Clustering Neural Populations with a Poisson Dynamic Factor Model

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 discrete population may be unclear. Neurons differ in their anatomical location, but, also, in their cell types and response properties. When the neural activities are globally nonlinear, using single population analysis is inappropriate. However, defining the populations is usually difficult and wrong cluster assignments will lead to bias in latent structure inferences. To tackle this challenge, here we develop a clustering method based on a mixture of Poisson dynamic factor model with the number of cluster is treated as a parameter in mixture of finite mixtures (MFM) model. To sample efficiently from the posteriors, we approximate the full conditional distribution of latent state by Gaussian and approximate the marginal likelihood by making use of the Poisson-Gamma conjugacy. We apply our method to multi-region recordings, neuropixels recordings for illustration. This approach may be a useful tool for clustering neurons based on function.

1 Introduction

[TODO]

2 Method

2.1 Poisson Dynamic Factor Model

Denote the observed spike-count of neuron $i \in \{1, \dots, N\}$ at time bin $t \in \{1, \dots, T\}$ as $y_{it} \in \mathbb{Z}_{\geq 0}$, and let $\boldsymbol{y}_i = (y_{i1}, \dots, y_{iT})'$. Further, let $z_i = j$ denote the cluster indicator of neuron i. To facilitate clustering, we re-parametrize the regular Poisson linear dynamical system (PLDS) model [Macke et al., 2011] to separate the mean log-firing-rate out. Assume neural responses are independently Poisson distributed, conditional on the low-dimensional latent state $\boldsymbol{x}_t^{(j)} \in \mathbb{R}^p$ as follows:

$$y_{it} \sim Poi(\lambda_{it})$$
$$\log \lambda_{it} = \mu_t^{(j)} + \boldsymbol{c}_i' \boldsymbol{x}_t^{(j)}$$

Preprint. Under review.

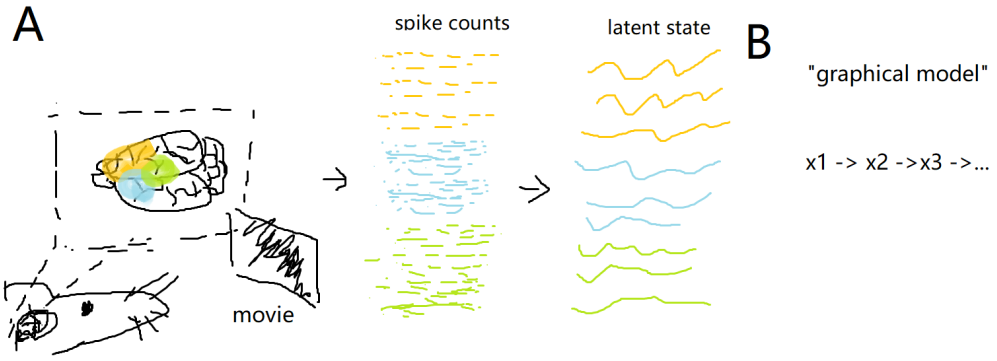

Figure 1: **Overview of the model. A.** Briefly describe the neuroscience motivation. **B.** The graphical model

, with $\boldsymbol{c}_i \sim N(\boldsymbol{0}, \boldsymbol{I}_p)$. We further assume the intercept $\mu_t^{(j)}$ and the latent state $\boldsymbol{x}_t^{(j)}$ evolve linearly over time with Gaussian noise as

$$\begin{aligned}
\mu_1^{(j)} &\sim N(\mu_0, \Sigma_0) \\
\mu_{t+1}^{(j)} &\sim N(f^{(j)}\mu_t^{(j)} + g^{(j)}, \sigma^{2(j)}) \\
\boldsymbol{x}_1^{(j)} &\sim N(\boldsymbol{x}_0, \boldsymbol{Q}_0) \\
\boldsymbol{x}_{t+1}^{(j)} &\sim N(\boldsymbol{A}^{(j)}\boldsymbol{x}_t^{(j)} + \boldsymbol{b}^{(j)}, \boldsymbol{Q}^{(j)})
\end{aligned}$$

The bias term $\boldsymbol{b}^{(j)}$ in the latent state is constant across time, unlike ind PLDS, which is $\boldsymbol{b}_t^{(j)}$. When the bias term changes across the time, the update order of $\boldsymbol{A}^{(j)}$ and $\boldsymbol{b}_t^{(j)}$ will influence the results (usually, the $\boldsymbol{A}^{(j)}$ is updated first). When ignoring the updating order, the linear dynamics will have infinite solutions. If the $\boldsymbol{A}^{(j)}$ is one solution, then $\boldsymbol{A}^{*(j)} = \boldsymbol{A}^{(j)} + \boldsymbol{M}$ and $\boldsymbol{b}_t^{*(j)} = \boldsymbol{b}_t^{(j)} - \boldsymbol{M}\boldsymbol{x}_t^{(j)}$ will be another set of solution, for any $\boldsymbol{M} \in \mathbb{R}^{p\times p}$. This causes some inconvenience for inference. The same reason for using the time-constant $g^{(j)}$.

If we denote $\boldsymbol{\lambda}_i = (\lambda_{i1}, \ldots, \lambda_{iT})'$, $\boldsymbol{\mu}^{(j)} = (\mu_1^{(j)}, \ldots, \mu_T^{(j)})'$ and $\boldsymbol{X}^{(j)} = (\boldsymbol{x}_1^{(j)}, \ldots, \boldsymbol{x}_T^{(j)})'$, the model can be equivalently written as regular Poisson factor model as

$$\begin{aligned}
\boldsymbol{y}_i &\sim Poi(\boldsymbol{\lambda}_i) \\
\log \boldsymbol{\lambda}_i &= \boldsymbol{\mu}^{(j)} + \boldsymbol{X}^{(j)}\boldsymbol{c}_i
\end{aligned}$$

However, since the condition $T/N \to 0$ doesn't hold [Johnstone and Lu, 2009], the latent state $\boldsymbol{X}^{(j)}$ cannot be consistently estimated, and assuming linear dynamics of $\boldsymbol{X}^{(j)}$ resolves the problem. Note that when $p > 1$, the model is not unique, since $\boldsymbol{X}^{*(j)} = \boldsymbol{X}^{(j)}\boldsymbol{U}$ also satisfies the equation for any orthogonal matrix $\boldsymbol{U}$ of order p. To ensure the model indefinability, we simply assume $\boldsymbol{A}^{(j)}$ and $\boldsymbol{Q}^{(j)}$ are both diagonal for convenience. See more detailed discussions of the constraints in discussion. Overall, denote the cluster-related parameters of cluster j as $\boldsymbol{\theta}^{(j)} = \{\boldsymbol{\mu}^{(j)}, \boldsymbol{X}^{(j)}, f^{(j)}, g^{(j)}, \sigma^{2(j)}, \boldsymbol{A}^{(j)}, \boldsymbol{b}^{(j)}, \boldsymbol{Q}^{(j)}\}$ and the spike counts of neuron i is generated by Poisson dynamic factor model (PDFM) as $\boldsymbol{Y}_i \sim PDFM(\boldsymbol{\theta}^{(z_i)})$, with the prior of $\boldsymbol{\theta}^{(j)}$ as $\boldsymbol{H}$. The priors for $\{f^{(j)}, g^{(j)}, \sigma^{2(j)}, \boldsymbol{A}^{(j)}, \boldsymbol{b}^{(j)}, \boldsymbol{Q}^{(j)}\}$ are regular normal and inverse-gamma distribution (or multivariate normal and inverse-Wishart when using other non-diagonal constraints).

The marginal likelihood of neuron i is

$$M_{\boldsymbol{\theta}^{(j)}}(\boldsymbol{y}_i) = P(\boldsymbol{y}_i|\boldsymbol{\theta}^{(j)}) = \int P(\boldsymbol{y}_i|\boldsymbol{\theta}^{(j)}, \boldsymbol{c}_i)P(\boldsymbol{c}_i)\, d\boldsymbol{c}_i$$

The marginal likelihood has no closed form and will be used for clustering. To help with fast clustering, instead of doing the Laplace approximation, we choose to make use of the Poisson-Gamma conjugacy. This approximation was originally used in El-Sayyad [1973] to derive approximate posterior and the same method was applied to derive other approximations in Chan and Vasconcelos [2009]. This approximation leads to the closed form approximation. By the conditional independency assumption, $M_{\boldsymbol{\theta}^{(j)}}(\boldsymbol{y}_i) = \prod_{t=1}^{T} P(y_{it}|\boldsymbol{\theta}^{(j)})$. Since $\boldsymbol{c}_i \sim N(\boldsymbol{0}, \boldsymbol{I}_p)$, $\lambda_{it} = \exp(\mu_t^{(j)} + \boldsymbol{c}_i' \boldsymbol{x}_t^{(j)}) \sim lognormal(\mu_t^{(j)}, \boldsymbol{x}_t'^{(j)} \boldsymbol{x}_t^{(j)})$. Approximate the lognormal distribution by Gamma distribution, s.t. $lognormal(\mu_t^{(j)}, \boldsymbol{x}_t'^{(j)} \boldsymbol{x}_t^{(j)}) \approx Gamma(a_{it}, b_{it})$ with $a_{it} = (\boldsymbol{x}_t'^{(j)} \boldsymbol{x}_t^{(j)})^{-1}$ and $b_{it} = \boldsymbol{x}_t'^{(j)} \boldsymbol{x}_t^{(j)} \cdot e^{\mu_t^{(j)}}$. Then by Poisson-Gamma conjugacy,

$$P(y_{it}|\boldsymbol{\theta}^{(j)}) = \int P(y_{it}|\lambda_{it}) P(\lambda_{it})\, d\lambda_{it} \approx NB(y_{it}|r_{it}, p_{it})$$

, with $r_{it} = a_{it}$ and $p_{it} = 1/(1 + b_{it})$.

Another more general idea is to approximate the log-likelihood by second-order polynomials, with coefficients determined by Chebyshev polynomial approximation Keeley et al. [2019]. However, the approximation doesn't work well in practice, especially when the neural spike counts have a wide range. When doing the integration, we need to exponentiate the log-likelihood and this will exaggerate the approximation error.

2.2 Cluster by Mixture of Finite Mixtures Model

When the label is unknown, we cluster the neurons by mixture models. In practice, it's usually impossible to know the number of neural population. One potential method is to do clustering by Dirichlet process mixtures (DPM) model. However, this is conceptually incorrect, since the number of neural populations is finite but unknown. Besides the conceptual incorrectness, using DPM is not easy to integrate the field knowledge about the number of neural populations. Here, we choose to use the mixture of finite mixtures (MFM, Miller and Harrison [2018]) model as follows

$$\begin{aligned}
K &\sim p_K && \text{where } p_K \text{ is a p.m.f. on} \{1, 2, \ldots\} \\
\boldsymbol{\pi} = (\pi_1, \ldots, \pi_k) &\sim Dir_k(\gamma, \ldots, \gamma) && \text{given} K = k \\
Z_1, \ldots, Z_n &\overset{i.i.d.}{\sim} \pi && \text{given} \pi \\
\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k &\overset{i.i.d.}{\sim} \boldsymbol{H} && \text{given} K = k \\
\boldsymbol{Y}_i = (y_{i1}, \ldots, y_{iT})' &\sim SSFM(\boldsymbol{\theta}^{(z_i)}) && \text{independently for } i = 1, \ldots, N, \text{ given } \boldsymbol{\theta}_{1:K} \text{ and } Z_{1:N}
\end{aligned}$$

Besides the conceptual correctness, using MFM model allows us integrate the prior knowledge easily. Moreover, compared to DPM, MFM has some better properties for clustering, for example, MFM posterior on number of cluster is more concentrated and consistent, and MFM tend to give clusters size at the same order of magnitude while DPM may lead to a few large clusters and many small clusters. See Miller and Harrison [2018] for detailed discussion.

2.3 Inference

In this paper, we choose to do inference by MCMC. Because of the Poisson likelihood, the latent state $\boldsymbol{X}^{(j)}$has no closed full conditional distribution. We can sample the posterior by particle MCMC directly, but this can be slow. However, due to the Markovian structure of the model, the conditional log-posterior is concave and its Hessian is block-tridiagonal. Thus, we can do the global Laplace approximation efficiently in $\mathcal{O}(T)$ [Paninski et al., 2010].The cluster index and number of cluster are sampled by the analog of partition-based algorithm in DPM [Neal, 2000]. See details of the MCMC update in the Appendix.

In practice, using variational Bayes (VB) instead of MCMC may be more favorable. The PLDS can be updated by variational EM. Using the stick-breaking representation of MFM model, we can do VB easily similar to Blei and Jordan [2006]. However, checking by the "gold standard" MCMC before doing VB is always a good choice.

For the dimensionality p of of the latent state, we can treat it as a parameter and sample the posterior by RJMCMC as in Lopes and West [2004] or borrow the idea of adaptive Gibbs sampling with shrinkage prior [Bhattacharya and Dunson, 2011]. Here, we simply pre-set the p or select the optimized value by the cross-validation, which can be easily conducted when switching to the deterministic algorithm in the future.

3 Simulations

3.1 Model Global Non-linearity by Clustering

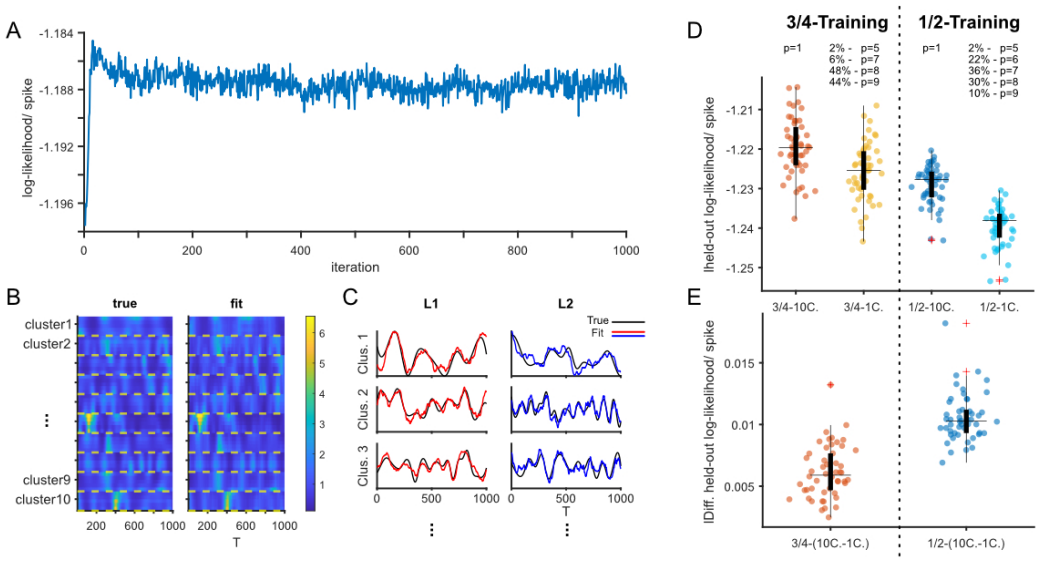

Figure 2: **Global non-linearity by multiple clusters** (The labels are known now. The purpose is to check MCMC do fit well.)Simulate 10 clusters without specifying linear dynamics, with 5 neurons each. The dimension in each cluster is $p = 1(q = 2)$. All the fitted posteriors are averages from iteration 500 to 1000. **A.** The trace plot of log-likelihood per spike. **B.** The fitted mean firing rate. **C.** The fitted latent vectors in the first 3 clusters.**D.** and **E.** Select 3/4 and 1/2 of the data as training in a speckled pattern, and replicate the sampling 50 times in each case. With each training sample, fit two models: 1) 10-cluster model: fit every single model for each, with $p = 1$ (true value) in each and 2) 1-cluster model: fit a single model for all neurons, with p selected by cross-validation. **D.** The held-out log-likelihood per spike decrease for less training samples. With the same training sample, 10-cluster model performs better than 1-cluster model. The selected p for 1-cluster model is shown at the top. **E.** The differences of the held-out log-likelihood per spike between 10-cluster model and 1-cluster model. The less the training data, the more significant the improvement by fitting model cluster-wise.

3.1.1 Clustering

4 Application: Neuropixels data

5 Discussion

As mentioned above, the factor model doesn't have unique solution. Although this issue had been thoroughly discussed in statistics [reference], it has been ignored in some neuroscience research. Since the (P)LDS related model are usually fitted by deterministic algorithms such as variants of EM, the issue is kind of hard to detect. And because of the ignorance, there are some problematic comments on linear dynamics $\boldsymbol{A}$ and $\boldsymbol{Q}$. When using the variants of factor model, e.g. (P)LDS and GPFA, we should only focus on the "shape" of latent state but not the specific values of them.

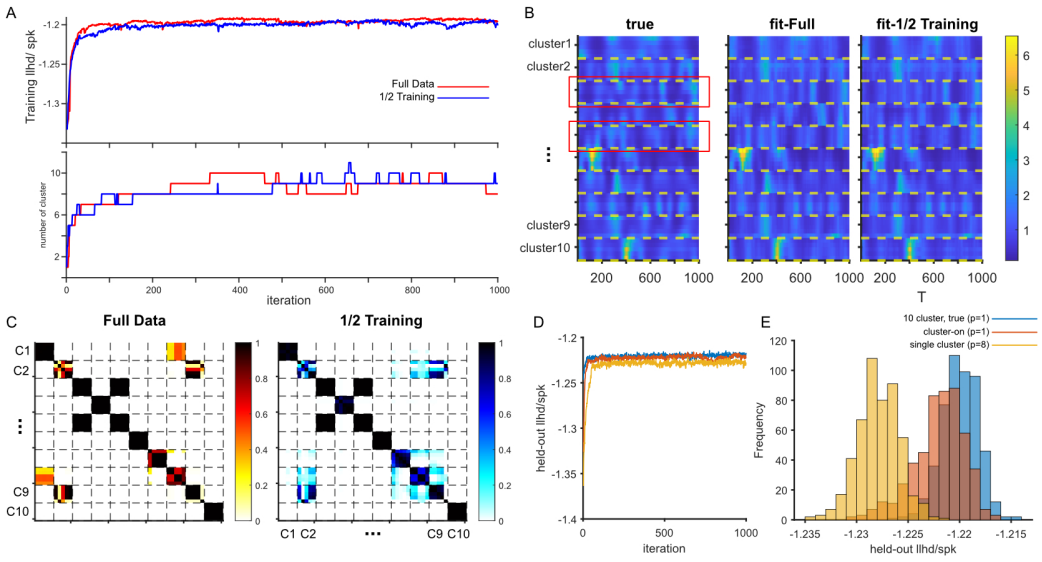

Figure 3: **Clustering** The same simulation setting as in Figure 2, but remove the labels. (The cluster 3 and 5 have nearly the same spiking patterns, because of the "bad luck"... May resample with another seed). The fitting is conducted using all and half data as training. **A.** The trace plot of training log-likelihood per spike and number of cluster. All the posterior results are averages from iteration 500 to 1000. **B.** The fitted firing rate. (This is different from figure 2, because the labels are removed) **C.** The similarity matrix. **D.** and **E.** Compare the held-out (1/2 data) likelihood for 3 fittings: 1) 10-cluster model: use the true labels and fit the model in each cluster with $p = 1$; 2) cluster-on model: unknown labels but turn on the clustering, with $p = 1$ (CV) and 3) 1-cluster model: single population model, with $p = 8$ (CV). **D.** The trace plot of held-out log-likelihood per spike. The traces for 10-cluster and cluster-on are close, but more variation (occasional "drop-down") in cluster-on due to estimation of unknown labels. **E.** The histograms of posterior held-out log-likelihood per spike for 3 models. 10-cluster and cluster-on models are close, and they perform better than 1-cluster model.

In this paper, to ensure the unique solution, we put the diagonal constraints in $\boldsymbol{A}^{(j)}$ and $\boldsymbol{Q}^{(j)}$ are used for convenience. However, only constraining on $p(p-1)/2$ s enough [reference]. Since we the label is unknown and will be switched when doing clustering, it's inappropriate and convenient to put constraint on $\boldsymbol{c}_i$. Therefore, we instead treat $\boldsymbol{X}^{(j)}$ as the "loading" and can put constraints on it. In traditional Gaussian factor model, there are two equivalent types of constraints: (1) diagonal constraint: $\boldsymbol{X}'^{(j)}\boldsymbol{X}^{(j)}$ is diagonal; (2) block lower triangular constraint [Fokoué and Titterington, 2003]: The first p rows of $\boldsymbol{X}^{(j)}$ is lower triagular and the diagonal elements are positive. Since we assume $\boldsymbol{X}^{(j)}$ has linear progression, using the diagonal $\boldsymbol{X}'^{(j)}\boldsymbol{X}^{(j)}$ constraint is more appropriate. This constraint can be easily achieved when conducting deterministic optimization algorithms (e.g. the VB mentioned in "method-inference" section). This can also be done in MCMC, by turning off the reflection of the projection (e.g. let the diagonal elements in the projection matrix be positive). Using diagonal $\boldsymbol{X}'^{(j)}\boldsymbol{X}^{(j)}$ leads to similar results.

Moreover, the idea of doing clustering by mixture model can be extended beyond the Poisson distribution. We can further include other information, such as, the dispersion information by assuming negative-binomial distributed or even Conway-Maxwell-Poisson distributed neural spikes [reference to our paper in CMP]. This further information can be used for more detailed clustering, by expanding the state space.

Acknowledgments

Broader Impact

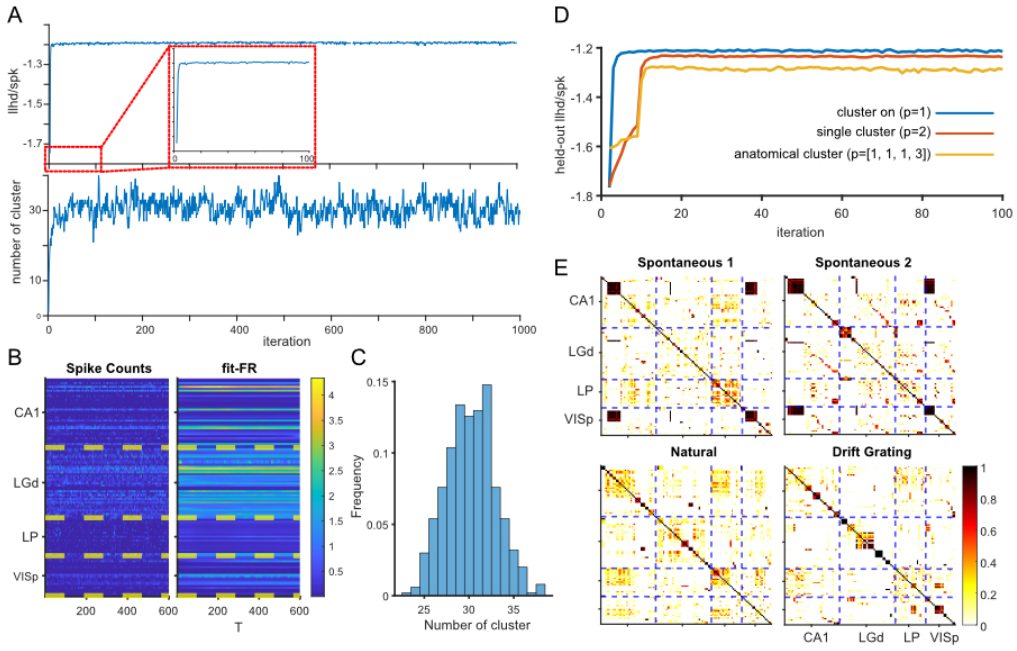

Figure 4: **Application in Neuropixels data.** We first choose a one minute recordings from spontaneous, with bin size in $0.1s$ (29.8 to 89.8).**A.** The trace plots of log-likelihood per spike and number of cluster (Maybe again overlaid by 1/2-training results?). All the following results are averages from iteration 500 to 1000. **B** The fitted firing rate. **C.** The histogram of posterior cluster number. **D.** We then hold out 1/2 of data in the speckled pattern and fit 3 models: 1) cluster-on model: turn on clustering with $p = 1$ (CV); 2) 1-cluster: single population with $p = 2$ (CV) and 3) anatomy-cluster: fit models each based on 4 anatomical labels, with $p = (1, 1, 1, 3)$ (CV). Doing clustering is the best. Anatomical labels give the "wrong" labels, in terms of the spiking pattern, which make things even worse. **E.** The similarity matrices when fitting four 1-min epochs: 1) spontaneous 1: 29.8s to 89.8s; 2) spontaneous 2: 1001.9s to 1061.9s; 3) natural movie: 2221.7s to 2281.7s and 4) drift grating: 1890.9s to 1950.9s. All are sorted by mode of posterior cluster index within each anatomical region. The patterns found in spontaneous are robust and there are a large confusion between VISp and CA1. The confusion between VISp and CA1 also exists in natural movie, but decrease a lot. However, the overall confusions become larger. When exposing to the drift grating movie, the confusions among all four regions decrease even more.

References

Jakob H. Macke, Lars Buesing, John P. Cunningham, Byron M. Yu, Krishna V. Shenoy, and Maneesh Sahani. Empirical models of spiking in neural populations. *Advances in Neural Information Processing Systems*, 24, 2011.

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

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

Antoni B. Chan and Nuno 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.

Stephen L. Keeley, David M. Zoltowski, Yiyi Yu, Jacob L. Yates, Spencer L. Smith, and Jonathan 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, jun 2019.

Jeffrey W. Miller and Matthew T. Harrison. Mixture models with a prior on the number of components. *Journal of the American Statistical Association*, 113(521):340, jan 2018. doi: 10.1080/01621459.2016.1255636.

Liam Paninski, Yashar Ahmadian, Daniel Gil Ferreira, Shinsuke Koyama, Kamiar Rahnama Rad, Michael Vidne, Joshua Vogelstein, and Wei Wu. A new look at state-space models for neural data, aug 2010. ISSN 09295313. URL https://link.springer.com/article/10.1007/s10827-009-0179-x.

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

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

Hedibert Freitas Lopes and Mike West. Bayesian Model Assessment in Factor Analysis. *Statistica Sinica*, 14: 41–67, 2004.

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

Alex H. Williams, Anthony Degleris, Yixin Wang, and Scott W. Linderman. Point process models for sequence detection in high-dimensional neural spike trains. *Advances in Neural Information Processing Systems*, 2020-Decem, oct 2020. ISSN 10495258.

Ernest Fokoué and D. M. Titterington. Mixtures of factor analysers. Bayesian estimation and inference by stochastic simulation. *Machine Learning*, 50(1-2):73–94, jan 2003. ISSN 08856125. doi: 10.1023/A:1020297828025.

Uri T Eden, Loren M. Frank, Riccardo Barbieri, Victor Solo, and Emery N. Brown. Dynamic Analysis of Neural Encoding by Point Process Adaptive Filtering. *Neural Computation*, 16(5):971–998, may 2004. ISSN 0899-7667. doi: 10.1162/089976604773135069. URL https://doi.org/10.1162/089976604773135069.

H E Rauch, F Tung, and C T Striebel. Maximum likelihood estimates of linear dynamic systems. *AIAA Journal*, 3(8):1445–1450, aug 1965. ISSN 0001-1452. doi: 10.2514/3.3166. URL https://doi.org/10.2514/3.3166.

Matthew D. Hoffman and Andrew Gelman. The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, 15:1593–1623, nov 2011. ISSN 15337928. URL https://arxiv.org/abs/1111.4246v1.

A Appendix

A.1 MCMC updates

The posteriors are sampled by a Gibbs sampler, the full conditional distributions, if available, are shown as follows:

Update $x_t^{(j)}$ and $\mu_t^{(j)}$: Let the t^{th} column observation and firing rate of the j^{th} cluster be $\widetilde{\boldsymbol{y}}_t = vec\left(\{y_{it}^{(j)}|z_i = j\}\right)$ and $\widetilde{\boldsymbol{\lambda}}_t^{(j)} = vec\left(\{\lambda_{it}|z_i = j\}\right)$. The number of neurons in cluster j is $n_j = |\{i : z_i = j\}|$. The corresponding loading for these n_j neurons is $\boldsymbol{C}^{(j)} \in \mathbb{R}^{n_j \times p}$, such that $\log \widetilde{\boldsymbol{\lambda}}_t^{(j)} = \mu_t^{(j)} \mathbf{1}_{n_j} + \boldsymbol{C}^{(j)} \boldsymbol{x}_t^{(j)} = \left(\mathbf{1}_{n_j}, \boldsymbol{C}^{(j)}\right) \cdot \left(\mu_t^{(j)}, \boldsymbol{x}_t^{'(j)}\right)'$. Denote $\widetilde{\boldsymbol{C}}^{(j)} = \left(\mathbf{1}_{n_j}, \boldsymbol{C}^{(j)}\right)$, $\widetilde{\boldsymbol{x}}_t^{(j)} = \left(\mu_t^{(j)}, \boldsymbol{x}_t^{'(j)}\right)'$, $\widetilde{\boldsymbol{x}}^{(j)} = \left(\widetilde{\boldsymbol{x}}_1^{'(j)}, \ldots, \widetilde{\boldsymbol{x}}_T^{'(j)}\right)'$, $\widetilde{\boldsymbol{A}}^{(j)} = diag(f^{(j)}, \boldsymbol{A}^{(j)})$, $\widetilde{\boldsymbol{b}}^{(j)} = \left(g^{(j)}, \boldsymbol{b}^{'(j)}\right)'$ and $\widetilde{\boldsymbol{Q}}^{(j)} = diag(\sigma^{2(j)}, \boldsymbol{Q}^{(j)})$. The full conditional distribution $P(\widetilde{\boldsymbol{x}}^{(j)}|\ldots) = P(\widetilde{\boldsymbol{x}}^{(j)}|\{\widetilde{\boldsymbol{y}}_t\}_{t=1}^T, \widetilde{\boldsymbol{C}}^{(j)}, \widetilde{\boldsymbol{A}}^{(j)}, \widetilde{\boldsymbol{b}}^{(j)}, \widetilde{\boldsymbol{Q}}^{(j)})$ is approximated by a global Laplace approximation, i.e. $P(\widetilde{\boldsymbol{x}}^{(j)}|\ldots) \approx N_{(p+1)T}(\widetilde{\boldsymbol{x}}^{(j)}|\boldsymbol{\mu}_{\widetilde{\boldsymbol{x}}^{(j)}}, \boldsymbol{\Sigma}_{\widetilde{\boldsymbol{x}}^{(j)}})$, with $\boldsymbol{\mu}_{\widetilde{\boldsymbol{x}}^{(j)}} = \arg\max_{\widetilde{\boldsymbol{x}}^{(j)}} P(\widetilde{\boldsymbol{x}}^{(j)}|\ldots)$ and $\boldsymbol{\Sigma}_{\widetilde{\boldsymbol{x}}^{(j)}}) = -\left(\nabla\nabla \log P(\widetilde{\boldsymbol{x}}^{(j)}|\ldots)\big|_{\widetilde{\boldsymbol{x}}^{(j)}=\boldsymbol{\mu}_{\widetilde{\boldsymbol{x}}^{(j)}}}\right)^{-1}$. The log full conditional distribution $h(\widetilde{\boldsymbol{x}}^{(j)}) = \log P(\widetilde{\boldsymbol{x}}^{(j)}|\ldots)$ is given by:

$$h = \text{const} + \sum_{t=1}^{T} \left(\widetilde{\boldsymbol{y}}_t' \widetilde{\boldsymbol{C}}^{(j)} \widetilde{\boldsymbol{x}}_t^{(j)} - \widetilde{\boldsymbol{\lambda}}_t\right) - \frac{1}{2}(\widetilde{\boldsymbol{x}}_1^{(j)} - \widetilde{\boldsymbol{x}}_0^{(j)})' \widetilde{\boldsymbol{Q}}_0^{(j)} (\widetilde{\boldsymbol{x}}_1^{(j)} - \widetilde{\boldsymbol{x}}_0^{(j)})$$

$$- \sum_{t=2}^{T} \frac{1}{2}(\widetilde{\boldsymbol{x}}_t^{(j)} - \widetilde{\boldsymbol{A}}^{(j)} \widetilde{\boldsymbol{x}}_{t-1}^{(j)} - \widetilde{\boldsymbol{b}}^{(j)})' \widetilde{\boldsymbol{Q}}^{(j)} (\widetilde{\boldsymbol{x}}_t^{(j)} - \widetilde{\boldsymbol{A}}^{(j)} \widetilde{\boldsymbol{x}}_{t-1}^{(j)} - \widetilde{\boldsymbol{b}}^{(j)})$$

, where "const" represents the constant. The $h(\widetilde{\boldsymbol{x}}^{(j)})$ is concave and hence unimodal, and the Markovian structure of the latent dynamics, and hence the tri-block-diagonal Hessian, makes it possible to compute a Newton update in $\mathcal{O}(T)$ [Paninski et al., 2010]. The same technique is used in the E-step for PLDS model [Macke et al., 2011].

To facilitate convergence, we use a smoothing estimate with local Gaussian approximation as a "warm start". The forward filtering for a dynamic Poisson model has been previously described in Eden et al. [2004]. Because the approximated filtering estimates are Gaussian distributed, we can further find the smoothing estimates using a backward pass as in Rauch et al. [1965].

Update $\boldsymbol{c}_i$: When writing the observation mapping in matrix form, $\log \boldsymbol{\lambda}_i = \boldsymbol{\mu}^{(j)} + \boldsymbol{X}^{(j)}\boldsymbol{c}_i$, the update of $\boldsymbol{c}_i$ reduces to a regular Poisson regression problem, given $\boldsymbol{\mu}^{(j)}$ and $\boldsymbol{X}^{(j)}$ are known. Here, we sample the posterior by no a No U-Turn Sampler (NUTS, Hoffman and Gelman [2011]), within the Gibbs sampler.

Update Linear dynamics of latent state: The parameters for linear dynamics are $f^{(j)}$, $g^{(j)}$, $\sigma^{2(j)}$, $\boldsymbol{A}^{(j)}$, $\boldsymbol{b}^{(j)}$ and $\boldsymbol{Q}^{(j)}$. To make the model identifiable, we simply assume $\boldsymbol{A}^{(j)} = diag(a_1^{(j)}, \ldots, a_p^{(j)})$ and $\boldsymbol{Q}^{(j)} = diag(q_1^{(j)}, \ldots, q_p^{(j)})$. Therefore, we can update $\boldsymbol{A}^{(j)}$, $\boldsymbol{b}^{(j)}$ and $\boldsymbol{Q}^{(j)}$ dimension-by-dimension, as the update in $f^{(j)}$, $g^{(j)}$ and $\sigma^{2(j)}$. Use the priors $\sigma^{2(j)} \sim IG\left(\nu_0/2, \nu_0\sigma_0^2/2\right)$ and $\left(g^{(j)}, f^{(j)}\right)' \sim N(\boldsymbol{\mu}_0, \sigma^{2(j)}\boldsymbol{\Lambda}_0^{-1})$. The prior μ_0 is specified as $(0,1)'$. Denote $\boldsymbol{y}_{\mu^{(j)}} = \left(\mu_2^{(j)}, \ldots, \mu_T^{(j)}\right)'$ and $\boldsymbol{X}_{\mu^{(j)}} = \left(\boldsymbol{1}_{T-1}, \boldsymbol{\mu}_{1:(T-1)}^{(j)}\right)$, with $\boldsymbol{\mu}_{1:(T-1)}^{(j)} = \left(\mu_1^{(j)}, \ldots, \mu_{T-1}^{(j)}\right)'$. The full conditional distributions are $\sigma^{2(j)}|\ldots \sim IG\left(\frac{\nu_0+T-1}{2}, \frac{\nu_0\sigma_0^2 + \boldsymbol{y}'_{\mu^{(j)}}\boldsymbol{y}_{\mu^{(j)}} + \boldsymbol{\mu}_0'\boldsymbol{\Lambda}_0\boldsymbol{\mu}_0 - \boldsymbol{\mu}_n'\boldsymbol{\Lambda}_n\boldsymbol{\mu}_n}{2}\right)$ and $\left(g^{(j)}, f^{(j)}\right)'|\ldots \sim N(\boldsymbol{\mu}_n, \sigma^{2(j)}\boldsymbol{\Lambda}_n^{-1})$, with $\boldsymbol{\Lambda}_n = \boldsymbol{X}'_{\mu^{(j)}}\boldsymbol{X}_{\mu^{(j)}} + \boldsymbol{\Lambda}_0$ and $\boldsymbol{\mu}_n = \boldsymbol{\Lambda}_n^{-1}\left(\boldsymbol{X}'_{\mu^{(j)}}\boldsymbol{X}_{\mu^{(j)}}\left(\boldsymbol{X}'_{\mu^{(j)}}\boldsymbol{X}_{\mu^{(j)}}\right)^{-1}\boldsymbol{X}'_{\mu^{(j)}}\boldsymbol{y}_{\mu^{(j)}} + \boldsymbol{\Lambda}_0\boldsymbol{\mu}_0\right)$. The updates of $\boldsymbol{A}^{(j)}$, $\boldsymbol{b}^{(j)}$ and $\boldsymbol{Q}^{(j)}$ are similar, with diagonal assumption.

Update z_i: To update the cluster assignments for each neuron i, we modify a partition based algorithm in DPM for MFM. Because of the non-conjugacy, the marginal likelihood in terms of all cluster parameters θ_{z_i} can not be easily computed. This issue can be solved by introducing auxiliary variable, as in "Algorithm 8" in Neal [2000] for DPM. We can use the same idea for inference in the MFM by two substitutions: 1) replace $|c_i|$ by $|c_i| + \gamma$ and 2) replace α by $\gamma V_n(t+1)/V_n(t)$. Here, t is the number of partition obtained by removing the neuron i. The $V_n(t) = \sum_{k=1}^{\infty} \frac{k_{(t)}}{(\gamma k)^{(n)}} p_K(k)$, with $x^{(m)} = x(x+1)\cdots(x+m-1)$, $x_{(m)} = x(x-1)\cdots(x-m+1)$, $x^{(0)} = 1$ and $x_{(0)} = 1$. See details in Miller and Harrison [2018]