Hierarchical Latent Variable Models for Neural Data Analysis: Q1 Project

Courtney Cheung c6cheung@ucsd.edu
Sha Lei s1lei@ucsd.edu
Yutian Shi yus029@ucsd.edu
Shuyu Wang shw043@ucsd.edu

Mikio Aoi maoi@ucsd.edu

Abstract

When the brain receives a stimulus, the associated information is encoded to inform a neural response. The goal of neural coding is to describe the sensory stimuli responsible for neural and behavioral responses, which can be measured through electrode arrays and imaging techniques. Decoding such measurements has several uses, including the development of brain-computer interface (BCI) prosthetics, which assist in and repair motor functions. Our research builds upon established dimensional reduction techniques and Bayesian inference in order to create machine learning models for decoding that account for both temporal correlations and the non-negative support of neural activity data. Specifically, we aim to learn latent neural trajectories (lower dimensional systems underlying observed neural behavior) by applying variational Gaussian process factor analysis (vGPFA) to spike train data recorded from studies on decision-making in mice. This modification of GPFA assumes that spike counts are poisson-distributed, and consequently utilizes variational inference to address the issue of non-conjugate priors in the latent space by maximizing the evidence-based lower bound (ELBo).

Code: https://github.com/courtneyacheung/Hierarchical-Latent-Variable-Models-for-Neural-Data-Analysis

1 Introduction

1.1 Literature

The work on neural spike sorting has evolved substantially, emerging after new techniques that allow monitoring hundreds of neurons at the same time in a sub-millisecond (Rey, Pedreira and Quiroga 2015). Nevertheless, due to the large amount of data that needs to be processed, new algorithms that could handle the vast neural data while filtering useful parameters are mandatory. Our approach seeks to innovate by integrating these historical strengths while mitigating their weaknesses when performing the dimensional-reduction task, aiming for better capture of the dynamic of neural trajectories in the neural population activity.

In the work on maximum likelihood factor analysis, statistics researchers Donald Rubin and Dorothy Thayer introduced the EM Algorithm. This algorithm comprises two essential steps in each iteration: the E step (expectation step) and the M step (maximization step). The algorithm ends when it converges to a maximum likelihood. The advantage of the algorithm is that it is applicable to a broad spectrum of problems and usually converges to a local maximum (Rubin and Thayer 1982). This method proves particularly valuable in scenarios where multiple maximums need to be identified, which is common in factor analysis (Rubin and Thayer 1982). In terms of implementation, the code for each iteration is relatively straightforward and boasts a low computational cost. However, it is worth noting a limitation: the method's performance diminishes when dealing with medium-sized samples (Rubin and Thayer 1982). Consequently, while the EM algorithm stands as a reliable tool, its effectiveness may vary depending on the specific problem at hand. Further research might be necessary to determine its suitability for particular applications.

In the paper "Gaussian-Process Factor Analysis for Low-Dimensional Single-Trial Analysis of Neural Population Activity", the authors introduced the Gaussian-process factor analysis (GPFA) for extracting neural trajectories. This method can help to combine smoothing and dimensionality reduction operations (Yu et al. 2008). During their experiment, they showed GPFA improved the prediction of their study by using metrics to measure how others can predict each neuron compared to two-stage methods because the two-stage methods have no use of time-label information. They also explained how GPFA is valuable for linking spiking activity across neural populations when it knows the time course (Yu et al. 2008). The Gaussian process enables the incorporation of correlation structures among low-dimensional states. This approach helps when dealing with systems that want to undergo smooth temporal evolution, and allows the extraction of a smooth low-dimensional neural trajectory, which can be seen as a balance between the low-dimensional representation of each data point achieved by factor analysis (FA) and the ambition to combine data points using continuous, smooth functions (Yu et al. 2008).

1.2 Motivation

We want to build a model that makes use of the advantages of GPFA and EM algorithms – namely, that GPFA works well with temporal data and that the EM algorithm is efficient for factor analysis, where multiple maximums need to be identified. Because Gaussian distributions produce positive and negative draws, a more appropriate modification to GPFA will use a Poisson distribution to reflect the true non-negative support of our data.

2 Methods

2.1 Overview

In this paper, we recreate the variational latent gaussian process model (Zhao and Park 2017), which relaxes the assumption of linear dynamics in order to reflect the more accurate nonlinear dynamics of the brain. This method differs from GPFA in that it uses "a point-process observation model with self-history dependence rather than an instantaneous gaussian observation model" (Zhao and Park 2017). vLGP sequentially integrates evidence into the model. Furthermore, it assumes that $y_n(t)$, the activity of a neuron at a certain time, is drawn from the poisson distribution. However with this assumption, the prior and posterior are nonconjugate. This means that there is no closed-form empirical Bayes solution, and we need to make a guess as to what the posterior of the latent process is. Thus, we use variational inference to minimize the difference between our guessed posterior and our real posterior. By maximmizing the evidence-based lower bound (ELBo), the model minimizes the Kullback-Leibler divergence, and therefore the difference between the guessed and real posterior (Zhao and Park 2017).

2.2 GPFA

In our project, we generated high-dimensional neural data. By using the GPFA model, we can reduce the data to the lower dimension and capture the variations in neural activities over time. Thus, we can decode data to predict conditions from neural trajectories. Compared to the PCA, GPFA is a more general method that considers the time dependence of neural activities. Additionally, it also includes an explicit noise model. These properties enable it to perform better than the traditional methods, including better performance in adjusting the degree of smoothness, taking account of various timescales, and distinguishing spike noises.

2.3 Why Poisson Observation?

In the traditional Gaussian-process factor analysis, the observed data are assumed to be generated from a Gaussian distribution centered around the latent variables, while the

gaussian distribution will produce noise values that are equally likely to be positive or negative centered around 0. However, our observed spike count neuron data are typically non-Gaussian and skewed (Keeley et al. 2020). Moreover, the spike trains are naturally non-negative count data. Therefore, to better fit the dynamics of the neural activity, we will apply the poisson distribution since it is a natural choice for modeling count data and avoid the negative values.

2.4 Variational Inference

Variational inference is a method that approximate the complex true posterior p(x|y) with a simpler distribution q(x) (Zhao and Park 2017). It is an essential foundation for us to construct our latent variable model.

2.4.1 Kullback-Leibler Divergence & Evidence Lower Bound

To measure the approximation of variational distribution q(x) to the true posterior, the Kullback-Leibler (KL) divergence was introduced (Zhao and Park 2017).

$$D_{KL}(q||p) = \int_{-\infty}^{\infty} q(x) \log \frac{q(x)}{p(x|y)} dx$$

When the KL divergence is equal to 0, q(x) is exactly the same as the true posterior. Thus, minimizing the KL divergence helps us to find the optimal q(x) that best approximates the actual distribution.

$$\log p(y) = L(q) + D_{KL}(q||p)$$

According to Zhao, minimizing the KL divergence is the same as maximizing the evidence lower bound $L(q)$ (ELBO), a lower bound of the log marginal likelihood (2016). Thus, q(x) can be optimized by maximizing ELBO.

2.5 vGPFA

By combining the advantages of variational inferences, the Variational Latent Gaussian Process model (vLGP) becomes more efficient and practical compared to previous methods (Zhao and Park 2017). This approach is especially beneficial for high-level neural activities such as decision-making, due to its flexibility in inferring latent trajectories for each trial (Zhao and Park 2017). Since we want to explore mice ' decision making process, which is high-level, vLGP will be an ideal starting point for us to learn the low dimensional latent trajectory. Additionally, compared to GPFA, vLGP is applicable to more diverse spike train observations (Zhao and Park 2017). Additionally, vLGP is considerably efficient compared to PLDS, another common dimensional reduction method (Zhao and Park 2017).

3 Data

3.1 Real Data

The International Brain Laboratory (IBL) collects data that records the neural activity across the entire mouse brain when it has a decision-making task. The dataset contains various data from 115 mice, 354 sessions, and 295501 neurons. Neural data is simultaneously acquired from up to two probes that record during each session (N=354).

In our project, we carefully process the data by using high-quality clusters (groups of spikes/ units) that meet certain standards (the lable variable of clusters equals 1). The data of each session includes details about the number of spikes, clusters, and trials. For every signal trial, a table is created, offering insights into trial events. This table, with dimensions equal to the number of trials, contains various variables related to trial parameters and mouse behavior.

These variables include 'firstMovement_times,' denoting the time of the initial movement detection during the trial, with NaN indicating no movement detected. The 'choice' variable signifies the response, with -1 indicating a counterclockwise wheel turn, +1 for a clockwise turn, and 0 indicating no wheel movement. Additionally, 'contrastRight' denotes the contrast of stimuli presented on the right side of the screen, while 'probabilityLeft' indicates the probability of the stimulus on the left side (with the probability on the right being 1 minus 'probabilityLeft').

For cluster data, we can get the variables like brain region and overall firing rate.

Studying the neural process behind decision-making is a basic and interesting aspect of neuroscience. The IBL dataset provides an opportunity to study the basic aspects of decision-related neural activities from patterns of trajectory and their correlation with behavioral outcomes. By exploring how to encode data, researchers can reveal the complex process behind decision-making. This dataset can also provide insights into neural variables, such as stimulus and reward clues, and show the representation and processing of key information by data.

3.2 Simulated Data

In our simulated data, we have spike data for 10 cells in 10 trials, where the number 1, 4, 6, 7, and 8 trials have stimulus on the right of the screen, and the remaining trials have stimulus on the left. Each trial contains 500 timepoints, with a time step of around 0.0126. We will discuss the data generating process in section 4.

3.3 EDA

In our Quarter 1 project, we use the simulated data to analyze the analog of cells' reaction with different conditions. In our simulated data, there are 10 cells' performance in 10 trials.

In order to check the general pattern of cells responding to left and right, we pick several cells to observe the general pattern over 10 trials. To find the overall pattern of one cell, we firstly locate the specific cells' trials data. Then, we took the average of the spike counts based on left or right across the 10 trials. This is an example cell (Figure 1) shown below. Based on the graph, we could observe that the cell has a disparate response for right and left, where the blue line represents the response corresponding to right, and the orange line represents the response corresponding to left. The graph implies that the cell responds to the right more actively than responds to the left.

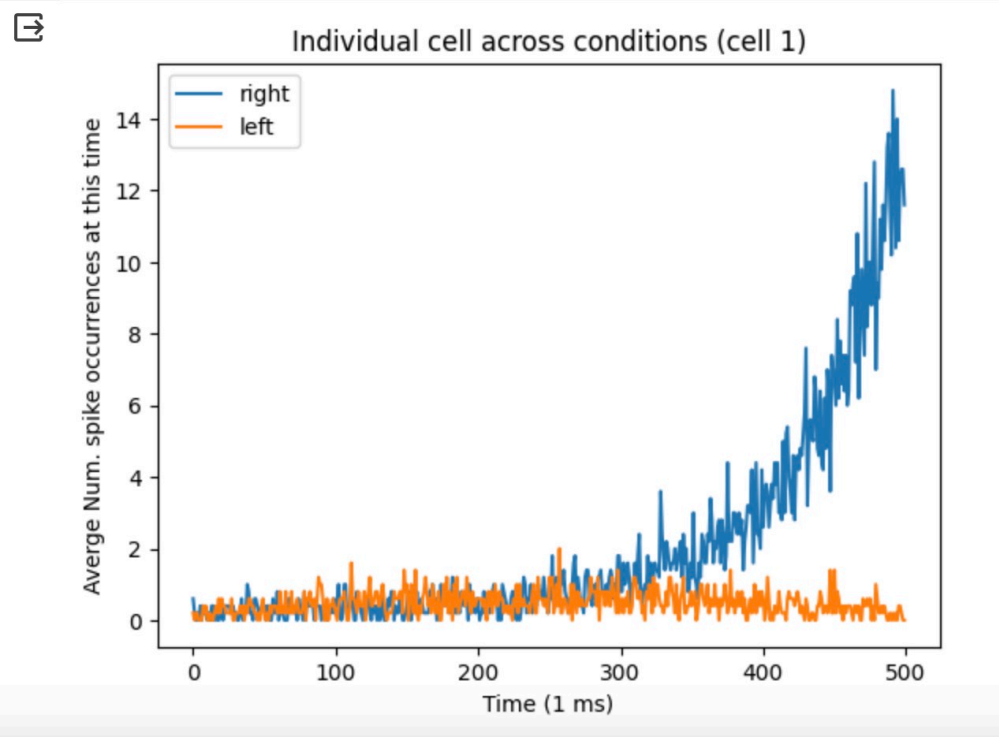
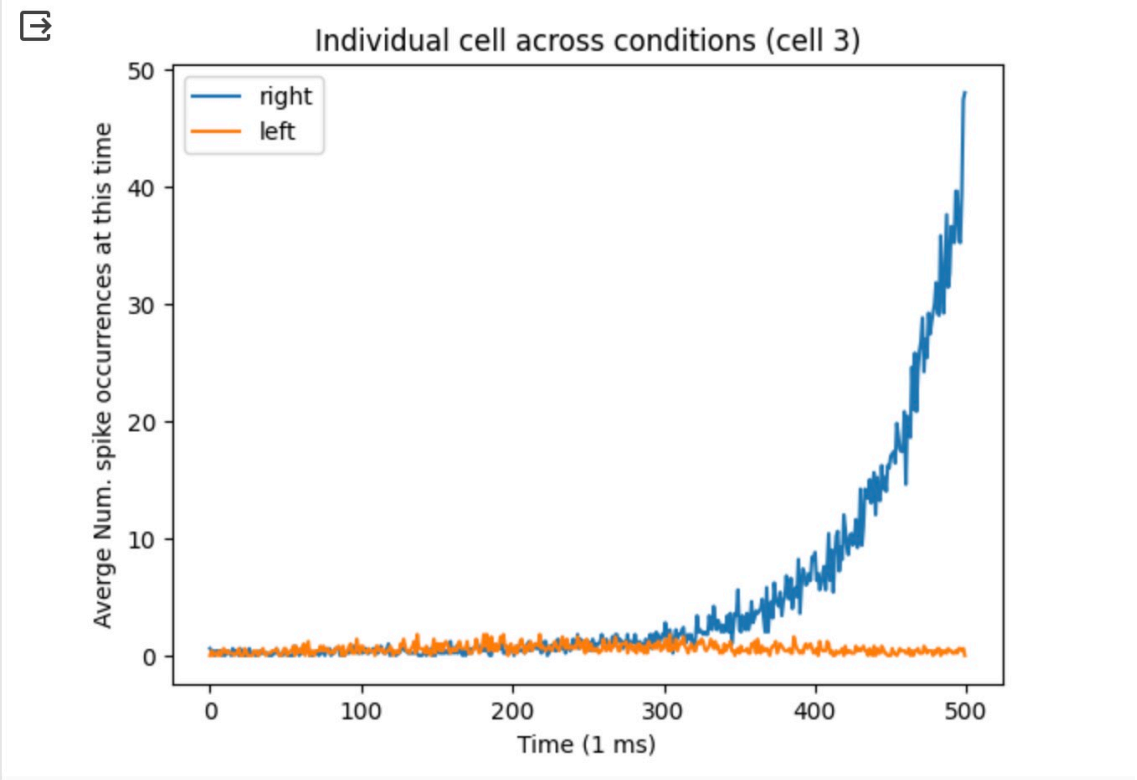
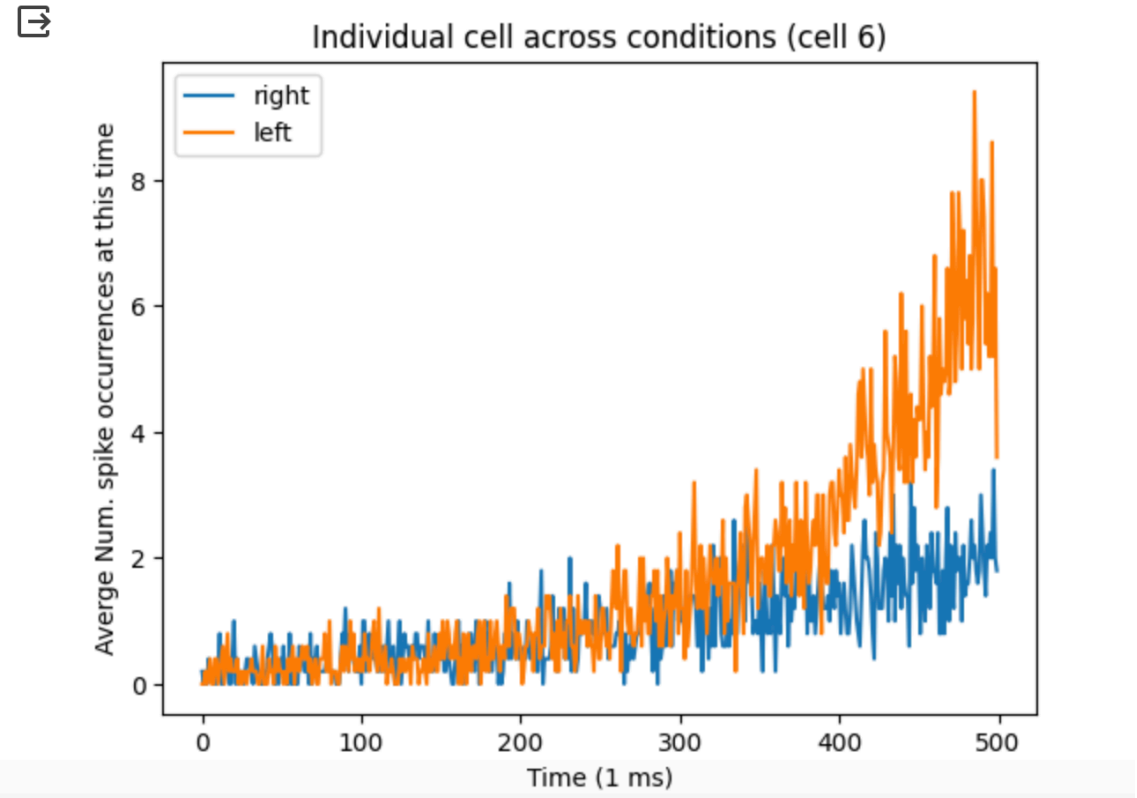


Figure 1: Individual cell 1's spike count graph with different conditions.

To expand our understanding on the neural data, we check on two more cells' average spike-count overtime graph. Cell 3 has a similar response with cell 1, on the contrary cell 6 appears to respond more actively to the left compared with right.



(a) first trial (cell 3)



(b) second trial (cell 6)

Figure 2: Spike count graph with different conditions.

The Peri-Stimulus Time Histogram(PSTH) measures and visualizes the firing rates of neurons in response to a specific stimulus over time. We draw the PSTH graph on the overall cell's response to the stimulus.

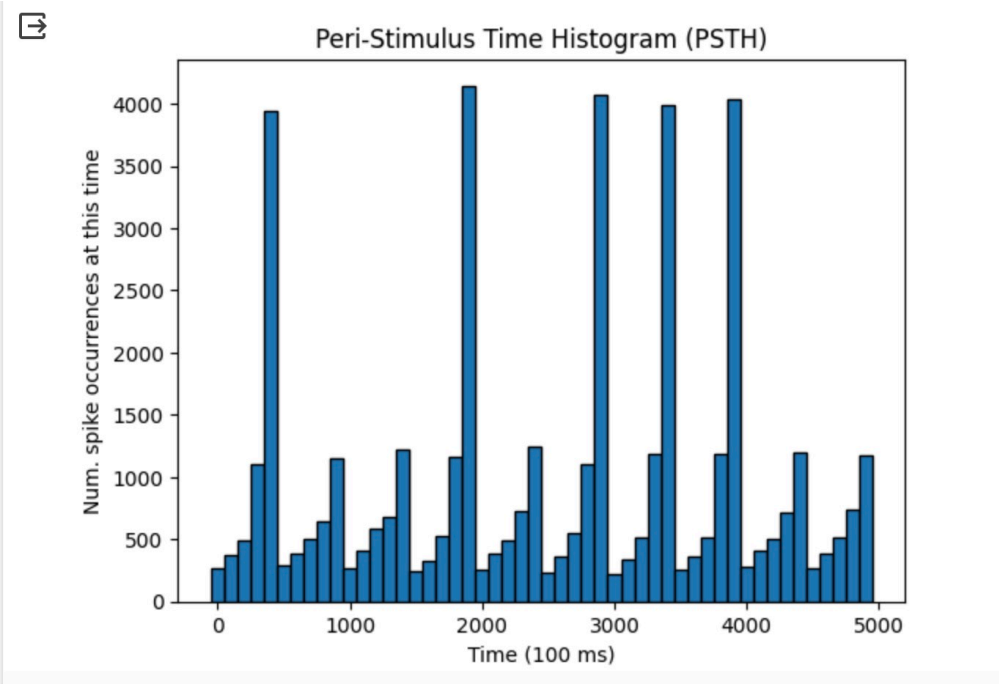


Figure 3: Peri-Stimulus Time Histogram (PSTH)

4 Proof of Concept

4.1 Simulation

To better understand the underlying structure of neural population data, we decided to generate data with the same format as real data.

4.1.1 One Trial

To achieve this, we broke down our task to first start with a single trial of data. We first generated 2-dimensional latent trajectories. This requires us to first decide the number of time points and the duration between time points. For simplicity, we chose 500 time points, each with a time step of 0.0126. We then chose two functions that behave differently over time (the time sequences we generated from the time points and time duration) to represent two latent trajectories (Figure 1), where the blue line represents the trajectory of the first latent variable (z1), and the orange line represents the trajectory of the second latent variable (z2).

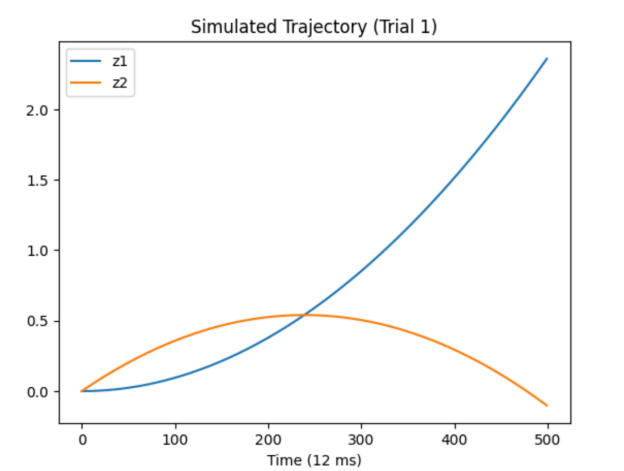


Figure 4: Simulated trajectory (trial 1)

We then created a new z with z1, z2, and an additional bias column.

$$X = zC \tag{1}$$

$$\Lambda = e^{X} \tag{2}$$

To add some variation to our data, we created a loading matrix (C) from the Gaussian distribution with our new z and the number of cells (10) and multiplied the new z with C to obtain a matrix X. By taking the exponential of X, we generated the firing rate Λ. Using Poisson distribution, we got our spikes observation (y).

4.1.2 Simulation of trials under two different conditions (the stimulus is on the right or on the left)

Since our real data contains two possible conditions, either the stimulus is on the right or on the left, we decided to generate two different latent variable trajectories for different conditions. We assumed that the trajectories we generated in 4.1.1 are for the condition that the stimulus is on the right, and our two latent variables will perform differently under the condition that the stimulus appears on the left. Thus, we generated two new trajectories for these 2 latent variables (Figure 2), where this time z1 (labeled blue) goes down over time, and z2 (labeled orange) goes up over time.

4.1.3 Multiple Trials

With previous experience of creating different trajectories based on conditions, we are then able to generate data for multiple trials. In our simulation, we generated 10 trials and picked the number 1, 4, 6, 7, and 8 trials to be the condition where the stimulus appears at

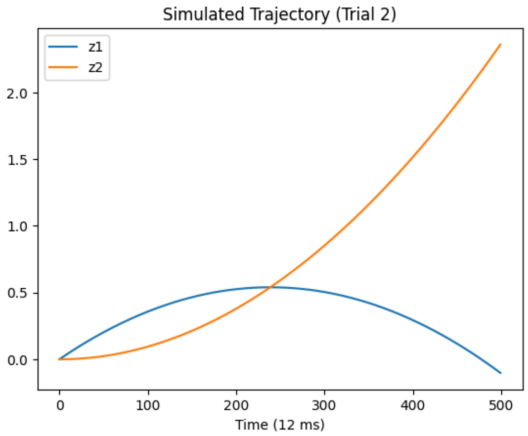


Figure 5: Simulated trajectory (trial 2)

right, and the remaining trials have stimulus at left. By joining them together, we got 5000 time points. With a similar process shown in 4.1.1, we are able to generate X and y.

4.1.4 Limitation

According to the description of real data in IBL's data release technical white paper, there is an inter-trial interval, a silence period between the end of a trial and the start of a new trial.In our simulation task, we did not include this period for simplification. However, we will take it into account when processing the real data.

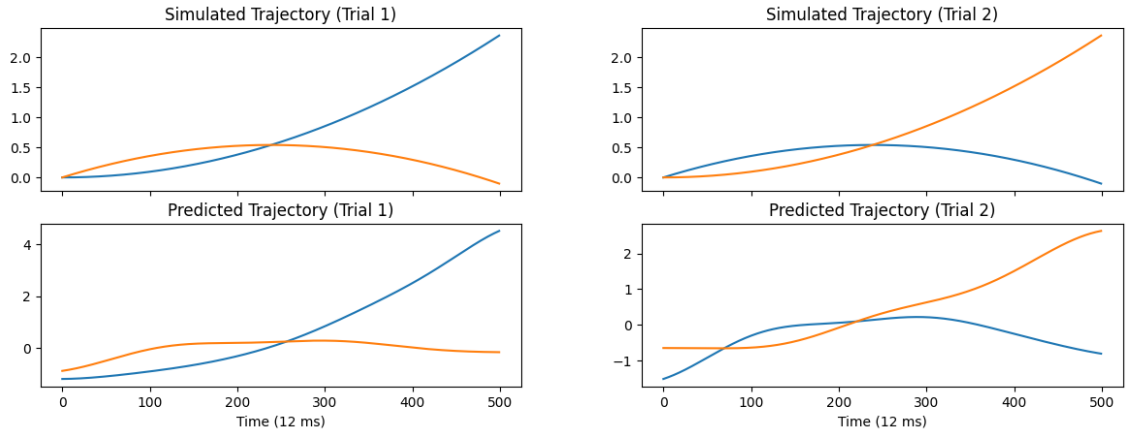
4.2 Applying vGPFA

To set up the inference, we split the spike train into 10 trials. Then we create session based on the number of time bins. Then, we add each trial into the session. Here, trial is the basic unit of observation. Then we will fit the whole session data into the model to extract the low-dimensional trajectories on each trial in the session. The parameters we use to fit the model including number of latent variables and kernel, where the kernel can be one single kernel function or a list of kernel functions. Here, we use 2 latent variables and RBF kernel with scale equals 1 and length-scale equals to 100 times of our number of time bins.

4.3 Performance

4.3.1 Observations

After training 50 EM iterations on 10 trials, we get the result of latent trajectories over time. The visualizations of the trajectories can reveal the dynamic of neural activity, helping us to identify patterns:



(a) condition: turning right

(b) condition: turning left

Figure 6: latent trajectories

From our result, we can notices that the patterns of graphs are similar to the trajectories of generated data. In figure1, we are getting the result of the first trial when the coressponding condition is turnning right. The original trajectories have the right blue line goes up, and left orange line goes down, and we can see the same trends in the result graph below that right blue line also goes up and left orange line goes down. Additionally, we get the correct directions of lines in the second trial trajectories. Thus, our model is capturing the neural dynamics associated with decision conditions.

4.3.2 Measurement Improvement

The visual inspection of latent trajectories is a good methods, but there are metics and methods provide more rigorous evaluation of the performance of the latent models.

Correlation Coefficient can be a better way. We can calculated it between the latent trajectories and the observed data. The high correlation value can help to indicates that there may be a strong correlation between the model prediction and the actual data.

5 Result Limitations and Future Improvement

Based on the model's result, we think the VPGPFA has a overall good ability on extracting a smoothing low-dimensional trajectories.

5.0.1 limitations

One limit of our simulated data is the small size of cells and trials, we get the general pattern of simulated cells across 10 trials. This might increase marginal error on the spike counts

in each time bin. And our simulated data might be far away from the real data, where the cell could respond very differently than our simulated cell.

For the quarter 1 project, we prove our concept on the snippet of simulated data to show that VPGPFA can work on the neural data. The model successfully picked out the overall pattern of cells that we simulated. But to test whether the model can be employed on the real dataset from IBL, we need to test the validity and effectiveness of VPGPFA again in the next quarter's project with.

5.0.2 Future Improvement

In the next quarter's project, we will come up with a better metric to evaluate the performance of the model that displays the deviation quantitatively. Also, we will explore other latent variable models and evaluate which model has the best performance on catching the dynamics of the neural trajectories. Furthermore, we will explore neurons in different region of the mice's brain. By comparing the trajectories differences across different conditions, we will get information on how different region of brains work coherently

References

Keeley, Stephen, David Zoltowski, Yiyi Yu, Spencer Smith, and Jonathan Pillow. 2020. "Efficient non-conjugate Gaussian process factor models for spike count data using polynomial approximations." In *International Conference on Machine Learning*. PMLR

Rey, Hernan Gonzalo, Carlos Pedreira, and Rodrigo Quian Quiroga. 2015. "Past, present and future of spike sorting techniques." *Brain research bulletin* 119: 106–117

Rubin, Donald B, and Dorothy T Thayer. 1982. "EM algorithms for ML factor analysis." *Psychometrika* 47: 69–76

Yu, Byron M, John P Cunningham, Gopal Santhanam, Stephen Ryu, Krishna V Shenoy, and Maneesh Sahani. 2008. "Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity." *Advances in neural information processing systems* 21

Zhao, Yuan, and Il Memming Park. 2017. "Variational latent gaussian process for recovering single-trial dynamics from population spike trains." *Neural computation* 29(5): 1293–1316