Gaussian Processes: large data and non-linear models

Richard E. Turner
University of Cambridge
Motivating application 1: Audio modelling

Audio time-series data

reconstruction using a GP model

$y(t)$

$T = 10^5 - 10^7$ datapoints
Motivating application 1: Audio modelling

How can we use GPs in this setting?
Motivating application 2: non-linear regression

<table>
<thead>
<tr>
<th>Dataset</th>
<th>N</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boston</td>
<td>506</td>
<td>13</td>
</tr>
<tr>
<td>Concrete</td>
<td>1030</td>
<td>8</td>
</tr>
<tr>
<td>Energy</td>
<td>768</td>
<td>8</td>
</tr>
<tr>
<td>Kin8nm</td>
<td>8192</td>
<td>8</td>
</tr>
<tr>
<td>Naval</td>
<td>11934</td>
<td>16</td>
</tr>
<tr>
<td>Power</td>
<td>9568</td>
<td>4</td>
</tr>
<tr>
<td>Protein</td>
<td>45730</td>
<td>9</td>
</tr>
<tr>
<td>Red Wine</td>
<td>1588</td>
<td>11</td>
</tr>
<tr>
<td>Yacht</td>
<td>308</td>
<td>6</td>
</tr>
<tr>
<td>Year</td>
<td>515345</td>
<td>90</td>
</tr>
</tbody>
</table>

**Average test log-likelihood/nats**

<table>
<thead>
<tr>
<th>Dataset</th>
<th>N</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boston</td>
<td>506</td>
<td>13</td>
</tr>
<tr>
<td>Concrete</td>
<td>1030</td>
<td>8</td>
</tr>
<tr>
<td>Energy</td>
<td>768</td>
<td>8</td>
</tr>
<tr>
<td>Kin8nm</td>
<td>8192</td>
<td>8</td>
</tr>
<tr>
<td>Naval</td>
<td>11934</td>
<td>16</td>
</tr>
<tr>
<td>Power</td>
<td>9568</td>
<td>4</td>
</tr>
<tr>
<td>Protein</td>
<td>45730</td>
<td>9</td>
</tr>
<tr>
<td>Red Wine</td>
<td>1588</td>
<td>11</td>
</tr>
<tr>
<td>Yacht</td>
<td>308</td>
<td>6</td>
</tr>
<tr>
<td>Year</td>
<td>515345</td>
<td>90</td>
</tr>
</tbody>
</table>
Motivating application 2: non-linear regression

![Graph showing average test log-likelihood/nats for various datasets and models.](image-url)
Motivating application 2: non-linear regression
Motivating application 2: non-linear regression

- **boston**
  - $N = 506$
  - $D = 13$
  - $-2.0$ to $3.2$

- **concrete**
  - $N = 1030$
  - $D = 8$
  - $-2.0$ to $3.0$

- **energy**
  - $N = 768$
  - $D = 8$
  - $-2.0$ to $2.6$

- **kin8nm**
  - $N = 8192$
  - $D = 8$
  - $1.6$ to $5.0$

- **naval**
  - $N = 11934$
  - $D = 16$
  - $-5.0$ to $7.0$

- **power**
  - $N = 9568$
  - $D = 4$
  - $-2.65$ to $-3.0$

- **protein**
  - $N = 45730$
  - $D = 9$
  - $-0.8$ to $0.0$

- **red wine**
  - $N = 1588$
  - $D = 11$
  - $-0.8$ to $-1.8$

- **yacht**
  - $N = 308$
  - $D = 6$
  - $-0.8$ to $-1.8$

- **year**
  - $N = 515345$
  - $D = 90$
  - $-3.0$ to $3.0$

---

BNN-deterministic  BNN-sampling  GP  DGP
Motivating application 2: non-linear regression

---

**boston**
- \( N = 506 \)
- \( D = 13 \)

**concrete**
- \( N = 1030 \)
- \( D = 8 \)

**energy**
- \( N = 768 \)
- \( D = 8 \)

**kin8nm**
- \( N = 8192 \)
- \( D = 8 \)

**naval**
- \( N = 11934 \)
- \( D = 16 \)

**power**
- \( N = 9568 \)
- \( D = 4 \)

**protein**
- \( N = 45730 \)
- \( D = 9 \)

**red wine**
- \( N = 1588 \)
- \( D = 11 \)

**yacht**
- \( N = 308 \)
- \( D = 6 \)

**year**
- \( N = 515345 \)
- \( D = 90 \)
Outline of the tutorial

- **An Introduction to GPs**
  - Mathematical foundations
  - Hyper-parameter learning
  - Covariance functions
  - Multi-dimensional inputs

- **Using GPs: Models, Applications and Connections**
  - Models and more on covariance functions
  - Applications
  - Connections

- **GPs for large data and non-linear models**
  - Scaling through pseudo-data: changing the generative model
  - Scaling through pseudo-data: variational Inference
  - General Approximate inference
Q1. What's the formal justification for how we were using GPs for regression?
Q1. What's the formal justification for how we were using GPs for regression?

generative model (like non-linear regression)

\[ y(x) = f(x) + \epsilon \sigma_y \]

\[ p(\epsilon) = \mathcal{N}(\epsilon; 0, 1) \]
Q1. What's the formal justification for how we were using GPs for regression?

Generative model (like non-linear regression)

\[ y(x) = f(x) + \epsilon \sigma_y \]

\[ p(\epsilon) = \mathcal{N}(\epsilon; 0, 1) \]

Place GP prior over the non-linear function

\[ p(f(x)|\theta) = \mathcal{GP}(f(x); 0, K_\theta(x, x')) \]

\[ K(x, x') = \sigma^2 \exp \left( -\frac{1}{2l^2} (x - x')^2 \right) \] (smoothly wiggling functions expected)
Q1. What's the formal justification for how we were using GPs for regression?

generative model (like non-linear regression)

\[ y(x) = f(x) + \epsilon \sigma_y \]

\[ p(\epsilon) = \mathcal{N}(\epsilon; 0, 1) \]

place GP prior over the non-linear function

\[ p(f(x)|\theta) = \mathcal{GP}(f(x); 0, K_\theta(x, x')) \]

\[ K(x, x') = \sigma^2 \exp \left( -\frac{1}{2l^2} (x - x')^2 \right) \quad \text{(smoothly wiggling functions expected)} \]

sum of Gaussian variables = Gaussian: induces a GP over \( y(x) \)

\[ p(y(x)|\theta) = \mathcal{GP}(y(x); 0, K_\theta(x, x') + \sigma_y^2) \]
Q4. How do we make predictions?

\[
p(y_1, y_2) = \mathcal{N} \left( \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} ; \begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \right)
\]

\[
p(y_1 | y_2) = \frac{p(y_1, y_2)}{p(y_2)}
\]

\[
\implies p(y_1 | y_2) = \mathcal{N} \left( y_1 ; a + BC^{-1}(y_2 - b), A - BC^{-1}B^T \right)
\]

**predictive mean**

\[
\mu_{y_1|y_2} = a + BC^{-1}(y_2 - b)
\]

\[= BC^{-1}y_2\]

\[= Wy_2\]
Q4. How do we make predictions?

\[
p(y_1, y_2) = \mathcal{N} \left( \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} ; \begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} A & B \\ B^T & C \end{bmatrix} \right)
\]

\[
p(y_1 | y_2) = \frac{p(y_1, y_2)}{p(y_2)}
\]

\[\implies p(y_1 | y_2) = \mathcal{N}(y_1; a + BC^{-1}(y_2 - b), A - BC^{-1}B^T)\]

**predictive mean**

\[
\mu_{y_1 | y_2} = a + BC^{-1}(y_2 - b)
\]

\[= BC^{-1}y_2\]

\[= Wy_2\]

linear in the data
Q4. How do we make predictions?

\[ p(y_1, y_2) = \mathcal{N} \left( \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}; \begin{bmatrix} a \\ b \end{bmatrix}, \begin{bmatrix} A & B \\ B^\top & C \end{bmatrix} \right) \]

\[ p(y_1 | y_2) = \frac{p(y_1, y_2)}{p(y_2)} \]

\[ \Rightarrow p(y_1 | y_2) = \mathcal{N}(y_1; a + BC^{-1}(y_2 - b), A - BC^{-1}B^\top) \]

**predictive mean**

\[ \mu_{y_1 | y_2} = a + BC^{-1}(y_2 - b) \]

\[ = BC^{-1}y_2 \]

\[ = W y_2 \]

linear in the data

**predictive covariance**

\[ \Sigma_{y_1 | y_2} = A - BC^{-1}B^\top \]

predictions more confident than prior

prior uncertainty

predictive uncertainty

reduction in uncertainty

linear in the data

predictions more confident than prior
Motivation: Gaussian Process Regression

\[ \mathbf{y} = \{y_n\}_{n=1}^N \]

\[ \mathbf{x} = \{x_n\}_{n=1}^N \]
Motivation: Gaussian Process Regression

\[ y = \{y_n\}_{n=1}^N \]

\[ x = \{x_n\}_{n=1}^N \]
Motivation: Gaussian Process Regression

\[ p(f|\theta) = \mathcal{GP}(f; 0, K_{\theta}) \]

\[ p(y_n|f, x_n, \theta) \]

outputs

\[ y = \{y_n\}_{n=1}^{N} \]

inputs

\[ x = \{x_n\}_{n=1}^{N} \]
Motivation: Gaussian Process Regression

\[ p(f|\theta) = \mathcal{GP}(f; 0, K_\theta) \]

\[ p(y_n|f, x_n, \theta) \]

\[ p(f|y, x, \theta) \]

\[ p(y|x, \theta) \]

inputs \[ x = \{x_n\}_{n=1}^N \]

outputs \[ y = \{y_n\}_{n=1}^N \]
Motivation: Gaussian Process Regression

\[ p(f|\theta) = \mathcal{GP}(f; 0, K_\theta) \]

\[ p(y_n|f, x_n, \theta) \]

\[ p(f|y, x, \theta) \]

\[ p(y|x, \theta) \]

\( y = \{y_n\}_{n=1}^N \)

\( x = \{x_n\}_{n=1}^N \)
Motivation: Gaussian Process Regression

\[ p(f|\theta) = \mathcal{GP}(f; 0, K_\theta) \]

\[ p(y_n|f, x_n, \theta) \]

\[ p(f|y, x, \theta) \]

\[ p(y|x, \theta) \]

Inference & learning

Intractabilities

Computational \( O(N^3) \)

Analytic

Outputs

\[ y = \{y_n\}_{n=1}^N \]

Ideas: summarise dataset by small number (M) pseudo-data

Inputs

\[ x = \{x_n\}_{n=1}^N \]
A Brief History of Gaussian Process Approximations

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

approximate generative model
exact inference

\[ \text{div}[p(f, y) || q(f, y)] \]

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs"
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

approximate generative model
exact inference
methods employing
pseudo-data

\text{div}[p(f, y)||q(f, y)]

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

- approximate generative model
- exact inference
- methods employing pseudo-data

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

approximate generative model
exact inference
methods employing pseudo-data

\[ \text{div} [p(f, y) || q(f, y)] \]

A Unifying View of Sparse Approximate Gaussian Process Regression
Quinonero-Candela & Rasmussen, 2005
(FITC, PITC, DTC)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

approximate generative model
exact inference

methods employing pseudo-data

exact generative model
approximate inference

\[ \text{div}[p(f, y) || q(f, y)] \]

A Unifying View of Sparse Approximate Gaussian Process Regression
Quinonero-Candela & Rasmussen, 2005 (FITC, PITC, DTC)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

approximate generative model
exact inference

methods employing
pseudo-data

exact generative model
approximate inference

div[p(f, y)||q(f, y)]

A Unifying View of Sparse Approximate Gaussian Process Regression
Quinonero-Candela & Rasmussen, 2005
(FITC, PITC, DTC)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

approximate generative model  methods employing pseudo-data  exact generative model

approximate inference

div[p(f, y)||q(f, y)]

div[p(f|y)||q(f)]

A Unifying View of Sparse Approximate Gaussian Process Regression Quinonero-Candela & Rasmussen, 2005 (FITC, PITC, DTC)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
A Brief History of Gaussian Process Approximations

- **FITC**: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
- **PITC**: Snelson et al. “Local and global sparse Gaussian process approximations”
- **VFE**: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
- **DTC / PP**: Seeger et al. “Fast Forward Selection to Speed Up Sparse Gaussian Process Regression”

**Methods Employing Pseudo-data**
- **FITC**
- **PITC**
- **DTC**

**Exact Generative Model and Inference**
- **approximate generative model**
- **exact inference**

**A Unifying View of Sparse Approximate Gaussian Process Regression**
Quinonero-Candela & Rasmussen, 2005
(FITC, PITC, DTC)

**A Unifying Framework for Sparse Gaussian Process Approximation using Power Expectation Propagation**
Bui, Yan and Turner, 2016
(VFE, EP, FITC, PITC ...)

*div[P(f,y)||q(f,y)]*
Factor Graphs: reminder (or introduction)

factor graph examples

\[ p(x_1, x_2, x_3) = g(x_1, x_2, x_3) \]

\[ p(x_1, x_2, x_3) = g_1(x_1, x_2)g_2(x_2, x_3) \]

\[ x_3 \perp x_1 | x_2 \]
Factor Graphs: reminder (or introduction)

factor graph examples

\[ p(x_1, x_2, x_3) = g(x_1, x_2, x_3) \]

\[ p(x_1, x_2, x_3) = g_1(x_1, x_2)g_2(x_2, x_3) \]

what is the minimal factor graph for this multivariate Gaussian?

\[ p(x | \mu, \Sigma) = \mathcal{N}(x; \mu, \Sigma) \]

\[ \Sigma = \begin{bmatrix} 1 & 1/2 & 1/2 & 1/4 \\ 1/2 & 5/4 & 1/4 & 1/8 \\ 1/2 & 1/4 & 5/4 & 5/8 \\ 1/4 & 1/8 & 5/8 & 21/16 \end{bmatrix} \quad \Sigma^{-1} = \begin{bmatrix} 1.5 & -1/2 & -1/2 & 0 \\ -1/2 & 1 & 0 & 0 \\ -1/2 & 0 & 5/4 & -1/2 \\ 0 & 0 & -1/2 & 1 \end{bmatrix} \]
Factor Graphs: reminder (or introduction)

factor graph examples

\[ p(x_1, x_2, x_3) = g(x_1, x_2, x_3) \]

\[ p(x_1, x_2, x_3) = g_1(x_1, x_2)g_2(x_2, x_3) \]

what is the minimal factor graph for this multivariate Gaussian?

\[ p(x|\mu, \Sigma) = \mathcal{N}(x; \mu, \Sigma) \]

4 dimensional

\[
\Sigma = \begin{bmatrix}
1 & 1/2 & 1/2 & 1/4 \\
1/2 & 5/4 & 1/4 & 1/8 \\
1/2 & 1/4 & 5/4 & 5/8 \\
1/4 & 1/8 & 5/8 & 21/16 \\
\end{bmatrix}
\]

\[
\Sigma^{-1} = \begin{bmatrix}
1.5 & -1/2 & -1/2 & 0 \\
-1/2 & 1 & 0 & 0 \\
-1/2 & 0 & 5/4 & -1/2 \\
0 & 0 & -1/2 & 1 \\
\end{bmatrix}
\]

solution:
A brief introduction to the Kullback-Leibler divergence

\[ \mathcal{KL}(p_1(z) \| p_2(z)) = \sum_z p_1(z) \log \frac{p_1(z)}{p_2(z)} \]

Important properties:
- **Gibb’s inequality**: \( \mathcal{KL}(p_1(z) \| p_2(z)) \geq 0 \), equality at \( p_1(z) = p_2(z) \)
  - proof via Jensen’s inequality or differentiation (see slide at end)
- **Non-symmetric**: \( \mathcal{KL}(p_1(z) \| p_2(z)) \neq \mathcal{KL}(p_2(z) \| p_1(z)) \)
  - hence named divergence and not distance

Example:
- binary variables \( z \in \{0, 1\} \)
- \( p(z = 1) = 0.8 \) and \( q(z = 1) = \rho \)
Fully independent training conditional (FITC) approximation

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
calibrated to original
Fully independent training conditional (FITC) approximation

1. augment model with $M<T$ pseudo data

$$p(f, u) = \mathcal{N}\left( \begin{bmatrix} f \\ u \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fu} \\ K_{uf} & K_{uu} \end{bmatrix} \right)$$

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
calibrated to original
1. augment model with $M<T$ pseudo data

$$p(f, u) = \mathcal{N}\left(\begin{bmatrix} f \\ u \end{bmatrix} ; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fu} \\
K_{uf} & K_{uu} \end{bmatrix}\right)$$

2. remove some of the dependencies
   (results in simpler model)

all factors

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
calibrated to original
1. augment model with $M<T$ pseudo data

$$p(f, u) = \mathcal{N}\left( \begin{bmatrix} f \\ u \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fu} \\ K_{uf} & K_{uu} \end{bmatrix} \right)$$

2. remove some of the dependencies
   (results in simpler model)

all factors

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
calibrated to original
Fully independent training conditional (FITC) approximation

1. augment model with $M<T$ pseudo data

$$p(f, u) = \mathcal{N}\left(\begin{bmatrix} f \\ u \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fu} \\ K_{uf} & K_{uu} \end{bmatrix}\right)$$

2. remove some of the dependencies (results in simpler model)

3. calibrate model (e.g. using KL divergence, many choices)

$$\arg \min_{q(u), \{q(f_t|u)\}_{t=1}^T} \text{KL}(p(f, u)||q(u) \prod_{t=1}^T q(f_t|u)) \implies q(u) = p(u), q(f_t|u) = p(f_t|u)$$

equal to exact conditionals

construct new generative model (with pseudo-data) cheaper to perform exact learning and inference calibrated to original
Fully independent training conditional (FITC) approximation

1. augment model with M<T pseudo data

\[ p(f, u) = \mathcal{N}\left( \begin{bmatrix} f \\ u \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \\ K_{ff} & K_{fu} \\ K_{uf} & K_{uu} \end{bmatrix} \right) \]

2. remove some of the dependencies (results in simpler model)

\[ f_i \leftrightarrow f_j \quad f_i \quad f_j \quad \text{all factors} \]

3. calibrate model (e.g. using KL divergence, many choices)

\[ \arg\min_{q(u), \{q(f_t|u)\}_{t=1}^T} \text{KL}(p(f, u)||q(u) \prod_{t=1}^T q(f_t|u)) \quad \Rightarrow \quad q(u) = p(u) \quad q(f_t|u) = p(f_t|u) \]

equal to exact conditionals

construct new generative model (with pseudo-data) cheaper to perform exact learning and inference calibrated to original indirect posterior approximation
Fully independent training conditional (FITC) approximation

construct new generative model (with pseudo-data) cheaper to perform exact learning and inference calibrated to original indirect posterior approximation
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]

\[ q(f_t | u) = p(f_t | u) \]
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]

\[ q(f_t | u) = p(f_t | u) \]

How do we make predictions?

\[ p(y_1 | y_2) = \mathcal{N}(y_1; \Sigma^{-1}_{12} \Sigma^{-1}_{22} y_2, \Sigma_{11} - \Sigma_{12} \Sigma^{-1}_{22} \Sigma_{12}^T) \]

Construct new generative model (with pseudo-data)

Cheaper to perform exact learning and inference

Calibrated to original indirect posterior approximation

Indirect posterior approximation
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]
\[ q(f_t|u) = p(f_t|u) = \mathcal{N}(f_t; K_{fu}K_{uu}^{-1}u, K_{ff} - K_{fu}K_{uu}^{-1}K_{uf}) \]

How do we make predictions?
\[ p(y_1|y_2) = \mathcal{N}(y_1; \Sigma_{12}\Sigma_{22}^{-1}y_2, \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{12}^T) \]

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
indirect posterior approximation
calibrated to original
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]

\[ q(f_t | u) = p(f_t | u) = \mathcal{N}(f_t; K_{f_t u} K_{uu}^{-1} u, K_{f_t f_t} - K_{f_t u} K_{uu}^{-1} K_{uf_t}) \]

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
indirect posterior approximation
calibrated to original
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]
\[ q(f_t | u) = p(f_t | u) \]
\[ = \mathcal{N}(f_t; K_{f_t u} K_{uu}^{-1} u, K_{f_t f_t} - K_{f_t u} K_{uu}^{-1} K_{uf_t}) \]

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
indirect posterior approximation
calibrated to original
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]
\[ q(f_t | u) = p(f_t | u) = \mathcal{N}(f_t; K_{fu}K_{uu}^{-1}u, K_{ff_t} - K_{fu}K_{uu}^{-1}K_{uf_t}) \]
\[ q(y_t | f_t) = p(y_t | f_t) = \mathcal{N}(y_t; f_t, \sigma_y^2) \]

construct new generative model (with pseudo-data)
cheaper to perform exact learning and inference
indirect
posterior
approximation
calibrated to original
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]

\[ q(f_t | u) = p(f_t | u) = \mathcal{N}(f_t; K_{f_t u} K_{uu}^{-1} u, K_{f_t f_t} - K_{f_t u} K_{uu}^{-1} K_{uf_t}) \]

\[ q(y_t | f_t) = p(y_t | f_t) = \mathcal{N}(y_t; f_t, \sigma_y^2) \]

cost of computing likelihood is \( \mathcal{O}(TM^2) \)

construct new generative model (with pseudo-data) cheaper to perform exact learning and inference calibrated to original indirect posterior approximation
Fully independent training conditional (FITC) approximation

\[ q(\mathbf{u}) = p(\mathbf{u}) = \mathcal{N}(\mathbf{u}; 0, K_{uu}) \]

\[ q(\mathbf{f}_t | \mathbf{u}) = p(\mathbf{f}_t | \mathbf{u}) \]

\[ = \mathcal{N}(\mathbf{f}_t; K_{f_t u} K_{uu}^{-1} \mathbf{u}, K_{f_t f_t} - K_{f_t u} K_{uu}^{-1} K_{u f_t}) \]

\[ q(\mathbf{y}_t | \mathbf{f}_t) = p(\mathbf{y}_t | \mathbf{f}_t) = \mathcal{N}(\mathbf{y}_t; \mathbf{f}_t, \sigma_y^2) \]

Cost of computing likelihood is \( \mathcal{O}(TM^2) \)

\[ p(\mathbf{y}_t | \theta) = \mathcal{N}(\mathbf{y}; 0, K_{fu} K_{uu}^{-1} K_{uu} K_{uu}^{-1} K_{uf} + D + \sigma_y^2 I) \]

cost of computing likelihood is \( \mathcal{O}(TM^2) \)

Construct new generative model (with pseudo-data) cheaper to perform exact learning and inference calibrated to original indirect posterior approximation
Fully independent training conditional (FITC) approximation

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}) \]

\[ q(f_t|u) = p(f_t|u) \]

\[ = \mathcal{N}(f_t; K_{f_t u} K_{u u}^{-1} u, K_{f_t f_t} - K_{f_t u} K_{u u}^{-1} K_{u f_t}) \]

\[ D_{tt} \]

\[ q(y_t|f_t) = p(y_t|f_t) = \mathcal{N}(y_t; f_t, \sigma_y^2) \]

cost of computing likelihood is \( O(TM^2) \)

\[ p(y_t|\theta) = \mathcal{N}(y; 0, K_{f u} K_{u u}^{-1} K_{u u} K_{u f} + D + \sigma_y^2 I) \]

\[ = \mathcal{N}(y; 0, K_{f u} K_{u u}^{-1} K_{u f} + D + \sigma_y^2 I) \]

construct new generative model (with pseudo-data) cheaper to perform exact learning and inference calibrated to original indirect posterior approximation
Fully independent training conditional (FITC) approximation

\[ q(\mathbf{u}) = p(\mathbf{u}) = \mathcal{N}(\mathbf{u}; 0, K_{uu}) \]
\[ q(f_t | \mathbf{u}) = p(f_t | \mathbf{u}) = \mathcal{N}(f_t; K_{f_t u}K_{uu}^{-1}\mathbf{u}, K_{f_t f_t} - K_{f_t u}K_{uu}^{-1}K_{uf_t}) \]

\[ q(y_t | f_t) = p(y_t | f_t) = \mathcal{N}(y_t; f_t, \sigma_y^2) \]

Cost of computing likelihood is \( \mathcal{O}(TM^2) \)

\[ p(y_t | \theta) = \mathcal{N}(\mathbf{y}; 0, K_{f_u}K_{uu}^{-1}K_{uu}K_{uf}^{-1}K_{uf} + D + \sigma_y^2 I) \]

Original variances along diagonal: stops variances collapsing

Construct new generative model (with pseudo-data) cheaper to perform exact learning and inference calibrated to original indirect posterior approximation
Initialize adversarially:

- amplitude and lengthscale too big
- noise too small
- pseudo-inputs bunched up
Pseudo-inputs and hyperparameters optimized
Fully independent training conditional (FITC) approximation

- introduces parametric bottleneck into non-parametric model (although in a clever way)
- if I see more data, should I add extra pseudo-data?
  - unnatural from a generative modelling perspective
  - natural from a prediction perspective (posterior gets more complex)
  \[\Rightarrow\] lost elegant separation of model, inference and approximation
- example of prior approximation

Extensions:
- methods for optimising pseudo-inputs (indirect approximations tend to over-fit)
- partially independent training conditional and tree-structured approximations (see extra slides)
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \, p(y, f|\theta) \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \, p(y, f|\theta) \]

\[ = \log \int df \, p(y, f|\theta) \frac{q(f)}{q(f)} \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(f|y, \theta)p(y|\theta)}{q(f)} \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(f|y, \theta)p(y|\theta)}{q(f)} = \log p(y|\theta) - \text{KL}(q(f)||p(f|y)) \]

KL between stochastic processes
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(f|y, \theta)p(y|\theta)}{q(f)} = \log p(y|\theta) - KL(q(f)||p(f|y)) \]

assume approximate posterior factorisation with special form

\[ q(f) = q(u, f\neq u) = q(f\neq u|u)q(u) = p(f\neq u|u)q(u) \]

exact: \[ q(f\neq u|u) = p(f\neq u|y, u) \]
Variational free-energy method (VFE)

\[ \mathcal{F}(\theta) = \log p(y|\theta) - \text{KL}(q(f) || p(f|y)) \]

- approximate posterior: 
  \[ q(f) = p(f \neq u|u)q(u) \]

- true posterior: 
  \[ p(f|y) \]
Variational free-energy method (VFE)

\[ \mathcal{F}(\theta) = \log p(\mathbf{y}|\theta) - \text{KL}(q(f)||p(f|\mathbf{y})) \]

approximate posterior

\[ q(f) = p(f \neq \mathbf{u}|\mathbf{u})q(\mathbf{u}) \]

true posterior

\[ p(f|\mathbf{y}) \]

same form as prediction from GP-regression
Variational free-energy method (VFE)

\[
\mathcal{F}(\theta) = \log p(y|\theta) - \text{KL}(q(f)||p(f|y))
\]

approximate posterior

\[
q(f) = p(f \neq u|u)q(u)
\]

true posterior

\[
p(f|y)
\]

inputs locations of 'pseudo' data

output locations and covariance 'pseudo' data

optimise variational free-energy wrt to these variational parameters
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \, p(y, f|\theta) \]

\[ = \log \int df \, p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \, q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]

\[ \mathcal{F}(\theta) = \int df \, q(f) \log \frac{p(f|y, \theta)p(y|\theta)}{q(f)} = \log p(y|\theta) - KL(q(f)||p(f|y)) \]

assume approximate posterior factorisation with special form

\[ q(f) = q(u, f \neq u) = q(f \neq u|u)q(u) = p(f \neq u|u)q(u) \]

exact: \[ q(f \neq u|u) = p(f \neq u|y, u) \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(f|y, \theta)p(y|\theta)}{q(f)} = \log p(y|\theta) - KL(q(f)||p(f|y)) \]

assume approximate posterior factorisation with special form

\[ q(f) = q(u, f\neq u) = q(f\neq u|u)q(u) = p(f\neq u|u)q(u) \]

exact: \[ q(f\neq u|u) = p(f\neq u|y, u) \]

plug into Free-energy:

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(y, f|\theta)}{p(f\neq u|u)q(u)} \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(f|y, \theta)p(y|\theta)}{q(f)} = \log p(y|\theta) - \text{KL}(q(f)||p(f|y)) \]

assume approximate posterior factorisation with special form

\[ q(f) = q(u, f\neq u) = q(f\neq u|u)q(u) = p(f\neq u|u)q(u) \quad \text{predictive from GP regression} \]

exact: \[ q(f\neq u|u) = p(f\neq u|y, u) \]

plug into Free-energy:

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(y, f|\theta)}{p(f\neq u|u)q(u)} = \int df \ q(f) \log \frac{p(y|f, \theta)p(f\neq u|u)p(u)}{p(f\neq u|u)q(u)} \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{L}(\theta) = \log p(y|\theta) = \log \int df \ p(y, f|\theta) \]

\[ = \log \int df \ p(y, f|\theta) \frac{q(f)}{q(f)} \geq \int df \ q(f) \log \frac{p(y, f|\theta)}{q(f)} = \mathcal{F}(\theta) \]

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(f|y, \theta)p(y|\theta)}{q(f)} = \log p(y|\theta) - KL(q(f)||p(f|y)) \]

assumption approximate posterior factorisation with special form

\[ q(f) = q(u, f\neq u) = q(f\neq u|u)q(u) = p(f\neq u|u)q(u) \]

exact: \[ q(f\neq u|u) = p(f\neq u|y, u) \]

plug into Free-energy:

\[ \mathcal{F}(\theta) = \int df \ q(f) \log \frac{p(y, f|\theta)}{p(f\neq u|u)q(u)} = \int df \ q(f) \log \frac{p(y|f, \theta)p(f\neq u|u)p(u)}{p(f\neq u|u)q(u)} \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{F}(\theta) = \int df \; q(f) \log \frac{p(y, f|\theta)}{p(f \neq u|u)q(u)} = \int df \; q(f) \log \frac{p(y|f, \theta)p(f \neq u|u)p(u)}{p(f \neq u|u)q(u)} \]

where \( q(f) = q(u, f \neq u) = q(f \neq u|u)q(u) = p(f \neq u|u)q(u) \)
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{F}(\theta) = \int df \, q(f) \log \frac{p(y, f|\theta)}{p(f \neq u|u)q(u)} = \int df \, q(f) \log \frac{p(y|f, \theta)p(f \neq u|u)p(u)}{p(f \neq u|u)q(u)} \]

where \( q(f) = q(u, f \neq u) = q(f \neq u|u)q(u) = p(f \neq u|u)q(u) \)

\[ \mathcal{F}(\theta) = \langle \log p(y|f, \theta) \rangle_{q(f)} - \text{KL}(q(u)||p(u)) \]

\[ \text{average of quadratic form} \quad \text{KL between two multivariate Gaussians} \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{F}(\theta) = \int df \, q(f) \log \frac{p(y, f|\theta)}{p(f \neq u|u)q(u)} = \int df \, q(f) \log \frac{p(y|f, \theta)p(f \neq u|u)p(u)}{p(f \neq u|u)q(u)} \]

where \( q(f) = q(u, f \neq u) = q(f \neq u|u)q(u) = p(f \neq u|u)q(u) \)

\[ \mathcal{F}(\theta) = \langle \log p(y|f, \theta) \rangle_{q(f)} - KL(q(u)||p(u)) \]

average of quadratic form

KL between two multivariate Gaussians

make bound as tight as possible:

\[ q^*(u) = \arg \max_{q(u)} \mathcal{F}(q, \theta) \]
Variational free-energy method (VFE)

lower bound the likelihood

\[ \mathcal{F}(\theta) = \int df \; q(f) \log \frac{p(y, f | \theta)}{p(f \neq u | u)q(u)} = \int df \; q(f) \log \frac{p(y | f, \theta)p(f \neq u | u)p(u)}{p(f \neq u | u)q(u)} \]

where \( q(f) = q(u, f \neq u) = q(f \neq u | u)q(u) = p(f \neq u | u)q(u) \)

\[ \mathcal{F}(\theta) = \langle \log p(y | f, \theta) \rangle_{q(f)} - \text{KL}(q(u) || p(u)) \]

average of quadratic form

KL between two multivariate Gaussians

make bound as tight as possible:

\[ q^*(u) = \arg \max_{q(u)} \mathcal{F}(q, \theta) \]

\[ q^*(u) \propto p(u)\mathcal{N}(y; K_{fu}K_{uu}^{-1}u, \sigma_y^2I) \quad \text{(DTC)} \]
Variational free-energy method (VFE)

lower bound the likelihood

$$\mathcal{F}(\theta) = \int df \, q(f) \log \frac{p(y, f|\theta)}{p(f \neq u|u)q(u)} = \int df \, q(f) \log \frac{p(y|f, \theta)p(f \neq u|u)p(u)}{p(f \neq u|u)q(u)}$$

where \( q(f) = q(u, f \neq u) = q(f \neq u|u)q(u) = p(f \neq u|u)q(u) \)

$$\mathcal{F}(\theta) = \langle \log p(y|f, \theta) \rangle_{q(f)} - KL(q(u)||p(u))$$

make bound as tight as possible: \( q^*(u) = \arg \max_{q(u)} \mathcal{F}(q, \theta) \)

\( q^*(u) \propto p(u)N(y; K_{fu}K_{uu}^{-1}u, \sigma_y^2 I) \) (DTC)

$$\mathcal{F}(q^*, \theta) = \log N(y; 0, K_{fu}K_{uu}^{-1}K_{uf}, \sigma_y^2 I) - \frac{1}{2\sigma_y^2} \text{trace}(K_{ff} - K_{fu}K_{uu}^{-1}K_{uf})$$

DTC like uncertainty based correction
Summary of VFE method

- optimisation pseudo point inputs **better behaved** in VFE methods (direct posterior approximation)
- variational methods known to **underfit** (and have other **biases**)
- no augmentation required: target is posterior over functions, which includes inducing variables
  - pseudo-input locations are pure variational parameters (do not parameterise the generative model)
  - coherent way of adding pseudo-data: more complex posteriors require more computational resources (more pseudo-points)
- Curious observation:
  - **VFE** returns better mean estimates
  - **FITC** returns better error-bar estimates
- **how should we select** $M = \text{number of pseudo-points}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M =$ number of pseudo-data?

![Graph showing the relationship between compute time and SMSE with two data sets: Exact and VFE. The graph illustrates the comparison between the two methods over a range of compute times.]
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select \( M = \) number of pseudo-data?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M = \text{number of pseudo-data}$?
How do we select $M =$ number of pseudo-data?
How do we select $M = \text{number of pseudo-data}$?
Power Expectation Propagation and Gaussian Processes
A Brief History of Gaussian Process Approximations

approximate generative model
exact inference
methods employing
pseudo-data
exact generative model
approximate inference

\[ \text{div}[p(f, y) || q(f, y)] \]

A Unifying View of Sparse Approximate Gaussian Process Regression
Quinonero-Candela & Rasmussen, 2005
(FITC, PITC, DTC)

A Unifying Framework for Sparse Gaussian Process Approximation using Power Expectation Propagation
Bui, Yan and Turner, 2016
(VFE, EP, FITC, PITC ...)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
$p^*(f) = p(f, y|x, \theta)$

true posterior $\mathcal{O}(N^3)$
EP pseudo-point approximation

\[ p^*(f) = p(f, y|x, \theta) = p(f|\theta) \prod_{n=1}^{N} p(y_n|f, x_n, \theta) \]
EP pseudo-point approximation

\[ p^*(f) = p(f, y|x, \theta) = p(f|\theta) \prod_{n=1}^{N} p(y_n|f, x_n, \theta) = p(y|x, \theta) p(f|y, x, \theta) \]

marginal likelihood posterior

true posterior \( O(N^3) \)
**EP pseudo-point approximation**

\[
p^*(f) = p(f, y|x, \theta) = p(f|\theta) \prod_{n=1}^{N} p(y_n|f, x_n, \theta) = p(y|x, \theta) p(f|y, x, \theta)
\]

\[
q^*(f) = p(f|\theta) \prod_{n=1}^{N} t_n(f)
\]

true posterior \(O(N^3)\)

\[y = \{y_n\}_{n=1}^{N}\]

\[x = \{x_n\}_{n=1}^{N}\]
EP pseudo-point approximation

\[
p^*(f) = p(f, y|x, \theta) \\
= p(f|\theta) \prod_{n=1}^{N} p(y_n|f, x_n, \theta) \\
= p(y|x, \theta) \underbrace{p(f|y, x, \theta)}_{\text{marginal likelihood}} \underbrace{p(f|\theta)}_{\text{posterior}}
\]

\[
q^*(f) = p(f|\theta) \prod_{n=1}^{N} t_n(f) \\
= Z_{EP} \underbrace{q(f)}_{\text{approximate posterior}}
\]
EP pseudo-point approximation

\[ p^*(f) = p(f, y | x, \theta) \]
\[ = p(f | \theta) \prod_{n=1}^{N} p(y_n | f, x_n, \theta) \]
\[ = \underbrace{p(y | x, \theta)}_{\text{marginal likelihood}} \cdot \underbrace{p(f | y, x, \theta)}_{\text{posterior}} \]

\[ q^*(f) = p(f | \theta) \prod_{n=1}^{N} t_n(f) \]
\[ = \frac{Z_{\text{EP}}}{q(f)} q(f) \]
\[ t_n(f) = \mathcal{N}(\mu_n, \Sigma_n) \]
\[ \dim(\mu) = M \quad f = \{\mu, f \neq \mu\} \]

true posterior $O(N^3)$

approximate posterior $O(NM^2)$
**EP pseudo-point approximation**

\[
p^*(f) = p(f, y|x, \theta) = p(f|\theta) \prod_{n=1}^{N} p(y_n|f, x_n, \theta) = p(y|x, \theta) \cdot p(f|y, x, \theta)
\]

**marginal likelihood**

**posterior**

**true posterior** $\mathcal{O}(N^3)$

\[
q^*(f) = p(f|\theta)p(\tilde{y}|u, \tilde{\Sigma}) = p(f|\theta) \prod_{n=1}^{N} t_n(f) = \frac{Z_{EP}}{q(f)}
\]

\[t_n(f) = \mathcal{N}(u; \mu_n, \Sigma_n)\]

\[\text{dim}(u) = M \quad f = \{u, f \neq u\}\]

**approximate posterior** $\mathcal{O}(NM^2)$
EP pseudo-point approximation

\[ p^*(f) = p(f, y|x, \theta) \]
\[ = p(f|\theta) \prod_{n=1}^{N} p(y_n|f, x_n, \theta) \]
\[ = p(y|x, \theta) \frac{p(f|y, x, \theta)}{p(f|\theta)} \]

true posterior \( \mathcal{O}(N^3) \)

\[ q^*(f) = p(f|\theta)p(\tilde{y}|u, \tilde{\Sigma}) \]
\[ = p(f|\theta) \prod_{n=1}^{N} t_n(f) \]
\[ = Z_{\text{EP}} \quad q(f) \]
\[ t_n(f) = \mathcal{N}(u; \mu_n, \Sigma_n) \]
\[ \dim(u) = M \quad f = \{u, f \neq u\} \]

approximate posterior \( \mathcal{O}(NM^2) \)

\( y = \{y_n\}_{n=1}^{N} \)
\( x = \{x_n\}_{n=1}^{N} \)

input locations of 'pseudo' data
outputs and covariance 'pseudo' data

exact joint of new GP regression model
EP algorithm
EP algorithm

1. remove pseudo-observation cavity

\[ q^n(f) = \frac{q^*(f)}{t_n(u)} \]

take out one pseudo-observation likelihood
EP algorithm

1. remove

\[ q_n(f) = \frac{q^*(f)}{t_n(u)} \]

take out one pseudo-observation likelihood

cavity

2. include

\[ p_n^{\text{tilt}}(f) = q_n(f)p(y_n|f, x_n, \theta) \]

add in one true observation likelihood

tilted
EP algorithm

1. remove

\[ q_n(f) = \frac{q^*(f)}{t_n(u)} \]

take out one pseudo-observation likelihood

cavity

2. include

\[ p_{n, \text{tilt}}(f) = q_n(f)p(y_n|f, x_n, \theta) \]

add in one true observation likelihood

tilted

KL between unnormalised stochastic processes

3. project

\[ q^*(f) = \arg\min_{q^*(f)} \text{KL}[p_{n, \text{tilt}}(f)\|q^*(f)] \]

project onto approximating family
**EP algorithm**

1. **Remove**
   
   $q_n(f) = \frac{q^*(f)}{tn(u)}$

   take out one pseudo-observation likelihood

2. **Include**
   
   $p_n^{\text{tilt}}(f) = q_n(f)p(y_n | f, x_n, \theta)$

   add in one true observation likelihood

   tilted

   KL between unnormalised stochastic processes

3. **Project**
   
   $q^*(f) = \arg\min_{q^*(f)} \text{KL} [p_n^{\text{tilt}}(f) \| q^*(f)]$

   project onto approximating family

4. **Update**
   
   $tn(u) = \frac{q^*(f)}{q_n(f)}$

   update pseudo-observation likelihood
EP algorithm

1. remove

\[ q^n(f) = \frac{q^*(f)}{t_n(u)} \]

take out one pseudo-observation likelihood

cavity

2. include

\[ p_{n}^{\text{tilt}}(f) = q^n(f)p(y_n|f, x_n, \theta) \]

add in one true observation likelihood

tilted

KL between unnormalised stochastic processes

3. project

\[ q^*(f) = \arg\min_{q^*(f)} \text{KL} \left[ p_{n}^{\text{tilt}}(f) || q^*(f) \right] \]

project onto approximating family

1. minimum: moments matched at pseudo-inputs \( O(NM^2) \)
2. Gaussian regression: matches moments everywhere

4. update

\[ t_n(u) = \frac{q^*(f)}{q^n(f)} \]

update pseudo-observation likelihood
**EP algorithm**

1. **remove**
   
   $q^n(f) = \frac{q^*(f)}{t_n(u)}$

   1. take out one pseudo-observation likelihood

2. **include**
   
   $p_n^{\text{tilt}}(f) = q^n(f)p(y_n|f, x_n, \theta)$

   1. add in one true observation likelihood

3. **project**

   $q^*(f) = \arg\min_{q^*(f)} \text{KL} [p_n^{\text{tilt}}(f) || q^*(f)]$

   1. minimum: moments matched at pseudo-inputs $O(NM^2)$
   2. Gaussian regression: matches moments everywhere

4. **update**

   $t_n(u) = \frac{q^*(f)}{q^n(f)}$

   $= z_n \mathcal{N}(K_{f_n}uK_{uu}^{-1}u; g_n, \nu_n)$

   1. update pseudo-observation likelihood

   rank 1
A Brief History of Gaussian Process Approximations

approximate generative model  methods employing pseudo-data  exact generative model
exact inference  approximate inference  

\[ \text{div}[p(f, y) \| q(f, y)] \]

A Unifying View of Sparse Approximate Gaussian Process Regression
Quinonero-Candela & Rasmussen, 2005
(FITC, PITC, DTC)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”

A Unifying Framework for Sparse Gaussian Process Approximation using Power Expectation Propagation
Bui, Yan and Turner, 2016
(VFE, EP, FITC, PITC ...)
Fixed points of EP = FITC approximation

approximate generative model
exact inference

methods employing
pseudo-data

exact generative model
approximate inference

\[ \text{div}[p(f, y)||q(f, y)] \]

A Unifying View of Sparse Approximate Gaussian Process Regression
Quinonero-Candela & Rasmussen, 2005
(FITC, PITC, DTC)

VFE
EP
FITC
PITC
DTC
PP

A Unifying Framework for Sparse Gaussian Process Approximation using Power Expectation Propagation
Bui, Yan and Turner, 2016
(VFE, EP, FITC, PITC ...)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes"
DTC / PP: Seeger et al. “Fast Forward Selection to Speed Up Sparse Gaussian Process Regression"
Fixed points of $\text{EP} = \text{FITC}$ approximation

approximate generative model
exact inference

methods employing pseudo-data

exact generative model
approximate inference

div$[p(f, y) || q(f, y)]$

div$[p(f|y) || q(f)]$

A Unifying View of Sparse Approximate Gaussian Process Regression
Quinonero-Candela & Rasmussen, 2005
(FITC, PITC, DTC)

A Unifying Framework for Sparse Gaussian Process Approximation using Power Expectation Propagation
Bui, Yan and Turner, 2016
(VFE, EP, FITC, PITC ...)

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”
Fixed points of $EP = FITC$ approximation

**approximate generative model**  
**methods employing pseudo-data**  
**exact generative model**  
**approximate inference**

$$\text{div}[p(f, y) \| q(f, y)]$$

A Unifying View of Sparse Approximate Gaussian Process Regression  
Quinonero-Candela & Rasmussen, 2005  
(FITC, PITC, DTC)

A Unifying Framework for Sparse Gaussian Process Approximation using Power Expectation Propagation  
Bui, Yan and Turner, 2016  
(VFE, EP, FITC, PITC ...)

**interpretation resolves issues with FITC:**  
why does it work so well?  
are we allowed to increase $M$ with $N$ 

FITC: Snelson et al. “Sparse Gaussian Processes using Pseudo-inputs”  
PITC: Snelson et al. “Local and global sparse Gaussian process approximations”  
VFE: Titsias “Variational Learning of Inducing Variables in Sparse Gaussian Processes”  
**EP algorithm**

1. **remove**
   
   $q^n(f) = \frac{q^*(f)}{t_n(u)}$

   
   take out one pseudo-observation likelihood

   cavity

2. **include**
   
   $p_{n}^{\text{tilt}}(f) = q^n(f)p(y_n|f, x_n, \theta)$

   
   add in one true observation likelihood

   tilted

   KL between unnormalised stochastic processes

3. **project**
   
   $q^*(f) = \text{argmin}_{q^*(f)} KL[p_{n}^{\text{tilt}}(f) || q^*(f)]$

   project onto approximating family

   1. minimum: moments matched at pseudo-inputs $O(\mathcal{N}\mathcal{M}^2)$
   2. Gaussian regression: matches moments everywhere

4. **update**
   
   $t_n(u) = \frac{q^*(f)}{q^n(f)}$

   update pseudo-observation likelihood

   rank 1

   $= z_n \mathcal{N}(K_{f_n u} K_{uu}^{-1} u; g_n, \nu_n)$
Power EP algorithm (as tractable as EP)

1. remove

\[ q_n^\backslash (f) = \frac{q^* (f)}{t_n (u)^{\alpha}} \]

cavity

2. include

\[ p_n^{\text{tilt}} (f) = q_n^\backslash (f) p(y_n | f, x_n, \theta)^{\alpha} \]

tilted

KL between unnormalised stochastic processes

3. project

\[ q^* (f) = \arg\min_{q^* (f)} \text{KL} [p_n^{\text{tilt}} (f) \| q^* (f)] \]

1. minimum: moments matched at pseudo-inputs \( O(NM^2) \)
2. Gaussian regression: matches moments everywhere

4. update

\[ t_n (u)^{\alpha} = \frac{q^* (f)}{q_n^\backslash (f)} \]

update pseudo-observation likelihood

\[ t_n (u) = z_n \mathcal{N}(K_{f_n u} K_{uu}^{-1} u; g_n, v_n) \]

rank 1
Power EP: a unifying framework

\[ \alpha \to 0 \quad \alpha \quad \alpha = 1 \]

VFE
Titsias, 2009

FITC
Csato and Opper, 2002
Snelson and Ghahramani, 2005
Power EP: a unifying framework

GP Regression

GP Classification

[5] Snelson et al., 2005

[10] Csató et al., 2002

[13] Qi et al., 2010
[14] Hensman et al., 2015
[16] Matthews et al., 2016
[17] Figueiras-Vidal et al., 2009

* = optimised pseudo-inputs
** = structured versions of VFE recover VFE
How should I set the power parameter $\alpha$?

8 UCI regression datasets
- 20 random splits
- $M = 0 - 200$
- hypers and inducing inputs optimised

6 UCI classification datasets
- 20 random splits
- $M = 10, 50, 100$
- hypers and inducing inputs optimised

$\alpha = 0.5$ does well on average
References (hyperlinked)

**Approximate inference in GPs:**
- Sparse Online Gaussian Processes, Csato and Opper, Neural Computation, 2002
- A Unifying View of Sparse Approximate Gaussian Process Regression, Quinonero-Candela and Rasmussen, JMLR, 2005
- Variational Learning of Inducing Variables in Sparse Gaussian Processes, Titsias, AISTats, 2009
- On Sparse Variational Methods and the Kullback-Leibler Divergence between Stochastic Processes, Matthews et al., ICML 2016
- A Unifying Framework for Gaussian Process Pseudo-Point Approximations using Power Expectation Propagation, Bui et al., JMLR 2017
- Streaming Sparse Gaussian Process Approximations, Bui et al., NIPS 2017
- Efficient Deterministic Approximate Bayesian Inference for Gaussian Process Models, Bui, thesis, 2018

**Deep Gaussian Processes:**
- Deep Gaussian Processes for Regression using Approximate Expectation Propagation, Bui et al., ICML 2016
- Doubly Stochastic Variational Inference for Deep Gaussian Processes, Salimbeni and Deisenroth, NIPS 2017
Appendix: proof of KL divergence properties

Minimise Kullback Leibler divergence (relative entropy) $\mathcal{KL}(q(x)||p(x))$: add Lagrange multiplier (enforce $q(x)$ normalises), take variational derivatives:

$$\frac{\delta}{\delta q(x)} \left[ \int q(x) \log \frac{q(x)}{p(x)} \, dx + \lambda (1 - \int q(x) \, dx) \right] = \log \frac{q(x)}{p(x)} + 1 - \lambda.$$

Find stationary point by setting the derivative to zero:

$$q(x) = \exp(\lambda - 1) p(x), \quad \text{normalization condition } \lambda = 1, \quad \text{so } q(x) = p(x),$$

which corresponds to a minimum, since the second derivative is positive:

$$\frac{\delta^2}{\delta q(x) \delta q(x)} \mathcal{KL}(q(x)||p(x)) = \frac{1}{q(x)} > 0.$$

The minimum value attained at $q(x) = p(x)$ is $\mathcal{KL}(p(x)||p(x)) = 0$, showing that $\mathcal{KL}(q(x)||p(x))$

- is non-negative and it attains its minimum $0$ when $p(x)$ and $q(x)$ are equal