Spatio-Temporal Variational Gaussian Processes

Oliver Hamelijnck* [1] William J. Wilkinson* [2] Niki Andreas Loppi [3]
Arno Solin [2] Theodoros Damoulas [1]

[1]University of Warwick, The Alan Turing Institute

[2]Aalto University

[3]Nvidia

NeurIPS 2021

Motivation

- We want to use Gaussian processes to model spatio-temporal phenomena
- However the computational burden of GPs can make this difficult
- Two popular methods to handle this are sparse GPs and state-space GPs
- But sparse GPs over smooth on large datasets and state-space GPs can computationally struggle with a large number of spatial points
- In this work we effectively combine both methods to attempt to get the best of both worlds!

Motivation

- We want to use Gaussian processes to model spatio-temporal phenomena
- However the computational burden of GPs can make this difficult
- Two popular methods to handle this are sparse GPs and state-space GPs
- But sparse GPs over smooth on large datasets and state-space GPs can computationally struggle with a large number of spatial points
- In this work we effectively combine both methods to attempt to get the best of both worlds!

Motivation

- We want to use Gaussian processes to model spatio-temporal phenomena
- However the computational burden of GPs can make this difficult
- Two popular methods to handle this are sparse GPs and state-space GPs
- But sparse GPs over smooth on large datasets and state-space GPs can computationally struggle with a large number of spatial points
- In this work we effectively combine both methods to attempt to get the best of both worlds!

Motivation

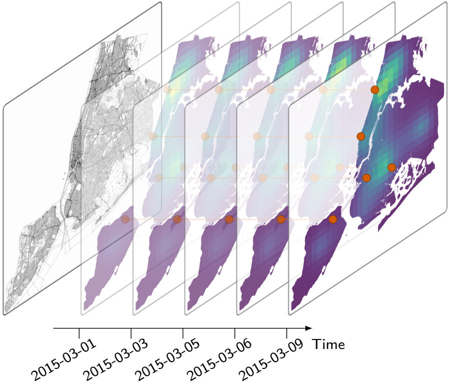
- We want to use Gaussian processes to model spatio-temporal phenomena
- However the computational burden of GPs can make this difficult
- Two popular methods to handle this are sparse GPs and state-space GPs
- But sparse GPs over smooth on large datasets and state-space GPs can computationally struggle with a large number of spatial points
- In this work we effectively combine both methods to attempt to get the best of both worlds!

Motivation

- We want to use Gaussian processes to model spatio-temporal phenomena
- However the computational burden of GPs can make this difficult
- Two popular methods to handle this are sparse GPs and state-space GPs
- But sparse GPs over smooth on large datasets and state-space GPs can computationally struggle with a large number of spatial points
- In this work we effectively combine both methods to attempt to get the best of both worlds!

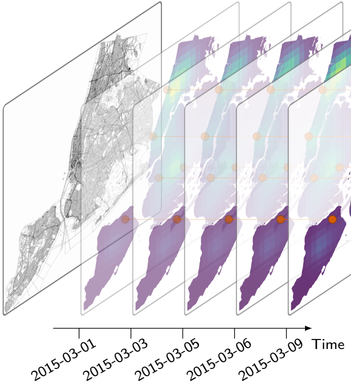
Overview - TLDR

- Propose a Sparse Variational GP that scales linearly in the number of temporal points
- The approximate posterior is represented as a state-space model
- The full ELBO can be computed efficiently through Kalman filtering and smoothing
- Recover the standard SVGP posterior at a fraction of the computational cost



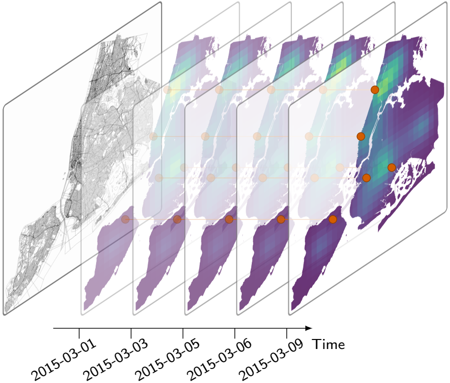
Overview - TLDR

- Propose a Sparse Variational GP that scales linearly in the number of temporal points
- The approximate posterior is represented as a state-space model
- The full ELBO can be computed efficiently through Kalman filtering and smoothing
- Recover the standard SVGP posterior at a fraction of the computational cost



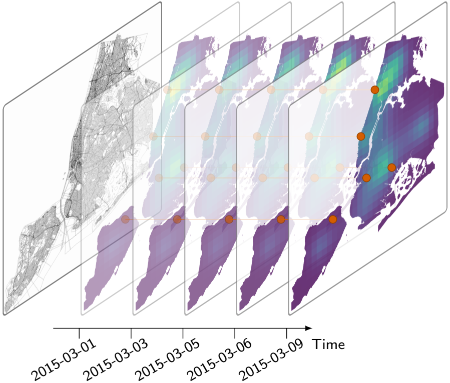
Overview - TLDR

- Propose a Sparse Variational GP that scales linearly in the number of temporal points
- The approximate posterior is represented as a state-space model
- The full ELBO can be computed efficiently through Kalman filtering and smoothing
- Recover the standard SVGP posterior at a fraction of the computational cost



Overview - TLDR

- Propose a Sparse Variational GP that scales linearly in the number of temporal points
- The approximate posterior is represented as a state-space model
- The full ELBO can be computed efficiently through Kalman filtering and smoothing
- Recover the standard SVGP posterior at a fraction of the computational cost



Talk Outline

- Gaussian Processes
- Sparse Variational Gaussian Processes
- State Space Gaussian Processes
- Natural Gradients as Conjugate Operations
- Spatio-temporal Variational GPs
- Experiments
- Conclusion

Gaussian Processes

Gaussian Processes - Example

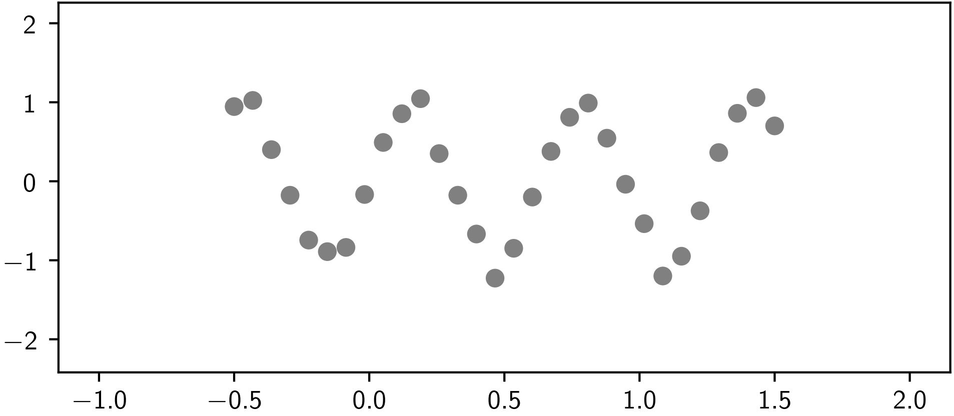


Figure: Observations corrupted by Gaussian noise.

Gaussian Processes - Example

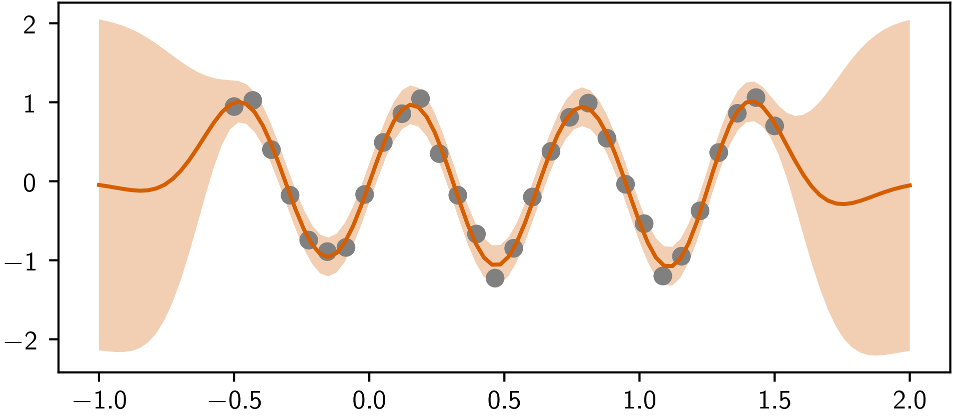


Figure: GP posterior fit.

Gaussian Processes

- Gaussian Processes are priors over *functions*, Rasmussen and Williams [2006]
- Infinite-dimensional extensions of multivariate Gaussians
- Fully defined by a mean and kernel function

$$\mathbf{f} \sim \mathcal{GP}(\mu(\mathbf{X}), \mathbf{K}(\mathbf{X})) \tag{1}$$

- Let $\mathbf{X} \in \mathbb{R}^{N \times D}, \mathbf{Y} \in \mathbb{R}^{N \times 1}$ be input-output observations
- Inference follows standard Bayesian machinery

$$\underbrace{p(\mathbf{f} \mid \mathbf{Y}, \mathbf{X})}_{\text{Posterior}} \propto \underbrace{p(\mathbf{Y} \mid \mathbf{f})}_{\text{Likelihood}} \underbrace{p(\mathbf{f})}_{\text{Prior}} \tag{2}$$

- Inference and training has a $\mathcal{O}(N^3)$ computational complexity

Gaussian Processes

- Gaussian Processes are priors over *functions*, Rasmussen and Williams [2006]
- Infinite-dimensional extensions of multivariate Gaussians
- Fully defined by a mean and kernel function

$$\mathbf{f} \sim \mathcal{GP}(\mu(\mathbf{X}), \mathbf{K}(\mathbf{X})) \tag{1}$$

- Let $\mathbf{X} \in \mathbb{R}^{N \times D}, \mathbf{Y} \in \mathbb{R}^{N \times 1}$ be input-output observations
- Inference follows standard Bayesian machinery

$$\underbrace{p(\mathbf{f} \mid \mathbf{Y}, \mathbf{X})}_{\text{Posterior}} \propto \underbrace{p(\mathbf{Y} \mid \mathbf{f})}_{\text{Likelihood}} \underbrace{p(\mathbf{f})}_{\text{Prior}} \tag{2}$$

- Inference and training has a $\mathcal{O}(N^3)$ computational complexity

Gaussian Processes

- Gaussian Processes are priors over *functions*, Rasmussen and Williams [2006]
- Infinite-dimensional extensions of multivariate Gaussians
- Fully defined by a mean and kernel function

$$\mathbf{f} \sim \mathcal{GP}(\mu(\mathbf{X}), \mathbf{K}(\mathbf{X})) \tag{1}$$

- Let $\mathbf{X} \in \mathbb{R}^{N \times D}, \mathbf{Y} \in \mathbb{R}^{N \times 1}$ be input-output observations
- Inference follows standard Bayesian machinery

$$\underbrace{p(\mathbf{f} \mid \mathbf{Y}, \mathbf{X})}_{\text{Posterior}} \propto \underbrace{p(\mathbf{Y} \mid \mathbf{f})}_{\text{Likelihood}} \underbrace{p(\mathbf{f})}_{\text{Prior}} \tag{2}$$

- Inference and training has a $\mathcal{O}(N^3)$ computational complexity

Gaussian Processes

- Gaussian Processes are priors over *functions*, Rasmussen and Williams [2006]
- Infinite-dimensional extensions of multivariate Gaussians
- Fully defined by a mean and kernel function

$$\mathbf{f} \sim \mathcal{GP}(\mu(\mathbf{X}), \mathbf{K}(\mathbf{X})) \tag{1}$$

- Let $\mathbf{X} \in \mathbb{R}^{N \times D}, \mathbf{Y} \in \mathbb{R}^{N \times 1}$ be input-output observations
- Inference follows standard Bayesian machinery

$$\underbrace{p(\mathbf{f} \mid \mathbf{Y}, \mathbf{X})}_{\text{Posterior}} \propto \underbrace{p(\mathbf{Y} \mid \mathbf{f})}_{\text{Likelihood}} \underbrace{p(\mathbf{f})}_{\text{Prior}} \tag{2}$$

- Inference and training has a $\mathcal{O}(N^3)$ computational complexity

Gaussian Processes

- Gaussian Processes are priors over *functions*, Rasmussen and Williams [2006]
- Infinite-dimensional extensions of multivariate Gaussians
- Fully defined by a mean and kernel function

$$\mathbf{f} \sim \mathcal{GP}(\mu(\mathbf{X}), \mathbf{K}(\mathbf{X})) \tag{1}$$

- Let $\mathbf{X} \in \mathbb{R}^{N \times D}$, $\mathbf{Y} \in \mathbb{R}^{N \times 1}$ be input-output observations
- Inference follows standard Bayesian machinery

$$\underbrace{p(\mathbf{f} \mid \mathbf{Y}, \mathbf{X})}_{\text{Posterior}} \propto \underbrace{p(\mathbf{Y} \mid \mathbf{f})}_{\text{Likelihood}} \underbrace{p(\mathbf{f})}_{\text{Prior}} \tag{2}$$

- Inference and training has a $\mathcal{O}(N^3)$ computational complexity

Gaussian Processes

– Gaussian Processes are priors over *functions*, Rasmussen and Williams [2006]

– Infinite-dimensional extensions of multivariate Gaussians

– Fully defined by a mean and kernel function

$$\mathbf{f} \sim \mathcal{GP}(\mu(\mathbf{X}), \mathbf{K}(\mathbf{X})) \tag{1}$$

– Let $\mathbf{X} \in \mathbb{R}^{N \times D}$, $\mathbf{Y} \in \mathbb{R}^{N \times 1}$ be input-output observations

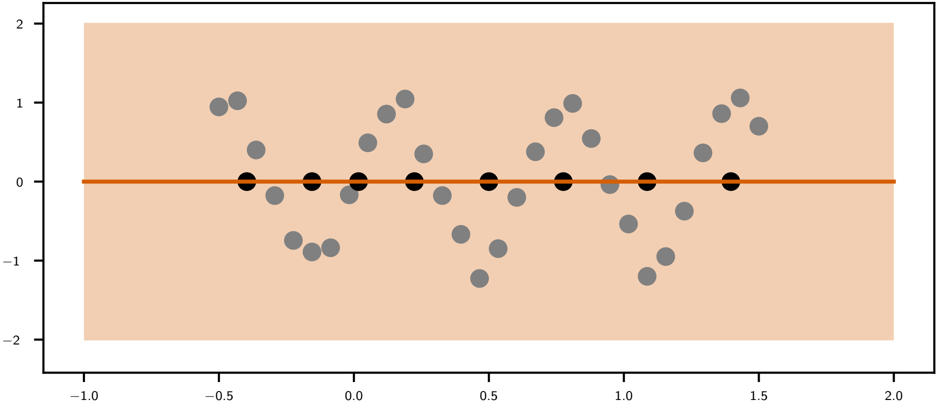
– Inference follows standard Bayesian machinery

$$\underbrace{p(\mathbf{f} \mid \mathbf{Y}, \mathbf{X})}_{\text{Posterior}} \propto \underbrace{p(\mathbf{Y} \mid \mathbf{f})}_{\text{Likelihood}} \underbrace{p(\mathbf{f})}_{\text{Prior}} \tag{2}$$

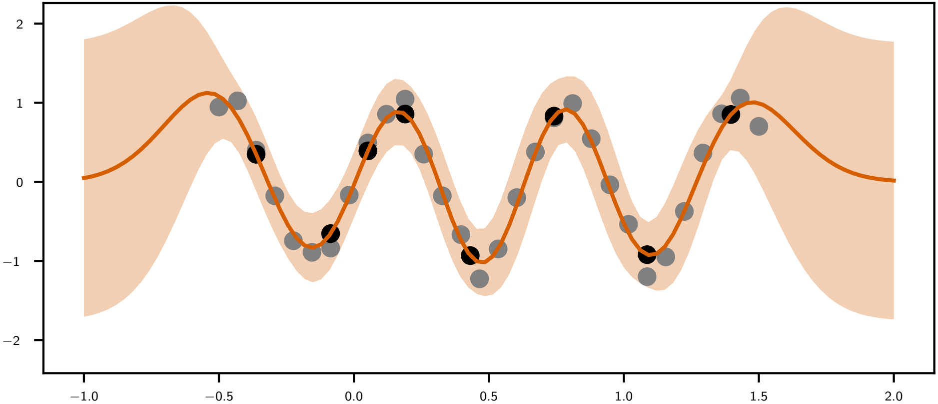
– Inference and training has a $\mathcal{O}(N^3)$ computational complexity

Sparse Variational GPs

Sparse Variational GPs - Example



Sparse Variational GPs - Example



Variational Sparse GPs

- Augment the prior with inducing points and inducing locations $\mathbf{Z} \in \mathbb{R}^{M \times D}$, $M << N$

$$p(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})\, p(\mathbf{u}) \tag{3}$$

  with

$$p(\mathbf{u}) = \mathrm{N}(\mathbf{u} \mid 0, \mathbf{K}(\mathbf{Z}, \mathbf{Z})), \quad p(\mathbf{f} \mid \mathbf{u}) = \mathrm{N}(\mathbf{f} \mid \mathbf{K}(\mathbf{X}, \mathbf{Z})\mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\mathbf{u}, \mathbf{Q}) \tag{4}$$

  and

$$\mathbf{Q} = \mathbf{K}(\mathbf{X}, \mathbf{X}) - \mathbf{K}(\mathbf{X}, \mathbf{Z})\, \mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\, \mathbf{K}(\mathbf{Z}, \mathbf{X}) \tag{5}$$

- VI provide a way to *learn* the inducing points!
- Minimise the distance between approximate posterior $q(\mathbf{f}, \mathbf{u})$ and the true $p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y})$

$$\underset{q(\mathbf{f},\mathbf{u})}{\arg\min}\, \mathrm{KL}\left[\, q(\mathbf{f}, \mathbf{u}) \,||\, p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y})\,\right] \quad \rightarrow \quad \underset{q(\mathbf{f},\mathbf{u})}{\arg\max}\, \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})p(\mathbf{f} \mid \mathbf{u})p(\mathbf{u})}{q(\mathbf{f} \mid \mathbf{u})q(\mathbf{u})}\right]$$

- But $p(\mathbf{f} \mid \mathbf{u})$ is N dimensional $\rightarrow$ cubic computational complexity!

Variational Sparse GPs

– Augment the prior with inducing points and inducing locations $\mathbf{Z} \in \mathbb{R}^{M \times D}$, $M << N$

$$p(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})\, p(\mathbf{u}) \tag{3}$$

with

$$p(\mathbf{u}) = \mathrm{N}(\mathbf{u} \mid 0, \mathbf{K}(\mathbf{Z}, \mathbf{Z})), \quad p(\mathbf{f} \mid \mathbf{u}) = \mathrm{N}(\mathbf{f} \mid \mathbf{K}(\mathbf{X}, \mathbf{Z})\mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\mathbf{u}, \mathbf{Q}) \tag{4}$$

and

$$\mathbf{Q} = \mathbf{K}(\mathbf{X}, \mathbf{X}) - \mathbf{K}(\mathbf{X}, \mathbf{Z})\, \mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\, \mathbf{K}(\mathbf{Z}, \mathbf{X}) \tag{5}$$

– VI provide a way to *learn* the inducing points!

– Minimise the distance between approximate posterior $q(\mathbf{f}, \mathbf{u})$ and the true $p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y})$

$$\underset{q(\mathbf{f},\mathbf{u})}{\arg\min}\, \mathrm{KL}\left[\, q(\mathbf{f}, \mathbf{u}) \,||\, p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y}) \,\right] \quad \to \quad \underset{q(\mathbf{f},\mathbf{u})}{\arg\max}\, \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})p(\mathbf{f} \mid \mathbf{u})p(\mathbf{u})}{q(\mathbf{f} \mid \mathbf{u})q(\mathbf{u})}\right]$$

– But $p(\mathbf{f} \mid \mathbf{u})$ is N dimensional $\to$ cubic computational complexity!

Variational Sparse GPs

- Augment the prior with inducing points and inducing locations $\mathbf{Z} \in \mathbb{R}^{M \times D}$, $M << N$

$$p(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})\, p(\mathbf{u}) \tag{3}$$

with

$$p(\mathbf{u}) = \mathrm{N}(\mathbf{u} \mid 0, \mathbf{K}(\mathbf{Z}, \mathbf{Z})), \quad p(\mathbf{f} \mid \mathbf{u}) = \mathrm{N}(\mathbf{f} \mid \mathbf{K}(\mathbf{X}, \mathbf{Z})\mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\mathbf{u}, \mathbf{Q}) \tag{4}$$

and

$$\mathbf{Q} = \mathbf{K}(\mathbf{X}, \mathbf{X}) - \mathbf{K}(\mathbf{X}, \mathbf{Z})\, \mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\, \mathbf{K}(\mathbf{Z}, \mathbf{X}) \tag{5}$$

- VI provide a way to *learn* the inducing points!
- Minimise the distance between approximate posterior $q(\mathbf{f}, \mathbf{u})$ and the true $p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y})$

$$\underset{q(\mathbf{f},\mathbf{u})}{\arg\min}\, \mathrm{KL}\left[\, q(\mathbf{f}, \mathbf{u}) \,||\, p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y}) \,\right] \quad \rightarrow \quad \underset{q(\mathbf{f},\mathbf{u})}{\arg\max}\, \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})p(\mathbf{f} \mid \mathbf{u})p(\mathbf{u})}{q(\mathbf{f} \mid \mathbf{u})q(\mathbf{u})}\right]$$

- But $p(\mathbf{f} \mid \mathbf{u})$ is N dimensional $\rightarrow$ cubic computational complexity!

Variational Sparse GPs

- Augment the prior with inducing points and inducing locations $\mathbf{Z} \in \mathbb{R}^{M \times D}$, $M << N$

$$p(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})\, p(\mathbf{u}) \tag{3}$$

  with

$$p(\mathbf{u}) = \mathrm{N}(\mathbf{u} \mid 0, \mathbf{K}(\mathbf{Z}, \mathbf{Z})), \quad p(\mathbf{f} \mid \mathbf{u}) = \mathrm{N}(\mathbf{f} \mid \mathbf{K}(\mathbf{X}, \mathbf{Z})\mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\mathbf{u}, \mathbf{Q}) \tag{4}$$

  and

$$\mathbf{Q} = \mathbf{K}(\mathbf{X}, \mathbf{X}) - \mathbf{K}(\mathbf{X}, \mathbf{Z})\, \mathbf{K}(\mathbf{Z}, \mathbf{Z})^{-1}\, \mathbf{K}(\mathbf{Z}, \mathbf{X}) \tag{5}$$

- VI provide a way to *learn* the inducing points!
- Minimise the distance between approximate posterior $q(\mathbf{f}, \mathbf{u})$ and the true $p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y})$

$$\arg\min_{q(\mathbf{f}, \mathbf{u})} \mathrm{KL}\left[\, q(\mathbf{f}, \mathbf{u}) \,||\, p(\mathbf{f}, \mathbf{u} \mid \mathbf{Y}) \,\right] \quad \rightarrow \quad \arg\max_{q(\mathbf{f}, \mathbf{u})} \mathbb{E}_{q(\mathbf{f}, \mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})p(\mathbf{f} \mid \mathbf{u})p(\mathbf{u})}{q(\mathbf{f} \mid \mathbf{u})q(\mathbf{u})}\right]$$

- **But $p(\mathbf{f} \mid \mathbf{u})$ is N dimensional $\rightarrow$ cubic computational complexity!**

Variational Sparse GPs

- For computational efficient inference Titsias [2009] proposed to define as:

$$q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})\, q(\mathbf{u})$$

where $q(\mathbf{u}) = \mathrm{N}\left(\mathbf{u} \mid \mathbf{m}, \mathbf{S} \right)$ is a free-form Gaussian:

$$\text{ELBO} = \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})\, \cancel{p(\mathbf{f} \mid \mathbf{u})}\, p(\mathbf{u})}{\cancel{p(\mathbf{f} \mid \mathbf{u})}\, q(\mathbf{u})}\right] \tag{6}$$

- Hensman et al. [2013] extended this to the stochastic VI case:

$$\text{ELBO} = \sum_{n}^{N} \mathbb{E}_{q(\mathbf{f}_n)}\left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right] - \mathrm{KL}\left[q(\mathbf{u}) || p(\mathbf{u})\right] \tag{7}$$

- Computable in $\mathcal{O}(NM^2 + M^3)$ or $\mathcal{O}(M^3)$ with mini-batching

Variational Sparse GPs

- For computational efficient inference Titsias [2009] proposed to define as:

$$q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f} \mid \mathbf{u})\, q(\mathbf{u})$$

where $q(\mathbf{u}) = \mathrm{N}\left(\mathbf{u} \mid \mathbf{m}, \mathbf{S} \right)$ is a free-form Gaussian:

$$\mathrm{ELBO} = \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})\, \cancel{p(\mathbf{f} \mid \mathbf{u})}\, p(\mathbf{u})}{\cancel{p(\mathbf{f} \mid \mathbf{u})}\, q(\mathbf{u})}\right] \tag{6}$$

- Hensman et al. [2013] extended this to the stochastic VI case:

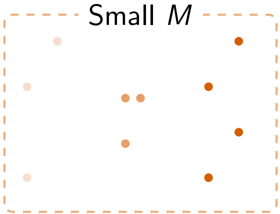
$$\mathrm{ELBO} = \sum_{n}^{N} \mathbb{E}_{q(\mathbf{f}_n)}\left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right] - \mathrm{KL}\left[q(\mathbf{u}) || p(\mathbf{u})\right] \tag{7}$$

- Computable in $\mathcal{O}(NM^2 + M^3)$ or $\mathcal{O}(M^3)$ with mini-batching

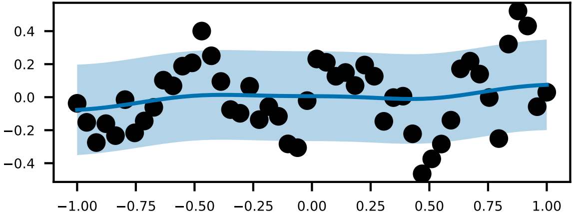
The Problem with SVGP for Time Series

Sparse Variational Gaussian Processes: The Problem

- With a low number of inducing points the GP cannot capture the structure

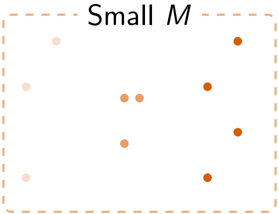


$\longrightarrow$

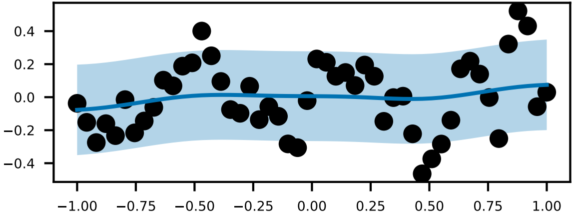


Sparse Variational Gaussian Processes: The Problem

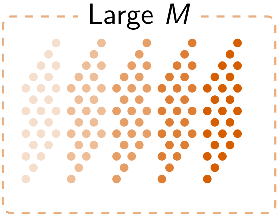
- With a low number of inducing points the GP cannot capture the structure



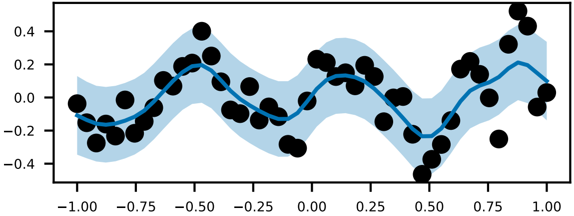
$\rightarrow$



- Assumed that $M << N$, which is not always appropriate!



$\rightarrow$



State-Space GPs

State Space GPs - Temporal Setting

For Markov kernels a GP $\mathbf{f}$ is the solution to a LTI-SDE, Särkkä and Solin [2019]:

$$\begin{aligned} f(\mathbf{t}) &\sim \mathcal{GP}(0, \mathbf{K}_t), \\ \mathbf{Y}_k &\sim p(\mathbf{Y}_k \mid f(\mathbf{t}_k)) \end{aligned} \quad \Rightarrow \quad \begin{aligned} \bar{\mathbf{f}}_k &= \mathbf{A}_k \bar{\mathbf{f}}_{k-1} + \mathbf{q}_{k-1}, \\ \mathbf{Y}_k &\sim p(\mathbf{Y}_k \mid \mathbf{H} \;\; \bar{\mathbf{f}}_k) \end{aligned}$$

where k represents time and $\bar{\mathbf{f}}_k \in R^d$ is a vector of derivatives of $\mathbf{f}$, and $q_k \sim \mathrm{N}(0, Q_k)$

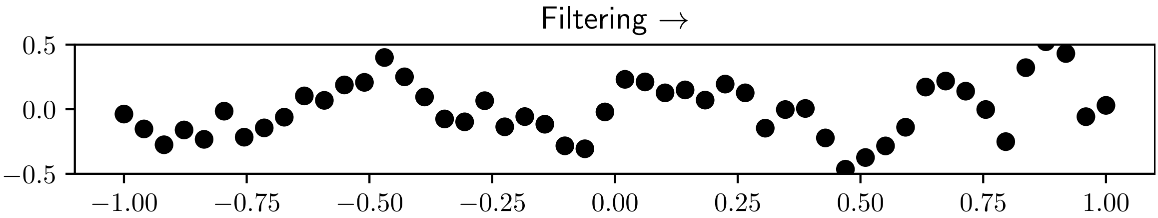
State Space GPs - Temporal Setting

For Markov kernels a GP **f** is the solution to a LTI-SDE, Särkkä and Solin [2019]:

$$f(\mathbf{t}) \sim \mathcal{GP}(0, \mathbf{K}_t), \qquad \Rightarrow \qquad \bar{\mathbf{f}}_k = \mathbf{A}_k \bar{\mathbf{f}}_{k-1} + \mathbf{q}_{k-1},$$
$$\mathbf{Y}_k \sim p(\mathbf{Y}_k \,|\, f(\mathbf{t}_k)) \qquad\qquad \mathbf{Y}_k \sim p(\mathbf{Y}_k \,|\, \mathbf{H} \;\; \bar{\mathbf{f}}_k)$$

where k represents time and $\bar{\mathbf{f}}_k \in R^d$ is a vector of derivatives of **f**, and $q_k \sim \mathrm{N}(0, Q_k)$

This can be efficiently solved in $\mathcal{O}(N_t d^3)$ through Kalman filtering and smoothing:



State Space GPs - Matérn-3/2

- Matérn-3/2 covariance is:

$$K_t(t, t') = \sigma^2 \left(1 + \frac{\sqrt{3} \mid t - t' \mid}{\ell}\right) \exp\left(-\frac{\sqrt{3} \mid t - t' \mid}{\ell}\right) \tag{8}$$

- Which has the following SDE representation

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{pmatrix}, \mathbf{L} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \mathbf{P}_\infty = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \lambda^2\sigma^2 \end{pmatrix} \tag{9}$$

where $\lambda = \sqrt{3}/\ell$.

State Space GPs - Matérn-3/2

- Matérn-3/2 covariance is:

$$K_t(t, t') = \sigma^2 \left(1 + \frac{\sqrt{3} \mid t - t' \mid}{\ell}\right) \exp\left(-\frac{\sqrt{3} \mid t - t' \mid}{\ell}\right) \tag{8}$$

- Which has the following SDE representation

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ -\lambda^2 & -2\lambda \end{pmatrix}, \mathbf{L} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \mathbf{P}_\infty = \begin{pmatrix} \sigma^2 & 0 \\ 0 & \lambda^2\sigma^2 \end{pmatrix} \tag{9}$$

where $\lambda = \sqrt{3}/\ell$.

Spatio-temporal State Space GPs

- Let the data lie on a spatio-temporal grid and the GP kernel be separable with a Markov kernel on time. Let N_t be the number of temporal locations and N_s the number of spatial then:
  - $\mathbf{X} = \left[(t, \mathbf{x}_s) \right]_{t=1}^{N_t}$
  - $K(\mathbf{x}, \mathbf{x}) = K_s(\mathbf{x}_s, \mathbf{x}_s) \cdot K_t(t, t)$
- Then the GP has the following SDE representation:

$$\bar{\mathbf{f}}_k = [\bar{\mathbf{f}}_{k,s}]_s^{N_s} \tag{10}$$

and

$$\mathbf{A} = \mathbf{I} \otimes \mathbf{A}_t, \quad \mathbf{L} = \mathbf{I} \otimes \mathbf{L}_t, \quad \mathbf{H} = \mathbf{I} \otimes \mathbf{H}_t, \quad \mathbf{Q} = \mathbf{K}_s \otimes \mathbf{Q}_t \tag{11}$$

- We again just run a Kalman filter and smoother but now in $\mathcal{O}(N_t(N_s \cdot d)^3)$!
- Equivalent to a batch GP with a Kronecer structured kernel $\mathbf{K}_s \otimes \mathbf{K}_t$

Spatio-temporal State Space GPs

- Let the data lie on a spatio-temporal grid and the GP kernel be separable with a Markov kernel on time. Let N_t be the number of temporal locations and N_s the number of spatial then:
  - $\mathbf{X} = \left[(t, \mathbf{x}_s)\right]_{t=1}^{N_t}$
  - $K(\mathbf{x}, \mathbf{x}) = K_s(\mathbf{x}_s, \mathbf{x}_s) \cdot K_t(t, t)$
- Then the GP has the following SDE representation:

$$\bar{\mathbf{f}}_k = [\bar{\mathbf{f}}_{k,s}]_s^{N_s} \tag{10}$$

and

$$\mathbf{A} = \mathbf{I} \otimes \mathbf{A}_t, \quad \mathbf{L} = \mathbf{I} \otimes \mathbf{L}_t, \quad \mathbf{H} = \mathbf{I} \otimes \mathbf{H}_t, \quad \mathbf{Q} = \mathbf{K}_s \otimes \mathbf{Q}_t \tag{11}$$

- We again just run a Kalman filter and smoother but now in $\mathcal{O}(N_t(N_s \cdot d)^3)$!
- Equivalent to a batch GP with a Kronecer structured kernel $\mathbf{K}_s \otimes \mathbf{K}_t$

Spatio-temporal State Space GPs

- Let the data lie on a spatio-temporal grid and the GP kernel be separable with a Markov kernel on time. Let N_t be the number of temporal locations and N_s the number of spatial then:
  - $\mathbf{X} = \left[(t, \mathbf{x}_s)\right]_{t=1}^{N_t}$
  - $K(\mathbf{x}, \mathbf{x}) = K_s(\mathbf{x}_s, \mathbf{x}_s) \cdot K_t(t, t)$
- Then the GP has the following SDE representation:

$$\bar{\mathbf{f}}_k = [\bar{\mathbf{f}}_{k,s}]_s^{N_s} \tag{10}$$

and

$$\mathbf{A} = \mathbf{I} \otimes \mathbf{A}_t, \quad \mathbf{L} = \mathbf{I} \otimes \mathbf{L}_t, \quad \mathbf{H} = \mathbf{I} \otimes \mathbf{H}_t, \quad \mathbf{Q} = \mathbf{K}_s \otimes \mathbf{Q}_t \tag{11}$$

- We again just run a Kalman filter and smoother but now in $\mathcal{O}(N_t(N_s \cdot d)^3)$!
- Equivalent to a batch GP with a Kronecer structured kernel $\mathbf{K}_s \otimes \mathbf{K}_t$

Spatio-temporal State Space GPs

- Let the data lie on a spatio-temporal grid and the GP kernel be separable with a Markov kernel on time. Let N_t be the number of temporal locations and N_s the number of spatial then:
  - $\mathbf{X} = \left[(t, \mathbf{x}_s)\right]_{t=1}^{N_t}$
  - $K(\mathbf{x}, \mathbf{x}) = K_s(\mathbf{x}_s, \mathbf{x}_s) \cdot K_t(t, t)$
- Then the GP has the following SDE representation:

$$\bar{\mathbf{f}}_k = [\bar{\mathbf{f}}_{k,s}]_s^{N_s} \tag{10}$$

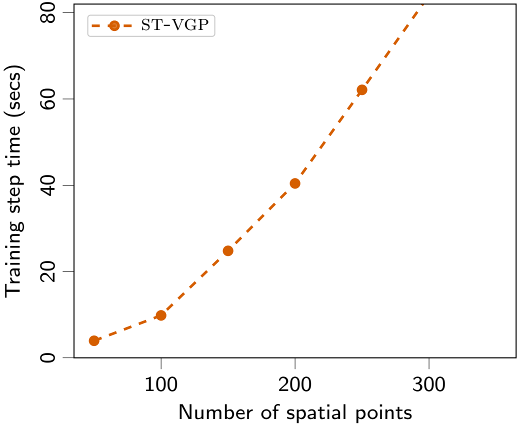
and

$$\mathbf{A} = \mathbf{I} \otimes \mathbf{A}_t, \quad \mathbf{L} = \mathbf{I} \otimes \mathbf{L}_t, \quad \mathbf{H} = \mathbf{I} \otimes \mathbf{H}_t, \quad \mathbf{Q} = \mathbf{K}_s \otimes \mathbf{Q}_t \tag{11}$$

- We again just run a Kalman filter and smoother but now in $\mathcal{O}(N_t(N_s \cdot d)^3)$!
- Equivalent to a batch GP with a Kronecer structured kernel $\mathbf{K}_s \otimes \mathbf{K}_t$

The Problem with State-space GPs for Spatial Data

- The run time is $\mathcal{O}(N_t(N_s \cdot d)^3)$
- Cubic is the size of the state!
- Limits the number of spatial points and Markov kernels that can be used



Natural Gradients as Conjugate Operations

Optimising a Variational Approximate Posterior

- Recall that the variational lower bound is

$$\text{ELBO} = \mathcal{L} = \sum_{n}^{N} \mathbb{E}_{q(\mathbf{f}_n)} \left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n) \right] - \text{KL} \left[q(\mathbf{u}) || p(\mathbf{u}) \right] \quad (12)$$

- And we want to solve

$$\arg\max_{q(\mathbf{u})} \mathcal{L} \quad (13)$$

- We can update the parameters of $q(\mathbf{u})$ using gradient descent:

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \beta \frac{\partial \text{ELBO}}{\partial \boldsymbol{\lambda}} \quad (14)$$

- But this depends on the parameterisation used for $q(\mathbf{u})$

Optimising a Variational Approximate Posterior

- Recall that the variational lower bound is

$$\text{ELBO} = \mathcal{L} = \sum_n^N \mathbb{E}_{q(\mathbf{f}_n)}\left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right] - \text{KL}\left[q(\mathbf{u})||p(\mathbf{u})\right] \tag{12}$$

- And we want to solve

$$\underset{q(\mathbf{u})}{\arg\max}\, \mathcal{L} \tag{13}$$

- We can update the parameters of $q(\mathbf{u})$ using gradient descent:

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \beta \frac{\partial \text{ELBO}}{\partial \boldsymbol{\lambda}} \tag{14}$$

- But this depends on the parameterisation used for $q(\mathbf{u})$

Optimising a Variational Approximate Posterior

– Recall that the variational lower bound is

$$\mathrm{ELBO} = \mathcal{L} = \sum_{n}^{N} \mathbb{E}_{q(\mathbf{f}_n)} \left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right] - \mathrm{KL}\left[q(\mathbf{u}) || p(\mathbf{u})\right] \tag{12}$$

– And we want to solve

$$\arg\max_{q(\mathbf{u})} \mathcal{L} \tag{13}$$

– We can update the parameters of $q(\mathbf{u})$ using gradient descent:

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} + \beta \frac{\partial \mathrm{ELBO}}{\partial \boldsymbol{\lambda}} \tag{14}$$

– But this depends on the parameterisation used for $q(\mathbf{u})$

Different Parameterisations of $q(\mathbf{u})$

A Gaussian distribution $q(\mathbf{u}) = \mathrm{N}(\mathbf{u} \,|\, \mathbf{m}, \mathbf{S})$ can be parameterised in different ways:

$$\boldsymbol{\theta} = (\mathbf{m}, \mathbf{S}), \tag{15}$$

$$\boldsymbol{\lambda} = (\mathbf{S}^{-1}\mathbf{m}, -\frac{1}{2}\mathbf{S}^{-1}), \tag{16}$$

$$\boldsymbol{\mu} = (\mathbf{m}, \mathbf{m}\mathbf{m}^\top + \mathbf{S}), \tag{17}$$

where θ are the moment parameters, $\boldsymbol{\lambda}$ are the natural parameters, and $\boldsymbol{\mu}$ are the expectation parameters. To make it clear when are talking about the prior vs approximate posterior we use $\boldsymbol{\eta}$ to denote the natural parameters of the model prior $p(\mathbf{u})$.

Natural Gradients

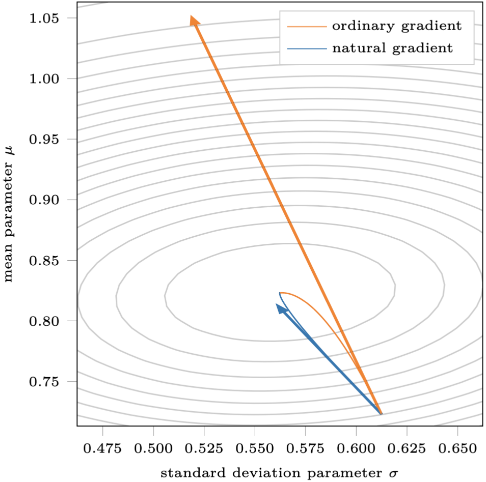


Figure: From Salimbeni et al. [2018]

Natural Gradients - (1)

The natural gradient ($\tilde{g}(\boldsymbol{\lambda})$) is a direction of steepest descent:

$$\tilde{g}(\boldsymbol{\lambda}) = \lim_{\epsilon \to 0} \arg\max \mathcal{L}(\boldsymbol{\lambda} + d\boldsymbol{\lambda}) \quad \text{s.t.} \quad D_{\mathrm{KLD}}\left(q(\mathbf{u} \,|\, \boldsymbol{\lambda}), q(\mathbf{u} \,|\, \boldsymbol{\lambda} + d\boldsymbol{\lambda})\right) < \epsilon \tag{18}$$

where the distance function is the (symmetric) KLD divergence

$$D_{\mathrm{KLD}}\left(q(\mathbf{u} \,|\, \boldsymbol{\lambda}), q(\mathbf{u} \,|\, \boldsymbol{\lambda}')\right) = \mathbb{E}_{q(\mathbf{u} \,|\, \boldsymbol{\lambda})}\left[\log \frac{q(\mathbf{u} \,|\, \boldsymbol{\lambda})}{q(\mathbf{u} \,|\, \boldsymbol{\lambda}')}\right] + \mathbb{E}_{q(\mathbf{u} \,|\, \boldsymbol{\lambda}')}\left[\log \frac{q(\mathbf{u} \,|\, \boldsymbol{\lambda}')}{q(\mathbf{u} \,|\, \boldsymbol{\lambda})}\right]. \tag{19}$$

(See Amari [1998], Hoffman et al. [2013])

Natural Gradients - (2)

The Natural Gradient simplifies to the preconditioned standard gradient:

$$\tilde{g}(\boldsymbol{\lambda}) = \left[I(\boldsymbol{\lambda}^T)^{-1} \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}^T} \right]^T = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} I(\boldsymbol{\lambda})^{-1} \tag{20}$$

Using the properties of the multivariate Gaussian this further simplifies. The Fisher information matrix is

$$\mathbf{I}(\boldsymbol{\lambda}) = \mathbb{E}\left[(\frac{\partial \log p(x \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}})(\frac{\partial \log p(x \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}})^T \right] = \frac{\mathrm{d}^2 A(\boldsymbol{\lambda})}{\mathrm{d}\boldsymbol{\lambda}^2} = \frac{\partial \mathbb{E}\left[\, T(x) \,\right]}{\partial \boldsymbol{\lambda}} = \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}} \tag{21}$$

And applying chain rule on Eq. (20):

$$\tilde{g}(\theta) = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} (\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}})^{-1} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}} (\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}})^{-1} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \tag{22}$$

(See Hensman et al. [2012], Khan and Rue [2021])

Natural Gradients - (2)

The Natural Gradient simplifies to the preconditioned standard gradient:

$$\tilde{g}(\boldsymbol{\lambda}) = \left[I(\boldsymbol{\lambda}^T)^{-1} \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}^T} \right]^T = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} I(\boldsymbol{\lambda})^{-1} \tag{20}$$

Using the properties of the multivariate Gaussian this further simplifies. The Fisher information matrix is

$$\mathbf{I}(\boldsymbol{\lambda}) = \mathbb{E}\left[(\frac{\partial \log p(x \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}})(\frac{\partial \log p(x \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}})^T \right] = \frac{\mathrm{d}^2 A(\boldsymbol{\lambda})}{\mathrm{d}\boldsymbol{\lambda}^2} = \frac{\partial \mathbb{E}\left[\, T(x) \,\right]}{\partial \boldsymbol{\lambda}} = \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}} \tag{21}$$

And applying chain rule on Eq. (20):

$$\tilde{g}(\theta) = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} (\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}})^{-1} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}} (\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}})^{-1} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \tag{22}$$

(See Hensman et al. [2012], Khan and Rue [2021])

Natural Gradients - (2)

The Natural Gradient simplifies to the preconditioned standard gradient:

$$\tilde{g}(\boldsymbol{\lambda}) = \left[I(\boldsymbol{\lambda}^T)^{-1} \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}^T} \right]^T = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} I(\boldsymbol{\lambda})^{-1} \tag{20}$$

Using the properties of the multivariate Gaussian this further simplifies. The Fisher information matrix is

$$\mathbf{I}(\boldsymbol{\lambda}) = \mathbb{E}\left[\left(\frac{\partial \log p(x \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}}\right)\left(\frac{\partial \log p(x \mid \boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}}\right)^T \right] = \frac{\mathrm{d}^2 A(\boldsymbol{\lambda})}{\mathrm{d}\boldsymbol{\lambda}^2} = \frac{\partial \mathbb{E}\left[\, T(x) \,\right]}{\partial \boldsymbol{\lambda}} = \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}} \tag{21}$$

And applying chain rule on Eq. (20):

$$\tilde{g}(\theta) = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} \left(\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}}\right)^{-1} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}} \left(\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\lambda}}\right)^{-1} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \tag{22}$$

(See Hensman et al. [2012], Khan and Rue [2021])

Natural Gradients - (3)

- The natural gradient update is given by:

$$\lambda \leftarrow \lambda + \beta\, \tilde{g}(\boldsymbol{\lambda}) = \lambda + \beta \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \tag{23}$$

- Compared to the 'standard' gradient

$$\lambda = \lambda + \beta \frac{\partial \mathcal{L}}{\partial \lambda} \tag{24}$$

- Take a gradient *w.r.t.* to $\boldsymbol{\mu}$ not $\boldsymbol{\lambda}$

Natural Gradients - (3)

– The natural gradient update is given by:

$$\lambda \leftarrow \lambda + \beta\, \tilde{g}(\boldsymbol{\lambda}) = \lambda + \beta\, \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \tag{23}$$

– Compared to the ‘standard’ gradient

$$\lambda = \lambda + \beta\, \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} \tag{24}$$

– Take a gradient *w.r.t.* to $\boldsymbol{\mu}$ not $\boldsymbol{\lambda}$

Natural Gradients - (3)

- The natural gradient update is given by:

$$\lambda \leftarrow \lambda + \beta\, \tilde{g}(\boldsymbol{\lambda}) = \lambda + \beta\, \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \tag{23}$$

- Compared to the 'standard' gradient

$$\lambda = \lambda + \beta\, \frac{\partial \mathcal{L}}{\partial \boldsymbol{\lambda}} \tag{24}$$

- Take a gradient *w.r.t.* to $\boldsymbol{\mu}$ not $\boldsymbol{\lambda}$

Conjugate Natural Gradients

The natural gradient update is:

$$\begin{aligned} \lambda &= \lambda + \beta \frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \\ &= \lambda + \beta \frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} - \beta \frac{\partial \mathrm{KLD}}{\partial \boldsymbol{\mu}} \end{aligned} \tag{25}$$

Which simplifies to:

$$\begin{aligned} \boldsymbol{\lambda} &= \underbrace{(1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta \frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}}}_{\text{Likelihood}} + \underbrace{\boldsymbol{\eta}}_{\text{Prior}} \\ &= \widetilde{\boldsymbol{\lambda}} + \boldsymbol{\eta} \end{aligned} \tag{26}$$

which is a Bayesian update from the model prior ($\boldsymbol{\eta}$) with an (approximate likelihood) parameterised by $\widetilde{\boldsymbol{\lambda}}$.
(See Khan and Lin [2017], Hamelijnck et al. [2021])

Natural Gradients - Key Points

Natural Gradients - Key Points

- A natural gradient can be computed by a (conjugate) Bayesian update!

$$\begin{aligned} \widetilde{\boldsymbol{\lambda}} &= (1-\beta)\widetilde{\boldsymbol{\lambda}}_0 + \beta \frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} \\ \lambda &\leftarrow \widetilde{\boldsymbol{\lambda}} + \boldsymbol{\eta} \end{aligned} \tag{27}$$

- The approximate likelihood is only updated additively by $\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}}$
- Reparameterise $\widetilde{\boldsymbol{\lambda}} \rightarrow \left[\widetilde{\mathbf{Y}}, \widetilde{\mathbf{V}}\right]$ then for the SVGP the natural gradient update can be written as:

$$q(\mathbf{u}) \propto N(\widetilde{\mathbf{Y}} \,|\, \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u}) \tag{28}$$

Spatio-Temporal Variational GPS

ST-VGP - Game Plan

ST-VGP - Game Plan

- We are going to define the inducing points on a spatio-temporal grid
- This causes the marginal $q(\mathbf{f}_n)$ to only depend on the spatial inducing points
- The natural gradients approximate likelihood ($N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})$) is now block diagonal

$$q(\mathbf{u}) \propto N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u}) \quad = \left[\prod_{t}^{N_t} N(\widetilde{\mathbf{Y}}_t \mid \mathbf{u}_t, \widetilde{\mathbf{V}}_{t,t}) \right] p(\mathbf{u}) \tag{29}$$

- We can then compute $q(\mathbf{u})$, and additionally the full ELBO , using a state-space GP!

ST-VGP - Game Plan

- We are going to define the inducing points on a spatio-temporal grid
- This causes the marginal $q(\mathbf{f}_n)$ to only depend on the spatial inducing points
- The natural gradients approximate likelihood ($N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})$) is now block diagonal

$$q(\mathbf{u}) \propto N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u}) = \left[\prod_t^{N_t} N(\widetilde{\mathbf{Y}}_t \mid \mathbf{u}_t, \widetilde{\mathbf{V}}_{t,t}) \right] p(\mathbf{u}) \tag{29}$$

- We can then compute $q(\mathbf{u})$, and additionally the full ELBO , using a state-space GP!

ST-VGP - Game Plan

- We are going to define the inducing points on a spatio-temporal grid
- This causes the marginal $q(\mathbf{f}_n)$ to only depend on the spatial inducing points
- The natural gradients approximate likelihood ($N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})$) is now block diagonal

$$q(\mathbf{u}) \propto N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u}) \quad = \left[\prod_{t}^{N_t} N(\widetilde{\mathbf{Y}}_t \mid \mathbf{u}_t, \widetilde{\mathbf{V}}_{t,t}) \right] p(\mathbf{u}) \tag{29}$$

- We can then compute $q(\mathbf{u})$, and additionally the full ELBO , using a state-space GP!

ST-VGP - Game Plan

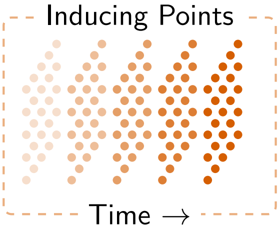
- We are going to define the inducing points on a spatio-temporal grid
- This causes the marginal $q(\mathbf{f}_n)$ to only depend on the spatial inducing points
- The natural gradients approximate likelihood ($N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})$) is now block diagonal

$$q(\mathbf{u}) \propto N(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u}) \quad = \left[\prod_{t}^{N_t} N(\widetilde{\mathbf{Y}}_t \mid \mathbf{u}_t, \widetilde{\mathbf{V}}_{t,t}) \right] p(\mathbf{u}) \tag{29}$$

- We can then compute $q(\mathbf{u})$, and additionally the full ELBO , using a state-space GP!

Spatial Sparsity

- We define a spatial sparsity as inducing points lying on spatio-temporal grid



- Assume $\mathbf{X}$ is also on a grid then:

$$\mathbf{K_{XX}} = \mathbf{K}^{(t)}_{tt} \otimes \mathbf{K}^{(\mathbf{s})}_{\mathbf{SS}}, \quad \mathbf{K_{XZ}} = \mathbf{K}^{(t)}_{tt} \otimes \mathbf{K}^{(\mathbf{s})}_{\mathbf{SZ_s}}, \quad \mathbf{K_{ZZ}} = \mathbf{K}^{(t)}_{tt} \otimes \mathbf{K}^{(\mathbf{s})}_{\mathbf{Z_sZ_s}} \tag{30}$$

- The inducing points only affect the spatial kernels!

Kronecker Structured Marginals

- Recall the SVGP ELBO :

$$\text{ELBO} = \sum_{n}^{N} \mathbb{E}_{q(\mathbf{f}_n)} \left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right] - \text{KL}\left[q(\mathbf{u}) || p(\mathbf{u})\right] \tag{31}$$

- The marginal $q(\mathbf{f}) = \int p(\mathbf{f} \mid \mathbf{u}) q(\mathbf{u}) \, \mathrm{d}\mathbf{u}$ is Kronecker Structured
- Starting with the mean:

$$\begin{aligned} \mathbf{m}_f &= \mathbf{K}_{\mathbf{X},\mathbf{Z}} \mathbf{K}_{\mathbf{Z},\mathbf{Z}}^{-1} \mathbf{m} \\ &= (\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{(t)} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)})(\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{-(t)} \otimes \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)}) \, \mathbf{m} \\ &= (\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{(t)} \mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{-(t)}) \otimes (\mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)} \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)}) \, \mathbf{m} \\ &= (\mathbf{I} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)} \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)}) \, \mathbf{m} \end{aligned} \tag{32}$$

- The term $(\mathbf{I} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)} \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)})$ is block diagonal hence $\mathbf{m}_{f,n}$ only depends on the inducing points in its spatial slice!

Kronecker Structured Marginals

- Recall the SVGP ELBO :

$$\text{ELBO} = \sum_{n}^{N} \mathbb{E}_{q(\mathbf{f}_n)} \left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n) \right] - \text{KL} \left[q(\mathbf{u}) || p(\mathbf{u}) \right] \tag{31}$$

- The marginal $q(\mathbf{f}) = \int p(\mathbf{f} \mid \mathbf{u}) q(\mathbf{u}) \, \mathrm{d}\mathbf{u}$ is Kronecker Structured
- Starting with the mean:

$$\begin{aligned} \mathbf{m}_f &= \mathbf{K}_{\mathbf{X},\mathbf{Z}} \mathbf{K}_{\mathbf{Z},\mathbf{Z}}^{-1} \mathbf{m} \\ &= (\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{(t)} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)})(\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{-(t)} \otimes \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)}) \, \mathbf{m} \\ &= (\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{(t)} \mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{-(t)}) \otimes (\mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)} \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)}) \, \mathbf{m} \\ &= (\mathbf{I} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)} \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)}) \, \mathbf{m} \end{aligned} \tag{32}$$

- The term $(\mathbf{I} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z_s}}^{(s)} \mathbf{K}_{\mathbf{Z_s},\mathbf{Z_s}}^{-(s)})$ is block diagonal hence $\mathbf{m}_{f,n}$ only depends on the inducing points in its spatial slice!

Kronecker Structured Marginals

- Recall the SVGP ELBO :

$$\text{ELBO} = \sum_{n}^{N} \mathbb{E}_{q(\mathbf{f}_n)} \left[\log p(\mathbf{Y}_n \,|\, \mathbf{f}_n)\right] - \mathrm{KL}\left[q(\mathbf{u}) || p(\mathbf{u})\right] \tag{31}$$

- The marginal $q(\mathbf{f}) = \int p(\mathbf{f} \,|\, \mathbf{u}) q(\mathbf{u})\, \mathrm{d}\mathbf{u}$ is Kronecker Structured
- Starting with the mean:

$$\begin{aligned} \mathbf{m}_f &= \mathbf{K}_{\mathbf{X},\mathbf{Z}} \mathbf{K}_{\mathbf{Z},\mathbf{Z}}^{-1} \mathbf{m} \\ &= (\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{(t)} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z}_\mathbf{s}}^{(s)})(\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{-(t)} \otimes \mathbf{K}_{\mathbf{Z}_\mathbf{s},\mathbf{Z}_\mathbf{s}}^{-(s)})\, \mathbf{m} \\ &= (\mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{(t)} \mathbf{K}_{\boldsymbol{t},\boldsymbol{t}}^{-(t)}) \otimes (\mathbf{K}_{\mathbf{s},\mathbf{Z}_\mathbf{s}}^{(s)} \mathbf{K}_{\mathbf{Z}_\mathbf{s},\mathbf{Z}_\mathbf{s}}^{-(s)})\, \mathbf{m} \\ &= (\mathbf{I} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z}_\mathbf{s}}^{(s)} \mathbf{K}_{\mathbf{Z}_\mathbf{s},\mathbf{Z}_\mathbf{s}}^{-(s)})\, \mathbf{m} \end{aligned} \tag{32}$$

- The term $(\mathbf{I} \otimes \mathbf{K}_{\mathbf{s},\mathbf{Z}_\mathbf{s}}^{(s)} \mathbf{K}_{\mathbf{Z}_\mathbf{s},\mathbf{Z}_\mathbf{s}}^{-(s)})$ is block diagonal hence $\mathbf{m}_{f,n}$ only depends on the inducing points in its spatial slice!

Natural Gradients with Spatial Sparsity - (1)

- Recall that a natural gradient is given by:

$$\begin{aligned} \widetilde{\boldsymbol{\lambda}} &= (1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} \\ \lambda &\leftarrow \widetilde{\boldsymbol{\lambda}} + \boldsymbol{\eta} \end{aligned} \tag{33}$$

- Expanding out the ELL term:

$$\frac{\partial \mathrm{ELL}}{\partial \mu} = \sum_{n}^{N} \frac{\partial \mathbb{E}_{q(\mathbf{f}_n)}\left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right]}{\partial \mu} \tag{34}$$

- $q(\mathbf{f}_n)$ only depends on the inducing points in the same time slice as $\mathbf{X}_n$
- Hence $\frac{\partial \mathrm{ELL}}{\partial \mu}$ is block-diagonal!

Natural Gradients with Spatial Sparsity - (1)

- Recall that a natural gradient is given by:

$$\begin{aligned} \widetilde{\boldsymbol{\lambda}} &= (1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} \\ \lambda &\leftarrow \widetilde{\boldsymbol{\lambda}} + \boldsymbol{\eta} \end{aligned} \tag{33}$$

- Expanding out the ELL term:

$$\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} = \sum_n^N \frac{\partial \mathbb{E}_{q(\mathbf{f}_n)}\left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right]}{\partial \boldsymbol{\mu}} \tag{34}$$

- $q(\mathbf{f}_n)$ only depends on the inducing points in the same time slice as $\mathbf{X}_n$
- Hence $\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}}$ is block-diagonal!

Natural Gradients with Spatial Sparsity - (1)

– Recall that a natural gradient is given by:

$$\widetilde{\boldsymbol{\lambda}} = (1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}}$$
$$\lambda \leftarrow \widetilde{\boldsymbol{\lambda}} + \boldsymbol{\eta} \tag{33}$$

– Expanding out the ELL term:

$$\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} = \sum_n^N \frac{\partial \mathbb{E}_{q(\mathbf{f}_n)}\left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right]}{\partial \boldsymbol{\mu}} \tag{34}$$

– $q(\mathbf{f}_n)$ only depends on the inducing points in the same time slice as $\mathbf{X}_n$

– Hence $\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}}$ is block-diagonal!

Natural Gradients with Spatial Sparsity - (1)

- Recall that a natural gradient is given by:

$$\begin{aligned} \widetilde{\boldsymbol{\lambda}} &= (1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \text{ELL}}{\partial \boldsymbol{\mu}} \\ \lambda &\leftarrow \widetilde{\boldsymbol{\lambda}} + \boldsymbol{\eta} \end{aligned} \tag{33}$$

- Expanding out the ELL term:

$$\frac{\partial \text{ELL}}{\partial \boldsymbol{\mu}} = \sum_{n}^{N} \frac{\partial \mathbb{E}_{q(\mathbf{f}_n)}\left[\log p(\mathbf{Y}_n \mid \mathbf{f}_n)\right]}{\partial \boldsymbol{\mu}} \tag{34}$$

- $q(\mathbf{f}_n)$ only depends on the inducing points in the same time slice as $\mathbf{X}_n$
- Hence $\frac{\partial \text{ELL}}{\partial \boldsymbol{\mu}}$ is block-diagonal!

Natural Gradients with Spatial Sparsity - (2)

- The approximate likelihood natural parameters are updated additively:

$$\begin{aligned}\widetilde{\boldsymbol{\lambda}} &= (1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} \\ \lambda &\leftarrow \widetilde{\boldsymbol{\lambda}} + \boldsymbol{\eta}\end{aligned} \tag{35}$$

- Hence $\widetilde{\boldsymbol{\lambda}}$ is also block diagonal!
- Natural gradient is equivalent to a Bayesian update with block-diagonal noise:

$$q(\mathbf{u}) \propto \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})p(\mathbf{u}) = \prod_{t}^{N_t} \mathrm{N}(\widetilde{\mathbf{Y}}_t \mid \mathbf{u}_t, \widetilde{\mathbf{V}}_t)p(\mathbf{u}) \tag{36}$$

- Standard GP update!

ST-VGP Variational Lower Bound

- Reparameterize the approximate likelihood: $\widetilde{\boldsymbol{\lambda}} \to \left[\widetilde{\mathbf{Y}}, \widetilde{\mathbf{V}}\right]$:

$$q(\mathbf{u}) = \frac{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u})}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid 0, \widetilde{\mathbf{V}} + \mathbf{K})} \tag{37}$$

- Following Chang et al. [2020], substitute this into the ELBO :

$$\begin{aligned}\mathcal{L}_{\text{ST-VGP}} &= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})\, \cancel{p(\mathbf{f} \mid \mathbf{u})}\, \cancel{p(\mathbf{u})} \int \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\, p(\mathbf{u})\, \mathrm{d}\mathbf{u}}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\, \cancel{p(\mathbf{f} \mid \mathbf{u})}\, \cancel{p(\mathbf{u})}}\right] \\ &= \mathbb{E}_{q(\mathbf{f})}\big[\log p(\mathbf{Y} \mid \mathbf{f})\big] - \mathbb{E}_{q(\mathbf{u})}\big[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\big] + \mathbb{E}_{q(\mathbf{u})}\big[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\big]\end{aligned}$$

ST-VGP Variational Lower Bound

- Reparameterize the approximate likelihood: $\tilde{\boldsymbol{\lambda}} \rightarrow \left[\widetilde{\mathbf{Y}}, \widetilde{\mathbf{V}} \right]$:

$$q(\mathbf{u}) = \frac{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u})}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid 0, \widetilde{\mathbf{V}} + \mathbf{K})} \tag{37}$$

- Following Chang et al. [2020], substitute this into the ELBO :

$$\begin{aligned}
\mathcal{L}_{\text{ST-VGP}} &= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f}) \cancel{p(\mathbf{f} \mid \mathbf{u})} \cancel{p(\mathbf{u})} \int \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\, p(\mathbf{u})\, \mathrm{d}\mathbf{u}}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\, \cancel{p(\mathbf{f} \mid \mathbf{u})}\, \cancel{p(\mathbf{u})}}\right] \\
&= \mathbb{E}_{q(\mathbf{f})}\left[\log p(\mathbf{Y} \mid \mathbf{f})\right] - \mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\right] + \mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\right]
\end{aligned}$$

ST-VGP Variational Lower Bound

- Reparameterize the approximate likelihood: $\widetilde{\boldsymbol{\lambda}} \rightarrow \left[\widetilde{\mathbf{Y}}, \widetilde{\mathbf{V}}\right]$:

$$q(\mathbf{u}) = \frac{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u})}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid 0, \widetilde{\mathbf{V}} + \mathbf{K})} \quad (37)$$

- Following Chang et al. [2020], substitute this into the ELBO :

$$\begin{aligned}\mathcal{L}_{\text{ST-VGP}} &= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f})\, \cancel{p(\mathbf{f} \mid \mathbf{u})}\, \cancel{p(\mathbf{u})} \int \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\, p(\mathbf{u})\, \mathrm{d}\mathbf{u}}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\, \cancel{p(\mathbf{f} \mid \mathbf{u})}\, \cancel{p(\mathbf{u})}}\right] \\ &= \mathbb{E}_{q(\mathbf{f})}\big[\log p(\mathbf{Y} \mid \mathbf{f})\big] - \mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\right] + \mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\right]\end{aligned}$$

Variational Lower Bound

- Reparameterize the approximate likelihood: $\widetilde{\boldsymbol{\lambda}} \to \left[\widetilde{\mathbf{Y}}, \widetilde{\mathbf{V}}\right]$:

$$q(\mathbf{u}) = \frac{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) p(\mathbf{u})}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid 0, \widetilde{\mathbf{V}} + \mathbf{K})} \tag{37}$$

- Following Chang et al. [2020], substitute this into the ELBO :

$$\begin{aligned}\mathcal{L}_{\text{ST-VGP}} &= \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{Y} \mid \mathbf{f}) \cancel{p(\mathbf{f} \mid \mathbf{u})} \cancel{p(\mathbf{u})} \int \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) \, p(\mathbf{u}) \, \mathrm{d}\mathbf{u}}{\mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}}) \cancel{p(\mathbf{f} \mid \mathbf{u})} \cancel{p(\mathbf{u})}}\right] \\ &= \mathbb{E}_{q(\mathbf{f})}\left[\log p(\mathbf{Y} \mid \mathbf{f})\right] - \mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\right] + \mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\right]\end{aligned}$$

Efficient Computation through State-space GPS

- Rewrite $q(\mathbf{u})$ as the solution to an LTI-SDE (Särkkä and Solin [2019])
- Compute $q(\mathbf{u})$ through Kalman filtering and Smoothing – $\mathcal{O}(N_t (M_s \cdot d)^3)$
- Compute full ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\mathbf{f})}\big[\log p(\mathbf{Y} \mid \mathbf{f})\big]}_{\mathcal{O}(N(M_s \cdot d)^3)} + \underbrace{\mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\right]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} - \underbrace{\mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\right]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} \quad (38)$$

- Can be computed in order $\mathcal{O}(N(M_s \cdot d)^3)$
- Equivalent to an SVGP with inducing points at every time point but only requires computation that is linear *w.r.t.* time!

Efficient Computation through State-space GPS

- Rewrite $q(\mathbf{u})$ as the solution to an LTI-SDE (Särkkä and Solin [2019])
- Compute $q(\mathbf{u})$ through Kalman filtering and Smoothing – $\mathcal{O}(N_t(M_s \cdot d)^3)$
- Compute full ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\mathbf{f})}\big[\log p(\mathbf{Y} \mid \mathbf{f})\big]}_{\mathcal{O}(N(M_s \cdot d)^3)} + \underbrace{\mathbb{E}_{q(\mathbf{u})}\big[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\big]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} - \underbrace{\mathbb{E}_{q(\mathbf{u})}\big[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\big]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} \quad (38)$$

- Can be computed in order $\mathcal{O}(N(M_s \cdot d)^3)$
- Equivalent to an SVGP with inducing points at every time point but only requires computation that is linear *w.r.t.* time!

Efficient Computation through State-space GPS

- Rewrite $q(\mathbf{u})$ as the solution to an LTI-SDE (Särkkä and Solin [2019])
- Compute $q(\mathbf{u})$ through Kalman filtering and Smoothing – $\mathcal{O}(N_t(M_s \cdot d)^3)$
- Compute full ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\mathbf{f})}\big[\log p(\mathbf{Y} \mid \mathbf{f})\big]}_{\mathcal{O}(N(M_s \cdot d)^3)} + \underbrace{\mathbb{E}_{q(\mathbf{u})}\big[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\big]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} - \underbrace{\mathbb{E}_{q(\mathbf{u})}\big[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\big]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} \tag{38}$$

- Can be computed in order $\mathcal{O}(N(M_s \cdot d)^3)$
- Equivalent to an SVGP with inducing points at every time point but only requires computation that is linear *w.r.t.* time!

Efficient Computation through State-space GPs

- Rewrite $q(\mathbf{u})$ as the solution to an LTI-SDE (Särkkä and Solin [2019])
- Compute $q(\mathbf{u})$ through Kalman filtering and Smoothing – $\mathcal{O}(N_t(M_s \cdot d)^3)$
- Compute full ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\mathbf{f})}\left[\log p(\mathbf{Y} \mid \mathbf{f})\right]}_{\mathcal{O}(N(M_s \cdot d)^3)} + \underbrace{\mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\right]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} - \underbrace{\mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\right]}_{\mathcal{O}(N_t(M_s \cdot d)^3)} \quad (38)$$

- Can be computed in order $\mathcal{O}(N(M_s \cdot d)^3)$
- Equivalent to an SVGP with inducing points at every time point but only requires computation that is linear *w.r.t.* time!

Efficient Computation through State-space GPs

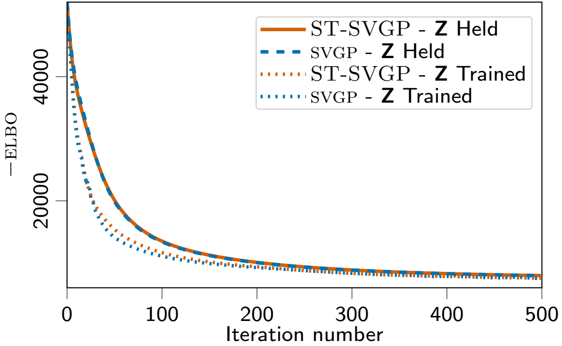
- Rewrite $q(\mathbf{u})$ as the solution to an LTI-SDE (Särkkä and Solin [2019])
- Compute $q(\mathbf{u})$ through Kalman filtering and Smoothing – $\mathcal{O}(N_{\boldsymbol{t}}(M_{\mathsf{s}} \cdot d)^3)$
- Compute full ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(\mathbf{f})}\big[\log p(\mathbf{Y} \mid \mathbf{f})\big]}_{\mathcal{O}(N(M_{\mathsf{s}} \cdot d)^3)} + \underbrace{\mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}} + \mathbf{K})\right]}_{\mathcal{O}(N_{\boldsymbol{t}}(M_{\mathsf{s}} \cdot d)^3)} - \underbrace{\mathbb{E}_{q(\mathbf{u})}\left[\log \mathrm{N}(\widetilde{\mathbf{Y}} \mid \mathbf{u}, \widetilde{\mathbf{V}})\right]}_{\mathcal{O}(N_{\boldsymbol{t}}(M_{\mathsf{s}} \cdot d)^3)} \quad (38)$$

- Can be computed in order $\mathcal{O}(N(M_{\mathsf{s}} \cdot d)^3)$
- Equivalent to an SVGP with inducing points at every time point but only requires computation that is linear *w.r.t.* time!

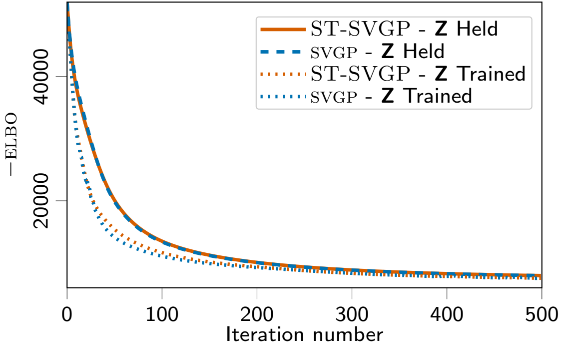
Experiments

Experiment: Equivalence to SVGP



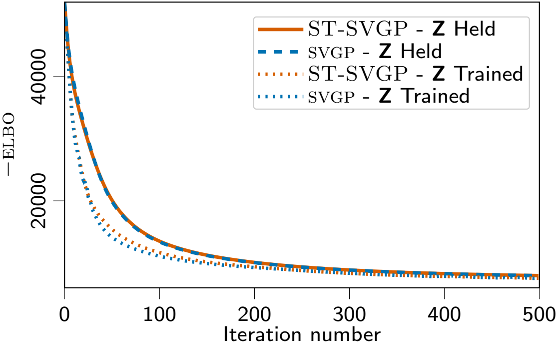
Train Z	Model	RMSE	NLPD
$\times$	ST-SVGP	3.02 ± 0.13	1.72 ± 0.04
$\times$	SVGP	3.02 ± 0.13	1.72 ± 0.04
✓	ST-SVGP	2.79 ± 0.15	1.64 ± 0.04
✓	SVGP	2.94 ± 0.12	1.65 ± 0.05

Experiment: Equivalence to SVGP



Train Z	Model	RMSE	NLPD
×	ST-SVGP	3.02 ± 0.13	1.72 ± 0.04
×	SVGP	3.02 ± 0.13	1.72 ± 0.04
✓	ST-SVGP	2.79 ± 0.15	1.64 ± 0.04
✓	SVGP	2.94 ± 0.12	1.65 ± 0.05

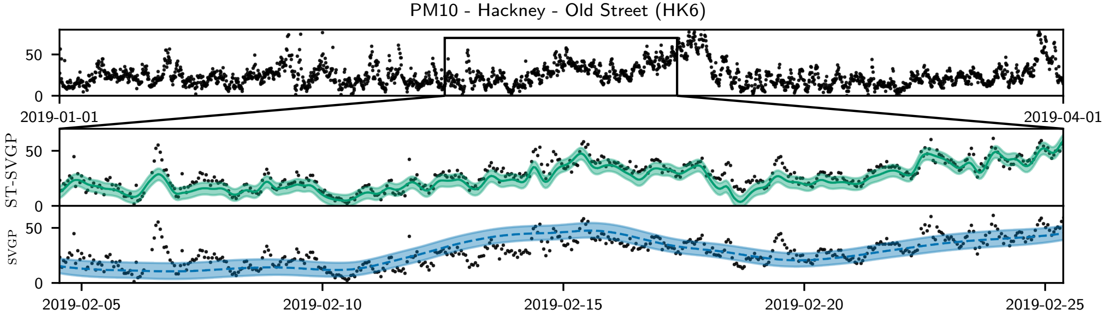
Experiment: Equivalence to SVGP



Train Z	Model	RMSE	NLPD
×	ST-SVGP	3.02 ± 0.13	1.72 ± 0.04
×	SVGP	3.02 ± 0.13	1.72 ± 0.04
✓	ST-SVGP	2.79 ± 0.15	1.64 ± 0.04
✓	SVGP	2.94 ± 0.12	1.65 ± 0.05

Experiment: PM$_{10}$ in London - 1

- Spatio-temporal PM10 data across London



- ST-SVGP with $\approx$ 60K inducing points has the same computational cost of SVGP with $2,000$

Experiment: PM_{10} in London - 2

Model (batch size)	Time (CPU)	Time (GPU)	RMSE	NLPD
ST-SVGP	16.79 ± 0.63	4.47 ± 0.01	**9.96** ± **0.56**	**8.29** ± **0.80**
MF-ST-SVGP	13.74 ± 0.49	0.85 ± 0.01	10.42 ± 0.63	8.52 ± 0.91
SVGP -2000 (600)	20.21 ± 0.28	0.17 ± 0.00	15.46 ± 0.44	12.93 ± 0.95
SVGP -2500 (800)	40.90 ± 1.11	0.25 ± 0.00	15.53 ± 1.09	13.48 ± 1.85
SVGP -5000 (2000)	—	1.19 ± 0.00	14.20 ± 0.44	12.73 ± 0.73
SVGP -8000 (3000)	—	4.09 ± 0.01	13.83 ± 0.47	12.40 ± 0.75
SKI	23.36 ± 1.01	3.61 ± 0.01	12.01 ± 0.55	10.32 ± 0.79

Experiment: PM_{10} in London - 2

Model (batch size)	Time (CPU)	Time (GPU)	RMSE	NLPD
ST-SVGP	16.79 ± 0.63	4.47 ± 0.01	**9.96 ± 0.56**	**8.29 ± 0.80**
MF-ST-SVGP	13.74 ± 0.49	0.85 ± 0.01	10.42 ± 0.63	8.52 ± 0.91
SVGP -2000 (600)	20.21 ± 0.28	0.17 ± 0.00	15.46 ± 0.44	12.93 ± 0.95
SVGP -2500 (800)	40.90 ± 1.11	0.25 ± 0.00	15.53 ± 1.09	13.48 ± 1.85
SVGP -5000 (2000)	—	1.19 ± 0.00	14.20 ± 0.44	12.73 ± 0.73
SVGP -8000 (3000)	—	4.09 ± 0.01	13.83 ± 0.47	12.40 ± 0.75
SKI	23.36 ± 1.01	3.61 ± 0.01	12.01 ± 0.55	10.32 ± 0.79

Experiment: PM_{10} in London - 2

Model (batch size)	Time (CPU)	Time (GPU)	RMSE	NLPD
ST-SVGP	16.79 ± 0.63	4.47 ± 0.01	**9.96 ± 0.56**	**8.29 ± 0.80**
MF-ST-SVGP	13.74 ± 0.49	0.85 ± 0.01	10.42 ± 0.63	8.52 ± 0.91
SVGP -2000 (600)	20.21 ± 0.28	0.17 ± 0.00	15.46 ± 0.44	12.93 ± 0.95
SVGP -2500 (800)	40.90 ± 1.11	0.25 ± 0.00	15.53 ± 1.09	13.48 ± 1.85
SVGP -5000 (2000)	—	1.19 ± 0.00	14.20 ± 0.44	12.73 ± 0.73
SVGP -8000 (3000)	—	4.09 ± 0.01	13.83 ± 0.47	12.40 ± 0.75
SKI	23.36 ± 1.01	3.61 ± 0.01	12.01 ± 0.55	10.32 ± 0.79

Extensions

Spatial Mini Batching

Spatial Minibatching

- We have computed everything as if $\mathbf{X}$ also lies on a spatiotemporal grid
- However we only really require that $\mathbf{X}$ has data at the same temporal points since we compute the required marginals at each time point independently!
- Hence we can easily minibatch in space

$$\begin{aligned}\mathbb{E}_{q(\mathbf{f})}\left[\log p(\mathbf{Y} \mid \mathbf{f})\right] &= \sum_t^{N_t} \sum_s^{N_s} \left[\log p(\mathbf{Y}_{t,s} \mid \mathbf{f}_{t,s})\right] \\ &\approx \sum_t^{N_t} \frac{N_s}{B} \sum_b^{B} \left[\log p(\mathbf{Y}_{t,b} \mid \mathbf{f}_{t,b})\right]\end{aligned} \tag{39}$$

- Computational complexity $\mathcal{O}(N\,(M_s \cdot d)^3) \rightarrow \mathcal{O}(N_t\,(M_s \cdot d)^3)$

Ensuring PSD Updates

Ensuring PSD Updates

- So far we have assumed that the natural gradient always results in a positive semi-definite update, however, this is not always the case
- Beyond Gaussian likelihoods we need a way to ensure our update is valid (beyond using a small learning rate)
- We can use an approximation to the natural gradient that is very similar to the Gauss-Newton approximation
- See Wilkinson et al. [2021], Khan and Rue [2021].

Derivative Observations

Derivative Observations

- We can write a GP prior over a latent function and its various derivatives as

$$p(\mathbf{f}(\mathbf{x}), \nabla_s \mathbf{f}(\mathbf{x}), \nabla_t \mathbf{f}(\mathbf{x}), \nabla_{st} \mathbf{f}(\mathbf{x})) = \mathrm{N}\left(\mathbf{F} \mid \mathbf{0},\ \nabla \mathbf{K} \nabla^T\right) \tag{40}$$

- When the kernel is separable and data lies on a grid we can write

$$p(\nabla \mathbf{f}(\mathbf{X})) = \mathrm{N}\left(\mathbf{f} \mid \mathbf{0},\ \mathbf{K}_t^{\nabla}(\mathbf{X}, \mathbf{X}) \otimes \mathbf{K}_s^{\nabla}(\mathbf{X}, \mathbf{X})\right). \tag{41}$$

where

$$\mathbf{K}_{\cdot}^{\nabla} = \begin{bmatrix} \mathbf{K} & \mathbf{K} \nabla_{\cdot}^T \\ \nabla_{\cdot} \mathbf{K} & \nabla_{\cdot} \mathbf{K} \nabla_{\cdot}^T \end{bmatrix} \tag{42}$$

- Which can immediately be written as a state-space GP when the time kernel is Markov
- Can easily extend to solving linear and non-linear PDEs through the collocation method

Future Work

Future Work

- **State Dimension** - Ultimately, the bottleneck is still the state-dimension size. We can use less inducing points, but then we over smooth!
  - Alternative reduced-rank methods? Ensemble methods?
- **Model Constructions** - There are lots of exciting model constructions to extend this framework to Deep GPs and Multi-fidelity GPs, physics-informed GPs, etc
- **Natural Gradient Approximations** - In general, the natural gradient approximation discussed is effective, however, there needs to be a proper evaluation of different approximation methods

Conclusion

Conclusion

- We have introduced ST-VGP which is a variational SVGP with a computational complexity that is linear *w.r.t.* to time.
- This is done within a natural gradient framework where we represent the approximate posterior with a state-space GP
- Future work could include exploring approximations to the spatial dimension to further improve the computation complexities

Conclusion

- We have introduced ST-VGP which is a variational SVGP with a computational complexity that is linear *w.r.t.* to time.
- This is done within a natural gradient framework where we represent the approximate posterior with a state-space GP
- Future work could include exploring approximations to the spatial dimension to further improve the computation complexities

Conclusion

- We have introduced ST-VGP which is a variational SVGP with a computational complexity that is linear *w.r.t.* to time.
- This is done within a natural gradient framework where we represent the approximate posterior with a state-space GP
- Future work could include exploring approximations to the spatial dimension to further improve the computation complexities

Conclusion

- Thank you for listening!
- Code and link to paper: `https://github.com/AaltoML/spatio-temporal-GPs`

References I

Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Computation*, 10(2): 251–276, 1998.

José M Bernardo and Adrian F. M Smith. *Bayesian theory*. Wiley, 2004.

Paul E. Chang, William J. Wilkinson, Mohammad Emtiyaz Khan, and Arno Solin. Fast variational learning in state-space Gaussian process models. In *30th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6, Espoo, Finland, 2020. IEEE.

Oliver Hamelijnck, William J. Wilkinson, Niki Andreas Loppi, Arno Solin, and Theodoros Damoulas. Spatio-temporal variational gaussian processes. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

James Hensman, Magnus Rattray, and Neil D. Lawrence. Fast variational inference in the conjugate exponential family. In *Advances in Neural Information Processing Systems 25 (NIPS)*, pages 2888–2896. Curran Associates Inc., 2012.

References II

James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 282–290. AUAI Press, 2013.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *J. Mach. Learn. Res.*, 14(1):1303–1347, may 2013. ISSN 1532-4435.

Mohammad Emtiyaz Khan and Wu Lin. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 54 of *Proceedings of Machine Learning Research*, pages 878–887. PMLR, 2017.

Mohammad Emtiyaz Khan and Håvard Rue. The bayesian learning rule, 2021. URL `https://arxiv.org/abs/2107.04562`.

CE Rasmussen and CKI Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, USA, 2006.

References III

Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. Natural gradients in practice: Non-conjugate variational inference in Gaussian process models. In *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 84 of *Proceedings of Machine Learning Research*, pages 689–697. PMLR, 2018.

Simo Särkkä and Arno Solin. *Applied Stochastic Differential Equations*. Cambridge University Press, 2019.

Michalis Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 5 of *Proceedings of Machine Learning Research*, pages 567–574. PMLR, 2009.

William J. Wilkinson, Simo Särkkä, and Arno Solin. Bayes-newton methods for approximate bayesian inference with psd guarantees, 2021.

Appendix

$\textsc{kld}$ Gradients

- We wish to compute:

$$\frac{\partial \mathrm{KLD}}{\partial \boldsymbol{\mu}} = \begin{bmatrix} \frac{\partial \mathrm{KLD}}{\partial \mathbf{m}} - 2\,\frac{\partial \mathrm{KLD}}{\partial \mathbf{S}}\mathbf{m} \\ \frac{\partial \mathrm{KLD}}{\partial \mathbf{S}} \end{bmatrix} \tag{43}$$

  where $\mathrm{KLD} = \frac{1}{2}\left[\log|\mathbf{K}| - \log|\mathbf{S}| - M + \mathbf{m}^T\mathbf{K}^{-1}\mathbf{m} + \mathrm{Tr}\left[\mathbf{K}^{-1}\mathbf{S}\right]\right]$

- The required gradients are:

$$\frac{\partial \mathrm{KLD}}{\partial \mathbf{m}} = \mathbf{K}^{-1}\mathbf{m}, \qquad \frac{\partial \mathrm{KLD}}{\partial \mathbf{S}} = -\frac{1}{2}\mathbf{S}^{-1} + \frac{1}{2}\mathbf{K}^{-1} \tag{44}$$

- Leading to:

$$\frac{\partial \mathrm{KLD}}{\partial \boldsymbol{\mu}} = \begin{bmatrix} +\mathbf{S}^{-1}\mathbf{m} \\ -\frac{1}{2}\mathbf{S}^{-1} + \frac{1}{2}\mathbf{K}^{-1} \end{bmatrix} \tag{45}$$

CVI Update Equations

CVI Equations - (1)

- Initialise natural parameters as a Bayesian update:

$$\boldsymbol{\lambda}_1 = \widetilde{\boldsymbol{\lambda}}_0 + \boldsymbol{\eta} \tag{46}$$

where $\boldsymbol{\eta}_1 = \left[\, 0, -\frac{1}{2}\mathbf{K} \,\right]$ and $\widetilde{\boldsymbol{\lambda}}_1$ are the initial parameters. This implies:

$$\boldsymbol{\lambda}_1 = \left[\mathbf{S}^{-1}\mathbf{m}, -\frac{1}{2}\mathbf{S}^{-1}, \right] \tag{47}$$

$\textsc{cvi}$ Equations - (2)

- The natural gradient update is:

$$\boldsymbol{\lambda}_2 = \boldsymbol{\lambda}_1 + \beta\,\frac{\partial \mathcal{L}}{\partial \boldsymbol{\mu}} \tag{48}$$

- Substituting $\mathcal{L} = \textsc{ell} - \textsc{kld}$ and simplifying:

$$\boldsymbol{\lambda}_2 = \boldsymbol{\lambda}_1 + \beta\,\frac{\partial \textsc{ell}}{\partial \boldsymbol{\mu}} + \beta \begin{bmatrix} -\mathbf{S}^{-1}\mathbf{m} \\ +\frac{1}{2}\mathbf{S}^{-1} - \frac{1}{2}\mathbf{K}^{-1} \end{bmatrix} \tag{49}$$

- Rewrite $\boldsymbol{\lambda}_2 = (1-\beta)\,\boldsymbol{\lambda}_2 + \beta\,\boldsymbol{\lambda}$ and substitute in:

$$\boldsymbol{\lambda}_2 = (1-\beta)\,\boldsymbol{\lambda}_1 + \beta \begin{bmatrix} \mathbf{S}^{-1}\mathbf{m} \\ -\frac{1}{2}\mathbf{S}^{-1} \end{bmatrix} + \beta\,\frac{\partial \textsc{ell}}{\partial \boldsymbol{\mu}} + \beta \begin{bmatrix} -\mathbf{S}^{-1}\mathbf{m} \\ +\frac{1}{2}\mathbf{S}^{-1} - \frac{1}{2}\mathbf{K}^{-1} \end{bmatrix} \tag{50}$$

CVI Equations - (3)

- Substitute $\boldsymbol{\lambda}_2 = \widetilde{\boldsymbol{\lambda}}_0 + \boldsymbol{\eta}$ and simplify:

$$
\begin{aligned}
\boldsymbol{\lambda}_2 &= (1-\beta)\,\boldsymbol{\lambda}_1 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} + \beta\begin{bmatrix} \cancel{\mathbf{S}^{-1}}\mathbf{m} \\ \cancel{-\frac{1}{2}\mathbf{S}^{-1}} \end{bmatrix} + \beta\begin{bmatrix} \cancel{-\mathbf{S}^{-1}}\mathbf{m} + 0 \\ \cancel{+\frac{1}{2}\mathbf{S}^{-1}} - \frac{1}{2}\mathbf{K}^{-1} \end{bmatrix} \\
&= (1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}} + (1-\beta)\,\boldsymbol{\eta} + \beta\,\boldsymbol{\eta} \\
&= \underbrace{(1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}}}_{\widetilde{\boldsymbol{\lambda}}_1} + \boldsymbol{\eta} \\
&= \widetilde{\boldsymbol{\lambda}}_1 + \boldsymbol{\eta} \quad \text{with} \quad \widetilde{\boldsymbol{\lambda}}_1 = (1-\beta)\,\widetilde{\boldsymbol{\lambda}}_0 + \beta\,\frac{\partial \mathrm{ELL}}{\partial \boldsymbol{\mu}}.
\end{aligned}
\tag{51}
$$

- Which recovers the CVI update equations.

Exponential Families

Exponential Family - (1)

A distribution is in the Exponential Family if it can be written as:

$$p(x \mid \theta) = h(x) \, \mathbb{E} \left[\eta(\theta) \cdot T(x) - A(\theta) \right] \tag{52}$$

A Gaussian distribution $q(\mathbf{u}) = \mathrm{N}(\mathbf{u} \mid \mathbf{m}, \mathbf{S})$ is part of the exponential family with:

$$\begin{aligned}
h(x) &= (2\pi)^{-D/2} \\
T(x) &= \left(x, xx^T\right) \\
\eta(\theta) &= \left(\mathbf{S}^{-1}\mathbf{m}, -\frac{1}{2}\mathbf{S}^{-1}\right) \\
A(\theta) &= \log \left[\int h(x) \, \mathbb{E} \left[\eta(\theta) \cdot T(x) \right] \mathrm{d}x \right]
\end{aligned} \tag{53}$$

Exponential Family - (2)

A Gaussian distribution $q(\mathbf{u}) = \mathrm{N}(\mathbf{u} \mid \mathbf{m}, \mathbf{S})$ can be parameterised in different ways:

$$\boldsymbol{\theta} = (\mathbf{m}, \mathbf{S}), \tag{54}$$

$$\boldsymbol{\lambda} = (\mathbf{S}^{-1}\mathbf{m}, -\frac{1}{2}\mathbf{S}^{-1}), \tag{55}$$

$$\boldsymbol{\mu} = (\mathbf{m}, \mathbf{m}\mathbf{m}^\top + \mathbf{S}), \tag{56}$$

where θ are the moment parameters, $\boldsymbol{\lambda}$ are the natural parameters, and $\boldsymbol{\mu}$ are the expectation parameters.

Properties of the Multivariate Gaussian

Property

In the exponential family the gradient of the log normaliser (A) is equal to the expectation parameter:

$$\frac{\partial A(\boldsymbol{\lambda})}{\partial \boldsymbol{\lambda}} = \mathbb{E}\left[\, T(x) \,\right] = \boldsymbol{\mu} \tag{57}$$

Property

When parameterised using natural parameters, conjugate inference in a Gaussian model can be written as:

$$\boldsymbol{\lambda}^{(post)} = \boldsymbol{\lambda}^{(lik)} + \boldsymbol{\eta}^{(prior)} \tag{58}$$

(See Bernardo and Smith [2004] etc)