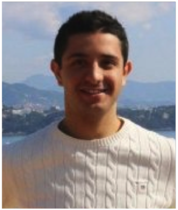
Practical and Scalable Inference for Deep Gaussian Processes




Kurt Cutajar[1] **Edwin V. Bonilla**[2] **Pietro Michiardi**[1] **Maurizio FIlippone**[1]

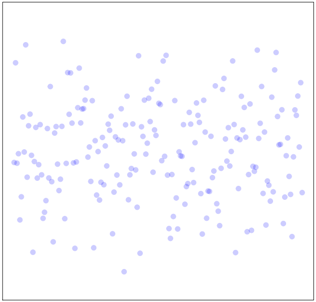
[1] EURECOM, Sophia Antipolis, France
[2] University of New South Wales, Sydney, Australia

April 14^{th}, 2017

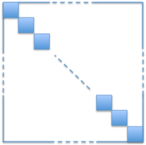
The Deep Learning Revolution

- Large representational power
- Mini-batch-based learning
- Exploit GPU and distributed computing
- Automatic differentiation
- Mature development of regularization (e.g., dropout)
- Application-specific representations (e.g., convolutional)

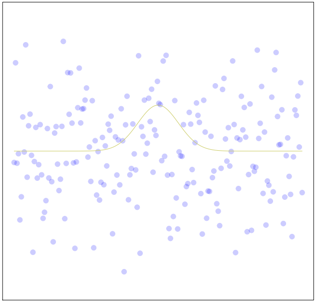
Gaussian Processes - Prior over Functions

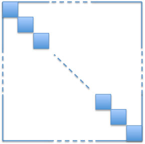


$K =$

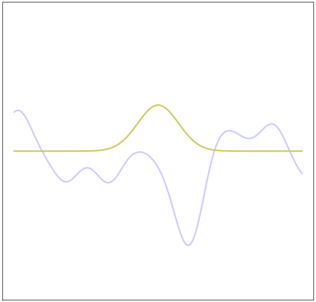


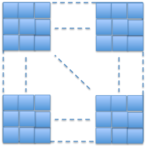
Gaussian Processes - Prior over Functions



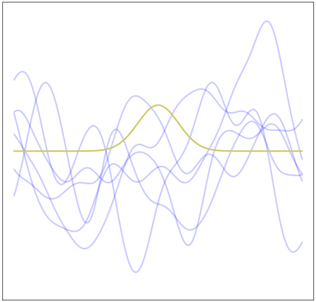
$K =$ 

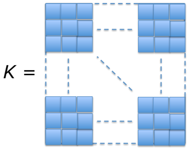
Gaussian Processes - Prior over Functions



$K =$ 

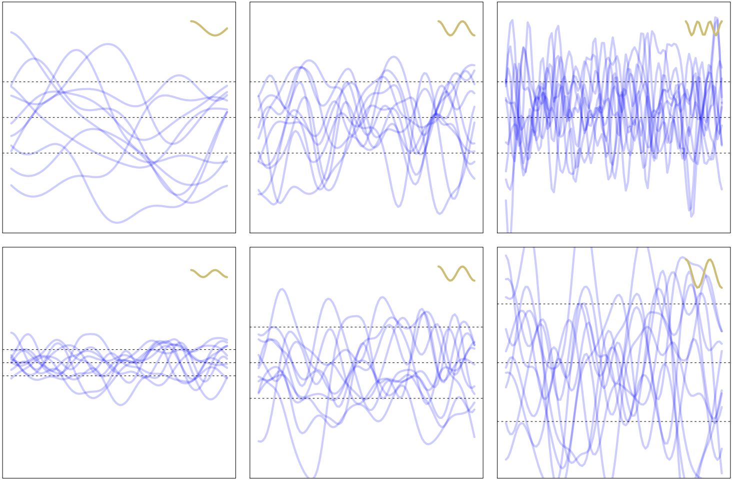
Gaussian Processes - Prior over Functions



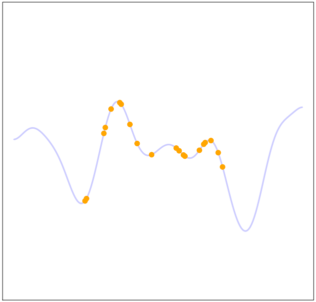


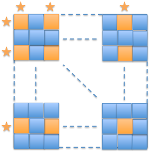
Gaussian Processes - Priors over Functions

- Infinite Gaussian random variables with parameterized and input-dependent covariance

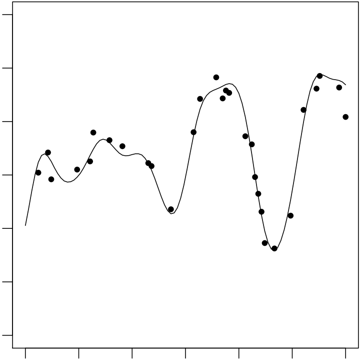


Gaussian Processes - Prior over Functions



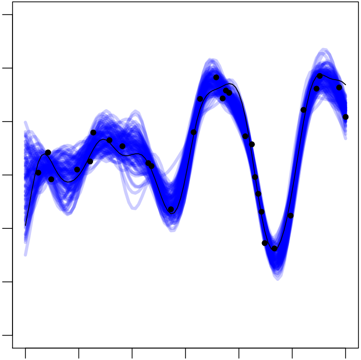
$K =$ 

- Regression example



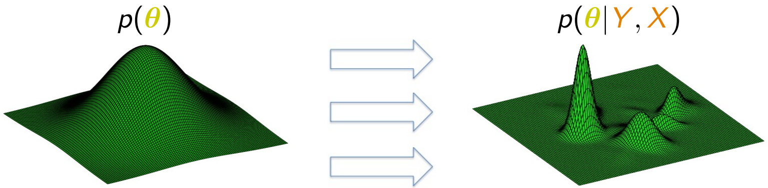
Gaussian Processes - Prior over Functions

- Regression example



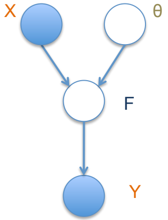
Bayesian Gaussian Processes

- Inputs $= X$ Labels $= Y$
- $K = K(X, \theta)$



$$p(\theta|Y,X) = \frac{p(Y|X,\theta)p(\theta)}{\int p(Y|X,\theta)p(\theta)d\theta}$$

- Can only model stationary functions (shallow model)
- $p(Y|X,\theta)$ might be expensive to compute
- $p(Y|X,\theta)$ might not even be computable!



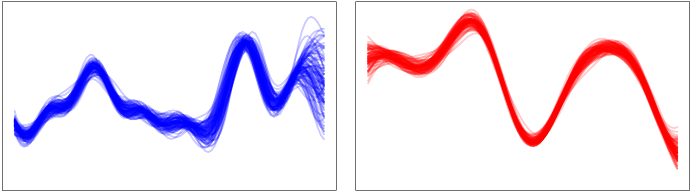
- Marginal likelihood

$$p(Y|X,\theta) = \int p(Y|F,X)p(F|\theta)dF$$

Is There Any Hope?

Can we exploit what made Deep Learning successful for practical and scalable learning of Gaussian processes?

- Composition of processes

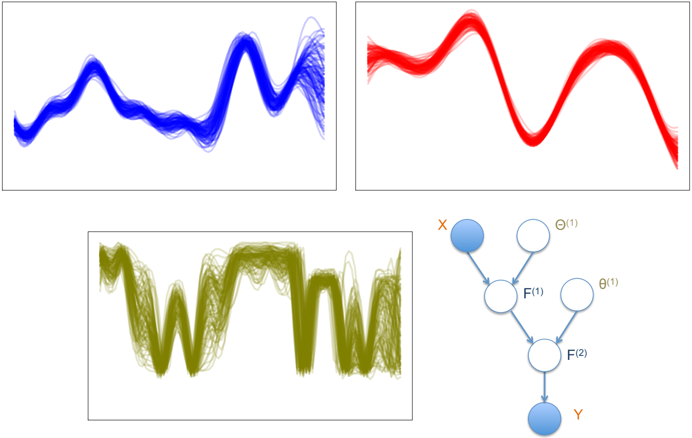


$$(f \circ g)(x)??$$

Damianou and Lawrence, *AISTATS*, 2013

Deep Gaussian Processes for Large Representational Power

- Composition of processes



Damianou and Lawrence, *AISTATS*, 2013

- Inference requires calculating integrals of this kind:

$$
\begin{aligned}
p(Y|X,\theta) \;=\; & \int p\left(Y|F^{(N_\mathrm{h})},\theta^{(N_\mathrm{h})}\right) \times \\
& p\left(F^{(N_\mathrm{h})}|F^{(N_\mathrm{h}-1)},\theta^{(N_\mathrm{h}-1)}\right) \times \ldots \times \\
& p\left(F^{(1)}|X,\theta^{(0)}\right) dF^{(N_\mathrm{h})} \ldots dF^{(1)}
\end{aligned}
$$

- Extremely challenging!

DGPs - Bochner's theorem

- Continuous shift-invariant covariance function

$$k(\mathbf{x}_i - \mathbf{x}_j | \boldsymbol{\theta}) = \sigma^2 \int p(\boldsymbol{\omega} | \boldsymbol{\theta}) \exp\left(\iota (\mathbf{x}_i - \mathbf{x}_j)^\top \boldsymbol{\omega}\right) d\boldsymbol{\omega}$$

DGPs - Bochner's theorem

- Continuous shift-invariant covariance function

$$k(\mathbf{x}_i - \mathbf{x}_j | \boldsymbol{\theta}) = \sigma^2 \int p(\boldsymbol{\omega} | \boldsymbol{\theta}) \exp\left(\iota (\mathbf{x}_i - \mathbf{x}_j)^\top \boldsymbol{\omega}\right) d\boldsymbol{\omega}$$

- Monte Carlo estimate

$$k(\mathbf{x}_i - \mathbf{x}_j | \boldsymbol{\theta}) \approx \frac{\sigma^2}{N_{\mathrm{RFF}}} \sum_{r=1}^{N_{\mathrm{RFF}}} \mathbf{z}(\mathbf{x}_i | \tilde{\boldsymbol{\omega}}_r)^\top \mathbf{z}(\mathbf{x}_j | \tilde{\boldsymbol{\omega}}_r)$$

with

$$\tilde{\boldsymbol{\omega}}_r \sim p(\boldsymbol{\omega} | \boldsymbol{\theta})$$

$$\mathbf{z}(\mathbf{x} | \boldsymbol{\omega}) = [\cos(\mathbf{x}^\top \boldsymbol{\omega}), \sin(\mathbf{x}^\top \boldsymbol{\omega})]^\top$$

Rahimi and Recht, *NIPS*, 2008 - Lázaro-Gredilla et al., *JMLR*, 2010

DGPs with Random Fourier Features

- Define

$$\Phi^{(l)} = \sqrt{\frac{\sigma^2}{N_{\mathrm{RFF}}^{(l)}}} \left[\cos\left(F^{(l)}\Omega^{(l)}\right), \sin\left(F^{(l)}\Omega^{(l)}\right)\right]$$

and

$$F^{(l+1)} = \Phi^{(l)} W^{(l)}$$

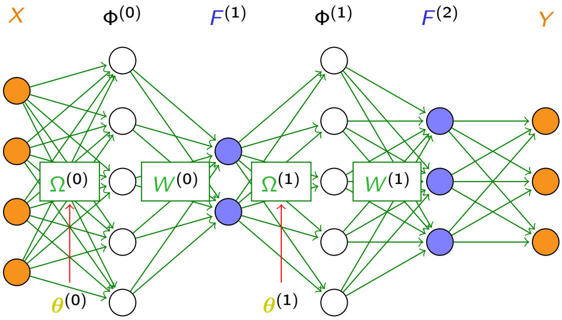
- At each layer, the priors over the weights are

$$p\left(\Omega_{\cdot j}^{(l)} \middle| \theta^{(l)}\right) = \mathcal{N}\left(\mathbf{0}, \left(\Lambda^{(l)}\right)^{-1}\right)$$

and

$$p\left(W_{\cdot i}^{(l)}\right) = \mathcal{N}\left(\mathbf{0}, I\right)$$

DGPs with random features become DNNs



Stochastic Variational Inference

- Define $\Psi = (\Omega^{(0)}, \ldots, W^{(0)}, \ldots)$
- Lower bound for $\log [p(Y|X, \theta)]$

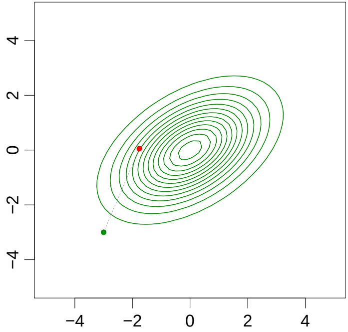
$$\mathrm{E}_{q(\Psi)} \left(\log \left[p \left(Y | X, \Psi, \theta \right) \right] \right) - \mathrm{DKL} \left[q(\Psi) \| p \left(\Psi | \theta \right) \right],$$

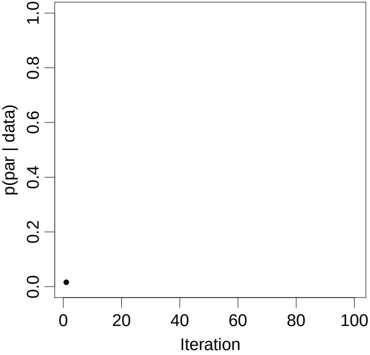
  where $q(\Psi)$ approximates $p(\Psi|Y, \theta)$.
- DKL computable analytically if q and p are Gaussian!

Optimize the lower bound wrt the parameters of $q(\Psi)$

Stochastic Variational Inference

$$\text{vpar}' = \text{vpar} + \frac{\alpha_t}{2} \widetilde{\nabla_{\text{vpar}}}(\text{LowerBound}) \qquad \alpha_t \to 0$$





Robbins and Monro, *AoMS*, 1951

Stochastic Variational Inference

- Assume that the likelihood factorizes

$$p(Y|X, \Psi, \theta) = \prod_k p(\mathbf{y}_k|\mathbf{x}_k, \Psi, \theta)$$

- Doubly stochastic **unbiased** estimate of the expectation term
  - Mini-batch

$$\mathrm{E}_{q(\Psi)}\left(\log\left[p\left(Y|X, \Psi, \theta\right)\right]\right) \approx \frac{n}{m} \sum_{k \in \mathcal{I}_m} \mathrm{E}_{q(\Psi)}\left(\log\left[p(\mathbf{y}_k|\mathbf{x}_k, \Psi, \theta)\right]\right)$$

  - Monte Carlo

$$\mathrm{E}_{q(\Psi)}\left(\log\left[p(\mathbf{y}_k|\mathbf{x}_k, \Psi, \theta)\right]\right) \approx \frac{1}{N_{\mathrm{MC}}} \sum_{r=1}^{N_{\mathrm{MC}}} \log[p(\mathbf{y}_k|\mathbf{x}_k, \tilde{\Psi}_r, \theta)]$$

with $\tilde{\Psi}_r \sim q(\Psi)$.

Stochastic Variational Inference

- Reparameterization trick

$$(\tilde{W}_r^{(l)})_{ij} = \sigma_{ij}^{(l)} \varepsilon_{rij}^{(l)} + \mu_{ij}^{(l)}, \tag{1}$$

  with $\varepsilon_{rij}^{(l)} \sim \mathcal{N}(0,1)$
- ... same for Ω
- Variational parameters

$$\mu_{ij}^{(l)}, (\sigma^2)_{ij}^{(l)} \dots$$

  ... and the ones for Ω
- Optimization with automatic differentiation in TensorFlow

Kingma and Welling, *ICLR*, 2014

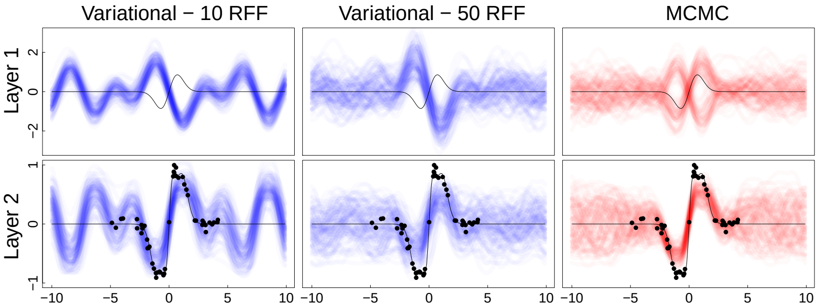
Comparison with MCMC

- Generate data from

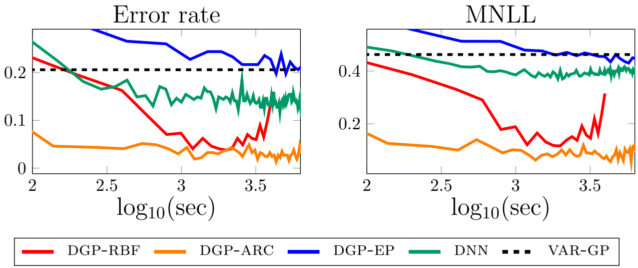
$$\mathcal{N}(y|h(h(x)), 0.01)$$

with

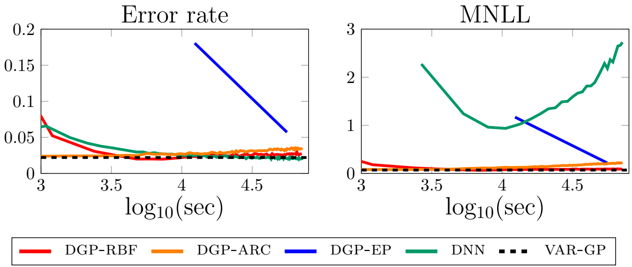
$$h(x) = 2x\exp(-x^2)$$



EEG dataset
$(n = 14979,\ d = 14)$



MNIST dataset
($n = 60000$, $d = 784$)

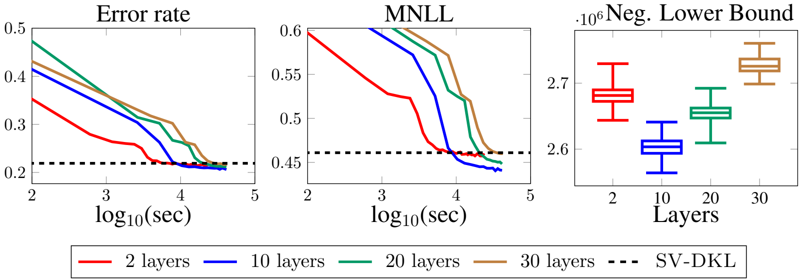


Results - MNIST-8M

- Variant of MNIST with 8.1M images
- 99+% accuracy!
- Also, check out Krauth et al., arXiv 2016

Airline dataset

($n = 5\mathrm{M}+,\ d = 8$)



Conclusions

- Contributions
  - Novel formulation of DGPs based on random features
  - We study the connections with DNNs
  - Scalable and practical DGPs inference - no inverses!

Conclusions

- Contributions
  - Novel formulation of DGPs based on random features
  - We study the connections with DNNs
  - Scalable and practical DGPs inference - no inverses!
- Ongoing work
  - Large dimensional problems with Fastfood
  - Other random features
  - Improving distributed implementation
  - Adding convolutional layers for image problems
  - Unsupervised learning, Bayesian Optimization, Calibration, ...

References and Acknowledgments

- Reference:

[1] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone. **Random feature expansions for deep Gaussian processes**, 2016. *arXiv:1610.04386*.

- Code:

`github.com/mauriziofilippone/deep_gp_random_features`

References and Acknowledgments

- Reference:

[1] K. Cutajar, E. V. Bonilla, P. Michiardi, and M. Filippone. **Random feature expansions for deep Gaussian processes**, 2016. *arXiv:1610.04386*.

- Code:

`github.com/mauriziofilippone/deep_gp_random_features`

Thank you!

AXA Research Fund