# Deep Gaussian Processes using Expectation Propagation and Monte Carlo Methods

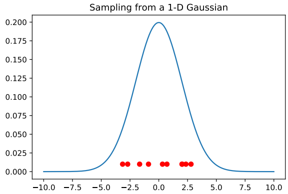
Gonzalo Hernández Muñoz

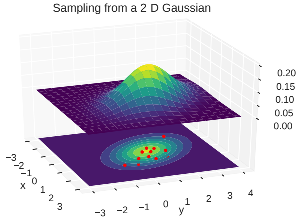
December 17, 2018

# Table of contents

# Gaussian Processes

- A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.



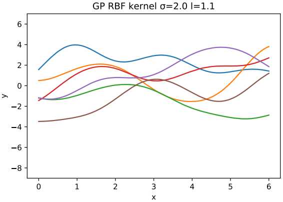
# Gaussian Processes

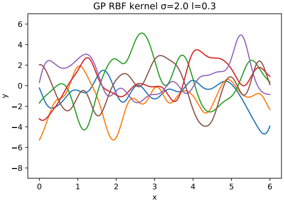
- Defined by its mean function and co-variance function (kernel).
- Sampling from a GP: each sample is a function.

$$\text{GP prior: } f(\mathbf{x}) \backsim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x'}))\,,$$

$$k_{\text{rbf}}(\mathbf{x}, \mathbf{x'}) = \sigma^2 exp\left\{ -\frac{||\mathbf{x} - \mathbf{x'}||^2}{\ell^2} \right\}\,.$$

- The **properties of the function** are specified by the kernel.



# Gaussian Processes Regression

- In a regression setting, we have pairs of training values and their corresponding observations $\{\mathbf{x}_i, y_i\}_{i=1}^{N}$.

$$y_i = f(\mathbf{x}_i) + \epsilon\,, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, \sigma^2)\,.$$

- We set a GP prior for the **joint distribution** for both vectors of function values, $\mathbf{f}_\star$ and $\mathbf{f}$:

$$p(\mathbf{f}, \mathbf{f}_\star) = \mathcal{N}\left(\begin{bmatrix}\mathbf{f}\\ \mathbf{f}_\star\end{bmatrix} \middle| \begin{bmatrix}\mathbf{0}\\ \mathbf{0}\end{bmatrix}, \begin{bmatrix}\mathbf{K}_{\mathbf{f},\mathbf{f}} & \mathbf{K}_{\mathbf{f},\star}\\ \mathbf{K}_{\star,\mathbf{f}} & \mathbf{K}_{\star,\star}\end{bmatrix}\right)\,.$$

- These **matrices** are computed with the kernel function $k(x, x')$:

$$[\mathbf{K}_{\mathbf{f},\mathbf{f}}]_{n,n'} = k(\mathbf{x}_n, \mathbf{x}_{n'})\,, \qquad [\mathbf{K}_{\star,\mathbf{f}}]_{k,n} = k(\mathbf{x}^\star_k, \mathbf{x}_n)\,,$$
$$[\mathbf{K}_{\mathbf{f},\star}]_{n,k} = k(\mathbf{x}_n, \mathbf{x}^\star_k)\,, \qquad [\mathbf{K}_{\star,\star}]_{k,k'} = k(\mathbf{x}^\star_k, \mathbf{x}^\star_{k'})\,.$$

- We combine it with the Gaussian **likelihood**:

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{y}|\mathbf{f},\, \sigma^2\mathbf{I})\,.$$

# Gaussian Processes Regression

- The **predictive distribution** is given by:

$$p(\mathbf{f}_\star|\mathbf{y}) = \mathcal{N}(\mathbf{f}_\star|\mathbf{m}, \mathbf{\Sigma}) ,$$
$$\mathbf{m} = \mathbf{K}_{\star,\mathbf{f}}(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I})^{-1}\mathbf{y} ,$$
$$\mathbf{\Sigma} = \mathbf{K}_{\star,\star} - \mathbf{K}_{\star,\mathbf{f}}(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I})^{-1}\mathbf{K}_{\mathbf{f},\star} .$$

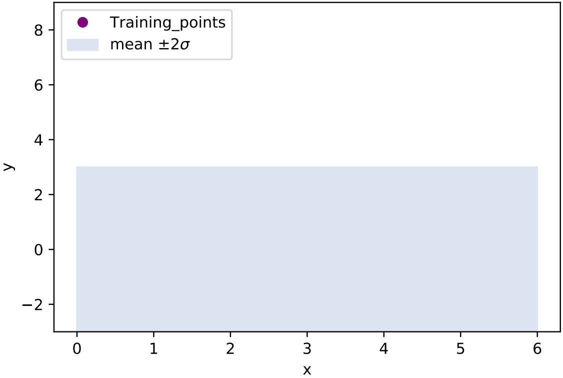
- The **marginal likelihood** is also given by a Gaussian:

$$p(\mathbf{y}) = \int p(\mathbf{y}|\mathbf{f})p(\mathbf{f}, \mathbf{f}_\star)\ d\mathbf{f}d\mathbf{f}_\star ,$$
$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}} + \sigma^2\mathbf{I}) .$$

- The above expressions require the inversion of a matrix of size $N \times N$ which requires $\mathcal{O}(N^3)$ operations!

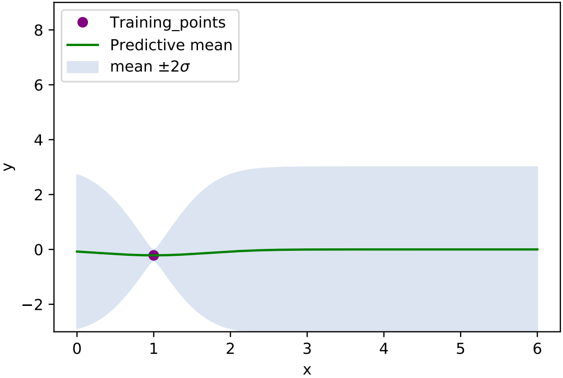
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



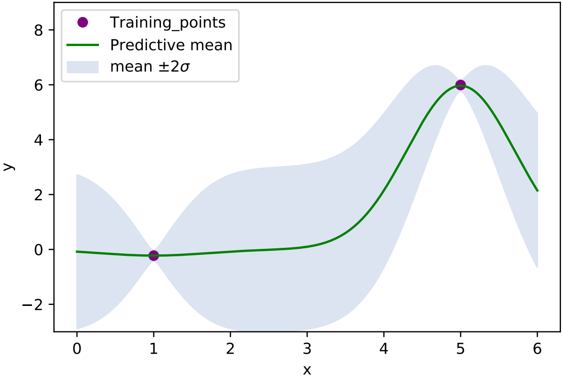
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



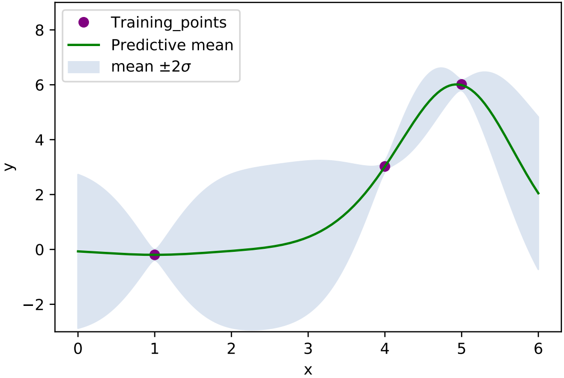
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



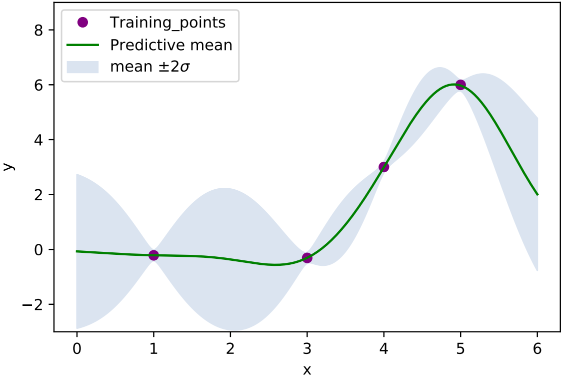
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



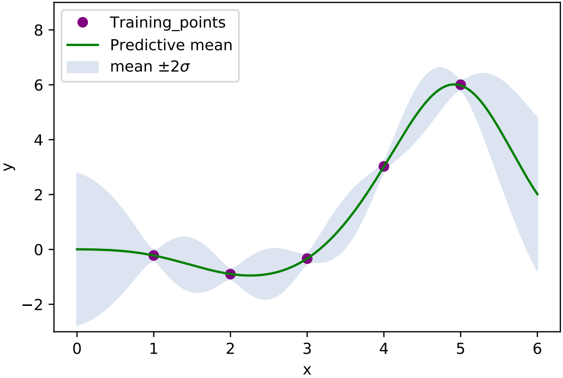
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



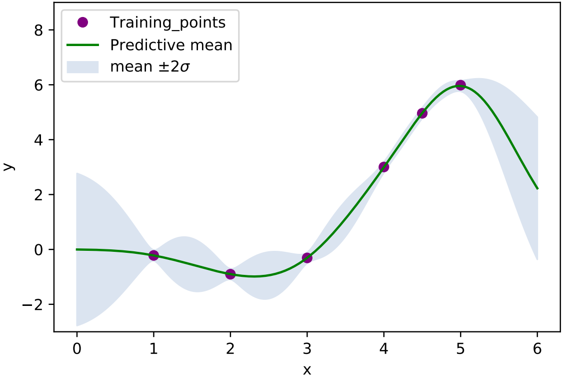
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



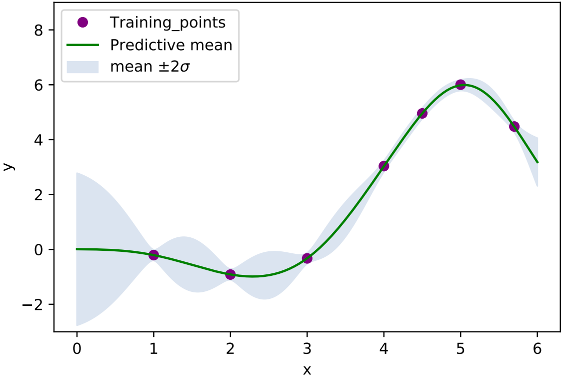
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



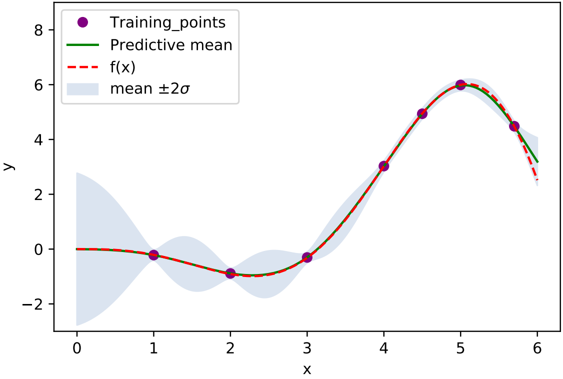
# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



# Gaussian Processes Regression

- GP regression provides a **closed-form** posterior distribution for $f(\cdot)$.



# The FITC Gaussian Process

- We introduce a set of $M$ **"inducing points"** $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^M$ with their corresponding latent function values:

$$\mathbf{u} = [f(\mathbf{z}_1), \ldots, f(\mathbf{z}_M)]^T .$$

- We also set a GP prior on the inducing points:

$$p(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}}) .$$

- We assume that $\mathbf{f}$ and $\mathbf{f}_\star$ are independent given $\mathbf{u}$:

$$p(\mathbf{f}, \mathbf{f}_\star) \approx \int p(\mathbf{f}|\mathbf{u}) p(\mathbf{f}_\star|\mathbf{u}) p(\mathbf{u}) d\mathbf{u} ,$$

Training conditional: $p(\mathbf{f}|\mathbf{u}) = \mathcal{N}(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\, \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{Q}_{\mathbf{f},\mathbf{f}}) ,$

Test conditional: $p(\mathbf{f}_\star|\mathbf{u}) = \mathcal{N}(\mathbf{K}_{\star,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\, \mathbf{K}_{\star,\star} - \mathbf{Q}_{\star,\star}) ,$

Where $\mathbf{Q}_{\mathbf{a},\mathbf{b}} \triangleq \mathbf{K}_{\mathbf{a},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{b}} .$

# The FITC Gaussian Process

- FITC assumes that the training conditional factorizes.

$$p(\mathbf{f}, \mathbf{f}_\star) \approx q_{\mathsf{FITC}}(\mathbf{f}, \mathbf{f}_\star) = \int q_{\mathsf{FITC}}(\mathbf{f}|\mathbf{u}) p(\mathbf{f}_\star|\mathbf{u}) p(\mathbf{u}) \, d\mathbf{u} \,,$$

$$p(\mathbf{f}|\mathbf{u}) \approx q_{\mathsf{FITC}}(\mathbf{f}|\mathbf{u}) = \prod_{i=1}^{N} p(f_i|\mathbf{u}) \,.$$

- The predictive distribution can be calculated in the same way as in the full GP case.

$$p(\mathbf{f}_\star|\mathbf{y}) = \mathcal{N}(\mathbf{f}_\star|\mathbf{K}_{\star,\mathbf{u}}\boldsymbol{\Sigma}\mathbf{K}_{\mathbf{u},\mathbf{f}}\boldsymbol{\Lambda}^{-1}\mathbf{y}, \;\; \mathbf{K}_{\star,\star} - \mathbf{Q}_{\star,\star} + \mathbf{K}_{\star,\mathbf{u}}\boldsymbol{\Sigma}\mathbf{K}_{\mathbf{u},\star}) \,,$$
$$\boldsymbol{\Sigma} = (\mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{u},\mathbf{f}}\boldsymbol{\Lambda}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}})^{-1} \,,$$
$$\boldsymbol{\Lambda} = \mathrm{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{Q}_{\mathbf{f},\mathbf{f}} + \sigma^2_{\mathsf{noise}}\mathbf{I}\right] \,.$$

- The computational cost is reduced to $\mathcal{O}(M^2 N)$ (and $M << N$)

# Approximate inference

- When doing inference in probabilistic models we usually use Bayes' theorem to calculate the posterior distribution of the parameters:

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

- Most times the integral required to calculate $p(\mathcal{D})$ is intractable.
- GPs only have a closed form expression if the likelihood is Gaussian $p(\mathcal{D}|\theta)$.
- Approximate inference techniques try to find a distribution $q(\theta)$ as close as possible to the true posterior by minimizing a distance measure KL$(\cdot||\cdot)$:

$$q(\theta) \approx p(\theta|\mathcal{D})$$

- Minimizing KL$(q||p)$ or KL$(p||q)$ yields **different results**.

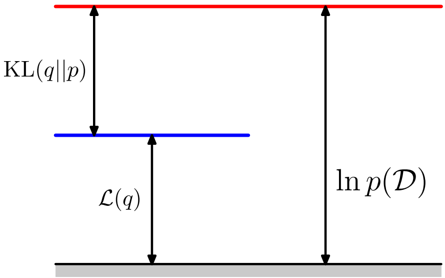
# Variational inference

- We could try to minimize $\mathsf{KL}(q||p)$ directly.

# Variational inference

- ~~We could try to minimize $\mathsf{KL}(q||p)$ directly.~~ We can not evaluate $\mathsf{KL}(q||p)$
- Alternatively we can maximize the lower bound. It is possible to evaluate

$$\mathcal{L}(q) = -\mathsf{KL}(q||p) + \ln p(D)$$




*Source*: Bishop, Christopher M. "Pattern recognition and machine learning, 2006."

# Expectation Propagation

- EP assumes that the likelihood **factorizes over the data**:

$$p(\theta|\mathcal{D}) \propto p(\theta)\prod_{i=1}^{N} p(y_i|\theta) = \prod_{i=0}^{N} f_i(\theta)$$

- The approximation also factorizes as:

$$q(\theta) \propto \prod_{i=0}^{N} \tilde{f}_i(\theta)$$

- The approximate factors are Gaussian while the exact factors may not.
- The ideal value for the $i$-th approximate factor would be given by:

$$\min_{\tilde{f}_i(\theta)} \mathrm{KL}(f_i(\theta)\prod_{j\neq i}\tilde{f}_j(\theta)||\prod_{i=0}^{N}\tilde{f}_i(\theta))$$

# Expectation Propagation

- EP solves this problem with an iterative procedure:
  1. Calculate **"cavity"** by removing one of the approx. factors from approx. posterior:

     $$q^{\setminus i}(\theta) \propto \frac{q(\theta)}{\tilde{f}_i(\theta)}$$

  2. Substitute the removed factor by the exact one into the **"tilted"** distribution:

     $$\hat{p}_i(\theta) \propto f_i(\theta) q^{\setminus i}(\theta)$$

  3. **Match** approx. posterior **moments** to those of the tilted:

     $$q_{\text{new}}(\theta) \leftarrow \text{min}_{q(\theta)} \text{KL}(\hat{p}_i(\theta) || q(\theta))$$

  4. **Update** the approx. factor:

     $$\tilde{f}_i(\theta) \propto \frac{q_{\text{new}}(\theta)}{q^{\setminus i}(\theta)}$$

# Why we need DGPs

- Some problems require complex covariance functions.
- Specifying a wrong kernel can lead to bad results.
- DGPs can repair the damage done by sparse approximations.

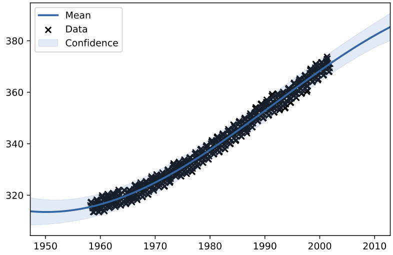



Figure: Fitting GP with RBF to Mauna Loa

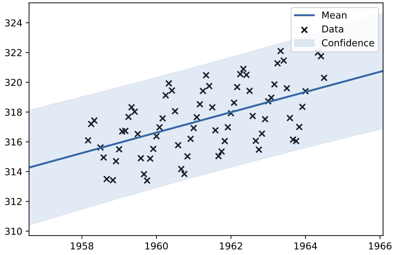
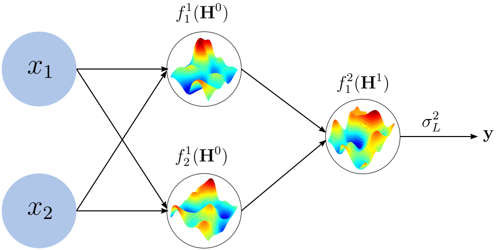



Figure: Fitting GP with RBF to Mauna Loa, Detail

# Deep Gaussian Processes

- Defined as a composition of functions.
- A DGP model is comprised of $L$ layers with $\{D^l\}_{l=1}^{L}$ nodes on each layer.
- Functions in each node are modeled by a GP and receive the output of the previous layer as input.




$$\mathbf{H}^l = \begin{bmatrix} h^l_{1,1} & \dots & h^l_{1,D_l} \\ \vdots & \ddots & \vdots \\ h^l_{N,1} & \dots & h^l_{N,D_l} \end{bmatrix}$$

$$h^l_{n,i} = f^l_i(\mathbf{h}^{l-1}_n)$$

# Deep Gaussian Processes

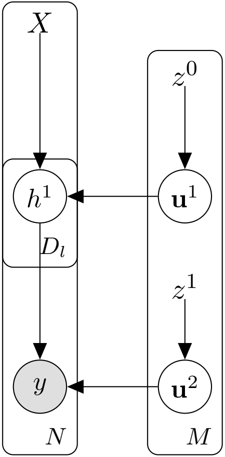
$$p(\mathbf{u}^l|\theta^l) = \mathcal{N}(\mathbf{u}^l|\mathbf{0}, \mathbf{K}_{\mathbf{u}^l,\mathbf{u}^l}), \quad l = 1, \ldots, L\,.$$

$$p(\mathbf{h}^l|\mathbf{u}^l, \mathbf{h}^{l-1}, \sigma_l^2) = \prod_{n=1}^{N} \mathcal{N}(h_n^l|\mathbf{A}_n^l\mathbf{u}^l,\ \mathbf{K}_{h_n^l,h_n^l} - \mathbf{Q}_n^l)\,,$$

$$p(\mathbf{y}|\mathbf{u}^L, \mathbf{h}^{L-1}, \sigma_L^2) = \prod_{n=1}^{N} \mathcal{N}(y_n|\mathbf{A}_n^L\mathbf{u}^L,\ \mathbf{K}_{h_n^L,h_n^L} - \mathbf{Q}_n^L)\,.$$

$$\mathbf{A}_n^l \triangleq \mathbf{K}_{h_n^l,\mathbf{u}^l}\mathbf{K}_{\mathbf{u}^l,\mathbf{u}^l}^{-1}\,,$$

$$\mathbf{Q}_n^l \triangleq \mathbf{K}_{h_n^l,\mathbf{u}^l}\mathbf{K}_{\mathbf{u}^l,\mathbf{u}^l}^{-1}\mathbf{K}_{\mathbf{u}^l,h_n^l} + \sigma_l^2\,,$$



# Example with $L = 2$ and $D_l = 1$

- We are interested in calculating the marginal likelihood to optimize the model parameters:

$$\boldsymbol{\alpha} = \{\mathbf{z}^0, \mathbf{z}^1, \theta^1, \theta^2, \sigma_1^2, \sigma_2^2\} ,$$
$$p(\mathbf{y}|\boldsymbol{\alpha}) = \int p(\mathbf{y}, \mathbf{h}^1, \mathbf{u}^1, \mathbf{u}^2|\boldsymbol{\alpha}) \; d\mathbf{h}^1 \, d\mathbf{u}^1 \, d\mathbf{u}^2 \, .$$

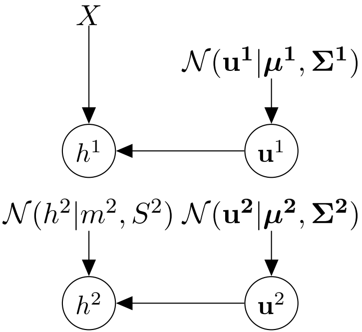
- The posterior distribution for the inducing points can be used to make predictions

$$p(\mathbf{u}^1, \mathbf{u}^2|\mathbf{y}) = \frac{1}{p(\mathbf{y}|\boldsymbol{\alpha})} \int p(\mathbf{y}, \mathbf{h}^1, \mathbf{u}^1, \mathbf{u}^2|\boldsymbol{\alpha}) \; d\mathbf{h}^1 \, .$$

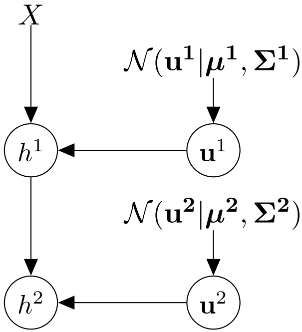
- Unfortunately some of the integrals are intractable.

# State of the art for DGP inference

| Reference | Approx. posterior | Technique |
|---|---|---|
| [Damianou and Lawrence, 2013] | $q(\mathbf{h},\mathbf{u}) = \prod_{l=1}^{L} q(\mathbf{h}^l)q(\mathbf{u}^l)$ | VI |
| [Bui et al., 2016] | $q(\mathbf{h},\mathbf{u}) = \prod_{l=1}^{L} p(\mathbf{h}^l \vert \mathbf{u}^l, \mathbf{h}^{l-1})p(\mathbf{u}^l)g(\mathbf{u}^l)^N$ | AEP |
| [Salimbeni and Deisenroth, 2017] | $q(\mathbf{h},\mathbf{u}) = \prod_{l=1}^{L} p(\mathbf{h}^l \vert \mathbf{u}^l, \mathbf{h}^{l-1})q(\mathbf{u}^l)$ | VI |




[Damianou and Lawrence, 2013]




[Bui et al., 2016]

[Salimbeni and Deisenroth, 2017]

# DGP-AEPMCM

- We approximate the posterior for the inducing points of each layer using **SEP**:

$$p(\mathbf{u}^l|\mathbf{y}) \approx q(\mathbf{u}^l) \propto p(\mathbf{u}^l)g(\mathbf{u}^l)^N \,.$$

- With the SEP approximation, the EP approximation to the marginal likelihood simplifies and is given by [Seeger, 2005]:

$$\ln p(\mathbf{y}|\boldsymbol{\alpha}) \approx \mathcal{F}(\boldsymbol{\alpha})$$
$$= \sum_{l=1}^{L} \left[ (1-N)\Phi(\theta^{q^l}) + N\Phi(\theta^{\backslash l}) - \Phi(\theta^l_{\text{prior}}) \right] + \sum_{n=1}^{N} \ln \mathcal{Z}_n \,,$$
$$\ln \mathcal{Z}_n = \ln \mathbb{E}_{q^{\backslash l}(\mathbf{u})} \left[ p(y_n|\mathbf{u}, \mathbf{x}_n) \right] \,.$$

- We optimize this quantity instead of doing the EP updates.

# DGP-AEPMCM, calculating $\mathcal{Z}_n$ with $L = 2$

- $\mathcal{Z}_n$ represents the probability of observing $y_n$ for a given input $x_n$ under the cavity distribution $q^{\backslash l}$.
- Expanding the expression for $\mathcal{Z}_n$:

$$\mathcal{Z}_n = \int p(y_n | h^1, \mathbf{u}^2) q^{\backslash 2}(\mathbf{u}^2) p(h^1 | \mathbf{x}_n, \mathbf{u}^1) q^{\backslash 1}(\mathbf{u}^1) \; d\mathbf{u}^1 \, d\mathbf{u}^2 \, dh^1 \, .$$

- We can exactly marginalize $\mathbf{u}^1$ and $\mathbf{u}^2$:

$$\mathcal{Z}_n = \int q(y_n | h^1) q(h^1) \; dh^1 \, .$$

- Still requires to calculate the integral of a kernel with respect to a random variable $h^1$.
- **Solution**: Take samples from $\hat{h}^1 \sim q(h^1)$ and propagate them.

$$\mathcal{Z}_n \approx \frac{1}{S} \sum_{s=1}^{S} q(y_n | \hat{h}^1_s) \, .$$

# DGP-AEPMCM

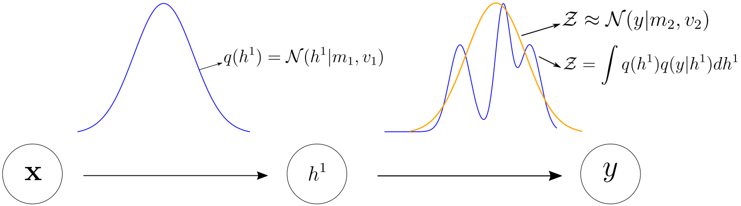



Figure: Work in [Bui et al., 2016]

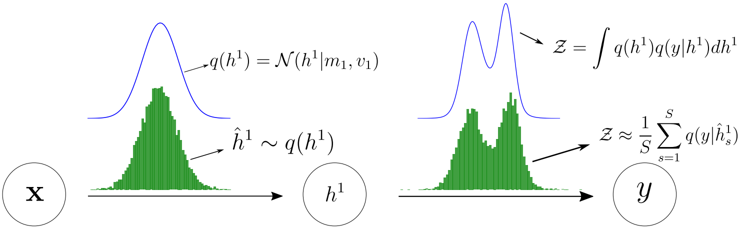



Figure: Our proposal

# Regression results

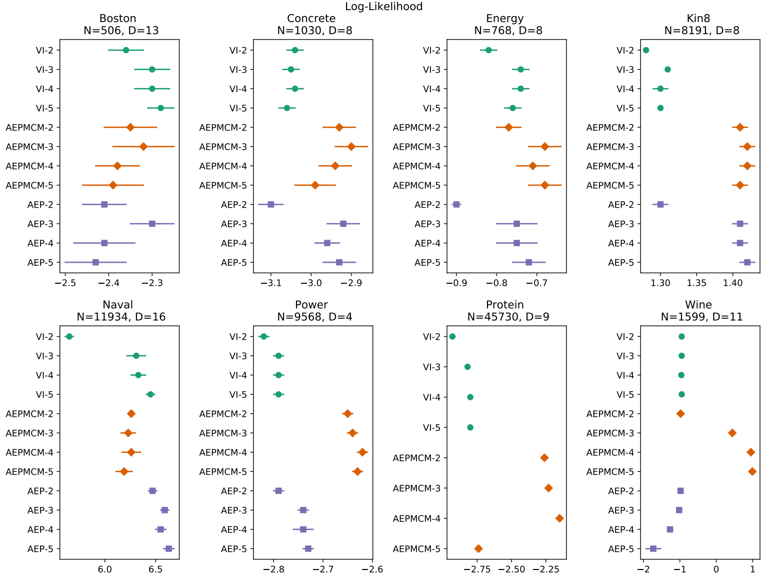



Figure: Test Log-Likelihood results (Higher, to the right is better)

# Regression results

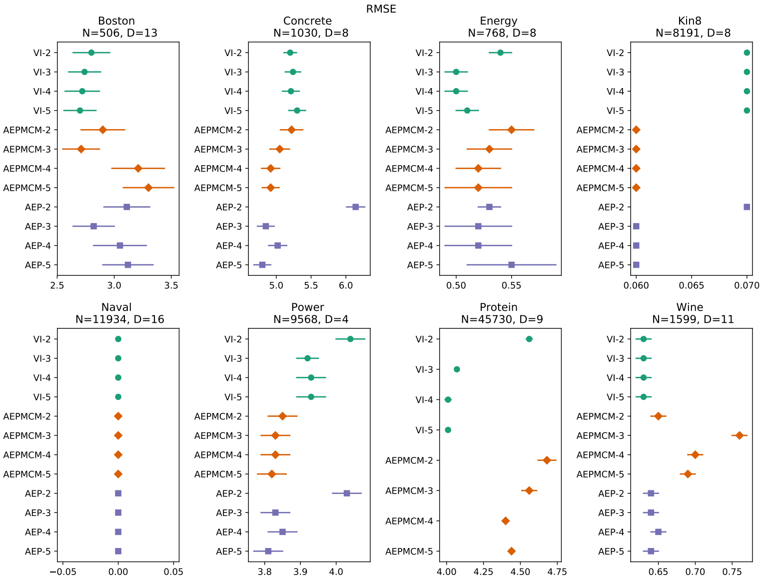



Figure: RMSE (Lower, to the left is better)

# Multi-modal Experiment

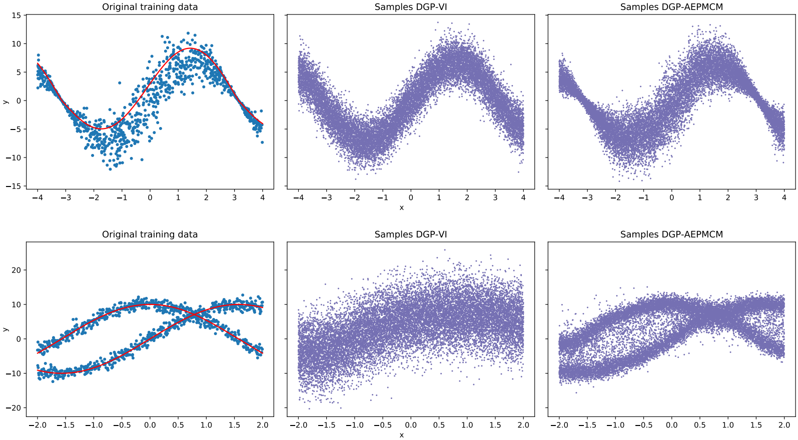
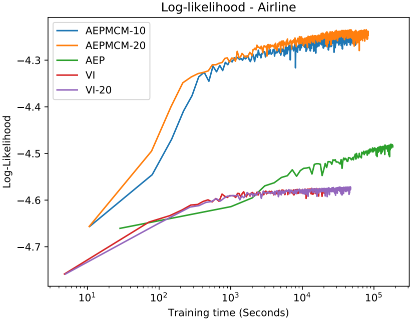



Figure: Samples taken from predictive distribution.

- This is due to differences in the function that each method is optimizing:

| VI | AEP |
|---|---|
| $\mathbb{E}_q\left[\ln p(y\|\mathbf{u},\mathbf{X})\right]$ | $\ln \mathbb{E}_{q\backslash}\left[p(y\|\mathbf{u},\mathbf{X})\right]$ |

# Big Data experiment



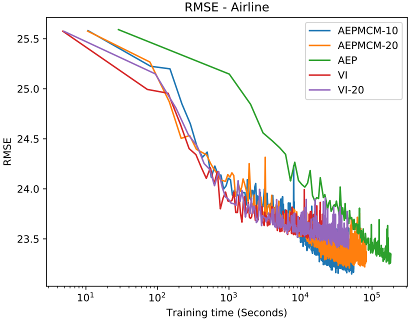




Table: Results for the Big data experiments. Airline N=2,082,007 D=8

| Model | Avg. gradient step (seconds) | RMSE | Log-Likelihood |
|---|---|---|---|
| DGP-AEPMCM-10 | 0.0221 | 23.22 | -4.25 |
| DGP-AEPMCM-20 | 0.0347 | 23.32 | -4.24 |
| DGP-VI-10 | 0.0202 | 23.55 | -4.58 |
| DGP-VI-20 | 0.0388 | 23.47 | -4.57 |
| DGP-AEP | 0.2914 | 23.32 | -4.48 |

# Conclusions

- We have shown that removing the Gaussian assumption for the output of the layers and propagating samples improve results.
- Our approximate inference method can capture complex properties about the process that generates data (like modeling multimodal distributions or noise dependent of the input).
- Our proposal is suited for big data problems.

# Future work

- The method can be adapted to tackle classification problems.
- Removing the hypothesis that the approximate posterior distributions are Gaussian could further improve results.

# Bibliography


Bui, T. D., Hernández-Lobato, J. M., Hernández-Lobato, D., Li, Y., and Turner, R. E. (2016).
Deep gaussian processes for regression using approximate expectation propagation.
In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, pages 1472–1481.

Damianou, A. and Lawrence, N. (2013).
Deep gaussian processes.
In Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics, volume 31 of Proceedings of Machine Learning Research, pages 207–215. PMLR.

Salimbeni, H. and Deisenroth, M. (2017).
Doubly stochastic variational inference for deep gaussian processes.
In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, Advances in Neural Information Processing Systems 30, pages 4588–4599. Curran Associates, Inc.

Seeger, M. (2005).
Expectation propagation for exponential families.
Technical report.

# Utils

$$\mathsf{KL}(q||p) = \int q(\theta) \ln \frac{q(\theta)}{p(\theta|\mathcal{D})} d\theta$$

$$\mathcal{L}(q) = \mathbb{E}_{q(\theta)} \left[ \ln \frac{p(\theta, \mathcal{D})}{q(\theta)} \right]$$