}) \\
& \kappa_{it} = (y_{it} - r_{it})/2 + \omega_{it}(\log r_{it} - \delta_i) \\
& \bar{y}_{it} = \omega_{it}^{-1} \kappa_{it} \\
& \end{align*}
\]

end

end

2. Forward-filtering-backward-sampling (FFBS) for \(\bar{X}\)

Denote \(\bar{y}_t = (\bar{y}_{1t}, \ldots, \bar{y}_{Nt})', \Omega_t = \text{Diag}(\omega_{1t}, \ldots, \omega_{Nt})\) and \(\bar{C} = (\bar{c}_1, \ldots, \bar{c}_N)'\)

\[
\begin{align*}
\text{for } t = 1, \ldots, T & \text{ do} \\
& m_{t|t-1} = \tilde{A}m_{t-1} + \tilde{b} \\
& V_{t|t-1} = AV_{t-1}A' + \tilde{Q} \\
& K_t = V_{t|t-1}C'(\bar{C}V_{t|t-1}C' + \Omega_t^{-1})^{-1} \\
& m_t = m_{t|t-1} + K_t(\bar{y}_{t} - \bar{C}m_{t|t-1}) \\
& V_t = (I - K_t\bar{C})V_{t|t-1} \\
& \text{end} \\
& \text{sample } \bar{x}^*_t \sim \mathcal{N}(m_T, V_T) \\
\text{for } t = T - 1, \ldots, 1 & \text{ do} \\
& J_t = V_tA'(AV_tA' + \tilde{Q})^{-1} \\
& m^*_t = m_t + J_t(x_{t+1} - \tilde{A}m_t - \tilde{b}) \\
& V^*_t = (I - J_t\tilde{A})V_t \\
& \text{end} \\
& \text{sample } x^*_t \sim \mathcal{N}(m^*_T, V^*_T) \\
\end{align*}
\]

3. Accept or reject the proposal \(\bar{X}^*\)

compute the acceptance ratio

\[
\zeta = \frac{\pi(\bar{X}^*|\{y_i\}_{i=1}^n, U)}{\pi(\bar{X}^{(G-1)}|\{y_i\}_{i=1}^n, U)} q(\bar{X}^{(G-1)}|\bar{X}^*, U) q(\bar{X}^*|\bar{X}^{(G-1)}, U)
\]

\[
\zeta = \frac{P(\{y_i\}_{i=1}^n|\bar{X}^*) \text{NB}(\{y_i\}_{i=1}^n|\bar{X}^{(G-1)}, R)}{P(\{y_i\}_{i=1}^n|\bar{X}^{(G-1)}) \text{NB}(\{y_i\}_{i=1}^n|\bar{X}^*, R)}
\]

where \(R = \{r_{it}\}\) is matrix for dispersion parameters for each neuron at all time points. \(P(\cdot)\) denotes the Poisson likelihood, and \(\text{NB}(\cdot)\) denotes the negative binomial likelihood. Accept the proposal \(\bar{X}^*\) with probability \(\min(1, \zeta)\).
Algorithm 2: Birth-death point process for $p_j$

In iteration $G$, set $t_{fa} = 0$ and $p_j = p_j^{(G-1)}$

repeat

- Compute $\xi_k(C) = \frac{M(C\setminus \{X_{j}^{(1)}\})\beta}{\delta}$
- Compute $\xi(C) = \sum_{k=1}^{p_j} = \xi_k(C)$
- Simulate $s \sim \text{Exp}(1/(\beta + \xi(C)))$ and set $t_{fa} = t_{fa} + s$
  if $\text{Bern}(\beta/(\beta + \xi(C))) = 1$ then
    Set $q = q + 1$
    Simulate $X_{s,q}^{(j)}$ from prior $H$
    Set $C = C \cup \{X_{s,q}^{(j)}\}$
  else
    Simulate $k' = \text{Mn}(\xi_1(C)/\xi(C), ..., \xi_q(C)/\xi(C))$
    Set $C = C \setminus \{X_{s,k'}^{(j)}\}$
    Set $q = q - 1$
  end

until $t_{fa} \geq \rho$