Tree-structured Gaussian Process Approximations

Thang Bui
joint work with Richard Turner

CBL, Cambridge

Oct 6th, 2014
Motivation
Motivation

Using Gaussian Processes for long time series
Motivation

Using Gaussian Processes for long time series
Motivation

Using Gaussian Processes for long time series

![Graph showing signal over time and bike rentals over days with missing data highlighted.](image)
Motivation

Using Gaussian Processes for long time series
Examples

Time/s
SMSE
10
0
10
-2
10
-1
10
0
y
x
-200 -150 -100 -50 0 50 100 150 200
-3
-2
-1
0
1
2
3
Exact
VFE
Examples

Time/s
SMSE
10
0
10
-2
10
-1
10
0
y
x
-200 -150 -100 -50 0 50 100 150 200
-3
-2
-1
0
1
2
3

Exact
VFE

3 / 17
Examples

Time/s
SMSE
10
0
10
-2
10
-1
10
0
y
x
-200 -150 -100 -50 0 50 100 150 200
-3
-2
-1
0
1
2
3
Exact
VFE

3 / 17
Examples

Time/s
SMSE
10
0
10
1
10
-2
10
-1
10
0
y
x
-500 -400 -300 -200 -100 0 100 200 300 400 500
-3
-2
-1
0
1
2
3
Exact
VFE
Examples

![Graph showing examples with time in seconds (Time/s), SMSE, and exact VFE. The graph includes a y-axis from -3 to 3 and an x-axis from -500 to 500. The SMSE is plotted on a logarithmic scale from $10^{-2}$ to $10^0$. The title 'Exact VFE' is visible.]
Examples

![Graph showing data points and lines for different time intervals and SMSE values.](image_url)
Examples

![Graph showing time vs. SMSE with 'Exact' and 'VFE' lines. The graph includes axes for time in seconds and SMSE, with a logarithmic scale for SMSE. The x-axis ranges from -1000 to 1000, and the y-axis ranges from -3 to 3. The graph also includes a legend for 'Exact' and 'VFE'.]
Examples

Time/s
SMSE
10
0
10
1
10
-2
10
-1
10
0
y
x
-1000 -800 -600 -400 -200 0 200 400 600 800 1000
-3
-2
-1
0
1
2
3
Exact
VFE

5 / 17
Examples

![Graph showing time/s and SMSE with exact and VFE lines, along with SMSE values on a logarithmic scale.]
Examples

```
SMSE
10
0
10
1
10
-2
10
-1
10
0
y
x
-1000 -800 -600 -400 -200 0 200 400 600 800 1000
-3
-2
-1
0
1
2
3
Exact
VFE
```

![Graph showing examples of SMSE, Time/s, and VFE with exact and VFE lines.](image-url)
Examples

![Graph showing time/s and SMSE](image-url)

- **Y** vs **X**
- **SMSE** vs **Time/s**
- **Exact** and **VFE** lines

---

5 / 17
Examples

Time/s
SMSE
10
0
10
1
10
-2
10
-1
10
0
y
x
-1000 -800 -600 -400 -200 0 200 400 600 800 1000
-3
-2
-1
0
1
2
3
Exact
VFE
Examples

Time/s
SMSE
10
0
10
1
10
-2
10
-1
10
0
y
x
-1000 -800 -600 -400 -200 0 200 400 600 800 1000
-3
-2
-1
0
1
2
3
Exact
VFE

5 / 17
Examples

![Graph showing examples of Time/s, SMSE, and SMSE vs. Time/s. The graph compares the performance of different methods, including Exact, VFE, and Our method.](image-url)
GPs for regression

Given $\{x_n, y_n\}_{n=1}^N$, and

$$y_n = f(x_n) + \epsilon_n,$$

$$\epsilon_n \sim \text{iid } \mathcal{N}(0, \sigma_n^2),$$

$$f \sim \text{GP}(0, k_\theta(*, *))$$
GPs for regression

Given \( \{x_n, y_n\}_{n=1}^N \), and

\[
\begin{align*}
y_n &= f(x_n) + \epsilon_n, \\
\epsilon_n &\sim \text{iid } \mathcal{N}(0, \sigma_n^2), \\
f &\sim \text{GP}(0, k_\theta(*, *))
\end{align*}
\]

or \( p(f) = \mathcal{N}(0, K_{ff}) \),

\( p(y|f) = \mathcal{N}(f, \sigma_n^2 I) \)
GPs for regression

Given \( \{x_n, y_n\}_{n=1}^N \), and

\[
y_n = f(x_n) + \epsilon_n,
\]

\[
\epsilon_n \sim \text{iid } \mathcal{N}(0, \sigma_n^2),
\]

\[
f \sim \text{GP}(0, k_\theta(*, *))
\]

or \( p(f) = \mathcal{N}(0, K_{ff}) \),

\[
p(y|f) = \mathcal{N}(f, \sigma_n^2 I)
\]
GPs for regression

Given \( \{x_n, y_n\}_{n=1}^N \), and

\[
y_n = f(x_n) + \epsilon_n,
\]

\[
\epsilon_n \sim \text{iid } \mathcal{N}(0, \sigma_n^2),
\]

\[
f \sim \text{GP}(0, k_\theta(*, *))
\]

or \( p(f) = \mathcal{N}(0, K_{ff}) \),

\[
p(y|f) = \mathcal{N}(f, \sigma_n^2 I)
\]

Prediction:

\[
p(f_*, y) = \mathcal{N}
\left(
\begin{bmatrix}
  f_* \\
  y
\end{bmatrix};
\begin{bmatrix}
  0 \\
  0
\end{bmatrix},
\begin{bmatrix}
  K_{**} & K_{*f} \\
  K_{f*} & K_{ff} + \sigma^2 I
\end{bmatrix}
\right)
\]
GPs for regression

Given \( \{x_n, y_n\}_{n=1}^{N} \), and

\[
y_n = f(x_n) + \epsilon_n, \\
\epsilon_n \sim \text{iid } \mathcal{N}(0, \sigma_n^2), \\
f \sim \text{GP}(0, k_{\theta}(\cdot, \cdot))
\]

or \( p(f) = \mathcal{N}(0, K_{ff}) \),

\( p(y|f) = \mathcal{N}(f, \sigma_n^2 I) \)

Prediction:

\[
p(f_*, y) = \mathcal{N} \left( \begin{bmatrix} f_* \\ y \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{**} & K_{*f} \\ K_{f*} & K_{ff} + \sigma^2 I \end{bmatrix} \right)
\]

\( \Rightarrow p(f_*|y) = \mathcal{N}(f_*; K_{*f}(K_{ff} + \sigma^2 I)^{-1}y; k_{**} - K_{*f}(K_{ff} + \sigma^2 I)^{-1}K_{f*}) \)
GPs for regression

Given \( \{ x_n, y_n \}_{n=1}^N \), and

\[
y_n = f(x_n) + \epsilon_n, \\
\epsilon_n \sim \text{iid } \mathcal{N}(0, \sigma_n^2), \\
f \sim \text{GP}(0, k_\theta(\ast, \ast))
\]

or \( p(f) = \mathcal{N}(0, K_{ff}) \),

\[
p(y|f) = \mathcal{N}(f, \sigma_n^2 I)
\]

Prediction:

\[
p(f_\ast, y) = \mathcal{N} \left( \begin{bmatrix} f_\ast \\ y \end{bmatrix} ; \begin{bmatrix} 0 \\ K_{f*} \end{bmatrix}, \begin{bmatrix} K_{**} & K_{*f} \\ K_{f*} & K_{ff} + \sigma^2 I \end{bmatrix} \right)
\]

\[
\Rightarrow p(f_\ast | y) = \mathcal{N}(f_\ast; K_{*f}(K_{ff} + \sigma^2 I)^{-1} y; \kappa_{**} - K_{*f}(K_{ff} + \sigma^2 I)^{-1} K_{f*})
\]

Hyperparameters learning: maximising \( \mathcal{L} = \log p(y) = \log \mathcal{N}(0, K_{ff} + \sigma_n^2 I) \)
GPs for regression

Given \( \{x_n, y_n\}_{n=1}^N \), and

\[
y_n = f(x_n) + \epsilon_n, \\
\epsilon_n \sim \text{iid } \mathcal{N}(0, \sigma_n^2), \\
f \sim \text{GP}(0, k_\theta(\ast, \ast))
\]

or \( p(f) = \mathcal{N}(0, K_{ff}) \),

\( p(y|f) = \mathcal{N}(f, \sigma_n^2 I) \)

Prediction:

\[
p(f_*, y) = \mathcal{N} \left( \begin{bmatrix} f_* \\ y \end{bmatrix}; \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} K_{**} & K_{f*} \\ K_{f*} & K_{ff} + \sigma^2 I \end{bmatrix} \right)
\]

\( \Rightarrow p(f_*|y) = \mathcal{N}(f_*; K_{f*}(K_{ff} + \sigma^2 I)^{-1}y; k_{**} - K_{f*}(K_{ff} + \sigma^2 I)^{-1}K_{f*}) \)

Hyperparameters learning: maximising \( \mathcal{L} = \log p(y) = \log \mathcal{N}(0, K_{ff} + \sigma_n^2 I) \)

Complexity: \( \mathcal{O}(N^3) \)
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)
The Fully Independent Training Conditional (FITC) approximation (Snelson and Ghahramani 2006)

1. **Augment the model with inducing points** \( \{x_m, u_m\}_{m=1}^M \)
The Fully Independent Training Conditional (FITC) approximation (Snelson and Ghahramani 2006)

Augment the model with inducing points \( \left\{ x_m, u_m \right\}_{m=1}^{M} \)

\[
p(f, u) = \mathcal{N} \left( \begin{bmatrix} f \\ u \end{bmatrix} ; \begin{bmatrix} 0 \\ 0 \end{bmatrix} , \begin{bmatrix} K_{ff} & K_{fu} \\ K_{uf} & K_{uu} \end{bmatrix} \right)
\]
Augment the model with inducing points $\{x_m, u_m\}_{m=1}^M$

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

Remove all direct dependencies between function values $f$, i.e. assume $f_i \perp f_j | u$, $\forall i, j$
The Fully Independent Training Conditional (FITC) approximation  
(Snelson and Ghahramani 2006)

1. Augment the model with inducing points \( \{ x_m, u_m \}_{m=1}^{M} \)

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

2. Remove all direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u \), \( \forall i, j \)
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^{M} \)

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

2. Remove all direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u, \forall i, j \)
The Fully Independent Training Conditional (FITC) approximation (Snelson and Ghahramani 2006)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)

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

2. Remove all direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u, \ \forall i, j \)
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

1 Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)

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

2 Remove all direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u, \forall i, j \)
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)

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

2. Remove all direct dependencies between function values \( f \), i.e. assume \( f_i \perp \perp f_j | u, \forall i, j \)
The Fully Independent Training Conditional (FITC) approximation (Snelson and Ghahramani 2006)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)

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

2. Remove all direct dependencies between function values \( f \), i.e. assume 

\[ f_i \perp f_j | u, \forall i, j \]

3. Calibrate model using a forward KL divergence
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

1. Augment the model with inducing points \( \{ x_m, u_m \}_{m=1}^M \)

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

2. Remove all direct dependencies between function values \( f \), i.e. assume 
\( f_i \perp f_j | u \), \( \forall i, j \)

3. Calibrate model using a forward KL divergence

\[
\arg \min_{q(u), \{ q(f_i|u) \}_{i=1}^N} \sum_{n=1}^N KL(p(f, u)||q(u)) \prod_{n=1}^N q(f_i|u))
\]
The Fully Independent Training Conditional (FITC) approximation  
(Snelson and Ghahramani 2006)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)

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

2. Remove all direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u \), \( \forall i, j \)

3. Calibrate model using a forward KL divergence

\[
\arg \min_{q(u), \{q(f_i|u)\}_{i=1}^N} \sum_{n=1}^N \text{KL}(p(f_i|u)||q(f_i|u))
\]

\[
\Rightarrow q(u) = p(u) \ , \ q(f_i|u) = p(f_i|u)
\]
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}), \]
\[ q(f_i|u) = p(f_i|u) = \mathcal{N}(f_i; K_{fu}K_u^{-1}uu, K_{fi} - K_{fi}uK_u^{-1}K_{fu}). \]
\[ p(y_i|f_i) = \mathcal{N}(y_i; f_i, \sigma_n^2 I). \]

Marginalising gives:
\[ p(y|u) = \mathcal{N}(y; K_{fu}K_u^{-1}uu, D + \sigma_n^2 I), \]
where
\[ D_{ii} = K_{fi} - K_{fi}uK_u^{-1}K_{fu}. \]

⇒ FITC gives a Factor Analysis model, with a strange parameterisation.
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

New generative model:

\[ q(u) = p(u) = N(u; 0, K_{uu}), \]
\[ q(f_i | u) = p(f_i | u) = N(f_i; K_{fu} K_{uu}^{-1} u, K_{f_i f_i} - K_{fi u} K_{uu}^{-1} K_{uf_i}) , \]
\[ p(y_i | f_i) = N(y_i; f_i, \sigma^2_n). \]

Marginalising gives:
\[ p(y | u) = N(y; K_{fu} K_{uu}^{-1} u, D_n + \sigma^2_n I) , \]
where
\[ D_{ii} = K_{f_i f_i} - K_{fi u} K_{uu}^{-1} K_{uf_i} . \]

⇒ FITC gives a Factor Analysis model, with a strange parameterisation.
The Fully Independent Training Conditional (FITC) approximation (Snelson and Ghahramani 2006)

New generative model:

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}), \]
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

New generative model:

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}), \]
\[ q(f_i|u) = p(f_i|u) = \mathcal{N}(f_i; K_{fu}K_{uu}^{-1}u, K_{fifi} - K_{fu}K_{uu}^{-1}K_{uf_i}), \]
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

New generative model:

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

\[
q(f_i | u) = p(f_i | u) = \mathcal{N}(f_i; K_{fi}uK_{uu}^{-1}u, K_{fi} - K_{fi}uK_{uu}^{-1}K_{ufi}),
\]

\[
p(y_i | f_i) = \mathcal{N}(y_i; f_i, \sigma_n^2).
\]
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

New generative model:

\[ q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}), \]
\[ q(f_i | u) = p(f_i | u) = \mathcal{N}(f_i; K_{fi}uK_{uu}^{-1}u, K_{fi} - K_{fi}uK_{uu}^{-1}K_{ufi}), \]
\[ p(y_i | f_i) = \mathcal{N}(y_i; f_i, \sigma_n^2). \]

Marginalising \( f \) gives:

\[ p(y | u) = \mathcal{N}(y; K_{fu}K_{uu}^{-1}u, D + \sigma_n^2 I), \]

where \( D_{ii} = K_{fi}f_i - K_{fi}uK_{uu}^{-1}K_{ufi}. \)
The Fully Independent Training Conditional (FITC) approximation
(Snelson and Ghahramani 2006)

New generative model:

\[
q(u) = p(u) = \mathcal{N}(u; 0, K_{uu}),
\]
\[
q(f_i|u) = p(f_i|u) = \mathcal{N}(f_i; K_{fiu}K_{uu}^{-1}u, K_{fifi} - K_{fiu}K_{uu}^{-1}K_{ufi}),
\]
\[
p(y_i|f_i) = \mathcal{N}(y_i; f_i, \sigma_n^2).
\]

Marginalising \( f \) gives:

\[
p(y|u) = \mathcal{N}(y; K_{fu}K_{uu}^{-1}u, D + \sigma_n^2 I),
\]

where \( D_{ii} = K_{fifi} - K_{fiu}K_{uu}^{-1}K_{ufi} \).

\( \Rightarrow \) FITC gives a Factor Analysis model, with a *strange* parameterisation.
The Partially Independent Training Conditional approximation
(Quiñonero-Candela and Rasmussen 2005)

Augment the model with inducing points \{x_m, u_m\}_{m=1}^M

Remove some direct dependencies between function values \(f_i\), i.e. assume \(f_i \succeq f_j | u, \forall i, j\)

Calibrate model using a forward KL divergence
\(\rightarrow q(u) = p(u)\) and \(q(f_i | u) = p(f_i | u)\)

PITC is also a Factor Analysis model, where
\(p(y | u) = N(y; K_{fu}K_u^{-1}uu, \text{blkdiag}(K_ff - K_{fu}K_u^{-1}uuK_{uf}) + \sigma^2 nI)\)

Complexity of both FITC and PITC: \(O(NM^2)\)
The Partially Independent Training Conditional approximation (Quiñonero-Candela and Rasmussen 2005)

Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)

Diagram:

\[
\begin{align*}
&u_1 \quad u_2 \quad u_3 \quad u_4 \quad \ldots \\
&f_1 \quad f_2 \quad f_3 \quad f_4 \quad f_5 \quad \ldots
\end{align*}
\]
The Partially Independent Training Conditional approximation
(Quiñonero-Candela and Rasmussen 2005)

1. Augment the model with inducing points \( \{ x_m, u_m \}_{m=1}^M \)

2. Remove some direct dependencies between function values \( f \),
   i.e. assume \( f_i \perp \!
\!
\! \perp f_j \mid u \), \( \forall i, j \)
The Partially Independent Training Conditional approximation
(Quiñonero-Candela and Rasmussen 2005)

Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^{M} \)

Remove some direct dependencies between function values \( f \),
 i.e. assume \( f_i \perp f_j | u, \forall i, j \)
The Partially Independent Training Conditional approximation
(Quiñonero-Candela and Rasmussen 2005)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)
2. Remove some direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u \), \( \forall i, j \)
The Partially Independent Training Conditional approximation  
(Quiñonero-Candela and Rasmussen 2005)

Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)

Remove *some* direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u, \forall i, j \)

**Calibrate model using a forward KL divergence**
The Partially Independent Training Conditional approximation
(Quiñonero-Candela and Rasmussen 2005)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)
2. Remove some direct dependencies between function values \( f \), i.e. assume \( f_i \perp\!\!\!\perp f_j | u \), \( \forall i, j \)
3. **Calibrate model using a forward KL divergence**

\[
\rightarrow q(u) = p(u) \text{ and } q(f_i | u) = p(f_i | u)
\]
Augment the model with inducing points \( \{ x_m, u_m \}_{m=1}^M \).

Remove some direct dependencies between function values \( f \), i.e. assume \( f_i \perp f_j | u \), \( \forall i, j \).

Calibrate model using a forward KL divergence

\[
\rightarrow q(u) = p(u) \text{ and } q(f_i | u) = p(f_i | u)
\]

PITC is also a Factor Analysis model, where

\[
p(y | u) = \mathcal{N}(y; \mathbf{K}_{fu} \mathbf{K}_{uu}^{-1} u, \text{blkdiag}(\mathbf{K}_{ff} - \mathbf{K}_{fu} \mathbf{K}_{uu}^{-1} \mathbf{K}_{uf}) + \sigma_n^2 \mathbf{I}).
\]
The Partially Independent Training Conditional approximation (Quiñonero-Candela and Rasmussen 2005)

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^{M} \)
2. Remove some direct dependencies between function values \( f_i \), i.e. assume \( f_i \perp f_j | u, \forall i, j \)
3. Calibrate model using a forward KL divergence

\[
\rightarrow q(u) = p(u) \quad \text{and} \quad q(f_i|u) = p(f_i|u)
\]

PITC is also a Factor Analysis model, where

\[
p(y|u) = \mathcal{N}(y; K_{fu}K_{uu}^{-1}u, \text{blkdiag}(K_{ff} - K_{fu}K_{uu}^{-1}K_{uf}) + \sigma_n^2 I).
\]

Complexity of both FITC and PITC: \( \mathcal{O}(NM^2) \)
The chain-structured inducing point approximation
Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)
Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \).

Remove some direct dependencies between function values \( f \),
Augment the model with inducing points \( \{ x_m, u_m \}_{m=1}^M \)

Remove *some* direct dependencies between function values \( f \), and assume a chain structure on inducing points \( u \)
The chain-structured inducing point approximation

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)
2. Remove some direct dependencies between function values \( f \), and assume a chain structure on inducing points \( u \)
The chain-structured inducing point approximation

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)
2. Remove some direct dependencies between function values \( f \), and assume a chain structure on inducing points \( u \)
The chain-structured inducing point approximation

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^{M} \)
2. Remove some direct dependencies between function values \( f \), and assume a chain structure on inducing points \( u \)
3. Calibrate model using a forward KL divergence
The chain-structured inducing point approximation

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)
2. Remove some direct dependencies between function values \( f \), and assume a chain structure on inducing points \( u \)
3. Calibrate model using a forward KL divergence

\[
\arg \min_{\{q(u_{B_k}|u_{B_{k-1}}), q(f_{B_k}|u_{B_k})\}_{k=1}^K} \quad \text{KL}(p(f, u) || \prod_k q(u_{B_k}|u_{B_{k-1}}) q(f_{B_k}|u_{B_k}))
\]
The chain-structured inducing point approximation

1. Augment the model with inducing points \( \{x_m, u_m\}_{m=1}^M \)
2. Remove some direct dependencies between function values \( f \), and assume a chain structure on inducing points \( u \)

3. Calibrate model using a forward KL divergence

\[
\arg\min _{\{q(u_{Bk}|u_{Bk-1}), q(f_{Bk}|u_{Bk})\}_{k=1}^K} \quad KL(p(f, u) \| \prod_k q(u_{Bk}|u_{Bk-1}) q(f_{Bk}|u_{Bk}))
\]

\[\Rightarrow q(u_{Bk}|u_{Bk-1}) = p(u_{Bk}|u_{Bk-1}) \quad \text{and} \quad q(f_{Bk}|u_{Bk}) = p(f_{Bk}|u_{Bk})\]
The chain-structured inducing point approximation
The chain-structured inducing point approximation

New generative model:

\[ q(u) = \prod_{k=1}^{K} q(u_{B_k} | u_{B_{k-1}}), \]
\[ q(f|u) = \prod_{k=1}^{K} q(f_{B_k} | u_{B_k}), \]
\[ p(y|f) = \prod_{n=1}^{N} p(y_n; f_n, \sigma_n^2). \]
The chain-structured inducing point approximation

New generative model:

\[
q(u) = \prod_{k=1}^{K} q(u_{B_k}|u_{B_{k-1}}),
\]

\[
q(f|u) = \prod_{k=1}^{K} q(f_{B_k}|u_{B_k}),
\]

\[
p(y|f) = \prod_{n=1}^{N} p(y_n; f_n, \sigma_n^2).
\]

where

\[
q(u_{B_k}|u_{B_{k-1}}) = p(u_{B_k}|u_{B_{k-1}}) = \mathcal{N}(u_{B_k}; A_k u_{B_{k-1}}, Q_k),
\]

\[
q(f_{B_k}|u_{B_k}) = p(f_{B_k}|u_{B_k}) = \mathcal{N}(f_{B_k}; C_k u_{B_k}, R_k).
\]
The chain-structured inducing point approximation

New generative model:

\[
q(u) = \prod_{k=1}^{K} q(u_{B_k} | u_{B_{k-1}}),
\]

\[
q(f|u) = \prod_{k=1}^{K} q(f_{B_k} | u_{B_k}),
\]

\[
p(y|f) = \prod_{n=1}^{N} p(y_n; f_n, \sigma^2_n).
\]

where

\[
q(u_{B_k} | u_{B_{k-1}}) = p(u_{B_k} | u_{B_{k-1}}) = \mathcal{N}(u_{B_k}; A_k u_{B_{k-1}}, Q_k),
\]

\[
q(f_{B_k} | u_{B_k}) = p(f_{B_k} | u_{B_k}) = \mathcal{N}(f_{B_k}; C_k u_{B_k}, R_k).
\]

- This is a Linear Dynamical System with a *strange* parameterisation!
The chain-structured inducing point approximation

New generative model:

\[
q(u) = \prod_{k=1}^{K} q(u_{B_k} | u_{B_{k-1}}),
\]

\[
q(f|u) = \prod_{k=1}^{K} q(f_{B_k} | u_{B_k}),
\]

\[
p(y|f) = \prod_{n=1}^{N} p(y_n; f_n, \sigma^2_n).
\]

where

\[
q(u_{B_k} | u_{B_{k-1}}) = p(u_{B_k} | u_{B_{k-1}}) = \mathcal{N}(u_{B_k}; A_k u_{B_{k-1}}, Q_k),
\]

\[
q(f_{B_k} | u_{B_k}) = p(f_{B_k} | u_{B_k}) = \mathcal{N}(f_{B_k}; C_k u_{B_k}, R_k).
\]

- This is a Linear Dynamical System with a *strange* parameterisation!
- Inference using Kalman smoothing algorithm
The chain-structured inducing point approximation

New generative model:

\[
q(u) = \prod_{k=1}^{K} q(u_{B_k} | u_{B_{k-1}}),
\]

\[
q(f|u) = \prod_{k=1}^{K} q(f_{B_k} | u_{B_k}),
\]

\[
p(y|f) = \prod_{n=1}^{N} p(y_n; f_n, \sigma_n^2).
\]

where

\[
q(u_{B_k} | u_{B_{k-1}}) = p(u_{B_k} | u_{B_{k-1}}) = \mathcal{N}(u_{B_k}; A_k u_{B_{k-1}}, Q_k),
\]

\[
q(f_{B_k} | u_{B_k}) = p(f_{B_k} | u_{B_k}) = \mathcal{N}(f_{B_k}; C_k u_{B_k}, R_k).
\]

- This is a Linear Dynamical System with a \textit{strange} parameterisation!
- Inference using Kalman smoothing algorithm
- Complexity: \( \mathcal{O}(ND^2) \), \( D \): average number of observations per block
The chain-structured inducing point approximation

- Hyperparameter learning

\[ p(y_{1:K} | \theta) = \prod_{k=1}^{K} p(y_k | y_{1:k-1}, \theta) \]

\[ \frac{d}{d\theta} \log p(y | \theta) = \sum_{k=1}^{K} \left[ \langle \frac{d}{d\theta} \log p(u_k | u_l) \rangle_{p(u_k, u_l | y)} + \langle \frac{d}{d\theta} \log p(y_k | u_k) \rangle_{p(u_k | y)} \right]. \]
The chain-structured inducing point approximation

- Hyperparameter learning

\[ p(y_{1:K} | \theta) = \prod_{k=1}^{K} p(y_k | y_{1:k-1}, \theta) \]

\[ \frac{d}{d\theta} \log p(y | \theta) = \sum_{k=1}^{K} \left[ \langle \frac{d}{d\theta} \log p(u_k | u_l) \rangle_{p(u_k,u_l|y)} + \langle \frac{d}{d\theta} \log p(y_k | u_k) \rangle_{p(u_k|y)} \right]. \]

- Extension to high dimensional settings: tree structured inducing points
The chain-structured inducing point approximation

- **Hyperparameter learning**

  \[
  p(y_{1:K} | \theta) = \prod_{k=1}^{K} p(y_k | y_{1:k-1}, \theta)
  \]

  \[
  \frac{d}{d\theta} \log p(y | \theta) = \sum_{k=1}^{K} \left[ \langle \frac{d}{d\theta} \log p(u_k | u_l) \rangle_{p(u_k, u_l | y)} + \langle \frac{d}{d\theta} \log p(y_k | u_k) \rangle_{p(u_k | y)} \right].
  \]

- **Extension to high dimensional settings: tree structured inducing points**

- **A large number of inducing points can be handled using our scheme**
Experimental results

Task: filling missing data

Data:

- Subband of a speech signal: $N = 50000$, SE kernel:
  $$k_{\theta}(t, t') = \sigma^2 \exp\left(-\frac{1}{2l^2}(t - t')^2\right)$$

- Filtered speech signal: $N = 50000$, spectral mixture kernel:
  $$k_{\theta}(t, t') = \sum_{k=1}^{2} \sigma_k^2 \cos(\omega_k(t - t')) \exp\left(-\frac{1}{2l_k^2}(t - t')^2\right)$$

Evaluation: Prediction error vs. Training/Test time
Experimental results

Task: filling missing data

Data:
- Subband of a speech signal: $N = 50000$, $SE$ kernel:
  $$k_\theta(t, t') = \sigma^2 \exp\left(-\frac{1}{2l^2}(t - t')^2\right)$$
- Filtered speech signal: $N = 50000$, spectral mixture kernel:
  $$k_\theta(t, t') = \sum_{k=1}^{2} \sigma_k^2 \cos(\omega_k(t - t')) \exp\left(-\frac{1}{2l_k^2}(t - t')^2\right)$$

Evaluation: Prediction error vs. Training/Test time

Left: Subband data, Right: Filtered signal
Experimental results

![Graph showing the relationship between training time and SMSE for different methods: Chain, Local, FITC, VFE, SSGP, and SDE.]
Experimental results

SMSE vs Training time/s

- Chain
- Local
- FITC
- VFE
- SSGP
- SDE

SMSE values:
- 0.01
- 0.1
- 0.2
- 0.5
- 1

Training time values:
- 10
- 100
- 1000
- 10000
Experimental results

<table>
<thead>
<tr>
<th>SMSE</th>
<th>Training time/s</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.01</td>
<td>10</td>
</tr>
<tr>
<td>0.1</td>
<td>100</td>
</tr>
<tr>
<td>0.2</td>
<td>1000</td>
</tr>
<tr>
<td>0.5</td>
<td>10000</td>
</tr>
<tr>
<td>1</td>
<td>15000</td>
</tr>
</tbody>
</table>

Chain, Local, FITC, VFE, SSGP, SDE

Training times for different datasets and models.
Experimental results

<table>
<thead>
<tr>
<th>Training time/s</th>
<th>SMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>32</td>
<td>32</td>
</tr>
<tr>
<td>64</td>
<td>64</td>
</tr>
<tr>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>256</td>
<td>256</td>
</tr>
<tr>
<td>256</td>
<td>256</td>
</tr>
<tr>
<td>512</td>
<td>512</td>
</tr>
<tr>
<td>512</td>
<td>512</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>20000</td>
<td>20000</td>
</tr>
</tbody>
</table>

Chain         
Local          
FITC           
VFE            
SSGP           
SDE

SMSE vs Training time/s graph
Experimental results

![Graph showing SMSE vs. Training time/s for different methods: Chain, Local, FITC, VFE, SSGP, SDE. The x-axis represents training time in seconds, and the y-axis represents SMSE. Different symbols and colors are used to distinguish between methods.](image-url)
Experimental results

**Task:** filling missing data

**Data:** Altitude of a 20km x 30km region, 80 missing blocks of 1km x 1km or 200k/40k training/test points, 2D SE kernel
Summary

- Tree-structured inducing points approximation
  - pseudo-dataset has tree/chain structure
  - model was calibrated using a KL divergence
  - inference and learning via Gaussian belief propagation
  - better time/accuracy trade-off compared to FITC, VFE
Thanks!