

Sparse Multi-output Gaussian Processes

Mauricio Alvarez Neil Lawrence

School of Computer Science
University of Manchester

- 1 Introduction
- 2 The Convolution Process
- 3 Sparse approximation
- 4 Experiments
- 5 Summary

Introduction I

- We consider the problem of modelling correlated outputs from a single Gaussian process (GP).
- Modelling multiple output variables is a challenge as we are required to compute cross covariances between the different outputs. The properties of those cross covariances are
 - *expressive* in the sense of capturing dependencies.
 - lead to a positive definite matrix for the full GP.

Introduction II

- One method used for this kind of modelling tasks employs a convolution process (CP)[Hig05, BF05].
- A convolution process is a moving-average construction that guarantees a valid covariance function.
- Full covariance function results in significant computational demand and memory requirements
- We introduce a sparse approximation for the full covariance exploiting conditional independence.

The Convolution Process I.

- Consider a set of functions $\{f_q(\mathbf{x})\}_{q=1}^Q$.
- Each function can be expressed as

$$f_q(\mathbf{x}) = \int_{-\infty}^{\infty} k_q(\mathbf{x} - \mathbf{z})u(\mathbf{z})d\mathbf{z} = k_q(\mathbf{x}) \otimes u(\mathbf{x}).$$

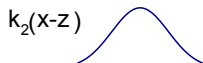
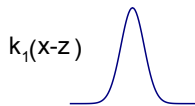
- Influence of more than one latent function, $\{u_r(\mathbf{z})\}_{r=1}^R$ and inclusion of an independent process $w_q(\mathbf{x})$

$$y_q(\mathbf{x}) = f_q(\mathbf{x}) + w_q(\mathbf{x}) = \sum_{r=1}^R \int_{-\infty}^{\infty} k_{qr}(\mathbf{x} - \mathbf{z})u_r(\mathbf{z})d\mathbf{z} + w_q(\mathbf{x}).$$

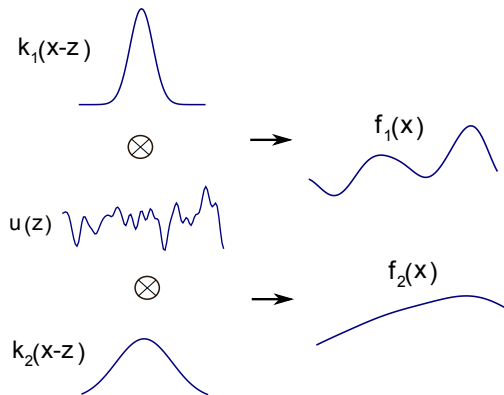
The Convolution Process II.



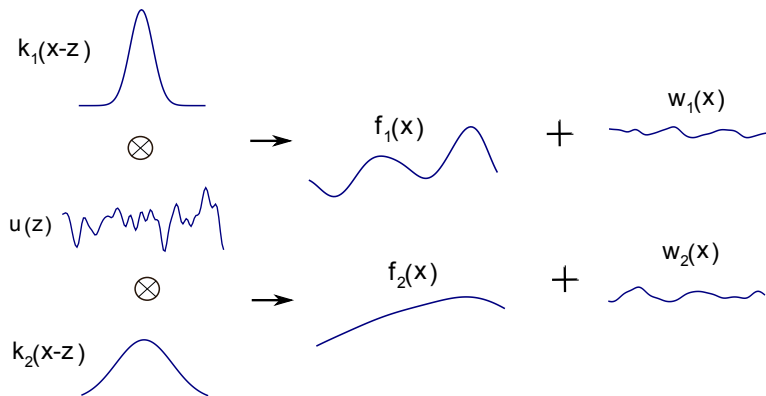
The Convolution Process II.



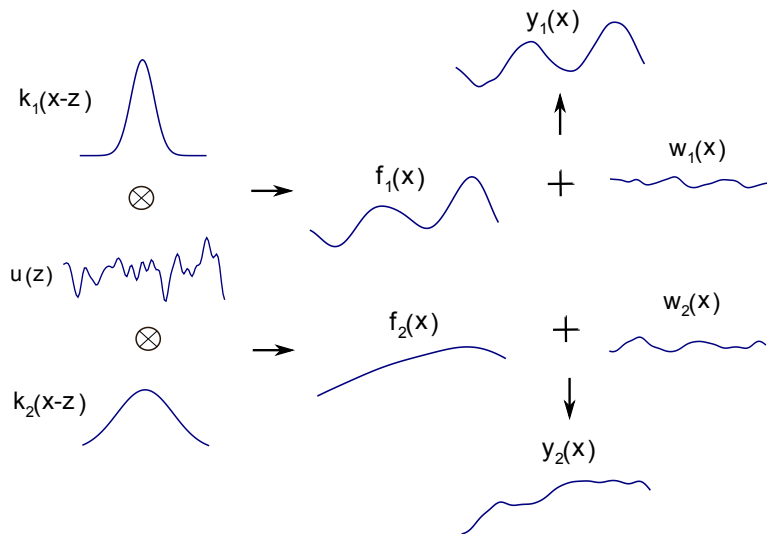
The Convolution Process II.



The Convolution Process II.



The Convolution Process II.



Covariance of the output functions.

The covariance between $y_q(\mathbf{x})$ and $y_s(\mathbf{x}')$ is given as

$$\text{cov} [y_q(\mathbf{x}), y_s(\mathbf{x}')] = \text{cov} [f_q(\mathbf{x}), f_s(\mathbf{x}')] + \text{cov} [w_q(\mathbf{x}), w_s(\mathbf{x}')] \delta_{qs}$$

where

$$\begin{aligned} \text{cov} [f_q(\mathbf{x}), f_s(\mathbf{x}')] &= \sum_{r,p}^R \int_{-\infty}^{\infty} k_{qr}(\mathbf{x} - \mathbf{z}) \int_{-\infty}^{\infty} k_{sp}(\mathbf{x}' - \mathbf{z}') \\ &\quad \times \text{cov} [u_r(\mathbf{z}), u_p(\mathbf{z}')] d\mathbf{z}' d\mathbf{z} \end{aligned}$$

Covariance of the latent functions.

- Independent *white noise processes*

$$\text{cov}[f_q, f_s] = \sum_{r=1}^R \int_{-\infty}^{\infty} k_{qr}(\mathbf{x} - \mathbf{z}) k_{sr}(\mathbf{x}' - \mathbf{z}) d\mathbf{z}$$

- Independent *Gaussian processes*

$$\text{cov}[f_q, f_s] = \sum_{r=1}^R \int_{-\infty}^{\infty} k_{qr}(\mathbf{x} - \mathbf{z}) \int_{-\infty}^{\infty} k_{sr}(\mathbf{x}' - \mathbf{z}') k_{u_r u_r}(\mathbf{z}, \mathbf{z}') d\mathbf{z}' d\mathbf{z}$$

- Covariance between output functions and latent functions

$$\text{cov}[f_q, u_r] = \int_{-\infty}^{\infty} k_{qr}(\mathbf{x} - \mathbf{z}') k_{u_r u_r}(\mathbf{z}', \mathbf{z}) d\mathbf{z}'$$

Likelihood of the full Gaussian process.

- The likelihood of the model is given by

$$p(\mathbf{y}|\mathbf{X}, \phi) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}} + \Sigma)$$

where $\mathbf{y} = [\mathbf{y}_1^\top, \dots, \mathbf{y}_Q^\top]^\top$ is the set of output functions, $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ covariance matrix with blocks $\text{cov}[f_q, f_s]$, Σ matrix of noise variances, ϕ is the set of parameters of the covariance matrix and $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ is the set of input vectors.

- Learning from the log-likelihood involves the inverse of $\mathbf{K}_{\mathbf{f},\mathbf{f}} + \Sigma$, which grows with complexity $\mathcal{O}(N^3 Q^3)$

Predictive distribution of the full Gaussian process.

- Predictive distribution at \mathbf{X}_*

$$p(\mathbf{y}_* | \mathbf{y}, \mathbf{X}, \mathbf{X}_*, \phi) = \mathcal{N}(\boldsymbol{\mu}_*, \boldsymbol{\Lambda}_*)$$

with

$$\boldsymbol{\mu}_* = \mathbf{K}_{\mathbf{f}_*, \mathbf{f}} (\mathbf{K}_{\mathbf{f}, \mathbf{f}} + \boldsymbol{\Sigma})^{-1} \mathbf{y}$$

$$\boldsymbol{\Lambda}_* = \mathbf{K}_{\mathbf{f}_*, \mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*, \mathbf{f}} (\mathbf{K}_{\mathbf{f}, \mathbf{f}} + \boldsymbol{\Sigma})^{-1} \mathbf{K}_{\mathbf{f}, \mathbf{f}_*} + \boldsymbol{\Sigma}$$

- Prediction is $\mathcal{O}(NQ)$ for the mean and $\mathcal{O}(N^2Q^2)$ for the variance.

The conditional independence assumption I.

- Our strategy for approximate inference is to exploit the natural conditional dependencies in the model.
- If we had observed the entire length of each latent function, $u_r(\mathbf{z})$, each $y_q(\mathbf{x})$ *would* be independent, *i.e.*

$$p(\{y_q(\mathbf{x})\}_{q=1}^Q | \{u_r(\mathbf{z})\}_{r=1}^R, \theta) = \prod_{q=1}^Q p(y_q(\mathbf{x}) | \{u_r(\mathbf{z})\}_{r=1}^R, \theta),$$

where θ are the parameters of the kernels and covariance functions.

The conditional independence assumption II.

- Our key assumption is that this independence will hold even if we have only observed M samples from $u_r(\mathbf{z})$ rather than the whole function.
- Define $\mathbf{u} = [\mathbf{u}_1^\top, \dots, \mathbf{u}_R^\top]^\top$ as samples from the latent functions, $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ as the cross-covariance matrix between the latent functions, $\mathbf{K}_{\mathbf{f},\mathbf{u}} = \mathbf{K}_{\mathbf{u},\mathbf{f}}^\top$ as the cross-covariance matrix between the latent and output functions and $\mathbf{Z} = \{\mathbf{z}_1, \dots, \mathbf{z}_M\}$ as the set of input vectors at which \mathbf{u} is evaluated.

The conditional independence assumption III.

- The output functions are conditional independent given the latent functions,

$$p(\mathbf{y}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \theta) = \prod_{q=1}^Q p(\mathbf{y}_q|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \theta) = \mathcal{N} \left(\mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \mathbf{D} + \Sigma \right)$$

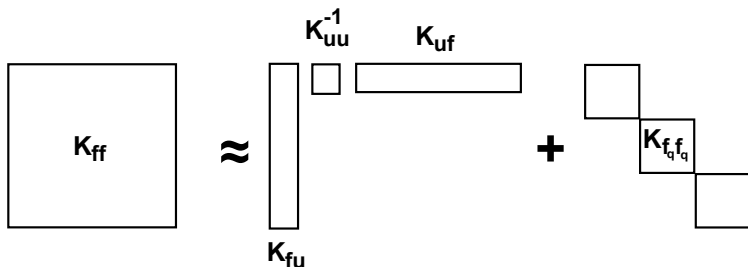
with $\mathbf{D} = \text{blockdiag} \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}} \right]$

- Integrating out the latent functions, the marginal likelihood is given as

$$p(\mathbf{y}|\mathbf{Z}, \mathbf{X}, \theta) = \mathcal{N} \left(\mathbf{0}, \mathbf{D} + \mathbf{K}_{\mathbf{f},\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u},\mathbf{f}} + \Sigma \right).$$

Comparison of covariances

- Full: $\mathbf{K}_{f,f}$
- Sparse: $\mathbf{K}_{f,u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f} + \text{blockdiag} [\mathbf{K}_{f,f} - \mathbf{K}_{f,u}\mathbf{K}_{u,u}^{-1}\mathbf{K}_{u,f}]$



Predictive distribution for the sparse approximation

Predictive distribution

$$p(\mathbf{y}_* | \mathbf{y}, \mathbf{X}, \mathbf{X}_*, \mathbf{Z}, \theta) = \mathcal{N}(\tilde{\boldsymbol{\mu}}_*, \tilde{\boldsymbol{\Lambda}}_*), \text{ with}$$

$$\tilde{\boldsymbol{\mu}}_* = \mathbf{K}_{\mathbf{f}_*, \mathbf{u}} \mathbf{A}^{-1} \mathbf{K}_{\mathbf{u}, \mathbf{f}} (\mathbf{D} + \boldsymbol{\Sigma})^{-1} \mathbf{y}$$

$$\tilde{\boldsymbol{\Lambda}}_* = \mathbf{D}_* + \mathbf{K}_{\mathbf{f}_*, \mathbf{u}} \mathbf{A}^{-1} \mathbf{K}_{\mathbf{u}, \mathbf{f}_*} + \boldsymbol{\Sigma}$$

$$\mathbf{A} = \mathbf{K}_{\mathbf{u}, \mathbf{u}} + \mathbf{K}_{\mathbf{u}, \mathbf{f}} (\mathbf{D} + \boldsymbol{\Sigma})^{-1} \mathbf{K}_{\mathbf{f}, \mathbf{u}}$$

$$\mathbf{D}_* = \text{blockdiag} \left[\mathbf{K}_{\mathbf{f}_*, \mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*, \mathbf{u}} \mathbf{K}_{\mathbf{u}, \mathbf{u}}^{-1} \mathbf{K}_{\mathbf{u}, \mathbf{f}_*} \right]$$

Computational requirements

- For learning the computational demand is in the calculation of \mathbf{D}^{-1} , which grows as $\mathcal{O}(N^3Q) + \mathcal{O}(NQM)$ (with $R = 1$).
- For inference, the computation of the mean grows as $\mathcal{O}(QM)$ and the computation of the variance as $\mathcal{O}(QM^2)$, after some pre-computations.

Remarks

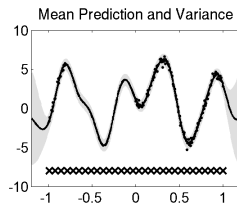
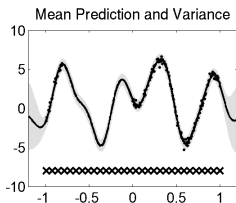
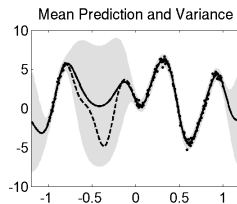
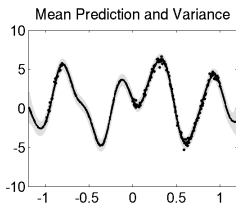
- The functional form of the approximation is almost identical to that of the Partially Independent Training Conditional (PITC) approximation [QCR05]. However, in PITC
 - it is not obvious which variables should be grouped together.
 - the inducing variables live in the same space as the observations.
- Assuming

$$p(\mathbf{y}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \theta) = \prod_{q=1}^Q \prod_{n=1}^N p(y_{qn}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \theta)$$

leads to the Fully Independent Training Conditional (FITC) approximation [QCR05, SG06].

Toy problem I

Four outputs generated from the full GP ($Q = 4$).



Toy problem II

Training observations: 200, testing observations: 300, both for each output.

Method	Output 1	Output 2	Output 3	Output 4
Full GP	1.07 ± 0.08	0.99 ± 0.03	1.12 ± 0.07	1.05 ± 0.07
FITC	1.08 ± 0.09	1.00 ± 0.03	1.13 ± 0.07	1.04 ± 0.07
PITC	1.07 ± 0.08	0.99 ± 0.03	1.12 ± 0.07	1.05 ± 0.07

Table: Standardized mean square error (SMSE) for ten runs of the toy problem. All numbers are to be multiplied by 10^{-2}

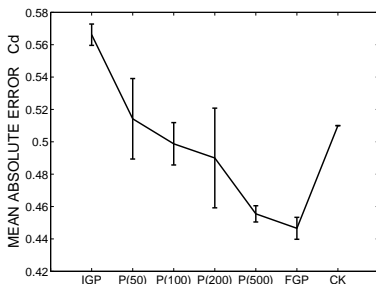
Jura Data set I

- Measurements of concentrations of seven heavy metals collected in the topsoil of a 14.5 km² region of the Swiss Jura.
- Prediction set (259 locations) and a validation set (100 locations).

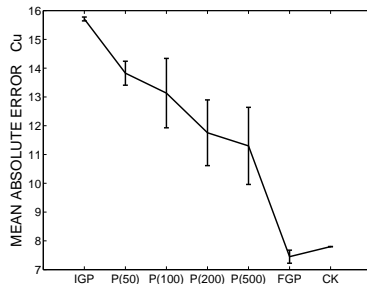
Primary variable	Secondary Variables
Cd	Ni, Zn
Cu	Pb, Ni, Zn

- Optimisation of the locations of the inducing inputs.

Jura Data set II



(a) Cadmium



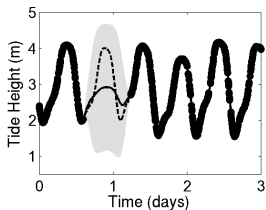
(b) Copper

Figure: Mean absolute error for IGP: Independent GP, P(M): PITC with M inducing points, FGP: Full GP, CK: Ordinary Co-kriging

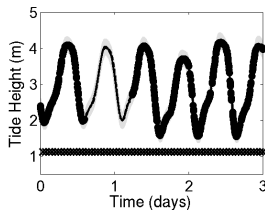
Sensor network dataset I

- Weather data collected from a sensor network located on the south coast of England.
- The network includes four sensors (named Bramblemet, Sotonmet, Cambermet and Chimet) each of which measures several environmental variables. See [ROR⁺08].
- We have selected one of the sensors signals, tide height, and applied the PITC approximation scheme with an additional *squared exponential* independent kernel.
- Training observations: 1000, testing observations: 3320, both for each output.

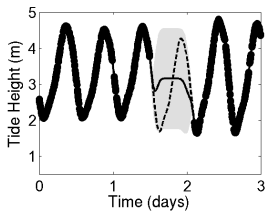
Sensor network dataset II



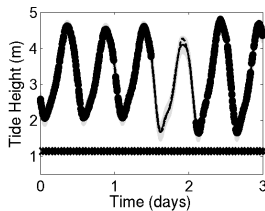
(a) Bramblemet with independent GP



(b) Bramblemet using PITC



(c) Cambermet with independent GP



(d) Cambermet using PITC

Summary

- We have presented a sparse approximation for multiple output Gaussian processes, capturing the correlated information among outputs and reducing the amount of computational load for prediction and optimization purposes.
- The computational complexity for the PITC approximation matches the computational complexity for modelling with independent Gaussian processes. However, with the independence assumption the missing ranges in the previous experiments are not accurately captured.
- Model selection issues
 - Size of the active set (Independent GP when $M = N$)
 - How to select the active set

Acknowledgments

We thank the authors of [ROR⁺08] who kindly made the sensor network database available.

References I



P. Boyle and M. Freaun.

Dependent Gaussian processes.

In Y. Weiss L. K. Saul and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 217–224. The MIT Press, 2005.



D. Higdon.

Space and space-time modelling using process convolutions.

In P. Chatwin C. Anderson, V. Barnett and A. El-Shaarawi, editors, *Quantitative methods for current environmental issues*, pages 37–56. Springer Verlag, 2005.

References II



J. Quiñero-Candela and C. E. Rasmussen.

A Unifying View of Sparse Approximate Gaussian Process Regression.

Journal of Machine Learning Research, 6:1939–1959, 2005.



A. Rogers, M. A. Osborne, S. D. Ramchurn, S. J. Roberts, and N. R. Jennings.

Towards Real-Time Information Processing of Sensor Network Data using Computationally Efficient Multi-output Gaussian Processes.

In Proceedings of the International Conference on Information Processing in Sensor Networks (IPSN 2008), 2008.

In press.

References III



E. Snelson and Z. Ghahramani.

Sparse Gaussian Processes using Pseudo-inputs.

In B. Schölkopf Y. Weiss and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1257–1264. MIT Press, Cambridge, MA, 2006.