UNIVERSITY OF CAMBRIDGE
Department of Engineering

Machine Learning for Neural Data Analysis (I)

Author Name: Lihao Jiao
Supervisor: Dr Guillaume Hennequin
Date: 1st June 2022

I hereby declare that, except where specifically indicated, the work submitted herein is my own original work.

Signed _____ *date* 1st June 2022

Contents

1 Technical Abstract

Neural data comes from various kinds of dynamics between a huge number of neurons. A core idea of processing neural data is to summarise the high-dimensional population recordings as the dynamics of interpretable latent states in a low-dimensional space. A popular tool for this purpose is Gaussian Process(GP-) based methods which provides uncertainty quantification and flexible model selection, including the popular Gaussian Process Factor Analysis(GPFA). However, the independence between the latent variables makes GPFA unlikely to capture latent trajectories with potential dynamical structures beyond basic smoothness properties, stopping researchers from further exploring the dynamics of the brain computations. This motivates the development of Gaussian Process Factor Analysis with Dynamical Structure(GPFADS) which uses specially constructed non-reversible kernels to search for dynamical features. Having studied through GPFADS, various experiments were conducted to verify my understanding of the theory and gain more insights into this method. I find that GPFADS is particularly suitable for extracting spherical, rotatory dynamics and phase relationships between latent variables, though it seems less sensitive to other forms of dynamics. Furthermore, having also studied Bayesian Gaussian Factor Analysis(bGPFA), efforts have been made to extend GPFADS by combining bGPFA together in hope of developing an inference method that not only explores the potential dynamical structure of neural data but also infer the dimensionality of neural activity directly from the training data during optimisation. For one of

the essential implementation of bGPFADS(the combination of GPFADS and bGPFA) in bGPFA framework, I carefully examined the similarity between the non-reversible kernel and its squared version. This includes numerically calculate the non-reversibility index, where I find it more challenging than expected due to the various factors that could potentially result in a difference between the true value of non-reversibility index and the numerically computed value. In spite of this, I argue it is feasible to actually use the non-reversible covariance matrix to approximate the square root of itself. Planar bGPFADS were implemented and as expected, it recovers the trajectories of the true latents while inferring correctly the number of true dimensions. However, during the experiments, the original bGPFA turns out to infer the trajectories more quickly, although large values of learnt scale parameters from bGPFADS indicate a higher confidence in the potential dynamics between the true latent dimensions.

2 Introduction

The brain can be considered as an extremely high-dimensional dynamical system which interacts with other parts of the body, and such high-dimensional neural activity is expected to be summarised as low-dimensional latent trajectories. There are generally two classes of methods, one focuses on learning the specific dynamical models that tend to be computationally expensive and limited by assumptions of the models. The other class of methods aims directly at modelling the statistics of the latent processes, which includes Gaussian Process Factor Analysis(GPFA). Such Gaussian process(GP)-based methods are more data efficient than the first class of methods. It also has closed form formulae of posteriors that supports uncertainty quantification and flexible model selection. However, such methods are dimension-reduction methods that can infer smooth latent trajectories but do not specially capture features from a dynamical system. In other words, there is a strong assumption of the neural data that is not taken into account when using GPFA: the data results from a complicated dynamical system.

To cope with this, a new set of GP covariance functions were introduced(Rutten et al. 2020) with a measure of second order non-reversibility(I will explain and demonstrate the nature of such non-reversibility in detail in the next section), and these covariance functions can be conveniently used as the priors in GPFA together with the usual scalar stationary covariance functions such as squared-exponential kernel. In addition, the non-reversible prior exhibits a special Kronecker structure, which allows scalability to large datasets. Such GPFA inference method with non-reversible kernels are called Gaussian Process Factor Analysis with Dynamical Structure(GPFADS). During this project, I have thoroughly studied findings from ibid. Having understood the mathematical forms, I have managed to prove some of them, and I concisely demonstrate one important result in the Appendix D. In addition, I will demonstrate my own implementations to reproduce/verify findings from ibid., including implementations to sample the trajectories of the various non-reversible covariance functions. I have also implemented the plannar non-reversible process and compare its performance with that of a fully reversible process with a regression example to illustrate how the dynamical characteristics can be learnt through the the introduction of non-reversibility.

The extension of the project is to extend GPFADS with another newly developed variant

of GPFA called Bayesian Gaussian Process Factor Analysis(bGPFA), which is a fully Bayesian and more data efficient version of GPFA. A main advantage of bGPFA is the use of automatic relevance determination to infer the dimensionality of neural activity directly from the training data during optimisation(Jensen, Kao, Stone, et al. 2021). Its inference strategy includes a nested variation approach(due to some intractability caused by the prior on the readout matrix) together with whitened parameterization technique and the direct parameterization of the positive definite square root of the usual Gaussian Process covariance functions(which will be detailed in section 4 and 5). Having carefully studied both bGPFA and GPFADS, I expect that the combination of GPFADS and bGPFA, **bGPFADS**, may have the potential of both inferring the dimensionality of the neural activity and compressing high dimensional neural data into low-dimensional latent trajectories with dynamical structures. Therefore, I have put efforts into incorporating the non-reversible kernels into bGPFA. In particular, I have focused on approximating the square root of the non-reversible kernels. Finally, much efforts were made into modifying the mgplvm-pytorch package with attempts to implement the **high-dimensional bGPFADS**. All the results of my implementations will be illustrated and discussed in section 5.

3 Background

3.1 Gaussian Process Factor Analysis(GPFA)

Gaussian Process Factor Analysis(Yu et al. 2008) is a popular latent variable model for simultaneous dimensionality reduction and smoothing of neural population recordings. For simplicity of illustration, it is assumed that there is no missing data for all the observations $\boldsymbol{y}(t) \in \mathbb{R}^N$ for each of N dimensions at each of T time points. GPFA then assumes the observed data result from the (noisy) **linear** combination of a smaller set of M $(M < N)$ latent variables, each modelled as an independent Gaussian process, $\boldsymbol{x}(t) = [x_1(t), \ldots, x_M(t)]^T \in \mathbb{R}^M$. Formally,

$$\begin{aligned}
&\text{Observations} = \boldsymbol{y}(t) \in \mathbb{R}^{N\times T} \\
&\text{Latent Variables} = \boldsymbol{x}(t) \in \mathbb{R}^{M\times T} \\
&\text{Readout Matrix} = \boldsymbol{C} \in \mathbb{R}^{N\times M}, \text{such that} \\
&\boldsymbol{y}(t) \sim \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{C}\boldsymbol{x}(t), \boldsymbol{R}) \\
&x_i(\cdot) \sim \mathcal{GP}(0, k_i(\cdot,\cdot))
\end{aligned} \tag{1}$$

where $k_i(\cdot,\cdot)$ denotes the covariance function of the i^{th} Gaussian Process. This model is then trained by maximizing the log-likelihood $\mathcal{L}(\theta)$ with respect to all the covariance function parameters, mean vector $\boldsymbol{\mu} \in \mathbb{R}^{N\times 1}$, a readout matrix (implementing the linear transformation) $\boldsymbol{C} \in \mathbb{R}^{N\times M}$ and a diagonal matrix $\boldsymbol{R} \in \mathbb{R}^{N\times N}$. Since everything is jointly Gaussian, the marginal likelihood for GPFA has a closed-form solution:

$$\begin{aligned}
\log p(\boldsymbol{y}(t)) &= \log \int p(\boldsymbol{y}(t)|\boldsymbol{x}(t)) p(\boldsymbol{x}(t) d\boldsymbol{x}(t) \\
&= \log \mathcal{N}(vec(\boldsymbol{y}(t)); \mu = 0, \boldsymbol{\Sigma} = \boldsymbol{K}_{yy})
\end{aligned} \tag{2}$$

where $\boldsymbol{K}_{yy} = (\boldsymbol{C} \otimes \boldsymbol{I_T})\boldsymbol{K_{xx}}(\boldsymbol{C^T} \otimes \boldsymbol{I_T}) + (\boldsymbol{R} \otimes \boldsymbol{I_T})$ with $\otimes$ denoting Kronecker product and $\boldsymbol{K_{xx}} \in \mathbb{R}^{MT\times MT}$ is the prior Gram matrix(Rutten et al. 2020). With the assumption

of independent Gaussian process, $\boldsymbol{K_{xx}}$ is block diagonal with the i^{th} diagonal block being the i^{th} $T \times T$ Gram matrix of latent x_i. The posterior mean and covariance over latent trajectories are therefore

$$\text{Posterior mean} = \boldsymbol{K_{xx}}(\boldsymbol{C^T} \otimes \boldsymbol{I_T})\boldsymbol{K_{yy}^{-1}}(\tilde{\boldsymbol{y}} - \boldsymbol{\mu} \otimes \mathbf{1}_T) \tag{3}$$

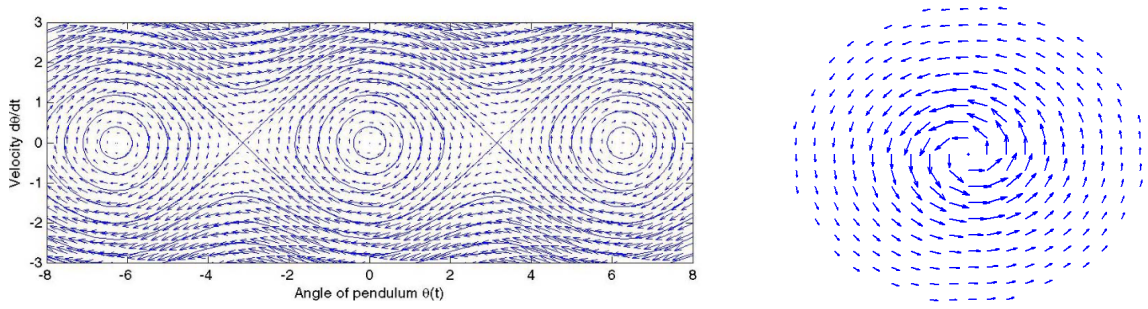
$$\text{Posterior covariance} = \boldsymbol{K_{xx}} - \boldsymbol{K_{xx}}(\boldsymbol{C^T} \otimes \boldsymbol{I_T})\boldsymbol{K_{yy}^{-1}}\boldsymbol{C} \otimes \boldsymbol{I_T}\boldsymbol{K_{xx}} \tag{4}$$

This allows for the prediction and the uncertainty quantification. However, the marginal likelihood, in this case, no longer factorizes across time and $\boldsymbol{K_{yy}} \in \mathbb{R}^{NT \times NT}$ is rank MT. This will lead to a computational complexity of $\mathcal{O}(M^3T^3)$, which can be prohibitive for longer time series. Therefore, Rutten et al. 2020 proposed some computation acceleration method, including efficiently computing the above quantities and using the Kronecker properties to speed up the computation of the posterior mean.

3.2 Dynamical Structures and Non-reversibility

In the context of the neural data, one always wants to find various kinds of relationships between the activities from neurons. This could be as simple as a phase lag, a magnitude amplification or proportional relationship. As mentioned previously, GPFA assumes the independence of the latent variables, which means GPFA can indeed reduce the dimensions and produces smooth trajectories but these trajectories are essentially independent. Therefore these latent trajectories are not of particular help to study, for example. the dynamics between neurons during a certain behaviour experiment.

So how can GPFA be improved so that it can produce latent trajectories with some characteristics of dynamical structures? How is this quantified and mathematically expressed?



(a) (b)

Figure 1: (a): Relationship between the velocity and the angle of a swinging pendulum released at a small angle to vertical. At each point (θ, v), the arrow deterministically points to $(\theta + \delta t \times v, v - \delta t \times \sin\theta)$(b): Lamb-Oseen vortex model: the fluid velocity are correlated such that the directions of their velocities are unlikely to change arbitrarily.

A typical feature of most dynamical systems is that, the relationships between the variables can be characterised by a consistent, directional mean flow field in state space, such that **any segment of state-trajectory from this dynamical system is unlikely to be produced in the opposite direction**(ibid.). Examples of such are shown in Fig.

1, where the dynamics of the simple pendulum motion[1] and the velocity vector field of Lamb-Oseen vortex model(Khan 2011) share a similarity to the neural activities in the brain: they interact and influence with each other, and it is very useful to study these patterns to gain useful information of these phenomenon. In contrast, the trajectories extracted by GPFA(from the data of a dynamical system) exhibit strong independence between the latent variables, making the trajectories look relatively messy, as shown in Fig. 2, and these messy trajectories are of less value to study the complicated dynamics behind the phenomenon.

Figure 2: Example of latent trajectories extracted from a **lawful dynamical system** with GPFA(Rutten et al. 2020), it can be seen that these trajectories are messy and barely show any correlations, making it difficult to find any inspiring dynamics, which is supposed to be the ultimate goal. In contrast, bottom right of Fig. 3 shows the trajectories extracted with GPFADS, which successfully recovers the state trajectories of the same dynamical system

In summary, GPFA is able to reduce the dimensions and generate smooth trajectories, but due to the assumption of independence between latent dimensions, such results often illustrate little or no dynamical evidence, making it difficult to study the intrinsic dynamics in depth. **One way of solving this is to incorporate an important property embedded by most dynamical system: the non-reversibility. I have drawn a schematic diagram (Fig. 3) and concisely summarise the main purpose of GPFADS in its caption**.

4 Theory

4.1 Gaussian Process Factor Analysis with Dynamical Structure(GPFADS)

Quantification of Non-reversibility

Since the major problem is the priors that are assumed independent, changes are made to these covariance functions that introduce dependence to each other and exhibit temporal non-reversibility.

Define x to be temporally reversible (Rutten et al. 2020) if and only if the covariance function $k_{ij}(\tau) = k_{ij}(-\tau)$ for all $i \neq j$ and $\tau \in \mathbb{R}$(which is the case for most usual Gaussian process covariance functions). This means that the covariance matrix $K(\tau) = \mathbb{E}[\boldsymbol{x}(t)\boldsymbol{x}(t+\tau)^T]$ be symmetric for any lag τ. This also means there must be multiple

[1] Available from https://www.dam.brown.edu/people/mumford/beyond/coursenotes/2006PartIIb.pdf

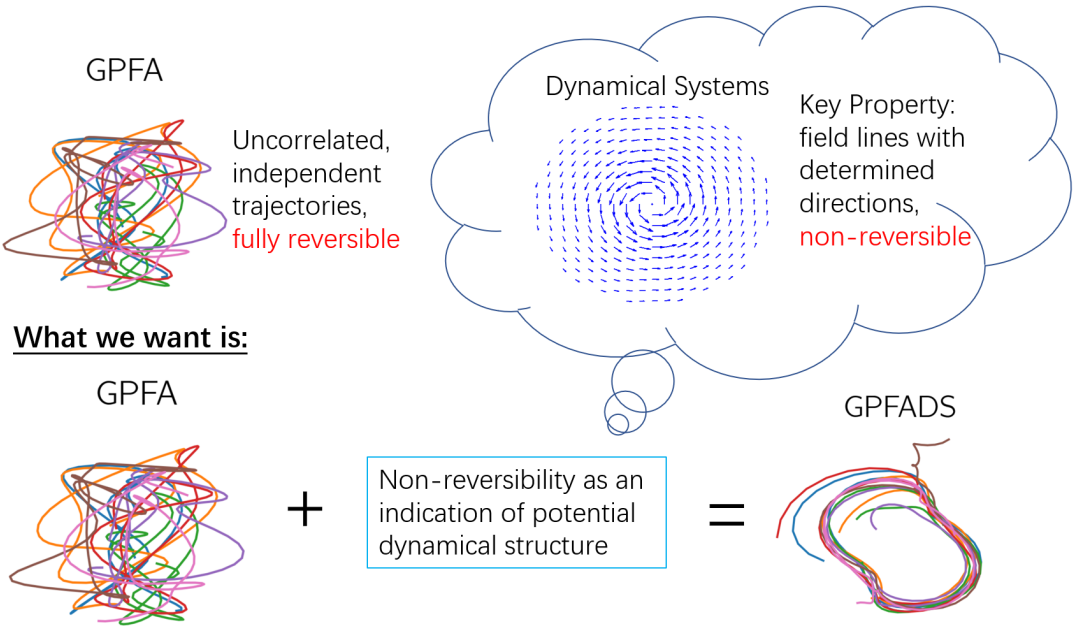


Figure 3: A scheme diagram of improving GPFA to incorporate dynamical structure for more insightful interpretation of data. **Brain is a complicated dynamical system, but GPFA only extracts uncorrelated, independent trajectories that show no sign of dynamical structures.** What we want to achieve is to enable GPFA to extracted more structured, dynamical latent trajectories instead of relatively messy and independent trajectories. Since non-reversibility is a prominent characteristic of most dynamical structures, we aim to implement non-reversibility into GPFA which serves as an indication of any potential dynamical structure, which is the newly developed GPFADS(Rutten et al. 2020).

outputs so that dependencies can be induced between them instead of a single output. To quantify the the extent of non-reversibility, define

$$\zeta = \left(\frac{\int_{-\infty}^{\infty} \|K(\tau) - K(-\tau)\|_F^2 \, d\tau}{\int_{-\infty}^{\infty} \|K(\tau) + K(-\tau)\|_F^2 \, d\tau} \right)^{1/2} \tag{5}$$

where $\|\cdot\|_F$ is the Frobenius norm. It is proved in Rutten et al. 2020 that $0 \leq \zeta \leq 1$ (and it is helpful to spot that when $K(\tau) = K(-\tau)$, $\zeta = 0$ and the process is fully reversible by definition). To start with the induction of dependence between latent variables, a new form is introduced as

$$\begin{aligned} &\text{Observations} = \boldsymbol{y}(t) \in \mathbb{R}^{N \times T} \\ &\text{Latent Variables} = \boldsymbol{x}(t) \in \mathbb{R}^{M \times T} \\ &\text{Readout Matrix} = \boldsymbol{C} \in \mathbb{R}^{N \times M}, \text{such that} \\ &\boldsymbol{y}(t) \sim \mathcal{N}(\boldsymbol{\mu} + \boldsymbol{C}\boldsymbol{x}(t), \boldsymbol{R}) \\ &x_i(\cdot) \sim \mathcal{GP}(0, k_i(\cdot, \cdot)) \end{aligned} \tag{6}$$

:

$$K(\tau) = \sum_{l=1}^{n^+} \lambda_l^+ A_l^+ f_l^+(\tau) + \sum_{l=1}^{n^-} \lambda_l^- A_l^- f_l^-(\tau) \tag{7}$$

where $n^+ + n^- = M^2$ and $\lambda_l^{\pm} \geq 0$. $A_l^{\pm}$ denotes $M \times M$ symmetric(+) or skew-symmetric(-matrices). $f_l^{\pm}$ denotes the corresponding set of orthonormal even(+) or odd(-) scalar functions. $\lambda_l^{\pm}$ denotes the corresponding weighting coefficients ordered by decreasing value within each (+) and (-) sets. Eq. 6 concisely isolates the kernel into two parts which strengthen (+) or break (-) reversibility respectively. A simple construction with only 2 dimensions will be illustrated in the next part. Further more, it has been proved (Rutten et al. 2020) that the non-reversibility index of the process can be rewritten as a function of the $\lambda_l^{\pm}$:

$$\zeta = \left(\frac{\sum (\lambda_l^-)^2}{\sum (\lambda_l^+)^2} \right)^{\frac{1}{2}} \tag{8}$$

This directly shows that presence of non-reversibility requires the skew-symmetric terms. However, such Kronecker decomposition only indicates how the non-reversibility can be induced, the next step is to construct the exact structures of these non-reversible covariance matrices such that the positive definiteness can be preserved. For the $A_l^+ f_l^+$ terms, they are simply the covariance matrices used in usual Gaussian Processes, therefore these covariance functions can be positive definite, However, it requires very specific structures of the $A_l^- f_l^-$ terms to preserve positive definiteness, to demonstrate the form of the skew-symmetric terms, a simple plannar (2-dimensional) non-reversible processes is first considered and will then expanded to higher dimensions.

Example: 2-dimensional Non-reversible Process

The planar(2d) process, $\boldsymbol{x}(t) = (x_1(t), x_2(t))^T$, is the smallest dimension possible for non-reversibility to be induced simply because all kinds of dynamics require at least 2 variables. In GPFA, $x_1(t)$ and $x_2(t)$ are independent to each other and therefore fully reversible. Consider the following construction:

$$K(\tau) = \underbrace{\begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho \\ \sigma_1\sigma_2\rho & \sigma_2^2 \end{bmatrix}}_{A^+(\text{symmetric})} f(\tau) + \alpha \underbrace{\begin{bmatrix} 0 & \sigma_1\sigma_2\sqrt{1-\rho^2} \\ -\sigma_1\sigma_2\sqrt{1-\rho^2} & 0 \end{bmatrix}}_{A^-(\text{skew-symmetric})} \mathcal{H}[f](\tau) \tag{9}$$

Eq. 9 shows how the non-reversible kernel is built. $f(\tau)$ denotes any commonly used scalar covariance function (even function), $\mathcal{H}[f](\tau)$ denotes the Hilbert transform of $f(\tau)$, which is an odd function. This form is particular chosen such that the non-reversibility can be induced while keeping the covariance function positive semi-definite provided $|\rho| \leq 1$ and $|\alpha| \leq 1$ (ibid.). Let $\sigma_1 = \sigma_2 = 1$ and $\rho = 0$ for illustration purpose, Fig. 4 helps visualise how exactly such a planar kernel is built. I have also sampled the non-reversible priors in contrast with fully reversible priors in Fig. 5.

Now, x_1 and x_2 are considered temporally correlated where non-reversibility is introduced as an indication of potential dynamical structure. Furthermore, it can be shown that $|\alpha|$ is related to the non-reversibility index as follows (I prove Eq. 10 in 2D case in Appendix D, and the result can easily be applied to high dimensions due to the special construction of the non-reversible kernel in high dimensions):

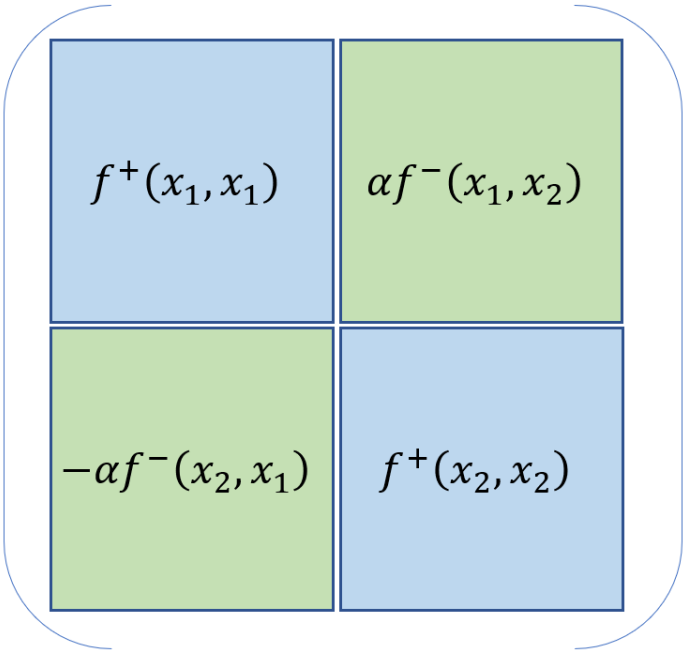


Figure 4: I draw this schematic diagram of the construction of a $2T \times 2T$ planar process kernel for illustration($\sigma_1 = \sigma_2$ and $\rho = 0$, which is an instantaneously spherical process). Each block is a $T \times T$ matrix. The 2 blue matrices are the symmetric covariance matrices commonly used in Gaussian processes, and the green matrices are the Hilbert transforms of the covariance matrices (the blue matrices). The two dimensions are therefore coupled in such a way that non-reversibility is induced.

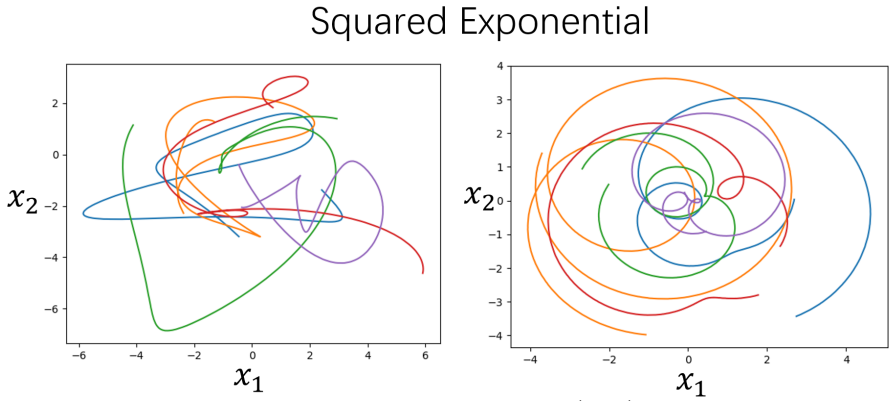


Figure 5: Sampled trajectories from fully-reversible(left) and non-reversible(right, $\alpha = 1$ in this case) planar squared exponential kernel. It can be seen that the reversible trajectories are messy and indicate little correlations between x_1 and x_2 just like the case in Fig. 2, while the non-reversible ones have shown evidence of rotatory structure that is unlikely to be produced in the other direction. This will be detailed in section 5

$$\zeta = |\alpha| \left(\frac{2(1-\rho^2)}{(\sigma_1/\sigma_2)^2 + (\sigma_2/\sigma_1)^2 + 2\rho^2} \right)^{1/2} \tag{10}$$

This has a maximum of $|\alpha|$ when $\sigma_1 = \sigma_2$ and $\rho = 0$, which is an instantaneously spherical process. The important indication here is that, learning the value of α allows us to construct this 2-dimension GPs with arbitrary degrees of non-reversibility.

To use such a structure as Eq. 9, the Hilbert transform of the selected covariance function must be available. Rutten et al. 2020 showed a list of stationary covariance functions and their corresponding Hilbert transforms, although in practice some exact function values need to be numerically approximated for some machine learning framework(for example, exact values of Dawson function is not supported by PyTorch). In addition, the listed functions have analytical derivatives which can be easily added into automatic differentiation software when conducting gradient computations for log-likelihood.

Construction of High-dimensional Non-reversible Priors

Mathematically, an M-output kernel can be constructed as follows:

$$K(\tau) = \sum_{1 \le i \le j \le M} A^{ij+} f_{ij}(\tau) + \alpha_{ij} A^{ij-} \mathcal{H}[f_i j](\tau) \tag{11}$$

where

$$\begin{aligned} A^{ij+}_{uv}(\text{symm. PSD matrix}) &= \sigma^2_{ij,1}\delta_{ui}\delta_{vi} + \sigma^2_{ij,2}\delta_{uj}\delta_{vj} + \sigma_{ij,1}\sigma_{ij,2}\rho_{ij}(\delta_{ui}\delta_{vj} + \delta_{uj}\delta_{vi}) \\ A^{ij-}_{uv}(\text{skew-symm. matrix}) &= \sigma_{ij,1}\sigma_{ij,2}\sqrt{1-\rho^2_{ij}}(\delta_{ui}\delta_{vj} - \delta_{uj}\delta_{vi}) \end{aligned} \tag{12}$$

and again $|\alpha_{ij}| \le 1$. It is important to note that, as $M > 2$, the latent variables are still associated in a plannar formibid., i.e. one latent variable is correlated to another latent variable and these two form a pair. In Eq. 10, the A^{ij+} and A^{ij-} terms are defined in the same way as A^+ and A^- in Eq. 9. An alternative way is to truncate the M dimensions to $M/2$ non-overlapping planes with no shared latent dimensions, which is illustrated in Fig. 6.

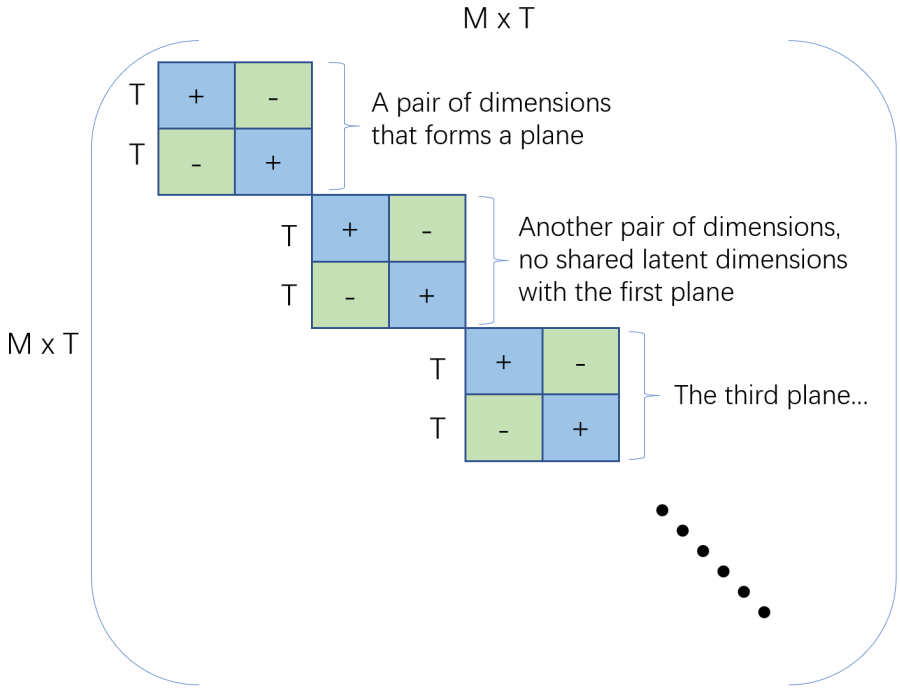


Figure 6: Schematic Diagram of $MT \times MT$ non-reversible covariance matrix, all the other entries apart from the diagonal blocks are zeros. The notation is the same with Fig. 4. Its origin is discussed in Appendix B

4.2 Bayesian Gaussian Process Factor Analysis(bGPFA)

Modifications on the GPFA model

Bayesian Gaussian Process Factor Analysis (Jensen, Kao, Stone, et al. 2021) is another Bayesian yet scalable variant of GPFA that can also infer the dimensionality of neural activity directly from the training data during optimisation. Recall that in GPFA(Yu et al. 2008):

$$
\begin{aligned}
&\text{Observations} = \boldsymbol{Y} \in \mathbb{R}^{N\times T} \\
&\text{Noise-free activity (such as firing rates)} \boldsymbol{F} \in \mathbb{R}^{N\times T} \\
&\text{Latent Variables} = \boldsymbol{X}(t) \in \mathbb{R}^{D\times T} \\
&\text{Readout Matrix} = \boldsymbol{C} \in \mathbb{R}^{N\times D}, \text{such that} \\
&p(\boldsymbol{Y}|\boldsymbol{t}) = \int p(\boldsymbol{Y}|\boldsymbol{F})p(\boldsymbol{F}|\boldsymbol{X})p(\boldsymbol{X}|\boldsymbol{X})d\boldsymbol{F}d\boldsymbol{X} \\
&p(\boldsymbol{Y}|\boldsymbol{F}) = \prod_{n,t}\mathcal{N}(y_{nt}; f_{nt}, \sigma_n^2) \\
&p(\boldsymbol{F}|\boldsymbol{X}) = \delta(\boldsymbol{F} - \boldsymbol{C}\boldsymbol{X}) \\
&p(\boldsymbol{X}|\boldsymbol{T}) = \prod_{d}\mathcal{N}(\boldsymbol{x}_d; \boldsymbol{0}, \boldsymbol{K}_d), \text{with } \boldsymbol{K}_d = k_d(\boldsymbol{t}, \boldsymbol{t})
\end{aligned}
\tag{13}
$$

Note that this is essentially the same with Eq. 1 but is rewritten for the below introduction of bGPFA implementation in accordance with notations in Jensen, Kao, Stone, et al. 2021. In this new framework, an additional Gaussian prior over the readout matrix C of form $c_{nd} \sim \mathcal{N}(0, s_d^2)$ is introduced, where the scale parameter s_d is associated with the latent dimension d (for inferring the dimensionality of neural activity). Integrating C out in $p(\boldsymbol{F}|\boldsymbol{X})$ gives the observation model:

$$
p(\boldsymbol{F}|\boldsymbol{X}) = \prod_{n}\mathcal{N}(\boldsymbol{f}_n); 0, \boldsymbol{X}^T\boldsymbol{S}^2\boldsymbol{X}), \text{with } \mathbf{S} = \text{diag}(s_1, \ldots, s_D) \tag{14}
$$

Note also that in bGPFA a general noise model is used:

$$
p(\boldsymbol{Y}|\boldsymbol{F}) = \prod_{n,t} p(y_{nt}|f_{nt}) \tag{15}
$$

where $p(y_{nt}|f_{nt})$ is any distribution of which the its density can be evaluated.

Due to the addition of the prior in the readout matrix C, the log marginal likelihood becomes intractable. Therefore, a variational inference strategy is developed(Wainwright, Jordan, et al. 2008), which "(i) provides a scalable implementation appropriate for long continuous neural recordings, and (ii) extends the model to general non-Gaussian likelihoods better suited for discrete spike counts.". (Jensen, Kao, Stone, et al. 2021)

Variational inference

The task is then to train both X and F from the data. The strategy used here is to construct two evidence lower bounds (ELBO;Wainwright, Jordan, et al. 2008) embedded together, which is reffered to as a "nested variational approach"Jensen, Kao, Stone, et al.

2021.

Specifically, a lower bound on $\log p(\boldsymbol{Y}|\boldsymbol{t})$ at the outer level and another lower bound on $\log p(\boldsymbol{Y}|\boldsymbol{X})$ at the inner level are introduced.

At the outer bound, a variational distribution $q(\boldsymbol{X})$ over latents are used to construct an evidence lower bound(Wainwright, Jordan, et al. 2008) on the log marginal likelihood:

$$\log p(\boldsymbol{Y}|\boldsymbol{t}) \geq \mathcal{L} = \mathbb{E}_{q(\boldsymbol{X})}[\log p(\boldsymbol{Y}|\boldsymbol{X})] - \mathrm{KL}[q(\boldsymbol{X})\|p(\boldsymbol{X}|\boldsymbol{t})] \tag{16}$$

Maximising this ELBO is equivalent to minimising the KL divergence Jensen, Kao, Stone, et al. 2021. This also yields the posterior over latents in form of $q(\boldsymbol{X})$. The first term of this ELBO will be estimated by Monte Carlo samples from $q(\boldsymbol{X})$ and compute the KL term analytically.

Importantly, the whitened parameterization (Hensman et al. 2015) of $q(\boldsymbol{X})$ is used that is both expressive and scalable to large datasets. Note that this whitened parameterization is closely related to combining bGPFA and GPFADS, since it requires that:

$$\begin{aligned} q(\boldsymbol{X}) &= \prod_{d=1}^{D} \mathcal{N}(\boldsymbol{x}_d; \boldsymbol{\mu}_d, \boldsymbol{\Sigma}_d), \text{ with} \\ \boldsymbol{\mu} &= \boldsymbol{K}_d^{\frac{1}{2}} \boldsymbol{\nu}_d \text{ and } \boldsymbol{\Sigma}_d = \boldsymbol{K}_d^{\frac{1}{2}} \boldsymbol{\Lambda}_d \boldsymbol{\Lambda}_d^T \boldsymbol{K}_d^{\frac{1}{2}^T} \end{aligned} \tag{17}$$

where $\boldsymbol{K}_d^{\frac{1}{2}}$ is the **square root of the prior covariance matrix** $\boldsymbol{K}_d$, and $\boldsymbol{\nu}_d \in \mathbb{R}^T$ is a vector of variational parameters to be optimised. $\boldsymbol{\Lambda}_d \in \mathbb{R}^{T \times T}$ is a positive semi-definite variational matrix whose structure is carefully chosen such that its squared Frobenius norm, log determinant, and matrix-vector products can all be computed efficiently which facilitates the evaluation of Eq. 18 and 19.

This whitened parameterization has several advantages. First, "it does not place probability mass where the prior itself does not. In addition to stabilizing learning(Murray and Adams 2010), this also guarantees that the posterior is temporally smooth for a smooth prior."(Jensen, Kao, Stone, et al. 2021) Second, the KL term in Eq. 16 is simplified to

$$\mathrm{KL}[q(\boldsymbol{X})\|p(\boldsymbol{X}|\boldsymbol{t})] = \frac{1}{2} \sum_d (\|\boldsymbol{\Lambda}_d\|_F^2 - 2 \log |\boldsymbol{\Lambda}_d| + \|\boldsymbol{\nu}_d\|^2 - \boldsymbol{T}) \tag{18}$$

Third, "$q(\boldsymbol{X})$ can now be sampled efficiently via a differentiable transform (i.e. the reparameterization trick) provided that fast differentiable $\boldsymbol{K}_d^{\frac{1}{2}} \boldsymbol{v}$ and $\boldsymbol{\Lambda}_d \boldsymbol{v}$ products are available for any vector"(ibid.):

$$\boldsymbol{x}_d^{(m)} = \boldsymbol{K}_d^{\frac{1}{2}} (\boldsymbol{\nu_d} + \boldsymbol{\Lambda}_d \boldsymbol{\eta}_d) \text{ with } \boldsymbol{\eta}_d \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \tag{19}$$

where $x_d^{(m)} \sim q(\boldsymbol{x}_d)$. This is important to form a Monte Carlo estimate of $\mathbb{E}_{q(\boldsymbol{X})}[\log p(\boldsymbol{Y}|\boldsymbol{X})]$ibid. Computation of $\boldsymbol{K}_d^{\frac{1}{2}} \boldsymbol{v}$ can be expensive for general $\boldsymbol{K}_d$ (Allen, Baglama, and Boyd 2000). In Jensen, Kao, Stone, et al. 2021, $\boldsymbol{K}_d^{\frac{1}{2}}$ is directly parameterised and the analytical expression of RBF kernel for $\boldsymbol{K}_d$ is given. Toeplitz acceleration (also used in GPFADS in Rutten et al. 2020) is used to compute the $\boldsymbol{K}_d^{\frac{1}{2}} \boldsymbol{v}$ products in $\mathcal{O}(T \log T)$ time and with

$\mathcal{O}(T)$ memory cost. However, in relation to this project where the focus is to combine bGPFA with GPFADS, an alternative approach was applied, which will be discussed in details in section 5.

The inner level ELBO is implemented in a similar way (Jensen, Kao, Stone, et al. 2021). Since the development of bGPFADS is mainly concerned with the modifications with respect to the outer bound ELBO, the details of the ELBO of $\log p(\boldsymbol{y}_n|\boldsymbol{X})$ is not introduced and discussed here. When **putting the two ELBO bounds together**, optimization proceeds at each iteration by drawing M Monte Carlo samples $\{\boldsymbol{X}_{\boldsymbol{m}_1}{}^M\}$ from $q(\boldsymbol{X})$. The exact analytical estimation formula and further acceleration methods were discussed in details in ibid.

4.3 Exploration of the Combination: bGPFADS

Motivation

Having thoroughly studied both GPFADS and bGPFA, I discuss the merits and potential limits of both methods here. GPFADS allows the potential dynamical structure of the observed data to be inferred by introducing the non-reversible kernels between dimensions. In particular, GPFADS can learn the degree of the non-reversibility (by the hyperparameters) that best explains the data, but the number of the latent variables are pre-determined without principled way (mostly by experience). Since in most cases the number of true variables are unknown, this pre-determined number of latent variables might lead to misinterpretation of the data. In contrast, bGPFA automatically infers the dimensionality of neural activity directly from the training data during optimization, but again, just like GPFA, the latent variables are assumed to be independent(I draw the multi-output covariance matrix of bGPFA in Fig. 7). This means bGPFA cannot explicitly capture the dynamical nature of neural activity beyond basic smoothness properties. Therefore, one will naturally think of combining these two inference methods together for a method with merits from both sides, which can be called Bayesian Gaussian Process Factor Analysis with Dynamical Structure, bGPFADS.

Implementation of non-reversibility into bGPFADS

The ability for bGPFA to determine the latent dimension comes from the prior of the readout matrix $\boldsymbol{C}$ in Eq. 14 and this causes intractability. The framework of bGPFA (which includes the nested variational approach consisting of 2 ELBO estimations and whitened parameterization mentioned in section 4.2) must be used. The main focus of developing bGPFADS is the $\boldsymbol{K}_d^{\frac{1}{2}}$ term, which is the square root of the prior covariance matrix $\boldsymbol{K}_d$. In ibid., to avoid the expensive computation of $\boldsymbol{K}_d$ from $\boldsymbol{K}_d^{\frac{1}{2}}$, $\boldsymbol{K}_d^{\frac{1}{2}}$ is directly parameterised by taking the square root of the chosen covariance function in the Fourier domain and computing the inverse Fourier transform. However, it is difficult to apply this direct parameterization technique to the non-reversible kernels because of the skew-symmetric Hilbert transform components: it is difficult to get analytical solutions for them (see Appendix A together with Appendix E.1 in ibid.). Therefore, in this project, attempts are first made to find efficient alternatives to approximate the square root of the prior covariance functions instead of finding analytical forms. This turns out to be

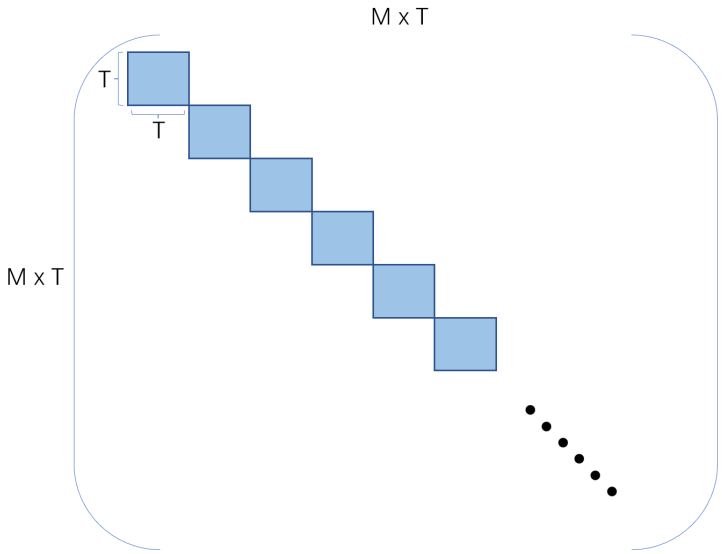


Figure 7: In contrast with Fig. 6, each dimension here in bGPFA is independent to each other.

very challenging considering the Hilbert transformed components and the constraint to preserve positive semi-definiteness. Finally, attempts are made to **directly apply the same form of the non-reversible kernels as the form of its square root**, apart from the lengthscales being divided by $\sqrt{2}$ i.e.

$$\boldsymbol{K}_d^{\frac{1}{2}} \approx \boldsymbol{K}_d, \tag{20}$$

$$l_{approx}^{\frac{1}{2}} = \frac{l}{\sqrt{2}} \tag{21}$$

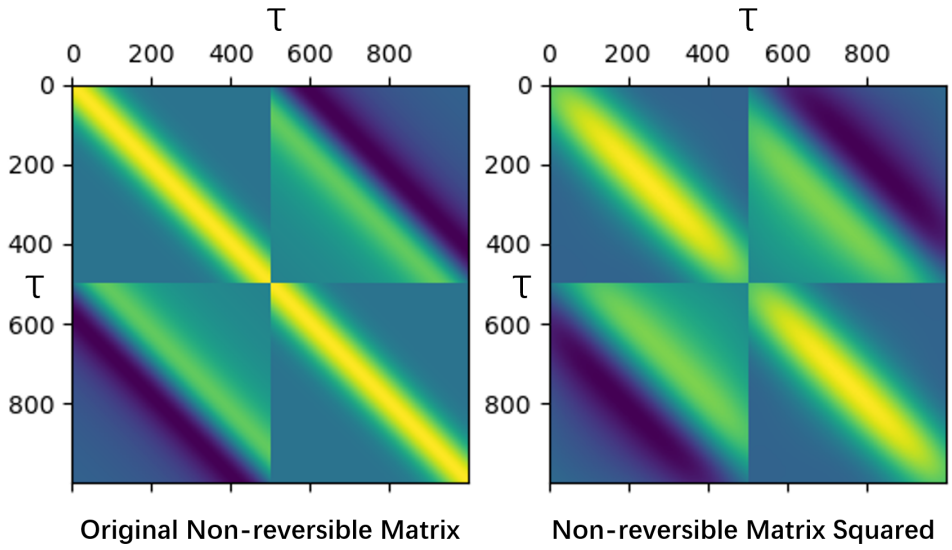


Figure 8: Comparison between the non-reversible matrix and its squared version. It can be clearly seen that the squared matrix still preserves the shape of the non-reversible matrix very well part from a little power loss at a few locations. This will be further discussed in section 5.

where $l_{approx}^{\frac{1}{2}}$ is the approximated length scale used in $\boldsymbol{K}_d^{\frac{1}{2}}$. Surprisingly, it turns that the squared version of this approximated $\boldsymbol{K}_d^{\frac{1}{2}}$ still exhibits very similar structures. I will show my full experiments on this in section 5.

After solving this main issue, there seems to be little unknown in to implement bGPFADS. Essentially, the main modifications need to be made on this $\boldsymbol{K}_d^{\frac{1}{2}}$ approximation based on the bGPFA framework and implement the corresponding changes.

5 Experiments

5.1 Visualisation of Reversible and Non-reversible Kernels

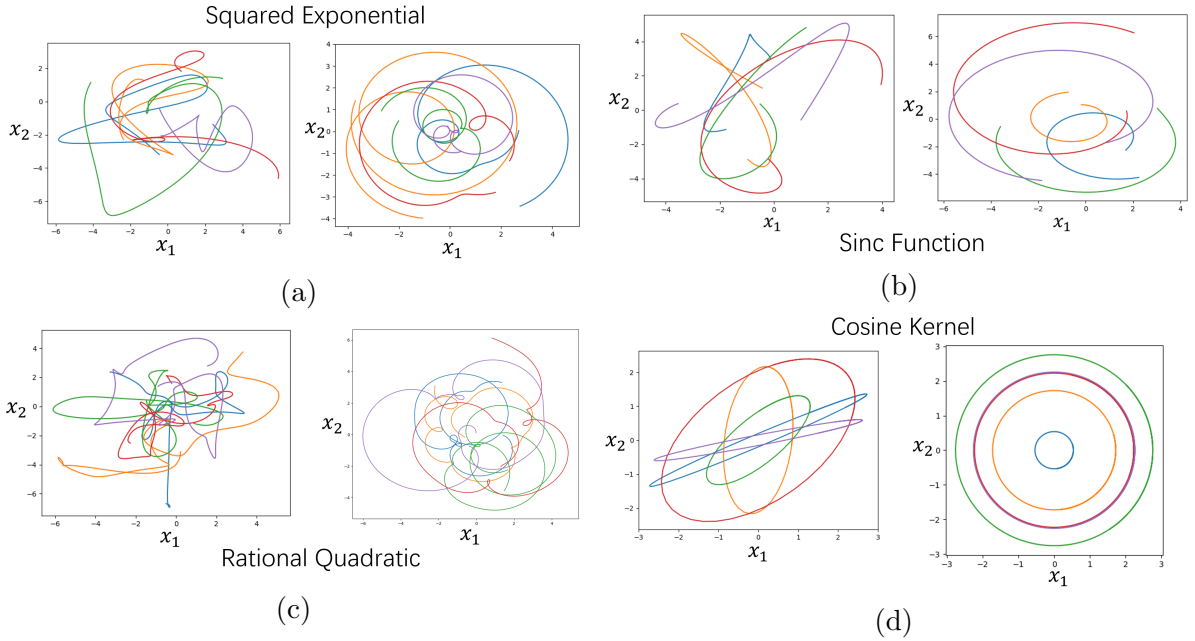


Figure 9: The left of each of (a),(b),(c),(d) shows the fully reversible kernel plot (i.e. $\zeta = 0$), where the plots on their right illustrate the non-reversible conditions($\zeta = 1$)

The first set of experiments were conducted to visualise the difference between reversible and non-reversible kernels, where trajectories of 4 different kernels (squared exponential, sinc, rational quadratic and cosine kernels) and their corresponding non-reversible versions are sampled in 2D as shown in Figure 9. After implementing the usual forms and their Hilbert transformed version of these kernels in Python, just like the usual Gaussian process, the trajectories are sampled (written in pseudo code):

$$\begin{aligned} &\text{z = randn(D,1);} \\ &\text{trajectories = chol(K)'*z + m} \end{aligned} \tag{22}$$

where K is the covariance matrix, chol is the Cholesky decomposition, and m is a mean vector. Recall from Eq. 10 how the non-reversibility index is related to these parameters. For illustration purpose, for all the non-reversible plots, $\sigma_1 = \sigma_2$ and $\rho = 0$, which results in an instantaneously spherical process.

It is clearly observed that indeed the trajectories from the fully reversible process (usual 2-output Gaussian process) gives smooth but uncorrelated trajectories where dynamical structures are hardly seen, while the non-reversible kernels produce very regular, spherical structures.

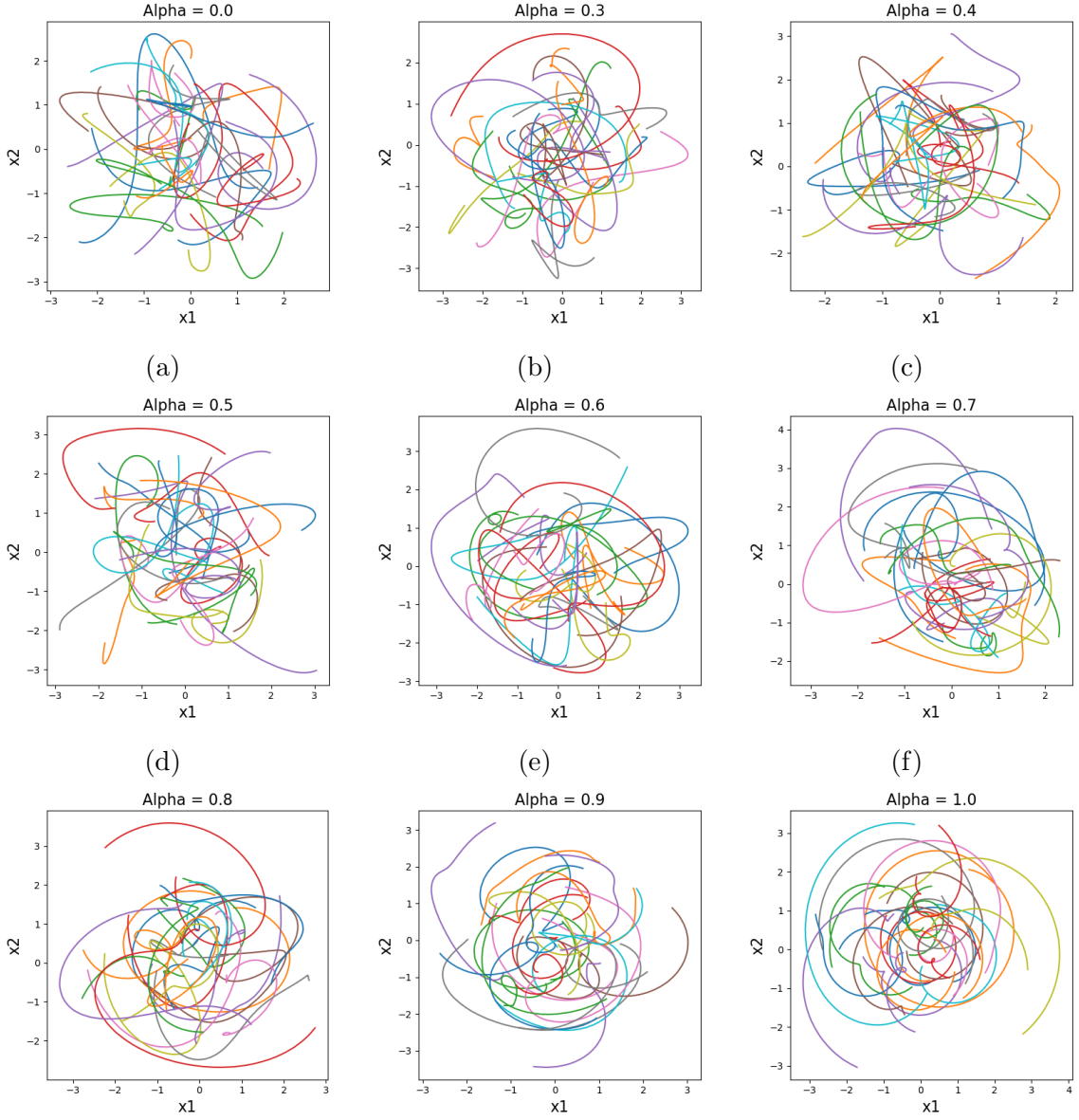


Figure 10: Sampled trajectories from the non-reversible squared exponential kernel, with α ranging from 0 to 1. For instantaneously spherical process, $\zeta = \alpha$

Furthermore, Fig. 10 shows how the non-reversibility gradually changes the shapes of these sampled trajectories as α changes. With 25 sampled trajectories plotted on each subplot, it can be observed that, as α gradually increases, the trajectories gradually untangle from the unstructured messy shapes and more and more of them show spherical structures. In other words, increasing value of α also increases the non-reversibility of the kernel, which causes stronger correlation between the two dimensions. Since $\sigma_1 = \sigma_2$ and $\rho = 0$ results in an instantaneously spherical process, the correlation is reflected more clearly with a spherical structure.

5.2 Illustrations of Non-reversible Properties with Regression Problems

1. Regression of Simple Dynamics

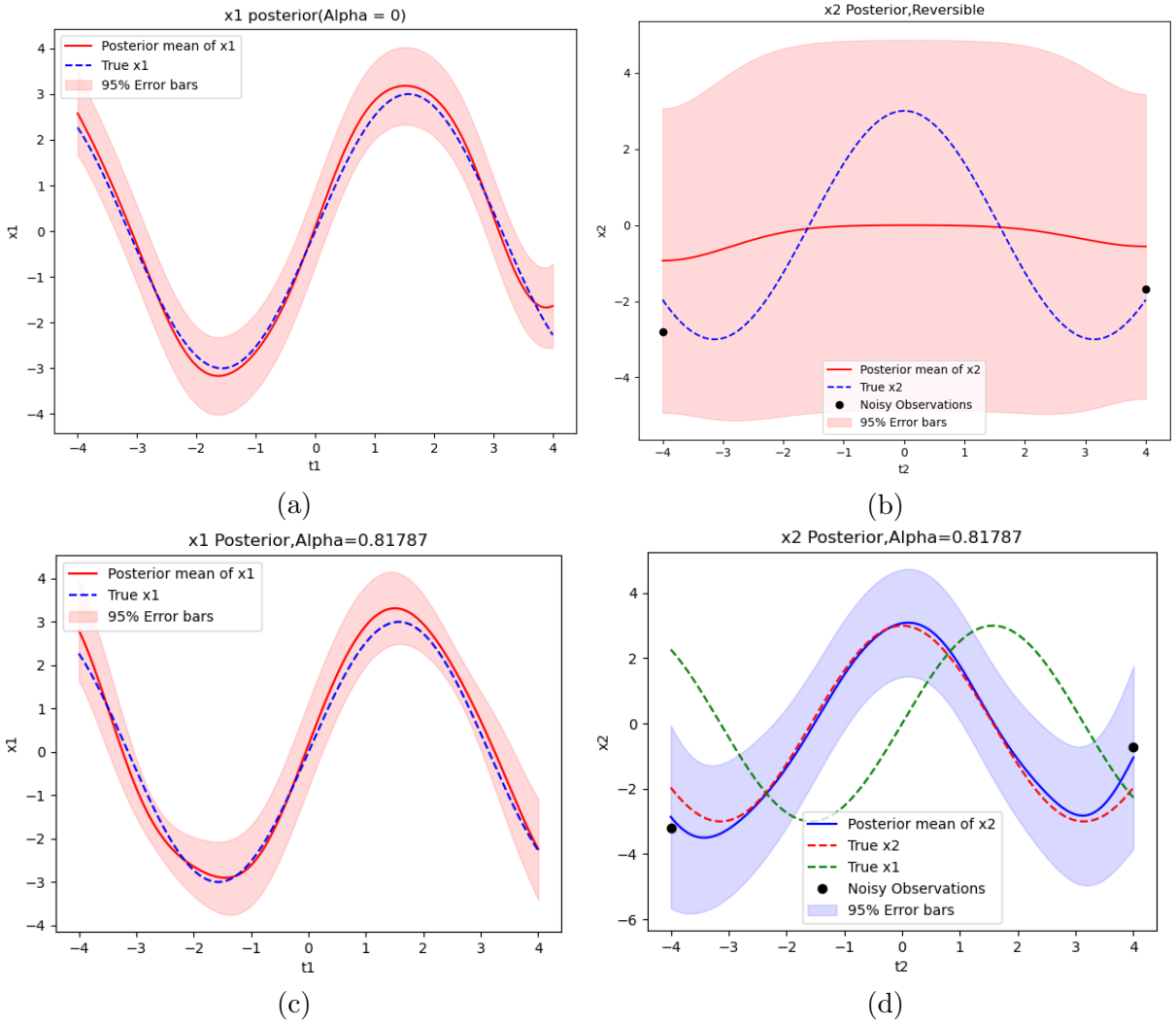


Figure 11: (a): x_1 posterior(fully reversible), this gives the expected result just like 1D Gaussian Process Regression; (b): x_2 posterior(fully reversible), which gives huge error bars and fails to extract any dynamic structure; (c): x_1 posterior(non-reversible), there is a slight difference at $t_1 = 4$ where the non-reversible prior gives better prediction as the direction of the posterior mean moves in the same direction as the true trajectory, while this is not the case in the fully reversible regression (compare (c) with (a) at $t_1 = 4$). (d): x_2 posterior(non-reversible case) which accurately reconstructs the true x_2 trajectory even with only 2 observations.

I made another comparison in the context of a regression problem. Pendulum trajectories ($\dot{x_1} = x_2$ and $\dot{x_2} = -\sin(x_1)$) are simulated. The special thing here is that, the x_2 posterior is conditioned on x_1 **and very few number of data points, in this case only 2**. Fig. 11 shows the results. For reversible GP regression(Fig.11 (a) and (b)), x_1 posterior gives decent predictions and error quantification but x_2 posterior fails to predict the shape of ground truth trajectories and give huge error bars. In contrast, for planar

non-reversible process, just like the case in Fig. 11-a, x_1 posterior with optimised α also gives decent performance (Fig.11-c). In fact, if we look at the tail of the posterior mean of x_1 and compare this with that in Fig.11-a, we can see that the non-reversible x_1 posterior mean better learns the shape(direction) of the ground truth, while in Fig.11-a, although the posterior mean is still within the $\pm 5\%$ error bars, the prediction at around $t_1 = 4$ has started to deviate from the ground truth. The biggest contrast between reversible and non-reversible process regression is seen in Fig. 11-d, where the x_2 posterior manages to give great predictions of the ground truth with decent error bar quantification **even with only 2 observations**.

To understand how such a difference is made, I illustrate plots of the fully reversible and non-reversible kernels in Fig. 12. This is the key to understand what happens during the regression. In Fig. 12-a(fully reversible), the two dimensions are independent to each other and there is no transmission of knowledge between them when it undergoes multiplication. This is because the skew-symmetric blocks are all zeros, thus during matrix multiplications there is no way for the 2 dimensions to communicate. In contrast, the non-reversible kernel provides opportunity for the 2 dimensions to transfer information through the skew-symmetric Hilbert transformed blocks. Recall from Fig. 4 in section 4.1 that the skew-symmetric (green) blocks' inputs are both x_1 and x_2, the "information" of both dimensions are therefore mixed and shared during the kernel multiplications. I would conclude this flow of information between the two dimensions as the intrinsic reason why the dynamical structure can be inferred with non-reversible kernels.

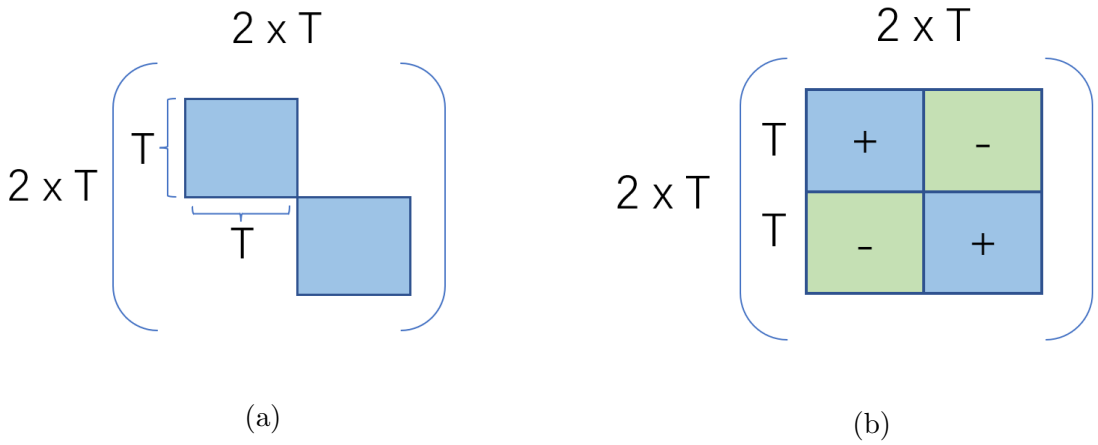


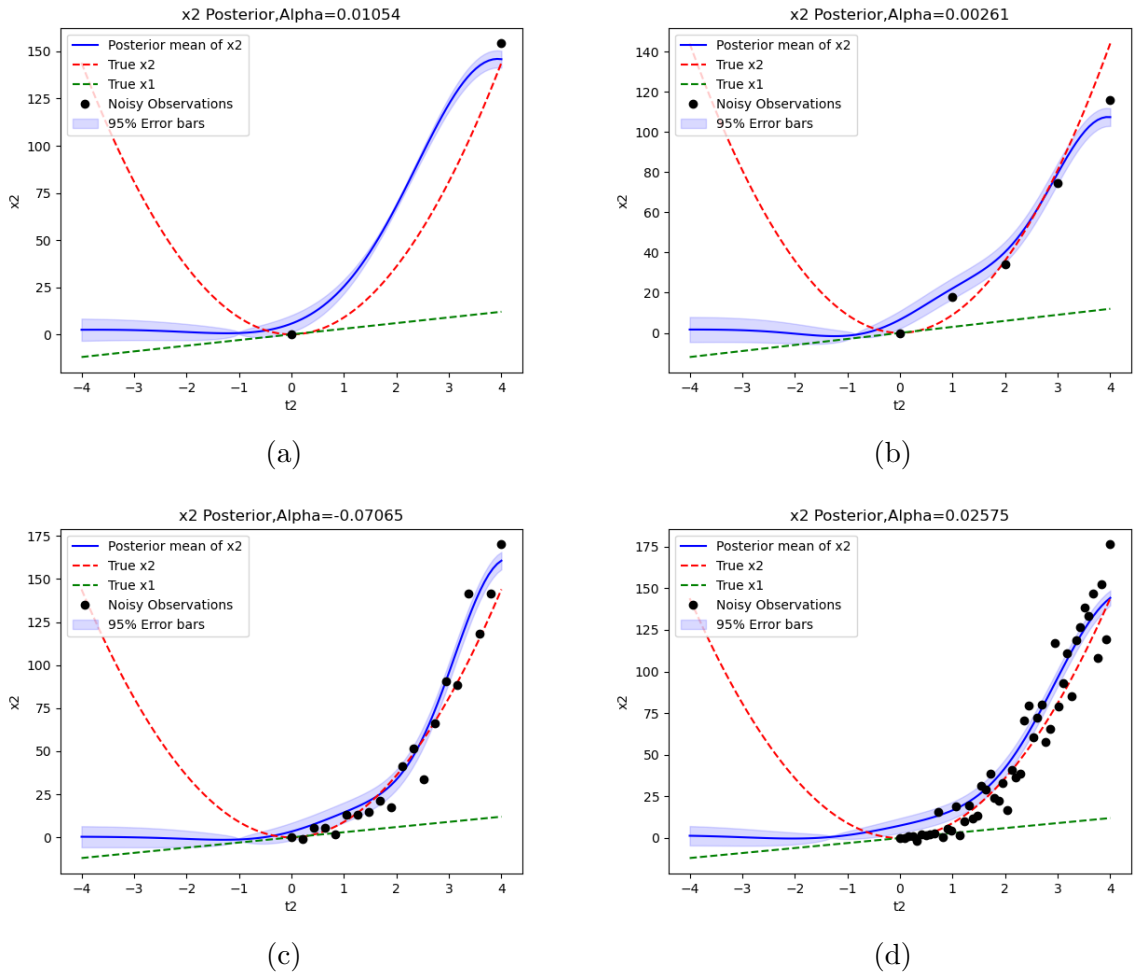
Figure 12: (a): Fully reversible kernel; (b): Non-reversible kernel

2. What happens if the dynamics becomes more complicated?

I would like to further discuss what happens when the dynamics between x_1 and x_2 becomes more complicated. We should realise that this non-reversible process is not magic: consider again the planar(2D) process, if the dynamics becomes more complicated, it would be unlikely for x_2 to infer the correct dynamics with only 2 observations. This is because, even the information from the other dimension can be shared between, the information, or evidence, required to infer the correct trajectories from very few observations of x_2 is likely to be insufficient given a complicated dynamics. In other words, a

full observation in one dimension still provide limited knowledge of the other dimension even if the two dimensions are correlated. This brings to an important question: how far can this non-reversible priors learn?

This is best answered by looking at the results in section 5.1 (see Fig. 9). All the sampled trajectories show strong rotatory structures, indicating that these non-reversible priors are best at learning specifically rotatory structures. This also means such non-reversible construction can also learn a certain extent of phase relationship, which explains why the non-reversible kernel does an impressive job in Fig. 11 and in Rutten et al. 2020 in finding rotatory structure in primary motor cortex (it has been suggested by Churchland et al. 2012 that strong rotatory latent dynamics are embedded in primary motor cortex).



(a) (b)

(c) (d)

Figure 13: Now $x_2 = x_1^2$, and no matter how many observations are available (2, 5, 20, 40 observations of x_2 in (a), (b), (c), (d)), the "stubborn" non-reversible posterior still refuses to learn the concave upward shape.

To better illustrate my argument, let $x_2 = x_1^2$ and conduct the regression again. The results are shown in Fig. 12. It can be seen that, for a concave upward shape, the posterior mean from non-reversible process still exhibits rotatory, sinusoidal prediction. Due to the contradiction between the concave upward shape and the rotatory shape, the optimised value of α are very close to 0 for all four cases. The more intrinsic reason of the results

in Fig. 13 relates to the fundamental structure of the non-reversible kernel and a few key assumptions. These discussions are in Appendix B.

Note, however, that one can obviously increase the latent dimensions in hope of capturing features of a more complicated dynamics instead of using 2 dimensions only, although at a cost of increased computations. Various acceleration methods have been provided in Rutten et al. 2020, but one should still consider carefully in practice the relationship between the computation cost and the performance.

5.3 Square Root of the Prior Matrix Implemented in bGPFA

As mentioned previously, attempts are made to directly use $\boldsymbol{K}_d$ as an approximation of $\boldsymbol{K}_d^{\frac{1}{2}}$. From another perspective, this approximation works if the squared of the non-reversible matrix can be a good approximation of itself!

1. Visualisation of the non-reversible matrix and its squared version

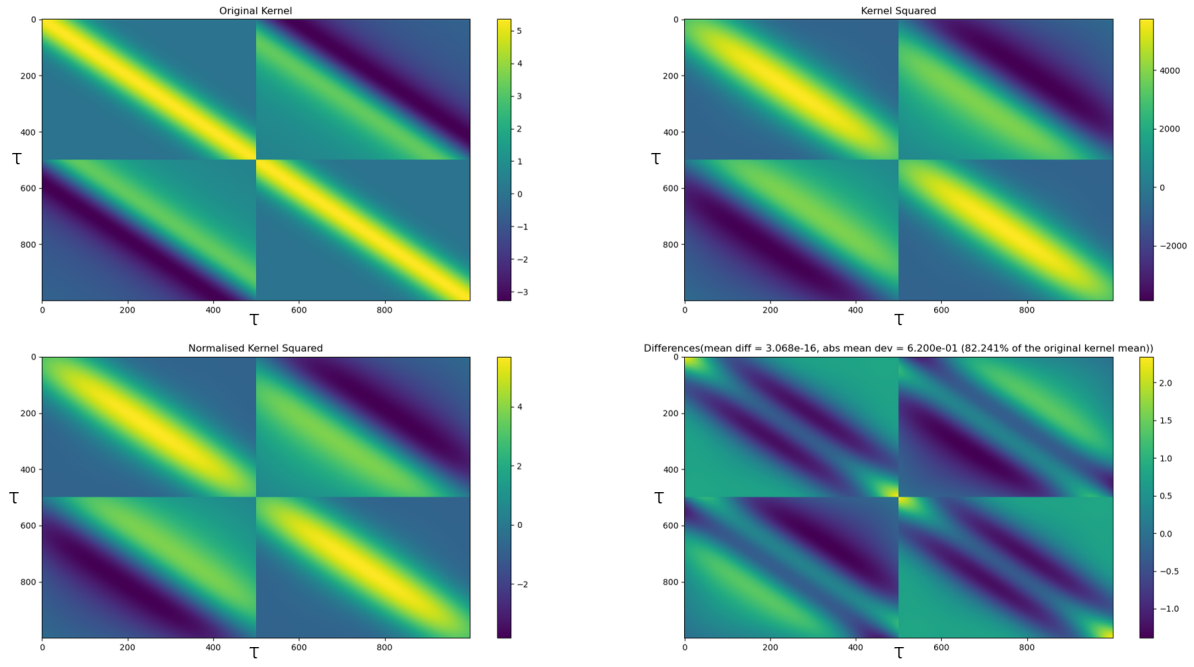
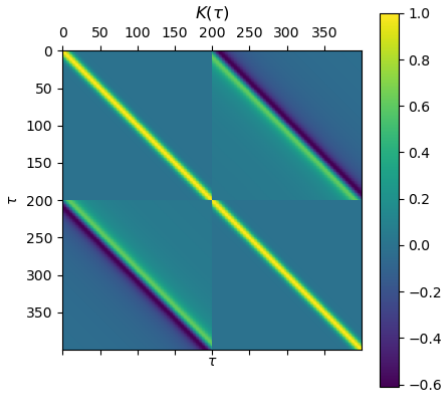
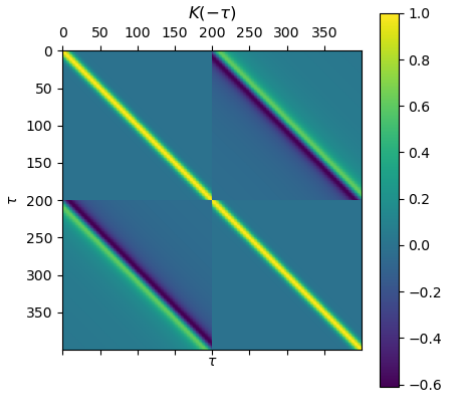


Figure 14: Top left: the original non-reversible kernel(squared exponential). Top right: matrix squared. Bottom left: the normalised squared matrix. Bottom right: Difference between the original matrix and the normalised squared version

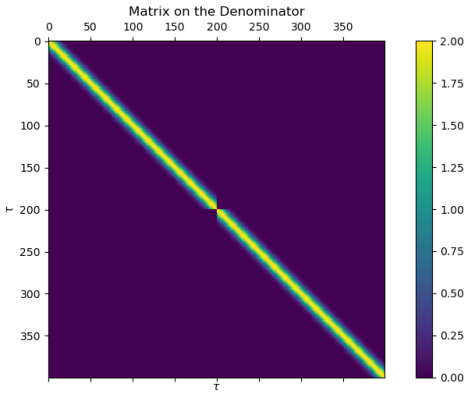
I illustrate my results in Fig. 14. It can be observed that this approximation works surprisingly well, with only a little loss of power in the corners of diagonal entries. In terms of these deviations shown in the bottom right subplot in Fig. 14, note that this is only a prior and there are many hyperparameters available to be tuned to compensate for the gaps. Therefore, these findings suggest this is a feasible and convenient approximation.



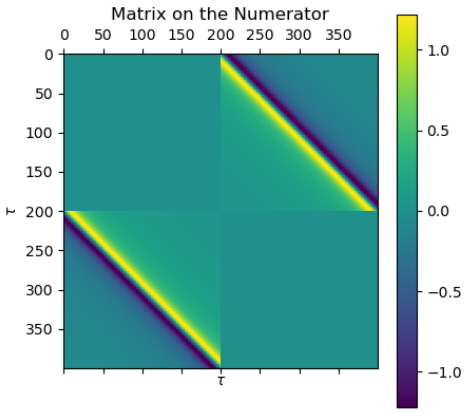
(a) $K(\tau)$



(b) $K(-\tau)$



(c) $K(\tau) + K(-\tau)$



(d) $K(\tau) - K(-\tau)$

Figure 15

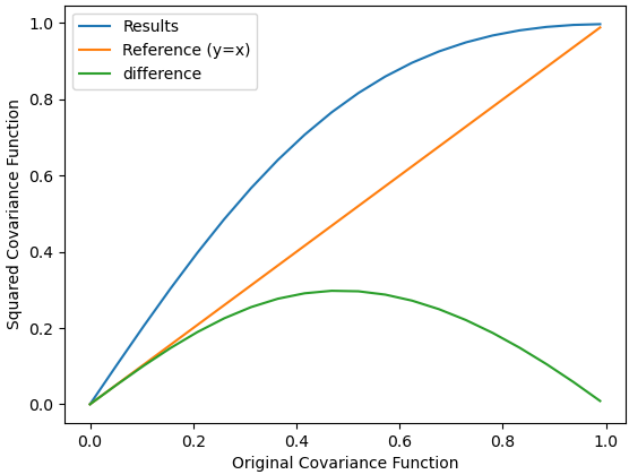


Figure 16: Comparisons in the values of non-reversibility indices for original and squared non-reversible kernel. Note that same results are observed for different kernels used

2. Change of the degree of non-reversibility with the proposed approximation method

A natural question to ask at this stage is, how much of the non-reversibility is changed due to this approximation method?

To quantify this, Eq. 5 is used to calculate the degree of non-reversibility for the squared matrix and compare it with that of the original non-reversible matrix (**in 2D, and the results can apply to higher dimensions due to the coupled-plane nature of the non-reversible kernel structure**). Fig. 15 illustrates the matrices of which their Frobenius norms are evaluated. Intuitively, it is the ratio of the norm of non-reversible components and reversible components. Great care was taken when numerically evaluating the non-reversibility index(see Appendix C). The results are shown in Fig. 16 (for a simultaneously spherical process). I have found the results are the same for different kernels used, and such figure can be used to find the effective reversibility index for the squared covariance function given the reversibility index of the original covariance function. It is also surprising to observe that, although there are slight losses of powers (Fig. 14-upper right), the squared matrix always has higher value of ζ. The biggest gap occurs when the reversibility index of the original covariance function is around 0.5. The difference can be around 0.3. This result further verifies the feasibility of the proposed approximation method, as Fig. 16 shows a monotonic (and increasing) relationship between the non-reversibility indices of original and squared of kernels.

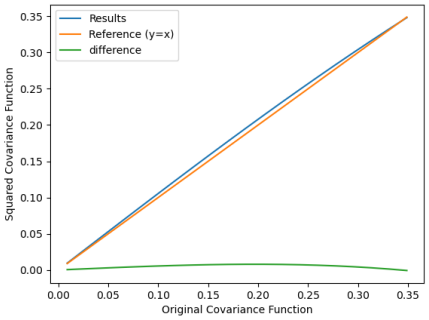
However, so far I have only discussed the case of a instantaneously spherical process ($\sigma_1 = \sigma_2$ and $\rho = 0$ in Eq. 10). What if $\sigma_1 \neq \sigma_2$ and $\rho \neq 0$?

Eq. 10 can be very helpful to answer this question. We observe that all of σ_1, σ_2 and ρ are involved in the coefficient of $|\alpha|$. First consider the effects of σ_1 and σ_2. In context of the non-reversible kernel, it is reasonable to assume that both σ_1 and $\sigma_2 \neq 0$. We can also observe that it is actually the ratio of σ_1 and σ_2 that influences this whole coefficient, and it is easy to prove that $(\sigma_1/\sigma_2)^2 + (\sigma_2/\sigma_1)^2 \geq 2$ as long as the ratio is valid(i.e. both denominators are non-zero). This means that, with ρ unchanged, whenever $\sigma_1 \neq \sigma_2 \neq 0$, the presence of them always decreases the value of the whole coefficient of $|\alpha|$. This further suggests that, when considering the difference in non-reversibility index of the original and squared non-reversible matrices, this difference will also be reduced. The same argument also applies to ρ where $|\rho| \leq 1$, as it is easy to show that:

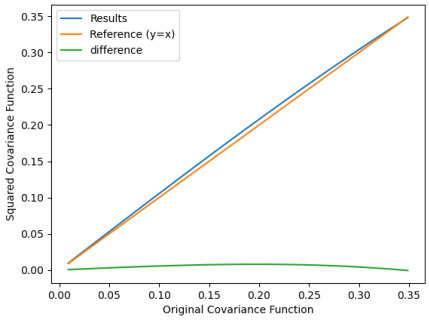
$$\frac{2(1-\rho^2)}{(\sigma_1/\sigma_2)^2 + (\sigma_2/\sigma_1)^2 + 2\rho^2} = \frac{(1-\rho^2)}{0.5[(\sigma_1/\sigma_2)^2 + (\sigma_2/\sigma_1)^2] + \rho^2} \in [0, 1] \tag{23}$$

given $|\rho| \leq 1$ and $(\sigma_1/\sigma_2)^2 + (\sigma_2/\sigma_1)^2 \geq 2$. Therefore, all of these parameters lead to decreasing the coefficient of $|\alpha|$ and thus the difference in the non-reversibility between the non-reversible kernel and its squared version. Fig. 17 helps visualise the argument.

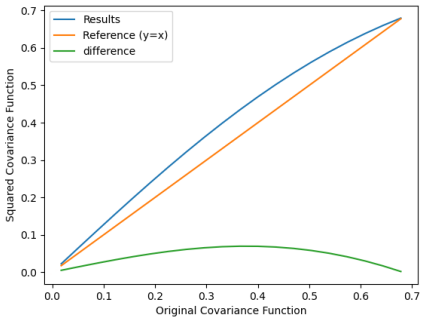
I conclude that, it is feasible to directly use $\boldsymbol{K}_d$ as an approximation to $\boldsymbol{K}_d^{\frac{1}{2}}$, as the square of $\boldsymbol{K}_d$ still shows strong non-reversible structure as $\boldsymbol{K}_d$, and the above analysis have shown that, the non-reversibility index of the squared kernel monotonically increases with that of the original kernel, and values of σ_1, σ_2 and ρ can actually further decrease this difference. In other words, the largest difference in non-reversibility occurs in instantaneously spherical process where $\sigma_1 = \sigma_2$ and $\rho = 0$, all other conditions will decrease



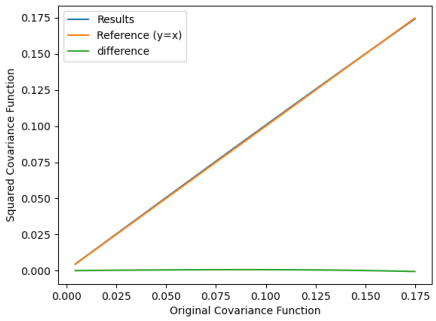
(a) $\sigma_1 = 1$ and $\sigma_2 = 4$



(b) $\sigma_1 = 10$ and $\sigma_2 = 40$



(c) $\sigma_1 = 1$ and $\sigma_2 = 2$



(d) $\sigma_1 = 1$ and $\sigma_2 = 8$

Figure 17: We can observe that (a) and (b) show identical plots, while the difference becomes large in (c) and much smaller in (d) as the ratio of σ_1 and σ_2 changes, which perfectly verifies my argument. The same argument also applies to ρ. In other words, the worst scenario of such difference occurs in instantaneously spherical process where $\sigma_1 = \sigma_2$ and $\rho = 0$, any other condition will decrease such difference.

such difference.

5.4 Implementation of bGPFADS

Modifications of mgplvm-pytorch package to build bGPFADS

The implementation of bGPFADS are conducted in Python under the bGPFA framework (using mgplvm-pytorch package from Jensen, Kao, Tripodi, et al. 2020). Due to the sophisticated coding framework of the package, modifications on the bGPFA framework to construct non-reversible kernels are challenging. The implementation started with its "K_half" function where directly parameterised square root of matrix is constructed. Here I directly use the non-reversible covariance function. In this bGPFA framework, the full structure has form shown in Fig. 7, but in each dimension, only first column of it is stored and will be recovered by Toeplitz multiplication. Thus efforts were made to accommodate this pipeline and reshape the full structure as the one shown in Fig. 6 (note that the covariance function matrix is never fully constructed for saving computational expense).

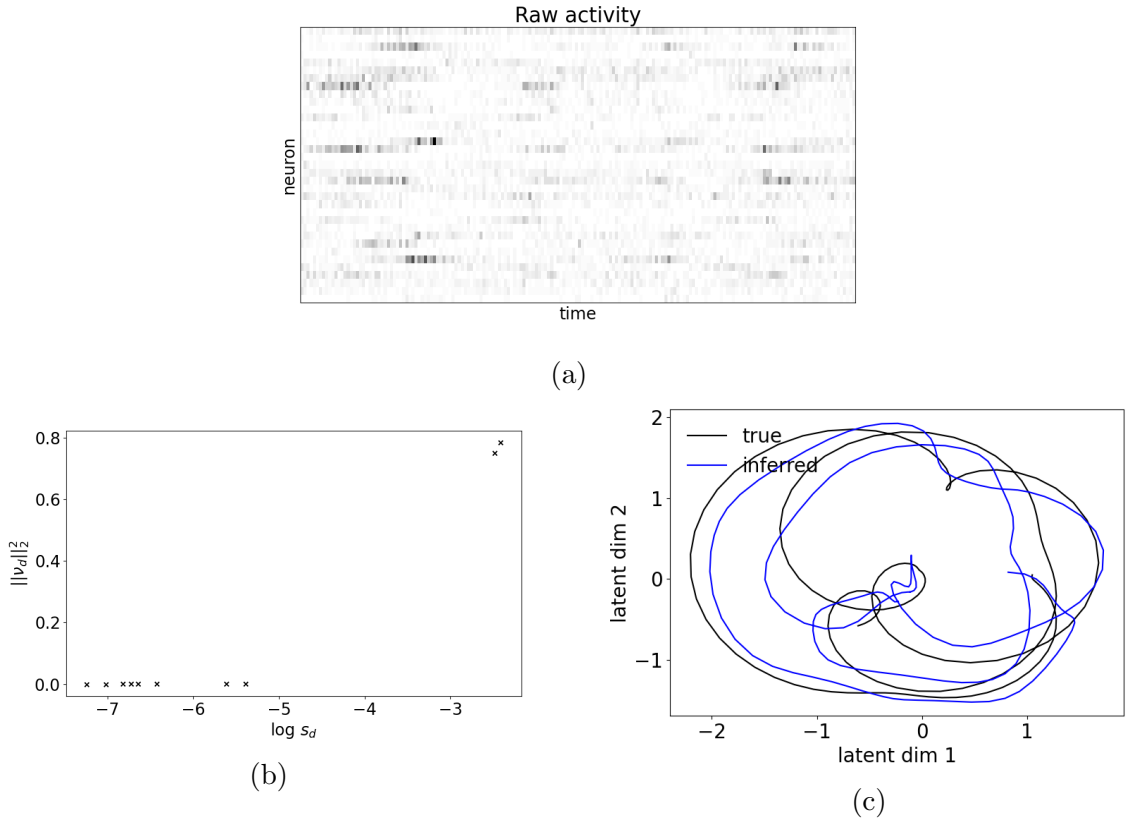
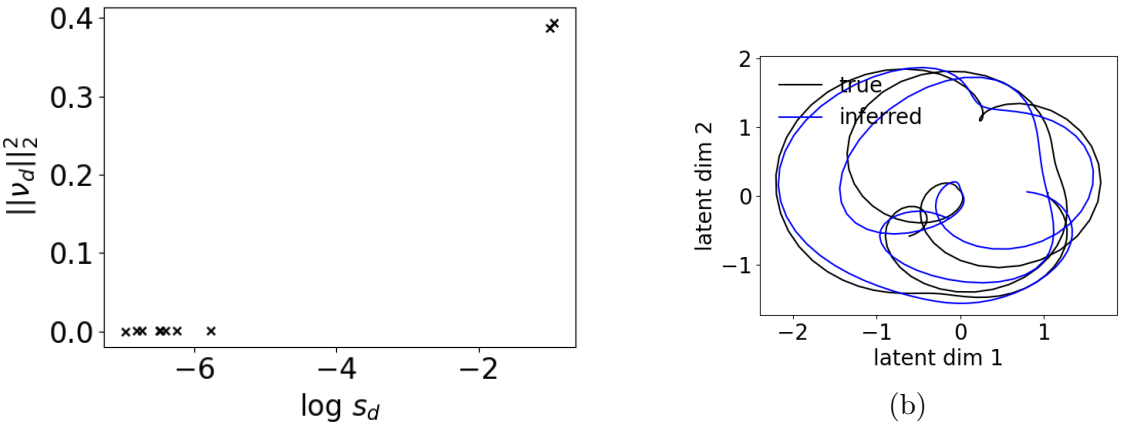


Figure 18: (a)(synthetic)Neural data set where 35 "neurons" produced the spike activities as illustrated; (b): A total of 10 latent dimensions are involved, and clearly bGPFADS successfully inferred the correct number of the latent variables; (C): The spherical structure is successfully inferred, although there seems to lack a little of smoothness (after having undergone 2500 iterations)

Planar bGPFADS

The plannar bGPFADS was successfully built, but being planar means that for D latent dimensions, only 1 plane (2 dimensions) contains the non-reversible structure. Experiments were conducted with the help of the inference demo file from the mgplvm-pytorch package. Briefly speaking, a synthetic dataset is generated from sampling from a non-reversible plannar kernel, such that: 1. There are only 2 'true' latent variables; 2. These 2 true latents are hidden in D testing latent dimensions, where D ranges from 6-10 due to my computer's limited capacity(in theory D can be much larger if wanted); 3. The trajectories are added with negative binomial noise to finally produce the (synthetic) neural dataset (shown in Fig.18-a). It is then trained to see if it can both find the number of true latent variables and infer the spherical structure from the non-reversible kernel. The results are shown in Fig. 18. From the plot of learnt scale parameters (Fig. 18-b), it is seen that the correct dimension is successfully recovered: 2 out of 10 parameters are significantly larger than the others. The latent rotatory trajectory is also recovered, although it lacks a little smoothness, which will be discussed with the performance of the original bGPFA.



(a) (b)

Figure 19: (a): Plot of learnt scale parameter; (b): The learnt latent trajectories(also 2500 iterations)

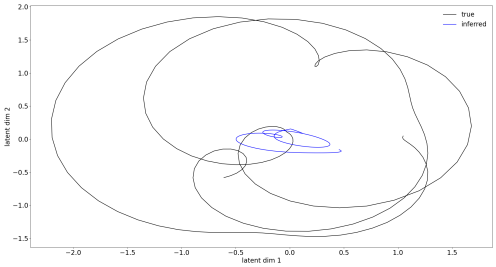
Surprise from bGPFA

The same task is conducted again with bGPFA. Surprisingly, bGPFA also perfectly recovers the latent trajectories (even more smooth than the result from bGPFADS) while also correct at the number of the true latent dimensions(results are shown in Fig. 19). Based on this task only, bGPFA seems to do an even better job at inference. However, although both methods successfully inferred the number of true latent variables, the magnitudes of the two learnt scale parameter in bGPFADS are significantly larger than those in bGPFA, which might suggest that bGPFADS are more confident about the number of the true dimensions. This might result from the learnt spherical dynamical structure.

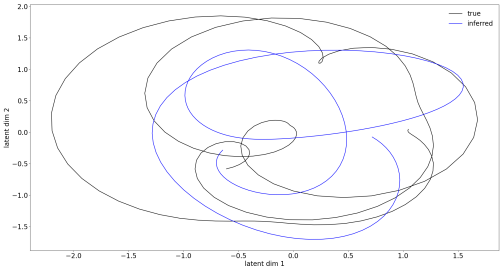
Further comparison is conducted in terms of the performance at different stages of iterations. The results are shown in Fig. 20 and 21. Again, bGPFA is able to learn faster the trajectories. This may be because of potentially different learning patterns of both methods. For bGPFADS, it seems to first detect the spherical structure and then gradually learn the amplitude/phase relationships, while bGPFA may just take this job as a simple Bayesian fitting. It seems in this task that bGPFA is again more effective. On the other hand, bGPFADS still have much larger values of the 2 learnt scale parameters than those of bGPFA, suggesting greater confidence in the number of the true latents.

6 Conclusions

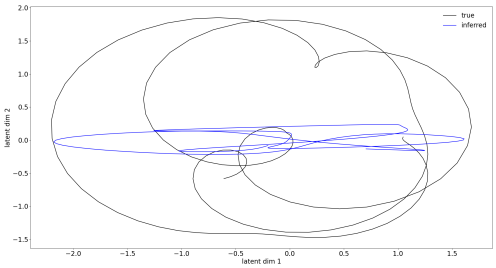
Extracting latent trajectories from high-dimensional observations can be very useful in studies in neuroscience: the brain undergoes various kinds of dynamics all the time, and it is impossible to collect simultaneously all the neural data and process them. Huge efforts have been made by researchers to develop such latent variable methods, among which is the Gaussian Process Factor Analysis(Yu et al. 2008). GPFA has been popular due to its ability to extract smooth trajectories, but it fails to provide these inferred latent variables with sufficient dynamical structure, preventing researchers from gaining insights into potential interactions between latent variables. This motivates the development of the covariance functions with the ability to detect potential dynamical structures(Rutten



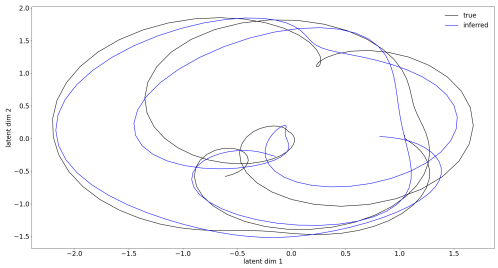
(a) Iteration = 50



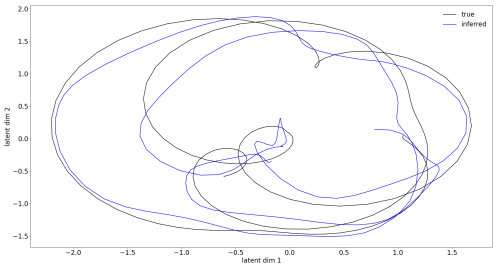
(b) Iteration = 50



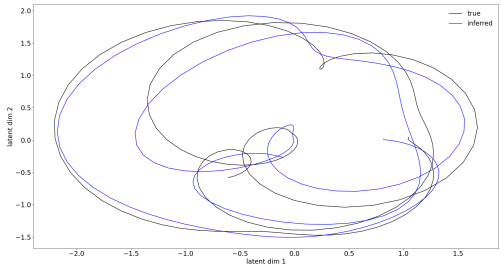
(c) Iteration = 300



(d) Iteration = 300



(e) Iteration = 1000

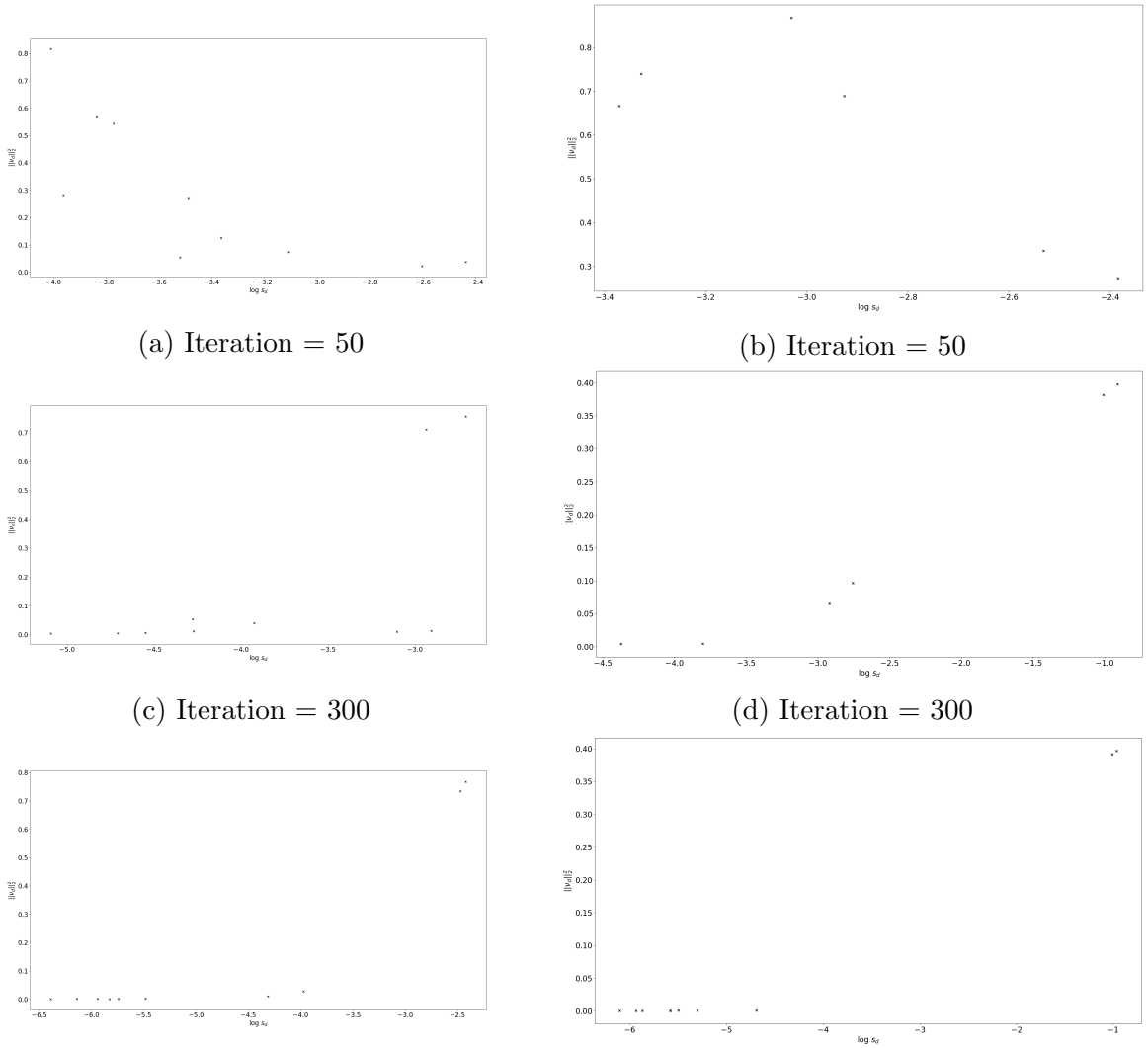


(f) Iteration = 1000

Figure 20: Trajectories inferred by bGPFADS(left column) and bGPFA(right column) respectively at 50, 300 and 1000 iterations.

et al. 2020) of neural data by enforcing a certain degree of non-reversibility such that the trajectories become unlikely to be produced in opposite directions, which therefore provides itself with characteristics of dynamical structures. In this project, I have thoroughly studied GPFADS and conducted several experiments. From experiments of sampling the trajectories of such non-reversible priors and planar regression tasks, both the advantages and limits of GPFADS have been illustrated: such non-reversible priors can make GPFADS very sensitive to spherical/sinusoidal/rotatory shapes (Rutten et al. 2020 reported that GPFADS found strongly rotatory structures in monkey M1 neural recordings), but this might also, in a way, limit GPFADS from discovering other forms of dynamical structures(as illustrated in section 5.4). I conclude that GPFADS is examined to be a very helpful tool in search of latent dynamics with rotatory structures.

In hope of enabling GPFADS to also infer the likely number of latent variables, Bayesian Gaussian Process Factor Analysis(Jensen, Kao, Stone, et al. 2021), a fully Bayesian yet scalable latent models were studied carefully. bGPFA's ability to infer the dimensionality



(a) Iteration = 50

(b) Iteration = 50

(c) Iteration = 300

(d) Iteration = 300

(e) Iteration = 1000

(f) Iteration = 1000

Figure 21: Learnt scale parameters inferred by bGPFADS(left column) and bGPFA(right column) respectively at 50, 300 and 1000 iterations.

comes from the extra prior on the linear readout matrix C (as introduced in section 4.2). Since this induces intractability, a nested variational inference method is applied where 2 ELBO were designed for maximum likelihood learning. Therefore, I have worked on incorporating non-reversible covariance function into the bGPFA framework. I examined the possibility to directly use the non-reversible kernel to be the square root of itself. This might sound discouraging but it actually works surprisingly well after examining its structure and its reversibility index. This is therefore used in implementing Bayesian Gaussian Process Factor Analysis with Dynamical Structure (bGPFADS). Planar bGPFADS were implemented and tested, where it successfully recovered the spherical trajectories while giving the correct number of latent variables. Surprisingly, bGPFA was able to discover the spherical trajectories quite successfully. It even exhibits the approximate shape of the true latent trajectories in the first 50 iterations. While it is a pity that the high-dimensional bGPFADS is not built successfully, efforts will be put into its coding work after the end of the project. Besides, the concern mentioned in Appendix C has no fix in the literature so far, I will also try to find an analytical solution to quantify the non-

reversibility index of the squared non-reversible kernel. Another possible fix is to use a kernel with its Hilbert transform also converges quickly(see the shape of spectral mixture in Rutten et al. 2020's Appendix D, Fig.5).

7 Appendix

A Hilbert transforms of commonly used scalar GP kernels

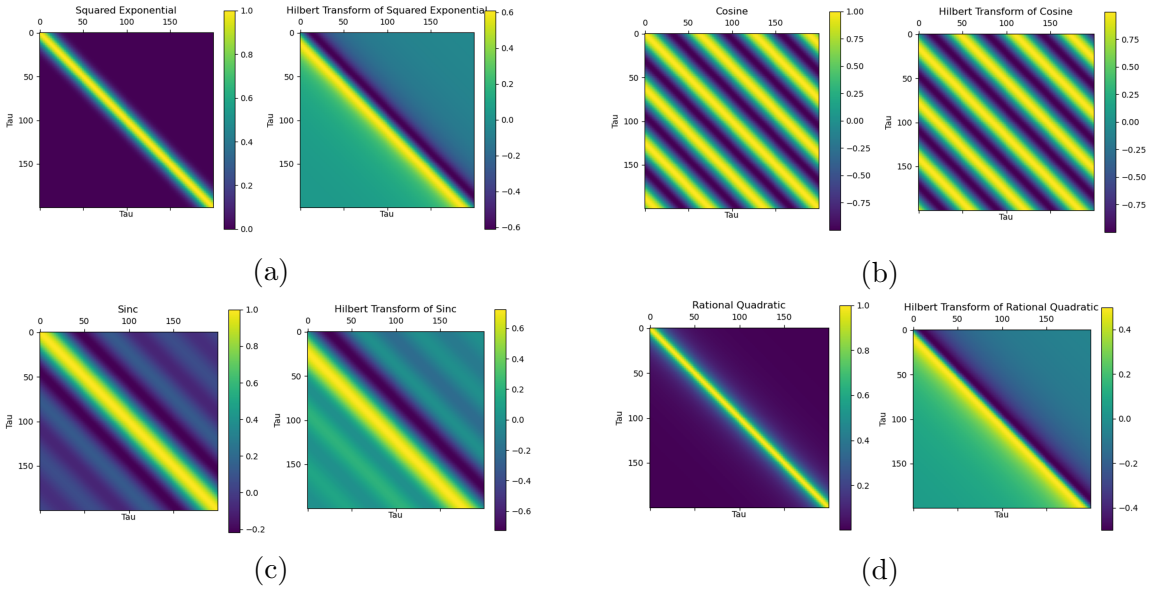


Figure 22: Kernel plots all the 4 kernels listed and their corresponding Hilbert transform plots

Table 1 gives a list of commonly used GP kernels and their corresponding Hilbert transforms. Note the slight difference between the kernels and their corresponding Hilbert transformed versions from the matrix plots in Fig. 22

Stationary Kernels	Hilbert Transform
$\exp(-\tau/2)$	$2\pi^{-0.5}D(\tau/\sqrt{2})$
$\cos(w_0\tau)$	$\sin(w_0\tau)$
$\sin(w_0\tau)/(w_0\tau)$	$[1\text{-}\cos(w_0\tau)]/\ (w_0\tau)$
$(1+\tau^2)^{-1}$	$\tau(1+\tau^2)^{-1}$

Table 1: Hilbert transforms of usual scalar GP kernels, where D represents Dawson function

B Construction of high-dimensional non-reversible covariance functions

In section 4.1 Fig. 6, the structure of high-dimensional non-reversible covariance matrix is built in such a way that each of the $M/2$ planes couple 2 dimensions together and

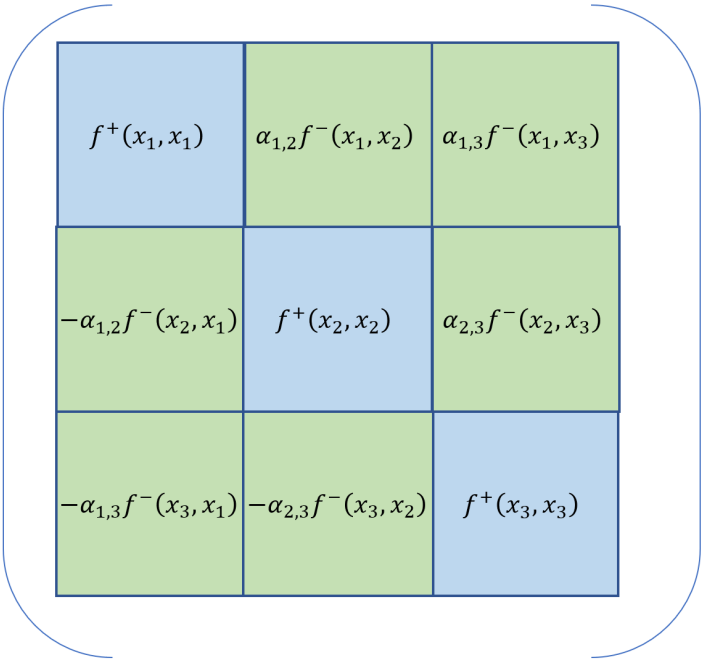


Figure 23: I plot this more intuitive matrix to show the form of Eq. 45 in Rutten et al. 2020.

allow them to share "information" between each other to break reversibility. Such form is used throughout the project, including the conducting the planar non-reversible GP regression and the implementation of bGPFADS, but how exactly does it end up having a structure like this?

If we assume $\rho_{ij} = 0$, $\sigma_{ij,1} = \sigma_{ij,2} = \sigma$ and all the scalar covariance functions are all the same, then the high-dimensional covariance matrix is over-parameterised. According to the simplified Eq. 15, the covariance matrix can then be build in Eq. 45 in Rutten et al. 2020. To better visualise its structure, it is illustrated in 3D in Fig. 23. Since the green blocks (corresponding to A^- in ibid.) is antisymmetric, there exists a unitary transformation of the latent space in which this covariance is restructured to form like Fig. 6. In GPFADS, this unitary transformation can be absorbed in the mixing matrix C (Eq. 1). Therefore, with this structure there are only $M/2$ parameters (M is the number of the latent dimensions), and each of them functions in one plane(2 dimensions). "In other words, one can always rotate the latent space and directly parameterise a set of independent planes, in which case one must enforce $|w_i| < 1$"(ibid.), where $\{\pm jw_1, \pm jw_1, \dots\}$ are the imaginary conjugate eigenvalues of A^-, i.e. the matrix consisting of the green blocks only. **I find it very important that, this sets of w can be considered as the origin, or the true interpretation of the hyperparameters $\boldsymbol{\alpha}$ in the structure of Fig. 6**, i.e. the α is expected to learn/capture the imaginary conjugate eigenvalues of A^- (the matrix consisting of the green blocks only).

It is also important to realise that, these assumptions made above lead to a simultaneously spherical process, and these assumptions were made also during regression experiments in section 5.2, which is essentially why: 1. the prior kernels always show strong rotatory structure (e.g. in Fig. 9); 2. the predicted shape is still rotatory or sinusoidal, and fails to learn to the concave shape in Fig. 13.

C Evaluation of the reversibility index for the non-reversible matrix

During this evaluation of the reversibility index for the non-reversible matrix and its squared version, it is very important to make sure the range of τ (shown in Fig. 15) and its number of samples are large enough. This is because ideally the evaluation of Frobenius norm is analogous to calculate the **square integral** of the covariance function (and its corresponding Hilbert transform) from 0 to positive infinity (Fig. 16), thus the sampling size and range is crucial to the value of the Frobenius norms and thus the non-reversibility measure, ζ. Essentially, it is about effective sampling. There is not much of a problem for the squared exponential kernel as the value converges to 0 zero very quickly, but this is not the case for the Hilbert transform function. In Fig. 24,I plot the curves of both squared exponential and its Hilbert transform. Clearly, the value of the Hilbert transform converges much more slowly than the squared exponential function. This indicates that, we really need to implement τ with a large enough range in order to find accurate value of the reversibility index from numerical computation.

However, this is still not the end of story because of the existence of other hyperparameters, which could also influence the shape of these functions and thus influence the numerical calculation of squared integral (and thus the Frobenius norm and the reversibility index). Fig. 25 shows the different shapes of the squared exponential and Hilbert transform with different lengthscales/signal variance(A). Looking at the two ends of the x-axis, it can be observed that the squared exponential still converges very quickly but Hilbert transform function's convergence can be influenced such that the numerical squared integral could deviate significantly from the true value, which is critical as the norm from the Hilbert transform directly influence the numerator of the reversibility index. As so far there is no analytical solution to the index of the squared non-reversible matrix in the literature, **this would cause a problem when we want to know accurately the index of the inferred trajectories to gain knowledge of how much potential dynamical structure is embedded.** Consider the case where the training is finished and you have obtained values of α. You cannot directly use this α to interpret how strongly non-reversible the latent trajectories are, since it is the α from the square root matrix. Eq. 10 can only approximate the non-reversibility, and numerically calculating this index might cause the problems mentioned above: there may be a significant difference in the numerical value of index and the true index. This motivates us to find the index of the squared non-reversible matrix analytically.

In summary, too few samples, too small range of τ and the hyperparameters all could lead to large deviations in the difference between the non-reversibility indices for the covariance matrix and its squared version. Therefore, during my experiments, these were carefully taken into account.

D A simple derivation of Eq. 10 by planar non-reversible kernel with $\rho = 0$

Eq. 10 is derived here using a planar non-reversible kernel. I assume that the covariance function used in both dimensions are the same, and that the kernel size is very large to satisfy the definition in Eq. 5. I also let $\rho = 0$ in this case for neat illustration(Eq. 10

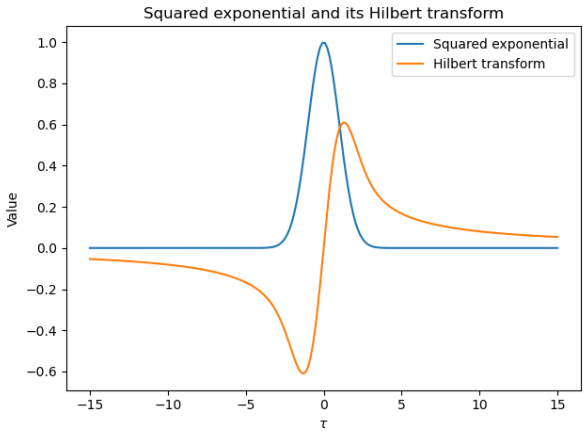
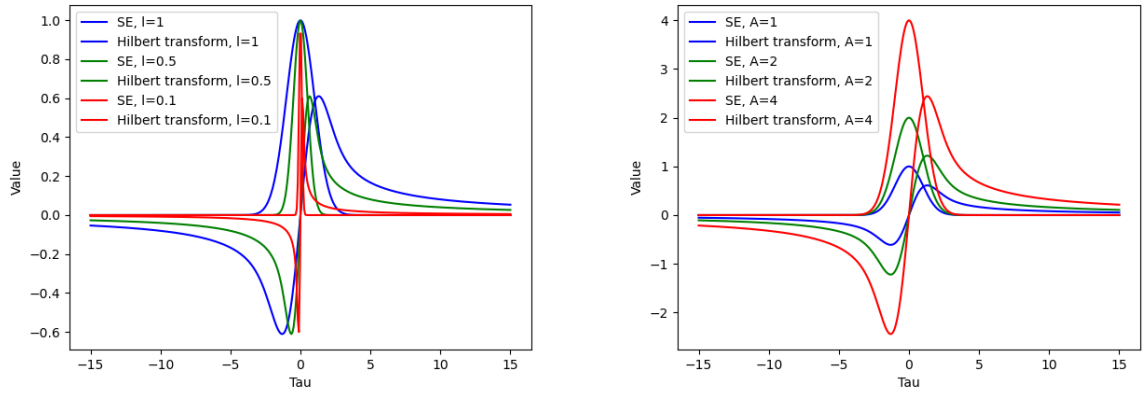


Figure 24: Squared exponential and its Hilbert transform functions with different lengthscales



(a) (b)

Figure 25: (a): Squared exponential and its Hilbert transform functions with different lengthscales; (b): Squared exponential and its Hilbert transform functions with different signal variance

also holds for $\rho \neq 0$) Note also that, the squared integral of the kernel function from 0 to positive infinity equals the squared integral of its corresponding Hilbert transform with the same interval(easy to derive and verify with numerical computation). Therefore an ideal plannar non-reversible matrix can be built as illustrated in Fig.26. Now compute $K(\tau)-K(-\tau)$ and $K(\tau)+K(-\tau)$ respectively, which are illustrated in Fig. 27.Therefore,

$$\zeta = \frac{\sqrt{8\alpha^2\sigma_1^2\sigma_2^2 \times N_{(-)}^2}}{\sqrt{(4\sigma_1^4 + 4\sigma_2^4) \times N_{(+)}^2}}, \text{ where} \tag{24}$$

$N_{(-)}$ and $N_{(+)}$ are Frobenius norms for each small block of the Hilbert transform function matrix and covariance matrix respectively. As mentioned previously, since the squared integral of the kernel function from 0 to positive infinity equals the squared integral of its corresponding Hilbert transform with the same integral, one can derive easily that $N_{(-)} = N_{(+)}$. Therefore,

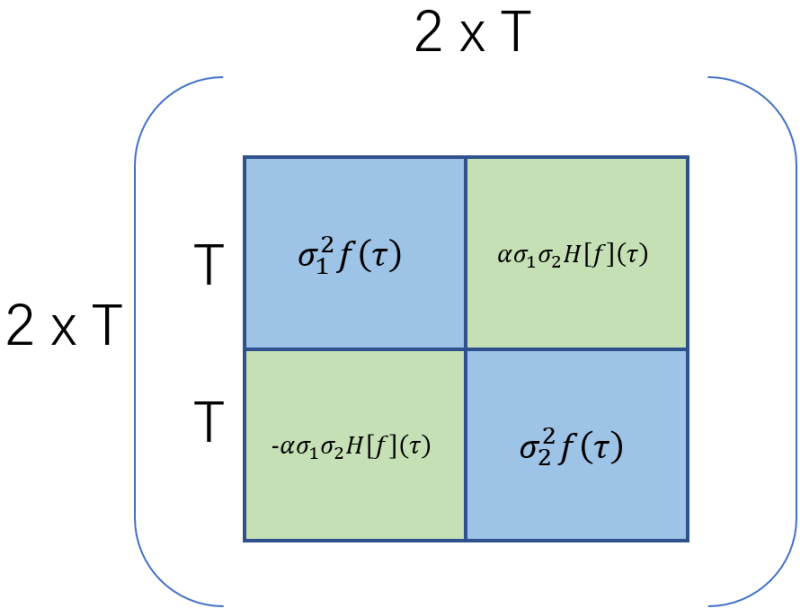


Figure 26: Schematic matrix plot, the notions are the same with Eq. 9

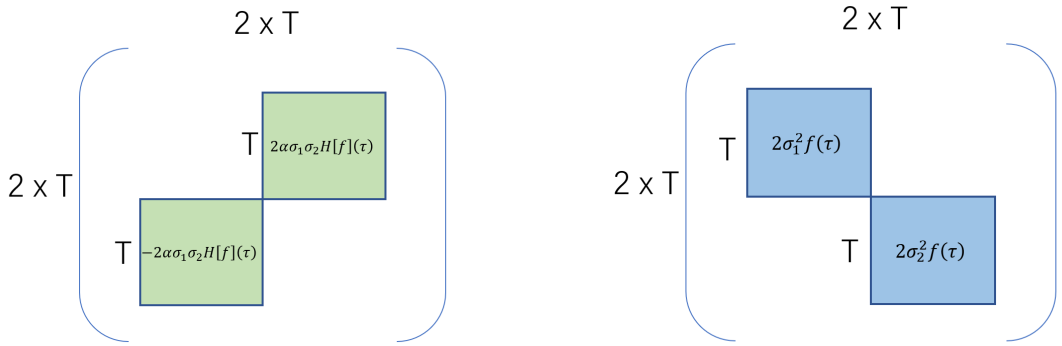


Figure 27: (a): $K(\tau) - K(-\tau)$; (b): $K(\tau) + K(-\tau)$, note the "2" in both plots

$$\begin{aligned} \zeta &= \frac{\sqrt{2\alpha^2\sigma_1^2\sigma_2^2}}{\sqrt{(\sigma_1^4 + 4\sigma_2^4)}} \\ &= |\alpha| \left(\frac{2}{(\sigma_1/\sigma_2)^2 + (\sigma_2/\sigma_1)^2} \right)^{1/2} \end{aligned} \tag{25}$$

which is the same as Eq. 10 with $\rho = 0$. Once again, here I let $\rho = 0$ only for the purpose of neat algebra illustration.

Bibliography

Allen, EJ, J Baglama, and SK Boyd (2000). "Numerical approximation of the product of the square root of a matrix with a vector". In: *Linear Algebra and its Applications* 310.1-3, pp. 167–181.

Churchland, Mark M et al. (2012). "Neural population dynamics during reaching". In: *Nature* 487.7405, pp. 51–56.

Hensman, James et al. (2015). "MCMC for variationally sparse Gaussian processes". In: *Advances in Neural Information Processing Systems* 28.

Jensen, Kristopher, Ta-Chu Kao, Jasmine Stone, et al. (2021). "Scalable Bayesian GPFA with automatic relevance determination and discrete noise models". In: *Advances in Neural Information Processing Systems* 34.

Jensen, Kristopher, Ta-Chu Kao, Marco Tripodi, et al. (2020). "Manifold GPLVMs for discovering non-Euclidean latent structure in neural data". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33, pp. 22580–22592.

Khan, Haider Adnan (2011). "Characterization of Flow Patterns in MRI Phase Contrast Data". PhD thesis. PhD Dissertation.

Murray, Iain and Ryan P Adams (2010). "Slice sampling covariance hyperparameters of latent Gaussian models". In: *Advances in neural information processing systems* 23.

Rutten, Virginia et al. (2020). "Non-reversible Gaussian processes for identifying latent dynamical structure in neural data". In: *Advances in neural information processing systems* 33, pp. 9622–9632.

Wainwright, Martin J, Michael I Jordan, et al. (2008). "Graphical models, exponential families, and variational inference". In: *Foundations and Trends® in Machine Learning* 1.1–2, pp. 1–305.

Yu, Byron M et al. (2008). "Gaussian-process factor analysis for low-dimensional single-trial analysis of neural population activity". In: *Advances in neural information processing systems* 21.