Scalable Gaussian Processes

Zhenwen Dai

Amazon

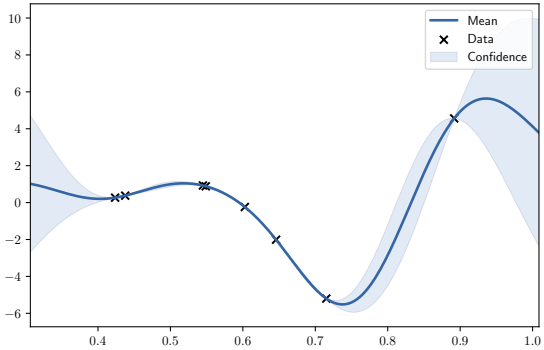
September 4, 2018 @GPSS2018

Gaussian process

Input and Output Data:

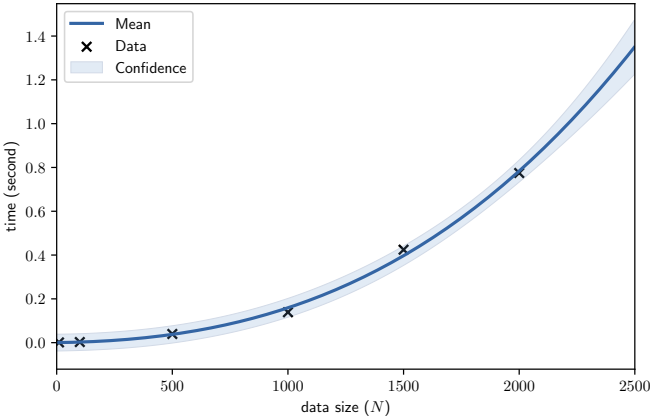
$$\mathbf{y} = (y_1, \ldots, y_N), \quad \mathbf{X} = (\mathbf{x}_1, \ldots, \mathbf{x}_N)^\top$$

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}\left(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}\right), \quad p(\mathbf{f}|\mathbf{X}) = \mathcal{N}\left(\mathbf{f}|0, \mathbf{K}(\mathbf{X}, \mathbf{X})\right)$$



The scaling behavior w.r.t. N

The computational cost of Gaussian process is $O(N^3)$.



Behind a Gaussian process fit

- Point Estimate / Maximum A Posteriori (MAP) of hyper-parameters.

$$\theta^* = \arg\max_\theta \log p(\mathbf{y}|\mathbf{X},\theta) = \arg\max_\theta \log \mathcal{N}\left(\mathbf{y}|0, \mathbf{K}+\sigma^2\mathbf{I}\right)$$

- Prediction on a test point given the observed data and the optimized hyper-parameters.

$$p(\mathbf{f}_*|\mathbf{X}_*,\mathbf{y},\mathbf{X},\theta) = \mathcal{N}\left(\mathbf{f}_*|\mathbf{K}_*(\mathbf{K}+\sigma^2\mathbf{I})^{-1}\mathbf{y}, \mathbf{K}_{**}-\mathbf{K}_*(\mathbf{K}+\sigma^2\mathbf{I})^{-1}\mathbf{K}_*^\top\right)$$

How to implement the log-likelihood (1)

- Compute the covariance matrix $\mathbf{K}$:

$$\mathbf{K} = \begin{pmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \mathbf{x}_1) & \cdots & k(\mathbf{x}_N, \mathbf{x}_N) \end{pmatrix}$$

  where $k(\mathbf{x}_i, \mathbf{x}_j) = \gamma \exp\left(-\frac{1}{2l^2}(\mathbf{x}_i - \mathbf{x}_j)^\top (\mathbf{x}_i - \mathbf{x}_j)\right)$

- The complexity is $O(N^2 Q)$.

How to implement the log-likelihood (2)

- Plug in the log-pdf of multi-variate normal distribution:

$$\begin{aligned}\log p(\mathbf{y}|\mathbf{X}) &= \log \mathcal{N}\left(\mathbf{y}|0, \mathbf{K}+\sigma^2\mathbf{I}\right) \\ &= -\frac{1}{2}\log|2\pi(\mathbf{K}+\sigma^2\mathbf{I})| - \frac{1}{2}\mathbf{y}^\top(\mathbf{K}+\sigma^2\mathbf{I})^{-1}\mathbf{y} \\ &= -\frac{1}{2}(||\mathbf{L}^{-1}\mathbf{y}||^2 + N\log 2\pi) - \sum_i \log \mathbf{L}_{ii}\end{aligned}$$

- Take a Cholesky decomposition: $\mathbf{L} = \texttt{chol}(\mathbf{K}+\sigma^2\mathbf{I})$.
- The computational complexity is $O(N^3 + N^2 + N)$. Therefore, the overall complexity including the computation of $\mathbf{K}$ is $O(N^3)$.

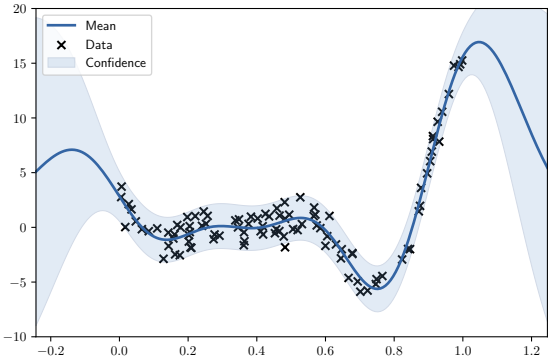
A quick profiling (N=1000, Q=10)

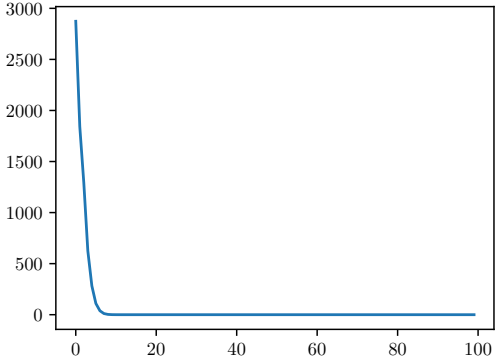
Time unit is microsecond.

```
Line #      Time    % Time      Line Contents
 2                          def log_likelihood(kern, X, Y, sigma2):
 3         6.0       0.0          N = X.shape[0]
 4     55595.0      58.7          K = kern.K(X)
 5      4369.0       4.6          Ky = K + np.eye(N)*sigma2
 6     30012.0      31.7          L = np.linalg.cholesky(Ky)
 7      4361.0       4.6          LinvY = dtrtrs(L, Y, lower=1)[0]
 8        49.0       0.1          logL = N*np.log(2*np.pi)/-2.
 9        82.0       0.1          logL += np.square(LinvY).sum()/-2.
10       208.0       0.2          logL += -np.log(np.diag(L)).sum()
11         2.0       0.0          return logL
```

Too slow or too many data points?

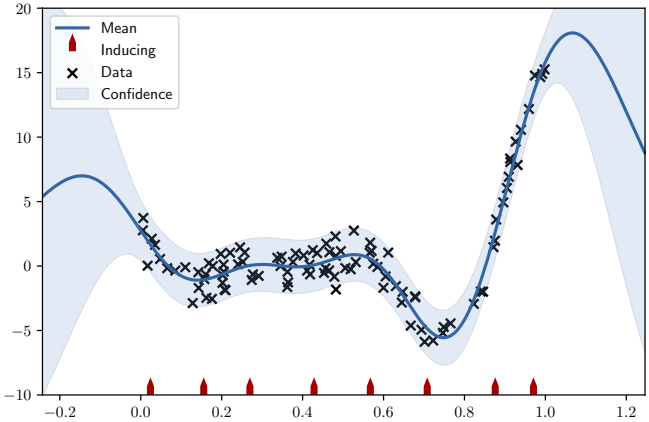
A lot of data does not necessarily mean a complex model.





Pseudo Data

Summarize *real* data into a small set of *pseudo* data.



Sparse Gaussian Process

Sparse GPs refers to a family of approximations:

- Nyström approximation [Williams and Seeger, 2001]
- Fully independent training conditional (FITC) [Snelson and Ghahramani, 2006]
- Variational sparse Gaussian process [Titsias, 2009]

Approximation by subset

- Let's randomly pick a subset from the training data: $\mathbf{Z} \in \mathbb{R}^{M \times Q}$.
- Approximate the covariance matrix $\mathbf{K}$ by $\tilde{\mathbf{K}}$.

$$\tilde{\mathbf{K}} = \mathbf{K}_z \mathbf{K}_{zz}^{-1} \mathbf{K}_z^\top, \text{ where } \mathbf{K}_z = \mathbf{K}(\mathbf{X}, \mathbf{Z}) \text{ and } \mathbf{K}_{zz} = \mathbf{K}(\mathbf{Z}, \mathbf{Z}).$$

- Note that $\tilde{\mathbf{K}} \in \mathbb{R}^{N \times N}$, $\mathbf{K}_z \in \mathbb{R}^{N \times M}$ and $\mathbf{K}_{zz} \in \mathbb{R}^{M \times M}$.
- The log-likelihood is approximated by

$$\log p(\mathbf{y}|\mathbf{X}, \theta) \approx \log \mathcal{N}\left(\mathbf{y}|0, \mathbf{K}_z \mathbf{K}_{zz}^{-1} \mathbf{K}_z^\top + \sigma^2 \mathbf{I}\right).$$

Efficient computation using Woodbury formula

- The naive formulation does not bring any computational benefits.

$$\tilde{\mathcal{L}} = -\frac{1}{2}\log|2\pi(\tilde{\mathbf{K}} + \sigma^2\mathbf{I})| - \frac{1}{2}\mathbf{y}^\top(\tilde{\mathbf{K}} + \sigma^2\mathbf{I})^{-1}\mathbf{y}$$

- Apply the Woodbury formula:

$$(\mathbf{K}_z\mathbf{K}_{zz}^{-1}\mathbf{K}_z^\top + \sigma^2\mathbf{I})^{-1} = \sigma^{-2}\mathbf{I} - \sigma^{-4}\mathbf{K}_z(\mathbf{K}_{zz} + \sigma^{-2}\mathbf{K}_z^\top\mathbf{K}_z)^{-1}\mathbf{K}_z^\top$$

- Note that $(\mathbf{K}_{zz} + \sigma^{-2}\mathbf{K}_z^\top\mathbf{K}_z) \in \mathbb{R}^{M\times M}$.
- The computational complexity reduces to $O(NM^2)$.

Nyström approximation

- The above approach is called Nyström approximation by Williams and Seeger [2001].
- The approximation is directly done on the covariance matrix without the concept of pseudo data.
- The approximation becomes exact if the whole data set is taken, *i.e.*, $\mathbf{K}\mathbf{K}^{-1}\mathbf{K}^{\top} = \mathbf{K}$.
- The subset selection is done randomly.

Gaussian process with Pseudo Data (1)

- Snelson and Ghahramani [2006] proposes the idea of having pseudo data. This approach is later referred to as Fully independent training conditional (FITC).
- Augment the training data $(\mathbf{X}, \mathbf{y})$ with pseudo data $\mathbf{u}$ at location $\mathbf{Z}$.

$$p\left(\begin{bmatrix}\mathbf{y}\\ \mathbf{u}\end{bmatrix} \mid \begin{bmatrix}\mathbf{X}\\ \mathbf{Z}\end{bmatrix}\right) = \mathcal{N}\left(\begin{bmatrix}\mathbf{y}\\ \mathbf{u}\end{bmatrix} | 0, \begin{bmatrix}\mathbf{K}_{ff} + \sigma^2\mathbf{I} & \mathbf{K}_{fu}\\ \mathbf{K}_{fu}^\top & \mathbf{K}_{uu}\end{bmatrix}\right)$$

where $\mathbf{K}_{ff} = \mathbf{K}(\mathbf{X}, \mathbf{X})$, $\mathbf{K}_{fu} = \mathbf{K}(\mathbf{X}, \mathbf{Z})$ and $\mathbf{K}_{uu} = \mathbf{K}(\mathbf{Z}, \mathbf{Z})$.

Gaussian process with Pseudo Data (2)

- Thanks to the marginalization property of Gaussian distribution,

$$p(\mathbf{y}|\mathbf{X}) = \int_{\mathbf{u}} p(\mathbf{y}, \mathbf{u}|\mathbf{X}, \mathbf{Z}).$$

- Further re-arrange the notation:

$$p(\mathbf{y}, \mathbf{u}|\mathbf{X}, \mathbf{Z}) = p(\mathbf{y}|\mathbf{u}, \mathbf{X}, \mathbf{Z})p(\mathbf{u}|\mathbf{Z})$$

where $p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{u}|0, \mathbf{K}_{uu})$,
$p(\mathbf{y}|\mathbf{u}, \mathbf{X}, \mathbf{Z}) = \mathcal{N}\left(\mathbf{y}|\mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{u}, \mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{fu}^{\top} + \sigma^2\mathbf{I}\right)$.

FITC approximation (1)

- So far, $p(\mathbf{y}|\mathbf{X})$ has not been changed, but there is no speed-up, $\mathbf{K}_{ff} \in \mathbb{R}^{N \times N}$ in $\mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{fu}^{\top} + \sigma^2\mathbf{I}$.
- The FITC approximation assumes

$$\tilde{p}(\mathbf{y}|\mathbf{u}, \mathbf{X}, \mathbf{Z}) = \mathcal{N}\left(\mathbf{y}|\mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{u}, \mathbf{\Lambda} + \sigma^2\mathbf{I}\right),$$

where $\mathbf{\Lambda} = (\mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{fu}^{\top}) \circ \mathbf{I}$.

FITC approximation (2)

- Marginalize $\mathbf{u}$ from the model definition:

$$\tilde{p}(\mathbf{y}|\mathbf{X},\mathbf{Z}) = \mathcal{N}\left(\mathbf{y}|0, \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{fu}^{\top} + \mathbf{\Lambda} + \sigma^2\mathbf{I}\right)$$

- Woodbury formula can be applied in the sam way as in Nyström approximation:

$$(\mathbf{K}_z\mathbf{K}_{zz}^{-1}\mathbf{K}_z^{\top} + \mathbf{\Lambda} + \sigma^2\mathbf{I})^{-1} = \mathbf{A} - \mathbf{A}\mathbf{K}_z(\mathbf{K}_{zz} + \mathbf{K}_z^{\top}\mathbf{A}\mathbf{K}_z)^{-1}\mathbf{K}_z^{\top}\mathbf{A},$$

where $\mathbf{A} = (\mathbf{\Lambda} + \sigma^2\mathbf{I})^{-1}$.

FITC approximation (3)

- FITC allows the pseudo data not being a subset of training data.
- The inducing inputs $\mathbf{Z}$ can be optimized via gradient optimization.
- Like Nyström approximation, when taking all the training data as inducing inputs, the FITC approximation is equivalent to the original GP:

$$\tilde{p}(\mathbf{y}|\mathbf{X}, \mathbf{Z} = \mathbf{X}) = \mathcal{N}\left(\mathbf{y}|0, \mathbf{K}_{ff} + \sigma^2\mathbf{I}\right)$$

- FITC can be combined easily with expectation propagation (EP). Bui et al. [2017] provides an overview and a nice connection with variational sparse GP.

Model Approximation vs. Approximate Inference

When the exact model/inference is intractable, typically there are two types of approaches:

- Approximate the original model with a simpler one such that inference becomes tractable, like Nyström approximation, FITC.
- Keep the original model but derive an approximate inference method which is often *not* able to return the true answer, like variational inference.

Model Approximation vs. Approximate Inference

A problem with model approximation is that

- when an approximated model requires some tuning, e.g., for hyper-parameters, it is unclear how to improve it based on training data.
- In the case of FITC, we know the model is correct if $\mathbf{Z} = \mathbf{X}$, however, optimizing $\mathbf{Z}$ will not necessarily lead to a better location.
- In fact, optimizing $\mathbf{Z}$ can lead to overfitting. [Quiñonero-Candela and Rasmussen, 2005]

Variational Sparse Gaussian Process (1)

- Titsias [2009] introduces a variational approach for sparse GP.
- It follows the same concept of pseudo data:

$$p(\mathbf{y}|\mathbf{X}) = \int_{\mathbf{f},\mathbf{u}} p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z})p(\mathbf{u}|\mathbf{Z})$$

where $p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{u}|0,\mathbf{K}_{uu})$,
$p(\mathbf{y}|\mathbf{u},\mathbf{X},\mathbf{Z}) = \mathcal{N}\left(\mathbf{y}|\mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{u}, \mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{fu}^{\top} + \sigma^2\mathbf{I}\right)$.

Variational Sparse Gaussian Process (2)

- Instead of approximate the model, Titsias [2009] derives a variational lower bound.
- Normally, a variational lower bound of a marginal likelihood, also known as evidence lower bound (ELBO), looks like

$$\begin{aligned}\log p(\mathbf{y}|\mathbf{X}) &= \log \int_{\mathbf{f},\mathbf{u}} p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z}) p(\mathbf{u}|\mathbf{Z}) \\ &\geq \int_{\mathbf{f},\mathbf{u}} q(\mathbf{f},\mathbf{u}) \log \frac{p(\mathbf{y}|\mathbf{f}) p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z}) p(\mathbf{u}|\mathbf{Z})}{q(\mathbf{f},\mathbf{u})}.\end{aligned}$$

Special Variational Posterior

- Titsias [2009] defines an unusual variational posterior:

$$q(\mathbf{f}, \mathbf{u}) = p(\mathbf{f}|\mathbf{u}, \mathbf{X}, \mathbf{Z})q(\mathbf{u}), \quad \text{where } q(\mathbf{u}) = \mathcal{N}(\mathbf{u}|\mu, \Sigma).$$

- Plug it into the lower bound:

$$\begin{aligned}
\mathcal{L} &= \int_{\mathbf{f},\mathbf{u}} p(\mathbf{f}|\mathbf{u}, \mathbf{X}, \mathbf{Z})q(\mathbf{u}) \log \frac{p(\mathbf{y}|\mathbf{f})\cancel{p(\mathbf{f}|\mathbf{u}, \mathbf{X}, \mathbf{Z})}p(\mathbf{u}|\mathbf{Z})}{\cancel{p(\mathbf{f}|\mathbf{u}, \mathbf{X}, \mathbf{Z})}q(\mathbf{u})} \\
&= \langle \log p(\mathbf{y}|\mathbf{f}) \rangle_{p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z})q(\mathbf{u})} - \mathrm{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u}|\mathbf{Z})\right) \\
&= \left\langle \log \mathcal{N}\left(\mathbf{y}|\mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{u}, \sigma^2\mathbf{I}\right) \right\rangle_{q(\mathbf{u})} - \mathrm{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u}|\mathbf{Z})\right)
\end{aligned}$$

Special Variational Posterior

- There is no inversion of any big covariance matrices in the first term:

$$-\frac{N}{2}\log 2\pi\sigma^2 - \frac{1}{2\sigma^2}\left\langle (\mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{u}-\mathbf{y})^\top (\mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{u}-\mathbf{y})\right\rangle_{q(\mathbf{u})}$$

- The overall complexity of the lower bound is $O(NM^2)$.

Tighten the Bound

- Find the optimal parameters of $q(\mathbf{u})$:

$$\mu^*, \Sigma^* = \arg\max_{\mu,\Sigma} \mathcal{L}(\mu, \Sigma).$$

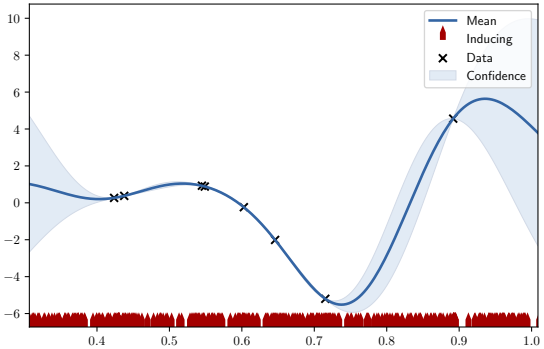
- Make the bound as tight as possible by plugging in μ^* and Σ^*:

$$\mathcal{L} = \log \mathcal{N}\left(\mathbf{y}|0, \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{fu}^\top + \sigma^2\mathbf{I}\right) - \frac{1}{2\sigma^2}\text{tr}\left(\mathbf{K}_{ff} - \mathbf{K}_{fu}\mathbf{K}_{uu}^{-1}\mathbf{K}_{fu}^\top\right).$$

- The overall complexity of the lower bound remains $O(NM^2)$.

Variational sparse GP

- Note that $\mathcal{L}$ is not a valid log-pdf, $\int_{\mathbf{y}} \exp(\mathcal{L}(\mathbf{y})) \leq 1$, due to the trace term.
- As inducing points are variational parameters, optimizing the inducing inputs $\mathbf{Z}$ always leads to a better bound.
- The model does not "overfit" with too many inducing points.



An alternative view of sparse GP

Is variational sparse GP a *hack* only working for GP?

The two key ingredients

The two key ingredients of variational sparse GP:

- Variational compression [Hensman and Lawrence, 2014]
- Variational posterior with "pseudo data" ("variational Gaussian process" by Tran et al. [2016])

A generic variational lower bound

Consider a generic probabilistic model $p(y|h)p(h|x)$ with h being a latent variable. A variational lower bound is typically like

$$\log p(y|x) \geq \int_h q(h) \log \frac{p(y|h)p(h|x)}{q(h)}$$

Use prior as posterior

We are free to choose the form of the variational posterior. It is always possible to choose $q(h) = p(h|x)$. This results into a lower bound:

$$\mathcal{L} = \int_h p(h|x) \log p(y|h).$$

The same idea can be easily applied to a deeper model:

$$\begin{aligned}\log p(y|x) &= \log \int_{h_1,h_2,h_3} p(y|h_1)p(h_1|h_2)p(h_2|h_3)p(h_3|x) \\ &\geq \int_{h_1,h_2,h_3} p(h_1|h_2)p(h_2|h_3)p(h_3|x) \log p(y|h_1)\end{aligned}$$

This is weird. Does it work?

Sigmoid Belief Networks

It works surprising well. Dai and Lawrence [2015] applied this trick to sigmoid belief networks (SBN):

$$p(\mathbf{y}|\mathbf{h}_1)\prod_{l=1}^{L-1}p(\mathbf{h}_l|\mathbf{h}_{l+1})p(\mathbf{h}_L),$$

where

$$\begin{aligned}
p(\mathbf{y}|\mathbf{h}_1) &= \prod_i \sigma(\mathbf{W}_{1,i}\mathbf{h}_1+\mathbf{b}_{1,i})^{y_i}\sigma(-\mathbf{W}_{1,i}\mathbf{h}_1-\mathbf{b}_{1,i})^{1-y_i},\\
p(\mathbf{h}_l|\mathbf{h}_{l+1}) &= \prod_i \sigma(\mathbf{W}_{l+1,i}\mathbf{h}_{l+1}+\mathbf{b}_{l+1,i})^{h_{l,i}}\sigma(-\mathbf{W}_{l+1,i}\mathbf{h}_{l+1}-\mathbf{b}_{l+1,i})^{1-h_{l,i}}\\
p(\mathbf{h}_L) &= \prod_i \pi_i^{h_{Li}}(1-\pi_i)^{1-h_{Li}}
\end{aligned}$$

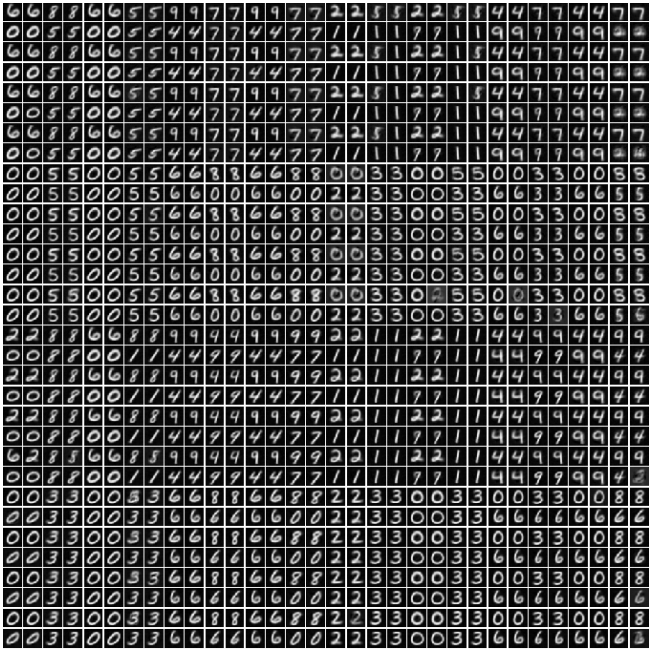
Sigmoid Belief Networks

Variational Inference of SBN is very hard [Mnih and Gregor, 2014]:

$$\log p(\mathbf{y}) \geq \sum_{\mathbf{h}_1,\ldots,\mathbf{h}_L} q(\mathbf{h}_1,\ldots,\mathbf{h}_L) \log \frac{p(\mathbf{y}|\mathbf{h}_1)\prod_{l=1}^{L-1} p(\mathbf{h}_l|\mathbf{h}_{l+1})p(\mathbf{h}_L)}{q(\mathbf{h}_1,\ldots,\mathbf{h}_L)}$$

$$\mathcal{L} = \sum_{\mathbf{h}_1,\ldots,\mathbf{h}_L} q(\mathbf{h}_L) \prod_{l=1}^{L-1} p(\mathbf{h}_l|\mathbf{h}_{l+1}) \log p(\mathbf{y}|\mathbf{h}_1)$$

- 3 hidden layers (100-100-10)
- the generated examples from each value in the top layer
- in total 1024 examples
- columns encode first 5 bits.
- rows encode later 5 bits.



What is the price?

The variational lower bound can also be written as

$$\log p(y|x) = \int_h q(h) \log \frac{p(y|h)p(h|x)}{q(h)} + \mathsf{KL}\left(q(h) \,\|\, p(h|x,y)\right)$$

With $q(h) = p(h|x)$,

$$\log p(y|x) = \int_h p(h|x) \log p(y|h) + \mathsf{KL}\left(p(h|x) \,\|\, p(h|x,y)\right)$$

What is the price?

The lower bound is exact *only if* the posterior is same as the prior.

$$\log p(y|x) = \int_h p(h|x) \log p(y|h) + \mathsf{KL}\left(p(h|x) \,\|\, p(h|x,y)\right)$$

$$p(h|x,y) = \frac{p(y|h)p(h|x)}{\int_{h'} p(y|h')p(h'|x)}$$

$\mathsf{KL}\left(p(h|x) \,\|\, p(h|x,y)\right)$ is zero, when

- y is independent of h, i.e., $p(y|h) = p(y)$.
- $p(h|x)$ is a deterministic relation, i.e., $p(h|x) = \delta(h(x))$.

A bias towards being a deterministic function

This variational posterior introduces a bias towards being a deterministic function.

Assume the model is parameterized by θ, i.e., $p(y|h,\theta)p(h|x,\theta)$.

Point Estimate:

$$\theta^* = \arg\max_\theta \mathcal{L}(\theta) = \arg\max_\theta \left(\log p(y|x,\theta) - \mathsf{KL}\left(p(h|x,\theta) \,\|\, p(h|x,y,\theta)\right)\right)$$

A bias towards being a deterministic function

Variational Inference:

$$\begin{aligned}\hat{\mathcal{L}} &= \int_{h,\theta} p(h|x,\theta)q(\theta)\log\frac{p(y|h,\theta)\cancel{p(h|x,\theta)}p(\theta)}{\cancel{p(h|x,\theta)}q(\theta)} \\ &= \langle \log p(y|x,\theta) - \mathsf{KL}\left(p(h|x,\theta) \,\|\, p(h|x,y,\theta)\right)\rangle_{q(\theta)} - \mathsf{KL}\left(q(\theta) \,\|\, p(\theta)\right) \\ &= \langle \log p(y|x,\theta)\rangle_{q(\theta)} - \mathsf{KL}\left(q(\theta) \,\|\, p(\theta)\right) \\ &\quad - \langle \mathsf{KL}\left(p(h|x,\theta) \,\|\, p(h|x,y,\theta)\right)\rangle_{q(\theta)}\end{aligned}$$

The bias in variational Sparse GP

Variational sparse GP often "under-fit".

$$
\begin{aligned}
\mathcal{L} &= \int_{\mathbf{f},\mathbf{u}} p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z}) q(\mathbf{u}) \log \frac{p(\mathbf{y}|\mathbf{f}) \cancel{p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z})} p(\mathbf{u}|\mathbf{Z})}{\cancel{p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z})} q(\mathbf{u})} \\
&= \int_{\mathbf{f},\mathbf{u}} p(\mathbf{f}|\mathbf{u},\mathbf{X},\mathbf{Z}) q(\mathbf{u}) \log p(\mathbf{y}|\mathbf{f}) - \mathsf{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u}|\mathbf{Z})\right) \\
&= \left\langle \log p(\mathbf{y}|\mathbf{X},\mathbf{u},\mathbf{Z}) \right\rangle_{q(\mathbf{u})} - \mathsf{KL}\left(q(\mathbf{u}) \,\|\, p(\mathbf{u}|\mathbf{Z})\right) \\
&\quad - \left\langle \mathsf{KL}\left(p(\mathbf{f}|\mathbf{X},\mathbf{u},\mathbf{Z}) \,\|\, p(\mathbf{f}|\mathbf{X},\mathbf{y},\mathbf{u},\mathbf{Z})\right) \right\rangle_{q(\mathbf{u})}
\end{aligned}
$$

Flexible variational posterior

Consider a generic probabilistic model, e.g., $p(\mathbf{y}|\mathbf{f})p(\mathbf{f})$.

Variational lower bound:

$$\mathcal{L} = \int_{\mathbf{f}} q(\mathbf{f}) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f})}{q(\mathbf{f})}$$

One flexible variational posterior:

$$q(\mathbf{f}) = \int_{\mathbf{f}} q(\mathbf{f}|\mathbf{u})q(\mathbf{u})$$

Auxiliary variable

It may not be tractable to compute $\int_{\mathbf{f}} q(\mathbf{f}|\mathbf{u})q(\mathbf{u})$.

One way to work around is to introduce $\mathbf{u}$ as an auxiliary variable to the model:

$$p(\mathbf{y}|\mathbf{f})p(\mathbf{f})p(\mathbf{u}|\mathbf{f})$$

Note that we are free to choose the form of $p(\mathbf{u}|\mathbf{f})$ and $q(\mathbf{u})$ and $q(\mathbf{f}|\mathbf{u})$.

A further lower bound

show the relation between two lower bounds.

$$\int_{\mathbf{f}} q(\mathbf{f}) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f})}{q(\mathbf{f})} = \int_{\mathbf{f},\mathbf{u}} q(\mathbf{u}|\mathbf{f})q(\mathbf{f}) \log \frac{p(\mathbf{y}|\mathbf{f})p(\mathbf{f})p(\mathbf{u}|\mathbf{f})}{q(\mathbf{u}|\mathbf{f})q(\mathbf{f})} - \int_{\mathbf{f}} q(\mathbf{f}) \int_{\mathbf{u}} q(\mathbf{u}|\mathbf{f}) \log \frac{p(\mathbf{u}|\mathbf{f})}{q(\mathbf{u}|\mathbf{f})}$$

$$\mathcal{L} = \mathcal{L}_{\mathbf{f},\mathbf{u}} - \langle \mathsf{KL}\left(q(\mathbf{u}|\mathbf{f}) \,\|\, p(\mathbf{u}|\mathbf{f})\right) \rangle_{q(\mathbf{f})} \geq \mathcal{L}_{\mathbf{f},\mathbf{u}}$$

In the case of spare GP

This leads back to the usual sparse GP bound that we know.

$$
\begin{aligned}
p(\mathbf{y}|\mathbf{f}) &= \mathcal{N}\left(\mathbf{y}|\mathbf{f}, \sigma^2\mathbf{I}\right) \\
p(\mathbf{f}|\mathbf{X}) &= \mathcal{N}\left(\mathbf{f}|0, \mathbf{K}(\mathbf{X},\mathbf{X})\right) \\
p(\mathbf{u}|\mathbf{Z},\mathbf{f},\mathbf{X}) &= \mathcal{N}\left(\mathbf{u}|\mathbf{K}_{uf}\mathbf{K}_{ff}^{-1}\mathbf{f}, \mathbf{K}_{uu} - \mathbf{K}_{fu}^{\top}\mathbf{K}_{ff}^{-1}\mathbf{K}_{fu}\right) \\
q(\mathbf{u}|\mathbf{f})q(\mathbf{f}) &= q(\mathbf{f}|\mathbf{u})q(\mathbf{u}) = p(\mathbf{f}|\mathbf{X},\mathbf{u},\mathbf{Z})q(\mathbf{u})
\end{aligned}
$$

Parallel Sparse Gaussian Process

- Beyond Approximate the inference method, maybe we could exploit parallelization.
- For Gaussian process, it turns out to be very hard, because parallel Cholesky decomposition is very difficult.
- Dai et al. [2014] and Gal et al. [2014] proposes a parallel inference method for sparse GP.

Data Parallelism

- Consider a training set: $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$.
- Assume there are C computational cores/machines.
- A data parallelism algorithm divides the data set into C partitions as evenly as possible: $\mathcal{D} = \bigcup_{c=1}^{C} \mathcal{D}_c$.
- The parallelism happens in the way that the function running on each core only requiring the data from the local partition.

A simple example: neural network regression

$$l = \sum_{n=1}^{N} ||y_n - f_\theta(\mathbf{x}_n)||^2 = \sum_{c=1}^{C} \sum_{n_c \in \mathcal{D}_c} ||y_{n_c} - f_\theta(\mathbf{x}_{n_c})||^2$$

1. Each core computes its local objective $l_c = \sum_{n_c \in \mathcal{D}_c} ||y_{n_c} - f_\theta(\mathbf{x}_{n_c})||^2$.
2. Each core computes the gradient of its local object $\partial l_c / \partial \theta$.
3. Aggregate all the local objectives and gradients $l = \sum_{c=1}^{C} l_c$ and $\partial l / \partial \theta = \sum_{c=1}^{C} \partial l_c / \partial \theta$.
4. Take a step along the gradient following a gradient descent algorithm.
5. Repeat Step 1 until converge.

Data Parallelism for Sparse GP

The variational lower bound (after applying Woodbury formula) is

$$\mathcal{L} = -\frac{N}{2}\log 2\pi\sigma^2 + \frac{1}{2}\log\frac{|\mathbf{K}_{uu}|}{|\mathbf{K}_{uu}+\sigma^{-2}\mathbf{\Phi}|} - \frac{1}{2\sigma^2}\mathbf{y}^\top\mathbf{y} + \frac{1}{2\sigma^4}\mathbf{y}^\top\mathbf{K}_{fu}(\mathbf{K}_{uu}+\mathbf{\Phi})^{-1}\mathbf{K}_{fu}^\top\mathbf{y} - \frac{1}{2\sigma^2}\phi + \frac{1}{2\sigma^2}\mathsf{tr}\left(\mathbf{K}_{uu}^{-1}\mathbf{\Phi}\right)$$

where $\mathbf{\Phi} = \mathbf{K}_{fu}^\top\mathbf{K}_{fu}$ and $\phi = \mathsf{tr}\left(\mathbf{K}_{ff}\right)$.

Data Parallelism for Sparse GP

- The lower bound is not fully distributable like in the simple example.
- All the terms involving data can be written as a sum across data points:

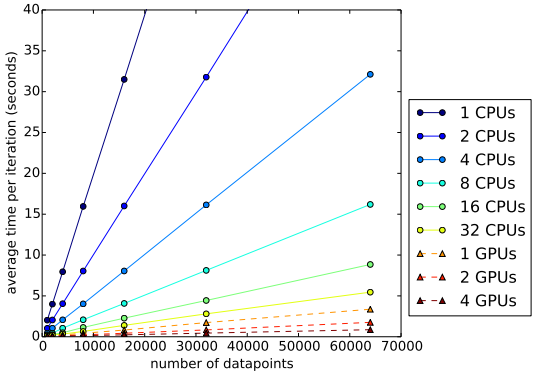
$$\mathbf{y}^\top \mathbf{y} = \sum_{n=1}^{N} y_n^2, \quad \mathbf{y}^\top \mathbf{K}_{fu} = \sum_{n=1}^{N} y_n \mathbf{K}_{f_n u}, \quad \mathbf{\Phi} = \sum_{n=1}^{N} \mathbf{K}_{f_n u}^\top \mathbf{K}_{f_n u}$$

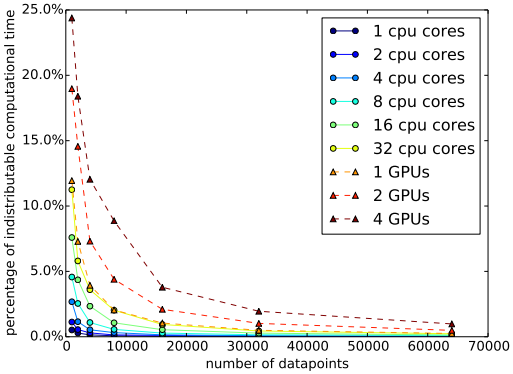
$$\phi = \sum_{n=1}^{N} \mathbf{K}_{f_n f_n}, \text{where } \mathbf{K}_{f_n u} = \mathbf{K}(\mathbf{x}_n, \mathbf{Z}), \quad \mathbf{K}_{f_n f_n} = \mathbf{K}(\mathbf{x}_n, \mathbf{x}_n).$$

Data Parallelism for Sparse GP

1. **[local]** Compute all the data related terms locally: $\mathbf{y}_c^\top \mathbf{y}_c$, $\mathbf{y}_c^\top \mathbf{K}_{f_c u}$, $\mathbf{\Phi}_c$ and ϕ_c.
2. **[global]** Aggregate all the local terms and compute the lower bound $\mathcal{L}$ on one node.
3. **[global]** Compute the gradient of the bound w.r.t. the model parameters.
4. **[global]** Compute the gradient w.r.t. the local terms $\partial \mathcal{L}/\partial \mathbf{K}_{f_c u}$, $\partial \mathcal{L}/\partial \mathbf{\Phi}_c$ and $\partial \mathcal{L}/\partial \phi_c$ and broadcast to individual nodes.
5. **[local]** Compute the gradient contribution of the local terms and aggregate the local gradients into the final gradient.
6. **[global]** Take a gradient step and repeat Step 1.

Data Parallelism for Sparse GP



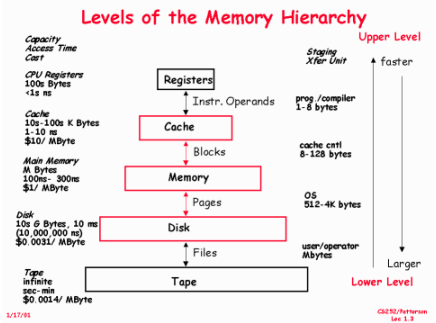


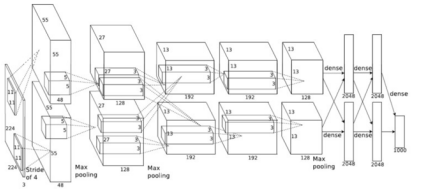
The emerge of deep learning platforms

- Deep learning platforms such as Theano, Tensorflow, Torch, Caffe, MXNet emerge in recent years.
- It standardizes deep neural networks programming.
- Auto-differentiation enables the flexible construction of DNNs.
- GPU acceleration enables scalability for real world applications.

GPU for machine learning

- Von Neumann architecture is not suitable for machine learning.
- Memory bandwidth:
  GPU(NVidia V100, AWS P3): 900 GB/s
  CPU (Intel Xeon E5-2660 v3, AWS C4): 68 GB/s





GTX 580 GPU has only 3GB of memory

Probabilistic Programming on deep learning platforms

- Edward `http://edwardlib.org`
- PyMC3 `https://github.com/pymc-devs/pymc3`
- pyprob `https://github.com/probprog/pyprob`
- Pyro `https://github.com/uber/pyro`

GP on deep learning platforms

GPflow `https://github.com/GPflow/GPflow`

GPyTorch `https://github.com/cornellius-gp/gpytorch`

MXFusion and GPy2

- Beyond GPU acceleration and auto-differentiation
- Use Gaussian process as a building block.
- MXFusion: modular probabilistic programming language
- GPy2 (a new interface based on MXFusion):
  - writing new kernel with auto-differentiation
  - scalable inference on GPU
  - Construct hybrid GP, deep GP, recurrent GP by re-using GP module with scalable approximate inference.

Acknowledgement.

James's blog: `https://www.prowler.io/blog/sparse-gps-approximate-the-posterior-not-the-model`

Thang D Bui, Josiah Yan, and Richard E Turner. A unifying framework for gaussian process pseudo-point approximations using power expectation propagation. *Journal of Machine Learning Research*, 18:3649–3720, 2017.

Zhenwen Dai and Neil D. Lawrence. Variational hierarchical community of experts. In *ICML Deep Learning workshop*, 2015.

Zhenwen Dai, Andreas Damianou, James Hensman, and Neil D. Lawrence. Gaussian process models with parallelization and gpu acceleration. In *NIPS workshop Software Engineering for Machine Learning*, 2014.

Yarin Gal, Mark van der Wilk, and Carl Edward Rasmussen. Distributed variational inference in sparse gaussian process regression and latent variable models. In *Advances in Neural Information Processing Systems 27*, pages 3257–3265, 2014.

James Hensman and Neil D. Lawrence. Nested variational compression in deep gaussian processes. *arXiv:1412.1370*, 2014.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, 2014.

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. A unifying view of sparse approximate gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005.

Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems*, pages 1257–1264. 2006.

Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, pages 567–574, 2009.

Dustin Tran, Rajesh Ranganath, and David M. Blei. The variational gaussian process. In *International Conference on Learning Representations*, 2016.

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