Google Brain

Imperial College London

Data Augmentation and Infinitely Wide Neural Networks

Mark van der Wilk

Department of Computing
Imperial College London

@markvanderwilk
m.vdwilk@imperial.ac.uk

Sep 21, 2021

About our research group

- 2020–: Lecturer (Assistant Prof) at Imperial College London.
- Currently growing a research group.
- Research focus:
  - Gaussian process inference, backed by theory to make it reliable.
  - Automatic learning of inductive bias in neural networks.
  - Central question: When should neurons be connected?

PhD Candidates

Artem Artemev

Jose Pablo Folch

Ruby Sedgwick

Seth Nabarro

Tycho van der Ouderaa

Overview

How can Sparse GP approximations help with studying infinitely wide NNs?

Overview

How can Sparse GP approximations help with studying infinitely wide NNs?

Outline:

1. How do Sparse GPs work?

Overview

How can Sparse GP approximations help with studying infinitely wide NNs?

Outline:

1. How do Sparse GPs work?
2. How accurate are Sparse GPs?

Overview

How can Sparse GP approximations help with studying infinitely wide NNs?

Outline:

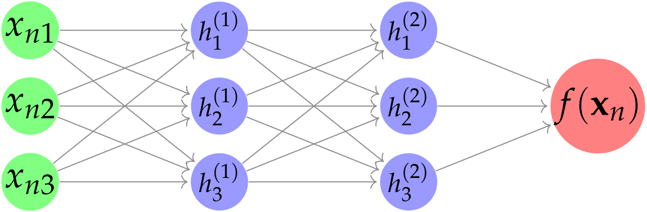
1. How do Sparse GPs work?
2. How accurate are Sparse GPs?
3. Sparse GPs, Data Augmentation and Invariance

Overview

Sparse Gaussian Processes

Data Augmentation and Invariance

Recap: Gaussian Processes & Infinite Width NNs

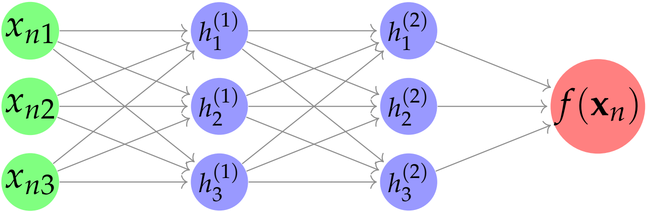


- Place Gaussian prior distribution on weights.
- As number of hidden units → ∞, we have (conditions apply)[1]:
    - Function values $\{f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots\}$ become jointly Gaussian.
    - $\mathrm{Cov}[f(\mathbf{x}), f(\mathbf{x}')] = k(\mathbf{x}, \mathbf{x}')$.
    - Kernel function depends on NN architecture, but can be computed for many![2]

[1] Neal (1996); Matthews et al. (2018); Lee et al. (2018)

[2] Garriga-Alonso et al. (2019); Novak et al. (2019); Yang (2019)

Recap: Gaussian Processes & Infinite Width NNs



- For sets of points $\mathrm{X} \in \mathbb{R}^{N \times D}, \mathrm{Z} \in \mathbb{R}^{M \times D}$, we denote the covariance of their function values as

$$\mathrm{Cov}[f(X), f(Z)] = \mathrm{K}_{\mathrm{XZ}} \in \mathbb{R}^{N \times M}, \qquad [\mathrm{K}_{\mathrm{XZ}}]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j). \tag{1}$$

- Prior on function values for any set of input points is

$$\begin{bmatrix} f(\mathbf{x}_1) \\ \vdots \\ f(\mathbf{x}_n) \end{bmatrix} = f(\mathrm{X}) \sim \mathcal{N}(\boldsymbol{\mu}, \mathrm{K}_{\mathrm{XX}}). \tag{2}$$

Recap: Gaussian Processes

Given observations through some likelihood $p(y_n|f(\mathbf{x}_n))$, find:

1. the distribution of function values at new points $f(\hat{\mathbf{x}})$,
2. the best hyperparameters $\boldsymbol{\theta}$ of the kernel $k_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{x}')$.

Use Bayes' rule (X$'$ includes training and testing points):

$$p(f(\mathrm{X}')|\mathbf{y}) = \frac{\prod_{n=1}^{N} p(y_n|f(\mathbf{x}_n))p(f(\mathrm{X}')|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y})} \tag{3}$$

$$= \underbrace{\frac{p(\mathbf{y}|f(X))p(f(X')|\boldsymbol{\theta})}{p(\mathbf{y}|\boldsymbol{\theta})}}_{p(f(X')|\mathbf{y},\boldsymbol{\theta}))} \cdot \underbrace{\frac{p(\mathbf{y}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{y})}}_{p(\boldsymbol{\theta}|\mathbf{y})} \tag{4}$$

For Gaussian likelihood $p(y_n|f(\mathbf{x}_n)) = \mathcal{N}\left(y_n; f(\mathbf{x}_n), \sigma^2\right)$:

1. $p(f(\hat{\mathbf{x}})|\mathbf{y}, \boldsymbol{\theta}) = \mathcal{N}\left(f(\hat{\mathbf{x}}); \mathrm{K}_{\hat{\mathbf{x}}\mathrm{X}}(\mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I})^{-1}\mathbf{y}, \ldots\right)$
2. $\boldsymbol{\theta}^* = \mathrm{argmax}_{\boldsymbol{\theta}} \log p(\mathbf{y}|\boldsymbol{\theta}) = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I}\right)$

The Problems

1. $p(f(\hat{\mathbf{x}})|\mathbf{y}, \boldsymbol{\theta}) = \mathcal{N}\left(f(\hat{\mathbf{x}}); \mathrm{K}_{\hat{\mathbf{x}}\mathrm{X}}(\mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I})^{-1}\mathbf{y}, \ldots\right)$
2. $\boldsymbol{\theta}^* = \mathrm{argmax}_{\boldsymbol{\theta}} \log p(\mathbf{y}|\boldsymbol{\theta}) = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I}\right)$

Scalability limited by $N \times N$ kernel matrix:

1. Storing $N \times N$ matrix requires $O(N^2)$ memory.
2. Inverting / log determinant takes $O(N^3)$ time.
3. Time for calculating K_{XX} asymptotically scales as $O(N^2)$
   ... but with huge constant, so this is the real bottleneck!

The GP side of my research develops solutions which have guarantees on quality and are automatic.

Solutions: Speed up Linear Algebra

Conjugate Gradient based solutions (Gibbs and Mackay 1997; Wang et al. 2019; Artemev, Burt, and van der Wilk, 2021)

- Speeds up inverse/logdet to $O(N^2 I)$ (how many iterations?)
- Still requires full K_{XX}: $\frac{1}{2}N^2 + N$ 👎

Nyström based solutions (Smola and Schölkopf, 2000; Williams and Seeger, 2001)

- Speeds up inverse/logdet to $O(NM^2)$ (how big is M?)
- Only requires $(N+1)M + \frac{1}{2}M^2$ kernel evals 👍

Nyström Approximation

We want to compute 3 quantities:

1. $c - \frac{1}{2}\log\left|\mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I}\right| - \frac{1}{2}\mathbf{y}^\intercal(\mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I})^{-1}\mathbf{y}$ (marginal likelihood)
2. $\mathrm{K}_{\hat{\mathbf{x}}\mathrm{X}}(\mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I})^{-1}\mathbf{y}$ (pred mean)
3. $\mathrm{K}_{\hat{\mathbf{x}}\hat{\mathbf{x}}} - \mathrm{K}_{\hat{\mathbf{x}}\mathrm{X}}(\mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I})^{-1}\mathrm{K}_{\mathrm{X}\hat{\mathbf{x}}}$ (pred variance)

Straightforward Nyström suggests:

- Select a set Z with $|\mathrm{Z}| = M \ll N$ training points
- Construct the approximation $\mathrm{K}_{\mathrm{XX}} \approx \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}}$
- Use Woodbury for cheap inverse approximation:

$$\begin{aligned}(\mathrm{K}_{\mathrm{XX}} + \sigma^2\mathrm{I})^{-1} &\approx (\mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}} + \sigma^2\mathrm{I})^{-1} \\ &= \sigma^{-2}\mathrm{I} - \sigma^{-4}\mathrm{K}_{\mathrm{XZ}}(\mathrm{K}_{\mathrm{ZZ}} + \sigma^{-2}\mathrm{K}_{\mathrm{ZX}}\mathrm{K}_{\mathrm{XZ}})^{-1}\mathrm{K}_{\mathrm{ZX}}\end{aligned}$$

Nyström and Inducing Variables

- Predicted variances can be negative 👎👎👎
- How good is the approximation?
- When is M large enough?
- How to select the points in Z?

In a single framework, variational inducing variable approximations elegantly gives:

- valid posterior approximations,
- a quantification of the quality of the approximation,
- a way to determine when M is sufficiently large,
- methods for selecting points in Z,

as well as an approximation of K_{XX} based on Nyström.

Variational Inference for Gaussian Processes

Problem: Computational scaling of posterior and marglik.
Three steps of Variational Inference:

1. Introduce a tractable family of variational distributions:
   GP posteriors for arbitrary Gaussian likelihoods $\tilde{q}(\tilde{\mathbf{y}}|f(\mathrm{Z}))$

$$
\begin{aligned}
q(f(\hat{\mathbf{x}}), f(\mathrm{X}), f(\mathrm{Z})) &= \frac{\tilde{q}(\tilde{\mathbf{y}}|f(\mathrm{Z}))p(f(\mathrm{Z}), f(\hat{\mathbf{x}}), f(\mathrm{X}))}{\tilde{q}(\tilde{\mathbf{y}})} && (5) \\
&= p(f(\hat{\mathbf{x}}), f(\mathrm{X})|f(\mathrm{Z}))q(\mathrm{Z}) && (6)
\end{aligned}
$$

2. Construct $\mathcal{L}$ such that[3] $\mathcal{L} = \log p(\mathbf{y}) - \mathrm{KL}[q(f)||p(f|\mathbf{y})]$

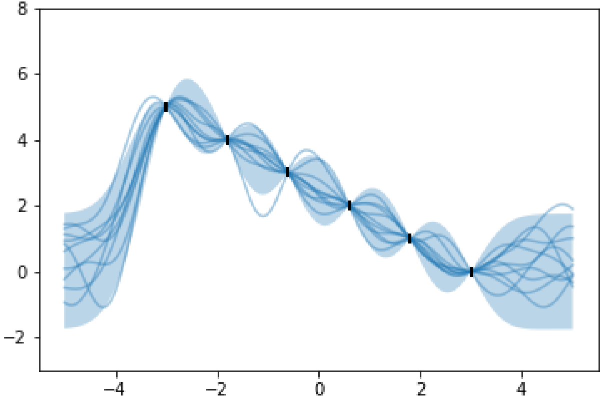
$$
\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(y_n|f(\mathbf{x}_n))] - \mathrm{KL}[q(f(\mathrm{Z}))||p(f(\mathrm{Z}))] \quad (7)
$$

3. Minimise KL divergence by maximising $\mathcal{L}$!

[3] Hensman et al. (2013); Matthews (2016)

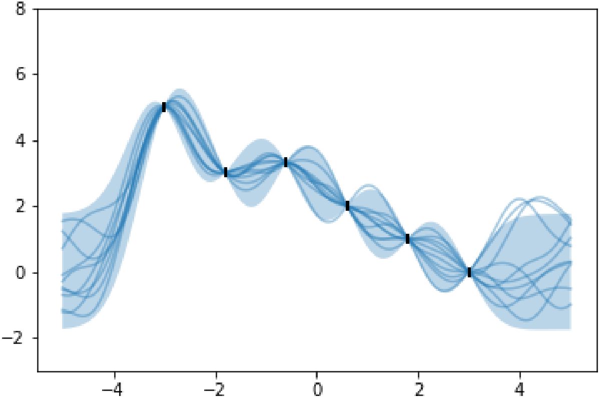
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



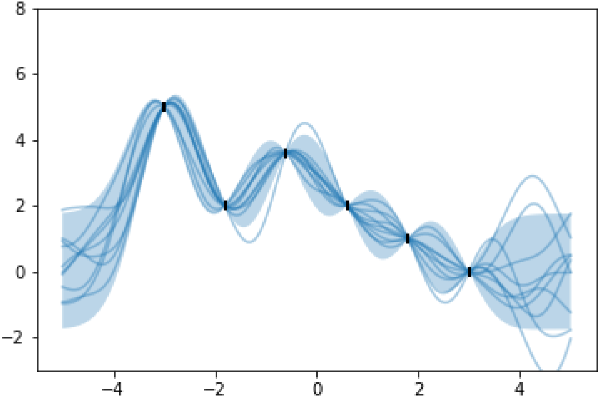
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



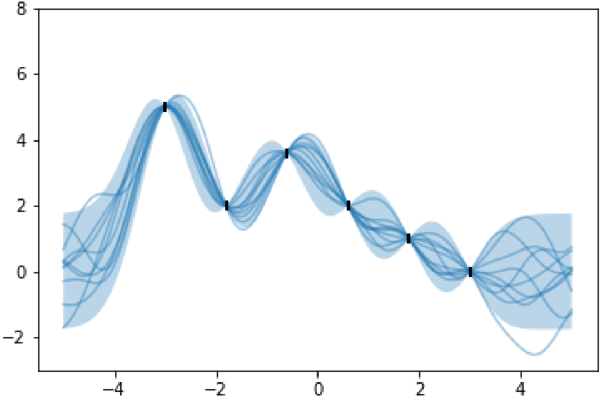
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



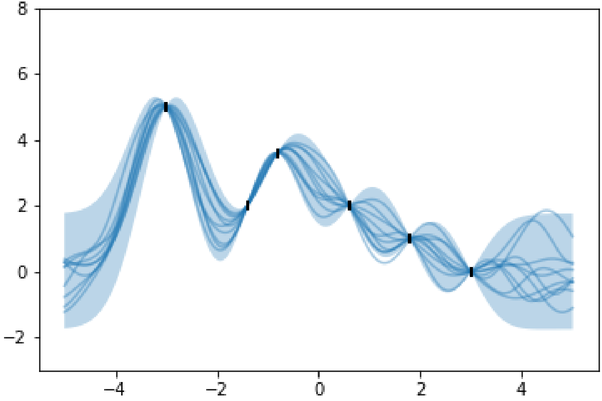
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



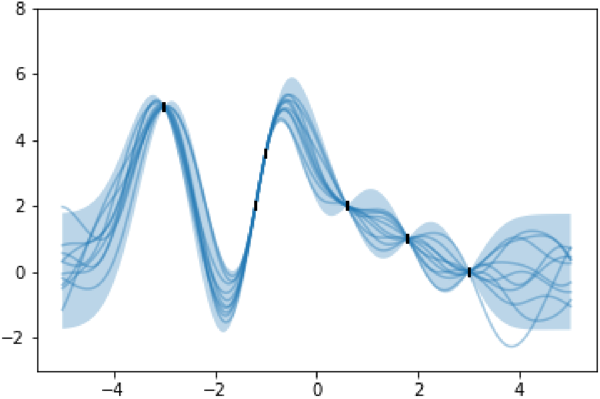
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



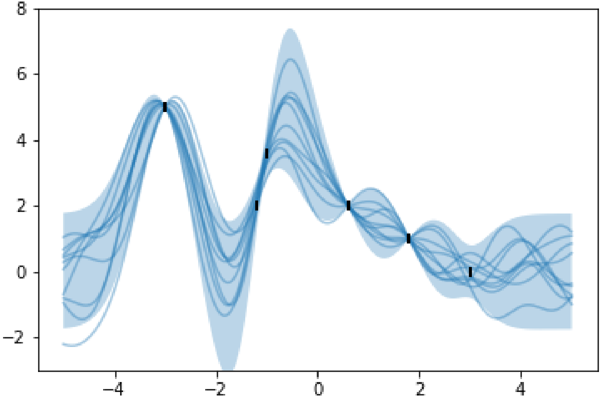
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



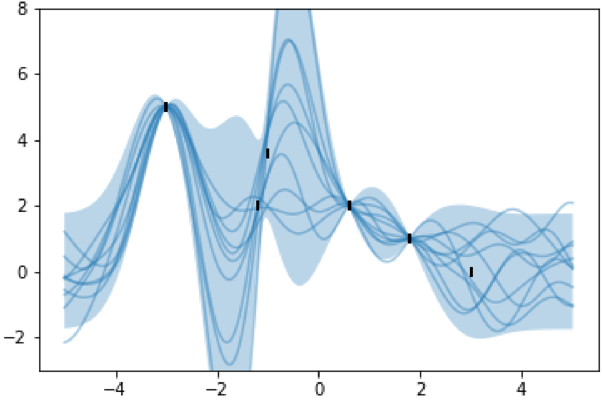
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



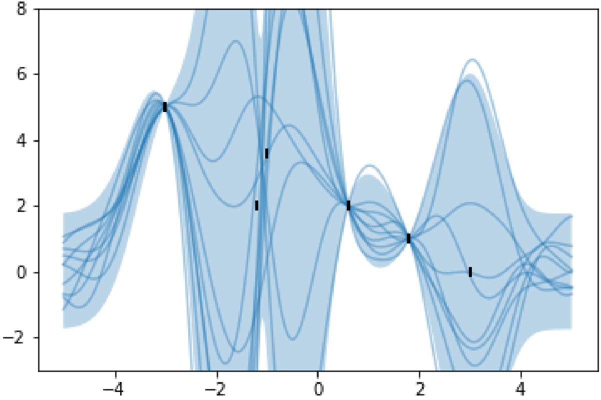
Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



Understanding Inducing Points

We can control Z, and μ, Σ of $q(f(Z))$.



Sparse GP Regression

Gaussian noise regression works particularly well, since the optimal $\boldsymbol{\mu}, \Sigma$ can be found in closed form[4], giving

$$\mathcal{L} = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}} + \sigma^2 \mathrm{I}\right) - \frac{1}{2\sigma^2}\mathrm{Tr}(\mathrm{K}_{\mathrm{XX}} - \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}}) \quad (8)$$

The ELBO helps us select every free paramter of the method!

- Q: How to select hyperparameters?

[4] Titsias (2009)

Sparse GP Regression

Gaussian noise regression works particularly well, since the optimal $\boldsymbol{\mu}, \Sigma$ can be found in closed form[4], giving

$$\mathcal{L} = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}} + \sigma^2\mathrm{I}\right) - \frac{1}{2\sigma^2}\operatorname{Tr}(\mathrm{K}_{\mathrm{XX}} - \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}}) \quad (8)$$

The ELBO helps us select every free paramter of the method!

- Q: How to select hyperparameters?
  A: Maximise $\mathcal{L}$. No overfitting, since it's a lower bound.

[4] Titsias (2009)

Sparse GP Regression

Gaussian noise regression works particularly well, since the optimal $\boldsymbol{\mu}, \Sigma$ can be found in closed form[4], giving

$$\mathcal{L} = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}} + \sigma^2\mathrm{I}\right) - \frac{1}{2\sigma^2}\mathrm{Tr}(\mathrm{K}_{\mathrm{XX}} - \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}}) \tag{8}$$

The ELBO helps us select every free paramter of the method!

- Q: How to select hyperparameters?
  A: Maximise $\mathcal{L}$. No overfitting, since it's a lower bound.
- Q: How to select inducing inputs Z?

[4] Titsias (2009)

Sparse GP Regression

Gaussian noise regression works particularly well, since the optimal $\boldsymbol{\mu}, \Sigma$ can be found in closed form[4], giving

$$\mathcal{L} = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K_{XZ}} \mathrm{K_{ZZ}^{-1}} \mathrm{K_{ZX}} + \sigma^2 \mathrm{I}\right) - \frac{1}{2\sigma^2} \mathrm{Tr}(\mathrm{K_{XX}} - \mathrm{K_{XZ}} \mathrm{K_{ZZ}^{-1}} \mathrm{K_{ZX}}) \quad (8)$$

The ELBO helps us select every free paramter of the method!

- Q: How to select hyperparameters?
  A: Maximise $\mathcal{L}$. No overfitting, since it's a lower bound.
- Q: How to select inducing inputs Z?
  A: Maximise $\mathcal{L}$ only reduces KL to true posterior.

[4] Titsias (2009)

Sparse GP Regression

Gaussian noise regression works particularly well, since the optimal $\boldsymbol{\mu}, \Sigma$ can be found in closed form[4], giving

$$\mathcal{L} = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K_{XZ}} \mathrm{K_{ZZ}^{-1}} \mathrm{K_{ZX}} + \sigma^2 \mathrm{I}\right) - \frac{1}{2\sigma^2} \mathrm{Tr}(\mathrm{K_{XX}} - \mathrm{K_{XZ}} \mathrm{K_{ZZ}^{-1}} \mathrm{K_{ZX}}) \quad (8)$$

The ELBO helps us select every free paramter of the method!

- Q: How to select hyperparameters?
  A: Maximise $\mathcal{L}$. No overfitting, since it's a lower bound.
- Q: How to select inducing inputs Z?
  A: Maximise $\mathcal{L}$ only reduces KL to true posterior.
- Q: When do we have enough inducing points?

[4] Titsias (2009)

Sparse GP Regression

Gaussian noise regression works particularly well, since the optimal $\boldsymbol{\mu}, \Sigma$ can be found in closed form[4], giving

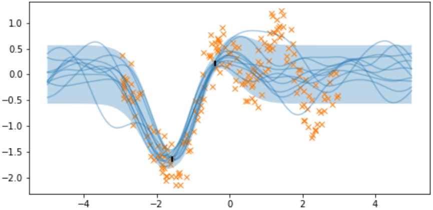
$$\mathcal{L} = \log \mathcal{N}\left(\mathbf{y}; 0, \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}} + \sigma^2 \mathrm{I}\right) - \frac{1}{2\sigma^2}\mathrm{Tr}(\mathrm{K}_{\mathrm{XX}} - \mathrm{K}_{\mathrm{XZ}}\mathrm{K}_{\mathrm{ZZ}}^{-1}\mathrm{K}_{\mathrm{ZX}}) \quad (8)$$

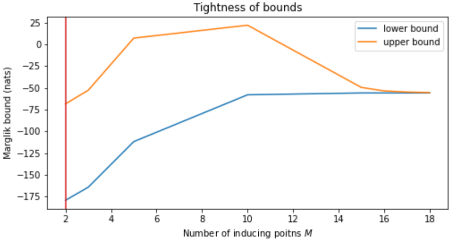
The ELBO helps us select every free paramter of the method!

- Q: How to select hyperparameters?
  A: Maximise $\mathcal{L}$. No overfitting, since it's a lower bound.
- Q: How to select inducing inputs Z?
  A: Maximise $\mathcal{L}$ only reduces KL to true posterior.
- Q: When do we have enough inducing points?
  A: Once $\mathcal{L}$ stops increasing (we also have upper bound).

[4] Titsias (2009)

Demo



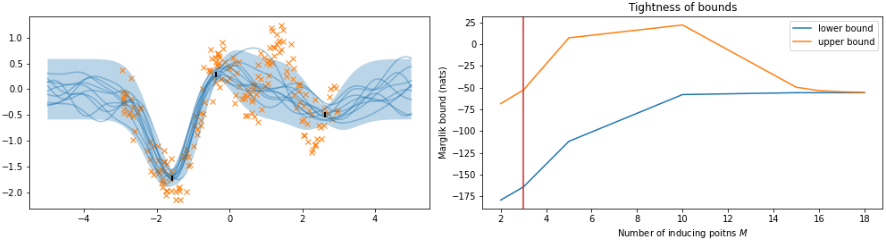


We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Demo

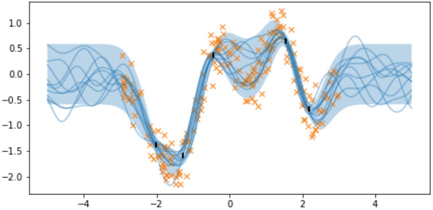


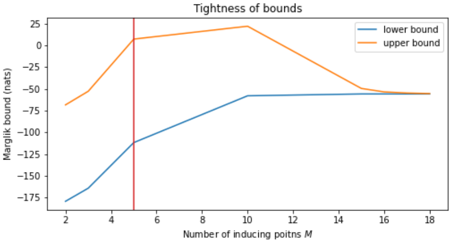
We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Demo



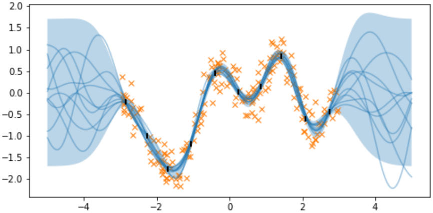


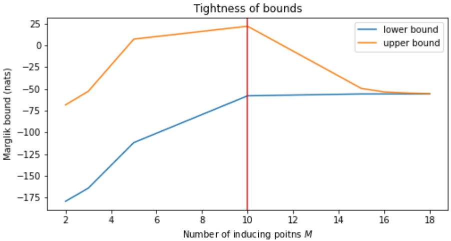
We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Demo



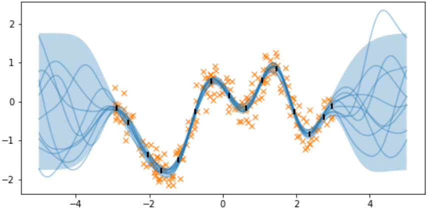


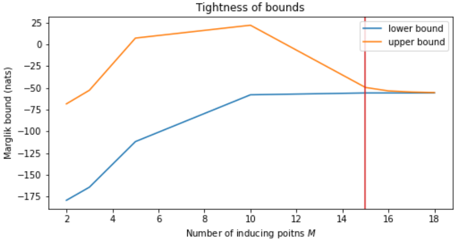
We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Demo



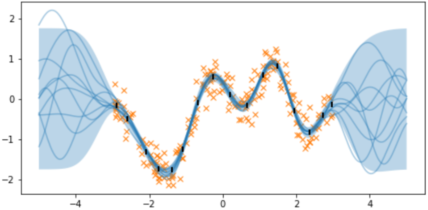


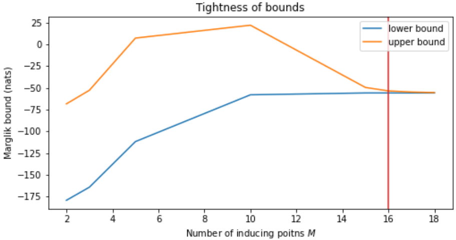
We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Demo



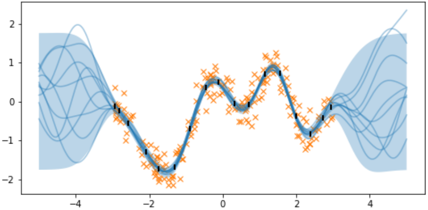


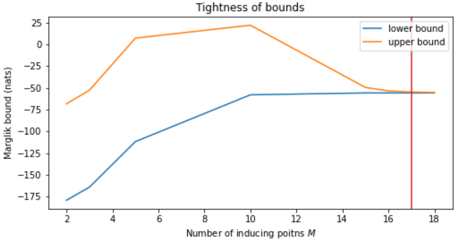
We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Demo



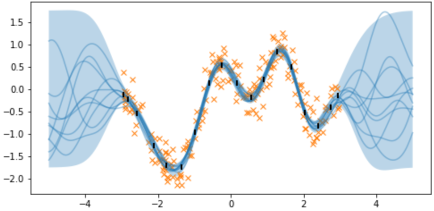


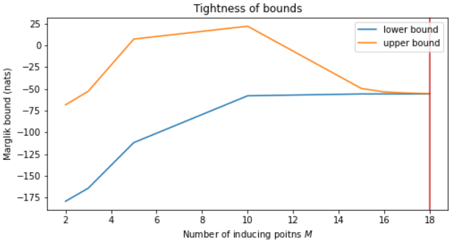
We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Demo





We jointly optimise $\mathcal{L}$ w.r.t. its two free parameters: $Z, \boldsymbol{\theta}$.

- Approximation and fit are poor when M is too small.
- ELBO convergences with $M \ll N$.
- Upper bound[5] converges later to confirm good quality.

[5] See Titsias (2014), or Burt et al. (2020) for a discussion.

Theory Gives Solutions

In “Convergence of Sparse Variational Inference in Gaussian Processes Regression” (Burt et al., 2020) we

- discussed a gradient-free inducing point initialisation scheme
- proved that it would give arbitrarily accurate results as $N \to \infty$
- proved that the asymptotic complexity was reasonable $O(N(\log N)^{2D}(\log\log N)^2))$ for SqExp, barely above linear[6]

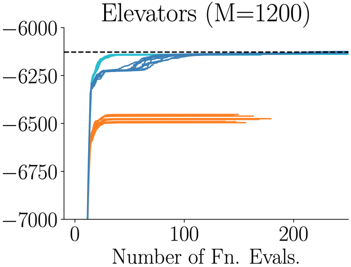
[6] Recall that $O(N(\log N)^D) = O(N^{1+\epsilon})$ for any $D \in \mathbb{N}$ and $\epsilon > 0$, and that D is fixed in our problem.

Theory Gives Solutions

In “Convergence of Sparse Variational Inference in Gaussian Processes Regression” (Burt et al., 2020) we

- discussed a gradient-free inducing point initialisation scheme
- proved that it would give arbitrarily accurate results as $N \to \infty$
- proved that the asymptotic complexity was reasonable $O(N(\log N)^{2D}(\log\log N)^2))$ for SqExp, barely above linear[6]

Practical implications for the Titsias (2009) method:



[6] Recall that $O(N(\log N)^D) = O(N^{1+\epsilon})$ for any $D \in \mathbb{N}$ and $\epsilon > 0$, and that D is fixed in our problem.

Sparse GPs: Summary

Recipe for Sparse Variational GP Regression[7]:

1. Select initial number of inducing points M to try.
2. Select $\mathbf{Z}$ with the greedy variance method (Burt et al., 2020).
3. Optional: Optimise $\mathcal{L}$ w.r.t. $\boldsymbol{\theta}$.
4. Stop if upper-lower gap is small, or if improvement in $\mathcal{L}$ is small. Otherwise repeat from step 2.

[7] as recommended in Burt, Rasmussen, and van der Wilk (2020)

Sparse GPs: Conclusion

Sparse Gaussian Process approximations provide a unified way to approximate GPs:

- Correct and consistent posterior approximations.
- Single objective function can be used for setting all parameters.
- Measurable quality of approximation.
- For certain kernels, guarantees of good and cheap approximation as $N \to \infty$ (+conditions).

Sparse GPs: Conclusion

Sparse Gaussian Process approximations provide a unified way to approximate GPs:

- Correct and consistent posterior approximations.
- Single objective function can be used for setting all parameters.
- Measurable quality of approximation.
- For certain kernels, guarantees of good and cheap approximation as $N \to \infty$ (+conditions).
- Burt et al. (2020) hints at a link between generalisation and approximation sparsity/quality. This would be very interesting to investigate in the context of infinite NN kernels.

Overview

Sparse Gaussian Processes

Data Augmentation and Invariance

Modelling Assumptions

Goal: Learn some mapping $f : \mathcal{X} \to \mathcal{Y}$.

Assumptions about f influence generalisation performance:

- Fully connected vs convolutional?
- How smooth is the function?
- Data augmentation? I.e. what transformations leave the label unchanged?

Modelling Assumptions

Goal: Learn some mapping $f : \mathcal{X} \rightarrow \mathcal{Y}$.

Assumptions about f influence generalisation performance:

- Fully connected vs convolutional?
- How smooth is the function?
- Data augmentation? I.e. what transformations leave the label unchanged?

Central question:

How can changes on the input affect the output?

Modelling Assumptions

Goal: Learn some mapping $f : \mathcal{X} \to \mathcal{Y}$.

Assumptions about f influence generalisation performance:

- Fully connected vs convolutional?
- How smooth is the function?
- Data augmentation? I.e. what transformations leave the label unchanged?

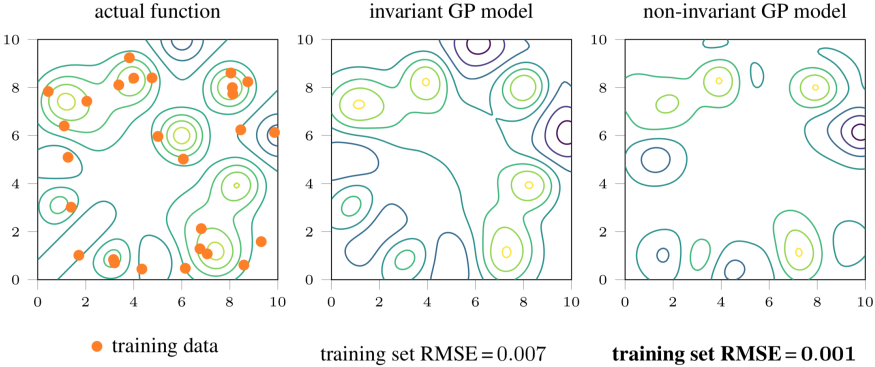
Central question:

How can changes on the input affect the output?

The fewer unnecessary degrees of variation we have, the better we will generalise.

⏩ Goal: Find the right degrees of freedom as well as learning f

Example: Symmetry



training set RMSE = 0.007
test set RMSE = 0.43

training set RMSE = 0.001
test set RMSE = 0.87

- Learn symmetric function
- Pick either symmetrically constrained model or flexible model
- Symmetric model generalises better

Data Augmentation and Invariances

Data augmentations express the knowledge about $f(\cdot)$ that the output doesn't change in response to changes in the input. This is invariance.

Data Augmentation and Invariances

Data augmentations express the knowledge about $f(\cdot)$ that the output doesn't change in response to changes in the input. This is invariance.

We can consider strict invariances:

$$f(\mathbf{x}) = f(t(\mathbf{x})) \qquad \forall \mathbf{x} \in \mathcal{X} \qquad \forall t \in \mathcal{T} \tag{9}$$

Data Augmentation and Invariances

Data augmentations express the knowledge about $f(\cdot)$ that the output doesn't change in response to changes in the input. This is invariance.

We can consider strict invariances:

$$f(\mathbf{x}) = f(t(\mathbf{x})) \qquad \forall \mathbf{x} \in \mathcal{X} \qquad \forall t \in \mathcal{T} \tag{9}$$

or softer invariance:

$$P\Big((\ f(\mathbf{x}) - f(t(\mathbf{x}))\ )^2 > L \Big) < \epsilon \qquad \forall \mathbf{x} \in \mathcal{X} \qquad t \sim p(t) \tag{10}$$

Data Augmentation and Invariance

Questions:

- How should we incorporate knowledge of invariances/DA into our models? (Particularly in the Bayesian context!)

[8] https://statmodeling.stat.columbia.edu/2019/12/02/a-bayesian-view-of-data-augmentation/

Data Augmentation and Invariance

Questions:

- How should we incorporate knowledge of invariances/DA into our models? (Particularly in the Bayesian context!)
- How can we select the right invariance/DA if we do not know it a priori?

[8] https://statmodeling.stat.columbia.edu/2019/12/02/a-bayesian-view-of-data-augmentation/

Data Augmentation and Invariance

Questions:

- How should we incorporate knowledge of invariances/DA into our models? (Particularly in the Bayesian context!)
- How can we select the right invariance/DA if we do not know it a priori?

[8] `https://statmodeling.stat.columbia.edu/2019/12/02/a-bayesian-view-of-data-augmentation/`

Data Augmentation and Invariance

Questions:

- How should we incorporate knowledge of invariances/DA into our models? (Particularly in the Bayesian context!)
- How can we select the right invariance/DA if we do not know it a priori?

[8] https://statmodeling.stat.columbia.edu/2019/12/02/a-bayesian-view-of-data-augmentation/

Data Augmentation and Invariance

Questions:

- How should we incorporate knowledge of invariances/DA into our models? (Particularly in the Bayesian context!)
- How can we select the right invariance/DA if we do not know it a priori?

In “Learning Invariances using the Marginal Likelihood” (v.d.Wilk et al., 2018) we

- Provide a clear formulation of how this can be done in a Bayesian context.[8]
- Provide a practical procedure for learning invariances using gradient descent in GPs.

[8] `https://statmodeling.stat.columbia.edu/2019/12/02/a-bayesian-view-of-data-augmentation/`

Model Selection according to Bayes

Model selection from a Bayesian point of view:

$$\begin{aligned} p(f, \boldsymbol{\theta} \mid \mathbf{y}) &= \frac{p(\mathbf{y} \mid f) p(f \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y})} \\ &= \underbrace{\frac{p(\mathbf{y} \mid f) p(f \mid \boldsymbol{\theta})}{p(\mathbf{y} \mid \boldsymbol{\theta})}}_{p(f \mid \mathbf{y}, \boldsymbol{\theta})} \underbrace{\frac{p(\mathbf{y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y})}}_{p(\boldsymbol{\theta} \mid \mathbf{y})} \end{aligned}$$

Model Selection according to Bayes

Model selection from a Bayesian point of view:

$$
\begin{aligned}
p(f, \boldsymbol{\theta} \,|\, \mathbf{y}) &= \frac{p(\mathbf{y} \,|\, f) p(f \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y})} \\
&= \underbrace{\frac{p(\mathbf{y} \,|\, f) p(f \,|\, \boldsymbol{\theta})}{p(\mathbf{y} \,|\, \boldsymbol{\theta})}}_{p(f \,|\, \mathbf{y}, \boldsymbol{\theta})} \underbrace{\frac{p(\mathbf{y} \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y})}}_{p(\boldsymbol{\theta} \,|\, \mathbf{y})}
\end{aligned}
$$

Key quantity for model selection is the marginal likelihood

$$
p(\mathbf{y} \,|\, \boldsymbol{\theta}) = \int p(\mathbf{y} \,|\, f) p(f \,|\, \boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}
$$

Model Selection according to Bayes

Model selection from a Bayesian point of view:

$$
\begin{aligned}
p(f, \boldsymbol{\theta} \,|\, \mathbf{y}) &= \frac{p(\mathbf{y} \,|\, f) p(f \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y})} \\
&= \underbrace{\frac{p(\mathbf{y} \,|\, f) p(f \,|\, \boldsymbol{\theta})}{p(\mathbf{y} \,|\, \boldsymbol{\theta})}}_{p(f \,|\, \mathbf{y}, \boldsymbol{\theta})} \underbrace{\frac{p(\mathbf{y} \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\mathbf{y})}}_{p(\boldsymbol{\theta} \,|\, \mathbf{y})}
\end{aligned}
$$

Key quantity for model selection is the marginal likelihood

$$
p(\mathbf{y} \,|\, \boldsymbol{\theta}) = \int p(\mathbf{y} \,|\, f) p(f \,|\, \boldsymbol{\theta}) \mathrm{d}\boldsymbol{\theta}
$$

By handing our uncertainty on $f(\cdot)$ in a Bayesian way, we also get the marginal likelihood for model selection.

Model Selection: Procedure

Our desired simplified procedure:

- Place prior on f with invariances described by $\boldsymbol{\theta}$.
- Find posterior over functions $p(f \mid \mathbf{y}, \boldsymbol{\theta})$.
- Perform Maximum Likelihood on $p(\mathbf{y} \mid \boldsymbol{\theta})$.

Model Selection: Procedure

Our desired simplified procedure:

- Place prior on f with invariances described by $\boldsymbol{\theta}$.
- Find posterior over functions $p(f \,|\, \mathbf{y}, \boldsymbol{\theta})$.
- Perform Maximum Likelihood on $p(\mathbf{y} \,|\, \boldsymbol{\theta})$.

This is more safe from over-fitting that performing Maximum Likelihood on $f, \boldsymbol{\theta}$ together.

Model Selection: Procedure

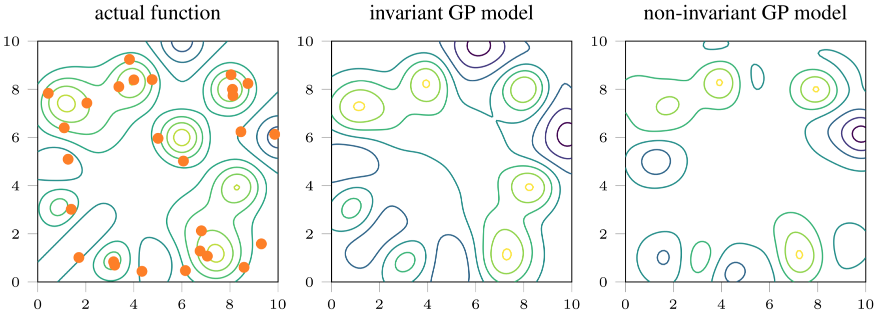
Our desired simplified procedure:

- Place prior on f with invariances described by $\boldsymbol{\theta}$.
- Find posterior over functions $p(f \,|\, \mathbf{y}, \boldsymbol{\theta})$.
- Perform Maximum Likelihood on $p(\mathbf{y} \,|\, \boldsymbol{\theta})$.

This is more safe from over-fitting that performing Maximum Likelihood on $f, \boldsymbol{\theta}$ together.

(If you're sceptical, ask me for an example at the end.)

Marginal Likelihood



log marginal lik. = −34.906
training set RMSE = 0.007
test set RMSE = 0.43

log marginal lik. = −43.768
training set RMSE = 0.001
test set RMSE = 0.87

$$\log p(\mathbf{y} \mid \boldsymbol{\theta}) = \log p(y_1 \mid \boldsymbol{\theta}) + \log p(y_2 \mid y_1, \boldsymbol{\theta}) + \log p(y_3 \mid \{y_i\}_{i=1}^{2}, \boldsymbol{\theta}) \ldots$$

$$= \sum_{n=1}^{N} \log p(y_n \mid \{y_i\}_{i=1}^{n-1}, \boldsymbol{\theta})$$

Practicalities

Procedure so far is completely general and abstract. We need to:

- Choose our model class through the prior $p(f \mid \boldsymbol{\theta})$
- Parameterise invariances in the prior through $\boldsymbol{\theta}$
- Show how to calculate $p(f \mid \mathbf{y}, \boldsymbol{\theta})$ and $p(\mathbf{y} \mid \boldsymbol{\theta})$

Practicalities

Procedure so far is completely general and abstract. We need to:

- Choose our model class through the prior $p(f \mid \boldsymbol{\theta})$
- Parameterise invariances in the prior through $\boldsymbol{\theta}$
- Show how to calculate $p(f \mid \mathbf{y}, \boldsymbol{\theta})$ and $p(\mathbf{y} \mid \boldsymbol{\theta})$

We choose

- Gaussian process priors $p(f \mid \boldsymbol{\theta})$
- A construction of invariant GPs following Kondor (2008) and Ginsbourger et al. (2012)
- Variational approximation for posterior and marginal likelihood (Titsias 2009; Hensman et al. 2013)

Invariant Gaussian Processes

Easy to place Gaussian process priors on non-invariant functions:

$$g(\cdot) \sim \mathcal{GP}\big(0, k_g(\cdot, \cdot')\big)\,, \quad g : \mathbb{R}^D \to \mathbb{R}\,, \quad k_g : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}\,. \tag{11}$$

Invariant Gaussian Processes

Easy to place Gaussian process priors on non-invariant functions:

$$g(\cdot) \sim \mathcal{GP}\big(0, k_g(\cdot, \cdot'))\big), \quad g : \mathbb{R}^D \to \mathbb{R}, \quad k_g : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}. \tag{11}$$

Can construct an invariant $f(\cdot)$ by summing over the orbit of the group of transformations we want to be invariant to.

$$f(\cdot) = \sum_{\mathbf{x}_a \in \mathcal{O}(\mathbf{x})} g(\mathbf{x}_a) \tag{12}$$

Invariant Gaussian Processes

Easy to place Gaussian process priors on non-invariant functions:

$$g(\cdot) \sim \mathcal{GP}\big(0, k_g(\cdot, \cdot'))\big)\,, \quad g : \mathbb{R}^D \to \mathbb{R}\,, \quad k_g : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}\,. \tag{11}$$

Can construct an invariant $f(\cdot)$ by summing over the orbit of the group of transformations we want to be invariant to.

$$f(\cdot) = \sum_{\mathbf{x}_a \in \mathcal{O}(\mathbf{x})} g(\mathbf{x}_a) \tag{12}$$

Example: $\mathcal{T}$ group of all rotations

⏩ Orbit of image is set of images rotated by all angles

Insensitivity

Parameterising orbits is hard, so we relax strict invariance constraint, and sum over arbitrary sets $\mathcal{A}(\mathbf{x})$

$$f(\cdot) = \sum_{\mathbf{x}_a \in \mathcal{A}(\mathbf{x})} g(\mathbf{x}_a) \tag{13}$$

Insensitivity

Parameterising orbits is hard, so we relax strict invariance constraint, and sum over arbitrary sets $\mathcal{A}(\mathbf{x})$

$$f(\cdot) = \sum_{\mathbf{x}_a \in \mathcal{A}(\mathbf{x})} g(\mathbf{x}_a) \tag{13}$$

Parameterising sets is also cumbersome, so we take the infinite limit, to get an expectation

$$f(\cdot) = \int g(\mathbf{x}_a) p(\mathbf{x}_a \mid \mathbf{x}) \mathrm{d}\mathbf{x}_a \tag{14}$$

Insensitivity

Parameterising orbits is hard, so we relax strict invariance constraint, and sum over arbitrary sets $\mathcal{A}(\mathbf{x})$

$$f(\cdot) = \sum_{\mathbf{x}_a \in \mathcal{A}(\mathbf{x})} g(\mathbf{x}_a) \tag{13}$$

Parameterising sets is also cumbersome, so we take the infinite limit, to get an expectation

$$f(\cdot) = \int g(\mathbf{x}_a) p(\mathbf{x}_a \,|\, \mathbf{x}) \mathrm{d}\mathbf{x}_a \tag{14}$$

- No longer strictly invariant
- Instead (roughly) a limit on $P\Big((f(\mathbf{x}_a) - f(\mathbf{x}))^2 > L\Big)$
- Can interpolate between non-invariant and invariant

Insensitive Kernels

The summation construction implies a kernel over $f(\cdot)$:

$$
\begin{aligned}
g(\cdot) &\sim \mathcal{GP}(0, k_g(\cdot, \cdot')) \\
f(\cdot) &\sim \mathcal{GP}(0, k_f(\cdot, \cdot')) && \text{(by linearity)} \\
k_f(\mathbf{x}, \mathbf{x}') &= \mathbb{E}_g\left[f(\mathbf{x})f(\mathbf{x}')\right] \\
&= \mathbb{E}_g\left[\left(\int g(\mathbf{x}_a)p(\mathbf{x}_a \,|\, \mathbf{x})\mathrm{d}\mathbf{x}_a\right)\left(\int g(\mathbf{x}_a')p(\mathbf{x}_a' \,|\, \mathbf{x}')\mathrm{d}\mathbf{x}_a'\right)\right] \\
&= \iint \mathbb{E}_g\left[g(\mathbf{x}_a)g(\mathbf{x}_a')\right]p(\mathbf{x}_a \,|\, \mathbf{x})p(\mathbf{x}_a' \,|\, \mathbf{x}')\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}_a' \\
&= \iint k_g(\mathbf{x}_a, \mathbf{x}_a')p(\mathbf{x}_a \,|\, \mathbf{x})p(\mathbf{x}_a' \,|\, \mathbf{x}')\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}_a'
\end{aligned}
$$

Insensitive Kernels

The summation construction implies a kernel over $f(\cdot)$:

$$\begin{aligned}
g(\cdot) &\sim \mathcal{GP}(0, k_g(\cdot, \cdot')) \\
f(\cdot) &\sim \mathcal{GP}(0, k_f(\cdot, \cdot')) && \text{(by linearity)} \\
k_f(\mathbf{x}, \mathbf{x}') &= \mathbb{E}_g\left[f(\mathbf{x})f(\mathbf{x}')\right] \\
&= \mathbb{E}_g\left[\left(\int g(\mathbf{x}_a)p(\mathbf{x}_a \,|\, \mathbf{x})\mathrm{d}\mathbf{x}_a\right)\left(\int g(\mathbf{x}'_a)p(\mathbf{x}'_a \,|\, \mathbf{x}')\mathrm{d}\mathbf{x}'_a\right)\right] \\
&= \iint \mathbb{E}_g\left[g(\mathbf{x}_a)g(\mathbf{x}'_a)\right]p(\mathbf{x}_a \,|\, \mathbf{x})p(\mathbf{x}'_a \,|\, \mathbf{x}')\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}'_a \\
&= \iint k_g(\mathbf{x}_a, \mathbf{x}'_a)p(\mathbf{x}_a \,|\, \mathbf{x})p(\mathbf{x}'_a \,|\, \mathbf{x}')\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}'_a
\end{aligned}$$

Parameterise insensitivity by parameterising $p_{\boldsymbol{\theta}}(\mathbf{x}_a \,|\, \mathbf{x})$!

Interpolating to strict invariance

$f(\mathbf{x})$ $g(\mathbf{x}_a),\ p_{\boldsymbol{\theta}}(\mathbf{x}_a \mid \mathbf{x})$

Overview

We have introduced GP priors with invariance properties controlled by $p_{\boldsymbol{\theta}}(\mathbf{x}_a \mid \mathbf{x})$.

Now we must compute

- The marginal likelihood $p(\boldsymbol{\theta} \mid \mathbf{y})$ and its gradients for selecting the invariance
- The posterior $p(f \mid \mathbf{y}, \boldsymbol{\theta})$ to make predictions

Computational difficulties

Approximations are necessary in Gaussian process models for the well-known reasons:

- Kernel inversions cost $O(N^3)$
- Non-conjugate likelihoods (classification)
- No minibatch training

Computational difficulties

Approximations are necessary in Gaussian process models for the well-known reasons:

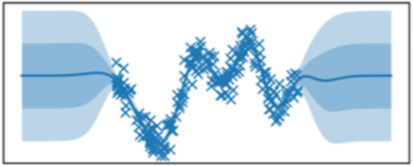
- Kernel inversions cost $O(N^3)$
- Non-conjugate likelihoods (classification)
- No minibatch training

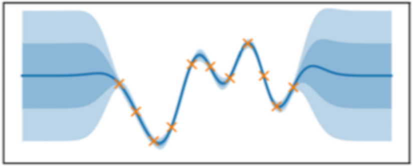
Here we have an additional problem:

We can't even evaluate the kernel!

Variational inference

Approximate posterior: Prior conditioned on $M \ll N$ noisy observations $f(Z) = \{f(\mathbf{z}_m)\}_{m=1}^{M}$:



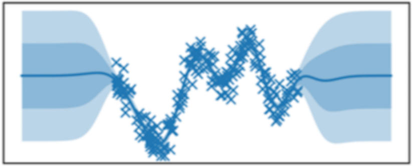


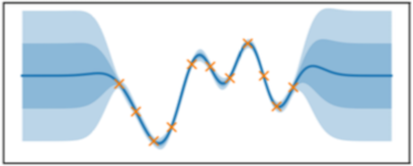
$$q(f(\mathbf{x})) = \mathcal{N}\left(f(\mathbf{x}); \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}\mathbf{m}, k_f(\mathbf{x},\mathbf{x}) - \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}(\boldsymbol{K}_{ZZ} - \boldsymbol{S})\boldsymbol{K}_{ZZ}^{-1}\mathbf{k}_{Z\mathbf{x}}\right)$$

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(\mathbf{y}_n \mid f(\mathbf{x}_n))] - \mathrm{KL}[q(f(Z))||p(f(Z))]$$

Variational inference

Approximate posterior: Prior conditioned on $M \ll N$ noisy observations $f(Z) = \{f(\mathbf{z}_m)\}_{m=1}^{M}$:





$$q(f(\mathbf{x})) = \mathcal{N}\Big(f(\mathbf{x}); \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}\mathbf{m}, k_f(\mathbf{x},\mathbf{x}) - \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}(\boldsymbol{K}_{ZZ} - \boldsymbol{S})\boldsymbol{K}_{ZZ}^{-1}\mathbf{k}_{Z\mathbf{x}}\Big)$$

$$\mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(\mathbf{y}_n \mid f(\mathbf{x}_n))] - \mathrm{KL}[q(f(Z))||p(f(Z))]$$

Gives: Approximate posterior , lower bound to marginal likelihood
Solves: inversion cost , non-conjugate likelihoods , minibatching

Variational Inference

For Gaussian likelihoods

$$\mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(\mathbf{y}_n \mid f(\mathbf{x}_n))] = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}(y_n - \mu_n)^2 - \frac{\sigma_n^2}{2\sigma^2}$$

$$q(f(\mathbf{x})) = \mathcal{N}(f(\mathbf{x}); \underbrace{\mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}\mathbf{m}}_{\mu_n}, \underbrace{k_f(\mathbf{x}_n, \mathbf{x}_n) - \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}(\boldsymbol{K}_{ZZ} - \boldsymbol{S})\boldsymbol{K}_{ZZ}^{-1}\mathbf{k}_{Z\mathbf{x}}}_{\sigma_n^2})$$

With

$$[\mathbf{k}_{Z\mathbf{x}}]_{mn} = \iint k_g(\mathbf{x}_a, \mathbf{x}'_a)p(\mathbf{x}_a \mid \mathbf{z}_m)p(\mathbf{x}'_a \mid \mathbf{z}'_m)\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}'_a$$

$$[\boldsymbol{K}_{ZZ}]_{mm} = \iint k_g(\mathbf{x}_a, \mathbf{x}'_a)p(\mathbf{x}_a \mid \mathbf{z}_m)p(\mathbf{x}'_a \mid \mathbf{z}'_m)\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}'_a$$

Variational Inference

For Gaussian likelihoods

$$\mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(\mathbf{y}_n \mid f(\mathbf{x}_n))] = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}(y_n - \mu_n)^2 - \frac{\sigma_n^2}{2\sigma^2}$$

$$q(f(\mathbf{x})) = \mathcal{N}(f(\mathbf{x}); \underbrace{\mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}\mathbf{m}}_{\mu_n}, \underbrace{k_f(\mathbf{x}_n, \mathbf{x}_n) - \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}(\boldsymbol{K}_{ZZ} - \boldsymbol{S})\boldsymbol{K}_{ZZ}^{-1}\mathbf{k}_{Z\mathbf{x}}}_{\sigma_n^2})$$

With

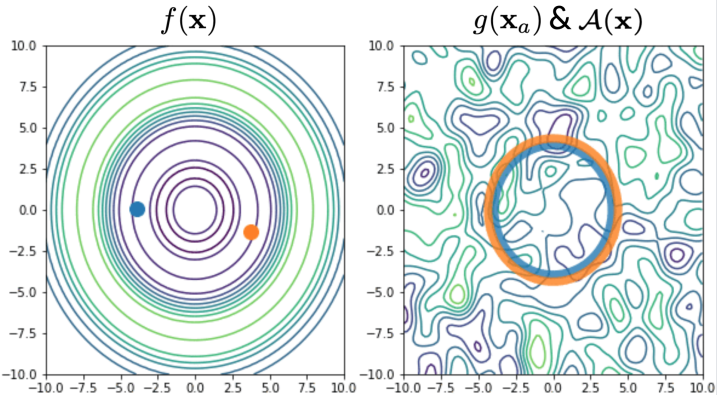
$$[\mathbf{k}_{Z\mathbf{x}}]_{mn} = \iint k_g(\mathbf{x}_a, \mathbf{x}'_a)p(\mathbf{x}_a \mid \mathbf{z}_m)p(\mathbf{x}'_a \mid \mathbf{z}'_m)\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}'_a$$

$$[\boldsymbol{K}_{ZZ}]_{mm} = \iint k_g(\mathbf{x}_a, \mathbf{x}'_a)p(\mathbf{x}_a \mid \mathbf{z}_m)p(\mathbf{x}'_a \mid \mathbf{z}'_m)\mathrm{d}\mathbf{x}_a\mathrm{d}\mathbf{x}'_a$$

Monte Carlo estimates could help if we didn't have the inverses...

Interdomain inducing variables

- The variational distribution is constructed by conditioning on "inducing observations".
- Which random variables we condition on determines the covariances



$f(\mathbf{x})$

$g(\mathbf{x}_a)$ & $\mathcal{A}(\mathbf{x})$

Interdomain Variational Inference

For Gaussian likelihoods

$$\mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(\mathbf{y}_n \mid f(\mathbf{x}_n))] = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}(y_n - \mu_n)^2 - \frac{\sigma_n^2}{2\sigma^2}$$

$$q(f(\mathbf{x})) = \mathcal{N}(f(\mathbf{x}); \underbrace{\mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}\mathbf{m}}_{\mu_n}, \underbrace{k_f(\mathbf{x}_n, \mathbf{x}_n) - \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}(\boldsymbol{K}_{ZZ} - \boldsymbol{S})\boldsymbol{K}_{ZZ}^{-1}\mathbf{k}_{Z\mathbf{x}}}_{\sigma_n^2})$$

With

$$[\mathbf{k}_{Z\mathbf{x}}]_{mn} = \int k_g(\mathbf{z}_m, \mathbf{x}_a)p(\mathbf{x}_a \mid \mathbf{x})\mathrm{d}\mathbf{x}_a$$

$$[\boldsymbol{K}_{ZZ}]_{mm'} = k_g(\mathbf{z}_m, \mathbf{z}_{m'})$$

Interdomain Variational Inference

For Gaussian likelihoods

$$\mathbb{E}_{q(f(\mathbf{x}_n))}[\log p(\mathbf{y}_n \mid f(\mathbf{x}_n))] = -\frac{1}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}(y_n - \mu_n)^2 - \frac{\sigma_n^2}{2\sigma^2}$$

$$q(f(\mathbf{x})) = \mathcal{N}(f(\mathbf{x}); \underbrace{\mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}\mathbf{m}}_{\mu_n}, \underbrace{k_f(\mathbf{x}_n, \mathbf{x}_n) - \mathbf{k}_{\mathbf{x}Z}\boldsymbol{K}_{ZZ}^{-1}(\boldsymbol{K}_{ZZ} - \boldsymbol{S})\boldsymbol{K}_{ZZ}^{-1}\mathbf{k}_{Z\mathbf{x}}}_{\sigma_n^2})$$

With

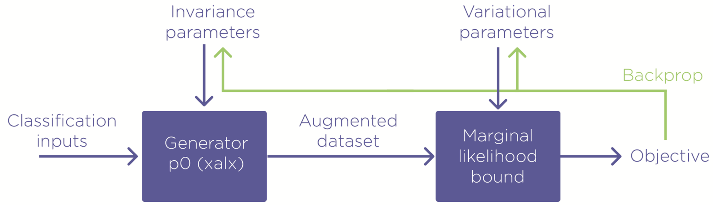
$$[\mathbf{k}_{Z\mathbf{x}}]_{mn} = \int k_g(\mathbf{z}_m, \mathbf{x}_a)p(\mathbf{x}_a \mid \mathbf{x})\mathrm{d}\mathbf{x}_a$$

$$[\boldsymbol{K}_{ZZ}]_{mm'} = k_g(\mathbf{z}_m, \mathbf{z}_{m'})$$

We can now find unbiased estimates of μ_n and σ_n^2! This trick also works with other likelihoods through the Pólya-Gamma trick!

Procedure

- Compute ELBO (marginal likelihood lower bound)
- Back-propagate to variational and invariance parameters through re-parameterisation
- Optimise jointly

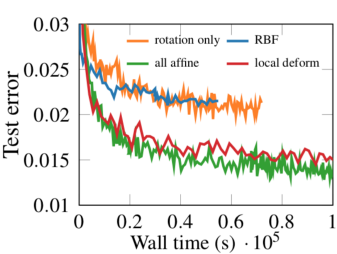


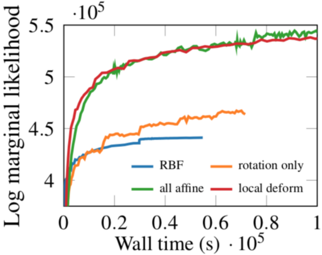
- No need for even a closed-form evaluation of the kernel k_f
- Insensitivity distribution $p(\mathbf{x}_a \mid \mathbf{x})$ can be implicit

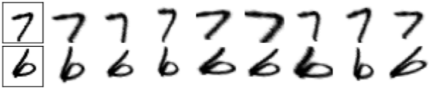
Results

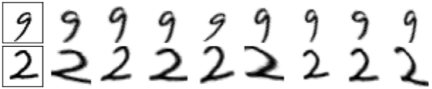
We used various $p(\mathbf{x}_a \mid \mathbf{x})$:

- Affine transformations (parameters: how much rotation / skew / scale to apply)
- Local deformations (parameters: how much deformation, how much to smooth deformations etc)









Conclusion

- Can express invariances in kernels (but kernels intractable)
- Can use the marginal likelihood for learning inductive biases
- We only need unbiased estimates of kernels to train!
- This is very much like learning the right data augmentation

Conclusion

- Can express invariances in kernels (but kernels intractable)
- Can use the marginal likelihood for learning inductive biases
- We only need unbiased estimates of kernels to train!
- This is very much like learning the right data augmentation

Going forward:

- Embed into deep structures (e.g. deep GPs/NNs, see latest arxiv pre-prints)
- Could we use infintely wide NN kernels?

References

Key references. See paper for more.

- Learning Invariances using the Marginal Likelihood; Mark van der Wilk, Matthias Bauer, ST John, James Hensman; NeurIPS (2018). (Main paper this was about)
- Gaussian Processes for Big Data; James Hensman, Nicolo Fusi, James D. Hensman; UAI (2013). (Variational bound we use)
- Variational Learning of Inducing Variables in Sparse Gaussian Processes; Michalis K. Titsias; AISTATS (2009). (First introduction of variational GP approx)
- Argumentwise invariant kernels for the approximation of invariant functions; David Ginsbourger, Xavier Bay, Olivier Roustant, Laurent Carraro; Annales de la Faculté des Sciences de Toulouse (2012). (Invariant kernels)
- Group theoretical methods in machine learning; Risi Kondor; PhD thesis (2008). (Invariant kernels)

Minimising training loss

We're looking for a fit that will generalise to new unseen test data. Let's minimise the training loss of the posterior mean.

$$\mathcal{L}(\theta, \sigma) = \sum_{n=1}^{N} \left[k_\theta(\mathbf{x}_n, X) \left(\mathbf{K}_\theta + \sigma^2 \mathbf{I} \right)^{-1} \mathbf{y} - y_n \right]^2 \tag{15}$$

$$\{\theta^*, \sigma^*\} = \operatorname*{argmin}_{\theta, \sigma} \mathcal{L}(\theta, \sigma) \tag{16}$$

Minimising training loss

We're looking for a fit that will generalise to new unseen test data.
Let's minimise the training loss of the posterior mean.

$$\mathcal{L}(\theta, \sigma) = \sum_{n=1}^{N} \left[k_\theta(\mathbf{x}_n, X) \left(\mathbf{K}_\theta + \sigma^2 \mathbf{I} \right)^{-1} \mathbf{y} - y_n \right]^2 \tag{15}$$

$$\{\theta^*, \sigma^*\} = \underset{\theta, \sigma}{\operatorname{argmin}}\, \mathcal{L}(\theta, \sigma) \tag{16}$$

We can fit anything with a tiny lengthscale and noise variance!

Marginal likelihood fixes things

Instead, choose hyperparameters by maximising marginal likelihood:

In above $\mathcal{L}$ is indicated by ‘datafit‘, while ‘ELBO‘ indicates the marginal likelihood.

- More sensible fit as the marginal likelihood rises
- Datafit gets worse!

Marginal likelihood trades off
data fit and model complexity.

References I

Artem Artemev, David R. Burt, and Mark van der Wilk. Tighter bounds on the log marginal likelihood of gaussian process regression using conjugate gradients. In Proceedings of the 38th International Conference on Machine Learning (ICML), 2021.

David R. Burt, Carl Edward Rasmussen, and Mark van der Wilk. Convergence of sparse variational inference in gaussian processes regression. Journal of Machine Learning Research, 21(131):1–63, 2020.

Adrià Garriga-Alonso, Carl Edward Rasmussen, and Laurence Aitchison. Deep convolutional networks as shallow Gaussian processes. In 7th International Conference on Learning Representations (ICLR), 2019.

Mark Gibbs and David Mackay. Efficient implementation of Gaussian processes. Technical report, Cavendish Laboratory, University of Cambridge, 1997.

James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI), pages 282–290, 2013.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as Gaussian processes. In 6th International Conference on Learning Representation (ICLR), 2018.

Alexander G. de G. Matthews. Scalable Gaussian Process Inference Using Variational Methods. PhD thesis, University of Cambridge, Cambridge, UK, 2016. available at http://mlg.eng.cam.ac.uk/matthews/thesis.pdf.

Alexander G. de G. Matthews, Jiri Hron, Mark Rowland, Richard E. Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. In 6th International Conference on Learning Representations (ICLR), 2018.

Radford M. Neal. Bayesian learning for neural networks, volume 118. Springer, 1996.

Roman Novak, Lechao Xiao, Yasaman Bahri, Jaehoon Lee, Greg Yang, Jiri Hron, Daniel A. Abolafia, Jeffrey Pennington, and Jascha Sohl-Dickstein. Bayesian deep convolutional networks with many channels are Gaussian processes. In 7th International Conference on Learning Representations (ICLR), 2019.

References II

Alex J. Smola and Bernhard Schölkopf. Sparse greedy matrix approximation for machine learning. In 17th International Conference on Machine Learning, Stanford, 2000, 2000.

Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In Proceedings of the 12th International Conference on Artificial Intelligence and Statistics, pages 567–574, 2009.

Michalis K. Titsias. Variational inference for Gaussian and determinantal point processes (talk slides). In Workshop on Advances in Variational Inference (NIPS), 2014.

Mark v.d.Wilk, Matthias Bauer, ST John, and James Hensman. Learning invariances using the marginal likelihood. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, Advances in Neural Information Processing Systems 31, pages 9938–9948. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/8199-learning-invariances-using-the-marginal-likelihood.pdf.

Ke Wang, Geoff Pleiss, Jacob Gardner, Stephen Tyree, Kilian Q Weinberger, and Andrew Gordon Wilson. Exact gaussian processes on a million data points. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/01ce84968c6969bdd5d51c5eeaa3946a-Paper.pdf.

Christopher Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In Advances in Neural Information Processing Systems 13, pages 682–688, 2001.

Greg Yang. Wide feedforward or recurrent neural networks of any architecture are Gaussian processes. In Advances in Neural Information Processing Systems 32 (NeurIPS). 2019.