Inference in Stochastic Processes Reading Group

Javier Antoran, Matt Ashman, Stratis Markou

24th February 2021

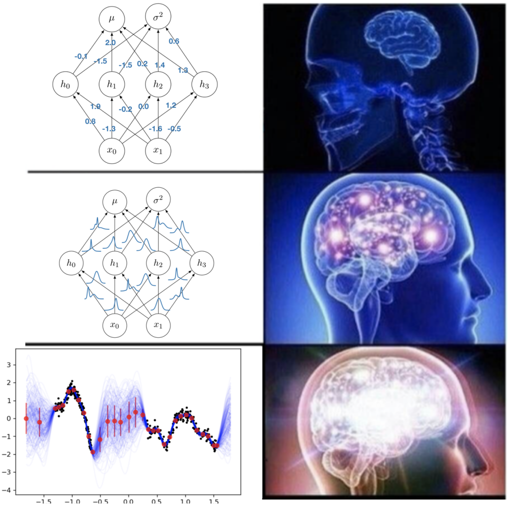
UNIVERSITY OF CAMBRIDGE

In the end we only care about functions

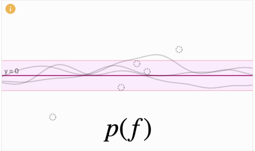
MAP Estimation

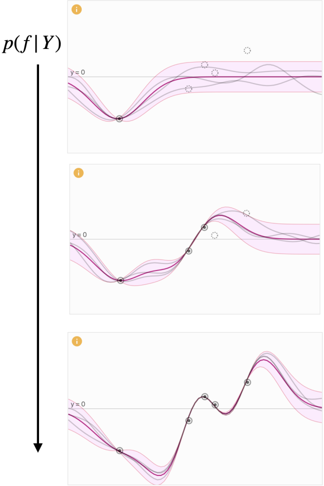
Parametric Inference

Functional Inference



Gaussian processes (GPs) as a motivating example





A Visual Exploration of Gaussian Processes

How to turn a collection of small building blocks into a versatile tool for solving regression problems.

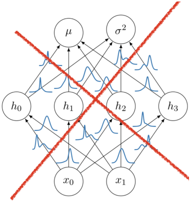
distill.pub/2019/visual-exploration-gaussian-processes/

Does functional inference mean non-parametrics?

NO!

Functional inference refers to performing probabilistic reasoning about functions $\mathbf{f}$ directly, as opposed to model parameters θ.

$$p(\theta|\mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$$

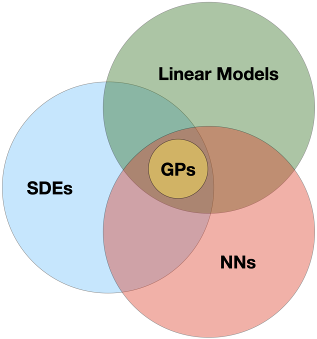


$$p(\mathbf{f}|\mathcal{D}) = \frac{p(\mathcal{D}|\mathbf{f})p(\mathbf{f})}{p(\mathcal{D})}$$

$\mathbf{f}$ could be the output of a parametric model.

Contents

1. Constructing (non-Gaussian) stochastic processes with linear combinations of basis functions
2. Functional inference in neural networks
3. Stochastic differential equations (SDE)



Relevant topics that will not be covered

- Rigorous measure theoretic background
- Approximate inference in Gaussian processes
- Infinite width limits of neural networks - NTK

Basis function view

$$f(\mathbf{x}) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$

Basis function view

$$f(\mathbf{x}) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$

- Basis function $\phi(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^M$.

e.g. $\phi(x) = [1, x, x^2, x^3, ..., x^{M-1}]^T$

Basis function view

$$f(\mathbf{x}) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$

- Basis function $\phi(\mathbf{x}) : \mathbb{R}^D \to \mathbb{R}^M$.

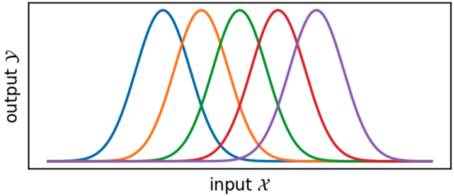
  e.g. $\phi(x) = [1, x, x^2, x^3, ..., x^{M-1}]^T$

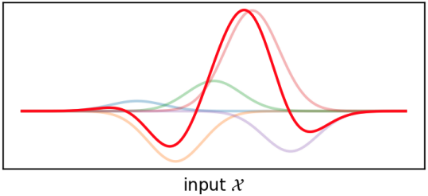
- Prior over $\mathbf{w} \implies$ prior over $f(\cdot)$.

$$p(f(\mathbf{X})) = \int \underbrace{\delta\left[f(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{w}\right]}_{p(f(\mathbf{X})|\mathbf{w})} p(\mathbf{w}) d\mathbf{w}$$

Basis function view

$$f(\mathbf{x}) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$





Left: $\{\phi_m(\mathbf{x})\}_{m=1}^{5}$. **Right**: $f(\mathbf{x}) = \sum_{m=1}^{M} w_m \phi_m(\mathbf{x})$.

Weight space view of Gaussian processes

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \mathbf{I})$$

Weight space view of Gaussian processes

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \mathbf{I})$$

- Function-space prior:

$$p(f(\mathbf{X})) = \int \delta\left[f(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{w}\right] p(\mathbf{w}) d\mathbf{w} = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, \mathbf{K}_{\mathbf{ff}})$$

$$\boldsymbol{\mu} = \Phi(\mathbf{X})\mathbb{E}[\mathbf{w}] = \mathbf{0} \quad \mathbf{K}_{\mathbf{ff}} = \Phi(\mathbf{X})\Phi(\mathbf{X})^T$$

Weight space view of Gaussian processes

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}; \mathbf{0}, \mathbf{I})$$

- Function-space prior:

$$p(f(\mathbf{X})) = \int \delta\left[f(\mathbf{X}) - \Phi(\mathbf{X})\mathbf{w}\right] p(\mathbf{w}) d\mathbf{w} = \mathcal{N}(\mathbf{f}; \boldsymbol{\mu}, \mathbf{K}_{\mathbf{ff}})$$

$$\boldsymbol{\mu} = \Phi(\mathbf{X})\mathbb{E}[\mathbf{w}] = \mathbf{0} \quad \mathbf{K}_{\mathbf{ff}} = \Phi(\mathbf{X})\Phi(\mathbf{X})^T$$

- Equivalent to GP prior with kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = \left[\Phi(\mathbf{X})\Phi(\mathbf{X})^T\right]_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j) = \sum_{m=1}^{M} \phi_m(\mathbf{x}_i)\phi_m(\mathbf{x}_j)$$

Non-Gaussian priors?

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$

$p(\mathbf{w})$ non-Gaussian? $\implies$ non-Gaussian process $p(f(\cdot))$.

Non-Gaussian priors?

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$

$$p(\mathbf{w}) \text{ non-Gaussian? } \implies \text{ non-Gaussian process } p(f(\cdot)).$$

Method for constructing non-Gaussian prior $p(\mathbf{w})$:

$$p_\theta(\mathbf{w}) = \int p_\theta(\mathbf{w}|\mathbf{z}) p(\mathbf{z}) d\mathbf{z}$$

- $p(\mathbf{z})$ simple, i.e. $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$
- $p(\mathbf{w})$ arbitrarily complex.

Non-Gaussian priors?

$$f(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}$$

$p(\mathbf{w})$ non-Gaussian? $\implies$ non-Gaussian process $p(f(\cdot))$.

Method for constructing non-Gaussian prior $p(\mathbf{w})$:

$$p_\theta(\mathbf{w}) = \int p_\theta(\mathbf{w}|\mathbf{z})p(\mathbf{z})d\mathbf{z}$$

- $p(\mathbf{z})$ simple, i.e. $p(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I})$
- $p(\mathbf{w})$ arbitrarily complex.

How to learn $p_\theta(\mathrm{w}|\mathrm{z})$? From data (functions).

Learning non-Gaussian priors

Given K samples from N functions: $\left\{ \underbrace{\{\mathbf{x}_i^n, y_i^n\}_{i=1}^K}_{\text{samples from } f_n(\cdot)} \right\}_{n=1}^N$

Learning non-Gaussian priors

Given K samples from N functions: $\left\{ \underbrace{\{\mathbf{x}_i^n, y_i^n\}_{i=1}^K}_{\text{samples from } f_n(\cdot)} \right\}_{n=1}^N$

- Model as

$$y_i^n = \phi(\mathbf{x}_i^n)^T \mathbf{w}^n + \epsilon_i^n$$

Learning non-Gaussian priors

Given K samples from N functions: $\left\{ \underbrace{\{\mathbf{x}_i^n, y_i^n\}_{i=1}^K}_{\text{samples from } f_n(\cdot)} \right\}_{n=1}^N$

- Model as

$$y_i^n = \phi(\mathbf{x}_i^n)^T \mathbf{w}^n + \epsilon_i^n$$

- ML learning of $\{\mathbf{w}^n\}_{n=1}^N$ and ϕ:

$$\arg\max_{\mathbf{w},\phi} \sum_{n=1}^N \log p(\mathbf{y}^n | \Phi(\mathbf{X}^n), \mathbf{w}^n)$$

Learning non-Gaussian priors

Given K samples from N functions: $\left\{ \underbrace{\{\mathbf{x}_i^n, y_i^n\}_{i=1}^K}_{\text{samples from } f_n(\cdot)} \right\}_{n=1}^N$

- Model as

$$y_i^n = \phi(\mathbf{x}_i^n)^T \mathbf{w}^n + \epsilon_i^n$$

- ML learning of $\{\mathbf{w}^n\}_{n=1}^N$ and ϕ:

$$\arg\max_{\mathbf{w},\phi} \sum_{n=1}^N \log p(\mathbf{y}^n | \Phi(\mathbf{X}^n), \mathbf{w}^n)$$

- Train generative model on $\{\mathbf{w}^n\}_{n=1}^N$ to learn $p_\theta(\mathbf{w}|\mathbf{z})$, i.e. VAE:

$$\arg\max_{\theta,\eta_e} \sum_{n=1}^N \mathbb{E}_{q_{\eta_e}(\mathbf{z}|\mathbf{w}^n)} \left[\log p_\theta(\mathbf{w}^n|\mathbf{z})\right] - \mathrm{KL}\left[q_{\eta_e}(\mathbf{z}|\mathbf{w}^n) || p(\mathbf{z})\right]$$

Efficient posterior inference

Given ϕ and $p_\theta(\mathbf{w}|\mathbf{z}) = \delta\left[d_\theta(\mathbf{z}) - \mathbf{w}\right]$

Efficient posterior inference

Given ϕ **and** $p_\theta(\mathbf{w}|\mathbf{z}) = \delta\left[d_\theta(\mathbf{z}) - \mathbf{w}\right]$

- Sampling functions from **prior**:

$$\mathbf{z}^{(s)} \sim p(\mathbf{z}) \quad \mathbf{w}^{(s)} = d_\theta(\mathbf{z}^{(s)})$$

$$\implies f^{(s)}(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}^{(s)}$$

Efficient posterior inference

Given ϕ **and** $p_\theta(\mathbf{w}|\mathbf{z}) = \delta\left[d_\theta(\mathbf{z}) - \mathbf{w}\right]$

- Sampling functions from **prior**:

$$\mathbf{z}^{(s)} \sim p(\mathbf{z}) \quad \mathbf{w}^{(s)} = d_\theta(\mathbf{z}^{(s)})$$

$$\implies f^{(s)}(\mathbf{x}) = \phi(\mathbf{x})^T \mathbf{w}^{(s)}$$

- Sampling functions from **posterior**? Perform MCMC in latent space:

$$p(\mathbf{z}|\mathcal{D}_*, \phi) \propto p(\mathbf{y}_*|\Phi(\mathbf{X}_*), \mathbf{z})p(\mathbf{z})$$

$$\mathbf{z}^{(s)}|\mathcal{D} \sim p(\mathbf{z}|\mathcal{D}_*, \phi) \quad \mathbf{w}^{(s)} = d_\theta(\mathbf{z}^{(s)})$$

$$\implies f^{(s)}(\mathbf{x}) \mid \mathcal{D}_* = \phi(\mathbf{x})^T \mathbf{w}^{(s)}$$

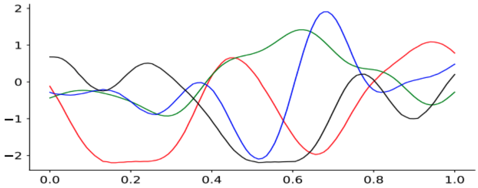
πVAE: end-to-end training (Mishra et al. 2020)

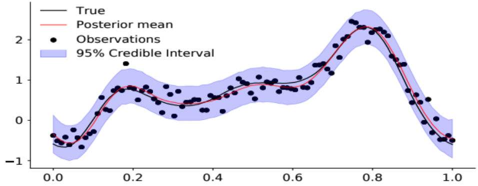
$$\mathcal{L} = \sum_{n=1}^{N} \log p(\mathbf{y}^n|\Phi(\mathbf{X}^n), \mathbf{w}^n) + \mathbb{E}_{q_{\eta_e}(\mathbf{z}|\mathbf{w}^n)p_\theta(\hat{\mathbf{w}}^n|\mathbf{z})}\left[\log p(\mathbf{y}^n|\Phi(\mathbf{X}^n), \hat{\mathbf{w}}^n)\right]$$
$$-\mathrm{KL}\left[q_{\eta_e}(\mathbf{z}|\mathbf{w}^n)||p(\mathbf{z})\right]$$

- $\log p(\mathbf{y}|\Phi(\mathbf{X}), \mathbf{w}) \implies \phi$ and $\mathbf{w}$ explain the data.
- $\mathbb{E}_{q_{\eta_e}(\mathbf{z}|\mathbf{w})}\left[\mathbb{E}_{p_\theta(\hat{\mathbf{w}}|\mathbf{z})}\left[\log p(\mathbf{y}|\Phi(\mathbf{X}), \hat{\mathbf{w}}\right]\right] \implies \phi$ and reconstructed $\mathbf{w}$ explain the data.
- $\mathrm{KL}\left[q_{\eta_e}(\mathbf{z}|\mathbf{w})||p(\mathbf{z})\right] \implies q_{\eta_e}(\mathbf{z}|\mathbf{w})$ close to the prior $p(\mathbf{z})$.

πVAE: learning a GP prior (Mishra et al. 2020)

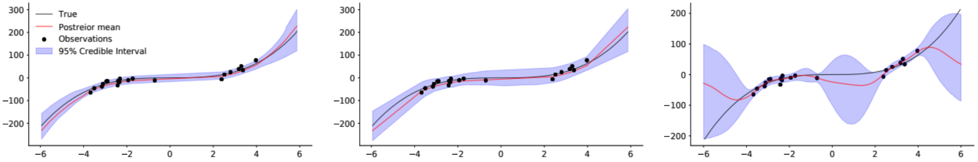
πVAE trained on Gaussian process samples





Left: prior samples. **Right**: posterior predictive distribution.

πVAE: posterior inference (Mishra et al. 2020)



Left: πVAE, samples from cubic functions. **Middle**: πVAE, samples from RBF kernel. **Right:** GP with RBF kernel

Stochastic process generator (Ma et al.)

Approximate stochastic process posterior

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})} \approx q_{\mathsf{SPG}}(f)$$

Stochastic process generator (Ma et al.)

Approximate stochastic process posterior

$$p(f|\mathcal{D}) = \frac{p(f)p(\mathcal{D}|f)}{p(\mathcal{D})} \approx q_{\mathsf{SPG}}(f)$$

'Stochastic process generator' (SPG) family:

$$f_{\mathsf{SPG}}(\mathbf{x}) = \sum_{m=1} w_m \phi_m(\mathbf{x}), \quad q(\mathbf{w}) = \int p_\theta(\mathbf{w}|\mathbf{z}) q(\mathbf{z}) d\mathbf{z}$$

- Non-Gaussian $q(\mathbf{w}) \implies$ non-Gaussian process $q_{\mathsf{SPG}}(f)$.
- ϕ_m are a set of trainable basis functions.

Functional variational inference

Function space ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(f)}\left[\log p(\mathcal{D}|f)\right]}_{\text{nice}} - \underbrace{\mathrm{KL}\left[q(f)||p(f)\right]}_{\text{not nice}}$$

Functional variational inference

Function space ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(f)}\left[\log p(\mathcal{D}|f)\right]}_{\text{nice}} - \underbrace{\mathrm{KL}\left[q(f)||p(f)\right]}_{\text{not nice}}$$

KL between stochastic processes? Sun et al. (2019):

$$\mathrm{KL}\left[q(f)||p(f)\right] = \sup_{n\in\mathbb{N},\mathbf{X}\in\mathcal{X}^n} \mathrm{KL}\left[q(f(\mathbf{X}))||p(f(\mathbf{X}))\right]$$

Functional variational inference

Function space ELBO:

$$\mathcal{L} = \underbrace{\mathbb{E}_{q(f)} \left[\log p(\mathcal{D}|f)\right]}_{\text{nice}} - \underbrace{\mathrm{KL}\left[q(f)||p(f)\right]}_{\text{not nice}}$$

KL between stochastic processes? Sun et al. (2019):

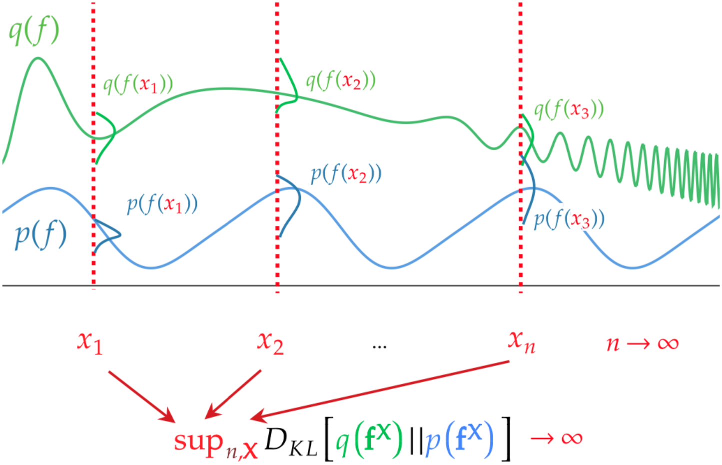
$$\mathrm{KL}\left[q(f)||p(f)\right] = \sup_{n\in\mathbb{N}, \mathbf{X}\in\mathcal{X}^n} \mathrm{KL}\left[q(f(\mathbf{X}))||p(f(\mathbf{X}))\right]$$

Can't compute supremum! $\implies$ **approximate with**

$$\mathrm{KL}\left[q(f)||p(f)\right] \geq \mathbb{E}_{\mathbf{X}_O\sim c}\left[\mathrm{KL}\left[q(f(\mathbf{X}_O))||p(f(\mathbf{X}_O))\right]\right]$$

(However, true KL may not be finite...)

Functional variational inference



FVI via SPGs (Ma et al.)

$$f_{\mathsf{SPG}}(\mathbf{x}) = \sum_{m=1} w_m \phi_m(\mathbf{x}), \quad q(\mathbf{w}) = \int p_\theta(\mathbf{w}|\mathbf{z}) q(\mathbf{z}) d\mathbf{z}$$

FVI via SPGs (Ma et al.)

$$f_{\mathsf{SPG}}(\mathbf{x}) = \sum_{m=1} w_m \phi_m(\mathbf{x}), \quad q(\mathbf{w}) = \int p_\theta(\mathbf{w}|\mathbf{z}) q(\mathbf{z}) d\mathbf{z}$$

- Approximate prior $p(f) \approx \tilde{p}_{\mathsf{SPG}}(f)$.

$\Longrightarrow$ Learns $\{\phi_m\}_{m=1}^M$, $p_\theta(\mathbf{w}|\mathbf{z})$ and $\tilde{q}(\mathbf{z}|f(\mathbf{X}_O)) \approx \tilde{p}_{\mathsf{SPG}}(\mathbf{z}|f(\mathbf{X}_O))$ through VAE-like ELBO.

FVI via SPGs (Ma et al.)

$$f_{\mathsf{SPG}}(\mathbf{x}) = \sum_{m=1} w_m \phi_m(\mathbf{x}), \quad q(\mathbf{w}) = \int p_\theta(\mathbf{w}|\mathbf{z}) q(\mathbf{z}) d\mathbf{z}$$

- Approximate prior $p(f) \approx \tilde{p}_{\mathsf{SPG}}(f)$.

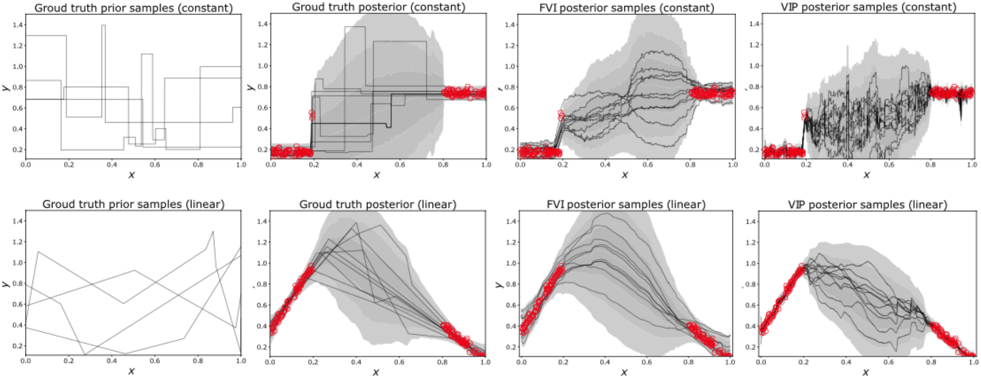
  $\Longrightarrow$ Learns $\{\phi_m\}_{m=1}^M$, $p_\theta(\mathbf{w}|\mathbf{z})$ and $\tilde{q}(\mathbf{z}|f(\mathbf{X}_O)) \approx \tilde{p}_{\mathsf{SPG}}(\mathbf{z}|f(\mathbf{X}_O))$ through VAE-like ELBO.

- Share $\{\phi_m\}_{m=1}^M$ and $p_\theta(\mathbf{w}|\mathbf{z})$ between $p_{\mathsf{SPG}}(f)$ and $q_{\mathsf{SPG}}(f)$.

  $\Longrightarrow$ Simplifies KL divergence:

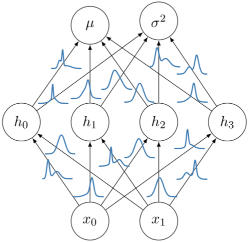
$$\begin{aligned}&\mathrm{KL}\left[q_{\mathsf{SPG}}(f(\mathbf{X}_O)) || p_{\mathsf{SPG}}(f(\mathbf{X}_O))\right] \\ &\qquad \approx \mathbb{E}_{p_{\mathsf{SPG}}(f(\mathbf{X}_O))}\left[\int \tilde{q}(\mathbf{z}|f(\mathbf{X}_O)) \frac{q(\mathbf{z})}{p_0(\mathbf{z})}\right] d\mathbf{z}\end{aligned}$$

FVI via SPGs (Ma et al.)



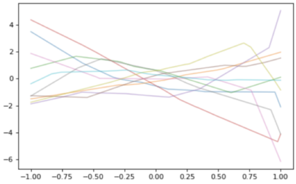
Neural Networks as Stochastic Processes

Functions sampled from NN prior:

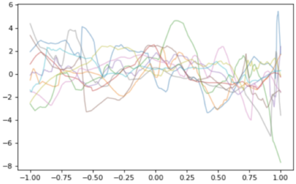


$$p(\theta) = \mathcal{N}(\theta; 0, I)$$

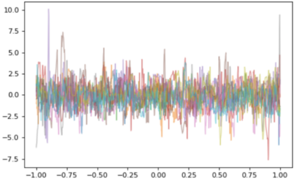
1 Hidden Layer



5 Hidden Layer



20 Hidden Layer



Can NNs be viewed as linear basis function models?

If we take a first order Taylor expansion of network outputs with respect to their weights we obtain a basis function linear model:

Can NNs be viewed as linear basis function models?

If we take a first order Taylor expansion of network outputs with respect to their weights we obtain a basis function linear model:

- Lets define some NN $f_\theta(\cdot)$ and some weight setting θ^*:

$$f_\theta(x) \approx f_\theta^{\text{lin}}(x) = f_{\theta^*}(x) + \left(\frac{\partial f_{\theta^*}(x)}{\partial \theta^*}\right)^\mathsf{T} (\theta - \theta^*)$$

- $f_\theta^{\text{lin}}(x)$ is a linear model in θ with basis functions $\phi(x) = \frac{\partial f_{\theta^*}(x)}{\partial \theta^*}$ [7].
- This corresponds to a GP with $k(x_1, x_2) = \left(\frac{\partial f_{\theta^*}(x_1)}{\partial \theta^*}\right)^\mathsf{T} \left(\frac{\partial f_{\theta^*}(x_2)}{\partial \theta^*}\right)$!

Can NNs be viewed as linear basis function models?

If we take a first order Taylor expansion of network outputs with respect to their weights we obtain a basis function linear model:

- Lets define some NN $f_\theta(\cdot)$ and some weight setting θ^*:

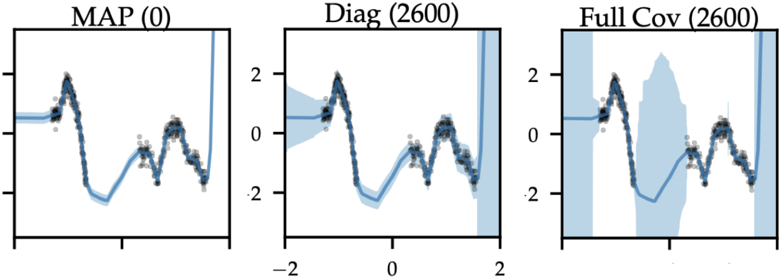
$$f_\theta(x) \approx f_\theta^{\text{lin}}(x) = f_{\theta^*}(x) + \left(\frac{\partial f_{\theta^*}(x)}{\partial \theta^*}\right)^\mathsf{T} (\theta - \theta^*)$$

- $f_\theta^{\text{lin}}(x)$ is a linear model in θ with basis functions $\phi(x) = \frac{\partial f_{\theta^*}(x)}{\partial \theta^*}$ [7].

- This corresponds to a GP with $k(x_1, x_2) = \left(\frac{\partial f_{\theta^*}(x_1)}{\partial \theta^*}\right)^\mathsf{T} \left(\frac{\partial f_{\theta^*}(x_2)}{\partial \theta^*}\right)$!

Could also make an argument about infinite width limits...

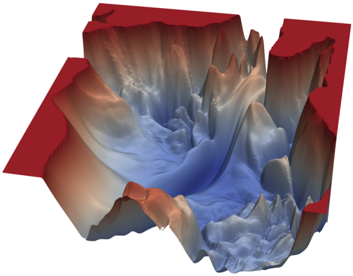
A Linearised NN-GP in action (Daxberger et. al.)

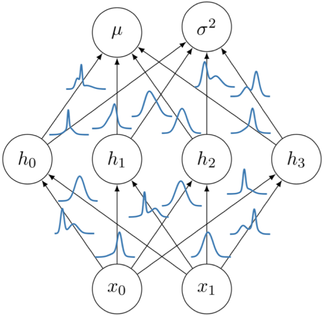


Inference in finite, non-Linearised NNs

- They are not GPs.

- Probabilistic inference over their weight space is intractable.

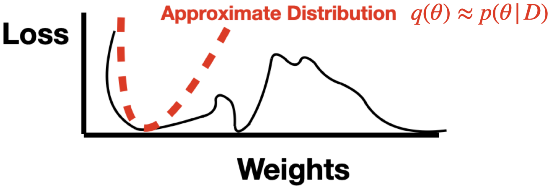


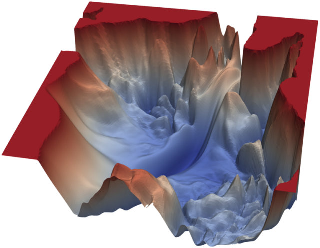


[Li et. al.]

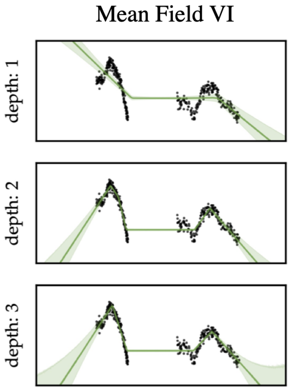
Lets try weight Space VI

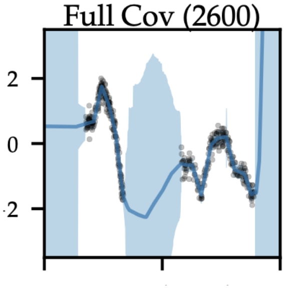
$$p(\theta|\mathcal{D}) \geq \mathsf{ELBO}_{q(\theta)} = E_{q(\theta)}[\log p(\mathcal{D}|\theta)] - KL(q(\theta)\,||\,p(\theta))$$





Empirical Underperformance





We know Mean Field distributions are flexible enough to do better [5]. It looks like the problem is the inference!

We resort to functional variational inference!

The functional posterior is intractable for NNs so we again resort to **functional VI** [12].

$$p(\theta|\mathcal{D}) \geq \mathsf{ELBO}_{q(f)} = E_{q(f))}[\log p(\mathcal{D}|f)] - KL(q\,||\,p);$$

$$KL(q\,||\,p) = \sup_{n\in\mathbb{N}, X\in\mathcal{X}^n} D_{\mathsf{KL}}(q(f(X))\,||\,p(f(X)))$$

We resort to functional variational inference!

The functional posterior is intractable for NNs so we again resort to **functional VI** [12].

$$p(\theta|\mathcal{D}) \geq \mathsf{ELBO}_{q(f)} = E_{q(f))}[\log p(\mathcal{D}|f)] - KL(q\,||\,p);$$

$$KL(q\,||\,p) = \sup_{n\in\mathbb{N}, X\in\mathcal{X}^n} D_{\mathsf{KL}}(q(f(X))\,||\,p(f(X)))$$

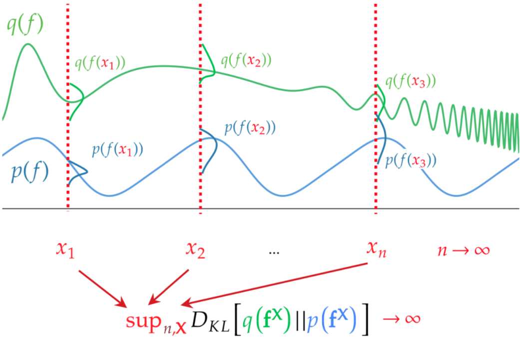
By the information processing inequality, this should yield a tighter ELBO than weight space VI [2].

$$\log p(\mathcal{D}) \geq \mathsf{ELBO}_{q(f)} \geq \mathsf{ELBO}_{q(\theta)} \tag{1}$$

Intuition: Different parameter settings induce different functions which explain the data.

$$\theta \to f \to y; \quad I(y:f) \geq I(y:\theta) \tag{2}$$

The functional KL, again



- Supremum over all input sets is intractable to compute!
- Functional KL between GP and parametric models or between different parametric models may not even be finite [2].

Approximations used by Sun et. al.

- Supremum formulation of functional KL suggests an **adversarial learning scheme**: One player chooses approximate process q and the other chooses the measurement set X.

$$\max_{q} \min_{X \in \mathcal{X}^n} E_{q(f))}[\log p(\mathcal{D}|f)] - D_{\mathsf{KL}}(q(f(X)) \,||\, p(f(X)))$$

  Sun et. al. find this to not work well in practise.

Approximations used by Sun et. al.

- Supremum formulation of functional KL suggests an **adversarial learning scheme**: One player chooses approximate process q and the other chooses the measurement set X.

$$\max_{q} \min_{X \in \mathcal{X}^n} E_{q(f))}[\log p(\mathcal{D}|f)] - D_{\mathsf{KL}}(q(f(X)) \,||\, p(f(X)))$$

  Sun et. al. find this to not work well in practise.

- **Sampling-based measurement sets**: Define a sampling distribution c from which to draw X.

$$\max_{q} E_{q(f)}[\log p(\mathcal{D}|f)] - \mathbb{E}_{X\sim c}[D_{\mathsf{KL}}(q(f(X)) \,||\, p(f(X)))]$$

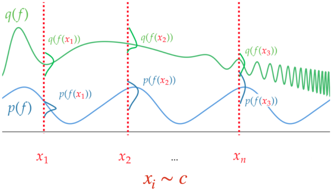
A remaining issue might be estimating $q(f(X))$...

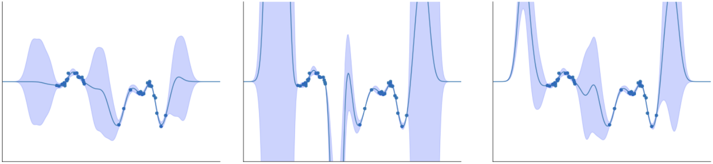
Choosing measurement points

Sun et. al. show that the resulting objective is still a lower bound on $\log p(\mathcal{D})$ as long as $X_{\mathcal{D}} \subset X$.

Burt et. al. compare approaches on linear models:

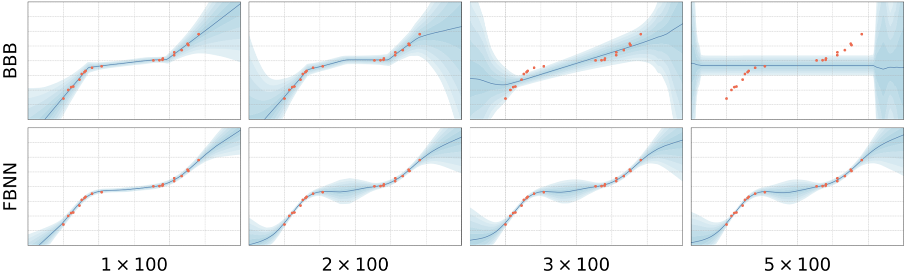
- Randomly sample X once and leave it **fixed**.
- Resample $X \sim c$ in every iteration (**random**.)





(a) Exact $\quad$ (b) FixedA $\quad$ (c) RandA

Results: BNNs with Random sampling (Sun et. al.)



- Approximate functional VI is more flexible than weight space VI.
- In agreement with [5], at least 2 hidden layers are needed to capture in between uncertainty with mean field weight parametrisation.
- Is functional VI a practical approach?

Stochastic Differential Equations

Stochastic Differential Equations

Appropriate when the data generating process is

Stochastic Differential Equations

Appropriate when the data generating process is

1. In continuous time

Stochastic Differential Equations

Appropriate when the data generating process is

1. In continuous time
2. Causal

Stochastic Differential Equations

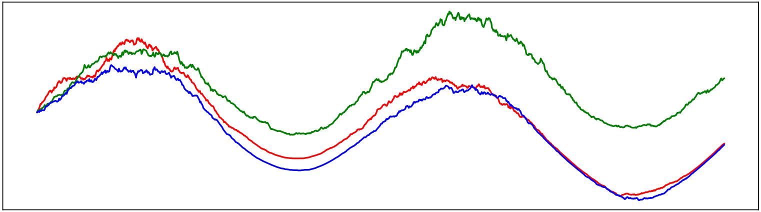
Appropriate when the data generating process is

1. In continuous time
2. Causal
3. Partly driven by noise

Stochastic Differential Equations

Appropriate when the data generating process is

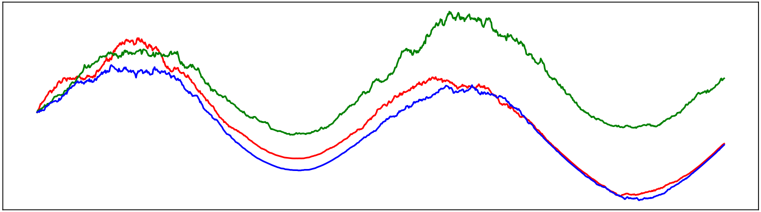
1. In continuous time
2. Causal
3. Partly driven by noise



Stochastic Differential Equations

Appropriate when the data generating process is

1. In continuous time
2. Causal
3. Partly driven by noise

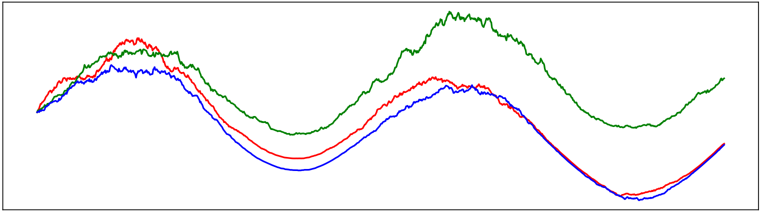


Applications satisfying these criteria:

Stochastic Differential Equations

Appropriate when the data generating process is

1. In continuous time
2. Causal
3. Partly driven by noise



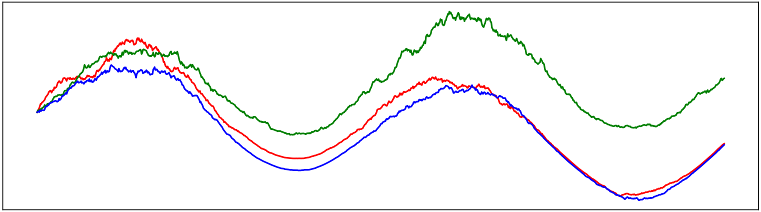
Applications satisfying these criteria:

1. Tracking and location

Stochastic Differential Equations

Appropriate when the data generating process is

1. In continuous time
2. Causal
3. Partly driven by noise



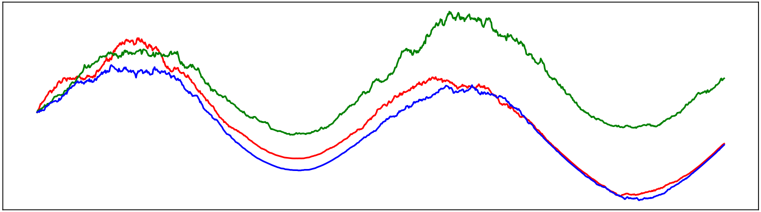
Applications satisfying these criteria:

1. Tracking and location
2. Medical

Stochastic Differential Equations

Appropriate when the data generating process is

1. In continuous time
2. Causal
3. Partly driven by noise

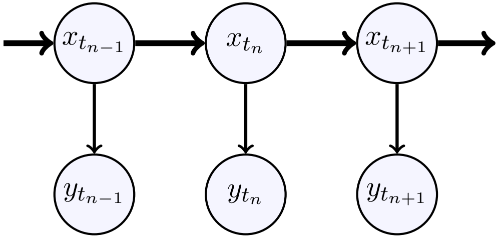


Applications satisfying these criteria:

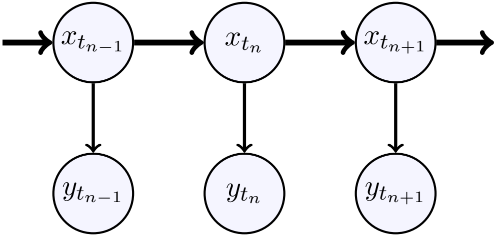
1. Tracking and location
2. Medical
3. Physical models, e.g. weather and climate

Stochastic Differential Equations

Stochastic Differential Equations

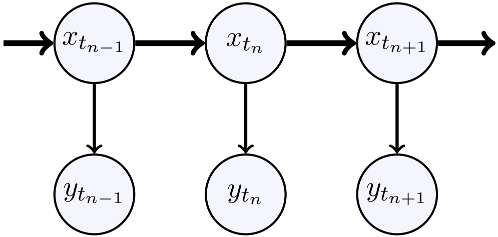


Stochastic Differential Equations



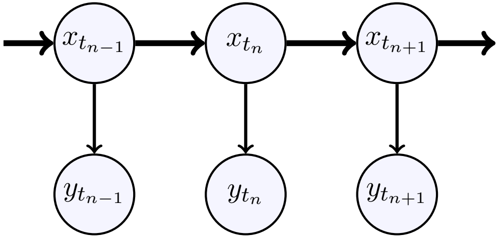
- SDE governs dynamics of latent x_t.

Stochastic Differential Equations



- SDE governs dynamics of latent x_t.
- Observation model $p(y_t|x_t)$ generates observed y_t.

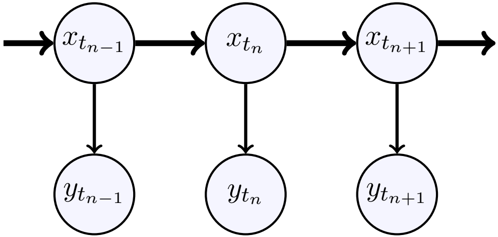
Stochastic Differential Equations



- SDE governs dynamics of latent x_t.
- Observation model $p(y_t|x_t)$ generates observed y_t.

What do we gain?

Stochastic Differential Equations

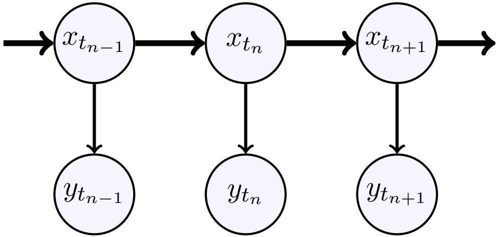


- SDE governs dynamics of latent x_t.
- Observation model $p(y_t|x_t)$ generates observed y_t.

What do we gain?

- Better inductive biases.

Stochastic Differential Equations

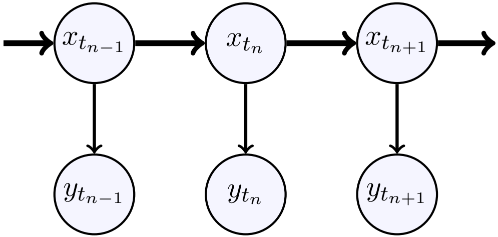


- SDE governs dynamics of latent x_t.
- Observation model $p(y_t|x_t)$ generates observed y_t.

What do we gain?

- Better inductive biases.
- Bake prior beliefs into the model.

Stochastic Differential Equations



- SDE governs dynamics of latent x_t.
- Observation model $p(y_t|x_t)$ generates observed y_t.

What do we gain?

- Better inductive biases.
- Bake prior beliefs into the model.
- Principled handling of irregularly spaced data.

An introduction to SDEs

An introduction to SDEs

We call x_t the solution to an SDE with drift f and diffusion g if

$$x_t = x_0 + \int_{t_0}^{t_1} f(x_\tau, \tau) d\tau + \int_{t_0}^{t_1} g(x_\tau, \tau) d\beta_\tau$$

where β_t is a standard Brownian motion

An introduction to SDEs

We call x_t the solution to an SDE with drift f and diffusion g if

$$x_t = x_0 + \int_{t_0}^{t_1} f(x_\tau, \tau) d\tau + \int_{t_0}^{t_1} g(x_\tau, \tau) d\beta_\tau$$

where β_t is a standard Brownian motion, with the properties

An introduction to SDEs

We call x_t the solution to an SDE with drift f and diffusion g if

$$x_t = x_0 + \int_{t_0}^{t_1} f(x_\tau, \tau) d\tau + \int_{t_0}^{t_1} g(x_\tau, \tau) d\beta_\tau$$

where β_t is a standard Brownian motion, with the properties

- **Initialisation:** $\beta_0 = 0$.

An introduction to SDEs

We call x_t the solution to an SDE with drift f and diffusion g if

$$x_t = x_0 + \int_{t_0}^{t_1} f(x_\tau, \tau) d\tau + \int_{t_0}^{t_1} g(x_\tau, \tau) d\beta_\tau$$

where β_t is a standard Brownian motion, with the properties

- **Initialisation:** $\beta_0 = 0$.
- **Rate of change:** $\beta_{t_2} - \beta_{t_1} \sim \mathcal{N}(0, t_2 - t_1)$ where $t_1 < t_2$.

An introduction to SDEs

We call x_t the solution to an SDE with drift f and diffusion g if

$$x_t = x_0 + \int_{t_0}^{t_1} f(x_\tau, \tau) d\tau + \int_{t_0}^{t_1} g(x_\tau, \tau) d\beta_\tau$$

where β_t is a standard Brownian motion, with the properties

- **Initialisation:** $\beta_0 = 0$.
- **Rate of change:** $\beta_{t_2} - \beta_{t_1} \sim \mathcal{N}(0, t_2 - t_1)$ where $t_1 < t_2$.
- **Independence:** $\beta_{t_3} \perp \beta_{t_1} | \beta_{t_2}$ whenever $t_1 < t_2 < t_3$.

An introduction to SDEs

We call x_t the solution to an SDE with drift f and diffusion g if

$$x_t = x_0 + \int_{t_0}^{t_1} f(x_\tau, \tau) d\tau + \int_{t_0}^{t_1} g(x_\tau, \tau) d\beta_\tau$$

where β_t is a standard Brownian motion, with the properties

- **Initialisation:** $\beta_0 = 0$.
- **Rate of change:** $\beta_{t_2} - \beta_{t_1} \sim \mathcal{N}(0, t_2 - t_1)$ where $t_1 < t_2$.
- **Independence:** $\beta_{t_3} \perp \beta_{t_1} | \beta_{t_2}$ whenever $t_1 < t_2 < t_3$.

For the moment, think the integrals as left-endpoint Riemann integrals

$$\int_a^b g(x_\tau, \tau) d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} g(x_{\tau_k}, \tau)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

where $a = \tau_1 < ... < \tau_N = b, \Delta = \max\{\tau_{k+1} - \tau_k\}$.

An introduction to SDEs

We call x_t the solution to an SDE with drift f and diffusion g if

$$x_t = x_0 + \int_{t_0}^{t_1} f(x_\tau, \tau) d\tau + \int_{t_0}^{t_1} g(x_\tau, \tau) d\beta_\tau$$

where β_t is a standard Brownian motion, with the properties

- **Initialisation:** $\beta_0 = 0$.
- **Rate of change:** $\beta_{t_2} - \beta_{t_1} \sim \mathcal{N}(0, t_2 - t_1)$ where $t_1 < t_2$.
- **Independence:** $\beta_{t_3} \perp \beta_{t_1} | \beta_{t_2}$ whenever $t_1 < t_2 < t_3$.

For the moment, think the integrals as left-endpoint Riemann integrals

$$\int_a^b g(x_\tau, \tau) d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} g(x_{\tau_k}, \tau)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

where $a = \tau_1 < ... < \tau_N = b, \Delta = \max\{\tau_{k+1} - \tau_k\}$. Alternatively written

$$dx_t = f(x_t, t)dt + g(x_t, t)d\beta_t.$$

Linear SDEs and GPs

Linear SDEs and GPs

When the SDE is linear

$$dx_t = F(t)x_t dt + G(t) d\beta_t,$$

x_t **is a GP**, which is also Markovian.

Linear SDEs and GPs

When the SDE is linear

$$dx_t = F(t)x_t dt + G(t)d\beta_t,$$

x_t **is a GP**, which is also Markovian. It is sometimes possible to convert a GP kernel to an equivalent SDE – reduces complexity from $\mathcal{O}(T)$. [6]

Linear SDEs and GPs

When the SDE is linear

$$dx_t = F(t)x_t dt + G(t)d\beta_t,$$

x_t **is a GP**, which is also Markovian. It is sometimes possible to convert a GP kernel to an equivalent SDE – reduces complexity from $\mathcal{O}(T)$. [6]

$$\textbf{Solution:} \quad x_t = \Psi(t, t_0)x_0 + \int_{t_0}^{t} \underbrace{\Psi(t, \tau)}_{\text{Impulse response fn.}} G(\tau)d\beta_\tau.$$

When $F(t) = F$, impulse response is $\Psi(t, t') = \exp[(t - t')F]$.

Linear SDEs and GPs

When the SDE is linear

$$dx_t = F(t)x_t dt + G(t)d\beta_t,$$

x_t **is a GP**, which is also Markovian. It is sometimes possible to convert a GP kernel to an equivalent SDE – reduces complexity from $\mathcal{O}(T)$. [6]

$$\textbf{Solution:} \quad x_t = \Psi(t, t_0)x_0 + \int_{t_0}^{t} \underbrace{\Psi(t, \tau)}_{\text{Impulse response fn.}} G(\tau)d\beta_\tau.$$

When $F(t) = F$, impulse response is $\Psi(t, t') = \exp[(t - t')F]$.

Given a factorising Gaussian observation model $p(y|x) = \prod_{n=1}^{N} p(y_n|x_n)$

Linear SDEs and GPs

When the SDE is linear

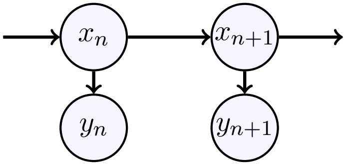
$$dx_t = F(t)x_t dt + G(t)d\beta_t,$$

x_t **is a GP**, which is also Markovian. It is sometimes possible to convert a GP kernel to an equivalent SDE – reduces complexity from $\mathcal{O}(T)$. [6]

Solution: $$x_t = \Psi(t, t_0)x_0 + \int_{t_0}^{t} \underbrace{\Psi(t, \tau)}_{\text{Impulse response fn.}} G(\tau)d\beta_\tau.$$

When $F(t) = F$, impulse response is $\Psi(t, t') = \exp[(t - t')F]$.

Given a factorising Gaussian observation model $p(y|x) = \prod_{n=1}^{N} p(y_n|x_n)$



Linear SDEs and GPs

When the SDE is linear

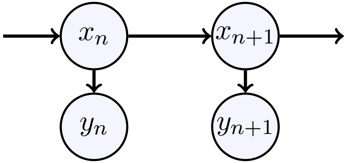
$$dx_t = F(t)x_t dt + G(t)d\beta_t,$$

x_t **is a GP**, which is also Markovian. It is sometimes possible to convert a GP kernel to an equivalent SDE – reduces complexity from $\mathcal{O}(T)$. [6]

$$\textbf{Solution:} \quad x_t = \Psi(t, t_0)x_0 + \int_{t_0}^{t} \underbrace{\Psi(t, \tau)}_{\text{Impulse response fn.}} G(\tau)d\beta_\tau.$$

When $F(t) = F$, impulse response is $\Psi(t, t') = \exp[(t - t')F]$.

Given a factorising Gaussian observation model $p(y|x) = \prod_{n=1}^{N} p(y_n|x_n)$



Compute posterior in $\mathcal{O}(T)$ time. (Kalman filtering/smoothing) [3, 10]

Learning SDEs with non-parametric priors [14]

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Conditioned on $h(t)$, x_t is a GP, with kernel

$$k_{f|h}(t_1, t_2) = h(t) * h(-t), \text{ where } t = |t_2 - t_1|.$$

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Conditioned on $h(t)$, x_t is a GP, with kernel

$$k_{f|h}(t_1, t_2) = h(t) * h(-t), \text{ where } t = |t_2 - t_1|.$$

Three challenges:

1. x_t depends on convolution between infinite-dimensional h and β_t.

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Conditioned on $h(t)$, x_t is a GP, with kernel

$$k_{f|h}(t_1, t_2) = h(t) * h(-t), \text{ where } t = |t_2 - t_1|.$$

Three challenges:

1. x_t depends on convolution between infinite-dimensional h and β_t.
2. The noise increments $d\beta_t$ must be treated carefully.

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Conditioned on $h(t)$, x_t is a GP, with kernel

$$k_{f|h}(t_1, t_2) = h(t) * h(-t), \text{ where } t = |t_2 - t_1|.$$

Three challenges:

1. x_t depends on convolution between infinite-dimensional h and β_t.
2. The noise increments $d\beta_t$ must be treated carefully.
3. x_t is nonlinear in h, β_t – marginal likelihood is intractable.

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Conditioned on $h(t)$, x_t is a GP, with kernel

$$k_{f|h}(t_1, t_2) = h(t) * h(-t), \text{ where } t = |t_2 - t_1|.$$

Three challenges:

1. x_t depends on convolution between infinite-dimensional h and β_t.
2. The noise increments $d\beta_t$ must be treated carefully.
3. x_t is nonlinear in h, β_t – marginal likelihood is intractable.

Solution: 1 use sparse GP inducing points [13],

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Conditioned on $h(t)$, x_t is a GP, with kernel

$$k_{f|h}(t_1, t_2) = h(t) * h(-t), \text{ where } t = |t_2 - t_1|.$$

Three challenges:

1. x_t depends on convolution between infinite-dimensional h and β_t.
2. The noise increments $d\beta_t$ must be treated carefully.
3. x_t is nonlinear in h, β_t – marginal likelihood is intractable.

Solution: 1 use sparse GP inducing points [13], 2 to infer h and a smoothed version of β_t,

Learning SDEs with non-parametric priors [14]

Assume the GP convolution model (GPCM) of the form

$$x_t = \int_{-\infty}^{\infty} h(\tau) d\beta_\tau, \text{ where } h \sim \mathcal{GP}(0, k_h).$$

Conditioned on $h(t)$, x_t is a GP, with kernel

$$k_{f|h}(t_1, t_2) = h(t) * h(-t), \text{ where } t = |t_2 - t_1|.$$

Three challenges:

1. x_t depends on convolution between infinite-dimensional h and β_t.
2. The noise increments $d\beta_t$ must be treated carefully.
3. x_t is nonlinear in h, β_t – marginal likelihood is intractable.

Solution: 1 use sparse GP inducing points [13], 2 to infer h and a smoothed version of β_t, 3 via the Evidence Lower Bound.

Learning SDEs with non-parametric priors [14]

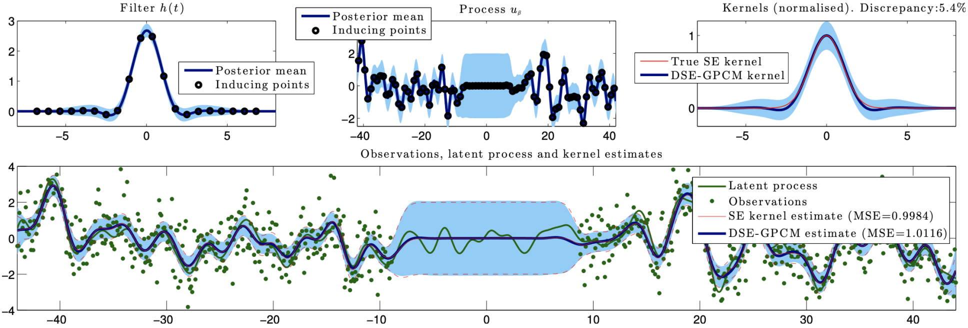


Figure 1: Posterior inference in GPCM from [14] (edited).

Variational Inference for SDEs [1]

Variational Inference for SDEs [1]

Given an SDE and observation model

$$dx_t = f(x_t, t)dt + \Sigma^{1/2} d\beta_t \qquad \text{(prior SDE, p)}$$

$$y_n = Hx_n + \sigma_n \epsilon, \text{ where } \epsilon \sim \mathcal{N}(0, 1). \qquad \text{(observation model)}$$

Variational Inference for SDEs [1]

Given an SDE and observation model

$$dx_t = f(x_t, t)dt + \Sigma^{1/2} d\beta_t \qquad \text{(prior SDE, p)}$$

$$y_n = Hx_n + \sigma_n \epsilon, \text{ where } \epsilon \sim \mathcal{N}(0, 1). \qquad \text{(observation model)}$$

Approximate its posterior using the linear SDE

$$dx_t = \underbrace{(-A(t)x_t + b(t))}_{g(x_t, t)} dt + \Sigma^{1/2} d\beta, \qquad \text{(approximating SDE, q)}$$

Variational Inference for SDEs [1]

Given an SDE and observation model

$$\begin{aligned} dx_t &= f(x_t, t)dt + \Sigma^{1/2} d\beta_t && \text{(prior SDE, p)} \\ y_n &= Hx_n + \sigma_n \epsilon, \ \text{where } \epsilon \sim \mathcal{N}(0, 1). && \text{(observation model)} \end{aligned}$$

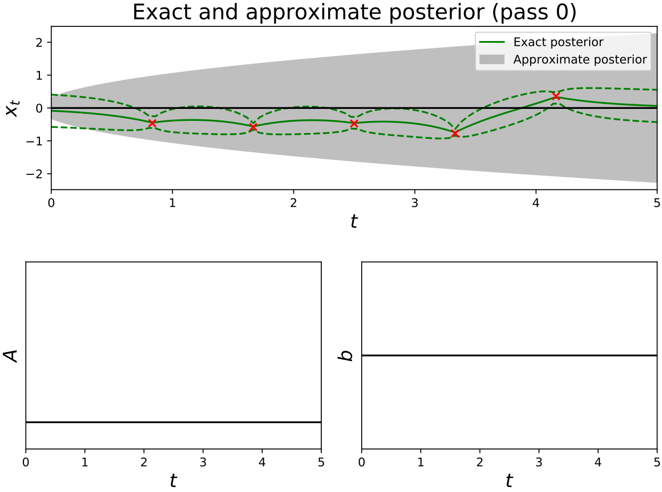
Approximate its posterior using the linear SDE

$$dx_t = \underbrace{(-A(t)x_t + b(t))}_{g(x_t,t)} dt + \Sigma^{1/2} d\beta, \qquad \text{(approximating SDE, q)}$$

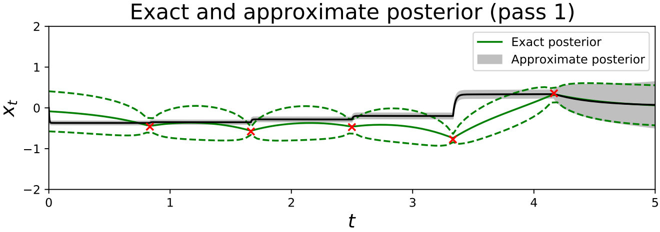
write $q(x_t)$ for its marginal distribution. KL between the exact SDE prior and the approximating SDE [1] is

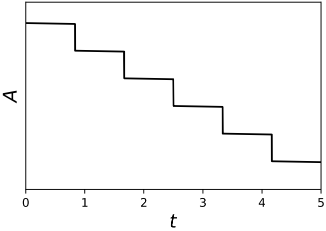
$$\begin{aligned} KL\left[q || p\right] &= KL\left[q(x_0) || p(x_0)\right] + \\ &+ \frac{1}{2} \int_{t_0}^{t_1} \int (f(x, \tau) - g(x, \tau))^\top \Sigma^{-1} (f(x, \tau) - g(x, \tau)) q(x_n) dx_n d\tau. \end{aligned}$$

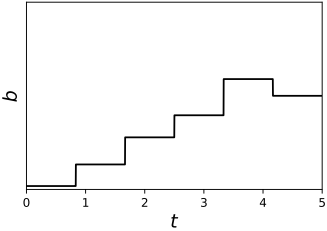
Variational Inference for SDEs [1]



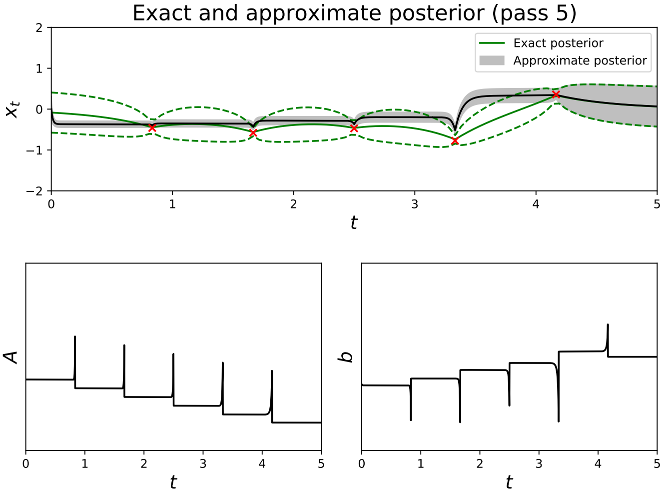
Variational Inference for SDEs [1]



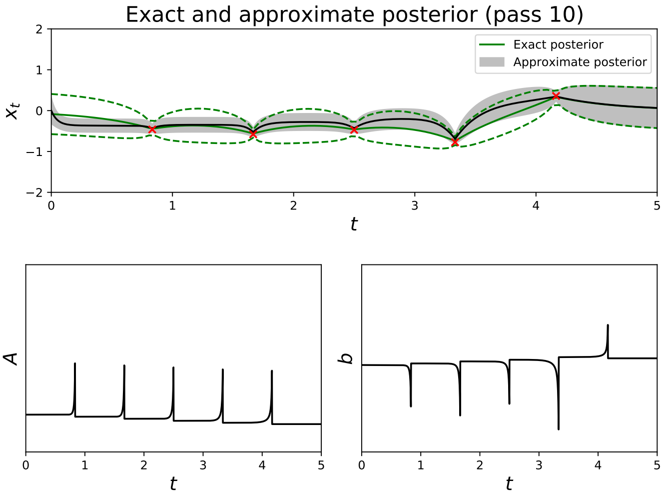




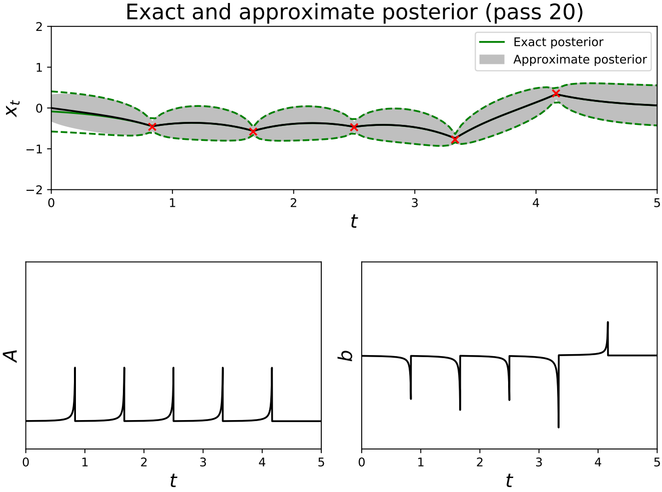
Variational Inference for SDEs [1]



Variational Inference for SDEs [1]



Variational Inference for SDEs [1]



Stochastic integrals

Stochastic integrals

Suppose we want to compute the integral

$$\int_a^b \Phi(x_\tau, \tau) d\beta_\tau.$$

Stochastic integrals

Suppose we want to compute the integral

$$\int_a^b \Phi(x_\tau, \tau) d\beta_\tau.$$

Ito definition (one we saw earlier)

Stochastic integrals

Suppose we want to compute the integral

$$\int_a^b \Phi(x_\tau, \tau) d\beta_\tau.$$

Ito definition (one we saw earlier)

$$\int \Phi(x_\tau, \tau) d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} \Phi(x_{\tau_k}, \tau)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

Stochastic integrals

Suppose we want to compute the integral

$$\int_a^b \Phi(x_\tau, \tau) d\beta_\tau .$$

Ito definition (one we saw earlier)

$$\int \Phi(x_\tau, \tau) d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} \Phi(x_{\tau_k}, \tau)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

Stratonovich definition

Stochastic integrals

Suppose we want to compute the integral

$$\int_a^b \Phi(x_\tau, \tau) d\beta_\tau.$$

Ito definition (one we saw earlier)

$$\int \Phi(x_\tau, \tau) d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} \Phi(x_{\tau_k}, \tau)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

Stratonovich definition

$$\int \Phi(x_\tau, \tau) \circ d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} \Phi\left(\frac{x_{\tau_{k+1}} + x_{\tau_k}}{2}, t\right)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

Stochastic integrals

Suppose we want to compute the integral

$$\int_a^b \Phi(x_\tau, \tau) d\beta_\tau.$$

Ito definition (one we saw earlier)

$$\int \Phi(x_\tau, \tau) d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} \Phi(x_{\tau_k}, \tau)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

Stratonovich definition

$$\int \Phi(x_\tau, \tau) \circ d\beta_\tau = \lim_{|\Delta| \to 0} \sum_{n=1}^{N} \Phi\left(\frac{x_{\tau_{k+1}} + x_{\tau_k}}{2}, t\right)(\beta_{\tau_{k+1}} - \beta_{\tau_k})$$

In regular calculus, these are equivalent. In stochastic calculus they are not – they give different answers!

Stochastic integrals

Stochastic integrals

In particular, the chain rule is different under Ito and Stratonovich.

Stochastic integrals

In particular, the chain rule is different under Ito and Stratonovich.

Ito calculus [9]

$$d\Phi(x,t) = \frac{\partial \Phi}{\partial t} dt + \frac{\partial \Phi}{\partial x} dx + \frac{1}{2} \mathsf{Tr}\left[\frac{\partial^2 \Phi}{\partial x^2} g(x,t) g(x,t)^\top \right] dt$$

Stochastic integrals

In particular, the chain rule is different under Ito and Stratonovich.

Ito calculus [9]

$$d\Phi(x,t) = \frac{\partial \Phi}{\partial t} dt + \frac{\partial \Phi}{\partial x} dx + \frac{1}{2} \mathsf{Tr} \left[\frac{\partial^2 \Phi}{\partial x^2} g(x,t) g(x,t)^\top \right] dt$$

Stratonovich calculus [11]

$$\circ d\Phi(x,t) = \frac{\partial \Phi}{\partial t} dt + \frac{\partial \Phi}{\partial x} dx$$

Scalable gradients for nonlinear SDEs [8]

Scalable gradients for nonlinear SDEs [8]

Uses KL between SDEs [1] and Stratonovich calculus [11] to train SDEs.

Scalable gradients for nonlinear SDEs [8]

Uses KL between SDEs [1] and Stratonovich calculus [11] to train SDEs.

Consider prior and approximating SDEs sharing the same noise model

$$dx_t = f_\theta(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(prior SDE. p)}$$

$$dx_t = f_\phi(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(approximating SDE, q)}$$

Scalable gradients for nonlinear SDEs [8]

Uses KL between SDEs [1] and Stratonovich calculus [11] to train SDEs.

Consider prior and approximating SDEs sharing the same noise model

$$dx_t = f_\theta(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(prior SDE. p)}$$

$$dx_t = f_\phi(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(approximating SDE, q)}$$

Train by optimising the ELBO

$$p(y_1, ..., y_n|\theta, \phi) \geq \mathbb{E}_q\left[\sum_{n=1}^{N} p(y_n|x_{t_n}) - \frac{1}{2}\int_0^T \left|\frac{f_\theta(x_\tau, \tau) - f_\phi(x_\tau, \tau)}{g(x_\tau, \tau)}\right|^2 d\tau\right],$$

where $\mathbb{E}_q$ is w.r.t. the approximating SDE.

Scalable gradients for nonlinear SDEs [8]

Uses KL between SDEs [1] and Stratonovich calculus [11] to train SDEs.

Consider prior and approximating SDEs sharing the same noise model

$$dx_t = f_\theta(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(prior SDE. p)}$$

$$dx_t = f_\phi(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(approximating SDE, q)}$$

Train by optimising the ELBO

$$p(y_1, ..., y_n|\theta, \phi) \geq \mathbb{E}_q \left[\sum_{n=1}^{N} p(y_n|x_{t_n}) - \frac{1}{2} \int_0^T \left| \frac{f_\theta(x_\tau, \tau) - f_\phi(x_\tau, \tau)}{g(x_\tau, \tau)} \right|^2 d\tau \right],$$

where $\mathbb{E}_q$ is w.r.t. the approximating SDE.

1. **Forward:** Solve approximating SDE numerically forwards.

Scalable gradients for nonlinear SDEs [8]

Uses KL between SDEs [1] and Stratonovich calculus [11] to train SDEs.

Consider prior and approximating SDEs sharing the same noise model

$$dx_t = f_\theta(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(prior SDE. p)}$$

$$dx_t = f_\phi(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(approximating SDE, q)}$$

Train by optimising the ELBO

$$p(y_1, ..., y_n|\theta, \phi) \geq \mathbb{E}_q \left[\sum_{n=1}^{N} p(y_n|x_{t_n}) - \frac{1}{2} \int_0^T \left| \frac{f_\theta(x_\tau, \tau) - f_\phi(x_\tau, \tau)}{g(x_\tau, \tau)} \right|^2 d\tau \right],$$

where $\mathbb{E}_q$ is w.r.t. the approximating SDE.

1. **Forward:** Solve approximating SDE numerically forwards.
2. **Backward:** Solve an augmented SDE backwards, keeping track of derivatives of objective w.r.t. θ, ϕ.

Scalable gradients for nonlinear SDEs [8]

Uses KL between SDEs [1] and Stratonovich calculus [11] to train SDEs.

Consider prior and approximating SDEs sharing the same noise model

$$dx_t = f_\theta(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(prior SDE. p)}$$

$$dx_t = f_\phi(x_t, t)dt + g(x_t, t)d\beta_t \qquad \text{(approximating SDE, q)}$$

Train by optimising the ELBO

$$p(y_1, ..., y_n|\theta, \phi) \geq \mathbb{E}_q \left[\sum_{n=1}^{N} p(y_n|x_{t_n}) - \frac{1}{2} \int_0^T \left| \frac{f_\theta(x_\tau, \tau) - f_\phi(x_\tau, \tau)}{g(x_\tau, \tau)} \right|^2 d\tau \right],$$

where $\mathbb{E}_q$ is w.r.t. the approximating SDE.

1. **Forward:** Solve approximating SDE numerically forwards.
2. **Backward:** Solve an augmented SDE backwards, keeping track of derivatives of objective w.r.t. θ, ϕ.

Stratonovich yields simplified equations for the dynamics of adjoint SDE.

Scalable gradients for nonlinear SDEs [8]

Scalable gradients for nonlinear SDEs [8]

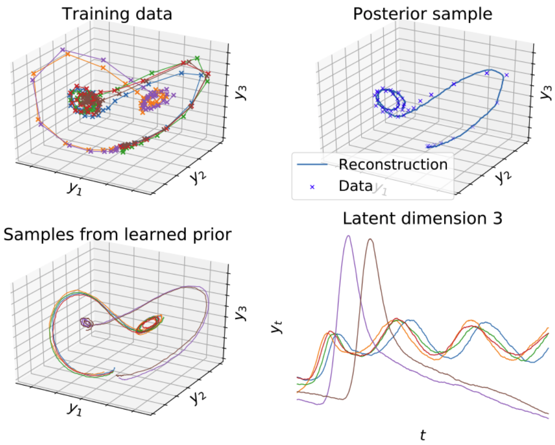


Figure 2: Training data, approximate q, learned p and latent dynamics. [8]

Thank you

Thank you for your attention!

References I

[1] C. Archambeau, M. Opper, Y. Shen, D. Cornford, and J. Shawe-Taylor. Variational inference for diffusion processes. 2008.

[2] D. R. Burt, S. W. Ober, A. Garriga-Alonso, and M. van der Wilk. Understanding variational inference in function-space.

[3] M. Y. Byron, K. V. Shenoy, and M. Sahani. Derivation of kalman filtering and smoothing equations. In *Technical report*. Stanford University, 2004.

[4] E. Daxberger, E. Nalisnick, J. U. Allingham, J. Antorán, and J. M. Hernández-Lobato. Bayesian deep learning via subnetwork inference, 2021.

[5] A. Y. Foong, D. R. Burt, Y. Li, and R. E. Turner. On the expressiveness of approximate inference in bayesian neural networks. *arXiv preprint arXiv:1909.00719*, 2019.

References II

[6] J. Hartikainen and S. Särkkä. Kalman filtering and smoothing solutions to temporal gaussian process regression models. In *2010 IEEE international workshop on machine learning for signal processing*, pages 379–384. IEEE, 2010.

[7] A. Immer, M. Korzepa, and M. Bauer. Improving predictions of bayesian neural networks via local linearization, 2020.

[8] X. Li, T.-K. L. Wong, R. T. Chen, and D. Duvenaud. Scalable gradients for stochastic differential equations. In *International Conference on Artificial Intelligence and Statistics*, pages 3870–3882. PMLR, 2020.

[9] B. Oksendal. *Stochastic differential equations: an introduction with applications*. Springer Science & Business Media, 2013.

[10] S. Särkkä. *Bayesian filtering and smoothing*. Number 3. Cambridge University Press, 2013.

References III

[11] R. Stratonovich. A new representation for stochastic integrals and equations. *SIAM Journal on Control*, 4(2):362–371, 1966.

[12] S. Sun, G. Zhang, J. Shi, and R. Grosse. Functional variational bayesian neural networks, 2019.

[13] M. Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

[14] F. Tobar, T. D. Bui, and R. E. Turner. Learning stationary time series using gaussian processes with nonparametric kernels. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.