Conditionally-Conjugate Gaussian Process Factor Analysis for Spike Count Data via Data Augmentation

Yididiya Y. Nadew [1] **Xuhui Fan** [2] **Christopher J. Quinn** [1]

Abstract

Gaussian process factor analysis (GPFA) is a latent variable modeling technique commonly used to identify smooth, low-dimensional latent trajectories underlying high-dimensional neural recordings. Specifically, researchers model spiking rates as Gaussian observations, resulting in tractable inference. Recently, GPFA has been extended to model spike count data. However, due to the non-conjugacy of the likelihood, the inference becomes intractable. Prior works rely on either black-box inference techniques, numerical integration or polynomial approximations of the likelihood to handle intractability. To overcome this challenge, we propose a conditionally-conjugate Gaussian process factor analysis (ccGPFA) resulting in both analytically and computationally tractable inference for modeling neural activity from spike count data. In particular, we develop a novel data augmentation based method that renders the model conditionally conjugate. Consequently, our model enjoys the advantage of simple closed-form updates using a variational EM algorithm. Furthermore, due to its conditional conjugacy, we show our model can be readily scaled using sparse Gaussian Processes and accelerated inference via natural gradients. To validate our method, we empirically demonstrate its efficacy through experiments.

1. Introduction

In neuroscience, recent advances in recording techniques have made large-scale neural recordings ubiquitous. Common techniques such as neural probes enable simultaneous recording from activity of population of neurons (Steinmetz et al., 2021). Analyzing such high-dimensional data is a challenging statistical problem. A recent line of works adopts a generative view using latent variable models (LVMs). These methods assume the activity of a neural population lies in a low-dimensional subspace and thus can be captured with a small number of latent variables. Such low-dimensional representations can provide insight through encoding internal neural activity (Yu et al., 2008) and also capture relevant information to decode external behaviour such as motor activities (Glaser et al., 2020).

Generally, these models range from classical techniques such as principal component analysis (PCA) and factor analysis (FA) to more advanced methods such as linear dynamical systems (LDS) (Semedo et al., 2014; Gao et al., 2015) and Gaussian process (GP) based models (Yu et al., 2008). Unlike classical methods, GP-based models assume the latent variables follow a smooth temporal structure. Extending the idea of factor analysis to time series, Yu et al. (2008) proposed a novel method called Gaussian process factor analysis (GPFA). GPFA couples dimensionality reduction and temporal smoothness of Gaussian processes in a probabilistic model. The observations are modeled as conditionally Gaussian, leading to tractable updates within expectation-maximization (EM) based inference.

Building on (Yu et al., 2008) for Gaussian observations, researchers have proposed GPFA variants for count observation models, such as for Poisson distributions or negative binomial distributions, and developed corresponding inference methods, to more accurately model spike count data (Keeley et al., 2020a;b; Jensen et al., 2021). While models with discrete distributions of the observations can provide a better fit than Gaussian observation models, they are non-conjugate, which makes Bayesian inference intractable.

To deal with this non-conjugacy, Fourier-domain black-box variational inference (BBVI) (Keeley et al., 2020a) and numerical integration methods (Jensen et al., 2021) have been proposed, though such approaches have the potential to result in unstable and inaccurate approximations. Furthermore, such methods may require complicated settings such as learning rate tuning to ensure convergence and annealing techniques to effectively balance model exploration and

[1]Department of Computer Science, Iowa State University, Ames, IA, USA [2]School of Computing, Macquarie University, Sydney, NSW, Australia. Correspondence to: Christopher J. Quinn <cjquinn@iastate.edu>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

model selection. Keeley et al. (2020b) leveraged polynomial approximate log-likelihood (PAL) to obtain a marginal likelihood. In particular, this method approximates the non-linear terms in the likelihood using polynomial functions. While this approach yields conjugacy over the latent process, it is not a fully Bayesian scheme since it assumes the factor loading values as parameters and not random variables, making it prone to over fitting. Furthermore, in order to get the polynomial coefficients, the method performs least square solutions over the entire data, posing scalability issues. In this work, we present a conditionally-conjugate Gaussian process factor analysis (ccGPFA) model that can form conjugate models for spike count data. It involves augmenting set of auxiliary variables that render the model conditionally conjugate. In contrast to commonly applied Pólya-gamma augmentation (Pillow & Scott, 2012; Klami, 2015; Soulat et al., 2021), our method does not require that some variables be fixed or learned as model parameters. Soulat et al. (2021) attempted such augmentation for a latent variable model, but relies on moment matching approximations to learn a parameter that controls dispersion of spike counts. As demonstrated by the authors, despite its effectiveness for over-dispersed data, it poses issues for under-dispersed data. Our approach allows for a conjugate inference for all of the model variables, including the negative binomial dispersion parameter. We employ an efficient variational expectation-maximization (EM) algorithm to derive simple closed-form updates for the model. In short, our contributions are summarized as follows

- We implement a data augmentation technique to make GPFA models conditionally conjugate for spike count data.
- Leveraging the conditional conjugacy, we develop efficient coordinate ascent inference updates where the posterior of all variables, including the dispersion parameters, are available in closed form.
- To make inference computationally efficient, we extend the model and inference method, incorporating sparse Gaussian process priors and accelerating inference via natural gradients.
- Lastly, we demonstrate the efficiency and efficacy of our model in experiments.

Related works

The closest related works are (Keeley et al., 2020a;b; Jensen et al., 2021), which we highlighted in the introduction. We briefly expand on related works here. See Table 1 for a summary of key properties and see Appendix A for a longer discussion.

The standard GPFA model Yu et al. (2008) assumes Gaussian likelihood of the observations. Due to the nature of spike count data, most subsequent works extend the GPFA model to handle non-conjugate likelihoods such as Poisson and negative binomial.

Keeley et al. (2020a) employ techniques such as black-box variational inference (BBVI) to handle non-conjugacy. Despite their flexibility, BBVIs do not exploit the structure of model, relying on high variance Monte Carlo estimates. In addition they are sensitive to the choice of hyperparameters (Locatello et al., 2018).

Keeley et al. (2020b) follow an alternative approach using a polynomial approximation of the non-linear terms in the likelihood. This transforms the likelihood into a quadratic form which makes marginalization of variables easier. While this approach yields conjugacy over the latent process, it is not a fully Bayesian scheme since it treats the factor loading values as parameters and not random variables, making it prone to overfitting. Furthermore, the method computes least square solutions over the entire data to obtain polynomial coefficients and applies expensive second-order optimization posing scalability issues.

Dowling et al. (2023) combined the Hida-Matern Kernels and conjuage computational variational inference to develop latent GP models for neural spikes. However, their method is unable to model the under/over-dispersed spike count data. Specifically, they propose non-conjugate inference for Poisson count models.

Lastly, we note that one of the challenges in latent variable inference is selecting the number of latent variables. Jensen et al. (2021) employ an *automatic relevance determination* (ARD) prior to select the number of latent dimensions in a principled way. Gokcen et al. (2023) recently proposed an extension of standard GPFA (Yu et al., 2008) with ARD for modeling activity from multiple areas.

2. GPFA Model

In this section, we formally introduce our conditionally-conjugate Gaussian Process Factor Analysis (ccGPFA) model for spike count data.

2.1. Negative Binomial Modeling for Spike Counts

We consider the problem of modeling non-negative spike count data. Let $\boldsymbol{Y} \in \mathbb{N}^{N \times T}$ represent the spike counts of N simultaneously recorded neurons over an interval partitioned into T time steps (bins). Let $y_{n,t}$ denote the count for neuron n at time step t.

We model spike counts with negative binomial distributions (later we will naturally extend our work to the binomial distribution). While Poisson distributions are easier to work with analytically, since the mean and variance are equal they poorly model over (or under) dispersed data. Conceptually,

Table 1: **Property comparison with relevant GPFA variants**. Cnt – models count data ($\sim$ indicates only Poisson count model); ARD – automatic relevance determination to select the number of latents; CF – closed form updates ($\sim$ indicates only some model parameters have closed form); Scl – computationally scalable; Trl – repeated trials ($\sim$ indicates implementations can process multiple trials are not designed for repeated trials; see Appendix B.3). † inference is specialized for approximating the RBF kernel. ‡ is designed for multi-area.

Work	Cnt	ARD	CF	Scl	Trl
Yu et al. (2008)			$\sim$		$\sim$
Keeley et al. (2020a)	$\sim$				✓
Keeley et al. (2020b)	✓				✓
Jensen et al. (2021) †	✓	✓		✓	$\sim$
Gokcen et al. (2023) ‡		✓	✓		✓
Ours	✓	✓	✓	✓	✓

the negative binomial distribution models the number of successes in repeated i.i.d. binomial trials before a specified number of failures occur. For the negative binomial distribution, let $\hat{p}_{n,t} \in [0,1]$ denote the success probability for neuron n at time step (bin) t. Let r_n model the specified number of failures for neuron n. We refer to r_n as the dispersion parameter since their value can be tuned to account for the ratio of the variance to the mean. Let $\boldsymbol{r} = \{r_n\}_{n=1}^N$ denote the set of dispersion parameters for all neurons.

Further, we model the neural spiking activity as arising from a linear combination of *latent processes* $\boldsymbol{f} = \boldsymbol{W}\boldsymbol{X} + \boldsymbol{\beta}\boldsymbol{1}^\top \in \mathbb{R}^{N\times T}$, where $\boldsymbol{W} \in \mathbb{R}^{N\times D} (D \ll N)$ is a loading matrix as the combination coefficients, and $\boldsymbol{X} \in \mathbb{R}^{D\times T}$ represent D-dimensional independent latent processes for T time steps, and $\boldsymbol{\beta}$ is a bias term that represents the base spiking rates of neurons. Assuming the success probability $\hat{p}_{n,t}$ is a logistic transformation of $f_{n,t}$, i.e., $\hat{p}_{n,t} = \frac{e^{f_{n,t}}}{1+e^{f_{n,t}}}$ and conditioning on $\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{\beta}$, the neural count data $\boldsymbol{Y}$ are independently generated across neurons and time and their joint distribution can be factorized as:

$$p(\boldsymbol{Y}|\boldsymbol{W},\boldsymbol{X},\boldsymbol{\beta},\boldsymbol{r}) = \prod_{n,t} \text{NegBin}(y_{n,t}; r_n, \hat{p}_{n,t}) \quad (1)$$

$$= \prod_{n,t} \frac{\Gamma(y_{n,t}+r_n)[e^{f_{n,t}}]^{y_{n,t}}}{y_{n,t}!\Gamma(r_n)[1+e^{f_{n,t}}]^{y_{n,t}+r_n}}. \quad (2)$$

In Eq. (2)'s likelihood, it is difficult to find conjugate priors for the following variables since:

- the dispersion parameter r_n appears in two Gamma functions.
- the variables $\boldsymbol{X}, \boldsymbol{W}$, and $\boldsymbol{\beta}$ appear in both the denominator's and the numerator's exponential terms (through $f_{n,t} = [\boldsymbol{W}\boldsymbol{X} + \boldsymbol{\beta}\boldsymbol{1}^\top]_{n,t}$).

2.2. Data Augmentation

In this subsection, we show that by augmenting a set of auxiliary variables, the likelihood in Eq. (2) can be made conditionally conjugate. As a result, we can develop an efficient, fully-Bayesian inference procedure for all variables, including $\{r_n\}_{n=1}^N$.

Augmentation for $\{r_n\}_{n=1}^N$ Due to the complex form of the gamma functions, it is hard to find a conjugate prior for r_n. However, using the following integral representations, identified in (He et al., 2019), we can transform the gamma function and reciprocal of the gamma function as follows

$$\Gamma(y_{n,t}+r_n) \propto \int_0^\infty \tau_{n,t}^{(y_{n,t}+r_n)} e^{-\tau_{n,t}} d\tau_{n,t} \quad (3)$$

$$\frac{1}{\Gamma(r_n)} = r_n e^{\gamma r_n} \int_0^\infty e^{-r_n^2 \xi_{n,t}} \text{P-IG}(\xi_{n,t}|0) d\xi_{n,t}, \quad (4)$$

where Eq. (3) represents the marginalization of a gamma variable $\tau_{n,t} \sim \Gamma(y_{n,t}+r_n, 1)$ and Eq. (4) is the convolution of a Pólya-inverse gamma (P-IG) density. Here, $\gamma \approx 0.577$ denotes Euler's constant. These representations are equivalent to augmenting the variables $\tau_{n,t}$ and $\xi_{n,t}$ into the likelihood. See Appendix F.1 for more details about the P-IG distribution. As will be shown shortly (see (8)), this yields conjugacy with respect to r_n.

Augmentation for $\boldsymbol{X}, \boldsymbol{W}$ and $\boldsymbol{\beta}$ Inspired by (Polson et al., 2013), we also augment Pólya-gamma variables $\{\omega_{n,t}\}_{n=1,t=1}^{N,T}$ into Eq. (2) and obtain a joint distribution

$$p(y_{n,t}, \omega_{n,t}|\boldsymbol{W},\boldsymbol{X},\boldsymbol{\beta},r_n) = \frac{\Gamma(y_{n,t}+r_n)}{y_{n,t}!\Gamma(r_n)} 2^{-(y_{n,t}+r_n)}$$
$$\cdot \exp\left(\left(\frac{y_{n,t}-r_n}{2}\right) f_{n,t} - \frac{\omega_{n,t} f_{n,t}^2}{2}\right) \text{PG}(\omega_{n,t}|y_{n,t}+r_n, 0), \quad (5)$$

where $\text{PG}(\omega_{n,t}|y_{n,t}+r_n, 0)$ denotes the Pólya-gamma distribution (Polson et al., 2013) with shape and tilting parameters $y_{n,t}+r_n$ and 0 respectively.

Therefore upon conditioning on the augmented variables $\{\omega_{n,t}, \tau_{n,t}, \xi_{n,t}\}$, dropping factors that are constant with respect to the conditioning variables $\boldsymbol{W}$, $\boldsymbol{X}$, and $\boldsymbol{\beta}$, and using notation $z_{n,t} = \frac{y_{n,t}-r_n}{2\omega_{n,t}}$, the likelihood becomes

$$p(y_{n,t}|\boldsymbol{W},\boldsymbol{X},\boldsymbol{\beta},\omega_{n,t},\tau_{n,t},\xi_{n,t},r_n)$$
$$\propto \mathcal{N}(z_{n,t}|f_{n,t},\omega_{n,t}^{-1}). \quad (6)$$

Notice that this likelihood is proportional to the probability density function (pdf) of a corresponding Gaussian variable $z_{n,t}$ with mean $f_{n,t}$ and variance $\omega_{n,t}^{-1}$. This implies,

the likelihood is now conditionally conjugate to a Gaussian prior on $\boldsymbol{W}, \boldsymbol{X}$ and $\boldsymbol{\beta}$. Generalizing the above derivation for all time steps, and writing $\boldsymbol{f}_n = \{f_{n,t}\}_{t=1}^T, \boldsymbol{\Omega}_n = \text{diag}(\{\omega_{n,t}\}_{t=1}^T), \boldsymbol{z}_n = (\{z_{n,t}\}_{t=1}^T)$, we get a multivariate Gaussian distribution with diagonal covariance (equivalently factorizing into a product of marginal distributions),

$$p(\boldsymbol{Y}_n|\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{\beta}, \{\omega_{n,t}, \tau_{n,t}, \xi_{n,t}\}_{t=1}^T, r_n) \propto \mathcal{N}(\boldsymbol{z}_n^\top|\boldsymbol{f}_n^\top, \boldsymbol{\Omega}_n^{-1}). \quad (7)$$

Furthermore, using Eq. (4), the likelihood $p(\boldsymbol{Y}_n|\cdot)$ can be simplified as a function of its dispersion variable r_n,

$$p(\boldsymbol{Y}_n|\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{\beta}, \{\omega_{n,t}, \tau_{n,t}, \xi_{n,t}\}_{t=1}^T, r_n) \propto r_n^T e^{r_n \sum_t (\log \tau_{n,t} + \gamma - \log 2 - \frac{1}{2} f_{n,t}) - r_n^2 \sum_t \xi_{n,t}}. \quad (8)$$

Following (He et al., 2019), we identify the above expression as an un-normalized density of the Power-Truncated-Normal (PTN) distribution. The authors show that the gamma distribution can be a conjugate prior for the above likelihood expression. Importantly, we can efficiently estimate the mean of a PTN distribution which is crucial for our inference method. See Appendix F.2 for more details about this distribution.

This result will be important in deriving the closed form updates for our variational distribution. Detailed steps of the augmentation is included in Appendix B.

2.3. Priors

In this subsection, we show that the following choices for prior distributions are indeed (conditionally) conjugate and ease forthcoming inference updates.

Prior for $\boldsymbol{X}$ We model the prior distribution for the D latent processes, $p(\boldsymbol{X})$, as a product of D independent multivariate Gaussian distributions, each with zero mean and a covariance matrix $\boldsymbol{K}_d$ induced by a stationary GP kernel function $k_d(\cdot, \cdot; \theta_d)$,

$$p(\boldsymbol{X}) = \prod_{d=1}^D \mathcal{N}(\boldsymbol{X}_d|0, \boldsymbol{K}_d), [\boldsymbol{K}_d]_{t,t'} = k_d(t, t'; \theta_d) \quad (9)$$

where θ_d denotes kernel specific parameters. Important example kernels include the radial basis kernel and the Matérn kernel. See Ch. 4.2 of (Rasmussen & Williams, 2006) for a discussion of different kernels. Any kernel for which parameters can be efficiently updated through automatic differentiation can be used with our method (see Section 3.2). For simplicity, in the following we consider the radial basis kernel, for which θ_d is simply a length scale.

Prior for $\boldsymbol{W}$ The prior distribution for the weights $\boldsymbol{W}$ are modelled as a product of independent multivariate Gaussian distributions along the number of latent dimensions D, with precisions $\{\tau_d\}_{d=1}^D$ (varying among the latent processes but shared across neurons), which in turn are modeled with a gamma prior distribution,

$$p(\boldsymbol{W}) = \prod_{n=1}^N \mathcal{N}(\boldsymbol{w}_n|\boldsymbol{0}, \text{diag}(\frac{1}{\boldsymbol{\tau}})) \quad p(\boldsymbol{\tau}) = \prod_{d=1}^D \mathcal{G}(a_d, b_d). \quad (10)$$

This prior over the precision values is a common choice for *automatic relevance determination* (ARD) of latent dimensions (Ch. 6.4 in (Bishop, 1999)). For the latent dimensions where the precision $\boldsymbol{\tau}$ is large, the variance will be small and thus the weights will be concentrated around their (prior) mean of 0, effectively discarding the latent dimension.

We model the prior over the bias terms $\boldsymbol{\beta}$ with independent Gaussian distributions. Similar to placing a prior over the weights' precisions, we add gamma priors over the common precision parameter τ_β of the distributions,

$$p(\boldsymbol{\beta}) = \prod_{n=1}^N \mathcal{N}(\beta_n|0, \tau_\beta^{-1}) \text{ and } p(\tau_\beta) = \mathcal{G}(c, d), \quad (11)$$

where c and d are the shape and scale parameters respectively of the gamma distribution.

Following (Bishop, 1999), we fix all shape and scale parameters of the gamma variables, $\{a_d, b_d\}, c$ and d, to 10^{-5}. These choices yield non-informative priors.

In addition, as revealed in the augmentation step in Eq. (5), we model priors over the augmented Pólya-gamma (PG) variables $\boldsymbol{\Omega} = \{\omega_{n,t}\}$ with

$$p(\boldsymbol{\Omega}) = \prod_{n=1}^N \prod_{t=1}^T \text{PG}(\omega_{n,t}|y_{n,t} + r_n, 0). \quad (12)$$

Prior for r_n Following (He et al., 2019), we use the improper Gamma distribution $\Gamma(1, 0)$ for the prior for r_n, i.e $p(r_n) \propto r^{-1}$. This choice yields a proper PTN distribution for its posterior distribution.

To this point we have described marginal prior distributions of the latent variables. After applying the augmentation, and the ARD gamma priors, the joint distribution of our model factorizes as

$$p(\boldsymbol{Y}|\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{\beta}, \{\omega_{n,t}, \tau_{n,t}, \xi_{n,t}\}, \{r_n\}) \cdot p(\boldsymbol{W}|\boldsymbol{\tau})p(\boldsymbol{X})p(\boldsymbol{\beta}|\tau_\beta)p(\boldsymbol{\tau})p(\tau_\beta) \prod_n p(r_n) \cdot \prod_{n,t} p(\omega_{n,t})p(\tau_{n,t})p(\xi_{n,t}) \quad (13)$$

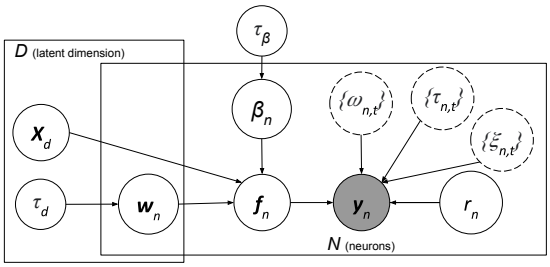

Figure 1: Plate diagram representing our ccGPFA model. Dashed circles indicate the variable is augmented.

with additional factorizations arising from equations Eq. (10), Eq. (11). See Figure 1 for a plate diagram of our model.

3. Inference

In this section, we detail our proposed inference procedure for our ccGPFA model. We show the key steps of formula derivations here, whereas more details can be seen in Appendix C. We use the mean-field variational inference framework to learn the posterior distributions of model variables. Specifically, we approximate the joint augmented posterior $p(\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \tau_\beta, \{\omega_{n,t}, \tau_{n,t}, \xi_{n,t}\}, \{r_n\} | \boldsymbol{Y}) \approx q(\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{\beta}, \boldsymbol{\tau}, \tau_\beta, \{\omega_{n,t}, \tau_{n,t}, \xi_{n,t}\}, \{r_n\})$ as a product of variational distributions

$$q(\boldsymbol{W}) \prod_d q(\boldsymbol{X}_d) q(\boldsymbol{\beta}) q(\tau) q(\{r_n\}) \cdot q(\tau_\beta) q(\{\omega_{n,t}\}) q(\{\tau_{n,t}\}) q(\{\xi_{n,t}\}). \quad (14)$$

We note that we impose independence over the set of latent processes $\boldsymbol{X} = \{\boldsymbol{X}_d\}$ for tractability.

Beyond the factorization above, we do not impose any parametric distributional assumptions, such as setting $q(\boldsymbol{X}_d)$ to be a whitened distribution of the GP priors

3.1. Modeling Objective

Nominally, we would like to optimize the variational distributions to maximize the marginal log-likelihood of the data $\log p(\boldsymbol{Y})$. As marginalization of all model variables is intractable, we instead optimize the variational distributions to maximize a lower bound of $\log p(\boldsymbol{Y})$. Denoting the set of all random variables as $\boldsymbol{\Theta} = \{\boldsymbol{W}, \boldsymbol{X}, \boldsymbol{\Omega}, \boldsymbol{\beta}, \boldsymbol{\tau}, \tau_\beta \{\omega_{n,t}, \tau_{n,t}, \xi_{n,t}\}, \{r_n\}\}$, the evidence lower bound (ELBO) $\mathcal{L}$ is

$$\log p(\boldsymbol{Y}) \geq \mathbb{E}_{q(\boldsymbol{\Theta})} \left[\log \frac{p(\boldsymbol{Y}|\boldsymbol{\Theta}) p(\boldsymbol{\Theta})}{q(\boldsymbol{\Theta})} \right] =: \mathcal{L}. \quad (15)$$

This (standard) bound follows from Jensen's inequality. For completeness, we show it in Appendix D.

3.2. Closed form updates

In this work, to optimize the approximate posterior distributions $q(\boldsymbol{\Theta})$, we employ a variational Expectation-Maximization (variational EM) algorithm. See Algorithm 1 for the high-level pseudocode. In the E-step of this algorithm, we apply sequential updates to the distributions in our variational family following the factorization in (14). This step of the algorithm is equivalent to Coordinate Ascent Variational Inference (CAVI). Following the conditional-conjugacy shown in Section 2.2, we are able to identify closed form updates for this step. In the M-step, we maximize the the marginal lower bound in (15) with respect to the model hyperparameters, such as the characteristic lengthscales $\{\theta_d\}$ of the latent processes $\{\boldsymbol{X}_d\}$.

Expectation step (CAVI): We first show that for our proposed model, we derive closed form coordinate ascent updates that are easy to implement and lead to a computationally efficient procedure. We use the notation $\mathbb{E}[q(-\boldsymbol{W})]$ to represent an expectation with respect to the joint variational distribution of all variables in the variational family except for $\boldsymbol{W}$. Fixing all variational distributions except one (at a time), we denote optimal marginal variational distributions with a super-script $*$.

By a well known property of mean field variational inference (see Section 10.1 in (Bishop, 2006)), the optimal marginal variational distribution $q^*(\boldsymbol{W})$ (with all others fixed) satisfies $q^*(\boldsymbol{W}) \propto \exp\{\mathbb{E}_{q(-\boldsymbol{W})}[\log p(\boldsymbol{Y}, \boldsymbol{\Theta})]\}$, likewise for other factors in Equation (14). We are able to obtain analytic expressions for those optimal marginal variational distributions. For the weights $\boldsymbol{W}$,

$$q^*(\boldsymbol{W}) = \prod_{n=1}^N \mathcal{N}(\boldsymbol{w}_n | m_n, S_n) \quad \text{with}$$
$$m_n = S_n (\mathbb{E}[\boldsymbol{X}]\, \mathbb{E}[\boldsymbol{\Omega}_n] (\boldsymbol{z}_n^\top - \mathbb{E}[\beta_n] \mathbf{1}))$$
$$S_n = (\mathrm{diag}(\mathbb{E}[\boldsymbol{\tau}]) + \mathbb{E}[\boldsymbol{X}\, \mathbb{E}[\boldsymbol{\Omega}_n] \boldsymbol{X}^\top])^{-1}.$$

For the base spike rate intensity $\boldsymbol{\beta}$,

$$q^*(\boldsymbol{\beta}) = \prod_{n=1}^N \mathcal{N}(\beta_n | \mu_n, \sigma_n^2) \quad \text{with}$$
$$\mu_n = \sigma_n^2 (\boldsymbol{z}_n - \mathbb{E}[\boldsymbol{w}_n]\, \mathbb{E}[\boldsymbol{X}])\, \mathbb{E}[\boldsymbol{\Omega}_n] \mathbf{1}$$
$$\sigma_n^2 = 1/(\mathbb{E}[\tau_\beta] + \mathrm{Tr}(\mathbb{E}[\boldsymbol{\Omega}_n])).$$

For the gamma precision variables $\{\boldsymbol{\tau}, \tau_\beta\}$,

$$q^*(\tau_\beta) = \Gamma(c + \frac{N}{2}, d + \frac{1}{2} \sum_{n=1}^N \mathbb{E}[\beta_n^2])$$
$$q^*(\boldsymbol{\tau}) = \prod_{d=1}^D \Gamma(a_d + \frac{N}{2}, b_d + \frac{1}{2} \sum_{n=1}^N \mathbb{E}\left[\boldsymbol{w}_{n,d}^2\right]).$$

For each of the D latent processes $\boldsymbol{X}_d$,

$$
\begin{aligned}
&q^*(\boldsymbol{X}_d) = \mathcal{N}(\hat{\boldsymbol{\mu}}_d, \hat{\boldsymbol{K}}_d) \quad \text{with} \\
&\hat{\boldsymbol{\mu}}_d = \hat{\boldsymbol{K}}_d \Bigg(\mathbb{E}[\boldsymbol{\Omega}_n] \sum_n \mathbb{E}[\boldsymbol{w}_{n,d}] \big(\boldsymbol{z}_n^\top - \mathbb{E}[\beta_n]\mathbf{1} \\
&\qquad\qquad - \sum_{d' \neq d} \mathbb{E}[\boldsymbol{w}_{n,d'}]\, \mathbb{E}[\boldsymbol{X}_{d'}^\top]\big) \Bigg) \\
&\hat{\boldsymbol{K}}_d = (\boldsymbol{K}_d^{-1} + \sum_n \mathbb{E}[w_{n,d}^2]\, \mathbb{E}[\boldsymbol{\Omega}_n])^{-1}.
\end{aligned}
\tag{16}
$$

For the PG variables $\boldsymbol{\Omega} = \{\omega_{n,t}\}$,

$$
q^*(\boldsymbol{\Omega}) = \prod_{n=1}^{N} \prod_{t=1}^{\top} \mathrm{PG}(y_{n,t} + r_n, c_{n,t}) \text{ with } c_{n,t} = \sqrt{\mathbb{E}[f_{n,t}^2]}.
$$

For the dispersion variables $\{r_n\}$,

$$
\begin{aligned}
&q^*(r_n) = \mathrm{PTN}(p, a, b), \text{with } p = T; a = \sum_t \mathbb{E}[\xi_{n,t}]; \\
&b = \sum_t (\mathbb{E}[\log \tau_{n,t}] + \gamma - \log 2 - \frac{1}{2}\, \mathbb{E}[f_{n,t}]).
\end{aligned}
\tag{17}
$$

For brevity, we defer details of update rules for the remaining augmented variables to Appendix C.

Joint analysis of repeated trials For simplicity sake, the above derivations show update rules for a single trial. However, it can be shown that the above updates can be seamlessly extended to handle repeated trials, such as for vision experiments with repeated trials under the same visual stimulus. See Appendix B.3 for details.

Non-identifiability In classical GPFA model, there is a problem of model non-identifiability due to the interaction of the loading weights and the latent processes (Yu et al., 2008). Orthonormalization can be applied on latent states as post processing step. In addition to this, in a negative binomial GPFA model, the dispersion variable r_n and the latent function $f_{n,t}$ compete to explain the variance of spike counts as $r_n e^{f_n}(1 + e^{f_n})$ represents the variance of a negative binomial variable. To make inference stable in practice, we apply clipping of mean values of the latent function, $\mathbb{E}[f_{n,t}]$, in the variational update of dispersion variables to a given threshold. This is analogous of gradient clipping, a technique common in the machine learning literature.

Updating hyperparameters In the M-step, we maximize the ELBO w.r.t the GP kernel parameters $\{\theta_d\}_{d=1}^D$. In practice, we optimize those parameters with respect to the ELBO (15) using the Adam optimizer algorithm (Kingma & Ba, 2017). We do not derive explicit formulas for the gradient here. In practice, we use the automatic differentiation engine in Pytorch (Paszke et al., 2019). We include the simplification of the ELBO formula in Appendix D.

Algorithm 1 Inference procedure of ccGPFA

```
Input data Y
Initialize latent variables Z ∈ Θ
Initialize hyperparameters: {θ_d}
while not converged (w.r.t. ELBO (15)) do

  { Expectation step }
  while E-step stopping criterion not reached do
    for Z ∈ Θ do
      Update q*(Z)
    end for
  end while

  { Maximization step }
  while M-step stopping criterion not reached do
    Update length scales {θ_d}
  end while
end while
Return variational q*(Θ) and hyperparameters {θ_d}
```

3.3. Scalable Inference

In Section 3.2, we identified closed form updates for the expectation step. When the observation length T is large, evaluating (16) can become challenging due to inverting $T \times T$ covariance matrices. For inference, we use $M < T$ uniformly-spaced inducing points in modeling the latent GPs $\boldsymbol{X}$ to improve efficiency (Quinonero-Candela & Rasmussen, 2005).

For each latent dimension d, we modify the model by adding set of M inducing points on top of the existing T time points. And for each set of inducing points, we have corresponding $\boldsymbol{U}_d$. We first define the joint prior distribution of $\boldsymbol{X}_d$ and $\boldsymbol{U}_d$

$$
p\left(\begin{bmatrix}\boldsymbol{X}_d \\ \boldsymbol{U}_d\end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{0} \\ \mathbf{0}\end{bmatrix}, \begin{bmatrix}\boldsymbol{K}_{d,tt} & \boldsymbol{K}_{d,tm} \\ \boldsymbol{K}_{d,mt} & \boldsymbol{K}_{d,mm}\end{bmatrix}\right), \tag{18}
$$

where $\boldsymbol{K}_{d,tm}$ and $\boldsymbol{K}_{d,mt}$ denote cross variances between $\boldsymbol{X}_d$ and $\boldsymbol{U}_d$.

Using a known property of the multivariate Gaussian distribution, the conditional distribution of $p(\boldsymbol{X}_d|\boldsymbol{U}_d)$ is given by

$$
\begin{aligned}
p(\boldsymbol{X}_d|\boldsymbol{U}_d) = \mathcal{N}(&\boldsymbol{U}_d \boldsymbol{K}_{d,mm}^{-1} \boldsymbol{K}_{d,mt}, \\
&\boldsymbol{K}_{d,tt} - \boldsymbol{K}_{d,tm} \boldsymbol{K}_{d,mm}^{-1} \boldsymbol{K}_{d,mt}).
\end{aligned}
\tag{19}
$$

Generalizing for all D latent processes we have

$$
p(\boldsymbol{X}, \boldsymbol{U}) = \prod_{d=1}^{D} p\left(\begin{bmatrix}\boldsymbol{X}_d \\ \boldsymbol{U}_d\end{bmatrix}\right). \tag{20}
$$

As pointed out in (Luttinen & Ilin, 2009), assuming the inducing points capture the information in the data well, we can approximate the joint posterior as

$$q(\boldsymbol{X}, \boldsymbol{U}) = \prod_{d=1}^{D} p(\boldsymbol{X}_d|\boldsymbol{U}_d)q(\boldsymbol{U}_d). \tag{21}$$

Careful derivations show all prior derived optimal distributions except for $q^*(\boldsymbol{X}_d)$ will remain the same, even after adding the inducing point variables. The updates for the latent processes would be simply marginalizing out $\boldsymbol{U}$ from the joint distribution $q(\boldsymbol{X}, \boldsymbol{U})$. Therefore, it suffices to derive the optimal distributions of the inducing variables.

$$q^*(\boldsymbol{U}_d) \propto \exp\{\mathbb{E}_{q(-\boldsymbol{U}_d)}[\log p(\boldsymbol{Y}, \boldsymbol{\Theta})]\} \propto \mathcal{N}(\boldsymbol{m}_d, \boldsymbol{S}_d),$$

where

$$\begin{aligned}
\boldsymbol{S}_d &= \left(\boldsymbol{K}_{d,mm}^{-1} + \boldsymbol{K}_{d,mm}^{-1}\boldsymbol{K}_{d,mt}\left(\sum_n \mathbb{E}[\boldsymbol{w}_{n,d}^2]\,\mathbb{E}[\boldsymbol{\Omega}_n]\right)\right. \\
&\quad \left.\boldsymbol{K}_{d,mt}^{\top}\boldsymbol{K}_{d,mm}^{-1}\right)^{-1} \\
\boldsymbol{m}_d &= \boldsymbol{S}_d\left(\boldsymbol{K}_{d,mm}^{-1}\boldsymbol{K}_{d,mt}\,\mathbb{E}[\boldsymbol{\Omega}_n]\right)\left(\sum_n \mathbb{E}[\boldsymbol{w}_{n,d}]\left(\mathbb{E}[\boldsymbol{z}_n^{\top}]\right.\right. \\
&\quad \left.\left. - \sum_{d' \neq d} \mathbb{E}[\boldsymbol{w}_{n,d'}]\,\mathbb{E}[\boldsymbol{X}_{d'}^{\top}] + \mathbb{E}[\boldsymbol{\beta}_n]\mathbf{1}\right)\right).
\end{aligned} \tag{22}$$

With that we can then identify $q^*(\boldsymbol{X}_d)$

$$\begin{aligned}
&q^*(\boldsymbol{X}_d) = \mathcal{N}(\boldsymbol{K}_{d,tm}\boldsymbol{K}_{d,mm}^{-1}\boldsymbol{m}_d, \\
&\boldsymbol{K}_{d,tt} - \boldsymbol{K}_{d,tm}\boldsymbol{K}_{d,mm}^{-1}(\boldsymbol{K}_{d,mm} - \boldsymbol{S}_d)\boldsymbol{K}_{d,mm}^{-1}\boldsymbol{K}_{d,tm}^{T}).
\end{aligned} \tag{23}$$

Using the above update, we effectively avoid an expensive $T \times T$ matrix inversion.

Accelerated Inference via Natural Gradients We note the above updates still require summations through the entire data, which could be prohibitive for analysing long, continuous recordings. To tackle this challenge, we construct a stochastic version of inference using exponential family distribution properties to derive stochastic natural gradients (Hoffman et al., 2013). Unlike other non-conjugate methods, computing the natural gradients does not require computing inverse Fisher information matrix. Deferring details to Appendix E, we provide the update rules for a natural parameter η of a variational distribution,

$$\eta \leftarrow (1 - step\_size) * \eta + step\_size * \eta_{new} \tag{24}$$

where the parameter η_{new} denotes the noisy natural gradient that uses a mini-batch from the dataset and is appropriately scaled. And $step\_size$ is a learning rate hyperparameter which can be close to 1 since we work with natural gradients (Hoffman et al., 2013). However, Adaptive learning rates (Ranganath et al., 2013) can also be applied.

Using sparse GP with M inducing points and mini-batching with batch size B, the time complexity of our model is $\mathcal{O}(D(M^3 + BM^2))$.

4. Experiments

4.1. Visual Coding Experiment

We first compared our method with other baselines on a drifting gratings recording from visual cortex in mice sourced from Allen Brain Observatory data.

Dataset We considered a passive observation segment from the *Allen Brain Observatory: Visual Coding Neuropixels Dataset* (Allen Institute, 2019). Specifically, we analysed a simultaneous recording of 176 neurons from the mouse primary visual (V1) cortex. The recording was over 75 trials under a drifting gratings stimulus. Each trial was 2 seconds long. We binned the spike train data into 15 ms bins. To evaluate the goodness of fit, we randomly shuffled and split the data into 50 held-in trials and 25 held-out trials.

Baseline Methods We compared against three common baselines: GPFA (Yu et al., 2008)[1], PAL (Keeley et al., 2020b)[2], and bGPFA (Jensen et al., 2021)[3]. For our method, we considered both binomial and negative binomial observation models. For multiple trial data, (Keeley et al., 2020b) and our methods fit a GPFA model with shared set of latents and loading weights across trial. We note that (Yu et al., 2008) and (Jensen et al., 2021), however, learn different latent processes for each trial (with a shared covariance matrix; see Appendix B.3 for details). In these experiments, to evaluate their fitted models on test data, we computed an average of the latents learned on training data trials. To ensure fair comparisons in terms of run time, in this experiment we monitored convergence of the training loglikelihood as a common stopping criteria. For the baselines (Yu et al., 2008; Keeley et al., 2020b), this required multiple runs of the algorithms and manually choosing the best number of iteration.

Metrics We tested goodness-of-fit of the inferred spike count probabilities and dispersion parameters by computing the test log likelihood on held-out trials, computing its mean and standard error per neuron and time step.

Implementation Notes In these experiments, since the trial

[1]https://github.com/NeuralEnsemble/elephant implemented in Python by (Denker et al., 2018)

[2]https://github.com/skeeley/Count_GPFA

[3]https://github.com/tachukao/mgplvm-pytorch

Table 2: Performance comparison between our method and baselines in terms of test negative log likelihood (normalized by the total number of bins) and run time (in seconds).

Method	Test NLL	Run time
Yu et al. (2008)	0.3565 ± 0.0013	65.91
Keeley et al. (2020b)	0.3407 ± 0.0010	62.95
Jensen et al. (2021)	0.3504 ± 0.0013	1321.97
Ours (Binomial)	0.3390 ± 0.0011	9.25
Ours (NegBinomial)	$\mathbf{0.3333 \pm 0.0010}$	**7.67**

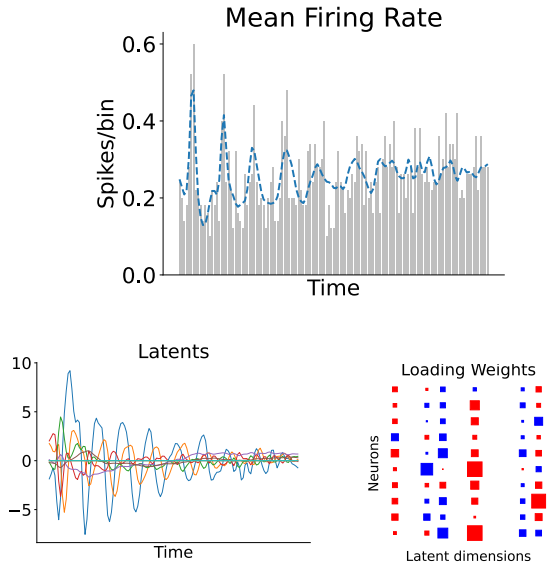

Figure 2: (**Top**) inferred mean firing rates by our method of a neuron across time along with peri-stimulus time histogram; (**Left**) orthonormalized latent processes; **(Right)** loading weights identified by our method ccGPFA for 10 neurons.

length was short (133 bins), we did not use inducing points or natural gradients (discussed in Section 3.3) to accelerate inference for our models. The inference procedure is implemented using as a PyTorch (v2.0.1) as a base library. We removed non-spiking neurons. See Appendix G.1 for details.

Results and Discussion We summarize the main results in terms of goodness of fit on test data and run time in Table 2. Our methods out-performed all of the baselines in terms of both accuracy and speed. Both of the models we fit had lower test negative log likelihoods than any baseline, with our negative binomial model performing the best. Our inference procedure ran in under 10 seconds for both, about $1/7$ of the time of the fastest and most accurate baseline (Keeley et al., 2020b).

Figure 2 depicts visualizations of our fitted negative binomial model. In Figure 2, the first plot shows a dotted curve representing the mean firing rate of a neuron, as inferred by our model. The background peri-stimulus histogram represents the neuron's empirical spike rate constructed by computing the mean number of spikes per bin across training trials. The plot of the latent Gaussian processes next to it captures the dominant cyclic pattern also seen in the neural activity. Furthermore, the Hinton diagram of the weight coefficients (blue means positive, red means negative; size proportional to weight magnitude) shows our model effectively eliminated 4 out of the 10 pre-specified set of latents to yield a concise representation.

We note Jensen et al. (2021)'s bGPFA converges slowly and in this experiment no latent processes were fully eliminated, despite employing ARD. We speculate that this may in part be due to over parameterization in how Jensen et al. (2021) models repeated trials, with distinct latents for each trial that have a shared covariance matrix. An additional factor that may explain why no latents were fully eliminated is that the whitened parameterization Jensen et al. (2021) employ constrains the flexibility of the inferred latents, which may necessitate using more latents to model the data than other methods without such constraints. Yu et al. (2008)'s GPFA shows relatively poor performance in terms of log likelihood, which may partly highlight the importance of count models instead of a Gaussian model. We also note that Yu et al. (2008)'s GPFA, like Jensen et al. (2021)'s bGPFA, had unique latents for each trial, so over-parameterization may have been a factor as well.

4.2. Primate Behavioral Experiment

In this subsection, we show the effectiveness of our model on a behavioral decoding task.

Dataset We considered a delayed-reaching task dataset, *MC_Maze* (Churchland et al., 2010). *MC_Maze* contains simultaneous recordings of neural and behavior activity (e.g. reach velocities) while a macaque performs a delayed reaching task. We selected a total of 9 experimental conditions (i.e. 9 different target locations), all with a single reaching target and no barrier.

Preprocessing For each experimental condition, there are a total of 18-19 trials. Each trials were 500 ms long recording of $N = 162$ neurons. We binned the spikes into 5 ms bins, yielding binned spike trains with $T = 100$ time steps.

Experiment Setup We first fit individual GPFA models for each condition dataset. Using the inferred parameters, we computed the firing rates of individual neurons. We assessed the predictive quality of these firing rates on decoding the simultaneously recorded behavior. Specifically, we conducted a 3-fold cross validation test by splitting the conditions into train (6 conditions) and test (3 conditions)

Table 3: **MC Maze** Performance comparison in terms of R2 regression scores for predicting behavior.

Method	Test R2
Yu et al. (2008)	0.290 ± 0.011
Keeley et al. (2020b)	0.585 ± 0.108
Jensen et al. (2021)	0.327 ± 0.027
Ours (Binomial)	0.576 ± 0.074
Ours (NegBinomial)	$\mathbf{0.755 \pm 0.028}$

sets. Using the train set, we fit a linear map from its firing rates to a simultaneously recorded hand velocities (along horizontal and vertical dimensions). For the linear map, we applied a grid search to select the best regularizer (from 50 evenly sampled points between 1e-4 to 1e+4 in logspace) through 5-fold cross validation with an L2 penalty. We tested the generalization of the linear maps by computing R2 score on the test set. We show the mean R2 scores and their corresponding standard error in Table 3). Since we only compared performance in regression scores for this experiment, not run times, we used default stopping criteria for the baselines. We removed neurons with low spiking in each condition, imputing zero firing rate if the neuron was active in other conditions. See Appendix G.1 for details.

Results and Discussion From Table 3, we can observe our method has a clear performance gain over the baselines. This implies the latent features our models inferred were more predictive of the simultaneously recorded behavior. (Jensen et al., 2021)'s bGFPA was originally developed for long recordings and models repeated trials as independent. However, the trials in the experiments are short and sparse. Therefore, it suffers from the loss of predictive accuracy, as can be seen from the R2 scores. In contrast (Keeley et al., 2020b)'s PAL achieves a competitive performance as it, similar to ours, exploits the i.i.d. nature of the repeated trials. Our negative binomial ccGPFA model shows the best predictive performance. The differences with the baselines highlight the predictive quality of latent processes inferred by our model.

4.3. Scalability Experiment

We setup a synthetic experiment to explore the scalability of our model.

Synthetic Dataset We simulated three neural datasets with $N = 100$ neurons and varying number of time steps $T = \{300, 900, 1500\}$. We first generated three latent processes drawn from a zero mean GP with a lengthscale of 10. For each neuron-latent pair, we generated loading weights from a standard normal distribution scaled with 0.1. For each neuron, we randomly sampled its dispersion parameter from Uniform$[1, 10]$. We then sampled a total of 10 i.i.d. trials from a negative binomial GPFA model Equation (2). We trained GPFA models on the first 7 trials and tested their negative log-likelihood results on the remaining 3 trials. We removed non-spiking neurons. See Appendix G.1 for details.

Table 4: **Scalability Experiment** Performance comparison in terms of average test negative loglikelihood (per spike count per trial) and running times in seconds (in parentheses). Bin/NegBin stands for our (negative) binomial ccGPFA model. The suffixes $\{50, 100, 200\}$ indicate the number of inducing points used.

Method	T=300		T=1500	
Keeley et al. (2020b)	1.517	(1658)	1.514	(112644)
Bin	1.458	(19)	1.451	(631)
NegBin	**1.416**	**(10)**	**1.415**	(1422)
Bin-50	1.458	(5)	1.460	(6)
Bin-100	1.458	(6)	1.452	(8)
Bin-200	1.458	(13)	1.452	(15)
NegBin-50	**1.416**	(12)	1.422	(31)
NegBin-100	**1.416**	(11)	1.416	(21)
NegBin-200	**1.416**	(21)	**1.415**	**(55)**

Methods We tested both binomial and negative binomial likelihood models with/without using sparse GPs as shown in Section 3.3. We used $M = \{50, 100, 200\}$ as choices for the number of inducing points. In the table, we indicate the use of inducing points with a suffix (e.g Ours (NegBinomial)-200 indicates negative binomial model with 200 inducing points). For comparison we added the (Keeley et al., 2020b) baseline which showed the most competitive performance in the previous experiments.

Results and Discussion Due to limited space, we highlight some results in Section 4.3. See Table 5 in Appendix G.2 for more details. Overall, our negative binomial method achieves the best test negative likelihood. In addition, we note our methods that use inducing points show comparable performance, with large reduction in running times. Keeley et al. (2020b)'s PAL yields competitive accuracy but takes orders of magnitude more time than the ccGPFA methods. For the scalability experiments, we capped the number of iterations to 1000 for (Keeley et al., 2020b). Despite this limit, for $T = 1500$, it took over 31 hours of runtime, in contrast to our best sparse ccGPFA model which finished under a minute. This can be partly explained by the expensive second-order optimization objective used in PAL's implementation.

A Python implementation of our algorithm and the above experiments is publicly available at `https://github.com/yididiyan/ccgpfa/tree/public`.

Acknowledgement

We would like to thank the anonymous reviewers for their valuable feedback which helped improve the final revision of the paper.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Allen Institute. Allen Institute MindScope Program, 2019. https://portal.brain-map.org/explore/circuits, 2019. Accessed: 2023-12-10.

Bishop, C. Variational principal components. In *1999 Ninth International Conference on Artificial Neural Networks ICANN 99.(Conf. Publ. No. 470)*, volume 1, pp. 509–514. IET, 1999.

Bishop, C. *Pattern Recognition and Machine Learning*. Springer New York, NY, 2006.

Churchland, M. M., Cunningham, J. P., Kaufman, M. T., Ryu, S. I., and Shenoy, K. V. Cortical preparatory activity: representation of movement or first cog in a dynamical machine? *Neuron*, 68(3):387–400, 2010.

Denker, M., Yegenoglu, A., and Grün, S. Collaborative HPC-enabled workflows on the HBP Collaboratory using the Elephant framework. In *Neuroinformatics 2018*, pp. P19, 2018.

Dowling, M., Zhao, Y., and Park, I. M. Linear time GPs for inferring latent trajectories from neural spike trains. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, 2023.

Duncker, L. and Sahani, M. Temporal alignment and latent Gaussian process factor inference in population spike trains. *Advances in Neural Information Processing Systems*, 31, 2018.

Gao, Y., Busing, L., Shenoy, K. V., and Cunningham, J. P. High-dimensional neural spike train analysis with generalized count linear dynamical systems. *Advances in Neural Information Processing Systems*, 28, 2015.

Glaser, J. I., Benjamin, A. S., Chowdhury, R. H., Perich, M. G., Miller, L. E., and Kording, K. P. Machine learning for neural decoding. *eNeuro*, 7(4), 2020.

Glynn, C., He, J., Polson, N. G., and Xu, J. Bayesian inference for Pólya inverse gamma models. *arXiv preprint arXiv:1905.12141*, 15:17, 2019.

Gokcen, E., Jasper, A. I., Semedo, J. D., Zandvakili, A., Kohn, A., Machens, C. K., and Yu, B. M. Disentangling the flow of signals between populations of neurons. *Nature Computational Science*, 2(8):512–525, 2022.

Gokcen, E., Jasper, A. I., Xu, A., Kohn, A., Machens, C. K., and Byron, M. Y. Uncovering motifs of concurrent signaling across multiple neuronal populations. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

He, J., Polson, N. G., and Xu, J. Data augementation with Pólya inverse gamma. *arXiv preprint arXiv:1905.12141*, 2019.

Hoffman, M. D., Blei, D. M., Wang, C., and Paisley, J. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

Jankowiak, M. Fast Bayesian variable selection in binomial and negative binomial regression. *arXiv preprint arXiv:2106.14981*, 2021.

Jensen, K., Kao, T.-C., Stone, J., and Hennequin, G. Scalable Bayesian GPFA with automatic relevance determination and discrete noise models. *Advances in Neural Information Processing Systems*, 34:10613–10626, 2021.

Jensen, K. T., Liu, D., Kao, T.-C., Lengyel, M., and Hennequin, G. Beyond the Euclidean brain: inferring non-Euclidean latent trajectories from spike trains. *bioRxiv*, pp. 2022–05, 2022.

Keeley, S., Aoi, M., Yu, Y., Smith, S., and Pillow, J. W. Identifying signal and noise structure in neural population activity with Gaussian process factor models. *Advances in Neural Information Processing Systems*, 33:13795–13805, 2020a.

Keeley, S., Zoltowski, D., Yu, Y., Smith, S., and Pillow, J. Efficient non-conjugate Gaussian process factor models for spike count data using polynomial approximations. In *International Conference on Machine Learning*, pp. 5177–5186. PMLR, 2020b.

Kim, T. D., Luo, T. Z., Pillow, J. W., and Brody, C. D. Inferring latent dynamics underlying neural population activity via neural differential equations. In *International Conference on Machine Learning*, pp. 5551–5561. PMLR, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2017.

Klami, A. Polya-gamma augmentations for factor models. In *Asian Conference on Machine Learning*, pp. 112–128. PMLR, 2015.

Lakshmanan, K. C., Sadtler, P. T., Tyler-Kabara, E. C., Batista, A. P., and Yu, B. M. Extracting low-dimensional latent structure from time series in the presence of delays. *Neural Computation*, 27(9):1825–1856, 2015.

Locatello, F., Dresdner, G., Khanna, R., Valera, I., and Rätsch, G. Boosting black box variational inference. *Advances in Neural Information Processing Systems*, 31, 2018.

Luttinen, J. and Ilin, A. Variational Gaussian-process factor analysis for modeling spatio-temporal data. *Advances in Neural Information Processing Systems*, 22, 2009.

Pandarinath, C., O'Shea, D. J., Collins, J., Jozefowicz, R., Stavisky, S. D., Kao, J. C., Trautmann, E. M., Kaufman, M. T., Ryu, S. I., Hochberg, L. R., et al. Inferring single-trial neural population dynamics using sequential auto-encoders. *Nature Methods*, 15(10):805–815, 2018.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019.

Petreska, B., Yu, B. M., Cunningham, J. P., Santhanam, G., Ryu, S., Shenoy, K. V., and Sahani, M. Dynamical segmentation of single trials from population neural data. *Advances in Neural Information Processing Systems*, 24, 2011.

Pillow, J. and Scott, J. Fully Bayesian inference for neural models with negative-binomial spiking. *Advances in Neural Information Processing Systems*, 25, 2012.

Polson, N. G., Scott, J. G., and Windle, J. Bayesian inference for logistic models using Pólya–gamma latent variables. *Journal of the American statistical Association*, 108(504):1339–1349, 2013.

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

Ranganath, R., Wang, C., David, B., and Xing, E. An adaptive learning rate for stochastic variational inference. In *International Conference on Machine Learning*, pp. 298–306. PMLR, 2013.

Rasmussen, C. E. and Williams, C. K. *Gaussian Processes for Machine Learning*, volume 1. Springer, 2006.

Rutten, V., Bernacchia, A., Sahani, M., and Hennequin, G. Non-reversible Gaussian processes for identifying latent dynamical structure in neural data. *Advances in Neural Information Processing Systems*, 33:9622–9632, 2020.

Schimel, M., Kao, T.-C., Jensen, K. T., and Hennequin, G. ilqr-vae: control-based learning of input-driven dynamics with applications to neural data. In *International Conference on Learning Representations*, 2021.

Semedo, J., Zandvakili, A., Kohn, A., Machens, C. K., and Yu, B. M. Extracting latent structure from multiple interacting neural populations. *Advances in Neural Information Processing Systems*, 27, 2014.

Soulat, H., Keshavarzi, S., Margrie, T., and Sahani, M. Probabilistic tensor decomposition of neural population spiking activity. *Advances in Neural Information Processing Systems*, 34:15969–15980, 2021.

Steinmetz, N. A., Aydin, C., Lebedeva, A., Okun, M., Pachitariu, M., Bauza, M., Beau, M., Bhagat, J., Böhm, C., Broux, M., et al. Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science*, 372(6539):eabf4588, 2021.

Titsias, M. Variational learning of inducing variables in sparse Gaussian processes. In *Artificial Intelligence and Statistics*, pp. 567–574. PMLR, 2009.

Wenzel, F., Galy-Fajou, T., Donner, C., Kloft, M., and Opper, M. Efficient Gaussian process classification using Pòlya-gamma data augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5417–5424, 2019.

Yu, B. M., Afshar, A., Santhanam, G., Ryu, S., Shenoy, K. V., and Sahani, M. Extracting dynamical structure embedded in neural activity. *Advances in Neural Information Processing Systems*, 18, 2005.

Yu, B. M., Cunningham, J. P., Santhanam, G., Ryu, S., Shenoy, K. V., and Sahani, M. Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity. *Advances in Neural Information Processing Systems*, 21, 2008.

Zhou, F., Li, Z., Fan, X., Wang, Y., Sowmya, A., and Chen, F. Efficient inference for nonparametric Hawkes processes using auxiliary latent variables. *The Journal of Machine Learning Research*, 21(1):9745–9775, 2020.

We include the following in the supplementary material:

- Appendix A: Related Works - Extended Discussion
- Appendix B: Augmentation Details
- Appendix C: Variational distribution updates
- Appendix D: ELBO Expression
- Appendix E: Natural Gradients
- Appendix F: Important Densities
- Appendix G: Experiment Details and Results

A. Related Works – Extended Discussion

In this section, we extend the review of the related work presented in Section 1.

A.1. Latent Variable Models

In general, latent variable models (LVMs) are used to encode neural dynamics from data, which can be later used to decode stimulus inputs and behavioural variables (Glaser et al., 2020; Schimel et al., 2021). Broadly, in terms of methodology, LVMs for inferring neural dynamics from data fall into two categories: Gaussian Process based methods (Yu et al., 2008; Lakshmanan et al., 2015; Keeley et al., 2020b; Jensen et al., 2021) and auto-regressive methods (Yu et al., 2005; Petreska et al., 2011; Semedo et al., 2014; Pandarinath et al., 2018; Kim et al., 2021). Both lines of works share an underlying assumption that activity of neurons is driven by a set of shared low-dimensional latent trajectories. The main distinction lies on the priors placed for the these factors. In addition, the inferred latent state is considered to be discrete-time samples. Duncker & Sahani (2018) extends the GPFA model to continuous time. Another common simplifying assumption is that the underlying latent trajectories are assumed independent *a priori*. Rutten et al. (2020) extends the GPFA model to address the limitation. In addition, these methods are mostly reserved to observations that are modeled using transformations of linear combinations of latent states. However, a recent works (Jensen et al., 2022) extend GPFA model to non-Euclidean manifold. Recently, in (Gokcen et al., 2022; 2023) extend the standard GPFA to model multiple interacting populations populations.

In this work, our focus lies on a non-conjugate extension of the GPFA model arising from modeling count data. The closest works to ours, (Keeley et al., 2020b; Jensen et al., 2021), employ approximate techniques to handle non-conjugacy. We present a data augmentation based method to handle non-conjugacy that results in simple closed form solutions. Jensen et al. (2021) applied variational inference using whitened parameterization tailored to radial basis function (RBF) kernels that resulted in a scalable procedure.

A.2. Data augmentation

Polson et al. (2013) introduced Pólya-gamma augmentation to yield model likelihood conditionally conjugate, which is vital in tractable Bayesian inference. This technique has been applied to logistic models (Jankowiak, 2021; Wenzel et al., 2019), point process models (Zhou et al., 2020), and factor models (Pillow & Scott, 2012; Klami, 2015; Soulat et al., 2021). However, methods using this technique specifically for negative binomial observation model treat dispersion variables as hyperparameters despite its importance in spike data. He et al. (2019) presents a rich set of augmentation techniques for Gamma based models, including for negative binomial regression model. We extend this augmentation for GP based latent variable model. To the best of our knowledge, our work is the first to utilize this augmentation technique on a latent variable model as GPFA. In section Section 3.2, we elaborate its practical challenges and solution in its application.

A.3. Sparse Gaussian processes and natural gradients

Sparse Gaussian processes (GP) have been applied to GP models to mitigate computational complexity, resulting in scalable inference (Quinonero-Candela & Rasmussen, 2005; Titsias, 2009). They are akin to a low-rank approximation of GP covariances, which results in efficient and scalable inference. In our model, the latent states are sampled at evenly spaced points in time. Therefore, we used fixed inducing points evenly spaced across time. When employing Gaussian processes

for modeling a set of n data points, using sparse Gaussian processes ($m \ll n$ inducing points) can lead to a reduction in computational complexity from $\mathcal{O}(n^3)$ to $\mathcal{O}(nm^2)$ by inverting an $m \times m$ matrix instead of inverting a larger $n \times n$ matrix.

B. Augmentation Details

Here we show detail of the augmentation steps necessary to make a negative binomial GPFA model conditional conjugate for efficient Bayesian inference.

B.1. NegBinomial

Recall, the likelihood of the observed spike counts $\boldsymbol{Y} \in \mathbb{N}^{N \times T}$ under a negative binomial GPFA model, given in Eq. 2

$$p(\boldsymbol{Y}|\cdot) \propto \prod_{n,t} \frac{\Gamma(y_{n,t} + r_n)}{\Gamma(r_n)} \frac{(e^{f_{n,t}})^{y_{n,t}}}{(1 + e^{f_{n,t}})^{y_{n,t}+r_n}}. \qquad \text{(removing } \{y_{n,t}!\} \text{ as constants)}$$

To apply Bayesian inference on model variables $\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{\beta}$ and $\{r_n\}$, one must either marginalize them out from the joint distribution $p(\boldsymbol{Y}, \boldsymbol{X}, \boldsymbol{W}, \boldsymbol{\beta}, \{r_n\})$ or find prior distributions for which the resulting posterior belongs to a known class of distributions. The former entails integration in higher dimensions which is analytically and computationally intractable. And unfortunately, there is also no known conjugate prior for the given model likelihood.

Following (He et al., 2019), we show series of steps to augment the GPFA model to make the likelihood tractable. We first breakdown the likelihood into three parts.

$$p(\boldsymbol{Y}|\cdot) \propto \Bigg(\prod_{n,t} \underbrace{\Gamma(y_{n,t} + r_n)}_{\text{term 1}} \Bigg) \Bigg(\prod_{n,t} \underbrace{\frac{1}{\Gamma(r_n)}}_{\text{term 2}} \Bigg) \Bigg(\prod_{n,t} \underbrace{\frac{(e^{f_{n,t}})^{y_{n,t}}}{(1 + e^{f_{n,t}})^{y_{n,t}+r_n}}}_{\text{term 3}} \Bigg). \tag{25}$$

Commonly used Pólya-gamma data augmentation techniques (Polson et al., 2013) apply an integral identity to term 3 into a tractable form yielding Gaussian likelihood over a transformed variable. Such augmentations treat the dispersion variable r_n as a parameter, making terms 1 and 2 constant upon conditioning. Then r_n is optimized in an outer loop using a common second order optimization techniques (Pillow & Scott, 2012; Soulat et al., 2021). To apply a fully Bayesian inference on the model, i.e treating r_n as a random variable, we have to also deal with terms 1 and 2. And the gamma function does not admit a natural conjugate prior distribution. To solve this, He et al. (2019) identifies the following integrals.

First, we can express the product of gamma functions in term 1 as follows,

$$\prod_{n,t} \Gamma(y_{n,t} + r_n) \propto \prod_{n,t} \int_0^\infty \tau_{n,t}^{(y_{n,t}+r_n)-1} e^{-\tau_{n,t}} d\tau_{n,t}. \tag{26}$$

Next, we apply another integral equivalence to the reciprocal gamma function in term 2,

$$\prod_{n,t} \frac{1}{\Gamma(r_n)} \propto \prod_{n,t} \int_0^\infty r_n e^{-r_n^2 \xi_{n,t} + \gamma r_n} \, \text{P-IG}(\xi_{n,t}|0) d\xi_{n,t} \tag{27}$$

$$\propto \prod_{n,t} r_n e^{\gamma r_n} \int_0^\infty e^{-r_n^2 \xi_{n,t}} \, \text{P-IG}(\xi_{n,t}|0) d\xi_{n,t}, \tag{28}$$

where (28) follows from pulling out constants and $\text{P-IG}(0)$ denotes the PDF of a Polya-Inverse Gamma distribution, a new class of distributions developed in (He et al., 2019), with tilting parameter 0. In the equation above, $\gamma \approx 0.577$ refers to Euler's constant.

Finally, we apply an integral identity to transform term 3,

$$\prod_{n,t} \frac{(e^{f_{n,t}})^{y_{n,t}}}{(1 + e^{f_{n,t}})^{y_{n,t}+r_n}} = \prod_{n,t} 2^{-b} \exp((a - b/2) f_{n,t}) \int_0^\infty \exp\Big\{ -\frac{\omega_{n,t} f_{n,t}^2}{2} \Big\} \, \text{PG}(\omega_{n,t}|b, 0) d\omega_{n,t}, \tag{29}$$

where $a = y_{n,t}$ and $b = y_{n,t} + r_n$. Simplifying the expression by removing terms with only $y_{n,t}$ dependence and setting $\kappa_{n,t} = \frac{y_{n,t}-r_n}{2}$,

$$\prod_{n,t} \frac{(e^{f_{n,t}})^{y_{n,t}}}{(1+e^{f_{n,t}})^{y_{n,t}+r_n}} \propto \prod_{n,t} \exp(-r_n \log 2 + \kappa_{n,t} f_{n,t}) \int_0^\infty \exp\Big\{ -\frac{\omega_{n,t} f_{n,t}^2}{2} \Big\} \mathrm{PG}(\omega_{n,t}|b,0) d\omega_{n,t}. \tag{30}$$

Treating the newly introduced variables as latent auxiliary variables in the model, we identify the joint conditional distribution $p(\boldsymbol{Y}, \{\tau_{n,t}, \xi_{n,t}, \omega_{n,t}\}_{n=1\ldots N, t=1\ldots T} | \boldsymbol{X}, \boldsymbol{W}, \boldsymbol{\beta}, \{r_n\})$. For simplicity, we omit the conditioning variables.

Gathering the above results, and augmenting the variables into the model,

$$\begin{aligned} p(\boldsymbol{Y}, \{\tau_{n,t}, \xi_{n,t}, \omega_{n,t}\}_{n=1\ldots N, t=1\ldots T}|\cdot) \propto & \left(\prod_{n,t} \tau_{n,t}^{(y_{n,t}+r_n)-1} e^{-\tau_{n,t}} \right) \times \left(\prod_{n,t} r_n e^{-r_n^2 \xi_{n,t} + \gamma r_n} \text{ P-IG}(\xi_{n,t}|0) \right) \\ & \times \prod_{n,t} \exp(-r_n \log 2 + \kappa_{n,t} f_{n,t}) \exp\Big\{ -\frac{\omega_{n,t} f_{n,t}^2}{2} \Big\} \mathrm{PG}(\omega_{n,t}). \end{aligned} \tag{31}$$

Conditioning on the augmented variables, we can note the augmented likelihood becomes *conditionally conjugate*. By placing Gaussian priors on $\boldsymbol{X}, \boldsymbol{W}$, and $\boldsymbol{\beta}$, their conditional posterior is a Gaussian density up to a constant factor. Similarly, placing a gamma prior on r_n, $p(r_n) \sim \Gamma(a_0, b_0)$ yields an exponential conditional posterior. Similar to (He et al., 2019), we use an improper gamma prior $\Gamma(1, 0)$.

As a function of $f_{n,t}$,

$$\begin{aligned} p_{f_{n,t}}(\boldsymbol{Y}|\{\tau_{n,t}, \xi_{n,t}\omega_{n,t}\}_{n=1\ldots N, t=1\ldots T}, \cdot) &\propto \prod_{n,t} \exp(\kappa_{n,t} f_{n,t} - \frac{\omega_{n,t} f_{n,t}^2}{2}) \\ &\propto \prod_{n,t} \exp(-\frac{1}{2}\omega_{n,t}(\frac{\kappa_{n,t}}{\omega_{n,t}} - f_{n,t})^2) \\ &\propto \mathcal{N}(z_{n,t}|f_{n,t}, \omega_{n,t}^{-1}). \qquad (z_{n,t} = \tfrac{\kappa_{n,t}}{\omega_{n,t}}) \end{aligned}$$

And as a function of r_n

$$p_{r_n}(\boldsymbol{Y}|\{\tau_{n,t}, \xi_{n,t}\omega_{n,t}\}_{n=1\ldots N, t=1\ldots T}, \cdot) \propto \exp\{\sum_t r_n \log \tau_{n,t} + \log r_n - r_n^2 \xi_{n,t} + \gamma r_n - r_n \log 2 - \frac{1}{2} r_n f_{n,t}\}.$$

This result lays the foundation for closed form variational updates in Appendix C. Unlike the original negative binomial likelihood, this augmented likelihood ensures conjugacy to all our priors and is fundamental in deriving closed form updates for all our variables, including the newly augmented ones.

B.2. Binomial GPFA

As mentioned in the main paper, similar augmentation can be applied to a GPFA model with Binomial observations. Consider the following binomial model, where k_n represents the total number of Bernoulli trials and p is the probability of success, linked to the latent function $f_{n,t}$ via log-odds.

$$\begin{aligned} p(y_{n,t}|\cdot) &\propto p^{y_{n,t}}(1-p)^{k_n - y_{n,t}} \\ &\propto \frac{(e^{f_n,t})^{y_{n,t}}}{(1+e^{f_n,t})^{k_n}}. \end{aligned} \tag{32}$$

By setting $a = y_{n,t}$ and $b = k_n$ and applying the integral identity in Equation (29), we can restore conjugacy to the model. In addition, the common choice of the total number of trials k_n is the maximum number of spikes for neuron n over the length of the recording. Therefore, the value would be fixed and removes the necessity for further augmentation, unlike the negative binomial GPFA model.

B.3. Repeated trials

In the following, we extend the problem setup from analysing single trial data to repeated trial data. Let $\{\boldsymbol{Y}^m\}_{m=1\ldots M}$ denote the spike counts from M repeated trials. Extending Eq. 2 to this setup, our likelihood becomes

$$p(\{\boldsymbol{Y}\}_m|\cdot) \propto \Big(\prod_{m,n,t} \underbrace{\Gamma(y_{m,n,t}+r_n)}_{1}\Big)\Big(\prod_{m,n,t} \underbrace{\frac{1}{\Gamma(r_n)}}_{2}\Big)\Big(\prod_{n,t} \underbrace{\frac{(e^{f_{n,t}})^{\sum_m y_{m,n,t}}}{(1+e^{f_{n,t}})^{\sum_m y_{m,n,t}+Mr_n}}}_{3}\Big). \tag{33}$$

Note that we model the observation across trials as arising from shared set of latents and loading weights.

By setting $a = \sum_m y_{m,n,t}$ and $b = \sum_m y_{m,n,t} + Mr_n$, we can apply the integral identity in Eq. 29. This is equivalent to augmenting $N \times T$ variables. We further augment the model for each product term in 1 and 2 following Eqs. 26 and 28. After augmentation, the subsequent derivations simply follow.

Prior works handle repeated trials differently. (Yu et al., 2008; Jensen et al., 2021) learned shared parameters such as GP lengthscales and weights and allowing for varying latents. (Keeley et al., 2020b) , similar to ours, model shared set of latents and weights to model the common structure across trials, which is beneficial in analyzing repeated trials as shown in the experimental results.

C. Variational distribution updates

Using the mean field assumption provided in Eq. (14), we derive the optimal distributions, denoted with $*$ superscript, for the variables using coordinate ascent variational inference (Bishop, 2006). We represent the set of all variables including the augmented variables with $\boldsymbol{\Theta}$.

C.1. Augmented variables

For augmented gamma variables, $\tau_{n,t}$

$$\begin{aligned} q^*(\tau_{n,t}) &\propto \exp\{\mathbb{E}_{q(-\tau_{n,t})}[\log p(\boldsymbol{Y}, \boldsymbol{\Theta})]\} \\ &\propto \exp\{\mathbb{E}[((y_{n,t}+r_n)-1)\log\tau_{n,t} - \tau_{n,t}]\} && \text{(removing constant terms)} \\ &\propto \exp\{((y_{n,t}+\mathbb{E}[r_n])-1)\log\tau_{n,t} - \tau_{n,t}\}. && \text{(expectation;)} \end{aligned}$$

Recognizing the above as gamma distribution with natural parameters $[(y_{n,t}+\mathbb{E}[r_n])-1, -1]$, the optimal distribution is given by

$$q^*(\tau_{n,t}) = \Gamma(y_{n,t}+\mathbb{E}[r_n], 1). \tag{34}$$

For augmented P-IG variables, $\xi_{n,t}$

$$\begin{aligned} q^*(\xi_{n,t}) &\propto \exp\{\mathbb{E}_{q(-\xi_{n,t})}[\log p(Y, \boldsymbol{\Theta})]\} \\ &\propto p(\xi_{n,t}; 1)\exp\{-\mathbb{E}[r_n^2]\xi_{n,t}\}. && \text{(removing constant terms)} \end{aligned}$$

Using the exponential tilting property (He et al., 2019), we can recognize this as a general class P-IG distribution,

$$q^*(\xi_{n,t}) = \text{P-IG}(\sqrt{\mathbb{E}[r_n^2]}). \tag{35}$$

For the set of augmented PG variables $\omega_{n,t}$ we can derive the optimal variational distributions as follows,

$$
\begin{aligned}
q^*(\omega_{n,t}) &\propto \exp\{\mathbb{E}_{q(-\omega_{n,t})}[\log p(Y, \boldsymbol{\Theta})]\} \\
&\propto \exp\left\{ \mathbb{E}\left[\log \frac{(e^{f_{n,t}})^{y_{n,t}}}{(1+e^{f_{n,t}})^{y_{n,t}+r_n}} \right] \right\} && \text{(removing terms with no } \omega_{n,t} \text{ dependence)} \\
&\propto \exp\left\{ \mathbb{E}\left[y_{n,t}\log(e^{f_{n,t}}) - (y_{n,t}+r_n)\log(1+e^{f_{n,t})} \right] \right\} && \text{(removing terms with no } \omega_{n,t} \text{ dependence)} \\
&\propto \exp\left\{ \mathbb{E}\left[y_{n,t}\log(e^{f_{n,t}}) - (y_{n,t}+\mathbb{E}[r_n])\log(1+e^{f_{n,t})} \right] \right\} && \text{(applying expectation w.r.t } r_n) \\
&\propto \exp\left\{ \mathbb{E}\left[\log \frac{(e^{f_{n,t}})^{y_{n,t}}}{(1+e^{f_{n,t}})^{y_{n,t}+\mathbb{E}[r_n]}} \right] \right\} && \text{(exponent in log; quotient rule)} \\
&\propto \exp\left\{ \mathbb{E}\left[\log \exp\{-\frac{\omega_{n,t} f_{n,t}^2}{2}\} \, \mathrm{PG}(\omega_{n,t} | y_{n,t}+\mathbb{E}[r_n], 0) \right] \right\} \\
& && \text{(applying the Pólya-gamma integral identity; dropping constants)} \\
&\propto \mathrm{PG}(\omega_{n,t} | y_{n,t}+\mathbb{E}[r_n], 0) \exp\left\{ \mathbb{E}[-\frac{\omega_{n,t} f_{n,t}^2}{2}] \right\} && \text{(simplifying expression)} \\
&\propto \mathrm{PG}(\omega_{n,t} | y_{n,t}+\mathbb{E}[r_n], \sqrt{\mathbb{E}[f_{n,t}^2]}). && \text{(exponential tilting)}
\end{aligned}
$$

Note all augmented variables $\{\omega_{n,t}\}$ do not have interdependence in the updates, hence the updates can be done in parallel fashion. The same holds for $\{\tau_{n,t}, \xi_{n,t}\}$.

For the repeated trial setting, B.3, the updates would simply differ in the shape parameter $PG(\omega_{n,t} | \sum_m y_{m,n,t} + E[r_n], \sqrt{E[f_{n,t}^2]})$

C.2. Dispersion variables

$$
\begin{aligned}
q^*(r_n) &\propto \exp\{\mathbb{E}_{q(-r_n)}[\log p(\boldsymbol{Y}, \boldsymbol{\Theta})]\} \\
&\propto \exp\{\mathbb{E}[\log p(\boldsymbol{Y}|\cdot) p(r_n)]\} \\
&\propto p(r_n) \exp\{\mathbb{E}[\log p(\boldsymbol{Y}|\cdot)]\} \\
&\propto p(r_n) \exp\{\mathbb{E}[\log \prod_t \tau_{n,t}^{(y_{n,t}+r_n)-1} r_n \exp(-r_n^2 \xi_{n,t} + \gamma r_n - r_n \log 2 + \kappa_{n,t} f_{n,t})]\} \\
& && \text{(expanding expression by droping terms with no } r_n \text{ dependence)} \\
&\propto p(r_n) \exp\{\mathbb{E}[\sum_t r_n \log \tau_{n,t} + \log r_n - r_n^2 \xi_{n,t} + \gamma r_n - r_n \log 2 - \frac{1}{2} r_n f_{n,t}]\} \\
& && \text{(product to summation; } \kappa_{n,t} = \tfrac{y_{n,t}-r_n}{2}) \\
&\propto p(r_n) \exp\{\sum_t r_n \, \mathbb{E}[\log \tau_{n,t}] + \log r_n - r_n^2 \, \mathbb{E}[\xi_{n,t}] + \gamma r_n - r_n \log 2 - \frac{1}{2} r_n \, \mathbb{E}[f_{n,t}]\} && \text{(applying expectation)} \\
&\propto p(r_n) r_n^T \exp\{\sum_t r_n \, \mathbb{E}[\log \tau_{n,t}] - r_n^2 \, \mathbb{E}[\xi_{n,t}] + \gamma r_n - r_n \log 2 - \frac{1}{2} r_n \, \mathbb{E}[f_{n,t}]\} \\
&\propto r_n^{T-1} \exp\{-r_n^2 \Big(\sum_t \mathbb{E}[\xi_{n,t}] \Big) + r_n \sum_t \Big(\mathbb{E}[\log \tau_{n,t}] + \gamma - \log 2 - \frac{1}{2} \, \mathbb{E}[f_{n,t}] \Big) \} \\
& && (p(r_n) \propto 1/r \text{; simplification}) \\
& && (36)
\end{aligned}
$$

We can recognize the above expression as a Power Truncated Normal (PTN) distribution (He et al., 2019) with parameters

$$p = T, \tag{37}$$

$$a = \sum_t \mathbb{E}[\xi_{n,t}] \;\; \& \;\; b = \sum_t \left(\mathbb{E}[\log \tau_{n,t}] + \gamma - \log 2 - \frac{1}{2}\,\mathbb{E}[f_{n,t}] \right). \tag{38}$$

The expectation in $E[\xi_{n,t}]$ is given closed form of the moment of a P-IG distribution with tilting parameter c (Theorem 2 in (He et al., 2019) for details)

$$\mathbb{E}[\xi_{n,t}] = \frac{1}{2c}\psi(c+1) - \psi(1). \tag{39}$$

Also by logarithmic expectation of gamma distribution, we can simplify,

$$\mathbb{E}[\log \tau_{n,t}] = \psi(\alpha) - \log \beta. \tag{40}$$

where α, β are shape and rate parameters of the variational distribution and ψ denotes the digamma function.

Repeated trials setting For M repeated trials,

$$p = MT, \tag{41}$$

$$a = \sum_{m,n,t} \mathbb{E}[\xi_{m,n,t}] \;\; \& \;\; b = \sum_{m,n,t} \left(\mathbb{E}[\log \tau_{m,n,t}] + \gamma - \log 2 - \frac{1}{2}\,\mathbb{E}[f_{n,t}] \right) \tag{42}$$

where $\{\xi_{m,n,t}, \tau_{m,n,t}\}$ are augmented variables.

C.3. Latent processes

$$
\begin{aligned}
q^*(\boldsymbol{X}_d) &\propto p(\boldsymbol{X}_d)\exp\{\mathbb{E}_{q(-\boldsymbol{X}_d)}[\log p(\boldsymbol{Y}|\Theta)]\} \\
&\propto p(\boldsymbol{X}_d)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n(\boldsymbol{z}_n-\boldsymbol{f}_n)\boldsymbol{\Omega}_n(\boldsymbol{z}_n-\boldsymbol{f}_n)^\top\right]\} \\
&\propto p(\boldsymbol{X}_d)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n -2\boldsymbol{f}_n\boldsymbol{\Omega}_n\boldsymbol{z}_n^\top+\boldsymbol{f}_n\boldsymbol{\Omega}_n\boldsymbol{f}_n^\top\right]\} \qquad \text{(distribute and remove constants w.r.t } \boldsymbol{X}_d\text{)} \\
&\propto p(\boldsymbol{X}_d)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n \boldsymbol{w}_n\boldsymbol{X}\boldsymbol{\Omega}_n\boldsymbol{X}^\top\boldsymbol{w}_n^\top-2\boldsymbol{w}_n\boldsymbol{X}\boldsymbol{\Omega}_n(\boldsymbol{z}_n^\top-\boldsymbol{\beta}_n\mathbf{1})\right]\} \qquad \text{(removing constants w.r.t } \boldsymbol{X}_d\text{)} \\
&\propto p(\boldsymbol{X}_d)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n\left(\sum_{d'}\boldsymbol{w}_{n,d}\boldsymbol{X}_d\boldsymbol{\Omega}_n\boldsymbol{X}_{d'}^\top\boldsymbol{w}_{n,d'}\right)-2\boldsymbol{w}_{n,d}\boldsymbol{X}_d\boldsymbol{\Omega}_n(\boldsymbol{z}_n^\top-\boldsymbol{\beta}_n\mathbf{1})\right]\} \\
&\qquad\qquad \text{(decomposing summations; removing constants)} \\
&\propto p(\boldsymbol{X}_d)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n \boldsymbol{X}_d(\boldsymbol{\Omega}_n\boldsymbol{w}_{n,d}^2)\boldsymbol{X}_d^\top-2\boldsymbol{X}_d\boldsymbol{w}_{n,d}\boldsymbol{\Omega}_n(\boldsymbol{z}_n^\top-\boldsymbol{\beta}_n\mathbf{1}-\sum_{d'}\boldsymbol{X}_{d'}\boldsymbol{w}_{n,d'})\right]\} \quad \text{(rearranging)} \\
&\propto p(\boldsymbol{X}_d)\exp\{-\frac{1}{2}\Bigg(\boldsymbol{X}_d(\sum_n E[\boldsymbol{\Omega}_n]\,\mathbb{E}[\boldsymbol{w}_{n,d}^2])\boldsymbol{X}_d^\top \\
&\quad -2\boldsymbol{X}_d\Bigg(\sum_n\mathbb{E}[\boldsymbol{w}_{n,d}]\,\mathbb{E}[\boldsymbol{\Omega}_n](\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{\beta}_n]\mathbf{1}-\sum_{d'\neq d}\mathbb{E}[\boldsymbol{X}_{d'}]\,\mathbb{E}[\boldsymbol{w}_{n,d'})]\Bigg)\Bigg)\} \\
&\qquad\qquad \text{(applying summations along } n\text{; applying expectations)} \\
&\propto \exp\{-\frac{1}{2}\Bigg(\boldsymbol{X}_d\underbrace{(\boldsymbol{K}_d^{-1}+\sum_n\mathbb{E}[\boldsymbol{\Omega}_n]\,\mathbb{E}[\boldsymbol{w}_{n,d}^2])}_{:=\boldsymbol{\Phi}_d}\boldsymbol{X}_d^\top \\
&\quad -2\boldsymbol{X}_d\Bigg(\sum_n\mathbb{E}[\boldsymbol{w}_{n,d}]\,\mathbb{E}[\boldsymbol{\Omega}_n](\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{\beta}_n]\mathbf{1}-\sum_{d'\neq d}\mathbb{E}[\boldsymbol{X}_{d'}]\,\mathbb{E}[\boldsymbol{w}_{n,d'})]\Bigg)\Bigg)\} \qquad \text{(definition of } p(\boldsymbol{X}_d)\text{ 9)} \\
&\propto \exp\Bigg\{-\frac{1}{2}\Bigg(\boldsymbol{X}_d-\Bigg(\boldsymbol{\Phi}_d^{-1}\Bigg(\sum_n\mathbb{E}[\boldsymbol{w}_{n,d}]\,\mathbb{E}[\boldsymbol{\Omega}_n](\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{\beta}_n]\mathbf{1}-\sum_{d'\neq d}\mathbb{E}[\boldsymbol{X}_{d'}]\,\mathbb{E}[\boldsymbol{w}_{n,d'})]\Bigg)\Bigg)^\top\Bigg)\boldsymbol{\Phi}_d \\
&\qquad \Bigg(\boldsymbol{X}_d-\Bigg(\boldsymbol{\Phi}_d^{-1}\Bigg(\sum_n\mathbb{E}[\boldsymbol{w}_{n,d}]\,\mathbb{E}[\boldsymbol{\Omega}_n](\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{\beta}_n]\mathbf{1}-\sum_{d'\neq d}\mathbb{E}[\boldsymbol{X}_{d'}]\,\mathbb{E}[\boldsymbol{w}_{n,d'})]\Bigg)\Bigg)^\top\Bigg)^\top\Bigg\} \\
&\qquad\qquad \text{(removing constant; applying expectations; completing the square;)}
\end{aligned}
$$

We can recognize the above expression as a multivariate Gaussian distribution over a random vector $\boldsymbol{X}_d$ with mean and variance $\boldsymbol{m}_d$ and variance $\boldsymbol{V}_d$

$$q^*(\boldsymbol{X}_d)=\mathcal{N}(\boldsymbol{m}_d,\boldsymbol{V}_d)$$

$$\boldsymbol{m}_d=\boldsymbol{V}_d\Bigg(\sum_n\mathbb{E}[\boldsymbol{w}_{n,d}]\,\mathbb{E}[\boldsymbol{\Omega}_n](\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{\beta}_n]\mathbf{1}-\sum_{d'\neq d}\mathbb{E}[\boldsymbol{X}_{d'}]\,\mathbb{E}[\boldsymbol{w}_{n,d'})]\Bigg) \tag{43}$$

$$\boldsymbol{V}_d=\boldsymbol{\Phi}_n^{-1} \tag{44}$$

C.4. Base intensities

$$
\begin{aligned}
q^*(\boldsymbol{\beta}) &\propto p(\boldsymbol{\beta}|\tau_\beta)\exp\{\mathbb{E}_{q(-\boldsymbol{\beta}}[\log p(\boldsymbol{Y}|\boldsymbol{\Theta})]\} \\
&\propto p(\boldsymbol{\beta}|\tau_\beta)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n (\boldsymbol{z}_n-\boldsymbol{f}_n)\boldsymbol{\Omega}_n(\boldsymbol{z}_n-\boldsymbol{f}_n)^\top\right]\} \\
&\propto p(\boldsymbol{\beta}|\tau_\beta)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n -2\boldsymbol{f}_n\boldsymbol{\Omega}_n\boldsymbol{z}_n^\top+\boldsymbol{f}_n\boldsymbol{\Omega}_n\boldsymbol{f}_n^\top\right]\} && \text{(distribute and remove constants w.r.t } \boldsymbol{\beta}_n\text{)} \\
&\propto p(\boldsymbol{\beta}|\tau_\beta)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n \boldsymbol{\beta}_n\mathbf{1}^\top\boldsymbol{\Omega}_n\mathbf{1}\boldsymbol{\beta}_n-2\boldsymbol{\beta}_n\mathbf{1}^\top\boldsymbol{\Omega}_n(\boldsymbol{z}_n^\top-\boldsymbol{X}^T\boldsymbol{w}_n^T)\right]\} && \text{(removing constants w.r.t } \boldsymbol{\beta}_n\text{)} \\
&\propto p(\boldsymbol{\beta}|\tau_\beta)\exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n \boldsymbol{\beta}_n^2\operatorname{Tr}(\boldsymbol{\Omega}_n)-2\boldsymbol{\beta}_n\operatorname{Tr}(\boldsymbol{\Omega}_n\operatorname{diag}(\boldsymbol{z}_n^\top-\boldsymbol{X}^T\boldsymbol{w}_n^T))\right]\} && \text{(simplificiation)} \\
&\propto \exp\{-\frac{1}{2}\sum_n \boldsymbol{\beta}_n^2\underbrace{(\mathbb{E}[\tau_\beta]+\operatorname{Tr}(\mathbb{E}[\boldsymbol{\Omega}_n]))}_{:=\boldsymbol{\Phi}_n}-2\boldsymbol{\beta}_n\operatorname{Tr}(\mathbb{E}[\boldsymbol{\Omega}_n]\operatorname{diag}\left(\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{X}^T]\,\mathbb{E}[\boldsymbol{w}_n^T]\right))\} \\
& && \text{(definition of } p(\boldsymbol{\beta}|\tau_\beta)\text{; applying expectation)} \\
&\propto \exp\{-\frac{1}{2}\sum_n \boldsymbol{\Phi}_n\left(\boldsymbol{\beta}_n-\frac{\operatorname{Tr}(\mathbb{E}[\boldsymbol{\Omega}_n]\operatorname{diag}\left(\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{X}^T]\,\mathbb{E}[\boldsymbol{w}_n^T]\right))}{\boldsymbol{\Phi}_n}\right)^2\} \\
& && \text{(completing the square; removing constants)}
\end{aligned}
$$

We can recognize the above expression as a product of univariate Gaussian distributions with mean and covariance denoted μ_n and σ_n^2.

$$
\begin{aligned}
q^*(\boldsymbol{\beta}) &= \prod_n \mathcal{N}(\mu_n,\sigma_d^2) \\
\mu_n = \sigma_n^2\operatorname{Tr}(\mathbb{E}[\boldsymbol{\Omega}_n]\operatorname{diag}\left(\boldsymbol{z}_n^\top-\mathbb{E}[\boldsymbol{X}^T]\,\mathbb{E}[\boldsymbol{w}_n^T]\right)) \;\;\&\;\; \sigma_n^2 &= \frac{1}{\mathbb{E}[\tau_\beta]+\operatorname{Tr}(\mathbb{E}[\boldsymbol{\Omega}_n])} \qquad (45)
\end{aligned}
$$

C.5. Loading weights

Recall $\boldsymbol{f}_n = \boldsymbol{w}_n\,\boldsymbol{X}_d + \boldsymbol{\beta}_n\mathbf{1}^\top$

$$
\begin{aligned}
q^*(\boldsymbol{W}) &\propto p(\boldsymbol{W}) \exp\{\mathbb{E}_{q(-\boldsymbol{W})}[\log p(\boldsymbol{Y}|\boldsymbol{\Theta})]\} \\
&\propto p(\boldsymbol{W}) \exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n (\boldsymbol{z}_n - \boldsymbol{f}_n)\boldsymbol{\Omega}_n(\boldsymbol{z}_n - \boldsymbol{f}_n)^\top\right]\} \\
&\propto p(\boldsymbol{W}) \exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n -2\boldsymbol{f}_n\boldsymbol{\Omega}_n\boldsymbol{z}_n^\top + \boldsymbol{f}_n\boldsymbol{\Omega}_n\boldsymbol{f}_n^\top\right]\} && \text{(distribute and remove constants w.r.t } \boldsymbol{W}) \\
&\propto \exp\{-\frac{1}{2}\boldsymbol{w}_n(\mathrm{diag}(\boldsymbol{\tau}))\boldsymbol{w}_n^\top\} \exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n \boldsymbol{w}_n\boldsymbol{X}\boldsymbol{\Omega}_n\boldsymbol{X}^\top\boldsymbol{w}_n^\top - 2\boldsymbol{w}_n\boldsymbol{X}\boldsymbol{\Omega}_n\boldsymbol{z}_n^\top\right]\} \\
& && \text{(removing constants w.r.t } \boldsymbol{W}\text{; } \hat{\boldsymbol{z}}_n^\top = \boldsymbol{z}_n^\top - \boldsymbol{\beta}_n\mathbf{1}) \\
&\propto \exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n \boldsymbol{w}_n(\boldsymbol{X}\boldsymbol{\Omega}_n\boldsymbol{X}^\top + \mathrm{diag}(\boldsymbol{\tau}))\boldsymbol{w}_n^\top - 2\boldsymbol{w}_n\boldsymbol{X}\boldsymbol{\Omega}_n\hat{\boldsymbol{z}}_n^\top\right]\} && \text{(removing constants w.r.t } \boldsymbol{W}) \\
&\propto \exp\{\mathbb{E}\left[-\frac{1}{2}\sum_n \boldsymbol{w}_n\boldsymbol{\Phi}_n\boldsymbol{w}_n^\top - 2\boldsymbol{w}_n\boldsymbol{X}\boldsymbol{\Omega}_n\hat{\boldsymbol{z}}_n^\top + \hat{\boldsymbol{z}}_n\boldsymbol{\Omega}^{-1}\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{\Omega}_n\hat{\boldsymbol{z}}_n^\top - \hat{\boldsymbol{z}}_n\boldsymbol{\Omega}^{-1}\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{\Omega}_n\hat{\boldsymbol{z}}_n^\top\right]\} \\
& && (\boldsymbol{\Phi}_n = (\boldsymbol{X}\boldsymbol{\Omega}_n\boldsymbol{X}^\top + \mathrm{diag}(\boldsymbol{\tau}))\text{; completing the square}) \\
&\propto \exp\{\Bigg[-\frac{1}{2}\sum_n \left(\boldsymbol{w}_n - \left((E[\boldsymbol{\Phi}_n])^{-1}\,\mathbb{E}[\boldsymbol{X}]\,\mathbb{E}[\boldsymbol{\Omega}_n\hat{\boldsymbol{z}}_n^\top]\right)^\top\right)\mathbb{E}[\boldsymbol{\Phi}_n] \\
&\qquad \left(\boldsymbol{w}_n - \left((E[\boldsymbol{\Phi}_n])^{-1}\,\mathbb{E}[\boldsymbol{X}]\,\mathbb{E}[\boldsymbol{\Omega}_n\hat{\boldsymbol{z}}_n^\top]\right)^\top\right)^\top\Bigg]\} \\
& && \text{(removing constant; applying expectations; completing the square;)}
\end{aligned}
$$

We can recognize the above expression as product of multivariate Gaussian distribution of random vectors $\boldsymbol{w}_n$ with mean and variance $\boldsymbol{m}_n$ and variance $\boldsymbol{V}_n$

$$
\begin{aligned}
q^*(\boldsymbol{W}) &= \prod_n \mathcal{N}(\boldsymbol{m}_n, \boldsymbol{V}_n) \\
\boldsymbol{m}_n = \boldsymbol{V}_n(\mathbb{E}[\boldsymbol{X}]\,\mathbb{E}[\boldsymbol{\Omega}_n\hat{\boldsymbol{z}}_n^\top]) \;\;\&\;\; \boldsymbol{V}_n &= (\mathbb{E}[\boldsymbol{\Phi}_n])^{-1} \qquad (46)
\end{aligned}
$$

C.6. Precision Variables

For the ARD precision variables,

$$
\begin{aligned}
q^*(\boldsymbol{\tau}) &\propto \exp\{\mathbb{E}_{q(-\boldsymbol{\tau})}[\log p(\boldsymbol{Y}, \boldsymbol{\Theta})]\} \\
&\propto p(\boldsymbol{\tau}) \exp\{\mathbb{E}\left[\log p(\boldsymbol{W}|\boldsymbol{\tau})\right]\} && \text{(no } \boldsymbol{\tau} \text{ dependence)} \\
&= p(\boldsymbol{\tau}) \exp\{\mathbb{E}\left[\sum_n \frac{1}{2}\log|\,\mathrm{diag}(\boldsymbol{\tau})| - \frac{1}{2}\boldsymbol{w}_n\,\mathrm{diag}(\boldsymbol{\tau})\boldsymbol{w}_n^\top\right]\} && \text{(definition of } p(\boldsymbol{W}|\boldsymbol{\tau})) \\
&= p(\boldsymbol{\tau}) \exp\{\mathbb{E}\left[\frac{N}{2}\sum_d \log\tau_d - \frac{1}{2}\sum_n \boldsymbol{w}_n\,\mathrm{diag}(\boldsymbol{\tau})\boldsymbol{w}_n^\top\right]\} && \text{(definition of } p(\boldsymbol{W}|\boldsymbol{\tau})) \\
&= p(\boldsymbol{\tau}) \exp\{\frac{N}{2}\sum_d \log\tau_d - \frac{1}{2}\sum_d\sum_n \mathbb{E}\left[\boldsymbol{w}_{n,d}^2\right]\tau_d + \mathrm{const}\} && \text{(taking expectations, rearranging)} \\
&\propto \exp\{\sum_d (a_d - 1)\log\tau_d - b_d\tau_d\} \exp\{\frac{N}{2}\sum_d \log\tau_d - \frac{1}{2}\sum_d\sum_n \mathbb{E}\left[\boldsymbol{w}_{n,d}^2\right]\tau_d\} && \text{(definition of } p(\tau)) \\
&= \exp\{\sum_d (a_d + \frac{N}{2} - 1)\log\tau_d - (b_d + \frac{1}{2}\sum_d\sum_n \mathbb{E}\left[\boldsymbol{w}_{n,d}^2\right])\tau_d\}. && \text{(rearranging)}
\end{aligned}
$$

Recognizing the above as a product of gamma distributions, we get

$$q^*(\boldsymbol{\tau}) = \prod_d \mathcal{G}(\hat{a}_d, \hat{b}_d) \qquad \text{where}$$
$$\hat{a}_d = a_d + \frac{N}{2} \qquad \hat{b}_d = b_d + \frac{1}{2}\sum_n \mathbb{E}\left[\boldsymbol{w}_{n,d}^2\right]. \tag{47}$$

The update rule for the precision variable, τ_β, is given as

$$\begin{aligned}
q^*(\tau_\beta) &\propto \exp\{\mathbb{E}_{q(-\tau_\beta)}[\log p(\boldsymbol{Y}, \boldsymbol{\Theta})]\} \\
&\propto p(\tau_\beta)\exp\{\mathbb{E}\left[\sum_n \log p(\beta_n|\tau_\beta)\right]\} && \text{(no } \tau_\beta \text{ dependence)} \\
&\propto p(\tau_\beta)\exp\{\mathbb{E}\left[\sum_n \frac{1}{2}\log\tau_\beta - \frac{1}{2}\beta_n^2\tau_\beta\right]\} && \text{(definition of } p(\beta_n|\tau_\beta)) \\
&= p(\tau_\beta)\exp\{\frac{N}{2}\log\tau_\beta - \frac{1}{2}\sum_n \mathbb{E}\left[\beta_n^2\right]\tau_\beta\} && \text{(taking expectations, rearranging)} \\
&= \exp\{(c + \frac{N}{2} - 1)\log\tau_\beta - (d + \frac{1}{2}\sum_n \mathbb{E}\left[\beta_n^2\right])\tau_\beta\}. && \text{(definition of } p(\tau_\beta))
\end{aligned}$$

Recognizing the above as a product of gamma distributions, we get

$$q^*(\boldsymbol{\tau}_\beta) = \mathcal{G}(\hat{c}, \hat{d}) \qquad \text{where}$$
$$\hat{c} = c + \frac{N}{2} \qquad \hat{d} = d + \frac{1}{2}\sum_n \mathbb{E}\left[\beta_n^2\right]. \tag{48}$$

C.7. Computing variational moments

All of the variables except the dispersion parameter r_n have a known closed form for their first and second moments. For r_n, we compute the its moments using an efficient gamma based sampler presented in (He et al., 2019) (see its Appendix C for details). Also for completeness, we review moments and other important facts of the less common Pólya-Gamma, Pólya-Inverse Gamma, and Power Truncated Normal distributions in Appendix F

D. ELBO expression

The evidence lower bound (ELBO), denoted as $\mathcal{L}$ defined in Equation (15) is given as follows

$$\mathcal{L} = \mathbb{E}_{q(\boldsymbol{\Theta})}\left[\log p(\boldsymbol{Y}|\boldsymbol{\Theta})\right] - \mathrm{KL}[q(\boldsymbol{\Theta})\|p(\boldsymbol{\Theta})]$$

where $\boldsymbol{\Theta} = \{\boldsymbol{X}, \boldsymbol{W}, \boldsymbol{\beta}, \boldsymbol{\tau}, \tau_\beta, \{\omega_{n,t}, \tau_{n,t}, \xi_{n,t},\}\}$ is the set of all latent variables and KL represents Kullback-Leibler divergence measure.

This follows from Jensen's inequality,

$$\begin{aligned}
\log p(\boldsymbol{Y}) &= \log \mathbb{E}_{p(\boldsymbol{\Theta})}\left[p(\boldsymbol{Y}|\Theta)\right] \\
&= \log \mathbb{E}_{q(\boldsymbol{\Theta})}\left[\frac{p(\boldsymbol{Y}|\boldsymbol{\Theta})p(\boldsymbol{\Theta})}{q(\boldsymbol{\Theta})}\right] \\
&\geq \mathbb{E}_{q(\boldsymbol{\Theta})}\left[\log\frac{p(\boldsymbol{Y}|\boldsymbol{\Theta})p(\boldsymbol{\Theta})}{q(\boldsymbol{\Theta})}\right] =: \mathcal{L}.
\end{aligned} \tag{49}$$

In the variational EM algorithm presented in Algorithm 1, we update the variational distributions utilizing closed form solutions. However, to update the GP parameters, we directly optimize them (via automatic differentiation) using the ELBO

formula. This requires further simplifications of the ELBO. In the following, we show details of the simplifications.

$$\mathcal{L} = \mathbb{E}_{q(\Theta)}[p(\boldsymbol{Y}|\boldsymbol{\Theta})] - \mathrm{KL}[q(\boldsymbol{\Theta})\|p(\boldsymbol{\Theta})] \tag{50}$$

We note that in the expression, the variational expectation term has no dependence of GP parameters. And by independence assumptions presented in Equation (14),

the above KL term decomposes as

$$\begin{aligned}
&\mathrm{KL}[q(\boldsymbol{\Theta})\|p(\boldsymbol{\Theta})] \\
&\quad= \sum_d \mathrm{KL}[q(\boldsymbol{X}_d\|p(\boldsymbol{X}_d))] + \sum_{n,t} \mathrm{KL}[q(\omega_{n,t}\|p(\omega_{n,t}))] + \sum_{n,t} \mathrm{KL}[q(\tau_{n,t})\|p(\tau_{n,t})] \\
&\qquad+ \sum_{n,t} \mathrm{KL}[q(\xi_{n,t})\|p(\xi_{n,t})] + \sum_n \mathrm{KL}[q(\boldsymbol{w}_n)\|p(\boldsymbol{w}_n|\boldsymbol{\tau})] \\
&\qquad+ \sum_n \mathrm{KL}[q(\beta_n)||p(\beta_n)] + \sum_d \mathrm{KL}[q(\boldsymbol{\tau}_d)\|p(\boldsymbol{\tau}_d)] + \mathrm{KL}[q(\tau_\beta)\|p(\tau_\beta)].
\end{aligned}$$

From the equation above, we note that only the first KL term is dependent on GP parameters. Thus, further simplifying the KL using divergence formula of two multivariate Gaussians distributions,

$$\begin{aligned}
\mathrm{KL}[q(\boldsymbol{X}_d)\|p(\boldsymbol{X}_d)] &= \mathrm{KL}[q(\boldsymbol{X}_d|\hat{\boldsymbol{\mu}}_d, \hat{\boldsymbol{K}}_d)\|p(\boldsymbol{X}_d|\boldsymbol{\mu}_d, \boldsymbol{K}_d)] \\
&= \frac{1}{2}\left[\log|\boldsymbol{K}_d| - \log|\hat{\boldsymbol{K}}_d| - T + \mathrm{Tr}\{\boldsymbol{K}_d^{-1}\hat{\boldsymbol{K}}_d\} + (\boldsymbol{\mu}_d - \hat{\boldsymbol{\mu}}_d)^\top \boldsymbol{K}_d^{-1}(\boldsymbol{\mu}_d - \hat{\boldsymbol{\mu}}_d)\right].
\end{aligned}$$

We can now define an objective function to optimize our GP parameters $\{\theta_d\}_{d=1}^D$.

$$\begin{aligned}
\mathcal{L}(\theta_d) &= -\mathrm{KL}[q(\boldsymbol{X}_d)\|p(\boldsymbol{X}_d)] + \text{const} && \text{(removing constants w.r.t } \theta_d\text{)} \\
&= \mathbb{E}_{q(\boldsymbol{X}_d)}[\log q(\boldsymbol{X}_d)] + \mathbb{E}_{q(\boldsymbol{X}_d)}[\log p(\boldsymbol{X}_d)] + \text{const} && \text{(KL def.)} \\
&= \mathbb{E}_{q(\boldsymbol{X}_d)}[\log p(\boldsymbol{X}_d)] + \text{const} && \text{(removing constants w.r.t } \theta_d\text{)} \\
&= -\frac{1}{2}\left(\log|\boldsymbol{K}_d| + \mathbb{E}_{q(\boldsymbol{X}_d)}[\boldsymbol{X}_d]\boldsymbol{K}_d^{-1}\,\mathbb{E}_{q(\boldsymbol{X}_d)}[\boldsymbol{X}_d^\top] + \mathrm{Tr}(\hat{\boldsymbol{K}}_d\boldsymbol{K}_d)\right) + \text{const}\,. && (51)
\end{aligned}$$

Recall $\boldsymbol{K}_d$ denotes the prior covariance of the d-th latent process with kernel parameters θ_d. In practice, we use PyTorch (Paszke et al., 2019) automatic differentiation engine to compute gradients with respect to the GP length scale parameters $\{\theta_d\}$. For a sparse GPFA variant Section 3.3, $\boldsymbol{K}_d$ would be an $M \times M$ prior covariance matrix instead of a large $T \times T$.

E. Natural Gradients

The sparse GPFA variant shown in Section 3.3 avoids the expensive $T \times T$ matrix. However, still in the variational updates it uses all the data i.e. spike counts of all neurons across time. Here we show how we can extend the accelerate the inference by using only a mini-batch of the data. Using mini-batch of the data we obtain a noisy natural gradient (Hoffman et al., 2013).

Consider a random minibatch of time points $\mathcal{T}$. The natural gradients based on these time points are give below. These gradients are scaled to match the dataset size using the ratio of the total number of time steps to the minibatch size $\frac{T}{|\mathcal{T}|}$.

E.1. Weights

The natural parameters for the loading weights of the n-th neuron are

$$\boldsymbol{\eta}_1^{(n)} = \frac{T}{|\mathcal{T}|}\,\mathbb{E}[\boldsymbol{X}]\,\mathbb{E}[\boldsymbol{\Omega}_n](\boldsymbol{z}_n^\top - \mathbb{E}[\beta_n]\mathbf{1}]) \quad \boldsymbol{\eta}_2^{(n)} = -\frac{1}{2}(\mathrm{diag}(\mathbb{E}[\boldsymbol{\tau}]) + \frac{T}{|\mathcal{T}|}\,\mathbb{E}[\boldsymbol{X}\,\mathbb{E}[\boldsymbol{\Omega}_n]\boldsymbol{X}^\top]) \tag{52}$$

E.2. Latent Processes

The natural parameters for the d-th latent process are

$$\boldsymbol{\eta}_1^{(d)} = \frac{T}{|\mathcal{T}|}\left(\sum_n \mathbb{E}[\boldsymbol{\Omega}_n]\,\mathbb{E}[\boldsymbol{w}_{n,d}]\big(\mathbb{E}[\boldsymbol{z}_n^\top] - \mathbb{E}[\beta_n]\mathbf{1} - \sum_{d'\neq d}\mathbb{E}[\boldsymbol{w}_{n,d'}]\,\mathbb{E}[\boldsymbol{X}_{d'}^\top]\big)\right)$$
$$\boldsymbol{\eta}_2^{(d)} = -\frac{1}{2}\left(\boldsymbol{K}_d^{-1} + \frac{T}{|\mathcal{T}|}\sum_n \mathbb{E}[w_{n,d}^2]\,\mathbb{E}[\boldsymbol{\Omega}_n]\right) \tag{53}$$

E.3. Base Intensities

The natural parameters for the base intensity of the n neuron are given by

$$\boldsymbol{\eta}_1^{(n)} = \frac{T}{|\mathcal{T}|}(\mathbb{E}[\boldsymbol{z}_n] - \mathbb{E}[\boldsymbol{w}_n]\,\mathbb{E}[\boldsymbol{X}])\,\mathbb{E}[\boldsymbol{\Omega}_n]\mathbf{1} \quad \boldsymbol{\eta}_2^{(n)} = -\frac{1}{2}(\mathbb{E}[\tau_\beta] + \frac{T}{|\mathcal{T}|}\operatorname{Tr}(\mathbb{E}[\boldsymbol{\Omega}_n]) \tag{54}$$

F. Important Densities

Glynn et al. (2019) and He et al. (2019) formally defined new distributions Pólya-Inverse Gamma(P-IG) and Power Truncated Normal distributions resepectively. For completeness we include their important details in this section. We also review Pólya-Gamma distribution.

F.1. Pólya-Inverse Gamma (P-IG)

The Pólya-Inverse Gamma (P-IG) distribution is defined with an infinite dimensional parameter vector $\boldsymbol{d} = \{d_1, d_2, ...\}$ and a scalar "tilting" parameter c. The distribution of a P-IG random variable x with parameter $\boldsymbol{d}$ and $c = 0$ is equivalent to an infinite convolution of the well known generalized inverse Gaussian (GIG) distribution,

$$\text{P-IG}(x|\boldsymbol{d}, c=0) \overset{D}{=} \sum_{k=1}^{\infty} GIG\left(-\frac{3}{2}, \frac{1}{\sqrt{2}d_k}, 0\right) \tag{55}$$

The general class of P-IG($\boldsymbol{d}, 0$) is defined as exponential tilting of P-IG($\boldsymbol{d}, c$), similar to Pólya Gamma distribution (Polson et al., 2013),

$$\text{P-IG}(x|\boldsymbol{d}, c) \propto \exp(-\frac{c}{2}x)\,\text{P-IG}(x|\boldsymbol{d}, 0), \tag{56}$$

and equivalently,

$$\text{P-IG}(x|\boldsymbol{d}, c) \overset{D}{=} \sum_{k=1}^{\infty} GIG\left(-\frac{3}{2}, 2c^2, \frac{1}{2k^2}\right). \tag{57}$$

Theorem 2 in (He et al., 2019) shows the first moment of the distributions available in closed form,

$$\mathbb{E}[x] = \frac{1}{2c}(\psi(c+1) - \psi(1)). \tag{58}$$

F.2. Power truncated normal (PTN)

The Power Truncated Normal distribution, PTN, is defined with three parameters $p, a > 0$ and $b \neq 0$. For a PTN variable x, its unnormalized density is given as

$$\text{PTN}(x) \propto x^{p-1}e^{-ax^2+bx}, x > 0. \tag{59}$$

He et al. (2019) showed (in Appendix C) an efficient sampling algorithm from the distribution. This is important in the variational updates to compute the mean.

F.3. Pólya-gamma distribution

The Pólya-gamma distribution is part of the class of infinite convolution of gamma distributions. For a Pólya-gamma variable ω with distribution $\mathrm{PG}(b, 0)$ where b denotes the shape parameter, the variable is equal in distribution with infinite sum of gamma variables.

$$\omega \overset{D}{=} \frac{1}{2\pi^2} \sum_{k=1}^{\infty} \frac{g_k}{(k - 1/2)^2} \tag{60}$$

where $g_k \sim \Gamma(b, 1)$. Polson et al. (2013) showed the distribution of the general class of $\mathrm{PG}(b, c)$ distribution can be expressed as exponential tilting of of PG ,

$$\mathrm{PG}(\omega|b, c) \propto \exp(-\frac{c^2}{2}x)p(\omega|b, 0). \tag{61}$$

For our purpose, in the variational updates it is important to compute the first moment of the distribution. As shown in (Polson et al., 2013), the first moment is given as

$$\mathbb{E}(\omega) = \frac{b}{2c} \tanh(c/2). \tag{62}$$

G. Experiments: details and results

G.1. General Experiment Details

Handling repeated trials Keeley et al. (2020b)'s PAL method and our ccGPFA infer shared firing rates for repeated trials. However, the other baselines do not. To fairly include the remaining baselines, we took additional steps to infer shared firing rates. Yu et al. (2008)'s GPFA infers separate latent processes for individual trials. To compute the firing rates of neurons, we took an average of the inferred latents across trials. In addition, since the model is fit on the square-root transformation of the spikes as Gaussian distribution, we simulated a total of 20 Monte Carlo samples and applied the inverse transformation (squaring samples) to simulate spike counts. By taking an average over the simulated spikes, we approximated the inferred firing rates. Similarly, for Jensen et al. (2021)'s bGPFA, we took an average of the latent processes across trials and computed the firing rates using the negative binomial likelihood.

Handling neurons with no/few spikes As a preprocessing step in each experiment, for each experimental condition we filtered out neurons which did not spike in any trial before fitting any GPFA models. There was one condition for the first experiment using Allen data, nine conditions for the second MC_Maze experiment with behavioral data, and one condition for the third experiment using synthetic data.

Since the second experiment with behavioral data involved fitting a linear model from inferred firing rates across multiple conditions, for neurons that had no spikes in some of the nine conditions (and thus removed before fitting GPFA models for those conditions) but had some spikes in other conditions (and thus were included in GPFA fitting for those), we imputed zero-firing rates for the conditions they were removed. This choice accounts for neurons which could be quiet for one condition but active for another. For Yu et al. (2008)'s GPFA model, we set a higher spike count threshold for removal. This was done to mitigate a rank-deficiency issue raised in the implementation. Specifically, we excluded neurons in conditions where across all trials there was exactly one spike. There were about 5-10 such neurons in each condition. Like for the other methods, if a neuron was active in other conditions, we imputed a zero-firing rate for that neuron in the fitted GPFA model for the conditions it was removed. Lastly, we note that in the first experiment with Allen Brain Observatory data, all of the neurons kept after removing zero-spiking neurons had more than one spike across the trials, so the neurons Yu et al. (2008)'s GPFA model were trained and tested on were identical to those the others were. Also, we did not use Yu et al. (2008)'s GPFA in the third experiment with synthetic data.

G.2. Scalability Results

Additional details on experiments For our methods that use inducing points, in practice, we used a stochastic extension of our sparse GPFA method presented in Section 3.1 and Appendix E. For the different choices of inducing points $M = \{50, 100, 200\}$, we set increasing number of batch sizes $B = \{100, 150, 200\}$. We used a constant step size of 0.25 for the natural gradient updates.

Table 5: **Scalability Experiment** Performance comparison in terms of average test negative loglikelihood (per spike count per trial) and running times in seconds (in parentheses). Bin/NegBin stands for our (negative) binomial ccGPFA model. The suffixes $\{50, 100, 200\}$ indicate the number of inducing points used.

Method	**T=300**	**T=900**	**T=1500**
Keeley et al. (2020b)	1.517 ± 0.004 (1657.51)	1.514 ± 0.002 (12524.83)	1.514 ± 0.002(112644.72)
Bin	1.458 ± 0.004 (19.12)	1.452 ± 0.002 (196.08)	1.451 ± 0.002 (631.37)
NegBin	$\mathbf{1.416 \pm 0.003}$ (10.21)	$\mathbf{1.415 \pm 0.002}$ (299.78)	$\mathbf{1.415 \pm 0.001}$ (1422.04)
Bin-50	1.458 ± 0.004 (5.23)	1.455 ± 0.002 (14.78)	1.460 ± 0.002 (6.34)
Bin-100	1.458 ± 0.004 (6.17)	1.452 ± 0.002 (11.46)	1.452 ± 0.002 (7.59)
Bin-200	1.458 ± 0.004 (12.99)	1.453 ± 0.002 (17.08)	1.452 ± 0.002 (15.01)
NegBin-50	$\mathbf{1.416 \pm 0.003}$ (12.22)	1.417 ± 0.002 (12.14)	1.422 ± 0.001 (31.09)
NegBin-100	$\mathbf{1.416 \pm 0.003}$ (10.74)	$\mathbf{1.415 \pm 0.002}$ (13.81)	1.416 ± 0.001 (21.09)
NegBin-200	$\mathbf{1.416 \pm 0.003}$ (20.86)	$\mathbf{1.415 \pm 0.002}$ (23.03)	$\mathbf{1.415 \pm 0.001}$ (54.94)

Effect of using inducing points From the table, we can see for $T = 300$ the runtime for the sparse NegBin-200 model takes twice as much time as the non-sparse NegBin model. However, the benefits of using inducing points can be clearly seen for $T = 900$ and $T = 1500$ where the NegBin-200 achieves similar test negative loglikelihood for roughly 1/10 and 1/25 respectively of the time taken by NegBin. This result highlights the scalability gained via sparse Gaussian Processes.

Choice of Likelihood Comparing the results for Bin and NegBin, we can see NegBin yields a better test negative loglikelihood result. This follows the fact the generative distribution uses a negative binomial likelihood.