CONVOLVED GAUSSIAN PROCESS PRIORS FOR MULTIVARIATE REGRESSION WITH APPLICATIONS TO DYNAMICAL SYSTEMS

A thesis submitted to the University of Manchester
for the degree of Doctor of Philosophy
in the Faculty of Engineering and Physical Sciences

2011

Mauricio A. Álvarez

School of Computer Science

Contents

Word count 45174

List of Figures

List of Tables

Abstract

In this thesis we address the problem of modeling correlated outputs using Gaussian process priors. Applications of modeling correlated outputs include the joint prediction of pollutant metals in geostatistics and multitask learning in machine learning. Defining a Gaussian process prior for correlated outputs translates into specifying a suitable covariance function that captures dependencies between the different output variables. Classical models for obtaining such a covariance function include the linear model of coregionalization and process convolutions. We propose a general framework for developing multiple output covariance functions by performing convolutions between smoothing kernels particular to each output and covariance functions that are common to all outputs. Both the linear model of coregionalization and the process convolutions turn out to be special cases of this framework. Practical aspects of the proposed methodology are studied in this thesis. They involve the use of domain-specific knowledge for defining relevant smoothing kernels, efficient approximations for reducing computational complexity and a novel method for establishing a general class of nonstationary covariances with applications in robotics and motion capture data.
Reprints of the publications that appear at the end of this document, report case studies and experimental results in sensor networks, geostatistics and motion capture data that illustrate the performance of the different methods proposed.

Declaration

I hereby declare that no portion of the work referred to in the thesis has been submitted in support of an application for another degree or qualification of this or any other university or other institute of learning.

Copyright

i. The author of this thesis (including any appendices and/or schedules to this thesis) owns certain copyright or related rights in it (the "Copyright") and s/he has given The University of Manchester certain rights to use such Copyright, including for administrative purposes.

ii. Copies of this thesis, either in full or in extracts and whether in hard or electronic copy, may be made **only** in accordance with the Copyright, Designs and Patents Act 1988 (as amended) and regulations issued under it or, where appropriate, in accordance with licensing agreements which the University has from time to time. This page must form part of any such copies made.

iii. The ownership of certain Copyright, patents, designs, trade marks and other intellectual property (the "Intellectual Property") and any reproductions of copyright works in the thesis, for example graphs and tables ("Reproductions"), which may be described in this thesis, may not be owned by the author and may be owned by third parties. Such Intellectual Property and Reproductions cannot and must not be made available for use without the prior written permission of the owner(s) of the relevant Intellectual Property and/or Reproductions.

iv. Further information on the conditions under which disclosure, publication and commercialisation of this thesis, the Copyright and any Intellectual Property and/or Reproductions described in it may take place is available in the University IP Policy (see `http://www.campus.manchester.ac.uk/medialibrary/policies/intellectual-property.pdf`), in any relevant Thesis restriction declarations deposited in the University Library, The University Library's regulations (see `http://www.manchester.ac.uk/library/aboutus/regulations`) and in The University's policy on presentation of Theses

Acknowledgements

Foremost, I would like to say thanks to my supervisor Neil Lawrence. Many steps in this long journey have only been possible due to his constant support and confidence. Neil encouraged me to think beyond given paradigms and was always a source of thoughtful advice and contagious enthusiasm.
Thanks to Michalis Titsias for his generous collaboration. Michalis was constantly keen to discuss research ideas or provide some light over concepts that at some point looked obscure to me. This work has also benefited greatly from discussions with David Luengo and Magnus Rattray, who were always willing to share their opinions and propose alternative research routes.
I'd also like to thank Jan Peters and Bernhard Schölkopf for giving me the opportunity to spend a couple of very productive months in a place of research excellence: the Department of Empirical Inference at MPI.
To my friends Richard, Michalis, Nicolò and Kevin for the trips, the good food and the beers and to all the people in the MLO group for offering me with a very pleasant environment to develop my work.
I take this opportunity to acknowledge my sponsors: the Overseas Research Student Award scheme, the School of Computer Science, the Google Research Award "Mechanistically Inspired Convolution Processes for Learning", the EPSRC Grant No EP/F005687/1 "Gaussian Processes for Systems Identification with Applications in Systems Biology", the Internal Visiting Programme, Conference & Workshop Organisation Programme and the Conference & Workshop Attendance Programme of the FP7 EU Network of Excellence PASCAL 2.

Thanks to my family for their permanent attention and for being there for me.

I would like to dedicate this work to Marcela. This thesis is yours as much as it is mine.

Notation

Mathematical notation

Generalities

p	dimensionality of the input space
D	number of outputs
K	number of inducing points
N	number of data points per output
Q	number of latent functions
$\mathcal{X}$	input space
$\mathbb{D}$	set of the integer numbers $\{1, 2, \dots, D\}$
$\mathbf{X}$	input training data, $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$
$\mathbf{Z}$	set of inducing inputs, $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^K$

Operators

$\text{cov}[\cdot, \cdot]$	covariance operator
$\text{E}[\cdot]$	expected value
$\text{tr}(\cdot)$	trace of a matrix
$\text{vec}(\mathbf{A})$ or $\mathbf{A}$:	vectorization of matrix $\mathbf{A}$
$\mathbf{A} \otimes \mathbf{B}$	Kronecker product between matrices $\mathbf{A}$ and $\mathbf{B}$
$\mathbf{A} \odot \mathbf{B}$	Hadamard product between matrices $\mathbf{A}$ and $\mathbf{B}$

Functions

$u_q(\mathbf{x})$	q-th latent function or latent process evaluated at $\mathbf{x}$
$k_q(\mathbf{x}, \mathbf{x}')$, $k_{u_q, u_q}(\mathbf{x}, \mathbf{x}')$	covariance function for the Gaussian process of $u_q(\mathbf{x})$
$f_d(\mathbf{x})$	d-th output evaluated at $\mathbf{x}$
$\mathbf{f}(\mathbf{x}_i)$, $\overline{\mathbf{f}}_i$	vector-valued function, $\mathbf{f}(\mathbf{x}_i) = [f_1(\mathbf{x}_i), \dots, f_D(\mathbf{x}_i)]^\top$
$k_{f_d, u_q}(\mathbf{x}, \mathbf{x}')$	cross-covariance between output $f_d(\mathbf{x})$ and function $u_q(\mathbf{x}')$

$k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$	cross-covariance between output $f_d(\mathbf{x})$ and output $f_{d'}(\mathbf{x}')$
$\delta_{k,k'}$	Kronecker delta for discrete arguments
$\delta(x)$	Dirac delta for continuous arguments

Vectors and matrices

$\mathbf{u}_q$	$u_q(\mathbf{x})$ evaluated at $\mathbf{X}$ or $\mathbf{Z}$, $\mathbf{u}_q = [u_q(\mathbf{z}_1),\dots,u_q(\mathbf{z}_K)]^\top$
$\mathbf{K}_q, \mathbf{K}_{\mathbf{u}_q,\mathbf{u}_q}$	covariance matrix with entries $k_q(\mathbf{x},\mathbf{x}')$ evaluated at $\mathbf{X}$ or $\mathbf{Z}$
$\mathbf{f}_d$	$f_d(\mathbf{x})$ evaluated at $\mathbf{X}$, $\mathbf{f}_d = [f_d(\mathbf{x}_1),\dots,f_d(\mathbf{x}_N)]^\top$
$\mathbf{f}$	vectors $\{\mathbf{f}_d\}_{d=1}^D$, stacked in a column vector
$\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{x},\mathbf{x}')$	covariance matrix with entries $k_{d,d'}(\mathbf{x},\mathbf{x}')$ with $d,d' \in \mathbb{D}$
$\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$	covariance matrix with entries $k_{d,d'}(\mathbf{x}_n,\mathbf{x}_m)$ with $\mathbf{x}_n,\mathbf{x}_m \in \mathbf{X}$
$\mathbf{K}_{\mathbf{f},\mathbf{f}}$	covariance matrix with blocks $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$ with $d,d' \in \mathbb{D}$
$\mathbf{K}_{\mathbf{f}_d,\mathbf{u}_q}$	cross-covariance matrix with elements $k_{f_d,u_q}(\mathbf{x},\mathbf{x}')$
$\mathbf{K}_{\mathbf{f},\mathbf{u}}$	cross-covariance matrix with blocks $\mathbf{K}_{\mathbf{f}_d,\mathbf{u}_q}$
$\mathbf{I}_N$	identity matrix of size N

Abbreviations

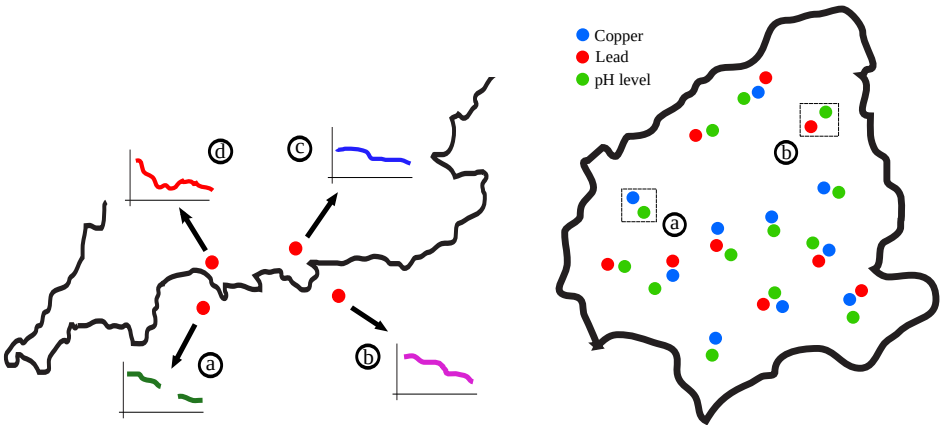
GP	Gaussian Process
LMC	Linear Model of Coregionalization
ICM	Intrinsic Coregionalization Model
SLFM	Semiparametric Latent Factor Model
MTGP	Multi-task Gaussian Processes
PC	Process Convolution
CMOC	Convolved Multiple Output Covariance
LFM	Latent Force Model
DTC	Deterministic Training Conditional
FIC	Fully Independent Conditional
FITC	Fully Independent Training Conditional
PIC	Partially Independent Conditional
PITC	Partially Independent Training Conditional
FI(T)C	either FIC, FITC or both
PI(T)C	either PIC, PITC or both
VIK	Variational Inducing Kernel
DTCVAR	Deterministic Training Conditional Variational
SDLFM	Switched Dynamical Latent Force Model

Chapter 1

Introduction

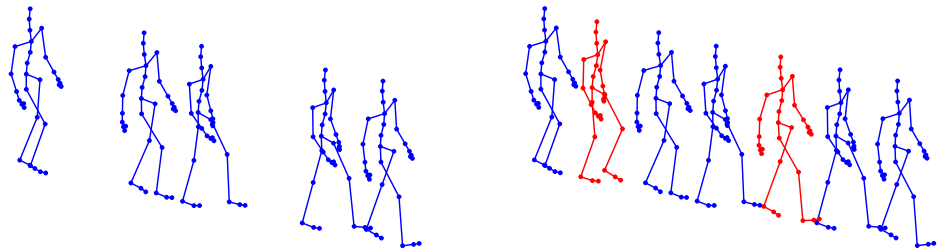
Accounting for dependencies between related processes has important applications in several areas. In *sensor networks*, for example, missing signals from certain sensors may be predicted by exploiting their correlation with observed signals acquired from other sensors (Osborne et al., 2008), as shown in figure 1.1(a). Figure 1.1(a) represents a sketch of the south coast of England, where several sensors (the red dots in the figure), that keep track of different environmental variables, such as temperature and air pressure, have been placed along the coastline. In a normal scenario, we have access to the readings of all these devices at all time instants. However, at some random points in time, a number of sensors can fail or segments of the sensor network can suffer disruptions, rendering inaccessible the information of certain environmental variables. Given that the sensors are located sufficiently close to each other and many of them make similar readings, we can make use of the signals obtained from the unbroken sensors (in figure 1.1(a), sensors b, c and d) to predict the missing signals for the broken ones (in figure 1.1(a), sensor a).

In *geostatistics*, predicting the concentration of heavy pollutant metals, which are expensive to measure, can be done using inexpensive and oversampled variables as a proxy (Goovaerts, 1997). Figure 1.1(b) illustrates a region of the Swiss Jura, for which we would like to know when the concentration of certain heavy metals, goes beyond a toxic threshold that can be risky for human health. It is usually cheaper to measure the level of pH in the soil (the green dots in the figure) than the concentration of Copper or Lead (the blue and read dots in the figure). We can exploit the correlations between the metals and pH level learned from different locations to predict, for example, the value of Lead in input location a,



(a) A network of sensors (b) Pollutant metals concentrations

Figure 1.1: Examples of the type of problems that we consider in this thesis.



(a) Walking movement with missing poses (b) Frames have been filled with plausible poses

Figure 1.2: An example of a walking exercise. In 1.2(a), there are missing poses that are filled in 1.2(b).

shown in figure 1.1(b) or the value of Copper in input location b.

In *computer graphics*, a common theme is the animation and simulation of physically plausible humanoid motion. As shown in figure 1.2(a), given a set of poses that delineate a particular movement (for example, walking), we are faced with the task of completing a sequence by filling in the missing frames with natural-looking poses, as in figure 1.2(b). Human movement exhibits a high-degree of correlation. Think, for example, of the way we walk. When moving the right leg forward, we unconsciously prepare the left leg, which is currently touching the ground, to start moving as soon as the right leg reaches the floor. At the same time, our hands move synchronously with our legs. We can exploit these implicit correlations for predicting new poses and for generating new natural-looking walking sequences.

In the examples above, it would be possible to assume that we do not have

access to secondary information (in the geostatistics example, pH levels) and consequently employ models that make predictions individually for each variable (Copper and Lead). However, these examples share an underlying principle: it is possible to exploit the interaction between the different variables to improve their joint prediction. Within the machine learning community this type of modeling is sometimes referred to as *multitask learning*. The idea in multitask learning is that information shared between different tasks can lead to improved performance in comparison to learning the same tasks individually. It also refers to systems that learn by transferring knowledge between different domains, for example, what can we learn about running through seeing walking? Therefore it is also known as "transfer learning" (Thrun, 1996; Caruana, 1997; Bonilla et al., 2008).

There are plenty of methods in the literature that have been used to approach the type of problems we have described, including neural networks (Caruana, 1997), Bayesian neural networks (Bakker and Heskes, 2003), Dirichlet process priors (Xue et al., 2007) and support vector machines (Evgeniou et al., 2005).

Essentially, this sort of problems conform to a *multivariate regression* analysis.[1] If we assume that the variables are independent given the inputs, then, the simultaneous regression transforms to a series of single variable regression problems. Nowadays, the most established technology for univariate regression, within the machine learning community, corresponds to *Gaussian process* (GP) regression (Rasmussen and Williams, 2006).

A Gaussian process specifies a *prior distribution* over functions $f(\mathbf{x})$, with $\mathbf{x} \in \Re^p$. The distribution is defined in terms of a positive semidefinite function $k(\mathbf{x}, \mathbf{x}')$, known as the *covariance function*, that encodes the degree of similarity or correlation between $f(\mathbf{x})$ and $f(\mathbf{x}')$ as a function of the inputs $\mathbf{x}$ and $\mathbf{x}'$. Covariance functions for single outputs are widely studied in machine learning (see, for example, Rasmussen and Williams, 2006) and some examples include the squared-exponential or the Matérn class of covariance functions. From a *Bayesian statistics* point of view, the Gaussian process specifies our prior beliefs about the properties of the functions we are modeling. Our beliefs are updated in the presence of data by means of a *likelihood function*, that relates our prior assumptions to the actual observations, leading to an updated distribution, the *posterior distribution*, that can be used, for example, for predicting test cases.

[1] Multivariate problems are also known as multivariable, multiple output or multiple response problems.

In this thesis, we consider the problem of extending the Gaussian process framework for modeling correlated outputs. The main challenge is the definition of a covariance function that encodes not only the degree of correlation of a process $f_d(\mathbf{x})$ as function of the input $\mathbf{x}$, but, concomitantly, expresses the correlation between two different processes $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x})$, for $d \neq d'$. Importantly, the covariance function must be valid, that is, positive semidefinite.

Much of the common practice in univariate Gaussian process regression, as it is done today in machine learning, has been rigorously systematized in Rasmussen and Williams (2006). Except for some isolated attempts, the counterpart for the multivariate case is yet to be written. In this thesis, we introduce some ideas towards that direction, from both theoretical and applied perspectives.

Outline of the thesis and contributions

One of the paradigms that has been considered for extending Gaussian processes to the multivariable scenario (Teh et al., 2005; Osborne et al., 2008; Bonilla et al., 2008) is known in the geostatistics literature as the *linear model of coregionalization* (LMC) (Journel and Huijbregts, 1978; Goovaerts, 1997). In the LMC the covariance function is expressed as the sum of products between *coregionalization matrices* and a set of underlying covariance functions. The correlations across the outputs are expressed in the coregionalization matrices, while the underlying covariance functions express the correlation between different data points in terms of the input vectors.

An alternative approach to constructing covariance functions for multiple outputs employs *process convolutions* (PC). To obtain a PC in the single output case, the output of a given process is convolved with a smoothing kernel function. For example, a white noise process may be convolved with a smoothing kernel to obtain a covariance function (Barry and Ver Hoef, 1996). Ver Hoef and Barry (1998) noted that if a single input process was convolved with different smoothing kernels to produce different outputs, then correlation between the outputs could be expressed. This idea was introduced to the machine learning audience by Boyle and Frean (2005a).

Although these base processes have been considered almost exclusively as white noise Gaussian processes, in this thesis we allow the bases processes to be Gaussian processes with more general covariance functions. The **first contribution**

in this thesis is to develop a unifying model for the covariance function for multiple outputs that contains the linear model of coregionalization and the process convolution as special cases. We refer to this model as the *convolved multiple output covariance* (CMOC). In chapter 2, we arrive at this covariance function by building upon previous work in the linear model of coregionalization literature and the process convolution literature.

The convolved multiple output covariance is obtained by convolving covariance functions with smoothing kernels. Usually it is difficult to specify in advance a functional form for the smoothing kernel that results in meaningful covariance functions, in the sense that the resulting multiple output covariance can represent important features of the data. Drawing connections with the theory of *differential equations* as in Lawrence et al. (2007), our **second contribution** is that we develop a general framework in which the smoothing kernels correspond to the *Green's function* associated to the differential equations used to describe the system. In chapter 3, we develop this idea under the name of *latent force models*.

A Gaussian process is a nonparametric technique and due to this nonparametric nature it carries the cross of being computationally expensive to use in practice. The expensive steps are related to the successive inversion of the covariance matrix computed from the covariance function of the Gaussian process. In the multiple output case, the computational complexity grows as $O(D^3N^3)$, where N is the number of observations per output and D is the number of outputs. Our **third** and **fourth contributions** are the development of efficient approximation techniques that reduce computational complexity to $O(DNK^2)$, where K is a user-specified parameter. In the third contribution, we develop *reduced rank approximations* for the covariance matrix of the full Gaussian process by exploiting different conditional independence assumptions in the likelihood function. In the fourth contribution, we introduce the concept of *inducing function*. An inducing function acts as a smooth surrogate for white noise processes, when these processes are used as base functions in the CMOC. Embedding these inducing functions in a variational framework, replicating ideas presented in Titsias (2009), we develop an efficient approximation that behaves as a lower bound of the marginal likelihood of the full multivariate Gaussian process. Both contributions and the relationships between them are presented in chapter 4.

Finally, in chapter 5 we present our **fifth contribution**. We propose a novel model that allows discrete changes in the parameter space of the smoothing kernel

and/or the parameter space of the covariances of the base processes. Our main focus is to develop a latent force model in which the parameters of the multivariate covariance change as a function of the input variable. Thus, the model obtained allows for the description of highly nonstationary multivariate time series courses.

How to read this thesis

This thesis follows the Alternative Format Thesis allowed by the University of Manchester thesis submission regulations,[2] that consents to incorporate sections that are in a format suitable for submission for publication in a peer-reviewed journal. The way in which the thesis has been developed is as follows. The main document serves as a backbone of a series of contributions already published at the *Annual Conference on Neural Information Processing Systems (NIPS)*, the *International Conference on Artificial Intelligence and Statistics (AISTATS)*, a journal paper at the *Journal of Machine Learning Research* and a technical report appearing in Álvarez et al. (2009). Each chapter includes in the introduction a remark that explains what sections of the chapter have been published. We then describe the theory involved in that chapter and include an experiment that illustrates the main ideas developed. At the end of each chapter, we comment about further experiments that accompany the theory and that are found in the publications. Reprints of the publications are attached at the end of the document.

We refer to the publications using the numbers in the list of publications of the following section. So for example, we will use expressions like "in publication **iv**" to refer to the publication [**iv**] in the list below.

List of publications

The main contributions in the thesis have been presented in the following publications and a submitted paper.

[**i**] Mauricio A. Álvarez and Neil D. Lawrence (2008): Sparse Convolved Gaussian Processes for Multi-output Regression, in D. Koller and D. Schuurmans and Y. Bengio and L. Bottou (Eds), Advances in Neural Information Processing Systems 21, pp 57-64, 2009.

[2] http://www.campus.manchester.ac.uk/researchoffice/graduate/ordinancesandregulations

[**ii**] Mauricio A. Álvarez, David Luengo and Neil D. Lawrence. Latent Force Models, in D. van Dyk and M. Welling (Eds.), Proceedings of The Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS) 2009, JMLR: W&CP 5, pp. 9-16, Clearwater Beach, Florida, April 16-18, 2009.

[**iii**] Mauricio A. Álvarez, David Luengo, Michalis K. Titsias and Neil D. Lawrence (2010): Efficient Multioutput Gaussian Processes through Variational Inducing Kernels, in Y. Whye Teh and M. Titterington (Eds.), Proceedings of The Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, JMLR: W&CP 9, pp. 25-32, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010.

[**iv**] Mauricio A. Álvarez, Jan Peters, Bernhard Schölkopf and Neil D. Lawrence (2011): Switched latent force models for movement segmentation, in J. Shawe-Taylor, R. Zemel, C. Williams and J. Lafferty (Eds), Advances in Neural Information Processing Systems 23, pp 55-63, 2011. See also the supplementary material accompanying the publication.

[**v**] Mauricio A. Álvarez and Neil D. Lawrence (2011): "Computationally Efficient Convolved Multiple Output Gaussian Processes", *Journal of Machine Learning Research 12*, pp 1425–1466.

In all publications, Álvarez had the main responsibility in writing a first draft of the paper and developing the software. Revisions of the writing were incorporated by the coauthors directly or by Álvarez after discussions with the other authors. For publication **ii**, Luengo developed the analytical expression for the covariance function of the second order latent force model. For publication **iii**, Luengo developed the covariance function for a latent force model driven by white noise and Titsias helped with the description of the variational framework. Publication **iv** was supervised by Peters, Schölkopf and Lawrence. All the other publications were supervised by Lawrence.

Chapter 2

Covariance functions for multivariate regression

In chapter 1, we discussed applications of multivariate regression that are encountered in machine learning problems, including multitask learning (see Bonilla et al., 2008, for example). In geostatistics these models are used for jointly predicting the concentration of different heavy metal pollutants (Goovaerts, 1997). In statistics more researchers are becoming interested in emulation of multiple output simulators (see Higdon et al., 2008; Rougier, 2008; Conti and O'Hagan, 2010, for example). In this chapter we provide a general review of structured covariance/kernel functions for multiple outputs.

There is a huge amount of work in geostatistics focused on constructing valid covariance functions for predicting spatial varying data. The basic approach is based on the so called "Linear Model of Coregionalization" (Journel and Huijbregts, 1978; Goovaerts, 1997). Similar methods have been suggested in several machine learning and statistics related papers, including special type of kernels proposed as a generalization of the regularization theory to vector-valued functions. We show how some of those methods can be seen as particular cases of the linear model of coregionalization.

An alternative approach for constructing the covariance function involves a moving average construction in the form of "Process Convolutions" (Ver Hoef and Barry, 1998; Higdon, 2002). In a process convolution a latent process is convolved with output-specific smoothing kernels to produce a valid covariance. The latent process is usually assumed to be a white noise process. If the latent process follows a Gaussian process with general covariance function, we will see that the

linear model of coregionalization and the process convolution framework can be interpreted as particular cases of this moving-average construction. We refer to this covariance as the "Convolved Multiple Output Covariance" (CMOC).

Having firstly presented alternatives for constructing multivariate kernel functions, we then embed these kernels in a Gaussian process prior and explain two important aspects of multivariate Gaussian process regression, namely, how to perform parameter estimation and prediction for test data. We also briefly review how parameter estimation and prediction is done in research areas such as geostatistics and statistics.

The chapter is organized as follows. In section 2.1, different methods for constructing the kernel for multiple outputs are reviewed, including the linear model of coregionalization and process convolutions. We then employ the defined covariances for multivariate regression with Gaussian process priors in section 2.2. Finally, in section 2.3, we present an example of multivariate regression in gene expression data.

Remark. In publication **v**, we introduced the main idea that appears in section 2.1 and that is the motivation for this chapter. Detailed analysis of related work, including the linear model of coregionalization in computer emulation, is new, though. The example of section 2.3 also appears in publication **v**.

2.1 Kernels for multiple outputs

In geostatistics, prediction over multivariate output data is known as cokriging. Geostatistical approaches to multivariate modelling are mostly formulated around the "linear model of coregionalization" (LMC, see, e.g., Journel and Huijbregts, 1978; Wackernagel, 2003). We will first consider this model and discuss how several recent models proposed in the machine learning and statistics literature are special cases of the LMC, including approaches to constructing "multitask" kernels in machine learning introduced from the perspective of regularization theory (Evgeniou and Pontil, 2004). We also review different alternatives for the moving average construction of the covariance function, under the generic name of process convolutions and introduce the model for the covariance function that is used in the thesis.

2.1.1 The linear model of coregionalization

In the linear model of coregionalization, the outputs are expressed as linear combinations of independent random functions. This is done in such a way that ensures that the resulting covariance function (expressed jointly over all the outputs and the inputs) is a valid positive semidefinite function. Consider a set of D variables $\{f_d(\mathbf{x})\}_{d=1}^{D}$ with $\mathbf{x} \in \Re^p$. In the LMC, each variable f_d is expressed as (Journel and Huijbregts, 1978)

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} a_{d,q} u_q(\mathbf{x}) + \mu_d,$$

where μ_d represents the mean of each process $f_d(\mathbf{x})$ and the functions $u_q(\mathbf{x})$, with $q = 1, \ldots, Q$, have mean equal to zero and covariance $\text{cov}[u_q(\mathbf{x}), u_{q'}(\mathbf{x}')] = k_q(\mathbf{x}, \mathbf{x}')\delta_{q,q'}$, where $\delta_{q,q'}$ is the Kronecker delta ($\delta_{q,q'} = 1$ if $q = q'$ and $\delta_{q,q'} = 0$ if $q \neq q'$). Therefore, the processes $\{u_q(\mathbf{x})\}_{q=1}^{Q}$ are independent. We will assume that $\mu_d = 0$ for all outputs, unless it is stated otherwise. Some of the basic processes $u_q(\mathbf{x})$ and $u_{q'}(\mathbf{x}')$ can have the same covariance $k_q(\mathbf{x}, \mathbf{x}')$, $k_q(\mathbf{x}, \mathbf{x}') = k_{q'}(\mathbf{x}, \mathbf{x}')$ while remaining orthogonal. A similar expression for $\{f_d(\mathbf{x})\}_{d=1}^{D}$ can be written grouping the functions $u_q(\mathbf{x})$ which share the same covariance (Journel and Huijbregts, 1978; Goovaerts, 1997)

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} a_{d,q}^i u_q^i(\mathbf{x}), \tag{2.1}$$

where the functions $u_q^i(\mathbf{x})$, with $q = 1, \ldots, Q$ and $i = 1, \ldots, R_q$, have mean equal to zero and covariance $\text{cov}[u_q^i(\mathbf{x}), u_{q'}^{i'}(\mathbf{x}')] = k_q(\mathbf{x}, \mathbf{x}')\delta_{i,i'}\delta_{q,q'}$. Expression (2.1) means that there are Q groups of functions $u_q^i(\mathbf{x})$ and, within each group, functions $u_q^i(\mathbf{x})$ share the same covariance, but are independent. We assume that the processes $f_d(\mathbf{x})$ are second-order stationary.[1] The cross covariance between any two functions $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x})$ is given in terms of the covariance functions

[1] The stationarity condition is introduced so that the prediction stage can be realized through a linear predictor using a single realization of the process (Cressie, 1993). Implicitly, ergodicity is also assumed. For nonstationary processes, description is done in terms of the so called *variogram functions* (Matheron, 1973).

for $u_q^i(\mathbf{x})$

$$\text{cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}+\mathbf{h})] = \sum_{q=1}^{Q}\sum_{q'=1}^{Q}\sum_{i=1}^{R_q}\sum_{i'=1}^{R_q} a_{d,q}^i a_{d',q'}^{i'} \,\text{cov}[u_q^i(\mathbf{x}), u_{q'}^{i'}(\mathbf{x}+\mathbf{h})],$$

with $\mathbf{h} = \mathbf{x}-\mathbf{x}'$ being the lag vector. We refer to the covariance $\text{cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}+\mathbf{h})]$ as $k_{f_d,f_{d'}}(\mathbf{h})$. Due to the independence of the functions $u_q^i(\mathbf{x})$, the above expression reduces to

$$k_{f_d,f_{d'}}(\mathbf{h}) = \sum_{q=1}^{Q}\sum_{i=1}^{R_q} a_{d,q}^i a_{d',q}^i k_q(\mathbf{h}) = \sum_{q=1}^{Q} b_{d,d'}^q k_q(\mathbf{h}), \tag{2.2}$$

with $b_{d,d'}^q = \sum_{i=1}^{R_q} a_{d,q}^i a_{d',q}^i$. For the D outputs, equation (2.1) can be expressed in matrix form as

$$\mathbf{f}(\mathbf{x}) = \sum_{q=1}^{Q} \mathbf{A}_q \mathbf{u}_q(\mathbf{x}),$$

where, for each $\mathbf{x}$, $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), \ldots, f_D(\mathbf{x})]^\top$, $\mathbf{A}_q \in \Re^{D\times R_q}$ is a matrix with entries $a_{d,q}^i$ and $\mathbf{u}_q(\mathbf{x}) = [u_q^1(\mathbf{x}), \ldots, u_q^{R_q}(\mathbf{x})]^\top$. The covariance function for $\mathbf{u}_q(\mathbf{x})$ is

$$\text{cov}[\mathbf{u}_q(\mathbf{x}), \mathbf{u}_{q'}(\mathbf{x}+\mathbf{h})] = k_q(\mathbf{h})\mathbf{I}_{R_q}\delta_{q,q'},$$

where $\mathbf{I}_{R_q} \in \Re^{R_q\times R_q}$ is the identity matrix.

The covariance function for the outputs is then given as

$$\begin{aligned}
\text{cov}[\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}+\mathbf{h})] &= \text{E}\left[\sum_{q=1}^{Q}\mathbf{A}_q\mathbf{u}_q(\mathbf{x})\left(\sum_{q'=1}^{Q}\mathbf{A}_{q'}\mathbf{u}_{q'}(\mathbf{x}+\mathbf{h})\right)^\top\right] \\
&= \sum_{q=1}^{Q}\sum_{q'=1}^{Q}\mathbf{A}_q\,\text{E}\left[\mathbf{u}_q(\mathbf{x})\mathbf{u}_{q'}^\top(\mathbf{x}+\mathbf{h})\right]\mathbf{A}_{q'}^\top \\
&= \sum_{q=1}^{Q}\mathbf{A}_q\mathbf{A}_q^\top k_q(\mathbf{h}).
\end{aligned} \tag{2.3}$$

Equation (2.3) can be written as

$$\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{h}) = \sum_{q=1}^{Q} \mathbf{B}_q k_q(\mathbf{h}), \tag{2.4}$$

where $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{h}) = \text{cov}[\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}+\mathbf{h})]$ and $\mathbf{B}_q = \mathbf{A}_q\mathbf{A}_q^\top$, with $\mathbf{B}_q \in \Re^{D\times D}$ is known as the *coregionalization matrix*. In general, we denote the covariance of $\mathbf{f}(\mathbf{x})$ as $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{x},\mathbf{x}') = \text{cov}[\mathbf{f}(\mathbf{x}), \mathbf{f}(\mathbf{x}')]$. For the stationary case, it reduces to $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{h})$. The elements of each $\mathbf{B}_q$ are the coefficients $b^q_{d,d'}$ appearing in equation (2.2). The covariance function $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{h})$ is positive semidefinite as long as the coregionalization matrices $\mathbf{B}_q$ are positive semi-definite and $k_q(\mathbf{h})$ is a valid covariance function. By definition, matrices $\mathbf{B}_q$ fulfill the positive semidefiniteness requirement and several models for the covariance function $k_q(\mathbf{h})$ can be used, for example the squared exponential covariance function, the Matérn class of covariance functions, among others (see Rasmussen and Williams, 2006, chap. 4).

Equation (2.1) can be interpreted as a nested structure (Wackernagel, 2003) in which the outputs $f_d(\mathbf{x})$ are first expressed as a linear combination of spatially uncorrelated processes $f_d(\mathbf{x}) = \sum_{q=1}^{Q} f_d^q(\mathbf{x})$, with $\text{E}[f_d^q(\mathbf{x})] = 0$ and $\text{cov}[f_d^q(\mathbf{x}), f_{d'}^{q'}(\mathbf{x}+\mathbf{h})] = b^q_{d,d'}k_q(\mathbf{h})\delta_{q,q'}$. At the same time, each process $f_d^q(\mathbf{x})$ can be represented as a set of uncorrelated functions weighted by the coefficients $a^i_{d,q}$, $f_d^q(\mathbf{x}) = \sum_{i=1}^{R_q} a^i_{d,q}u^i_q(\mathbf{x})$ where again, the covariance function for $u^i_q(\mathbf{x})$ is $k_q(\mathbf{h})$.

The linear model of coregionalization represents the covariance function as the sum of the products of two covariance functions. One of the covariance functions models the dependence between the functions, independently of the input vector $\mathbf{x}$, this is given by the coregionalization matrix $\mathbf{B}_q$, whilst the other covariance function models the input dependence, independently of the particular set of functions $f_d(\mathbf{x})$, this is the covariance function $k_q(\mathbf{h})$. In equation (2.4), the output covariance for a particular value of the lag vector $\mathbf{h}$, is represented as a weighted sum of the same set of coregionalization matrices $\mathbf{B}_q$, where the weights depend on the input $\mathbf{x}$, given by the factors $k_q(\mathbf{h})$.

For a number N of input vectors, let $\mathbf{f}_d$ be the vector of values from the output d evaluated at $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$. If each output has the same set of inputs the system is known as *isotopic*. In general, we can allow each output to be associated with a different set of inputs, $\mathbf{X}^{(d)} = \{\mathbf{x}_n^{(d)}\}_{n=1}^{N_d}$, this is known as *heterotopic*.[2] For

[2]These names come from geostatistics (Wackernagel, 2003). Heterotopic data is further classified into *entirely heterotopic data*, where the variables have no sample locations in common,

notational simplicity, we restrict ourselves to the isotopic case, but our analysis can be easily used for heterotopic setups. The covariance matrix for $\mathbf{f}_d$ is obtained by expressing equation (2.2) as

$$\text{cov}[\mathbf{f}_d, \mathbf{f}_{d'}] = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} a_{d,q}^i a_{d',q}^i \mathbf{K}_q = \sum_{q=1}^{Q} b_{d,d'}^q \mathbf{K}_q, \tag{2.5}$$

where $\mathbf{K}_q \in \Re^{N \times N}$ has entries $k_q(\mathbf{h})$, for the different values that $\mathbf{h}$ may take for the particular set $\mathbf{X}$. Define $\mathbf{f}$ as $\mathbf{f} = [\mathbf{f}_1^\top, \ldots, \mathbf{f}_D^\top]^\top$. The covariance matrix for $\mathbf{f}$ in terms of equation (2.5) can be written as

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \sum_{q=1}^{Q} \mathbf{B}_q \otimes \mathbf{K}_q, \tag{2.6}$$

with the symbol $\otimes$ representing the Kronecker product between matrices (Brookes, 2005).

Intrinsic coregionalization model

A simplified version of the LMC, known as the intrinsic coregionalization model (ICM) (see Goovaerts, 1997), assumes that the elements $b_{d,d'}^q$ of the coregionalization matrix $\mathbf{B}_q$ can be written as $b_{d,d'}^q = \upsilon_{d,d'} b_q$. In other words, as a scaled version of the elements b_q which do not depend on the particular output functions $f_d(\mathbf{x})$. Using this form for $b_{d,d'}^q$, equation (2.2) can be expressed as

$$\text{cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')] = \sum_{q=1}^{Q} \upsilon_{d,d'} b_q k_q(\mathbf{x}, \mathbf{x}') = \upsilon_{d,d'} \sum_{q=1}^{Q} b_q k_q(\mathbf{x}, \mathbf{x}') = \upsilon_{d,d'} k(\mathbf{x}, \mathbf{x}'),$$

where $k(\mathbf{x}, \mathbf{x}') = \sum_{q=1}^{Q} b_q k_q(\mathbf{x}, \mathbf{x}')$ is an equivalent covariance function. The covariance matrix for $\mathbf{f}$ takes the form

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \boldsymbol{\Upsilon} \otimes \mathbf{K}, \tag{2.7}$$

where $\boldsymbol{\Upsilon} \in \Re^{D \times D}$, with entries $\upsilon_{d,d'}$, and $\mathbf{K} = \sum_{q=1}^{Q} b_q \mathbf{K}_q$ is an equivalent valid covariance matrix.

and *partially heterotopic data*, where the variables share some sample locations. In machine learning, the partially heterotopic case is sometimes referred to as *asymmetric multitask learning* (Xue et al., 2007; Chai, 2010).

The intrinsic coregionalization model can also be seen as a linear model of coregionalization where we have $Q = 1$. In such case, equation (2.6) takes the form

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \mathbf{A}_1\mathbf{A}_1^\top \otimes \mathbf{K}_1 = \mathbf{B}_1 \otimes \mathbf{K}_1, \tag{2.8}$$

where the coregionalization matrix $\mathbf{B}_1$ has elements $b^1_{d,d'} = \sum_{i=1}^{R_1} a^i_{d,1}a^i_{d',1}$. The value of R_1 determines the rank of the matrix $\mathbf{B}_1$.

As pointed out by Goovaerts (1997), the ICM is much more restrictive than the LMC since it assumes that each basic covariance $k_q(\mathbf{x}, \mathbf{x}')$ contributes equally to the construction of the autocovariances and cross covariances for the outputs. However, for inference purposes, the inverse of the covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ can be computed using the properties of the Kronecker product (as along as the input space follows the isotopic configuration) reducing the computational complexity involved when compared to the matrix inversion of the full covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ obtained from LMC. This property is employed by Rougier (2008) to speed-up the inference process in an emulator for multiple outputs. First, it assumes that the multiple output problem can be seen as a single output problem considering the output index as another variable of the input space. Then, the new output $f_{\mathbb{D}}(\widetilde{\mathbf{x}})$, with $\widetilde{\mathbf{x}} \in \Re^p \times \mathbb{D}$ and $\mathbb{D}$ the set $\mathbb{D} = \{1, \ldots, D\}$, is expressed as a weighted sum of Q deterministic regressors, $\{g_q(\widetilde{\mathbf{x}})\}_{q=1}^Q$, plus a Gaussian error term $e(\widetilde{\mathbf{x}})$ with covariance $\kappa(\widetilde{\mathbf{x}}, \widetilde{\mathbf{x}}')$. The set of regressors explain the mean of the output process, while the Gaussian error term explains the variance in the output. Both, the set of regressors and the covariance for the error, are assumed to be separable in the input space, this is, each regressor $g_q(\widetilde{\mathbf{x}}) \approx g_q(\mathbf{x})g_q(d)$ and the covariance $\kappa(\widetilde{\mathbf{x}}, \widetilde{\mathbf{x}}') = \kappa(\mathbf{x}, \mathbf{x}')\kappa(d, d')$. For isotopic spaces (Rougier (2008) refers to this condition as regular outputs, meaning outputs that are evaluated at the same set of inputs $\mathbf{X}$), the mean and covariance for the output $f_{\mathbb{D}}(\widetilde{\mathbf{x}})$, can be obtained through Kronecker products for the regressors and the covariances involved in the error term. For inference, the inversion of the necessary terms is accomplished using properties of the Kronecker product. We will see in the next section that the model that replaces the set of outputs for a single output as described before, can be seen as a particular case of the intrinsic coregionalization model (Conti and O'Hagan, 2010).

It can be shown that if the outputs are considered to be noise-free, prediction using the intrinsic coregionalization model under an isotopic data case is equivalent to independent prediction over each output (Helterbrand and Cressie, 1994). This

circumstance is also known as autokrigeability (Wackernagel, 2003).

Linear Model of Coregionalization in Machine Learning and Statistics

The linear model of coregionalization has already been used in machine learning in the context of Gaussian processes for multivariate regression, and regularization theory for multitask learning. It has also been used in statistics for computer emulation of expensive multivariate computer codes.

Before looking at related work, let us define the parameter space for the LMC. The set of parameters in the LMC includes the coregionalization matrices, $\{\mathbf{B}_q\}_{q=1}^{Q}$, and the parameters associated to the basic covariances $k_q(\mathbf{x}, \mathbf{x}')$, that we denote here as $\boldsymbol{\psi}_q$, also called *hyperparameters*. We use $\boldsymbol{\theta}_{\text{LMC}} = \{\{\mathbf{B}_q\}_{q=1}^{Q}, \{\boldsymbol{\psi}_q\}_{q=1}^{Q}\}$ to denote the whole set of parameters involved in the LMC or $\boldsymbol{\theta}_{\text{ICM}} = \{\mathbf{B}_1, \boldsymbol{\psi}_1\}$ if $Q = 1$.

As we have seen before, the linear model of coregionalization imposes the correlation of the outputs explicitly through the set of coregionalization matrices. A recurrent idea in the early days of Gaussian processes for multi-output modeling, within the machine learning literature, was based on the intrinsic coregionalization model and assumed $\mathbf{B}_1 = \mathbf{I}_D$. In other words, the outputs were considered to be conditionally independent given the parameters $\boldsymbol{\psi}_1$. Correlation between the outputs was assumed to exist implicitly by imposing the same set of hyperparameters $\boldsymbol{\psi}_1$ for all outputs, and estimating those parameters, or directly the kernel matrix $\mathbf{K}_1$, using data from all the outputs (Minka and Picard, 1999; Lawrence and Platt, 2004; Yu et al., 2005).

In this section, we review more recent approaches for multiple output modeling that are different versions of the linear model of coregionalization.

Semiparametric latent factor model. The semiparametric latent factor model (SLFM) proposed by Teh et al. (2005) turns out to be a simplified version of equation (2.6). In particular, if $R_q = 1$ (see equation (2.1)), we can rewrite equation (2.6) as

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \sum_{q=1}^{Q} \mathbf{a}_q \mathbf{a}_q^{\top} \otimes \mathbf{K}_q,$$

where $\mathbf{a}_q \in \Re^{D\times 1}$ with elements $\{a_{d,q}\}_{d=1}^{D}$ and q fixed. With some algebraic manipulations, that exploit the properties of the Kronecker product, we can write

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \sum_{q=1}^{Q}(\mathbf{a}_q \otimes \mathbf{I}_N)\mathbf{K}_q(\mathbf{a}_q^\top \otimes \mathbf{I}_N) = (\widetilde{\mathbf{A}} \otimes \mathbf{I}_N)\widetilde{\mathbf{K}}(\widetilde{\mathbf{A}}^\top \otimes \mathbf{I}_N),$$

where $\widetilde{\mathbf{A}} \in \Re^{D\times Q}$ is a matrix with columns $\mathbf{a}_q$, and $\widetilde{\mathbf{K}} \in \Re^{QN\times QN}$ is a block diagonal matrix with blocks given by $\mathbf{K}_q$.
The functions $u_q(\mathbf{x})$ are considered to be latent factors and the semiparametric name comes from the fact that it is combining a nonparametric model, this is a Gaussian process, with a parametric linear mixing of the functions $u_q(\mathbf{x})$. The kernel for each basic process q, $k_q(\mathbf{x}, \mathbf{x}')$, is assumed to be of Gaussian type with a different length scale per input dimension. For computational speed up the informative vector machine (IVM) is employed (Lawrence et al., 2003).

Multi-task Gaussian processes. The intrinsic coregionalization model has been employed by Bonilla et al. (2008) for multitask learning. We refer to this approach as multi-task Gaussian processes (MTGP). The covariance matrix is expressed as $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{x}, \mathbf{x}') = K^f k(\mathbf{x}, \mathbf{x}')$, with K^f being constrained positive semi-definite and $k(\mathbf{x}, \mathbf{x}')$ a covariance function over inputs. It can be noticed that this expression is similar to the one in (2.7), when it is evaluated for $\mathbf{x}, \mathbf{x}' \in \mathbf{X}$. In Bonilla et al. (2008), K^f (equal to $\boldsymbol{\Upsilon}$ in equation (2.7) or $\mathbf{B}_1$ in equation (2.8)) expresses the correlation between tasks or inter-task dependencies, and it is represented through a probabilistic principal component analysis (PPCA) model. In turn, the spectral factorization in the PPCA model is replaced by an incomplete Cholesky decomposition to keep numerical stability, so that $K^f \approx \widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top$, where $\widetilde{\mathbf{L}} \in \Re^{D\times R_1}$. The authors also refer to the autokrigeability effect as the cancellation of inter-task transfer (Bonilla et al., 2008). An application of MTGP for obtaining the inverse dynamics of a robotic manipulator was presented in Chai et al. (2009).

Multi-output Gaussian processes. The intrinsic coregionalization model has been also used by Osborne et al. (2008). Matrix $\boldsymbol{\Upsilon}$ in expression (2.7) is assumed to be of the spherical parametrisation kind, $\boldsymbol{\Upsilon} = \text{diag}(\mathbf{e})\mathbf{S}^\top\mathbf{S}\,\text{diag}(\mathbf{e})$, where $\mathbf{e}$ gives a description for the length scale of each output variable and $\mathbf{S}$ is an upper triangular matrix whose i-th column is associated with particular

spherical coordinates of points in $\Re^i$ (for details see Osborne and Roberts, 2007, sec. 3.4). Function $k(\mathbf{x}, \mathbf{x}')$ is represented through a Matérn kernel, where different parametrisations of the covariance allow the expression of periodic and non-periodic terms. Sparsification for this model is obtained using an IVM style approach.

Multi-task kernels in regularization theory. Kernels for multiple outputs have also been studied in the context of regularization theory. The approach is based mainly on the definition of kernels for multitask learning provided in Evgeniou and Pontil (2004); Micchelli and Pontil (2005); Evgeniou et al. (2005). The derivation is based on the theory of kernels for vector-valued functions. According to Evgeniou et al. (2005), the following lemma can be used to construct multitask kernels,

Lemma. *If G is a kernel on $\mathcal{T} \times \mathcal{T}$ and, for every $d \in \mathbb{D}$ there are prescribed mappings $\Phi_d : \mathcal{X} \to \mathcal{T}$ such that*

$$k_{d,d'}(\mathbf{x}, \mathbf{x}') = k((\mathbf{x}, d), (\mathbf{x}', d')) = G(\Phi_d(\mathbf{x}), \Phi_{d'}(\mathbf{x}')), \quad \mathbf{x}, \mathbf{x}' \in \Re^p,\ d, d' \in \mathbb{D},$$

then $k(\cdot)$ is a multitask or multioutput kernel.

A linear multitask kernel can be obtained if we set $\mathcal{T} = \Re^m$, $\Phi_d(\mathbf{x}) = \mathbf{C}_d\mathbf{x}$ with $\Phi_d \in \Re^m$ and $G : \Re^m \times \Re^m \to \Re$ as the polynomial kernel $G(\mathbf{z}, \mathbf{z}') = (\mathbf{z}^\top \mathbf{z}')^n$ with $n = 1$, leading to $k_{d,d'}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{C}_d^\top \mathbf{C}_{d'}\mathbf{x}'$. The lemma above can be seen as the result of applying kernel properties to the mapping $\Phi_d(\mathbf{x})$ (see Genton, 2001, pag. 2). Notice that this corresponds to a generalization of the semiparametric latent factor model where each output is expressed through its own basic process acting over the linear transformation $\mathbf{C}_d\mathbf{x}$, this is, $u_d(\Phi_d(\mathbf{x})) = u_d(\mathbf{C}_d\mathbf{x})$. In general, it can be obtained from $f_d(\mathbf{x}) = \sum_{q=1}^{D} a_{d,q} u_q(\Phi_q(\mathbf{x}))$, where $a_{d,q} = 1$ if $d = q$ or zero, otherwise.

Computer emulation. A computer emulator is a statistical model used as a surrogate for a computationally expensive deterministic model or computer code, also known as simulator. Gaussian processes have become the preferred statistical model among computer emulation practitioners (for a review see O'Hagan, 2006). Different Gaussian process emulators have been recently proposed to deal with several outputs (Higdon et al., 2008; Conti and O'Hagan, 2010; Rougier, 2008). In Higdon et al. (2008), the linear model of coregionalization was used to

model images representing the evolution of the implosion of steel cylinders after using TNT, and obtained employing the so called Neddemeyer simulation model (see Higdon et al., 2008, for further details). The input variable $\mathbf{x}$ represents parameters of the simulation model, while the output is an image of the radius of the inner shell of the cylinder over a fixed grid of times and angles. In the version of the LMC that the authors employed, $R_q = 1$, and the Q vectors $\mathbf{a}_q$ were obtained as the eigenvectors of a PCA decomposition of the set of training images.

In Conti and O'Hagan (2010), the intrinsic coregionalization model is employed for emulating the response of a vegetation model called the Sheffield Dynamic Global Vegetation Model (SDGVM) (Woodward et al., 1998). Authors refer to the ICM as the Multiple-Output (MO) emulator. The inputs to the model are variables related to broad soil, vegetation and climate data, while the outputs are time series of the index Net Biome Productivity (NBP) measured at different sites. The NBP index accounts for the residual amount of carbon at a vegetation site after some natural processes have taken place. In the paper, the authors assume that the outputs correspond to the different sampling time points, so that $D = T$, being T the number of time points, while each observation corresponds to a spatial sampling site.

As we mentioned before, Rougier (2008) introduces an emulator for multiple-outputs that assumes that the set of output variables can be seen as a single variable while augmenting the input space with an additional index over the outputs. In other words, it considers the output variable as an input variable. Conti and O'Hagan (2010) refer to the model in Rougier (2008) as the Time input (TI) emulator. They discussed how the TI model turns out to be a particular case of the MO model, by using a squared-exponential kernel (see Rasmussen and Williams, 2006, chapter 4) for computing the entries in the coregionalization matrix $\mathbf{B}_1$.

2.1.2 Process convolutions for multiple outputs

The approaches introduced above involve some form of instantaneous mixing through a linear weighted sum of independent processes to construct correlated processes. By instantaneous mixing we mean that the output function $f(\mathbf{x})$ evaluated at the input point $\mathbf{x}$ only depends on the values of the latent functions $\{u_q(\mathbf{x})\}_{q=1}^{Q}$ at the same input $\mathbf{x}$. Instantaneous mixing has some limitations. If

we wanted to model two output processes in such a way that one process was a blurred version of the other, we cannot achieve this through instantaneous mixing. We can achieve blurring through convolving a base process with a smoothing kernel.[3] If the base process is a Gaussian process, it turns out that the convolved process is also a Gaussian process. We can therefore exploit convolutions to construct covariance functions (Barry and Ver Hoef, 1996; Ver Hoef and Barry, 1998; Higdon, 1998, 2002). A recent review of several extensions of this approach for the single output case is presented in Calder and Cressie (2007). Applications include the construction of nonstationary covariances (Higdon, 1998; Higdon et al., 1998; Fuentes, 2002a,b; Paciorek and Schervish, 2004) and spatiotemporal covariances (Wikle et al., 1998; Wikle, 2002, 2003).

We will first introduce the idea of moving average construction for multivariate responses as it has traditionally been presented in the statistics and geostatistics literature (Ver Hoef and Barry, 1998; Ver Hoef et al., 2004; Higdon, 2002). They normally consider convolutions between kernels and white Gaussian noise processes. We then describe the covariance model used in the thesis, that allows for convolutions of kernels with more general Gaussian processes. A similar idea was presented by Fuentes (2002a,b) using discrete convolutions for developing nonstationary covariances, in the single output case.

Consider again a set of D functions $\{f_d(\mathbf{x})\}_{d=1}^D$. Each function could be expressed through a convolution integral between a kernel, $\{G_d(\mathbf{x})\}_{d=1}^D$, and a function $\{r_d(\mathbf{x})\}_{d=1}^D$,

$$f_d(\mathbf{x}) = \int_{\mathcal{X}} G_d(\mathbf{x} - \mathbf{z}) r_d(\mathbf{z}) \mathrm{d}\mathbf{z}.$$

For the integral to exist, it is assumed that the kernel $G_d(\mathbf{x})$ is a continuous function with compact support (Hörmander, 1983) or square-integrable (Ver Hoef and Barry, 1998; Higdon, 2002). The kernel $G_d(\mathbf{x})$ is also known as the moving average function (Ver Hoef and Barry, 1998) or the smoothing kernel (Higdon, 2002).

The function $r_d(\mathbf{x})$ is given by $r_d(\mathbf{x}) = \rho_d w_d(\mathbf{x}) + a_d u(\mathbf{x})$. It is composed of two elements: a white Gaussian noise random process associated to each output, $w_d(\mathbf{x})$, with mean zero and variance one, and a white Gaussian noise random

[3] We use kernel to refer to both reproducing kernels and smoothing kernels. Reproducing kernels are those used in machine learning that conform to Mercer's theorem. Smoothing kernels are functions which are convolved with a signal to create a smoothed version of that signal.

process that is common to all outputs, $u(\mathbf{x})$, also with mean zero and variance one. We can choose ρ_d, such that the variance of $r_d(\mathbf{x})$ be equal to one. For example, in Ver Hoef and Barry (1998), $\rho_d = \sqrt{1 - a_d^2}$. The processes $\{w_d(\mathbf{x})\}_{d=1}^D$ and $u(\mathbf{x})$ are assumed to be independent.

The cross-covariance between $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x}')$ is then given by

$$\begin{aligned}
\operatorname{cov}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right] &= \mathrm{E}\left[\int_{\mathcal{X}} G_d(\mathbf{x}-\mathbf{z}) r_d(\mathbf{z})\mathrm{d}\mathbf{z} \int_{\mathcal{X}} G_{d'}(\mathbf{x}'-\mathbf{z}') r_{d'}(\mathbf{z}')\mathrm{d}\mathbf{z}'\right] \\
&= \int_{\mathcal{X}}\int_{\mathcal{X}} G_d(\mathbf{x}-\mathbf{z}) G_{d'}(\mathbf{x}'-\mathbf{z}')\,\mathrm{E}\left[r_d(\mathbf{z}) r_{d'}(\mathbf{z}')\right]\mathrm{d}\mathbf{z}'\mathrm{d}\mathbf{z} \\
&= \operatorname{cov}_u\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right] + \operatorname{cov}_w\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right]\delta_{d,d'},
\end{aligned}$$

where the subscripts u and w appearing in the covariance operator, emphasize the components of $r_d(\mathbf{x})$ and $r_{d'}(\mathbf{x})$ involved in the covariance expression. We refer to $\operatorname{cov}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right]$ as $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$, $\operatorname{cov}_u\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right]$ as $k^u_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ and $\operatorname{cov}_w\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right]$ as $k^w_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$. The covariances $k^u_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ and $k^w_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ are given as

$$\begin{aligned}
k^u_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') &= a_d a_{d'} \int_{\mathcal{X}} G_d(\mathbf{x}-\mathbf{z}) G_{d'}(\mathbf{x}'-\mathbf{z})\mathrm{d}\mathbf{z} \qquad (2.9)\\
k^w_{f_d,f_d}(\mathbf{x},\mathbf{x}') &= \rho_d^2 \int_{\mathcal{X}} G_d(\mathbf{x}-\mathbf{z}) G_d(\mathbf{x}'-\mathbf{z})\mathrm{d}\mathbf{z}.
\end{aligned}$$

With the equations above, we can indirectly specify a functional form for the covariance $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ through a particular choice of the smoothing kernels $G_d(\mathbf{x})$. The smoothing kernels are usually parameterized functions and we refer to these parameters as $\boldsymbol{\theta}_{\mathrm{PC}} = \{\boldsymbol{\theta}_{G_d}\}_{d=1}^D$, where the subscript PC stands for process convolution. Under the above construction, the covariance function for the vector $\mathbf{f}(\mathbf{x})$ is positive semidefinite, for any fixed values of $\mathbf{x} = \mathbf{x}_i$ and $\mathbf{x}' = \mathbf{x}_j$.

In Ver Hoef and Barry (1998), the input space over which the integrals are computed is $\mathcal{X} = \Re^p$. Higdon (2002) depicted a similar construction to the one presented above, but assuming that the functions $r_d(\mathbf{x})$ only included the common component $u(\mathbf{x})$, so that $r_d(\mathbf{x}) = r(\mathbf{x}) = u(\mathbf{x})$. The input space is partitioned into disjoint subsets $\mathcal{X} = \bigcap_{d=0}^D \mathcal{X}_d$, and the latent process $u(\mathbf{x})$ is assumed to be independent on these separate subspaces. Each function $f_d(\mathbf{x})$ is expressed by

$$f_d(\mathbf{x}) = \int_{\mathcal{X}_d \cup \mathcal{X}_0} G_d(\mathbf{x}-\mathbf{z}) u(\mathbf{z})\mathrm{d}\mathbf{z}.$$

The functions $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x})$ are dependent within $\mathcal{X}_0$, but independent within $\mathcal{X}_d$ and $\mathcal{X}_{d'}$.

The approach described so far was introduced to the machine learning community with the name of "Dependent Gaussian Processes" (DGP) by Boyle and Frean (Boyle and Frean, 2005a,b) and further developed in Boyle (2007). Similar to Ver Hoef and Barry (1998), the input space is again $\mathcal{X} = \Re^p$. The components of the function $r_d(\mathbf{x})$, $w_d(\mathbf{x})$ and $u(\mathbf{x})$ are considered independently (ρ_d and a_d are assumed to be one), and additional moving average functions $\{H_d(\mathbf{x})\}_{d=1}^D$ are convolved with the processes $w_d(\mathbf{x})$ to allow for a different functional form for the covariance $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ when $d = d'$. In this way each output $f_d(\mathbf{x})$ follows

$$f_d(\mathbf{x}) = \int_{\mathcal{X}} G_d(\mathbf{x}-\mathbf{z})u(\mathbf{z})\mathrm{d}\mathbf{z} + \int_{\mathcal{X}} H_d(\mathbf{x}-\mathbf{z})w_d(\mathbf{z})\mathrm{d}\mathbf{z},$$

and the covariance $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ is again given by $k^u_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ plus $k^w_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')\delta_{d,d'}$, where $k^u_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ follows equation (2.9), with $a_d = 1$ for all outputs, and $k^w_{f_d,f_d}(\mathbf{x},\mathbf{x}')$ is

$$k^w_{f_d,f_d}(\mathbf{x},\mathbf{x}') = \int_{\mathcal{X}} H_d(\mathbf{x}-\mathbf{z})H_d(\mathbf{x}'-\mathbf{z})\mathrm{d}\mathbf{z}. \tag{2.10}$$

Additional modeling flexibility is allowed if instead of using a single and common white noise process $u(\mathbf{x})$, we consider a set of Q independent and common white noise processes $\{u_q(\mathbf{x})\}_{q=1}^Q$, with zero mean and variance equal to one. Then the outputs $\{f_d(\mathbf{x})\}_{d=1}^D$ can be expressed as

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q}\int_{\mathcal{X}} G_{d,q}(\mathbf{x}-\mathbf{z})u_q(\mathbf{z})\mathrm{d}\mathbf{z} + \int_{\mathcal{X}} H_d(\mathbf{x}-\mathbf{z})w_d(\mathbf{z})\mathrm{d}\mathbf{z},$$

leading to a covariance $k^u_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ given as

$$k^u_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = \sum_{q=1}^{Q}\int_{\mathcal{X}} G_{d,q}(\mathbf{x}-\mathbf{z})G_{d',q}(\mathbf{x}'-\mathbf{z})\mathrm{d}\mathbf{z}.$$

The covariance $k^w_{f_d,f_d}(\mathbf{x},\mathbf{x}')$ follows the same expression appearing in equation (2.10). The parameter vector for the DGP includes the parameters for the kernels $\{G_{d,q}(\mathbf{x})\}_{d=1,q=1}^{D,Q}$, $\{\boldsymbol{\theta}_{G_{d,q}}\}_{d=1,q=1}^{D,Q}$, and the parameters for the kernels $\{H_d(\mathbf{x})\}_{d=1}^D$, $\{\boldsymbol{\theta}_{H_d}\}_{d=1}^D$. We also denote this set of parameters jointly as $\boldsymbol{\theta}_{\mathrm{PC}}$.

In Majumdar and Gelfand (2007), a different moving average construction for the covariance of multiple outputs was introduced. It is obtained as a convolution over covariance functions in contrast to the process convolution approach where the convolution is performed over processes. Assuming that the covariances involved are isotropic, Majumdar and Gelfand (2007) show that the cross-covariance obtained from

$$\operatorname{cov}\left[f_d(\mathbf{x}+\mathbf{h}), f_{d'}(\mathbf{x})\right] = \int_{\mathcal{X}} k_d(\mathbf{h}-\mathbf{z}) k_{d'}(\mathbf{z}) \mathrm{d}\mathbf{z},$$

where $k_d(\mathbf{h})$ and $k_{d'}(\mathbf{h})$ are covariances associated to the outputs d and d', lead to a valid covariance function for the outputs $\{f_d(\mathbf{x})\}_{d=1}^{D}$. If we assume that the smoothing kernels are not only square integrable, but also positive definite functions, then the covariance convolution approach turns out to be a particular case of the process convolution approach (square-integrability might be easier to satisfy than positive definiteness).

When developing a covariance function for multivariable processes, our aim is to specify jointly the dependencies over the input space $\mathcal{X}$ and over the set of outputs $\mathbb{D}$. We have seen how in the linear model of coregionalization this compound dependency is broken in two different components, the coregionalization matrices $\mathbf{B}_q$, that specify dependencies in the set $\mathbb{D}$ independently of $\mathcal{X}$, and the basic kernels $k_q(\mathbf{x}, \mathbf{x}')$, specifying dependencies over $\mathcal{X}$ independently of $\mathbb{D}$. Therefore, the LMC assumption implies that all cross-covariances $\operatorname{cov}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right]$ share the same input dependencies that are allowed by $k_q(\mathbf{x}, \mathbf{x}')$. For a large value of Q, the weighted sum of covariances $k_q(\mathbf{x}, \mathbf{x}')$ might indirectly reflect the different levels of variation in the input space. However, a large value for Q also leads to a larger parameter space for the vector $\boldsymbol{\theta}_{\mathrm{LMC}}$, particularly when the system involves a large number of outputs D. In practice, though, one usually has prior knowledge about the diverse degrees of variation that the outputs have as functions of $\mathbf{x}$ and, subsequently, selects a set of covariances $\{k_q(\mathbf{x}, \mathbf{x}')\}_{q=1}^{Q}$ that better reflects that amount of variation. On the other hand, the process convolution framework attempts to model the variance of the set of outputs by the direct association of a different smoothing kernel $G_d(\mathbf{x})$ to each output $f_d(\mathbf{x})$. By specifying $G_d(\mathbf{x})$, one can model, for example, the degree of smoothness and the length-scale that characterizes each output. If each output happens to have more than one degree of variation (marginally, it is a sum of functions of varied smoothness) one is

faced with the same situation than in LMC, namely, the need to augment the parameter space $\boldsymbol{\theta}_{\text{PC}}$ so as to satisfy a required precision. However, due to the local description of each output that the process convolution performs, it is likely that the parameter space $\boldsymbol{\theta}_{\text{PC}}$ grows slower than the parameter space for LMC.

Here we have two methods that allows us to specify the degree of variation in each output $f_d(\mathbf{x})$ while leading to a valid covariance function for multiple outputs: *globally*, by means of Q underlying covariance functions $k_q(\mathbf{x}, \mathbf{x}')$ via the linear model of coregionalization or *locally*, through a set of smoothing kernels $G_{d,q}(\mathbf{x})$ by way of the process convolution framework.

In this thesis, we employ an extension of the process convolution framework that lies somewhere between the local and the global methods we described above. We refer to this extension as the "convolved multiple output covariance" (CMOC) In a similar way to the linear model of coregionalization, we consider Q groups of functions, where a particular group q has elements $u_q^i(\mathbf{z})$, for $i = 1, \ldots, R_q$. Each member of the group has the same covariance $k_q(\mathbf{x}, \mathbf{x}')$, but is sampled independently. Any output $f_d(\mathbf{x})$ is described by

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}) u_q^i(\mathbf{z}) \mathrm{d}\mathbf{z} + w_d(\mathbf{x}) = \sum_{q=1}^{Q} f_d^q(\mathbf{x}) + w_d(\mathbf{x}),$$

where

$$f_d^q(\mathbf{x}) = \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}) u_q^i(\mathbf{z}) \mathrm{d}\mathbf{z}, \tag{2.11}$$

and $\{w_d(\mathbf{x})\}_{d=1}^D$ are independent Gaussian processes with mean zero and covariance $k_{w_d}(\mathbf{x}, \mathbf{x}')$. We have included the superscript q for $f_d^q(\mathbf{x})$ in (2.11) to emphasize the fact that the function depends on the set of latent processes $\{u_q^i(\mathbf{x})\}_{i=1}^{R_q}$. Notice that in the process convolution formulation, each function $w_d(\mathbf{x})$ was a part of $r_d(\mathbf{x})$ and led to the covariance $k_{f_d,f_d}^w(\mathbf{x}, \mathbf{x}')$, which was a function of either $G_d(\mathbf{x})$ or both $G_d(\mathbf{x})$ and $H_d(\mathbf{x})$. Here we allow the covariance $k_{w_d}(\mathbf{x}, \mathbf{x}')$ to be any valid covariance function. Importantly, the latent functions $u_q^i(\mathbf{z})$ are Gaussian processes with general covariances $k_q(\mathbf{x}, \mathbf{x}')$, in contrast to the process convolution framework is which they were assumed to be white noise Gaussian processes.

Under the same independence assumptions used in the linear model of coregionalization, the covariance between $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x}')$ follows

$$k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = \sum_{q=1}^{Q} k_{f_d^q,f_{d'}^q}(\mathbf{x},\mathbf{x}') + k_{w_d}(\mathbf{x},\mathbf{x}')\delta_{d,d'}, \tag{2.12}$$

where

$$k_{f_d^q,f_{d'}^q}(\mathbf{x},\mathbf{x}') = \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x}-\mathbf{z}) \int_{\mathcal{X}} G_{d',q}^i(\mathbf{x}'-\mathbf{z}')k_q(\mathbf{z},\mathbf{z}')\mathrm{d}\mathbf{z}'\mathrm{d}\mathbf{z}. \tag{2.13}$$

Specifying $G_{d,q}^i(\mathbf{x}-\mathbf{z})$ and $k_q(\mathbf{z},\mathbf{z}')$ in (2.13), the covariance for the outputs $f_d(\mathbf{x})$ can be constructed indirectly.

Notice that if the smoothing kernels are taken to be the Dirac delta function such that $G_{d,q}^i(\mathbf{x}-\mathbf{z}) = a_{d,q}^i\delta(\mathbf{x}-\mathbf{z})$,[4] the double integral is easily solved and the linear model of coregionalization is recovered. On the other hand, if the latent processes $u_q^i(\mathbf{x})$ are white Gaussian noise processes with mean zero and variance one, we recover the process convolution formulation (with $R_q = 1$),

$$k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = \sum_{q=1}^{Q} \int_{\mathcal{X}} G_{d,q}(\mathbf{x}-\mathbf{z})G_{d',q}(\mathbf{x}'-\mathbf{z})\mathrm{d}\mathbf{z} + k_{w_d}(\mathbf{x},\mathbf{x}').$$

As well as the covariance across outputs, the covariance between the latent function, $u_q^i(\mathbf{z})$, and any given output, $f_d(\mathbf{x})$, can be computed,

$$k_{f_d,u_q^i}(\mathbf{x},\mathbf{z}) = \mathrm{cov}\left[f_d(\mathbf{x}), u_q^i(\mathbf{z})\right] = \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x}-\mathbf{z}')k_q(\mathbf{z}',\mathbf{z})\mathrm{d}\mathbf{z}'. \tag{2.14}$$

The convolved multiple output covariance can also be seen as a dynamic version of the linear model of coregionalization: the latent functions are dynamically combined with the help of the kernel smoothing functions, as opposed to the static combination of the latent functions in the LMC case. We are interested in allowing the latent Gaussian processes $u_q^i(\mathbf{x})$ to go beyond the white noise assumption for different reasons. First, in several problems in systems biology, the latent functions represent physical quantities, for example, transcription factor

[4] We have slightly abused of the delta notation to indicate the Kronecker delta for discrete arguments and the Dirac function for continuous arguments. The particular meaning should be understood from the context.

proteins (Gao et al., 2008; Honkela et al., 2010; Perkins et al., 2006). We expect a priori that these transcription factors behave smoothly, so that the white noise assumption would not be correct. Second, from the perspective of dynamical systems, for which the smoothing kernel corresponds to the impulse response of a particular system, the latent functions usually have very structured forms like step responses or sinusoidal waveforms, even piece-wise continuous functions. Third, even if we do not believe that the convolved multiple output covariance can represent some physical quantity (the latent function or the dynamical system), by introducing this more informative or more redundant latent function, we can propose efficient approximations of the convolved multiple output covariance for inference with large numbers of outputs and observations. An example of this approach will be presented in chapter 4.

The parameter space for the convolved multiple output covariance include the parameters of the moving-average functions and the parameters of the basic covariances $k_q(\mathbf{x}, \mathbf{x}')$, this is $\boldsymbol{\theta}_{\text{CMOC}} = \{\{\boldsymbol{\theta}_{G_{d,q}^i}\}_{d=1,q=1,i=1}^{D,Q,R_q}, \{\boldsymbol{\psi}_q\}_{q=1}^{Q}\}$.

Murray-Smith and Pearlmutter (2005) introduced the idea of transforming a Gaussian process prior using a discretized process convolution, $\mathbf{f}_d = \mathbf{G}_d\mathbf{u}$, where $\mathbf{G}_d \in \Re^{N\times K}$ is a so called *design matrix* with elements $\{G_d(\mathbf{x}_n, \mathbf{z}_k)\}_{n=1,k=1}^{N,K}$ and $\mathbf{u}^\top = [u(\mathbf{x}_1), \ldots, u(\mathbf{x}_K)]$. Such transformation could be applied for the purposes of fusing the information from multiple sensors, for solving inverse problems in reconstruction of images or for reducing computational complexity working with the filtered data in the transformed space (Shi et al., 2005).

The process convolution framework is mostly based on the assumption that the latent functions $u_q(\mathbf{x})$ are white Gaussian noise processes. A similar model to ours was proposed by Fuentes (2002a,b), but instead of the continuous convolution, Fuentes (2002a,b) used a discrete convolution. The purpose in Fuentes (2002a,b) was to develop a spatially varying covariance for single outputs, by allowing the parameters of the covariance of a base process to change as a function of the input domain. We will come back to this type of model in chapter 5.

We have seen that by going beyond the white noise process assumption, we can link models appearing in the linear model of coregionalization literature and the process convolution literature. In a similar way, Calder (2003, 2007, 2008) also

allows more general latent processes, but instead of attempting to compute cross-covariances in the multiple output setting, she embeds the latent processes into a Kalman filtering framework, while dependency between the outputs and the latent functions is given by a discretized convolution. In Calder (2008), two particulate matter (PM) levels measured in the air (10 μm in diameter and 25 μm in diameter) are modeled as the added influence of coarse and fine particles. In turn, these coarse and fine particles are modeled as random walks and then transformed by discrete convolutions to actually represent the levels of PM at 10 μm and 25 μm.

Process convolutions are also closely related to the Bayesian kernel method (Pillai et al., 2007; Liang et al., 2009) to construct reproducible kernel Hilbert spaces (RKHS) assigning priors to signed measures and mapping these measures through integral operators. In particular, define the following space of functions,

$$\mathcal{F} = \left\{ f \middle| f(x) = \int_{\mathcal{X}} G(x,z)\gamma(\mathrm{d}z),\ \gamma \in \Gamma \right\},$$

for some space $\Gamma \subseteq \mathcal{B}(\mathcal{X})$ of signed Borel measures. In Pillai et al. (2007, proposition 1), the authors show that for $\Gamma = \mathcal{B}(\mathcal{X})$, the space of all signed Borel measures, $\mathcal{F}$ corresponds to a RKHS. Examples of these measures that appear in the form of stochastic processes include Gaussian processes, Dirichlet processes and Lévy processes. In principle, we can extend this framework for the multiple output case, expresing each output as

$$f_d(x) = \int_{\mathcal{X}} G_d(x,z)\gamma(\mathrm{d}z).$$

An example of a convolved multiple output covariance.

A simple general purpose kernel for multiple outputs based on the convolved multiple output covariance framework can be constructed assuming that the kernel smoothing function, $G_{d,q}(\mathbf{x})$ (with $R_q = 1$), and the covariance for the latent function, $k_q(\mathbf{x}, \mathbf{x}')$, follow both a Gaussian form. A similar construction using a Gaussian form for $G(\mathbf{x})$ and a white noise process for $u(\mathbf{x})$ has been used in Paciorek and Schervish (2004) to propose a nonstationary covariance function in single output regression. It has also been used in Boyle and Frean (2005a) as an example of constructing dependent Gaussian processes.

The kernel smoothing function is given as

$$G_{d,q}(\boldsymbol{\tau}) = \frac{S_{d,q}|\mathbf{P}_d|^{1/2}}{(2\pi)^{p/2}} \exp\left[-\frac{1}{2}\boldsymbol{\tau}^\top \mathbf{P}_d \boldsymbol{\tau}\right],$$

where $S_{d,q}$ is a variance coefficient that depends both on the output d and the latent function q, and $\mathbf{P}_d$ is the precision matrix associated to the particular output d. The covariance function for the latent process is expressed as

$$k_q(\mathbf{x}, \mathbf{x}') = \frac{|\boldsymbol{\Lambda}_q|^{1/2}}{(2\pi)^{p/2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top \boldsymbol{\Lambda}_q (\mathbf{x} - \mathbf{x}')\right], \tag{2.15}$$

with $\boldsymbol{\Lambda}_q$ the precision matrix of the latent function q.

Expressions for the kernels are obtained applying systematically the identity for the product of two Gaussian distributions. Let $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}, \mathbf{P}^{-1})$ denote a Gaussian for $\mathbf{x}$, then

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1, \mathbf{P}_1^{-1})\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_2, \mathbf{P}_2^{-1}) = \mathcal{N}(\boldsymbol{\mu}_1|\boldsymbol{\mu}_2, \mathbf{P}_1^{-1} + \mathbf{P}_2^{-1})\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c, \mathbf{P}_c^{-1}), \tag{2.16}$$

where $\boldsymbol{\mu}_c = (\mathbf{P}_1 + \mathbf{P}_2)^{-1}(\mathbf{P}_1\boldsymbol{\mu}_1 + \mathbf{P}_2\boldsymbol{\mu}_2)$ and $\mathbf{P}_c^{-1} = (\mathbf{P}_1 + \mathbf{P}_2)^{-1}$. For all integrals we assume that $\mathcal{X} = \Re^p$. Since the Gaussian covariance is stationary, we can write it as $\mathcal{N}(\mathbf{x} - \mathbf{x}'|\mathbf{0}, \mathbf{P}^{-1}) = \mathcal{N}(\mathbf{x}' - \mathbf{x}|\mathbf{0}, \mathbf{P}^{-1}) = \mathcal{N}(\mathbf{x}|\mathbf{x}', \mathbf{P}^{-1}) = \mathcal{N}(\mathbf{x}'|\mathbf{x}, \mathbf{P}^{-1})$. Using the identity in equation (2.16) twice, we get

$$k_{f_d,f_{d'}}(\mathbf{x}, \mathbf{x}') = \sum_{q=1}^{Q} \frac{S_{d,q}S_{d',q}}{(2\pi)^{p/2}|\mathbf{P}_{\text{eqv}}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top \mathbf{P}_{\text{eqv}}^{-1}(\mathbf{x} - \mathbf{x}')\right], \tag{2.17}$$

where $\mathbf{P}_{\text{eqv}} = \mathbf{P}_d^{-1} + \mathbf{P}_{d'}^{-1} + \boldsymbol{\Lambda}_q^{-1}$. For a large value of the input dimension, p, the factor $1/[(2\pi)^{p/2}|\mathbf{P}_{\text{eqv}}|^{1/2}]$ in each of the terms above will dominate, making values go quickly to zero. We can fix this problem, by scaling the outputs using the factors $1/[(2\pi)^{p/4}|2\mathbf{P}_d^{-1} + \boldsymbol{\Lambda}_q^{-1}|^{1/4}]$ and $1/[(2\pi)^{p/4}|2\mathbf{P}_{d'}^{-1} + \boldsymbol{\Lambda}_q^{-1}|^{1/4}]$. Each of these scaling factors correspond to the standard deviations associated to $k_{f_d,f_d}(\mathbf{x}, \mathbf{x})$ and $k_{f_{d'},f_{d'}}(\mathbf{x}, \mathbf{x})$.

Equally, for the covariance $\text{cov}[f_d(\mathbf{x}), u_q(\mathbf{x}')]$ in equation (2.14), we obtain

$$k_{f_d,u_q}(\mathbf{x}, \mathbf{x}') = \frac{S_{d,q}}{(2\pi)^{p/2}|\mathbf{P}_{d,q}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top (\mathbf{P}_{d,q})^{-1}(\mathbf{x} - \mathbf{x}')\right], \tag{2.18}$$

where $\mathbf{P}_{d,q} = \mathbf{P}_d^{-1} + \boldsymbol{\Lambda}_q^{-1}$. Again, this covariance must be standardized when

working in higher dimensions.

2.2 Multivariate Gaussian Process Priors

It is important to highlight the fact that once a multivariate Gaussian process prior has been specified, through the definition of a corresponding valid covariance function, one can use the traditional Gaussian process methodology for single outputs (Rasmussen and Williams, 2006) to perform parameter estimation (section 2.2.1) and model prediction (section 2.2.2).

A Gaussian process is defined as a collection of random variables, such that any finite number of them follow a joint Gaussian distribution. Section 2.1 described a series of models for the set of outputs $\{f_d(\mathbf{x})\}_{d=1}^D$, that led to a valid covariance function for the vector $\mathbf{f}(\mathbf{x})$. As mentioned before, we denote this covariance as $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{x},\mathbf{x}') \in \Re^{D\times D}$, with elements given by $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$, and define the Gaussian process prior for the vector $\mathbf{f}(\mathbf{x})$,

$$\mathbf{f}(\mathbf{x}) \sim \mathcal{GP}(\boldsymbol{\mu}(\mathbf{x}), \mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{x},\mathbf{x}')),$$

where $\boldsymbol{\mu}(\mathbf{x}) \in \Re^{D\times 1}$ is a vector that contains the mean functions $\{\mu_d(\mathbf{x})\}_{d=1}^D$ of each output. For a finite number of inputs, $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$, the prior distribution over the vector $\mathbf{f} = [\mathbf{f}_1^\top, \ldots, \mathbf{f}_D^\top]^\top$ is given as $\mathbf{f} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{K}_{\mathbf{f},\mathbf{f}})$ or

$$\begin{bmatrix}\mathbf{f}_1\\ \mathbf{f}_2\\ \vdots\\ \mathbf{f}_D\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_1\\ \boldsymbol{\mu}_2\\ \vdots\\ \boldsymbol{\mu}_D\end{bmatrix}, \begin{bmatrix}\mathbf{K}_{\mathbf{f}_1,\mathbf{f}_1} & \mathbf{K}_{\mathbf{f}_1,\mathbf{f}_2} & \cdots & \mathbf{K}_{\mathbf{f}_1,\mathbf{f}_D}\\ \mathbf{K}_{\mathbf{f}_2,\mathbf{f}_1} & \mathbf{K}_{\mathbf{f}_2,\mathbf{f}_2} & \cdots & \mathbf{K}_{\mathbf{f}_2,\mathbf{f}_D}\\ \vdots & \vdots & \cdots & \vdots\\ \mathbf{K}_{\mathbf{f}_D,\mathbf{f}_1} & \mathbf{K}_{\mathbf{f}_D,\mathbf{f}_2} & \cdots & \mathbf{K}_{\mathbf{f}_D,\mathbf{f}_D}\end{bmatrix}\right),$$

where each vector $\mathbf{f}_d = [f_d(\mathbf{x}_1), f_d(\mathbf{x}_2), \ldots, f_d(\mathbf{x}_N)]^\top$; $\boldsymbol{\mu} = [\boldsymbol{\mu}_1^\top, \boldsymbol{\mu}_2^\top, \ldots, \boldsymbol{\mu}_N^\top]^\top$ with $\boldsymbol{\mu}_d = [\boldsymbol{\mu}_d(\mathbf{x}_1), \boldsymbol{\mu}_d(\mathbf{x}_2), \ldots, \boldsymbol{\mu}_d(\mathbf{x}_N)]^\top$; the covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}} \in \Re^{DN\times DN}$ is a block partitioned matrix with blocks given by $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$, and in turn each block $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$ has entries $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ for all values of $\mathbf{X}$. The kernel function $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ is obtained with any of the methods introduced in section 2.1. Without loss of generality, we assume that the vector $\boldsymbol{\mu}$ is zero.

In practice, we usually have access to noisy observations, so we model the outputs

$\{y_d(\mathbf{x})\}_{d=1}^D$ using

$$y_d(\mathbf{x}) = f_d(\mathbf{x}) + \epsilon_d(\mathbf{x}),$$

where $\{\epsilon_d(\mathbf{x})\}_{d=1}^D$ are independent white Gaussian noise processes with variance σ_d^2. The marginal likelihood is given as

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\phi}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}), \tag{2.19}$$

where $\mathbf{y} = \left[\mathbf{y}_1^\top, \mathbf{y}_2^\top \ldots, \mathbf{y}_D^\top\right]^\top$ is the set of output functions and each vector $\mathbf{y}_d$ has elements $\{y_d(\mathbf{x}_n)\}_{n=1}^N$; $\boldsymbol{\Sigma} = \Sigma \otimes \mathbf{I}_N$, where $\Sigma \in \Re^{D\times D}$ is a diagonal matrix with elements $\{\sigma_d^2\}_{d=1}^D$ and $\boldsymbol{\phi}$ is the set of parameters of the covariance matrix, including the parameters associated to the covariance function ($\boldsymbol{\theta}_{\text{LMC}}$, $\boldsymbol{\theta}_{\text{PC}}$ or $\boldsymbol{\theta}_{\text{CMOC}}$ for the multivariate case) and $\{\sigma_d^2\}_{d=1}^D$.

Notice that we have obtained the marginal likelihood in equation 2.19 after integrating out the latent functions $u_q^i(\mathbf{x})$.

In subsection 2.2.1, we describe how the estimation of the parameter vector $\boldsymbol{\phi}$ is accomplished, using the marginal likelihood in equation (2.19). In subsection 2.2.2, we present prediction for test inputs.

2.2.1 Parameter estimation

In this section we refer to the parameter estimation problem for the models presented in section 2.1.

In the machine learning literature, the maximization of the marginal likelihood has become the preferred method among practitioners for estimating parameters. The method also goes by the names of evidence approximation, type II maximum likelihood, empirical Bayes, among others (Bishop, 2006).

Our objective function is the logarithm of the marginal likelihood in equation (2.19),

$$\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\phi}) = -\frac{1}{2}\mathbf{y}^\top(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma})^{-1}\mathbf{y} - \frac{1}{2}\log|\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}| - \frac{ND}{2}\log 2\pi. \tag{2.20}$$

Parameters $\boldsymbol{\phi}$ are obtained by maximizing $\log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\phi})$ with respect to each

element in $\boldsymbol{\phi}$, using a gradient-descent method. Derivatives follow

$$\frac{\partial \log p(\mathbf{y}|\mathbf{X}, \boldsymbol{\phi})}{\partial \phi_i} = \frac{1}{2}\mathbf{y}^\top \mathbf{K}_{\mathbf{y},\mathbf{y}}^{-1} \frac{\partial \mathbf{K}_{\mathbf{y},\mathbf{y}}}{\partial \phi_i} \mathbf{K}_{\mathbf{y},\mathbf{y}}^{-1}\mathbf{y} - \frac{1}{2}\operatorname{tr}\left(\mathbf{K}_{\mathbf{y},\mathbf{y}}^{-1}\frac{\partial \mathbf{K}_{\mathbf{y},\mathbf{y}}}{\partial \phi_i}\right), \tag{2.21}$$

where ϕ_i is an element of the vector $\boldsymbol{\phi}$ and $\mathbf{K}_{\mathbf{y},\mathbf{y}} = \mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}$.

Another method used for parameter estimation, more common in the geostatistics literature, consists of optimizing an objective function which involves some empirical measure of the correlation between the functions $f_d(\mathbf{x})$, $\widehat{\mathbf{K}}_{\mathbf{f},\mathbf{f}}(\mathbf{h})$, and the multivariate covariance obtained using a particular model, $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{h})$, (Goulard and Voltz, 1992; Künsch et al., 1997; Pelletier et al., 2004),

$$\text{WSS} = \sum_{i=1}^{N} w(\mathbf{h}_i) \operatorname{tr}\left\{\left[\left(\widehat{\mathbf{K}}_{\mathbf{f},\mathbf{f}}(\mathbf{h}_i) - \mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{h}_i)\right)\right]^2\right\}, \tag{2.22}$$

where $w(\mathbf{h}_i)$ is a weight coefficient, $\widehat{\mathbf{K}}_{\mathbf{f},\mathbf{f}}(\mathbf{h}_i)$ is an experimental covariance matrix and $\mathbf{K}_{\mathbf{f},\mathbf{f}}(\mathbf{h}_i)$ is the covariance matrix obtained, for example, using the linear model of coregionalization.[5] One of the first algorithms for obtaining $\boldsymbol{\theta}_{\text{LMC}}$ was proposed by Goulard and Voltz (1992). It assumed that the parameters of the basic covariance functions $k_q(\mathbf{h})$ had been determined a priori and then used a weighted least squares method to fit the coregionalization matrices. In Pelletier et al. (2004) the efficiency of other least squares procedures was evaluated experimentally, including ordinary least squares and generalized least squares. Other more general algorithms in which all the parameters in $\boldsymbol{\theta}_{\text{LMC}}$ are estimated simultaneously include simulated annealing (Lark and Papritz, 2003) and the EM algorithm (Zhang, 2007). Ver Hoef and Barry (1998) also proposed the use of an objective function like (2.22), to estimate the parameters in $\boldsymbol{\theta}_{\text{PC}}$.

Both methods described above, the evidence approximation or the least-square method, give point estimates of the parameter vector $\boldsymbol{\phi}$. Several authors have

[5]Note that the common practice in geostatistics is to work with variograms instead of covariances. A variogram characterizes a general class of random functions known as intrinsic random functions (Matheron, 1973), which are random processes whose increments follow a stationary second-order process. For clarity of exposition, we will avoid the introduction of the variogram and its properties. The interested reader can follow the original paper by Matheron (1973) for a motivation of their existence, Gneiting et al. (2001) for a comparison between variograms and covariance functions and Goovaerts (1997) for a definition of the linear model of coregionalization in terms of variograms.

employed full Bayesian inference by assigning priors to $\boldsymbol{\phi}$ and computing the posterior distribution through some sampling procedure. Examples include Higdon et al. (2008) and Conti and O'Hagan (2010) under the LMC framework or Boyle and Frean (2005a) and Calder (2007) under the process convolution approach.

In the thesis, we use the marginal likelihood to find point estimates of the parameter vector, $\widehat{\boldsymbol{\phi}}$. In the case of the LMC, in which the coregionalization matrices must be positive semidefinite, we use an incomplete Cholesky decomposition $\mathbf{B}_q = \widetilde{\mathbf{L}}_q \widetilde{\mathbf{L}}_q^\top$, with $\widetilde{\mathbf{L}}_q \in \Re^{D \times R_q}$, as suggested in Bonilla et al. (2008). The elements of the matrices $\mathbf{L}_q$ are considered to be part of the vector $\boldsymbol{\phi}$.

2.2.2 Prediction

The predictive distribution for a new set of input vectors $\mathbf{X}_*$ is (Rasmussen and Williams, 2006)

$$p(\mathbf{y}_*|\mathbf{y}, \mathbf{X}, \mathbf{X}_*, \widehat{\boldsymbol{\phi}}) = \mathcal{N}\left(\mathbf{y}_*|\boldsymbol{\mu}_{\mathbf{y}_*|\mathbf{y}}, \mathbf{K}_{\mathbf{y}_*|\mathbf{y}}\right), \tag{2.23}$$

with

$$\begin{aligned}\boldsymbol{\mu}_{\mathbf{y}_*|\mathbf{y}} &= \mathbf{K}_{\mathbf{f}_*,\mathbf{f}}(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma})^{-1}\mathbf{y}, \\ \mathbf{K}_{\mathbf{y}_*|\mathbf{y}} &= \mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{f}}(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma})^{-1}\mathbf{K}_{\mathbf{f}_*,\mathbf{f}}^\top + \boldsymbol{\Sigma}_*,\end{aligned}$$

where we have used $\mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*}$ as a compact notation to indicate when the covariance matrix is evaluated at the inputs $\mathbf{X}_*$, with a similar notation for $\mathbf{K}_{\mathbf{f}_*,\mathbf{f}}$.

Once we have provided a model to describe the uncertainty in the predictions, the next step usually involves making a decision related to the problem at hand (Bishop, 2006). *Decision theory* is the mathematical framework that studies how to make *optimal* decisions given the uncertainty in the prediction. These optimal decisions are made by minimizing an *expected loss* or *risk*, by taking the average of a *loss function* with respect to the predictive distribution (Rasmussen and Williams, 2006). For the single output case, define $L(y_*, \hat{y}(\mathbf{x}_*))$ as the loss we would incur in approximating y_* (which is the true unknown value) with the value $\hat{y}(\mathbf{x}_*)$, provided by the model. In practice, we do not know the probability distribution for y_*, so we use the predictive distribution given by the model to

average over all values of y_*. The expected loss is then

$$\mathrm{E}[L] = \int L(y_*, \hat{y}(\mathbf{x}_*))p(y_*|\mathbf{x}_*, \mathrm{data})\mathrm{d}y_*.$$

A common loss function in regression problems is the squared-loss, $(y_* - \hat{y}(\mathbf{x}_*))^2$. With this loss function, the risk follows

$$\mathrm{E}[L] = \int (y_* - \hat{y}(\mathbf{x}_*))^2 p(y_*|\mathbf{x}_*, \mathrm{data})\mathrm{d}y_*.$$

Minimizing the above expression with respect to $\hat{y}(\mathbf{x}_*)$ leads to the following expression for the optimal prediction

$$\hat{y}_{\mathrm{optimal}}(\mathbf{x}_*) = \int y_* p(y_*|\mathbf{x}_*, \mathrm{data})\mathrm{d}y_* = \mathrm{E}_{y*}[y_*],$$

that corresponds to the mean prediction given by the predictive distribution. In the multivariate case, it just corresponds to $\boldsymbol{\mu}_{\mathbf{y}_*|\mathbf{y}}$.

In geostatistics, the framework that allows for optimal predictions in the multivariate case is known by the general name of *cokriging* (Goovaerts, 1997). Prediction for a particular output $f_d(\mathbf{x}_*)$ is obtained as the result of the following linear estimator,

$$\hat{f}_d(\mathbf{x}_*) - \mu_d(\mathbf{x}_*) = \sum_{s=1}^{D} \sum_{\alpha_s=1}^{n_s(\mathbf{x}_*)} \lambda_{\alpha_s}(\mathbf{x}_*) \left[f_s(\mathbf{x}_{\alpha_s}) - \mu_s(\mathbf{x}_{\alpha_s}) \right],$$

where $\lambda_{\alpha_s}(\mathbf{x}_*)$ are the weights assigned to the output data $f_s(\mathbf{x}_{\alpha_s})$, $\mu_s(\mathbf{x}_{\alpha_s})$ are the expected values of $f_s(\mathbf{x}_{\alpha_s})$, and $n_s(\mathbf{x}_*) \leq N$. Cokriging estimators are required to be unbiased ($E[f_d(\mathbf{x}_*) - \hat{f}_d(\mathbf{x}_*)] = 0$), and minimize the error variance, σ_E^2, between the true value and the estimator prediction,

$$\sigma_E^2(\mathbf{x}_*) = \mathrm{var}\left[f_d(\mathbf{x}_*) - \hat{f}_d(\mathbf{x}_*) \right].$$

The weights $\lambda_{\alpha_s}(\mathbf{x}_*)$ are those that minimize $\sigma_E^2(\mathbf{x}_*)$.

In the cokriging literature, each output function $f_d(\mathbf{x})$ is decomposed into a residual component $R_d(\mathbf{x})$ and a trend component $\mu_d(\mathbf{x})$, $f_d(\mathbf{x}) = R_d(\mathbf{x}) + \mu_d(\mathbf{x})$, $\forall d$. Residuals $R_d(\mathbf{x})$ are assumed to be stationary Gaussian processes with mean equal

zero and covariance $k_{d,d}(\mathbf{h})$, while two different residuals, $R_d(\mathbf{x})$ and $R_{d'}(\mathbf{x})$, have cross-covariance $k_{d,d'}(\mathbf{h})$. If the means $\{\mu_d(\mathbf{x})\}_{d=1}^D$ are considered to be known and constants,[6] in particular zero, it can be shown that for a matrix of inputs $\mathbf{X}_*$, the cokriging weights are given as (Goovaerts, 1997),

$$\boldsymbol{\lambda} = \mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{K}_{\mathbf{f}_*,\mathbf{f}}^\top,$$

where $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ is a valid positive semidefinite matrix. Predictions for $\mathbf{f}$ are then given as $\hat{\mathbf{f}} = \boldsymbol{\lambda}^\top\mathbf{f} = \mathbf{K}_{\mathbf{f}_*,\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{f}}^{-1}\mathbf{f}$, that matches the mean prediction of the predictive distribution in equation (2.23), for the case of noise-free observations.
Cokriging can be considered as a non-Bayesian version of prediction with Gaussian processes, in which the predictor is obtained directly from the loss function (the variance of the error in this case). Contrast this with the Bayesian perspective of GPs, where the predictive distribution is obtained in a first stage of the analysis, while the loss function appears at the end, as part of the decision stage. This difference between Bayesian and non-Bayesian paradigms was highlighted in Rasmussen and Williams (2006, page 22).

As part of the inference process, we might also be interested in computing the posterior distribution over the latent functions. To keep the notation uncluttered, assume $R_q = 1$. Let $\mathbf{u} = [\mathbf{u}_1^\top, \mathbf{u}_2^\top, \ldots, \mathbf{u}_Q^\top]^\top$, where $\mathbf{u}_q = [u_q(\mathbf{x}_1), u_q(\mathbf{x}_2), \ldots, u_q(\mathbf{x}_N)]^\top$, then the posterior distribution over the latent functions $\mathbf{u}$, $p(\mathbf{u}|\mathbf{y}, \mathbf{X}, \widehat{\boldsymbol{\phi}})$ can be computed as

$$p(\mathbf{u}|\mathbf{y}, \mathbf{X}, \widehat{\boldsymbol{\phi}}) = \mathcal{N}(\mathbf{u}|\boldsymbol{\mu}_{\mathbf{u}|\mathbf{y}}, \mathbf{K}_{\mathbf{u}|\mathbf{y}}), \tag{2.24}$$

with

$$\begin{aligned}\boldsymbol{\mu}_{\mathbf{u}|\mathbf{y}} &= \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}\right)^{-1} \mathbf{y},\\ \mathbf{K}_{\mathbf{u}|\mathbf{y}} &= \mathbf{K}_{\mathbf{u},\mathbf{u}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top \left(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}\right)^{-1} \mathbf{K}_{\mathbf{f},\mathbf{u}},\end{aligned}$$

where $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is a block-diagonal matrix with blocks given by $\mathbf{K}_{\mathbf{u}_q,\mathbf{u}_q}$. In turn, the elements of $\mathbf{K}_{\mathbf{u}_q,\mathbf{u}_q}$ are given by $k_q(\mathbf{x}, \mathbf{x}')$ for $\{\mathbf{x}_n\}_{n=1}^N$. Also $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ is a matrix with blocks $\mathbf{K}_{\mathbf{f}_d,\mathbf{u}_q}$, where $\mathbf{K}_{\mathbf{f}_d,\mathbf{u}_q}$ has entries given by $k_{f_d,u_q}(\mathbf{x}, \mathbf{x}')$ in equation (2.14).

[6] This is a special case of cokriging, known as *simple cokriging*. Different assumptions over $\{\mu_d(\mathbf{x})\}_{d=1}^D$ lead to different cokriging estimators. For other variants of cokriging, the reader is referred to Goovaerts (1997).

In the next section, we present an example of multivariate regression with Gaussian processes using the tools developed so far.

2.3 Multivariate regression for gene expression

In this section we present an example in which we compare the linear model of coregionalization and the convolved multiple output covariance for multivariate regression in gene expression data.

Microarray technology has made the simultaneous measurement of mRNA from thousands of genes possible. Transcription is governed by the presence or absence of transcription factor (TF) proteins that act as switches to turn on and off the expression of the genes. Most of these methods are based on assuming that there is an instantaneous linear relationship between the gene expression and the protein concentration. We compare the performance of the intrinsic coregionalization model and the convolved GPs for two independent time series or replicates of 12 time points collected hourly throughout Drosophila embryogenesis in wild-type embryos (Tomancak et al., 2002). For preprocessing the data, we follow Honkela et al. (2010). We concentrate on a particular transcription factor protein, namely `twi`, and the genes associated with it. The information about the network connections is obtained from the ChIP-chip experiments. This particular TF is a key regulator of mesoderm and muscle development in Drosophila (Zinzen et al., 2009).

After preprocessing the data, we end up with a dataset of 1621 genes with expression data for $N = 12$ time points. It is believed that this set of genes are regulated by at least the `twi` transcription factor. For each one of these genes, we have access to 2 replicates. Each gene is considered to be an output, $y_d(t)$, while the transcription factor is assumed to be a latent function, $u(t)$, that triggers the expression of the genes. Out of the 1621 genes, we randomly select $D = 50$ genes from replicate 1 for training a full multiple output GP model based on either the LMC framework or the convolved multiple output covariance. The corresponding 50 genes of replicate 2 are used for testing, and results are presented in terms of the standardized mean square error (SMSE) and the mean standardized log loss (MSLL) as defined in Rasmussen and Williams (2006).[7] The parameters of both

[7] The definitions for the SMSE and the MSLL we have used here are slightly different from the ones provided in Rasmussen and Williams (2006). Instead of comparing against a Gaussian with a global mean and variance computed from all the outputs in the training data, we compare

the LMC and the convolved GPs are found through the maximization of the log-marginal likelihood in equation (2.20), by means of a scaled conjugate gradient procedure, and using only the data from replicate 1. We run the scaled conjugate gradient routine for 100 iterations. Once we have estimated the hyperparameters of the covariance function, we test both methods over the outputs in replicate 2, by conditioning on the outputs of replicate 1. In practice then, we assume indirectly that both replicates are correlated, in the sense that they are realizations of the same multivariate Gaussian process. We repeated the experiment 10 times using a different set of 50 genes each time. We also repeated the experiment selecting the 50 genes for training from replicate 2 and the corresponding 50 genes of replicate 1 for testing. For testing over the genes of replicate 1, we condition on the genes of replicate 2.

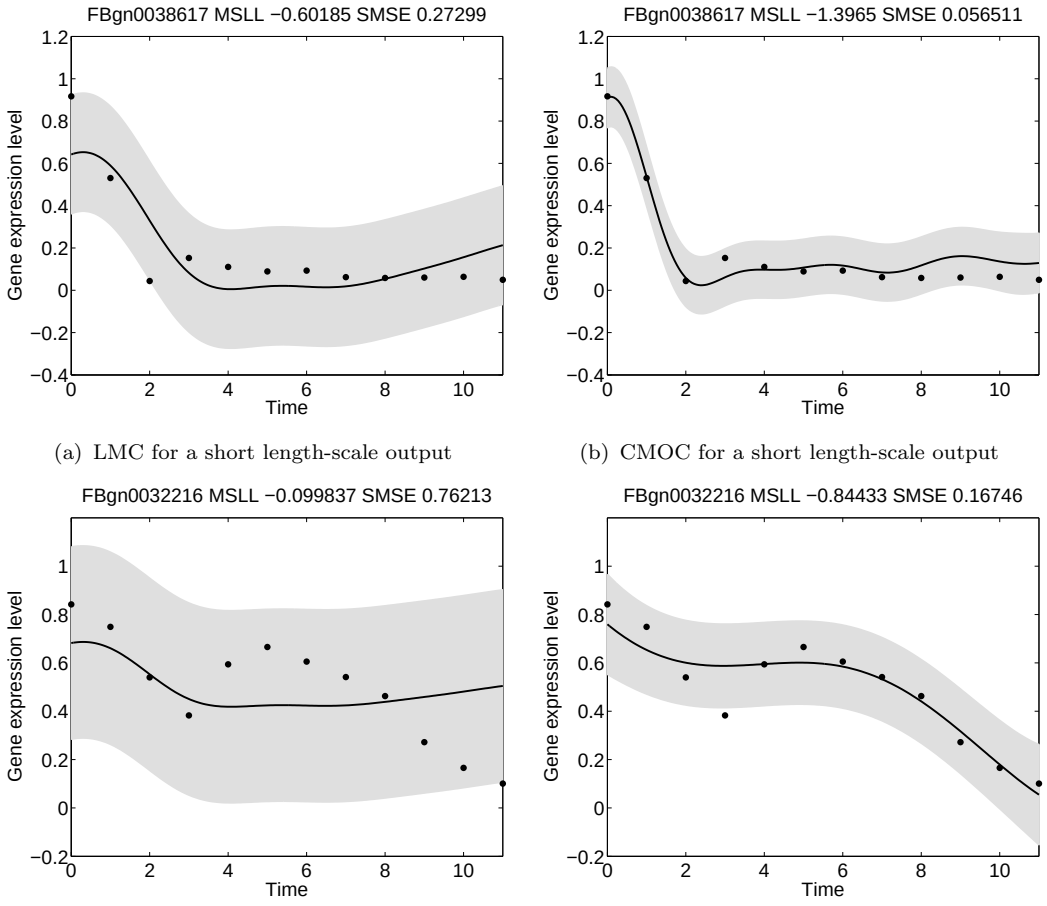
Since we are interested in a reduced representation of the data, we assume that $Q = 1$ and $R_q = 1$, for the LMC and the convolved multiple output GP in equations (2.1) and (2.12), respectively. For the LMC model, we follow Bonilla et al. (2008) and assume an incomplete Cholesky decomposition for $\mathbf{B}_1 = \widetilde{\mathbf{L}}_1\widetilde{\mathbf{L}}_1^\top$, where $\widetilde{\mathbf{L}}_1 \in \Re^{50\times 1}$ and as the basic covariance $k_q(\mathbf{x}, \mathbf{x}')$ we assume the squared exponential covariance function (Rasmussen and Williams, 2006, p. 83). For the convolved multiple output GP we employ the covariance in equation (2.17), with the appropriate scaling factors.

Train set	Test set	Method	Average SMSE	Average MSLL
Replicate 1	Replicate 2	LMC	0.6069 ± 0.0294	-0.2687 ± 0.0594
		CMOC	0.4859 ± 0.0387	-0.3617 ± 0.0511
Replicate 2	Replicate 1	LMC	0.6194 ± 0.0447	-0.2360 ± 0.0696
		CMOC	0.4615 ± 0.0626	-0.3811 ± 0.0748

Table 2.1: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the gene expression data for 50 outputs. CMOC stands for convolved multiple output covariance. The experiment was repeated ten times with a different set of 50 genes each time. Table includes the value of one standard deviation over the ten repetitions. More negative values of MSLL indicate better models.

Table 2.1 shows the results of both methods over the test set for the two different replicates. It can be seen that the convolved multiple output covariance (appearing as CMOC in the table), outperforms the LMC covariance both in terms of SMSE and MSLL.

against a Gaussian with local means and local variances computed from the training data associated to each output.



(a) LMC for a short length-scale output

(b) CMOC for a short length-scale output

(c) LMC for a long length-scale output

(d) CMOC for a long length-scale output

Figure 2.1: Predictive mean and variance for genes FBgn0038617 (first row) and FBgn0032216 (second row) using the linear model of coregionalization (figures 2.1(a) and 2.1(c)) and the convolved multiple-output covariance (figures 2.1(b) and 2.1(d)) with $Q = 1$ and $R_q = 1$. The training data comes from replicate 1 and the testing data from replicate 2. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure and appear also in table 2.2. The adjectives "short" and "long" given to the length-scales in the captions of each figure, must be understood relative to each other.

Figure 2.1 shows the prediction made over the test set (replicate 2 in this case) by the two models for two particular genes, namely FBgn0038617 (figure 2.1, first row) and FBgn0032216 (figure 2.1, second row). The black dots in the figures represent the gene expression data of the particular genes. Figures 2.1(a) and 2.1(c) show the response of the LMC and figures 2.1(b) and 2.1(d) show the response of the convolved multiple output covariance. It can be noticed from the data that the two genes differ in their responses to the action of the transcription factor, that is, while gene FBgn0038617 has a rapid decay around

time 2 and becomes relatively constant for the rest of the time interval, gene FBgn0032216 has a smoother response within the time frame. The linear model of coregionalization is driven by a latent function with a length-scale that is shared across the outputs. Notice from figures 2.1(a) and 2.1(c) that the length-scale for both responses is the same. On the other hand, due-to the non-instantaneous mixing of the latent function, the convolved multiple output framework, allows for the description of each output by using its own length-scale, which gives an added flexibility for describing the data.

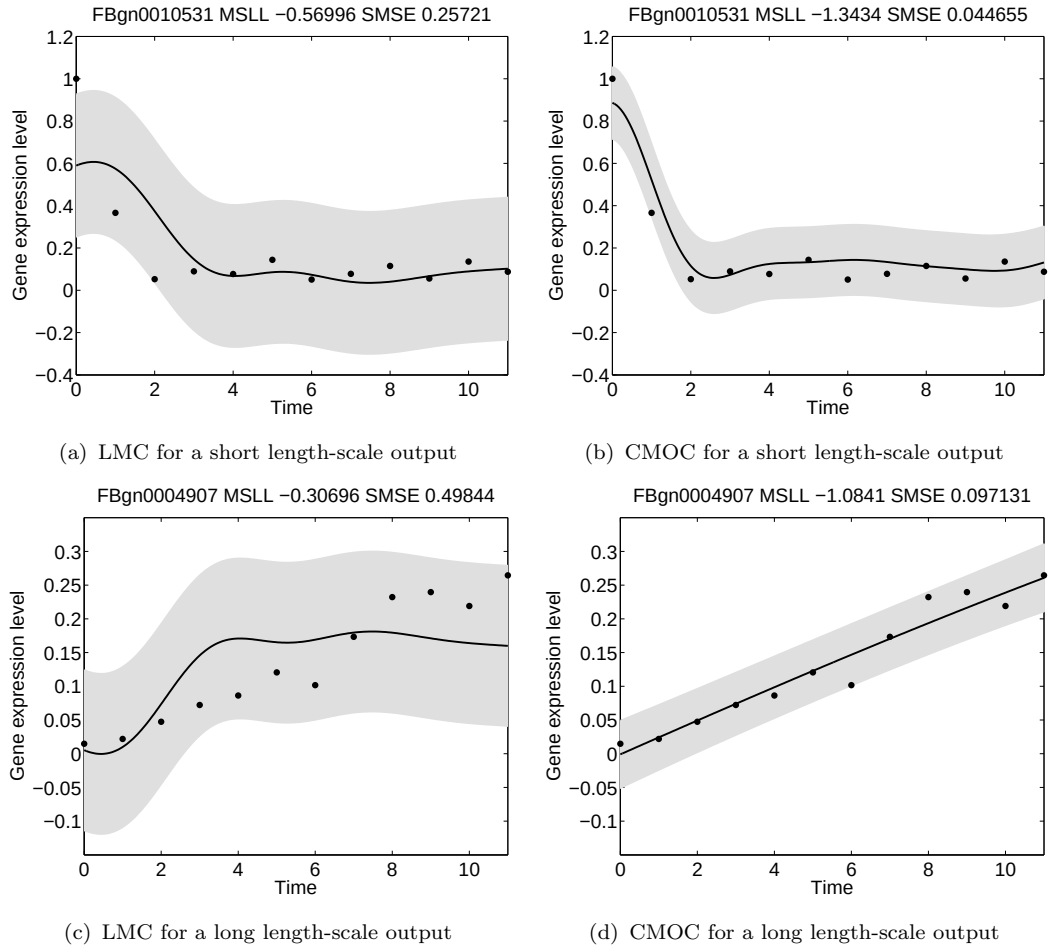
Table 2.2 (first four rows) shows the performances of both models for the genes of figure 2.1. CMOC outperforms the linear model of coregionalization for both genes in terms of SMSE and MSLL.

Test replicate	Test genes	Method	SMSE	MSLL
Replicate 2	FBgn0038617	LMC	0.2729	−0.6018
		CMOC	0.0565	−1.3965
	FBgn0032216	LMC	0.7621	−0.0998
		CMOC	0.1674	−0.8443
Replicate 1	FBgn0010531	LMC	0.2572	−0.5699
		CMOC	0.0446	−1.3434
	FBgn0004907	LMC	0.4984	−0.3069
		CMOC	0.0971	−1.0841

Table 2.2: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the genes in figures 2.1 and 2.2 for LMC and CMOC. Gene FBgn0038617 and gene FBgn0010531 have a shorter length-scale when compared to the length-scales of genes FBgn0032216 and FBgn0004907.

A similar analysis can be made for figures 2.2(a), 2.2(b), 2.2(c) and 2.2(d). In this case, the test set is replicate 1 and we have chosen two different genes, FBgn0010531 and FBgn0004907 with a similar behavior to the ones in figure 2.1. Table 2.2 (last four rows) also highlights the performances of both models for the genes of figure 2.2. Again, CMOC outperforms the linear model of coregionalization for both genes and in terms of SMSE and MSLL.

Having said this, we can argue that the performance of the LMC model can be improved by either increasing the value of Q or the value R_q, or both. For the intrinsic coregionalization model, we would fix the value of $Q = 1$ and increase the value of R_1. Effectively, we would be increasing the rank of the coregionalization matrix $\mathbf{B}_1$, meaning that more latent functions sampled from the same covariance function are being used to explain the data. In a extreme case in which each output has its own length scale, this translates into equating the number of latent



(a) LMC for a short length-scale output

(b) CMOC for a short length-scale output

(c) LMC for a long length-scale output

(d) CMOC for a long length-scale output

Figure 2.2: Predictive mean and variance for genes FBgn0010531 (first row) and FBgn0004907 (second row) using the linear model of coregionalization (figures 2.2(a) and 2.2(c)) and the convolved multiple-output covariance (figures 2.2(b) and 2.2(d)) with $Q = 1$ and $R_q = 1$. The difference with figure 2.1 is that now the training data comes from replicate 2 while the testing data comes from replicate 1. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure.

functions to the number of outputs, or in other words assuming a full rank for the matrix $\mathbf{B}_1$. This leads to the need of estimating the matrix $\mathbf{B}_1 \in \Re^{D\times D}$, that might be problematic if D is high. For the semiparametric latent factor model, we would fix the value of $R_q = 1$ and increase Q, the number of latent functions sampled from Q different GPs. Again, in the extreme case of each output having its own length-scale, we might need to estimate a matrix $\widetilde{\mathbf{A}} \in \Re^{D\times D}$, which could be problematic for a high value of outputs. In a more general case, we could also combine values of $Q > 1$ and $R_q > 1$. We would need then, to find values of Q and R_q that fit the different outputs with different length scales.

Therefore, in the above context, we have seen that the convolved covariance could offer an explanation of the data through a simpler model or converge to the LMC, if needed.

2.4 Summary

In this chapter we have presented different alternatives for constructing valid covariance functions to be used in a multivariate Gaussian process framework. We introduced the convolved multiple output covariance and saw that it contains the Linear Model of Coregionalization and the Process Convolutions as particular cases. We have also specified the elements that we will use in the following chapters, namely, parameter estimation by maximum likelihood and predictive posterior distribution.

The linear model of coregionalization can be interpreted as an instantaneous mixing of latent functions, in contrast to a convolved multiple output framework, where the mixing is not necessarily instantaneous. Experimental results presented in publication **v** showed that there is a benefit in using this non-instantaneous mixing in terms of predictive precision. This augmented performance was more noticeable in systems with a presence of some dynamics.

One important question in the process convolution framework and in the convolved multiple output covariance is how to choose the kernel smoothing functions $\{G_{d,q}\}_{d=1,q=1}^{D,Q}$. Although, there are non-parametric alternatives (Ver Hoef and Barry, 1998) as well as plenty of parametric ones (Higdon, 2002), in this thesis we are interested in dynamical systems, for which the present alternatives are not suitable. In the next chapter, we will study moving-average functions obtained from linear differential equations, that will allow us to encode prior knowledge of the system's dynamics in the covariance function.

Chapter 3

Linear Latent force models

In chapter 2 we established a general framework to develop covariance functions for multivariate regression in a Gaussian processes context. We proposed the convolved multiple output covariance as a method that generalizes different other alternatives in the literature and that is parameterized in terms of a set of moving-average functions $G^i_{d,q}(\mathbf{x})$ and a set of covariances $k_q(\mathbf{x}, \mathbf{x}')$. One important question for making the approach practical is how to specify the moving-average functions and the covariances of the latent functions.

It is well known, from the theory of dynamical systems, that there exists a correspondence between a linear differential equation and a convolution transform, and that this correspondence is established through what is called as the *impulse response* of the system. From a mathematical point of view, the impulse response is better known as the *Green's function* and it is a standard method used to solve differential equations (Griffel, 2002; Rynne and Youngson, 2008).

In this chapter, we motivate the use of Green's functions as alternatives to smoothing kernels by introducing a generative model of the noisy outputs, which we call linear latent force models or simply *latent force models* (LFM). A latent force model introduces basic mechanistic principles in the formulation of a traditional latent variable model. Our motivation is to augment a latent variable model with the ability to incorporate salient characteristics of the data (for example, in a mechanical system *inertia* or *resonance*), even knowing that the differential equation from which it is derived does not reflect the real dynamics of the system. For example, for a human motion capture dataset, we develop a mechanistic

model of motion capture that does not exactly replicate the *physics* of human movement, but nevertheless captures important features of the movement.

The linear latent force model is a generalization of the work of Lawrence et al. (2007) and Gao et al. (2008), who encoded first order differential equations in the covariance function of a multivariate Gaussian process.

In section 3.1, we introduce the linear latent force model as a latent variable model. We then see how the latent force model translates into a multivariate Gaussian process with a convolved multiple output covariance in section 3.2. In section 3.3, we present a second order latent force model for motion capture data. Finally, section 3.4 presents related work.

Remark. Section 3.1 and 3.3 were originally presented in publication **ii**. Section 3.2, which connects the latent force model with chapter 2, is new. The section on related work is also new.

3.1 From latent variables to latent forces

From the perspective of machine learning, the linear latent force model can be seen as a type of latent variable model. In a latent variable model we may summarize a high dimensional data set with a reduced dimensional representation. For example, if our data consists of N points in a D dimensional space we might seek a linear relationship between the data, $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_D] \in \Re^{N\times D}$ with $\mathbf{y}_d \in \Re^{N\times 1}$, and a reduced dimensional representation, $\mathbf{U} = [\mathbf{u}_1, \ldots, \mathbf{u}_Q] \in \Re^{N\times Q}$ with $\mathbf{u}_q \in \Re^{N\times 1}$, where $Q < D$. From a probabilistic perspective this involves an assumption that we can represent the data as

$$\mathbf{Y} = \mathbf{U}\mathbf{W}^\top + \mathbf{E}, \tag{3.1}$$

where $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_D]$ is a matrix-variate Gaussian noise: each column, $\mathbf{e}_d \in \Re^{N\times 1}$ $(1 \leq d \leq D)$, is a multivariate Gaussian with zero mean and covariance $\mathbf{\Sigma}_d$, this is $\mathbf{e}_d \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma}_d)$. The usual approach, as undertaken in factor analysis (FA) and principal component analysis (PCA), to dealing with the unknowns in this model is to integrate out $\mathbf{U}$ under a Gaussian prior and optimize with respect to $\mathbf{W} \in \Re^{D\times Q}$ (for a non-linear variant of the model it can be convenient to do this the other way around, this is, integrate out $\mathbf{W}$ and optimize $\mathbf{U}$, see for example Lawrence (2005)). If the data has a temporal nature, then the Gaussian

prior in the latent space could express a relationship between the rows of $\mathbf{U}$, $\mathbf{u}_{t_n} = \mathbf{\Gamma}\mathbf{u}_{t_{n-1}} + \boldsymbol{\eta}$, where $\mathbf{\Gamma}$ is a transformation matrix, $\boldsymbol{\eta}$ is a general noise process, usually Gaussian, and $\mathbf{u}_{t_n}$ is the n-th row of $\mathbf{U}$, which we associate with time t_n. This is known as the *Kalman filter/smoother*. Normally the times t_n, are taken to be equally spaced, but more generally we can consider a joint distribution for $p(\mathbf{U}|\mathbf{t})$, with $\mathbf{t} = [t_1 \dots t_N]^\top$, which has the form of a Gaussian process,

$$p(\mathbf{U}|\mathbf{t}) = \prod_{q=1}^{Q} \mathcal{N}\left(\mathbf{u}_q|\mathbf{0}, \mathbf{K}_{\mathbf{u}_q,\mathbf{u}_q}\right),$$

where we have assumed zero mean and independence across the Q dimensions of the latent space. The GP makes explicit the fact that the latent variables are functions, $\{u_q(t)\}_{q=1}^{Q}$, and we have now described them with a process prior. The elements of the vector $\mathbf{u}_q = [u_q(t_1), \dots, u_q(t_N)]^\top$, represent the values of the function for the q-th dimension at the times given by $\mathbf{t}$. The matrix $\mathbf{K}_{\mathbf{u}_q,\mathbf{u}_q}$ is the covariance function associated to $u_q(t)$ computed at the times given in $\mathbf{t}$.

Such a GP can be readily implemented. Given the covariance functions for $\{u_q(t)\}_{q=1}^{Q}$, the implied covariance functions for $\{y_d(t)\}_{d=1}^{D}$ are straightforward to derive. In Teh et al. (2005) this is known as a semiparametric latent factor model. If the latent functions $u_q(t)$ share the same covariance, but are sampled independently, this is known as the multi-task Gaussian process prediction model (Bonilla et al., 2008) with a similar model introduced in Osborne et al. (2008). Both models were introduced in chapter 2 as particular cases of the linear model of coregionalization. Historically the Kalman filter approach has been preferred, perhaps because of its linear computational complexity in N. However, recent advances in sparse approximations have made the general GP framework practical (see Quiñonero-Candela and Rasmussen, 2005b, for a review).

So far the model described relies on the latent variables to provide the dynamic information. The novelty here is that we include a further dynamical system with a *mechanistic* inspiration. We now use a mechanical analogy to introduce it. Consider the following physical interpretation of equation (3.1): the latent functions, $u_q(t)$, are Q forces and we observe the displacement of D springs, $y_d(t)$, to the forces. Then we can reinterpret (3.1) as the force balance equation, $\mathbf{Y}\boldsymbol{\kappa} = \mathbf{U}\mathbf{S}^\top + \widetilde{\mathbf{E}}$. We have assumed that the forces are acting, for example, through levers, so that we have a matrix of sensitivities, $\mathbf{S} \in \Re^{D\times Q}$, and a diagonal matrix of spring constants, $\boldsymbol{\kappa} \in \Re^{D\times D}$, with elements $\{\kappa_d\}_{d=1}^{D}$. The original model is

recovered by setting $\mathbf{W}^\top = \mathbf{S}^\top \boldsymbol{\kappa}^{-1}$ and $\tilde{\mathbf{e}}_d \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{\kappa}^\top \boldsymbol{\Sigma}_d \boldsymbol{\kappa}\right)$. The model can be extended by assuming that the spring is acting in parallel with a damper and that the system has mass, allowing us to write,

$$\ddot{\mathbf{Y}}\mathbf{M} + \dot{\mathbf{Y}}\mathbf{V} + \mathbf{Y}\boldsymbol{\kappa} = \mathbf{U}\mathbf{S}^\top + \widehat{\mathbf{E}}, \tag{3.2}$$

where $\mathbf{M}$ and $\mathbf{V}$ are diagonal matrices of masses, $\{m_d\}_{d=1}^D$, and damping coefficients, $\{v_d\}_{d=1}^D$, respectively, $\dot{\mathbf{Y}}$ is the first derivative of $\mathbf{Y}$ with respect to time (with entries $\{\dot{y}_d(t_n)\}$ for $d = 1, \dots D$ and $n = 1, \dots, N$), $\ddot{\mathbf{Y}}$ is the second derivative of $\mathbf{Y}$ with respect to time (with entries $\{\ddot{y}_d(t_n)\}$ for $d = 1, \dots D$ and $n = 1, \dots, N$), and $\widehat{\mathbf{E}}$ is once again a matrix-variate Gaussian noise. Equation (3.2) specifies a particular type of interaction between the outputs $\mathbf{Y}$ and the set of latent functions $\mathbf{U}$, namely, that a weighted sum of the second derivative for $y_d(t)$, $\ddot{y}_d(t)$, the first derivative for $y_d(t)$, $\dot{y}_d(t)$, and $y_d(t)$ is equal to the weighted sum of functions $\{u_q(t)\}_{q=1}^Q$ plus random noise. The second order mechanical system that this model describes will exhibit several characteristics which are impossible to represent in the simpler latent variable model given by (3.1), such as inertia and resonance. This model is not only appropriate for data from mechanical systems. There are many analogous systems which can also be represented by second order differential equations, for example Resistor-Inductor-Capacitor circuits. A unifying characteristic for all these models is that the system is being forced by latent functions, $\{u_q(t)\}_{q=1}^Q$. Hence, we refer to them as latent force models.

One way of thinking about this model is to consider puppetry. A marionette is a representation of a human (or animal) controlled by a limited number of inputs through strings (or rods) attached to the character. This limited number of inputs can lead to a wide range of character movements. In the model, the data is the movements of the marionette, and the latent forces are the inputs to the system from the puppeteer.

3.2 From latent forces to convolved covariances

In the last section we provided a general description of the latent force model idea and commented how it compares to previous models in the machine learning and statistics literature. In this section we specify the operational procedure

to obtain the Gaussian process model related to the outputs. We illustrate the methodology using a first-order latent force model, for which we assume there are no masses associated to the outputs and the damper constants are equal to one. We then generalize the operational procedure for latent force models of higher order and multidimensional inputs.

3.2.1 First-order Latent Force Model

Assume a simplified latent force model, for which only the first derivative of the outputs is included. This is a particular case of equation (3.2), with masses equal to zero and damper constants equal to one. With these assumptions, equation (3.2) can be written as

$$\dot{\mathbf{Y}} + \mathbf{Y}\boldsymbol{\kappa} = \mathbf{U}\mathbf{S}^{\top} + \widehat{\mathbf{E}}. \tag{3.3}$$

Individual elements in equation (3.3) follow

$$\frac{\mathrm{d}y_d(t)}{\mathrm{d}t} + \kappa_d y_d(t) = \sum_{q=1}^{Q} S_{d,q} u_q(t) + \hat{e}_d(t). \tag{3.4}$$

Given the parameters $\{\kappa_d\}_{d=1}^{D}$ and $\{S_{d,q}\}_{d=1,q=1}^{D,Q}$, the uncertainty in the outputs is given by the uncertainty coming from the set of functions $\{u_q(t)\}_{q=1}^{Q}$ and the noise $\hat{e}_d(t)$. Strictly speaking, this equation belongs to a more general set of equations known as *stochastic differential equations* (SDE) that are usually solved using special techniques from stochastic calculus (Øksendal, 2003). The representation used in equation (3.4) is more common in physics, where it receives the name of *Langevin equations* (Reichl, 1998, p. 251-254). For the simpler equation (3.4), the solution is found using standard calculus techniques and is given as

$$y_d(t) = y_d(t_0)e^{-\kappa_d t} + \sum_{q=1}^{Q} S_{d,q}\mathcal{G}_d[u_q](t) + \mathcal{G}_d[\hat{e}_d](t), \tag{3.5}$$

where $y_d(t_0)$ correspond to the value of $y_d(t)$ for $t = t_0$ (or the initial condition) and $\mathcal{G}_d$ is a linear integral operator that follows

$$\mathcal{G}_d[u_q](t) = f_d(t, u_q(t)) = \int_0^t e^{-\kappa_d(t-\tau)} u_q(\tau)\mathrm{d}\tau.$$

The integral operation corresponds to a convolution transform and we can immediately recognize the analogy with equation (2.11), with $R_q = 1$, where the smoothing kernel is given by $G_d(\mathbf{x} - \mathbf{z}) = e^{-\kappa_d(x-z)}$. Notice that the smoothing kernel here does not depend on parameters related to the latent function, $u_q(t)$. However, the sensitivity coefficient $S_{d,q}$ could be easily incorporated into the moving average function, and we would have $G_{d,q}(\mathbf{x} - \mathbf{z}) = S_{d,q}e^{-\kappa_d(x-z)}$.
Our noise model $\mathcal{G}_d[\hat{e}_d](t)$ in equation (3.5) has a particular form depending on the linear operator $\mathcal{G}_d$. For example, assuming a white noise process prior for $e_d(t)$, it can be shown that the process $\mathcal{G}_d[\hat{e}_d](t)$ corresponds to the Ornstein-Uhlenbeck (OU) process (Reichl, 1998). In what follows, we will allow the noise model to be a more general process and we denote it by $w_d(t)$. It could be an independent Gaussian process as in chapter 2, a noise process, or even both. For the current formulation, we also assume that the initial conditions $\{y_d(t_0)\}_{d=1}^D$ are zero, so that we can write again equation (3.5) as

$$y_d(t) = \sum_{q=1}^{Q} S_{d,q}\mathcal{G}_d[u_q](t) + w_d(t).$$

We assume that the latent functions $\{u_q(t)\}_{q=1}^Q$ are independent and each of them follows a Gaussian process prior, this is, $u_q(t) \sim \mathcal{GP}(0, k_{u_q,u_q}(t,t'))$. Following a similar framework to the one exposed in chapter 2, $\{y_d(t)\}_{d=1}^D$ correspond to a Gaussian process with covariances

$$k_{y_d,y_{d'}}(t,t') = k_{f_d,f_{d'}}(t,t') + k_{w_d,w_{d'}}(t,t')\delta_{d,d'},$$

where $k_{f_d,f_{d'}}(t,t')$ is given as

$$k_{f_d,f_{d'}}(t,t') = \sum_{q=1}^{Q} S_{d,q}S_{d',q}k_{f_d^q,f_{d'}^q}(t,t').$$

Furthermore, the covariance $k_{f_d^q,f_{d'}^q}(t,t')$ is equal to

$$k_{f_d^q,f_{d'}^q}(t,t') = \int_0^t e^{-\kappa_d(t-\tau)} \int_0^{t'} e^{-\kappa_{d'}(t'-\tau')}k_{u_q,u_q}(\tau,\tau')\mathrm{d}\tau'\mathrm{d}\tau. \tag{3.6}$$

Notice that this equation has the same form than equation (2.13), used to establish the basic covariance in the convolved multiple output covariance framework.

In our current framework, the form for the function $k_{u_q,u_q}(t,t')$ must be such that we can solve both integrals in equation (3.6) and find an analytical expression for the covariance $k_{f_d,f_{d'}}(t,t')$. In this chapter, we assume that the covariance for each latent force $u_q(t)$ follows the squared-exponential (SE) form (Rasmussen and Williams, 2006)

$$k_{u_q,u_q}(t,t') = \exp\left(-\frac{(t-t')^2}{\ell_q^2}\right), \tag{3.7}$$

where ℓ_q is known as the length-scale. We can compute the covariance $k_{f_d^q,f_{d'}^q}(t,t')$ obtaining (Lawrence et al., 2007)

$$k_{f_d^q,f_{d'}^q}(t,t') = \frac{\sqrt{\pi}\ell_q}{2}[h_{d',d}(t',t) + h_{d,d'}(t,t')], \tag{3.8}$$

where

$$\begin{aligned} h_{d',d}(t',t) = \frac{\exp(\nu_{q,d'}^2)}{\kappa_d + \kappa_{d'}} \exp(-\kappa_{d'}t') \Bigg\{ & \exp(\kappa_{d'}t) \\ & \times \left[\operatorname{erf}\left(\frac{t'-t}{\ell_q} - \nu_{q,d'}\right) + \operatorname{erf}\left(\frac{t}{\ell_q} + \nu_{q,d'}\right)\right] \\ & - \exp(-\kappa_d t)\left[\operatorname{erf}\left(\frac{t'}{\ell_q} - \nu_{q,d'}\right) + \operatorname{erf}(\nu_{q,d'})\right]\Bigg\}, \end{aligned}$$

with $\operatorname{erf}(x)$ the real valued error function, $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x \exp(-y^2)\mathrm{d}y$, and $\nu_{q,d} = \ell_q\kappa_d/2$.

The covariance function in equation (3.8) is nonstationary. For the stationary regime, the covariance function can be obtained by writing $t' = t + \tau$ and taking the limit as t tends to infinity. This is, $k^{\mathrm{STAT}}_{f_d^q,f_{d'}^q}(\tau) = \lim_{t\to\infty} k_{f_d^q,f_{d'}^q}(t,t+\tau)$. The stationary covariance could also be obtained making use of the power spectral density for the stationary processes $u_q(t)$, $U_q(\omega)$ and the transfer function $H(\omega)$ associated to $h(t-s) = e^{-\kappa_d(t-s)}$, the impulse response of the first order dynamical system. Then applying the convolution property of the Fourier transform to obtain the power spectral density of $f_d(t)$, $F_d(\omega)$ and finally using the *Wiener-Khinchin theorem* to find the solution for $f_d(t)$ (Shanmugan and Breipohl, 1988).

Due to the independence between $u_q(t)$ and $w_d(t)$, the cross-covariance between

the output $y_d(t)$ and the latent force $u_q(t)$ reduces to $k_{f_d,u_q}(t,t')$, given as

$$k_{f_d,u_q}(t,t') = \frac{\sqrt{\pi}\ell_q S_{d,q}}{2}\exp(\nu_{q,d}^2)\exp(-\kappa_d(t-t')) \times\left[\operatorname{erf}\left(\frac{t-t'}{\ell_q}-\nu_{q,d}\right)+\operatorname{erf}\left(\frac{t'}{\ell_q}+\nu_{q,d}\right)\right].$$

3.2.2 Higher-order Latent Force Models

In general, a latent force model of order M can be described by the following equation

$$\sum_{m=0}^{M}\mathcal{D}^m[\mathbf{Y}]\mathbf{A}_m = \mathbf{U}\mathbf{S}^\top + \widehat{\mathbf{E}}, \tag{3.9}$$

where $\mathcal{D}^m$ is a linear differential operator such that $\mathcal{D}^m[\mathbf{Y}]$ is a matrix with elements given by $\mathcal{D}^m y_d(t) = \frac{\mathrm{d}^m y_d(t)}{\mathrm{d}^m t}$ and $\mathbf{A}_m$ is a diagonal matrix with elements $A_{m,d}$ that weights the contribution of $\mathcal{D}^m y_d(t)$.

We follow the same procedure described in section 3.2.1 for the model in equation (3.9) with $M=1$. Each element in expression (3.9) can be written as

$$\mathcal{D}_0^M y_d = \sum_{m=0}^{M} A_{m,d}\mathcal{D}^m y_d(t) = \sum_{q=1}^{Q} S_{d,q}u_q(t) + \hat{e}_d(t), \tag{3.10}$$

where we have introduced a new operator $\mathcal{D}_0^M$ that is equivalent to applying the weighted sum of operators $\mathcal{D}^m$. For a homogeneous differential equation in (3.10), this is $u_q(t)=0$ for $q=1,\ldots,Q$ and $\hat{e}_d(t)=0$, and a particular set of initial conditions $\{\mathcal{D}^m y_d(t_0)\}_{m=0}^{M-1}$, it is possible to find a linear integral operator $\mathcal{G}_d$ associated to $\mathcal{D}_0^M$ that can be used to solve the non-homogeneous differential equation. The linear integral operator is defined as

$$\mathcal{G}_d[u](t) = f_d(t,u(t)) = \int_{\mathcal{T}} G_d(t,\tau)u(\tau)\mathrm{d}\tau, \tag{3.11}$$

where $G_d(t,s)$ is known as the Green's function associated to the differential operator $\mathcal{D}_0^M$, $u(t)$ is the input function for the non-homogeneous differential equation

and $\mathcal{T}$ is the input domain. The particular relation between the differential operator and the Green's function is given by

$$\mathcal{D}_0^M[G_d(t,s)] = \delta(t-s), \tag{3.12}$$

with s fixed, $G_d(t,s)$ a fundamental solution that satisfies the initial conditions and $\delta(t-s)$ the Dirac delta function (Griffel, 2002). Strictly speaking, the differential operator in equation (3.12) is the adjoint for the differential operator appearing in equation (3.10). For a more rigorous introduction to Green's functions applied to differential equations, the interested reader is referred to Roach (1982). In the signal processing and control theory literature, the Green's function is known as the impulse response of the system. Following the general latent force model framework, we write the outputs as

$$y_d(t) = \sum_{q=1}^{Q} S_{d,q}\mathcal{G}_d[u_q](t) + w_d(t),$$

where $w_d(t)$ is again an independent process associated to each output. We assume once more that the latent forces follow independent Gaussian process priors with zero mean and covariance $k_{u_q,u_q}(t,t')$. The covariance for the outputs $k_{y_d,y_{d'}}(t,t')$ is given as $k_{f_d,f_{d'}}(t,t') + k_{w_d,w_{d'}}(t,t')\delta_{d,d'}$, with $k_{f_d,f_{d'}}(t,t')$ equal to

$$\begin{aligned} k_{f_d,f_{d'}}(t,t') &= \sum_{q=1}^{Q} S_{d,q}S_{d',q}\int_{\mathcal{T}}\int_{\mathcal{T}'} G_d(t-\tau)G_{d'}(t'-\tau')k_{u_q,u_q}(\tau,\tau')\mathrm{d}\tau'\mathrm{d}\tau \\ &= \sum_{q=1}^{Q} S_{d,q}S_{d',q}k_{f_d^q,f_{d'}^q}(t,t'), \end{aligned} \tag{3.13}$$

and $k_{f_d^q,f_{d'}^q}(t,t')$ following

$$k_{f_d^q,f_{d'}^q}(t,t') = \int_{\mathcal{T}}\int_{\mathcal{T}'} G_d(t-\tau)G_{d'}(t'-\tau')k_{u_q,u_q}(\tau,\tau')\mathrm{d}\tau'\mathrm{d}\tau. \tag{3.14}$$

Learning and inference for the higher-order latent force model is done as explained in subsections 2.2.1 and 2.2.2, respectively. The Green's function is described by a parameter vector $\boldsymbol{\theta}_{G_d}$ and with the length-scales $\{\boldsymbol{\psi}_q\}_{q=1}^Q = \{\ell_q\}_{q=1}^Q$ describing the latent GPs, the vector of hyperparameters is given as $\boldsymbol{\theta}_{\text{LFM}} = \{\boldsymbol{\theta}_{G_d}\}_{d=1}^D, \{S_{d,q}\}_{d=1,q=1}^{D,Q}, \{\boldsymbol{\psi}_q\}_{q=1}^Q\}$. The parameter vector $\boldsymbol{\theta}_{\text{LFM}}$ is estimated by

maximizing the logarithm of the marginal likelihood in equation (2.20), where the elements of the matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ are computed using expression (3.13) with $k_{f_d^q,f_{d'}^q}(t,t')$ given by (3.14). For prediction we use expression (2.23) and the posterior distribution is found using expression (2.24), where the elements of the matrix $\mathbf{K}_{\mathbf{f},\mathbf{u}}$, $k_{f_d,u_q}(t,t') = k_{f_d^q,u_q}(t,t')$, are computed using

$$S_{d,q}\int_{\mathcal{T}} G_d(t-\tau)k_{u_q,u_q}(\tau,t')\mathrm{d}\tau. \tag{3.15}$$

3.2.3 Multidimensional inputs

In the sections above we have introduced latent force models for which the input variable is one-dimensional. For higher-dimensional inputs, $\mathbf{x} \in \Re^p$, we can use linear partial differential equations to establish the dependency between the latent forces and the outputs. The initial conditions turn into boundary conditions, specified by a set of functions that are linear combinations of $y_d(\mathbf{x})$ and its lower derivatives, evaluated at a set of specific points of the input space. Inference and learning is done in a similar way to the one-input dimensional latent force model. Once the Green's function associated to the linear partial differential operator has been established, we employ similar equations to (3.14) and (3.15) to compute $k_{f_d,f_d'}(\mathbf{x},\mathbf{x}')$ and $k_{f_d,u_q}(\mathbf{x},\mathbf{x}')$ and the hyperparameters appearing in the covariance function are estimated by maximizing the marginal likelihood.

3.3 A Latent Force Model for Motion Capture Data

In section 3.1 we introduced the analogy of a marionette's motion being controlled by a reduced number of forces. Human *motion capture data* consists of a skeleton and multivariate time courses of angles which summarize the motion. This motion can be modelled with a set of second order differential equations which, due to variations in the centers of mass induced by the movement, are non-linear. The simplification we consider for the latent force model is to linearize these differential equations, resulting in the following second order system,

$$m_d\frac{\mathrm{d}^2y_d(t)}{\mathrm{d}t^2} + \upsilon_d\frac{\mathrm{d}y_d(t)}{\mathrm{d}t} + \kappa_d y_d(t) = \sum_{q=1}^{Q} S_{d,q}u_q(t) + \hat{e}_d(t). \tag{3.16}$$

Whilst (3.16) is not the correct physical model for our system, it will still be helpful when extrapolating predictions across different motions, as we shall see later. Note also that, although similar to (3.4), the dynamic behavior of this system is much richer than that of the first order system, since it can exhibit inertia and resonance. In what follows, we will assume without loss of generality that the masses are equal to one.

For the motion capture data, $y_d(t)$ corresponds to a given observed angle over time, and its derivatives represent angular velocity and acceleration. The system is summarized by the undamped natural frequency, $\omega_{0d} = \sqrt{\kappa_d}$, and the damping ratio, $\zeta_d = \frac{1}{2}\upsilon_d/\sqrt{\kappa_d}$. Systems with a damping ratio greater than one are said to be overdamped, whereas underdamped systems exhibit resonance and have a damping ratio less than one. For critically damped systems $\zeta_d = 1$, and finally, for undamped systems (this is no friction) $\zeta_d = 0$.

Ignoring the initial conditions, the solution of (3.16) is given by the integral operator of equation (3.11), with Green's function

$$G_d(t,s) = \frac{1}{\omega_d}\exp(-\alpha_d(t-s))\sin(\omega_d(t-s)), \tag{3.17}$$

where $\omega_d = \sqrt{4\kappa_d - \upsilon_d^2}/2$ and $\alpha_d = \upsilon_d/2$.

According to the general framework described in section 3.2.2, the covariance function between the outputs is obtained by solving expression (3.14), where $k_{u_q,u_q}(t,t')$ follows the SE form in equation (3.7). Solution for $k_{f_d^q,f_{d'}^q}(t,t')$ is then given as (Álvarez et al., 2009)

$$\begin{aligned} K_0\big[&h_q(\widetilde{\gamma}_{d'},\gamma_d,t,t') + h_q(\gamma_d,\widetilde{\gamma}_{d'},t',t) + h_q(\gamma_{d'},\widetilde{\gamma}_d,t,t') + h_q(\widetilde{\gamma}_d,\gamma_{d'},t',t) \\ &- h_q(\widetilde{\gamma}_{d'},\widetilde{\gamma}_d,t,t') - h_q(\widetilde{\gamma}_d,\widetilde{\gamma}_{d'},t',t) - h_q(\gamma_{d'},\gamma_d,t,t') - h_q(\gamma_d,\gamma_{d'},t',t)\big], \end{aligned}$$

where $K_0 = \ell_q\sqrt{\pi}/8\omega_d\omega_{d'}$, $\gamma_d = \alpha_d + j\omega_d$ and $\widetilde{\gamma}_d = \alpha_d - j\omega_d$, and the functions $h_q(\widetilde{\gamma}_{d'},\gamma_d,t,t')$ follow

$$h_q(\gamma_{d'},\gamma_d,t,t') = \frac{\Upsilon_q(\gamma_{d'},t',t) - e^{-\gamma_d t}\Upsilon_q(\gamma_d,t',0)}{\gamma_d + \gamma_{d'}},$$

with $\Upsilon_q(\gamma_{d'}, t, t')$

$$2e^{\left(\frac{\ell_q^2\gamma_{d'}^2}{4}\right)}e^{-\gamma_{d'}(t-t')} - e^{\left(-\frac{(t-t')^2}{\ell_q^2}\right)}\mathrm{w}(jz_{d',q}(t)) - e^{\left(-\frac{(t')^2}{\ell_q^2}\right)}e^{(-\gamma_{d'}t)}\mathrm{w}(-jz_{d',q}(0)), \tag{3.18}$$

and $z_{d',q}(t) = (t-t')/\ell_q - (\ell_q\gamma_{d'})/2$. Note that $z_{d',q}(t) \in \mathbb{C}$, and $\mathrm{w}(jz)$ in (3.18), for $z \in \mathbb{C}$, denotes Faddeeva's function $\mathrm{w}(jz) = \exp(z^2)\mathrm{erfc}(z)$, where $\mathrm{erfc}(z)$ is the complex version of the complementary error function, $\mathrm{erfc}(z) = 1 - \mathrm{erf}(z) = \frac{2}{\sqrt{\pi}}\int_z^\infty \exp(-v^2)\mathrm{d}v$. Faddeeva's function is usually considered the complex equivalent of the error function, since $|\mathrm{w}(jz)|$ is bounded whenever the imaginary part of jz is greater or equal than zero, and is the key to achieving a good numerical stability when computing (3.18) and its gradients.

Similarly, the cross-covariance between latent functions and outputs in equation (3.15) is given by

$$k_{f_d^q,u_q}(t,t') = \frac{\ell_q S_{d,q}\sqrt{\pi}}{j4\omega_d}[\Upsilon_q(\widetilde{\gamma}_d, t, t') - \Upsilon_q(\gamma_d, t, t')].$$

Motion Capture data

Our motion capture data set is from the CMU motion capture data base.[1] We considered 3 balancing motions (18, 19, 20) from subject 49. The subject starts in a standing position with arms raised, then, over about 10 seconds, he raises one leg in the air and lowers his arms to an outstretched position. Of interest to us was the fact that, whilst motions 18 and 19 are relatively similar, motion 20 contains more dramatic movements. We were interested in training on motions 18 and 19 and testing on the more dramatic movement to assess the model's ability to extrapolate. The data was down-sampled by 32 (from 120 frames per second to just 3.75) and we focused on the subject's left arm. For the left arm, we chose $D = 9$ outputs including the humerus (X, Y and Z rotations), the radius (X rotation), the wrist (the Y rotation), the hand (X and Z rotations), and the thumb (X and Z rotations). The number of latent forces is fixed to $Q = 2$. Our objective was to reconstruct the movement of this arm for motion 20 given the angles of the humerus and the parameters learned from motions 18 and 19

[1]The CMU Graphics Lab Motion Capture Database was created with funding from NSF EIA-0196217 and is available at `http://mocap.cs.cmu.edu`.

using two latent functions. First, we train the second order differential equation latent force model on motions 18 and 19, treating the sequences as independent, but sharing parameters (this is, the damping coefficients and natural frequencies of the two differential equations associated with each angle were constrained to be the same). The training is done by maximizing the log-marginal likelihood in equation (2.20). Once the parameters are learned, we use them for testing the extrapolation ability of the model over movement 20. For the test data, we condition on the observations of the humerus orientation to make predictions for the rest of the arm's angles.

For comparison, we considered a regression model that directly predicts the angles of the arm given the orientation of the humerus using standard independent GPs with SE covariance functions. A similar setup is used, this is, we learn hyperparameters for 6 independent GPs, having as inputs the humerus' angle rotations (three rotations) of motions 18 and 19. For testing, we use the three angles of humerus for motion 20, and predict over the 6 other outputs. Results are summarized in table 3.1, with some example plots of the tracks of the angles given in figure 3.1.

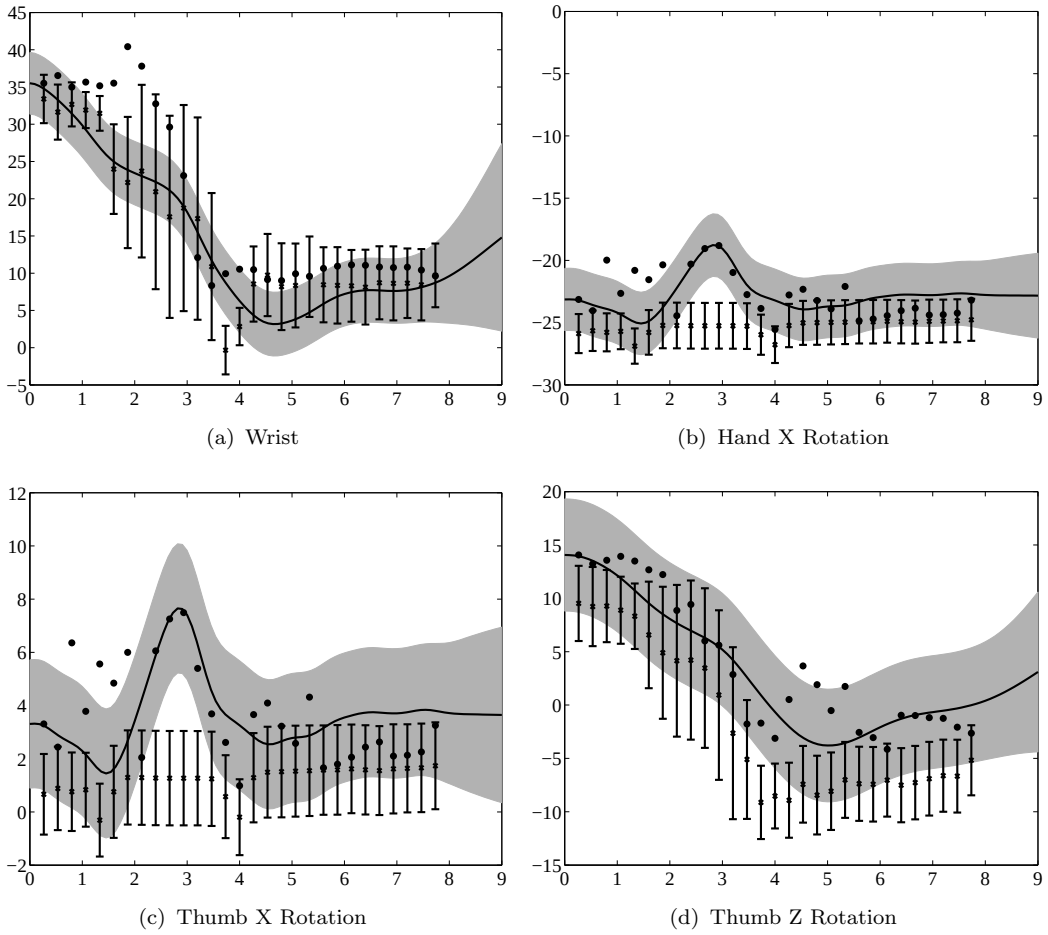
Angle	Latent Force Error	Regression Error
Radius	4.11	4.02
Wrist	6.55	6.65
Hand X rotation	1.82	3.21
Hand Z rotation	2.76	6.14
Thumb X rotation	1.77	3.10
Thumb Z rotation	2.73	6.09

Table 3.1: Root mean squared (RMS) angle error for prediction of the left arm's configuration in the motion capture data. Prediction with the latent force model outperforms the prediction with regression for all apart from the radius's angle.

In the next section, we present related work of differential equations in statistics and machine learning.

3.4 Related work

Differential equations are the cornerstone in a diverse range of engineering fields and applied sciences. However, their use for inference in statistics and machine



(a) Wrist (b) Hand X Rotation

(c) Thumb X Rotation (d) Thumb Z Rotation

Figure 3.1: Predictions from the latent force model (solid line, grey error bars) and from direct regression from the humerus angles (crosses with stick error bars). For these examples noise is high due to the relatively small length of the bones. Despite this the latent force model does a credible job of capturing the angle, whereas direct regression with independent GPs fails to capture the trends.

learning has been less studied. The main field in which they have been used is known as *functional data analysis* (Ramsay and Silverman, 2005).

From the frequentist statistics point of view, the literature in functional data analysis has been concerned with the problem of parameter estimation in differential equations (Poyton et al., 2006; Ramsay et al., 2007): given a differential equation with unknown coefficients $\{\mathbf{A}_m\}_{m=0}^{M}$, how do we use data to fit those parameters? Notice that there is a subtle difference between those techniques and the latent force model. While these parameter estimation methods start with a very accurate description of the interactions in the system via the differential equation (the differential equation might even be non-linear as in Perkins et al.

(2006)), in the latent force model, we use the differential equation as part of the modeling problem: the differential equation is used as a way to introduce prior knowledge over a system for which we do not know the real dynamics, but for which we hope, some important features of that dynamics could be expressed. Having said that, we review some of the parameter estimation methods because they also deal with differential equations with an uncertainty background.

Classical approaches to fit parameters $\boldsymbol{\theta}$ of differential equations to observed data include numerical approximations of initial value problems and collocation methods (Ramsay et al. (2007) and Brewer et al. (2008) provide reviews and detailed descriptions of additional methods).

The solution by numerical approximations include an iterative process in which given an initial set of parameters $\boldsymbol{\theta}_0$ and a set of initial conditions $\mathbf{y}_0$, a numerical method is used to solve the differential equation. The parameters of the differential equation are then optimized by minimizing an error criterion between the approximated solution and the observed data. For exposition, we assume in equation (3.10) that $D = 1$, $Q = 1$ and $S_{1,1} = 1$. We are interested in finding the solution $y(t)$ to the following differential equation, with unknown parameters $\boldsymbol{\theta} = \{A_m\}_{m=0}^{M}$,

$$\mathcal{D}_0^M y(t) = \sum_{m=0}^{M} A_m \mathcal{D}^m y(t) = u(t),$$

In the classical approach, we assume that we have access to a vector of initial conditions, $\mathbf{y}_0$, and data for $u(t)$, $\mathbf{u}$. We start with an initial guess for the parameter vector $\boldsymbol{\theta}_0$ and solve numerically the differential equation to find a solution $\widetilde{\mathbf{y}}$. An updated parameter vector $\widetilde{\boldsymbol{\theta}}$ is obtained by minimizing

$$E(\boldsymbol{\theta}) = \sum_{n=1}^{N} \|\widetilde{y}(t_n) - y(t_n)\|.$$

through any gradient descent method. To use any of those methods, we must be able to compute $\partial E(\boldsymbol{\theta})/\partial \boldsymbol{\theta}$, which is equivalent to compute $\partial y(t)/\partial \boldsymbol{\theta}$. In general, when we do not have access to $\partial y(t)/\partial \boldsymbol{\theta}$, we can compute it using what is known as the *sensitivity equations* (see Bard, 1974, chapter 8, for detailed explanations), which are solved along with the ODE equation that provides the partial solution $\widetilde{\mathbf{y}}$. Once a new parameter vector $\widetilde{\boldsymbol{\theta}}$ has been found, the same steps are repeated

until some convergence criterion is satisfied. If the initial conditions are not available, they can be considered as additional elements of the parameter vector $\boldsymbol{\theta}$ and optimized in the same gradient descent method.

In collocation methods, the solution of the differential equation is approximated using a set of basis functions, $\{\phi_i(t)\}_{i=1}^J$, this is $y(t) = \sum_{i=1}^J \beta_i \phi_i(t)$. The basis functions must be sufficiently smooth so that the derivatives of the unknown function, appearing in the differential equation, can be obtained by differentiation of the basis representation of the solution, this is, $\mathcal{D}^m y(t) = \sum \beta_i \mathcal{D}^m \phi_i(t)$. Collocation methods also use an iterative procedure for fitting the additional parameters involved in the differential equation. Once the solution and its derivatives have been approximated using the set of basis functions, minimization of an error criteria is used to estimate the parameters of the differential equation. Principal differential analysis (PDA) (Ramsay, 1996) is one example of a collocation method in which the basis functions are *splines*. In PDA, the parameters of the differential equation are obtained by minimizing the squared residuals of the higher order derivative $\mathcal{D}^M y(t)$ and the weighted sum of derivatives $\{\mathcal{D}^m y(t)\}_{m=0}^{M-1}$, instead of the squared residuals between the approximated solution and the observed data.

An example of a collocation method augmented with Gaussian process priors was introduced by Graepel (2003). Graepel starts with noisy observations, $\widehat{y}(t)$, of the differential equation $\mathcal{D}_0^M y(t)$, such that $\widehat{y}(t) \sim \mathcal{N}(\mathcal{D}_0^M y(t), \sigma_y)$. The solution $y(t)$ is expressed using a basis representation, $y(t) = \sum \beta_i \phi_i(t)$. A Gaussian prior is placed over $\boldsymbol{\beta} = [\beta_1, \ldots, \beta_J]$, and its posterior computed under the above likelihood. With the posterior over $\boldsymbol{\beta}$, the predictive distribution for $\widehat{y}(t_*)$ can be readily computed, being a function of the matrix $\mathcal{D}_0^M \boldsymbol{\Phi}$ with elements $\{\mathcal{D}_0^M \phi_i(t_n)\}_{n=1,i=1}^{N,J}$. It turns out that products $\mathcal{D}_0^M \boldsymbol{\Phi} \left(\mathcal{D}_0^M \boldsymbol{\Phi}\right)^\top$ that appear in this predictive distribution have individual elements that can be written as $\sum_{i=1}^J \mathcal{D}_0^M \phi_i(t_n) \mathcal{D}_0^M \phi_i(t_{n'}) = \mathcal{D}_{0,t}^M \mathcal{D}_{0,t'}^M \sum_{i=1}^J \phi_i(t_n)\phi_i(t_{n'})$ or, using a kernel representation for the inner products $k(t_n, t_{n'}) = \sum_{i=1}^J \phi_i(t_n)\phi_i(t_{n'})$, as $k_{\mathcal{D}_{0,t}^M, \mathcal{D}_{0,t'}^M}(t_n, t_{n'})$, where this covariance is obtained by taking $\mathcal{D}_0^M$ derivatives of $k(t, t')$ with respect to t and $\mathcal{D}_0^M$ derivatives with respect to t'. In other words, the result of the differential equation $\mathcal{D}_0^M y(t)$ is assumed to follow a Gaussian process prior with covariance $k_{\mathcal{D}_{0,t}^M, \mathcal{D}_{0,t'}^M}(t, t')$. An approximated solution $\widetilde{y}(t)$ can be computed through the expansion $\widetilde{y}(t) = \sum_{n=1}^N \alpha_n k_{\mathcal{D}_{0,t'}^M}(t, t_n)$, where α_n is an element of the vector $(\mathbf{K}_{\mathcal{D}_{0,t}^M, \mathcal{D}_{0,t'}^M} + \sigma_y \mathbf{I}_N)^{-1} \widehat{\mathbf{y}}$, where $\mathbf{K}_{\mathcal{D}_{0,t}^M, \mathcal{D}_{0,t'}^M}$ is a matrix with entries $k_{\mathcal{D}_{0,t}^M, \mathcal{D}_{0,t'}^M}(t_n, t_{n'})$

and $\widehat{\mathbf{y}}$ are noisy observations of $\mathcal{D}_0^M y(t)$.

Although, we presented the above methods in the context of linear ODEs, solutions by numerical approximations and collocation methods are applied to non-linear ODEs as well.

Gaussian processes have been used as models for systems identification (Solak et al., 2003; Kocijan et al., 2005; Thompson, 2009). In Solak et al. (2003), a non-linear dynamical system is linearized around an equilibrium point by means of a Taylor series expansion (Thompson, 2009),

$$y(t) = \sum_{j=0}^{\infty} \frac{y^{(j)}(a)}{j!}(t-a)^j,$$

with a the equilibrium point. For a finite value of terms, the linearization above can be seen as a regression problem in which the covariates correspond to the terms $(t-a)^j$, and the derivatives $y^{(j)}(a)$ as regression coefficients. The derivatives are assumed to follow a Gaussian process prior with a covariance function that is obtained as $k^{(j,j')}(t,t')$, where the superscript j indicates how many derivative of $k(t,t')$ are taken with respect to t and the superscript j' indicates how many derivatives of $k(t,t')$ are taken with respect to t'. Derivatives are then estimated a posteriori through standard Bayesian linear regression. An important consequence of including derivative information in the inference process is that the uncertainty in the posterior prediction is reduced as compared to using only function observations. This aspect of derivative information have been exploited in the theory of computer emulation to reduce the uncertainty in experimental design problems (Morris et al., 1993; Mitchell and Morris, 1994).

Gaussian processes have also been used to model the output $y(t)$ at time t_k as a function of its L previous samples $\{y(t-t_{k-l})\}_{l=1}^{L}$, a common setup in the classical theory of systems identification (Ljung, 1999). The particular dependency $y(t) = g(\{y(t-t_{k-l})\}_{l=1}^{L})$, where $g(\cdot)$ is a general non- linear function, is modelled using a Gaussian process prior, and the predicted value for the output $y_*(t_k)$ is used as a new input for multi-step ahead prediction at times t_j, with $j > k$ (Kocijan et al., 2005). Uncertainty about $y_*(t_k)$ can also be incorporated for predictions of future output values (Girard et al., 2003).

There has been a recent interest in introducing Gaussian processes in the state space formulation of dynamical systems (Ko et al., 2007; Deisenroth et al., 2009; Turner et al., 2010) for the representation of the possible nonlinear relationships

between the latent space, and between the latent space and the observation space. Going back to the formulation of the dimensionality reduction model, we have

$$\mathbf{u}_{t_n} = \mathbf{g}_1(\mathbf{u}_{t_{n-1}}) + \boldsymbol{\eta},$$
$$\mathbf{y}_{t_n} = \mathbf{g}_2(\mathbf{u}_{t_n}) + \boldsymbol{\epsilon},$$

where $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ are noise processes and $\mathbf{g}_1(\cdot)$ and $\mathbf{g}_2(\cdot)$ are general non-linear functions. Usually $\mathbf{g}_1(\cdot)$ and $\mathbf{g}_2(\cdot)$ are unknown, and research on this area has focused on developing a practical framework for inference when assigning Gaussian process priors to both functions.

Finally, it is important to highlight here that in chapter 2 we introduced the work of Calder (Calder, 2003, 2007, 2008) as an alternative to multiple-output modeling. Her work can be seen in the context of state-space models,

$$\mathbf{u}_{t_n} = \mathbf{u}_{t_{n-1}} + \boldsymbol{\eta},$$
$$\mathbf{y}_{t_n} = \mathbf{G}_{t_n}\mathbf{u}_{t_n} + \boldsymbol{\epsilon},$$

where $\mathbf{y}_{t_n}$ and $\mathbf{u}_{t_n}$ are related through a discrete convolution over an independent spatial variable. This is, for a fixed t_n, $y_d^{t_n}(\mathbf{s}) = \sum_q \sum_i G_d^{t_n}(\mathbf{s} - \mathbf{z}_i) u_q^{t_n}(\mathbf{z}_i)$ for a grid of I spatial inputs $\{\mathbf{z}_i\}_{i=1}^I$.

3.5 Summary

In this chapter we introduced the latent force model. A latent force model combines linear differential equations with Gaussian processes to formulate a probabilistic generative model of the data.

The moving-average functions $G_d(\mathbf{x})$ from chapter 2 were chosen in this chapter as the Green's functions associated to several differential equations. This is an example of how we may introduce prior knowledge into the convolved multiple output covariance formulation, namely, through the specification of a linear differential equation. An important point here is that we actually construct a kernel for multiple outputs that encodes dynamical interactions between different systems, allowing the use of non-parametric methods for prediction.

In some biology-oriented applications the differential equation is not a direct function of the latent force $u(t)$, but of a non-linear transformation of $u(t)$, $g(u(t))$.

Although, from the functional perspective, the solution for $f_d(t)$ is still a convolution transform, the joint process $\{f_d(t)\}_{d=1}^{D}$ is not a Gaussian process anymore due to the non-linearity $g(\cdot)$. In those cases, the posterior distribution can be still approximated using a Laplace approximation (Lawrence et al., 2007) or sampling (Titsias et al., 2009).

For the above presentation of the latent force model, we assumed that the covariances $k_q(t, t')$ were squared-exponential. However, more structured covariances can be used. For example, in Honkela et al. (2010), the authors used a cascaded system to describe gene expression data for which a first order system, like the one presented in equation (3.4), has as inputs $u_q(t)$ Gaussian processes with covariance function (3.8).

In publication **ii**, we applied the latent force model of order $M = 1$ to gene expression data with several transcription factors, as opposed to Lawrence et al. (2007) where only a single transcription factor was considered. Additionally, we established a physical interpretation of the multivariate Gaussian covariance presented in section 2.1 as the latent force model covariance obtained from a Heat equation with p input variables (Polyanin, 2002).

In the next chapter, we will present various methods that allow learning and inference of latent force models for large values of N and D.

Chapter 4

Efficient Approximations

In this thesis, the parameters in the covariance function for the multivariate Gaussian processes are estimated using type II maximum likelihood. In type II maximum likelihood, a gradient descent method is usually employed to obtain point estimates of the parameter vector $\boldsymbol{\phi}$. The objective function corresponds to the logarithm of the marginal likelihood in equation (2.20), and the gradients with respect to the parameters are computed according to (2.21). In both expressions, the most computationally expensive step is the inversion of the covariance matrix $\mathbf{K}_{\mathbf{y},\mathbf{y}}$. Gradient descent methods require the iterative evaluation of both equations (2.20) and (2.21), and, thus, the continuous evaluation of the inverse of the covariance matrix.

The preferred method for finding the inverse of the symmetric positive definite matrix $\mathbf{K}_{\mathbf{y},\mathbf{y}}$ is the *Cholesky decomposition*, which has computational complexity of $O(D^3N^3)$ and associated storage of $O(D^2N^2)$. The polynomial complexity in both the computation of the inverse matrix and its storage makes that the direct use of the matrix $\mathbf{K}_{\mathbf{y},\mathbf{y}}$ becomes prohibitively expensive for values of DN greater than a few hundreds.

In recent years, different efforts in the GP machine learning community have been spent in finding ways to reduce the computational complexity associated with the inversion of $\mathbf{K}_{\mathbf{y},\mathbf{y}}$ for the case of a single output (see Quiñonero-Candela and Rasmussen, 2005b, for a review).

In this chapter, our main concern is reducing computational complexity for multivariate Gaussian processes regression in the context of type II marginal likelihood parameter estimation. We first show how through making specific conditional independence assumptions, inspired by the model structure, we arrive at a series of

efficient approximations that represent the covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ using a reduced rank approximation $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$ plus a matrix $\mathbf{D}$, where $\mathbf{D}$ has a specific structure that depends on the particular independence assumption made to obtain the approximation. Approximations can reduce the computational complexity to $O(NDK^2)$ and storage to $O(NDK)$ with K representing a user specified value that determines the rank of the approximation. These approximations were presented in publication **i** and further developed in publication **v**.

The entries in the covariance matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$, which is used to obtain the reduced rank matrix $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$, are evaluated at a set of points $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^{K}$ known in the literature as the *inducing points* or the *inducing inputs*. The location of these inducing points is obtained by maximizing the approximated marginal likelihood. In practice, though, such maximization can lead to overfitting. Recently Titsias (2009) introduced a new inducing point approximation through a variational formalism. In this approach, the inducing points are treated as additional variational parameters, and their locations can be optimized without fear of overfitting. The variational lower bound obtained has a similar form to the approximation provided by the Deterministic Training Conditional (DTC) approximation (Csató and Opper, 2001; Seeger et al., 2003; Quiñonero-Candela and Rasmussen, 2005b). It differs through an additional trace term that favors configurations of inducing points for which the variance of the posterior process at the data points is minimized. We show how the variational approach may be extended to the multiple output case.

All the approximations described above rely on summarizing the behavior of the base process $u(\mathbf{x})$ through a set of *inducing variables*, $\mathbf{u} = [u(\mathbf{z}_1), \ldots, u(\mathbf{z}_K)]^{\top}$, this is, the process $u(\mathbf{x})$ evaluated at $\mathbf{Z}$, the inducing inputs. If the behavior of the base process is accurately represented by the inducing variables, then the approximations work well. However, if the base process is rapidly fluctuating (for example it is white noise) then it will not be well characterized by the inducing points. Our final contribution is to extend the concept of inducing variables to *inducing functions* which arise by convolving the base process with *variational inducing kernels* (VIK). By applying the variational framework to the inducing function, rather than the latent base process, we can approximate multiple output models even for the case where the base process is white noise. In this sense, we introduce approximations for the process convolution framework of section 2.1.2, in particular, for Dependent Gaussian processes. The variational version of the

DTC approximation for multiple outputs and the concepts of inducing function and inducing kernel were presented in publication **iii**. More technical details can be found in Álvarez et al. (2009).

The chapter is organized as follows. In section 4.1, we describe the set of reduced rank approximations. In section 4.2, we illustrate the performance of these approximations over a subset of the gene expression dataset already described in section 2.3. The variational approximation and the extension to base processes which are white Gaussian noise are presented in section 4.3. Finally, in section 4.4, we present alternative approaches that have been used for efficient computations in multivariate Gaussian processes.

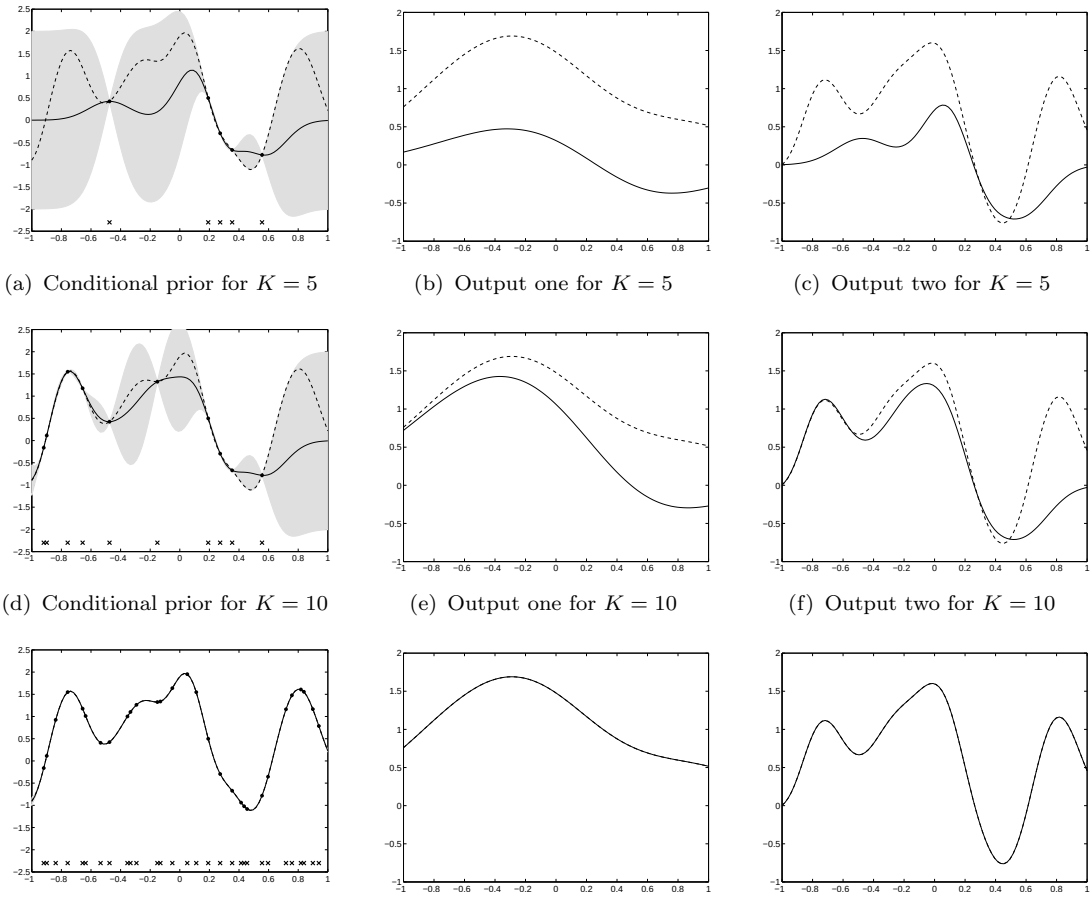
Remark. Section 4.1 and section 4.2 appear with minor modifications in publication **v**. Section 4.3 appears with minor modifications in publication **iii** and the technical report Álvarez et al. (2009). Section 4.4 includes some additional related work that did not appear in the publications.

4.1 Latent functions as conditional means

For notational simplicity, we restrict the analysis of the approximations to one latent function $u(\mathbf{x})$ (this is, $Q = 1$ and $R_q = 1$). The key to all approximations is based on the form we assume for the latent functions. From the perspective of a generative model, equation (2.11) can be interpreted as follows: first we draw a sample from the Gaussian process prior $p(u(\mathbf{z}))$ and then solve the integral for each of the outputs $f_d(\mathbf{x})$ involved. Uncertainty about $u(\mathbf{z})$ is also propagated through the convolution transform.

For the following set of approximations, instead of drawing a sample from $u(\mathbf{z})$, we first draw a sample from a finite representation of $u(\mathbf{z})$, $\mathbf{u}_\mathbf{Z} = [u(\mathbf{z}_1), \ldots, u(\mathbf{z}_K)]^\top$, where $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^K$ is the set of input vectors at which $u(\mathbf{z})$ is evaluated. The vectors $\mathbf{Z}$ are usually known as the *inducing points*, while the elements in $\mathbf{u}_\mathbf{Z}$ are known as the *inducing variables*. Due to the properties of a Gaussian process, $p(\mathbf{u}_\mathbf{Z})$ follows a multivariate Gaussian distribution. Conditioning on $\mathbf{u}_\mathbf{Z}$, we next sample from the conditional prior $p(u(\mathbf{z})|\mathbf{u}_\mathbf{Z})$ and use this function to solve the convolution integral for each $f_d(\mathbf{x})$.[1] Under this generative approach, we can

[1]For simplicity in the notation, we just write $\mathbf{u}$ to refer to $\mathbf{u}_\mathbf{Z}$.



(a) Conditional prior for $K = 5$ (b) Output one for $K = 5$ (c) Output two for $K = 5$

(d) Conditional prior for $K = 10$ (e) Output one for $K = 10$ (f) Output two for $K = 10$

(g) Conditional prior for $K = 30$ (h) Output one for $K = 30$ (i) Output two for $K = 30$

Figure 4.1: Conditional prior and two outputs for different values of K. The first column, figures 4.1(a), 4.1(d) and 4.1(g), shows the mean and confidence intervals of the conditional prior distribution using one input function and two output functions. The dashed line represents a sample from the prior. Conditioning over a few points of this sample, shown as black dots, the conditional mean and conditional covariance are computed. The solid line represents the conditional mean and the shaded region corresponds to 2 standard deviations away from the mean. The second column, 4.1(b), 4.1(e) and 4.1(h), shows the solution to equation (2.11) for output one using a sample from the prior (dashed line) and the conditional mean (solid line), for different values of K. The third column, 4.1(c), 4.1(f) and 4.1(i), shows the solution to equation (2.11) for output two, again for different values of K.

approximate each function $f_d(\mathbf{x})$ using

$$f_d(\mathbf{x}) \approx \int_{\mathcal{X}} G_d(\mathbf{x} - \mathbf{z}) \, \mathrm{E}\left[u(\mathbf{z})|\mathbf{u}\right] \mathrm{d}\mathbf{z}. \tag{4.1}$$

Replacing $u(\mathbf{z})$ for $\mathrm{E}\left[u(\mathbf{z})|\mathbf{u}\right]$ is a reasonable approximation as long as $u(\mathbf{z})$ is a smooth function so that the infinite dimensional object $u(\mathbf{z})$ can be summarized by $\mathbf{u}$. Figure 4.1 shows a cartoon example of the quality of the approximations

for two outputs as the size of the set $\mathbf{Z}$ increases. The first column represents the conditional prior $p\left(u(\mathbf{z})|\mathbf{u}\right)$ for a particular choice of $u(\mathbf{z})$. The second and third columns represent the outputs $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ obtained using equation (4.1).

The conditional probability function $p(u(\mathbf{z})|\mathbf{u})$ is given as

$$p(u(\mathbf{z})|\mathbf{u}) = \mathcal{N}(u(\mathbf{z})|\mathbf{k}_{u,\mathbf{u}}^\top(\mathbf{z},\mathbf{Z})\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, k_{u,u}(\mathbf{z},\mathbf{z}') - \mathbf{k}_{u,\mathbf{u}}^\top(\mathbf{z},\mathbf{Z})\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{k}_{u,\mathbf{u}}(\mathbf{z}',\mathbf{Z})),$$

where $\mathbf{k}_{u,\mathbf{u}}(\mathbf{z},\mathbf{Z})$ is a vector with elements $\{k_{u,u}(\mathbf{z},\mathbf{z}_k)\}_{k=1}^K$, and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is the covariance matrix between the samples from the latent function $\mathbf{u}_\mathbf{Z}$, with elements given by $k_{u,u}(\mathbf{z},\mathbf{z}')$.

We use this conditional probability prior over $u(\mathbf{x})$ to marginalize the latent functions in the expression for each output $\{f_d(\mathbf{x})\}_{d=1}^D$, appearing in equation (2.11). Under this linear model, the likelihood function for $f_d(\mathbf{x})$ is Gaussian, and we need to compute $\mathrm{E}_{u|\mathbf{u}}\left[f_d(\mathbf{x})\right]$ and $\mathrm{cov}_{u|\mathbf{u}}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right]$, where the expectation and the covariance are taken under $p(u(\mathbf{z})|\mathbf{u})$.

The expected value $\mathrm{E}_{u|\mathbf{u}}\left[f_d(\mathbf{x})\right]$ under $p(u(\mathbf{z})|\mathbf{u})$ is given by

$$\mathrm{E}_{u|\mathbf{u}}\left[f_d(\mathbf{x})\right] = \int_\mathcal{X} G_d(\mathbf{x}-\mathbf{z})\,\mathrm{E}_{u|\mathbf{u}}[u(\mathbf{z})]\mathrm{d}\mathbf{z} = \mathbf{k}_{f_d,\mathbf{u}}^\top(\mathbf{x},\mathbf{Z})\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},$$

where $\mathbf{k}_{f_d,\mathbf{u}}(\mathbf{x},\mathbf{Z})$ is a vector with elements $\{k_{f_d,u}(\mathbf{x},\mathbf{z}_k)\}_{k=1}^K$ computed through equation (2.14).

The covariance $\mathrm{cov}_{u|\mathbf{u}}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right]$ under $p(u(\mathbf{z})|\mathbf{u})$ follows

$$\begin{aligned}\mathrm{cov}_{u|\mathbf{u}}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right] &= \int_\mathcal{X} G_d(\mathbf{x}-\mathbf{z}) \int_\mathcal{X} G_{d'}(\mathbf{x}'-\mathbf{z}')\,\mathrm{cov}_{u|\mathbf{u}}[u(\mathbf{z}), u(\mathbf{z}')]\mathrm{d}\mathbf{z}\mathrm{d}\mathbf{z}' \\ &= k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') - \mathbf{k}_{f_d,\mathbf{u}}^\top(\mathbf{x},\mathbf{Z})\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{k}_{f_d,\mathbf{u}}(\mathbf{x}',\mathbf{Z}),\end{aligned}$$

where $k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}')$ is computed through equation (2.12).

Using the above expressions, for a set of input data $\mathbf{X}$, the likelihood function for $\mathbf{f}$ is expressed as

$$p(\mathbf{f}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\phi}) = \mathcal{N}\left(\mathbf{f}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^\top\right), \tag{4.2}$$

where $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ is the cross-covariance matrix between the latent function $u(\mathbf{z})$ and the outputs $f_d(\mathbf{x})$, with elements $\mathrm{cov}\left[f_d(\mathbf{x}), u(\mathbf{z})\right]$ in (2.14). Given the set of points $\mathbf{u}$, we can have different assumptions about the uncertainty of the outputs

in the likelihood term. For example, we could assume that the outputs are independent or uncorrelated, keeping only the uncertainty involved for each output in the likelihood term. Another approximation assumes that the outputs are deterministic, that is $\mathbf{K}_{\mathbf{f},\mathbf{f}} = \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$. The only uncertainty left would be due to the conditional prior $p(u|\mathbf{u})$. Next, we present different approximations of the covariance of the likelihood that lead to a reduction in computational complexity.

Partial Independence

We assume that the individual outputs in $\mathbf{f}$ are independent given the latent variable samples $\mathbf{u}$, leading to the following expression for the likelihood

$$\begin{aligned} p(\mathbf{f}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\phi}) &= \prod_{d=1}^{D} p(\mathbf{f}_d|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\phi}) \\ &= \prod_{d=1}^{D} \mathcal{N}\left(\mathbf{f}_d|\mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f}_d,\mathbf{u}}^{\top}\right). \end{aligned}$$

We rewrite this product of multivariate Gaussian distributions as a single Gaussian distribution with a block diagonal covariance matrix, including the uncertainty about the independent processes

$$p(\mathbf{y}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\phi}) = \mathcal{N}\left(\mathbf{y}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{D}+\boldsymbol{\Sigma}\right), \tag{4.3}$$

where $\mathbf{D} = \text{blockdiag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right]$, and we have used the notation blockdiag $[\mathbf{G}]$ to indicate that the block associated with each output of the matrix $\mathbf{G}$ should be retained, but all other elements should be set to zero. We can also write this as $\mathbf{D} = \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right] \odot \mathbf{M}$ where $\odot$ is the Hadamard product and $\mathbf{M} = \mathbf{I}_D \otimes \mathbf{1}_N$, $\mathbf{1}_N$ being the $N \times N$ matrix of ones. We now marginalize the values of the samples from the latent function by using its process prior, this means $p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}})$. This leads to the following marginal likelihood,

$$\begin{aligned} p(\mathbf{y}|\mathbf{Z},\mathbf{X},\boldsymbol{\phi}) &= \int p(\mathbf{y}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\phi}) p(\mathbf{u}|\mathbf{Z}) \mathrm{d}\mathbf{u}, \\ &= \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{D} + \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top} + \boldsymbol{\Sigma}\right). \end{aligned} \tag{4.4}$$

Notice that, compared to (2.19), the full covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ has been replaced by the low rank covariance $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$ in all entries except in the diagonal

blocks corresponding to $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$. Depending on our choice of K, the inverse of the low rank approximation to the covariance is either dominated by a $O(DN^3)$ term or a $O(K^2DN)$ term. Storage of the matrix is $O(N^2D) + O(NDK)$. Note that if we set $K = N$, these reduce to $O(N^3D)$ and $O(N^2D)$ respectively. Rather neatly this matches the computational complexity of modeling the data with D independent Gaussian processes across the outputs.

The functional form of (4.4) is almost identical to that of the partially independent training conditional (PITC) approximation (Quiñonero-Candela and Rasmussen, 2005b) or the partially independent conditional (PIC) approximation (Quiñonero-Candela and Rasmussen, 2005b; Snelson and Ghahramani, 2007), with the samples we retain from the latent function providing the same role as the *inducing values* in the PITC or PIC.[2] This is perhaps not surprising given that the PITC or PIC approximations are also derived by making conditional independence assumptions. A key difference is that in PITC and PIC it is not obvious which variables should be grouped together when making these conditional independence assumptions; here it is clear from the structure of the model that each of the outputs should be grouped separately.

Full Independence

We can be inspired by the analogy of our approach to the PI(T)C approximation and consider a more radical factorization of the likelihood term. In the fully independent training conditional (FITC) approximation or the fully independent conditional (FIC) approximation (Snelson and Ghahramani, 2006, 2007), a factorization across the data points is assumed. For us that would lead to the following expression for the conditional distribution of the output functions given the inducing variables,

$$p(\mathbf{f}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\phi}) = \prod_{d=1}^{D}\prod_{n=1}^{N} p(f_{n,d}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\phi}),$$

which can be briefly expressed through (4.3) with $\mathbf{D} = \operatorname{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right]$ $= \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right] \odot \mathbf{M}$, with $\mathbf{M} = \mathbf{I}_D \otimes \mathbf{I}_N$ or simply $\mathbf{M} = \mathbf{I}_{DN}$. The marginal likelihood, including the uncertainty about the independent processes,

[2] We refer to both PITC and PIC by PI(T)C.

is given by equation (4.4) with the diagonal form for $\mathbf{D}$. Training with this approximated likelihood reduces computational complexity to $O(K^2DN)$ and the associated storage to $O(KDN)$.

Deterministic likelihood

In Quiñonero-Candela and Rasmussen (2005b), the relationship between the projected process approximation (Csató and Opper, 2001; Seeger et al., 2003) and the FI(T)C and PI(T)C approximations is elucidated. They show that if, given the set of values $\mathbf{u}$, the outputs are assumed to be deterministic, the likelihood term of equation (4.2) can be simplified as

$$p(\mathbf{f}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\phi}) = \mathcal{N}\left(\mathbf{f}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{0}\right).$$

Marginalizing with respect to the latent function using $p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}})$, and including the uncertainty about the independent processes, we obtain the marginal likelihood as

$$p(\mathbf{y}|\mathbf{Z}, \mathbf{X}, \boldsymbol{\phi}) = \int p(\mathbf{y}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\phi})p(\mathbf{u}|\mathbf{Z})\mathrm{d}\mathbf{u} = \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top} + \boldsymbol{\Sigma}\right).$$

In other words, we approximate the full covariance $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ using the low rank approximation $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$. Employing this new marginal likelihood to estimate the parameters $\boldsymbol{\phi}$ reduces computational complexity to $O(K^2DN)$ and the associated storage to $O(KDN)$. The approximation obtained has similarities with the projected latent variables (PLV) method also known as the projected process approximation (PPA) or the deterministic training conditional (DTC) approximation (Csató and Opper, 2001; Seeger et al., 2003; Quiñonero-Candela and Rasmussen, 2005b; Rasmussen and Williams, 2006; Boyle, 2007).

Additional independence assumptions.

As mentioned before, we can consider different conditional independence assumptions for the likelihood term. One further assumption that is worth mentioning considers conditional independencies across data points and dependence across

outputs. This would lead to the following likelihood term

$$p(\mathbf{f}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\phi}) = \prod_{n=1}^{N} p(\overline{\mathbf{f}}_n|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\phi}),$$

where $\overline{\mathbf{f}}_n = [f_1(\mathbf{x}_n), f_2(\mathbf{x}_n), \ldots, f_D(\mathbf{x}_n)]^\top$. We can use again equation (4.3) to express the likelihood. In this case, though, the matrix $\mathbf{D}$ is a partitioned matrix with blocks $\mathbf{D}_{d,d'} \in \Re^{N\times N}$ and each block $\mathbf{D}_{d,d'}$ would be given as $\mathbf{D}_{d,d'} = \text{diag}\left[\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f}_{d'},\mathbf{u}}^\top\right]$. For cases in which $D > N$, that is, the number of outputs is greater than the number of data points, this approximation may be more accurate than the one obtained with the partial independence assumption. For cases where $D < N$ it may be less accurate, but faster to compute.[3]

4.1.1 Posterior and predictive distributions

Combining the likelihood term for each approximation with $p(\mathbf{u}|\mathbf{Z})$ using Bayes' theorem, the posterior distribution over $\mathbf{u}$ is obtained as

$$p(\mathbf{u}|\mathbf{y}, \mathbf{X}, \mathbf{Z}, \boldsymbol{\phi}) = \mathcal{N}\left(\mathbf{u}|\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^\top(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{y}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{u}}\right), \tag{4.5}$$

where $\mathbf{A} = \mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}$ and $\mathbf{D}$ follows a particular form according to the different approximations: it equals $\mathbf{D} = \text{blockdiag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^\top\right]$ for partial independence, it is $\mathbf{D} = \text{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^\top\right]$ for the full independence, and $\mathbf{D} = \mathbf{0}$ for the deterministic likelihood. The posterior distribution over $u(\mathbf{x})$ is given by

$$\begin{aligned} p(u|\mathbf{y}, \mathbf{X}, \mathbf{Z}, \boldsymbol{\phi}) &= \int p(u|\mathbf{u}, \mathbf{Z}, \boldsymbol{\phi}) p(\mathbf{u}|\mathbf{y}, \mathbf{X}, \mathbf{Z}, \boldsymbol{\phi}) \mathrm{d}\mathbf{u}, \\ &= \mathcal{N}(u(\mathbf{x})|\mu_u^{\text{POST}}(\mathbf{x}), k_{u,u}^{\text{POST}}(\mathbf{x}, \mathbf{x}')), \end{aligned}$$

[3]Notice that if we work with the block diagonal matrices $\mathbf{D}_{d,d'}$, we would need to invert the full matrix $\mathbf{D}$. However, since the blocks $\mathbf{D}_{d,d'}$ are diagonal matrices themselves, the inversion can be done efficiently using, for example, a block Cholesky decomposition. Furthermore, we would be restricted to work with isotopic input spaces. Alternatively, we could rearrange the elements of the matrix $\mathbf{D}$ so that the blocks of the main diagonal are the covariances associated with the vectors $\overline{\mathbf{f}}_n$.

where

$$\mu_u^{\text{POST}}(\mathbf{x}) = \mathbf{k}_{u,\mathbf{u}}^\top \mathbf{A}^{-1} \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top (\mathbf{D} + \boldsymbol{\Sigma})^{-1} \mathbf{y},$$
$$k_{u,u}^{\text{POST}}(\mathbf{x}, \mathbf{x}') = k_{u,u}(\mathbf{x}, \mathbf{x}') - \mathbf{k}_{u,\mathbf{u}}^\top (\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} - \mathbf{A}^{-1}) \mathbf{k}_{u,\mathbf{u}},$$

where the superscript POST stands for posterior.

For computing the predictive distribution we have two options, either use the posterior for $\mathbf{u}$ and the approximated likelihoods or the posterior for $\mathbf{u}$ and the likelihood of equation (4.2) (plus the corresponding noise term), that corresponds to the likelihood of the model without any approximations. The difference between both options is reflected in the covariance for the predictive distribution. Quiñonero-Candela and Rasmussen (2005b) proposed a taxonomy of different approximations according to the type of likelihood used for the predictive distribution, in the context of single output Gaussian processes.

For the multivariate case, if we choose the approximated likelihoods, the predictive distribution evaluated at $\mathbf{X}_*$ is expressed by

$$\begin{aligned}\widetilde{p}(\mathbf{y}_*|\mathbf{y}, \mathbf{X}, \mathbf{X}_*, \mathbf{Z}, \boldsymbol{\phi}) &= \int \widetilde{p}(\mathbf{y}_*|\mathbf{u}, \mathbf{Z}, \mathbf{X}_*, \boldsymbol{\phi}) p(\mathbf{u}|\mathbf{y}, \mathbf{X}, \mathbf{Z}, \boldsymbol{\phi}) \mathrm{d}\mathbf{u} \\ &= \mathcal{N}\left(\mathbf{y}_*|\widetilde{\boldsymbol{\mu}}_{\mathbf{y}_*}, \widetilde{\mathbf{K}}_{\mathbf{y}_*,\mathbf{y}_*}\right),\end{aligned}$$

where

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

and $\widetilde{p}(\mathbf{y}_*|\mathbf{u}, \mathbf{Z}, \mathbf{X}_*, \boldsymbol{\phi})$ refers to the approximated likelihood. The values for $\mathbf{D}_*$ depend again of the approximations, being $\mathbf{D}_* = \mathbf{0}$ for the deterministic likelihood, $\mathbf{D}_* = \text{diag}\left[\mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top\right]$ for full independence and $\mathbf{D}_* = \text{blockdiag}\left[\mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top\right]$ for partial independence. For the single output case, the deterministic likelihood leads to the *subset of regressors* approximation (Smola and Bartlett, 2001; Quiñonero-Candela and Rasmussen, 2005b); the full independence approximation leads to the fully independent conditional (FIC) (Snelson and Ghahramani, 2006; Quiñonero-Candela and Rasmussen, 2005b); and the partial independence leads to the partially independent conditional (PIC) (Snelson and Ghahramani, 2007; Quiñonero-Candela and Rasmussen, 2005b).

If we choose the exact likelihood term in equation (4.2), the predictive distribution follows as

$$\begin{aligned} p(\mathbf{y}_*|\mathbf{y}, \mathbf{X}, \mathbf{X}_*, \mathbf{Z}, \boldsymbol{\phi}) &= \int p(\mathbf{y}_*|\mathbf{u}, \mathbf{Z}, \mathbf{X}_*, \boldsymbol{\phi}) p(\mathbf{u}|\mathbf{y}, \mathbf{X}, \mathbf{Z}, \boldsymbol{\phi}) \mathrm{d}\mathbf{u} \\ &= \mathcal{N}\left(\mathbf{y}_*|\boldsymbol{\mu}_{\mathbf{y}_*}, \mathbf{K}_{\mathbf{y}_*,\mathbf{y}_*}\right), \end{aligned} \tag{4.6}$$

where

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{y}_*} &= \mathbf{K}_{\mathbf{f}_*,\mathbf{u}} \mathbf{A}^{-1} \mathbf{K}_{\mathbf{f},\mathbf{u}}^\top (\mathbf{D} + \boldsymbol{\Sigma})^{-1} \mathbf{y}, \\ \mathbf{K}_{\mathbf{y}_*,\mathbf{y}_*} &= \mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{u}} \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top + \mathbf{K}_{\mathbf{f}_*,\mathbf{u}} \mathbf{A}^{-1} \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top + \boldsymbol{\Sigma}_*. \end{aligned}$$

For the single output case, the assumption of the deterministic likelihood is equivalent to the deterministic training conditional (DTC) approximation; the full independence approximation leads to the fully independent training conditional (FITC) approximation (Quiñonero-Candela and Rasmussen, 2005b); and the partial independence leads to the partially independent training conditional (PITC) approximation (Quiñonero-Candela and Rasmussen, 2005b).

From a comparison between $\mathbf{K}_{\mathbf{y}_*,\mathbf{y}_*}$ and $\widetilde{\mathbf{K}}_{\mathbf{y}_*,\mathbf{y}_*}$, we can notice that the entries in the predictive covariance matrix $\mathbf{K}_{\mathbf{y}_*,\mathbf{y}_*}$ are higher than the entries in the covariance matrix $\widetilde{\mathbf{K}}_{\mathbf{y}_*,\mathbf{y}_*}$ (for the partial independence assumption, both matrices are equal in the block-diagonal matrices and for the full independence assumption, both matrices are equal in the diagonal), indicating that predictions made using the predictive distribution $\widetilde{p}(\mathbf{y}_*|\mathbf{u}, \mathbf{Z}, \mathbf{X}_*, \boldsymbol{\phi})$ are usually overconfident. This is more critical in the case of the deterministic approximation. For the fully independence assumption and the partial independence assumption, the variance for both predictive distributions is the same, this is $k_{y_*,y_*}(\mathbf{x}_*, \mathbf{x}_*) = \widetilde{k}_{y_*,y_*}(\mathbf{x}_*, \mathbf{x}_*)$.

The similarities of the above approximations for multivariate GPs with respect to the approximations presented in Quiñonero-Candela and Rasmussen (2005b) for single output GPs are such, that we find it convenient to follow the same terminology, and also refer to above approximations as DTC, FITC and PITC approximations for multioutput Gaussian processes.

4.1.2 Model selection in approximated models

The marginal likelihood approximations derived above are functions of both the hyperparameters of the covariance function and the location of the inducing variables. For estimation purposes, there seems to be a consensus in the GP community that hyperparameters for the covariance function can be obtained by maximization of the marginal likelihood.[4] For selecting the inducing variables, though, there are different alternatives that can be used. A simple procedure consists of grouping the input data using a clustering method like K-means and then use the K resulting vectors as inducing variables. More sophisticated alternatives consider that the set of inducing variables must be restricted to be a subset of the input data (Csató and Opper, 2001; Williams and Seeger, 2001). This set of methods require a criterion for choosing the optimal subset of the training points (Smola and Bartlett, 2001; Seeger et al., 2003). Such approximations are truly sparse in the sense that only few data points are needed at the end for making predictions. Recently, Snelson and Ghahramani (2006) suggested using the marginal likelihood not only for the optimization of the hyperparameters in the covariance function, but also for the optimization of the location of these inducing variables. Although, using such procedure to find the optimal location of the inducing inputs might look in principle like an overwhelming optimization problem (inducing points usually appear non-linearly in the covariance function), in practice it has been shown that performances close to the full GP model can be obtained in a fraction of the time that it takes to train the full model (see publication **v**). In that respect, the inducing points that are finally found are optimal in the same optimality sense that the hyperparameters of the covariance function.[5]

4.2 Regression over gene expression data

We now present an example of the approximations described above, for performing multiple output regression over gene expression data. The setup was described in section 2.3. The difference with that example, is that instead of using $D = 50$

[4] As mentioned in chapter 2, full Bayesian approaches are also possible.

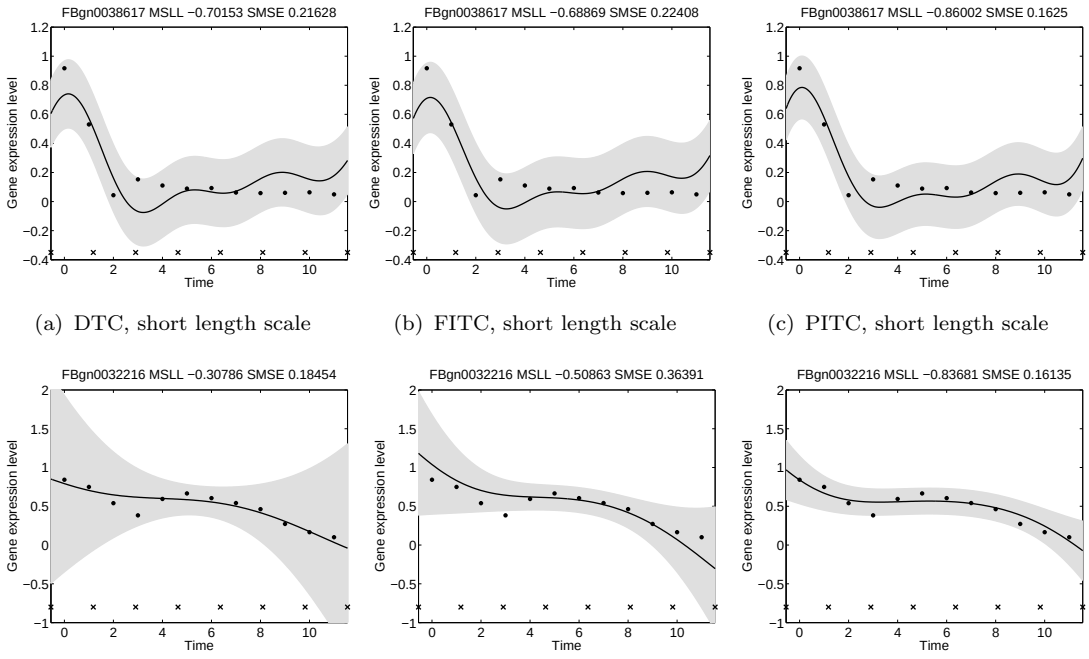
[5] Essentially, it would be possible to use any of the methods just mentioned above together with the multiple-output GP regression models presented in chapter 2. In the software accompanying this thesis, though, we follow Snelson and Ghahramani (2006) and optimize the locations of the inducing variables using the approximated marginal likelihoods.

outputs, here we use $D = 1000$ outputs. We then have access to gene expression data for $D = 1000$ outputs and $N = 12$ time points per output.
We do multiple output regression using DTC, FITC and PITC fixing the number of inducing points to $K = 8$, equally spaced in the interval $[-0.5, 11.5]$. Since it is a 1-dimensional input dataset, we do not optimize the location of the inducing points, but fix them to equally spaced initial positions. As for the full GP model in the example of section 2.3, we set $Q = 1$ and $R_q = 1$. Covariances $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ used in the approximations, are computed using the kernels $k_{f_d,f_{d'}}(\mathbf{x}, \mathbf{x}')$ and $k_{f_d,u_q}(\mathbf{x}, \mathbf{x}')$, derived in equations (2.17) and (2.18), respectively. For the diagonal blocks of $\mathbf{K}_{\mathbf{u},\mathbf{u}}$, we use a Gaussian kernel as in equation (2.15). All these kernels have the appropriate normalization constants.

Train set	Test set	Method	Average SMSE	Average MSLL
Replicate 1	Replicate 2	DTC	0.5421 ± 0.0085	-0.2493 ± 0.0183
		FITC	0.5469 ± 0.0125	-0.3124 ± 0.0200
		PITC	0.5537 ± 0.0136	-0.3162 ± 0.0206
Replicate 2	Replicate 1	DTC	0.5454 ± 0.0173	0.6499 ± 0.7961
		FITC	0.5565 ± 0.0425	-0.3024 ± 0.0294
		PITC	0.5713 ± 0.0794	-0.3128 ± 0.0138

Table 4.1: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the gene expression data for 1000 outputs using the efficient approximations for the convolved multiple output GP. The experiment was repeated ten times with a different set of 1000 genes each time. Table includes the value of one standard deviation over the ten repetitions.

Again we use a scaled conjugate gradient to find the parameters that maximize the marginal likelihood in each approximation using the data from replicate 1. Once we have estimated the hyperparameters of the covariance functions, we test all the approximation methods over the outputs in replicate 2, by conditioning on the outputs of replicate 1. We also repeated the experiment selecting the genes for training from replicate 2 and the corresponding genes of replicate 1 for testing. The optimization procedure runs for 100 iterations.
Table 4.1 shows the results of applying the approximations in terms of SMSE and MSLL. DTC and FITC slightly outperforms PITC in terms of SMSE, but PITC outperforms both DTC and FITC in terms of MSLL. This pattern repeats itself when the training data comes from replicate 1 or from replicate 2.
In figure 4.2 we show the performance of the approximations over the same two genes of figure 2.1, these are FBgn0038617 and FBgn0032216. The non-instantaneous mixing effect of the model can still be observed. Performances for



(a) DTC, short length scale (b) FITC, short length scale (c) PITC, short length scale

(d) DTC, long length scale (e) FITC, long length scale (f) PITC, long length scale

Figure 4.2: Predictive mean and variance for genes FBgn0038617 (first row) and FBgn0032216 (second row) using the different approximations. In the first column DTC (figures 4.2(a) and 4.2(d)), second column FITC (figures 4.2(b) and 4.2(e)) and in the third column PITC (figures 4.2(c) and 4.2(f)). The training data comes from replicate 1 and the testing data from replicate 2. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure. The crosses in the bottom of each figure indicate the positions of the inducing points.

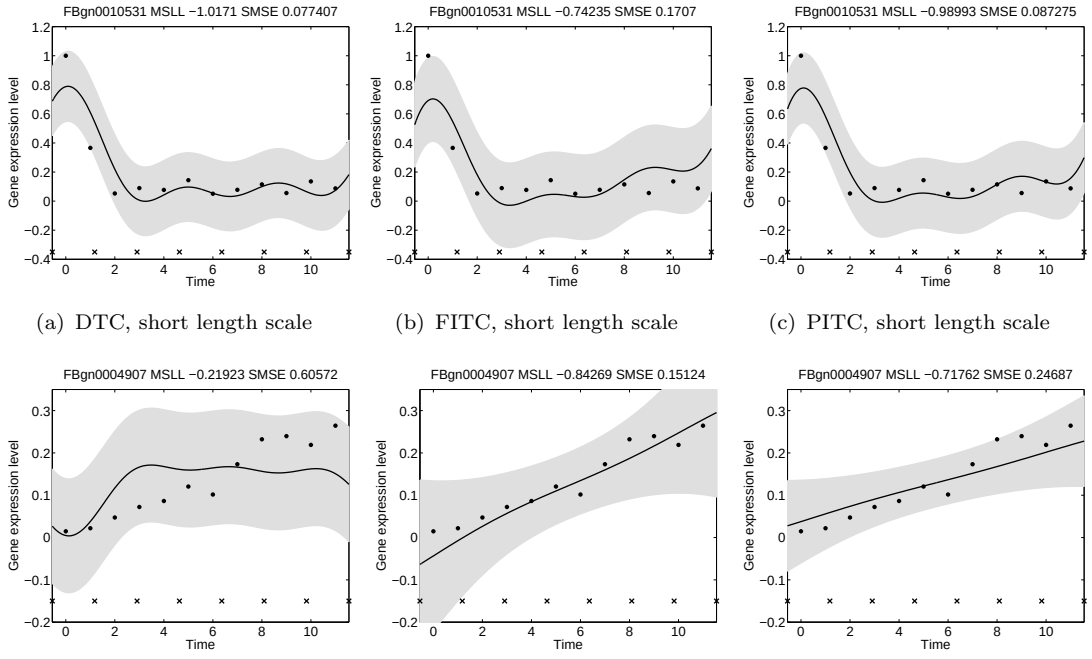
these particular genes are highlighted in table 4.2.

Notice that the performances are between the actual performances for the LMC and the CMOC appearing in table 2.2. We include these figures only for illustrative purposes, since both experiments use a different number of outputs. Figures 2.1 and 2.2 were obtained as part of multiple output regression problem of $D = 50$ outputs, while figures 4.2 and 4.3 were obtained in a multiple output regression problem with $D = 1000$ outputs.

In figure 4.3, we replicate the same exercise for the genes FBgn0010531 and FBgn0004907, that also appeared in figure 2.2. Performances for DTC, FITC and PITC are shown in table 4.2 (last six rows), which compare favourably with the performances for the linear model of coregionalization in table 2.2 and close to the performances for the CMOC. In average, PITC outperforms the other methods for the specific set of genes in both figures.

Test replicate	Test genes	Method	SMSE	MSLL
Replicate 2	FBgn0038617	DTC	0.2162	−0.7015
		FITC	0.2240	−0.6886
		PITC	0.1625	−0.8600
	FBgn0032216	DTC	0.1845	−0.3078
		FITC	0.3639	−0.5086
		PITC	0.1613	−0.8368
Replicate 1	FBgn0010531	DTC	0.0774	−1.0171
		FITC	0.1707	−0.7423
		PITC	0.0872	−0.9899
	FBgn0004907	DTC	0.6057	−0.2192
		FITC	0.1512	−0.8426
		PITC	0.2468	−0.7176

Table 4.2: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the genes in figures 4.2 and 4.3 for DTC, FITC and PITC with $K = 8$.



(a) DTC, short length scale (b) FITC, short length scale (c) PITC, short length scale

(d) DTC, long length scale (e) FITC, long length scale (f) PITC, long length scale

Figure 4.3: Predictive mean and variance for genes FBgn0010531 (first row) and FBgn0004907 (second row) using the different approximations. In the first column DTC (figures 4.3(a) and 4.3(d)), second column FITC (figures 4.3(b) and 4.3(e)) and in the third column PITC (figures 4.3(c) and 4.3(f)). The training data comes now from replicate 2 and the testing data from replicate 1. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure. The crosses in the bottom of each figure indicate the positions of the inducing points, which remain fixed during the training procedure.

Train set	Test set	Method	Average TTPI
Replicate 1	Replicate 2	DTC	2.04
		FITC	2.31
		PITC	2.59
Replicate 2	Replicate 1	DTC	2.10
		FITC	2.32
		PITC	2.58

Table 4.3: Training time per iteration (TTPI) for the gene expression data for 1000 outputs using the efficient approximations for the convolved multiple output GP. The experiment was repeated ten times with a different set of 1000 genes each time.

With respect to the training times, table 4.3 shows the average training time per iteration (average TTPI) for each approximation. To have an idea of the saving times, one iteration of the full GP model for the same 1000 genes would take around 4595.3 seconds. This gives a speed up factor of 1780, approximately.[6]

4.3 Variational approximations

In section 4.1 we introduced a series of approximations that reduce computational complexity for performing multivariate regression with Gaussian processes. However, that reduction came at the price of introducing new parameters in the model, namely, the inducing locations $\mathbf{Z}$. Marginal likelihoods now are dependent of $\mathbf{Z}$ and without additional restrictions over them, estimating these quantities and the covariance hyperparameters using the marginal likelihood as cost function, might lead to overfitting. In a Bayesian setup, one would be tempted to put a prior over $\mathbf{Z}$ and then marginalize, but unfortunately $\mathbf{Z}$ appears in the inverse of $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ and performing this integration becomes intractable. One potential detour consists of approximating the marginal likelihood in equation (2.19) directly, finding an expression of the posterior distribution through the minimization of Kullback-Leibler divergences.

In subsection 4.3.1, we follow the lines of Titsias (2009) and propose a variational approximation for multiple output GPs. For this new approximation, we augment the prior over the latent functions and condition this prior on the location of the inducing variables. We assume the posterior over the inducing variables is

[6]The experiments were run in workstations with 2.59 GHz, AMD Opteron's and up to 16 GHz of RAM. Only one processor was used on each run. The speed up factor is computed as the relation between the slower method and the faster method, using the corresponding training times.

unknown and we approximate it by maximizing a lower bound on the exact marginal likelihood given by equation (2.19). This is equivalent to minimizing the Kullback-Leibler distance between the true posterior and the approximated one.

In subsection 4.3.2, we introduce the idea of *inducing kernels*, that enhance the variational approximation for multiple outputs with the ability to deal with latent functions which are white noise processes.

4.3.1 A variational lower bound

In section 4.1, the principle underlying the different approximations was the replacement of the original prior $p(u)$ with the conditional prior $p(u|\mathbf{u})$, under the fundamental assumption that the latent function $u(\mathbf{z})$ might be as well represented by the conditional mean $\mathrm{E}[u(\mathbf{z})|\mathbf{u}_\mathbf{Z}]$, where the uncertainty in $\mathbf{u}_\mathbf{Z}$ is described by $p(\mathbf{u}_\mathbf{Z})$.

The above assumption modifies the initial multivariate Gaussian process model and introduces the inducing inputs $\mathbf{Z}$ as extra kernel hyperparameters in the modified GP model without any restrictions about the values that the vectors $\mathbf{Z}$ might take. The maximization of the marginal likelihood for the modified Gaussian process models (for example in equation (4.4)) with respect to $(\mathbf{Z}, \boldsymbol{\phi})$, may be prone to overfitting especially when the number of variables in $\mathbf{Z}$ is large. Moreover, fitting a modified GP model implies that the full GP model is not approximated in a systematic and rigorous way since there is no distance or divergence between the two models that is minimized.

We follow Titsias (2009) to establish a rigorous lower bound for the marginal likelihood of equation (2.19). Given target data $\mathbf{y}$ and inputs $\mathbf{X}$, the marginal likelihood of the original model is given by integrating over the latent function[7]

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\phi}) = \int_u p(\mathbf{y}|u, \mathbf{X}, \boldsymbol{\phi}) p(u) \mathrm{d}u.$$

Instead of working directly with a prior distribution $p(\mathbf{u})$, as in the approximations of section 4.1, we augment the original prior as $p(u, \mathbf{u}) = p(u|\mathbf{u})p(\mathbf{u})$ with

[7]Strictly speaking, the distributions associated to u correspond to random signed measures, in particular, Gaussian measures.

the consistency relation

$$p(u(\mathbf{z})) = \int_{\mathbf{u}} p(u(\mathbf{z})|\mathbf{u})p(\mathbf{u})\mathrm{d}\mathbf{u}. \tag{4.7}$$

The augmented joint model can be expressed as

$$p\left(\mathbf{y}, u, \mathbf{u}\right) = p(\mathbf{y}|u)p(u|\mathbf{u})p(\mathbf{u}),$$

which can be also written as $p(\mathbf{y}|u)p(u)$ using the marginalization property in (4.7) for the latent function. Using standard variational approximation techniques (see for example Bishop, 2006, chap. 10), we establish a lower bound on the true exact log marginal likelihood $\log p(\mathbf{y})$ by approximating the true posterior $p(u, \mathbf{u}|\mathbf{y})$ with a variational distribution $q(u, \mathbf{u})$.
The variational posterior distribution is taken to be

$$q(u, \mathbf{u}) = p(u|\mathbf{u})\varphi(\mathbf{u}),$$

where $\varphi(\mathbf{u})$ is the unknown a posteriori distribution over the inducing variables. The lower bound is now computed as

$$\begin{aligned}
\mathcal{L}(\mathbf{Z}, \boldsymbol{\phi}, \varphi(\mathbf{u})) &= \int_{u,\mathbf{u}} q(u, \mathbf{u}) \log \left\{ \frac{p(\mathbf{y}, u, \mathbf{u})}{q(u, \mathbf{u})} \right\} \mathrm{d}\mathbf{u}\, \mathrm{d}u \\
&= \int_{u,\mathbf{u}} p(u|\mathbf{u})\varphi(\mathbf{u}) \log \left\{ \frac{p(\mathbf{y}|u)p(u|\mathbf{u})p(\mathbf{u})}{p(u|\mathbf{u})\varphi(\mathbf{u})} \right\} \mathrm{d}\mathbf{u}\, \mathrm{d}u \\
&= \int_{u,\mathbf{u}} p(u|\mathbf{u})\varphi(\mathbf{u}) \log \left\{ \frac{p(\mathbf{y}|u)p(\mathbf{u})}{\varphi(\mathbf{u})} \right\} \mathrm{d}\mathbf{u}\, \mathrm{d}u \\
&= \int_{\mathbf{u}} \varphi(\mathbf{u}) \int_{u} p(u|\mathbf{u}) \left\{ \log p(\mathbf{y}|u) + \log \frac{p(\mathbf{u})}{\varphi(\mathbf{u})} \right\} \mathrm{d}u\, \mathrm{d}\mathbf{u}.
\end{aligned}$$

It can be shown that this bound is given as (see Álvarez et al., 2009, for detailed derivations),

$$\mathcal{L}(\mathbf{Z}, \boldsymbol{\phi}, \varphi(\mathbf{u})) = \int_{\mathbf{u}} \varphi(\mathbf{u}) \log \left\{ \frac{\mathcal{N}\left(\mathbf{y}|\boldsymbol{\alpha}, \boldsymbol{\Sigma}\right) p(\mathbf{u})}{\varphi(\mathbf{u})} \right\} \mathrm{d}\mathbf{u} - \frac{1}{2} \sum_{d=1}^{D} \mathrm{tr}\left(\boldsymbol{\Sigma}_{w_d}^{-1} \widetilde{\mathbf{K}}_{dd} \right),$$

where $\boldsymbol{\alpha} = \left[\boldsymbol{\alpha}_1^\top, \ldots, \boldsymbol{\alpha}_D^\top\right]^\top$ with $\boldsymbol{\alpha}_d = \mathbf{K}_{\mathbf{f}_d\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}$, $\boldsymbol{\Sigma}_{w_d}$ is the covariance matrix associated to the independent process $w_d(\mathbf{x})$, $\widetilde{\mathbf{K}}_{dd} = \mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f}_d,\mathbf{u}}^\top$, and $\boldsymbol{\Sigma}$ is a block-diagonal matrix with blocks given by $\boldsymbol{\Sigma}_{w_d}$.

After analytically maximizing the lower bound with respect to $\varphi(\mathbf{u})$ (which is equivalent to apply Jensen's inequality in the integral over $\mathbf{u}$ for $\log(x)$, this is, a concave function), we have

$$\mathcal{L}(\mathbf{Z}, \boldsymbol{\phi}) = \log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top} + \boldsymbol{\Sigma}\right) - \frac{1}{2}\operatorname{tr}\left(\boldsymbol{\Sigma}^{-1}\widetilde{\mathbf{K}}\right), \tag{4.8}$$

with $\widetilde{\mathbf{K}} = \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$. The approximated covariance appearing in the lower bound, $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$, has the same form as the Deterministic Training Conditional (DTC) approximation discussed in Quiñonero-Candela and Rasmussen (2005b) for a single output. Since this approximation is obtained by applying a variational approximation, we refer to this approximation as DTCVAR. Note that this bound consists of two parts. The first part is the log of a GP prior with the only difference that now the covariance matrix has a particular low rank form. This form allows the inversion of the covariance matrix to take place in $O(NDK^2)$ time rather than $O(N^3D^3)$. The second part can be seen as a penalization term that regulates the estimation of the parameters. Notice also that only the diagonal of the exact covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ needs to be computed, when compared to PI(T)C, for which we need to compute the block covariances $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d}$. According to the complexity of $\operatorname{cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')]$ this might be an important extra computational saving.

The approximated posterior $\varphi(\mathbf{u})$ is obtained by maximizing the lower bound $\mathcal{L}(\mathbf{Z}, \boldsymbol{\phi}, \varphi)$, following a similar expression to (4.5). Equally, the expression for the predictive distribution is similar to the one obtained in (4.6) with $\mathbf{D} = \mathbf{0}$.

4.3.2 Variational inducing kernels

Our key assumption for the approximations in section 4.1 lies in the fact that the latent functions can be summarized using just a few points in the latent space. This assumption appears also implicitly in the variational approximation in subsection 4.3.1. Indeed, when writing the factor $p(u|\mathbf{u})$ in the distribution for the joint model $p(\mathbf{y}, u, \mathbf{u})$ and in the approximated posterior $q(u, \mathbf{u})$, we implicit assume that this infinite-dimensional objects, the functions $\{u_q(\mathbf{x})\}_{q=1}^{Q}$, may be abbreviated using a finite number of points $\{\mathbf{u}_q\}_{q=1}^{Q}$. If the locations of the inducing points, $\{\mathbf{z}_k\}_{k=1}^{K}$, are close relative to the length scale of the latent function, the approximations on section 4.1 will be accurate. However, if the length scale becomes small the approximation requires very many inducing points. In the

worst case, the latent process could be white noise (as suggested by the process convolution constructions of chapter 2). An important class of problems where we have to deal with white noise processes arise in linear *stochastic differential equations* (Álvarez et al., 2010) where the above approximation methods do not reflect the characteristics of the latent functions.

In this subsection, we develop the variational approximation to allow us to work with rapidly fluctuating latent functions. This is achieved by augmenting the output functions with one or more additional functions. We refer to these additional outputs as the *inducing functions*. Our variational approximation is developed through the inducing functions. There are also smoothing kernels associated with the inducing functions. The quality of the variational approximation can be controlled both through these *inducing kernels* and through the number and location of the inducing inputs.

To motivate the idea, we first explain why the $\mathbf{u}$ variables can work when the latent functions are smooth and fail when these functions become white noises.

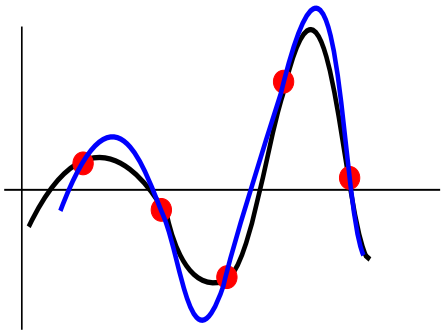
In approximations like PI(T)C, we assume each latent function $u_q(\mathbf{x})$ is smooth and we sparsify the GP model through introducing $\mathbf{u}_q$, this is, inducing variables which are direct observations of the latent function $u_q(\mathbf{x})$, at particular input points. Because of the latent function's smoothness, the $\mathbf{u}_q$ variables also carry information about other points in the function through the imposed prior over the latent function. So, having observed $\mathbf{u}_q$, we can reduce the uncertainty of the whole function $u_q(\mathbf{x})$.

With the vector of inducing variables $\mathbf{u}$, if chosen to be sufficiently large relative to the length scales of the latent functions, we can efficiently represent the functions $\{u_q(\mathbf{x})\}_{q=1}^{Q}$ and subsequently variables $\mathbf{f}$ which are just convolved versions of the latent functions.[8] When the reconstruction of $\mathbf{f}$ from $\mathbf{u}$ is perfect, the conditional prior $p(\mathbf{f}|\mathbf{u})$ becomes a delta function and the PITC-like approximations become exact. Figure 4.4(a) shows a cartoon example of the description of $u_q(\mathbf{x})$ by $\mathbf{u}_q$.

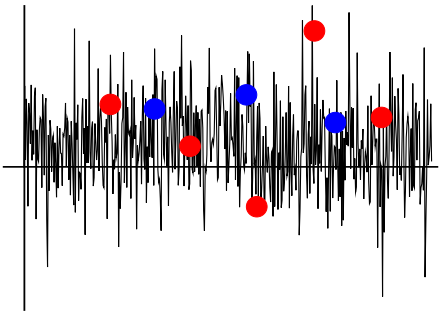
In contrast, when some of the latent functions are *white noise* processes the approximation will fail. If $u_q(\mathbf{z})$ is white noise,[9] it has a covariance function $\delta(\mathbf{z}-\mathbf{z}')$.

[8] This idea is like a "soft version" of the Nyquist-Shannon sampling theorem. If the latent functions were bandlimited, we could compute exact results given a high enough number of inducing points. In general it won't be bandlimited, but for smooth functions low frequency components will dominate over high frequencies, which will quickly fade away.

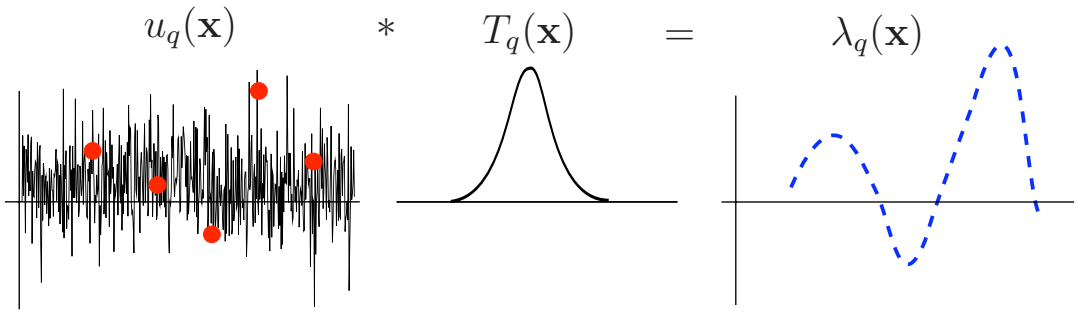
[9] Such a process can be thought as the "time derivative" of the Wiener process.



(a) Latent function is smooth



(b) Latent function is noise



(c) Generation of an inducing function

Figure 4.4: With a smooth latent function as in (a), we can use some inducing variables $\mathbf{u}_q$ (red dots) from the complete latent process $u_q(\mathbf{x})$ (in black) to generate smoothed versions (for example the one in blue), with uncertainty described by $p(u_q|\mathbf{u}_q)$. However, with a white noise latent function as in (b), choosing inducing variables $\mathbf{u}_q$ (red dots) from the latent process (in black) does not give us any information about other points (for example the blue dots). In (c) the inducing function $\lambda_q(\mathbf{x})$ acts as a surrogate for a smooth function. Indirectly, it contains information about the inducing points and it can be used in the computation of the lower bound. In this context, the symbol $*$ refers to the convolution integral.

Such processes naturally arise in the application of *stochastic differential equations* and are the ultimate non-smooth processes where two values $u_q(\mathbf{z})$ and $u_q(\mathbf{z}')$ are uncorrelated when $\mathbf{z} \neq \mathbf{z}'$. When we apply the approximation, a vector of "white-noise" inducing variables $\mathbf{u}_q$ does not carry information about $u_q(\mathbf{z})$ at any input $\mathbf{z}$ that differs from all inducing inputs $\mathbf{Z}$. In other words there is no additional information in the conditional prior $p(u_q(\mathbf{z})|\mathbf{u}_q)$ over the unconditional prior $p(u_q(\mathbf{z}))$. Figure 4.4(b) shows a pictorial representation. The lack of structure makes it impossible to exploit the correlations in standard methods like PI(T)C.[10]

[10]Returning to our sampling theorem analogy, the white noise process has infinite bandwidth. It is therefore impossible to represent it by observations at a few fixed inducing points.

Our solution to this problem is the following. We will define a more powerful form of inducing variable, one based not around the latent function at a point, but one given by the convolution of the latent function with a smoothing kernel. More precisely, let us replace each inducing vector $\mathbf{u}_q$ with the variables $\boldsymbol{\lambda}_q$ which are evaluated at the inputs $\mathbf{Z}$ and are defined according to

$$\lambda_q(\mathbf{z}) = \int T_q(\mathbf{z} - \mathbf{v}) u_q(\mathbf{v}) \mathrm{d}\mathbf{v}, \tag{4.9}$$

where $T_q(\mathbf{x})$ is a smoothing kernel which we call the *inducing kernel* (IK). This kernel is not necessarily related to the model's smoothing kernels. Notice that the new inducing variables $\boldsymbol{\lambda}_q$ correspond to a finite set of points taken from the $\lambda_q(\mathbf{z})$ process, this is, $\boldsymbol{\lambda}_q = [\lambda_q(\mathbf{z}_1), \ldots, \lambda_q(\mathbf{z}_K)]^\top$. These newly defined inducing variables can carry information about $u_q(\mathbf{z})$ not only at a single input location but from the entire input space. Figure 4.4(c) shows how the inducing kernel generates the artificial construction $\lambda_q(\mathbf{x})$, that sheds some light over the, otherwise, obscure inducing points. We can even allow a separate IK for each inducing point, this is, if the set of inducing points is $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^K$, then

$$\lambda_q(\mathbf{z}_k) = \int T_{q,k}(\mathbf{z}_k - \mathbf{v}) u_q(\mathbf{v}) \mathrm{d}\mathbf{v}, \tag{4.10}$$

with the advantage of associating to each inducing point $\mathbf{z}_k$ its own set of adaptive parameters in $T_{q,k}$.

If $u_q(\mathbf{z})$ has a white noise GP prior, the covariance function for $\lambda_q(\mathbf{x})$ is

$$\mathrm{cov}[\lambda_q(\mathbf{x}), \lambda_q(\mathbf{x}')] = \int T_q(\mathbf{x} - \mathbf{z}) T_q(\mathbf{x}' - \mathbf{z}) \mathrm{d}\mathbf{z},$$

and the cross-covariance function between $f_d(\mathbf{x})$ and $\lambda_q(\mathbf{x}')$ is

$$\mathrm{cov}[f_d(\mathbf{x}), \lambda_q(\mathbf{x}')] = \int G_{d,q}(\mathbf{x} - \mathbf{z}) T_q(\mathbf{x}' - \mathbf{z}) \mathrm{d}\mathbf{z}.$$

Notice that this cross-covariance function maintains a weighted integration over the whole input space, unlike the case of using $\mathbf{u}$ as the inducing variables. This implies that a single inducing variable $\lambda_q(\mathbf{x})$ can properly propagate information from the full-length process $u_q(\mathbf{x})$ into the set of outputs $\mathbf{f}$.

It is possible to combine the IKs defined above with the approximations of subsection 4.3.1. However, we would arrive again at the problem of overfitting. We

therefore include the inducing functions and inducing kernels within the variational framework of Titsias (2009), and refer to the functions as variational inducing functions (VIFs) and to the kernels as variational inducing kernels (VIKs). A lower bound for the multivariate Gaussian process can be obtained again by replacing the vector of inducing variables $\mathbf{u} = \{\mathbf{u}_q\}_{q=1}^{Q}$ for the vector of variables $\boldsymbol{\lambda} = \{\boldsymbol{\lambda}_q\}_{q=1}^{Q}$. Posterior and predictive distributions are obtained in the same way as in section 4.3.2, replacing the vector $\mathbf{u}$ with the vector $\boldsymbol{\lambda}$ (Álvarez et al., 2009).

4.4 Related work

As we saw in section 4.1, assuming that the latent function $u(\mathbf{x})$ can be replaced by the conditional mean $\mathrm{E}[u(\mathbf{x})|\mathbf{u}_\mathbf{Z}]$ in the convolved multiple output covariance and using different conditional independence assumptions for the likelihood function, we arrived to a series of approximations summarized in a reduced rank valid covariance matrix.

Other alternatives for reducing computational complexity for multivariate Gaussian processes include Ver Hoef et al. (2004) and Boyle (2007, chapter 6). In both works, the covariance function is obtained using the process convolution formalism of subsection 2.1.2.

Ver Hoef et al. (2004) presents a simulation example with $D = 2$. Prediction over one of the variables is performed using cokriging. In cokriging scenarios, usually one has access to a few measurements of a primary variable, but plenty of observations for a secondary variable. Following a suggestion by Stein (1999, p. 172), the authors partition the secondary observations into subgroups of observations and assume the likelihood function is the sum of the partial likelihood functions of several systems that include the primary observations and each of the subgroups of the secondary observations. In other words, the joint probability distribution $p(\mathbf{f}_1, \mathbf{f}_2)$ is factorised as $p(\mathbf{f}_1, \mathbf{f}_2) = \prod_{j=1}^{J} p(\mathbf{f}_2^{(j)}|\mathbf{f}_1)p(\mathbf{f}_1)$, where $\mathbf{f}_2^{(j)}$ indicates the observations in the subgroup j out of J subgroups of observations. It is not clear from this method, though, how the groups in the secondary variable should be selected. The authors recognize that this is an area than requires further research. The authors also use a Fast Fourier Transform for computing the autocovariance matrices $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d}$ and cross-covariance matrices $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$.

Boyle (2007) proposed an extension of the *reduced rank approximation* method of Quiñonero-Candela and Rasmussen (2005a), to be applied to the Dependent

Gaussian process construction. The reduced rank approximation in the single output case, starts with the representation of the output using a generalised linear model, this is, the output $\mathbf{f}_1 = \mathbf{f}$ is represented as $\mathbf{f} = \mathbf{K}_{\mathbf{f},\mathbf{w}}\mathbf{w}$, where $\mathbf{w}$ is a vector of weights with prior distribution $p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \mathbf{K}_{\mathbf{w},\mathbf{w}}^{-1})$, that define the so called *support points*.[11] Elements in both matrices $\mathbf{K}_{\mathbf{f},\mathbf{w}}$ and $\mathbf{K}_{\mathbf{w},\mathbf{w}}$ are computed using, for example, the squared-exponential kernel. The marginal likelihood for the noisy observations $\mathbf{y}$ is given by $p(\mathbf{y}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{w}}\mathbf{K}_{\mathbf{w},\mathbf{w}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{w}}^{\top} + \sigma^2\mathbf{I})$, and the predictive distribution by $p(\mathbf{f}_*|\mathbf{y}) = \mathcal{N}(\mathbf{f}_*|\sigma^{-2}\mathbf{K}_{\mathbf{f}*,\mathbf{w}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{w}}^{\top}\mathbf{y}, \mathbf{K}_{\mathbf{f}*,\mathbf{w}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{f}*,\mathbf{w}}^{\top} + \sigma^2\mathbf{I})$, where $\mathbf{A} = \sigma^{-2}\mathbf{K}_{\mathbf{f},\mathbf{w}}^{\top}\mathbf{K}_{\mathbf{f},\mathbf{w}} + \mathbf{K}_{\mathbf{w},\mathbf{w}}$. It can be noticed that if the number of training points equals the number support points, $N = K$, the predictive mean $\sigma^{-2}\mathbf{K}_{\mathbf{f}*,\mathbf{w}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{w}}^{\top}\mathbf{y}$ converges to the predictive mean of the full Gaussian process. If the kernel used is decaying (for example, it has compact support), then the predictive variance for test inputs away from the support points will be equal to the noise variance. Such a behavior is inadequate, because it underestimates the uncertainty of $f_*(\mathbf{x}_*)$ and this was not originally intended in the full Gaussian process prior. To alleviate this problem, Quiñonero-Candela and Rasmussen (2005a) augmented the generalised linear model with a weight and a basis function localized in the test input $\mathbf{x}_*$. The new vector of weights is given as $\widetilde{\mathbf{w}} = [\mathbf{w}^{\top} w_*]^{\top}$ and the output $\mathbf{f}_1$ is defined as $\mathbf{f} = \mathbf{K}_{\mathbf{f},\widetilde{\mathbf{w}}}\widetilde{\mathbf{w}}$. It can be shown that for $N = K$, the predictive variance for this model converges to the predictive variance of the full GP, even for test inputs away from the training data (Boyle, 2007, chapter 5). From the perspective of Quiñonero-Candela and Rasmussen (2005b), the reduced rank approximation is equivalent to the DTC approximation by augmenting the set of inducing variables with an additional variable corresponding to the test point f_* (of course, we do not need access to the actual value of f_*, since we are working with kernels).

Boyle (2007) presented the development of the same idea for $D = 2$. The outputs $\mathbf{f}_1$ and $\mathbf{f}_2$ are defined as

$$\begin{bmatrix}\mathbf{f}_1\\ \mathbf{f}_2\end{bmatrix} = \begin{bmatrix}\mathbf{K}_{\mathbf{f}_1,\widetilde{\mathbf{w}}_1} & \mathbf{K}_{\mathbf{f}_1,\widetilde{\mathbf{w}}_2}\\ \mathbf{K}_{\mathbf{f}_2,\widetilde{\mathbf{w}}_1} & \mathbf{K}_{\mathbf{f}_1,\widetilde{\mathbf{w}}_2}\end{bmatrix}\begin{bmatrix}\widetilde{\mathbf{w}}_1\\ \widetilde{\mathbf{w}}_2\end{bmatrix},$$

where $\widetilde{\mathbf{w}}_d$ are vectors of weights associated to each output including additional weights corresponding to the test inputs, one for each output. Based on this likelihood, a predictive distribution for the joint prediction of $\mathbf{f}_1$ and $\mathbf{f}_2$ can be

[11] A more recent name for support points is inducing variables.

obtained, with the characteristic that the variances approach to variances for the full predictive of the Gaussian process for test points away from the training data. The elements in the matrices $\mathbf{K}_{\mathbf{f}_d,\widetilde{\mathbf{w}}_{d'}}$ are computed using the covariances and cross-covariances developed in chapter 2. Notice that compared to the approximations presented in sections 4.1 and 4.3, where the inducing inputs are common to all outputs, here the inducing inputs are particular to each output.

On the other hand, the idea of inducing function and inducing kernel is closely related to sparse multiscale Gaussian process regression (Walder et al., 2008) and inter-domain Gaussian processes (Lázaro-Gredilla and Figueiras-Vidal, 2010). Both methods share a similar purpose, namely, augmenting the flexibility of the inducing points in order to improve the quality of the approximations when compared to the traditional defined inducing points used in Snelson and Ghahramani (2006). In a nutshell, in Snelson and Ghahramani (2006), the entries of the covariance matrices appearing in the reduced rank approximation $\mathbf{K}_{\mathbf{f},\bar{\mathbf{f}}}\mathbf{K}_{\bar{\mathbf{f}},\bar{\mathbf{f}}}^{-1}\mathbf{K}_{\mathbf{f},\bar{\mathbf{f}}}^{\top}$, where $\bar{\mathbf{f}}$ is the vector of inducing variables, are computed using the same kernel function $k_{f,f}(\mathbf{x},\mathbf{x}')$ that is used to compute the full rank kernel matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$. In other words, the inducing variables are point evaluations of the single output $f(\mathbf{x})$, this is, $\bar{\mathbf{f}} = [\bar{f}(\mathbf{z}_1),\ldots,\bar{f}(\mathbf{z}_K)]^{\top}$, where $\bar{f}(\mathbf{z}_k) = f(\mathbf{x})\delta(\mathbf{x}-\mathbf{z}_k)$. Lázaro-Gredilla and Figueiras-Vidal (2010) defined the inducing variables $\bar{f}(\mathbf{Z})$ using a similar expression to (4.9), where the process $u_q(\mathbf{x})$ corresponded to $f(\mathbf{x})$. The elements of $\mathbf{K}_{\mathbf{f},\bar{\mathbf{f}}}$ and $\mathbf{K}_{\bar{\mathbf{f}},\bar{\mathbf{f}}}$ are defined by the covariance between $f(\mathbf{x})$ and $\bar{f}(\mathbf{x}')$ and the covariance between $\bar{f}(\mathbf{x})$ and $\bar{f}(\mathbf{x}')$, respectively. The inducing variables can exist in a different domain, through the suitable selection of the *feature extraction function* $T(\mathbf{x},\mathbf{z})$ in equation (4.9). Walder et al. (2008) allowed each of the inducing variables $\{\bar{f}(\mathbf{z}_k)\}_{k=1}^{K}$ to be described by its own parameter vector, θ_{z_k} and developed a kernel basis $\{k_{f,\bar{f}_{z_k}}(\mathbf{x},\mathbf{z}_k)\}_{k=1}^{K}$, where each basis function has its own set of parameters. This is equivalent to equation (4.10), where each inducing point is allowed to have its own inducing kernel. As we said before, the two methods intend to improve performance over the pseudo-inputs approach of Snelson and Ghahramani (2006) in sparse GP models for single outputs, using a new definition of the pseudo-inputs. Our use of inducing functions and inducing kernels is motivated by the need to deal with non-smooth latent functions in the process convolution model for multiple outputs.

4.5 Summary

In this chapter, we presented a series of methods that allow the reduction in computational complexity for multivariate Gaussian processes when parameters are estimated using type II maximum likelihood.

Using these approximations we can capture the correlated information among outputs while reducing the computational overhead involved in the optimization and prediction tasks. The computational complexity for training using the deterministic marginal likelihood and the fully independent marginal likelihood is $O(NDK^2)$. The computational complexity for the PI(T)C approximation reduces from $O(N^3D^3)$ to $O(N^3D)$. This matches the computational complexity for modeling with independent GPs.

In publications **i** and **v** we present applications of this type of approximations for multivariate regression of pollutant metals and exam score predictions. We showed experimentally how the training of the models can be done in a fraction of the time it takes to train the full Gaussian process without sacrificing the prediction performance.

We presented also a variational version of the deterministic training conditional approximation for multiple outputs, that allows for a rigorous measure of the distance between the true marginal likelihood and the approximated one. We have introduced the concept of an inducing function, which generalizes the idea of inducing point, traditionally employed in sparse GP methods and provide efficient mechanisms for learning in multiple output Gaussian processes when the latent function is fluctuating rapidly.

In publication **iii**, we present applications of the variational approximation with inducing functions that include a latent force model, where the latent functions are white noise processes.

As a byproduct of seeing the linear model of coregionalization as a particular case of the convolved multioutput covariance, we can extend all the approximations to work under the linear model of coregionalization regime.

In the next chapter, we will go back to the latent force model framework and present an extension that admits the modeling of non-stationary multivariate time series by allowing changes in the parameter vector $\boldsymbol{\theta}_{\text{LFM}}$.

Chapter 5

Switching dynamical latent force models

In chapter 2, we described how a flexible type of covariance function for multivariate Gaussian processes can be developed by convolving univariate Gaussian processes with moving-average functions specific to each output. If the moving-average function corresponds to a Green's function, then the resulting covariance function encodes the dynamics associated to the differential equation from which the Green's function comes from. We coined the resulting multivariate Gaussian process with the name of latent force model in chapter 3.

In the latent force model, the parameter vector $\boldsymbol{\theta}_{\text{LFM}}$, composed of the parameters $\{\boldsymbol{\theta}_{G_d}\}_{d=1}^{D}$, $\{S_{d,q}\}_{d=1,q=1}^{D,Q}$ and $\{\ell_q\}_{q=1}^{Q}$, remains fixed for all values of the input variable t. In many applied scenarios, though, the parameter vector is a function of the input variable, $\boldsymbol{\theta}_{\text{LFM}}(t)$, either because there is a change in the parameters of the differential equation, $\{\boldsymbol{\theta}_{G_d}\}_{d=1}^{D}$, or because there is a change in the sensitivity parameters $\{S_{d,q}\}_{d=1,q=1}^{D,Q}$, or because there is a change in the length-scales of the functions $\{\ell_q\}_{q=1}^{Q}$ driving the set of systems or because all these parameters change simultaneously. In this chapter we introduce an extension of the latent force model that allows for *discrete changes* in the elements of the parameter vector $\boldsymbol{\theta}_{\text{LFM}}$.[1] Practical motivations for this extension include the discovery of *motor primitives* within a multivariate time series of angles and the augmentation of the latent force model with the ability to incorporate *discontinuous forces*. Both

[1]The current version of the software accompanying this thesis allows discrete changes in either the sensitivity parameters $\{S_{d,q}(t)\}_{d=1,q=1}^{D,Q}$ and/or in the length-scales of the forcing functions $\{\ell_q(t)\}_{q=1}^{Q}$, and assumes that the parameters $\{\boldsymbol{\theta}_{G_d}\}_{d=1}^{D}$ remain constant in the time-input domain.

applications are explained in the following paragraphs.

First, a current line of thought in *neuroscience* considers that elaborated movements or motor actions can be represented by the transformation of a reduced set of building blocks known as *motor primitives* (Flash and Hochner, 2005). Flash and Hochner (2005) present a recent review of definitions of motor primitives in the neuroscience literature that include interpretations at a behavioral-level (movements are composed of submovements, which are not further decomposed (Vecchio et al., 2003)), at a muscle level (a motor primitive is defined as a *synergy*, this is, a co-activation of muscles that produce a torque or a force), and at a neural level (a motor primitive corresponds to a particular assembly of neurons). From a more physics-related point of view, motor primitives can be either kinematic (defined as strokes or submovements), dynamic (defined as a static field of forces, time-varying synergies or control policies) or both.

The idea of motor primitive has been used in *humanoid robotics* with the purpose of creating a vocabulary of basic motor skills that can be used to teach a robot how to reproduce movements performed by a human teacher (Schaal et al., 2003). Once the library of primitives has been defined, the robot can generate more complex movements by the sequential, concurrent or hierarchical combination of the motor primitive actions (Peters, 2007). Motor primitives are learned from a sequence of multivariate time course data that corresponds to the angles of the several degrees of freedom of the robot. Before labeling the different motor primitives, it is necessary to determine where a motor primitive occurred within the multivariate time series. In other words, we need to segment the multivariate time series into a sequence of motor primitives. This step resembles the segmentation problem in a speech recognition system, in which a series of acoustic features are first segmented, before to their identification as *phonemes*.

We assume that each motor primitive can be represented through a second order differential equation, a traditional practice in imitation learning for humanoid robotics (Ijspeert et al., 2003; Schaal et al., 2007; Vecchio et al., 2003) and through the discrete changes in the parameter vector $\boldsymbol{\theta}_{\mathrm{LFM}}$ of the latent force model, where $\boldsymbol{\theta}_{G_d}$ correspond to the parameters of the Green's function associated to a second order ordinary differential equation, we identify the presence of a motor primitive.

Second, in publication **ii**, we introduced the latent force model of second order to represent the movement of a human actor while performing a particular activity.

The movements are recorded as multivariate time courses of angles referenced to the skeleton of the actor; the resulting data is known as *motion capture data*. In this context, the latent forces are seen as a reduced multivariate time series representation of the movement that summarizes the high-dimensional multivariate time series of recorded angles. Furthermore, using a few set of inferred latent forces, we can generate a diverse range of new movements. An important restriction of this model, though, is that the latent forces are continuous functions. However, discontinuities and segmented latent forces are omnipresent in real-world data. For example, impact forces due to contacts in a mechanical dynamical system (when grasping an object or when the feet touch the ground), or a switch in an electrical circuit, result in discontinuous latent forces.

With both applications in mind, the rest of the chapter is organized as follows. In section 5.1, we present the convolved multiple output covariance for the second order latent force model. In contrast to publication **ii** and chapter 3, where we ignored the initial conditions in the differential equations, in this section the initial conditions are included, and their role will be to allow for smooth transitions in the outputs between two different regimes each one described by a different latent force model. We refer to this model as the *switching dynamical latent force model*. In section 5.2, we describe the extension proposed that allows for *switching* between sequential latent force models. Section 5.3 includes an example of the application of the model for segmenting striking movements recorded using a Barrett WAM robot as haptic input device. In section 5.4 we present the related work.

Remark. This chapter is mainly based on publication **iv** and the supplementary material that accompanies it. Section 5.4 includes a description of related models that were not mentioned in publication **iv**.

5.1 Second order latent force models

In section 3.1 we introduced a latent force model of order two, as a generalization of a classical latent variable model that allowed to include behaviors like inertia and resonance, typical features in a mechanical system. The variables in the model are described as the outputs of a set of second order ordinary differential

equations driven by a set of forces $\{u_q(\mathbf{x})\}_{q=1}^{Q}$. The second order differential equations represent mass-spring-damper systems with masses described by the set of parameters $\{m_d\}_{d=1}^{D}$, springs described by parameters $\{\kappa_d\}_{d=1}^{D}$ and dampers represented by parameters $\{v_d\}_{d=1}^{D}$. The solution for the output functions was given in terms of a convolution transform and we assumed that the initial conditions were zero, in section 3.3. In this section, we present the second order latent force model with non-zero initial conditions, that will allow the extension of the model to a switching regime.

The set of D functions $\{y_d(t)\}_{d=1}^{D}$ in a second order LFM, with $Q = 1$, has elements given as

$$y_d(t) = y_d(0)c_d(t) + \dot{y}_d(0)e_d(t) + f_d(t,u), \tag{5.1}$$

where $y_d(0)$ and $\dot{y}_d(0)$ are the output and the velocity at time $t = 0$, respectively, known as the initial conditions (IC). In the above solution, we deliberately assumed that there is not an independent process $w_d(\mathbf{x})$ associated to the output. The independent process can be included later, once the switching covariance function has been constructed. The angular frequency is given by $\omega_d = \sqrt{(4m_d\kappa_d - v_d^2)/(4m_d^2)}$ and the remaining variables follow

$$c_d(t) = e^{-\alpha_d t}\Big[\cos(\omega_d t) + \frac{\alpha_d}{\omega_d}\sin(\omega_d t)\Big], \quad e_d(t) = \frac{e^{-\alpha_d t}}{\omega_d}\sin(\omega_d t),$$
$$f_d(t,u) = \frac{S_d}{m_d\omega_d}\int_0^t G_d(t-\tau)u(\tau)\mathrm{d}\tau = \frac{S_d}{m_d\omega_d}\int_0^t e^{-\alpha_d(t-\tau)}\sin[(t-\tau)\omega_d]u(\tau)\mathrm{d}\tau,$$

with $\alpha_d = v_d/(2m_d)$. The uncertainty in the model of equation (5.1) is due to the fact that the latent force $u(t)$ and the initial conditions $y_d(0)$ and $\dot{y}_d(0)$ are not known. Recall that in the LFM, we assume that the latent function $u(t)$ is sampled from a zero mean Gaussian process prior with covariance function $k_{u,u}(t,t')$.

We assume that the initial conditions, $\mathbf{y}_{\mathrm{IC}} = [y_1(0), y_2(0), \ldots, y_D(0), \dot{y}_1(0), \dot{y}_2(0), \ldots, \dot{y}_D(0)]^\top$, are independent of $u(t)$ and distributed as a zero mean Gaussian with covariance $\mathbf{K}_{\mathrm{IC}}$,

$$\mathbf{K}_{\mathrm{IC}} = \begin{bmatrix} \mathbf{K}^{\mathrm{IC}}_{\mathbf{y},\mathbf{y}} & \mathbf{K}^{\mathrm{IC}}_{\mathbf{y},\dot{\mathbf{y}}} \\ \mathbf{K}^{\mathrm{IC}}_{\dot{\mathbf{y}},\mathbf{y}} & \mathbf{K}^{\mathrm{IC}}_{\dot{\mathbf{y}},\dot{\mathbf{y}}} \end{bmatrix},$$

where the matrices $\mathbf{K}^{\text{IC}}_{\mathbf{y},\mathbf{y}}$, $\mathbf{K}^{\text{IC}}_{\mathbf{y},\dot{\mathbf{y}}}$, $\mathbf{K}^{\text{IC}}_{\dot{\mathbf{y}},\mathbf{y}}$ and $\mathbf{K}^{\text{IC}}_{\dot{\mathbf{y}},\dot{\mathbf{y}}}$ have entries $\sigma_{y_d,y_{d'}}$, $\sigma_{y_d,\dot{y}_{d'}}$, $\sigma_{\dot{y}_d,y_{d'}}$ and $\sigma_{\dot{y}_d,\dot{y}_{d'}}$, respectively, that specify the prior covariance between the elements of $\mathbf{y}_{\text{IC}}$. Then, the covariance function between any two output functions, d and d' at any two times, t and t', $k_{y_d,y_{d'}}(t,t')$, is given by

$$k_{y_d,y_{d'}}(t,t') = c_d(t)c_{d'}(t')\sigma_{y_d,y_{d'}} + c_d(t)e_{d'}(t')\sigma_{y_d,\dot{y}_{d'}} + e_d(t)c_{d'}(t')\sigma_{\dot{y}_d,y_{d'}} + e_d(t)e_{d'}(t')\sigma_{\dot{y}_d,\dot{y}_{d'}} + k_{f_d,f_{d'}}(t,t'),$$

where $k_{f_d,f_{d'}}(t,t')$ follows

$$k_{f_d,f_{d'}}(t,t') = K_0 \int_0^t G_d(t-\tau) \int_0^{t'} G_{d'}(t'-\tau')k_{u,u}(t,t')\mathrm{d}\tau'\mathrm{d}\tau, \tag{5.2}$$

with $K_0 = S_d S_{d'}/(m_d m_{d'}\omega_d\omega_{d'})$. The covariance function $k_{f_d,f_{d'}}(t,t')$ depends on the covariance function of the latent force $u(t)$. Assuming that the covariance function for the latent function follows a squared-exponential form, like in chapter 3, $k_{u,u}(t,t') = \exp[-(t-t')^2/\ell^2]$, then $k_{f_d,f_{d'}}(t,t')$ can be computed analytically. The corresponding expression appears in chapter 3, section 3.3 (see also publication **ii** and the supplementary material of publication **iv**).
In the next section we look to extend the second order LFM to the case where there can be discontinuities in the latent functions. We do this through switching between different Gaussian process models to drive the system.

5.2 Switching dynamical latent force models

We now consider switching the system between different latent forces. This allows us to change the dynamical system and the driving force for each segment. By constraining the displacement and velocity at each switching time to be the same, the output functions remain continuous.

5.2.1 Definition of the model

We assume that the input space is divided in a series of non-overlapping intervals $[t_{q-1}, t_q]_{q=1}^{Q}$. During each interval, only one force $u_{q-1}(t)$ out of Q forces is active, that is, there are $\{u_{q-1}(t)\}_{q=1}^{Q}$ forces.[2] The force $u_{q-1}(t)$ is activated after time

[2]Note that we employ the same variable Q that we used to denote the number of latent functions in the latent force model. In the switching dynamical LFM, the variable Q refers

t_{q-1} (switched on) and deactivated (switched off) after time t_q. We can use the basic model in equation (5.1) to describe the contribution to the output due to the sequential activation of these forces. A particular output $z_d(t)$ at a particular time instant t, in the interval (t_{q-1}, t_q), is expressed as

$$z_d(t) = y_d^q(t - t_{q-1}) = c_d^q(t - t_{q-1})y_d^q(t_{q-1}) + e_d^q(t - t_{q-1})\dot{y}_d^q(t_{q-1}) + f_d^q(t - t_{q-1}, u_{q-1}).$$

This equation is assumed to be valid for describing the output only inside the interval (t_{q-1}, t_q). Here we highlighted this idea by including the superscript q in $y_d^q(t - t_{q-1})$ to represent the interval q for which the equation holds, although later we will omit it to keep a simpler notation. Note that for $Q = 1$ and $t_0 = 0$, we recover the original latent force model given in equation (5.1). We also define the velocity $\dot{z}_d(t)$ at each time interval (t_{q-1}, t_q) as

$$\dot{z}_d(t) = \dot{y}_d^q(t - t_{q-1}) = g_d^q(t - t_{q-1})y_d^q(t_{q-1}) + h_d^q(t - t_{q-1})\dot{y}_d^q(t_{q-1}) + r_d^q(t - t_{q-1}, u_{q-1}),$$

where

$$g_d(t) = -e^{-\alpha_d t}\sin(\omega_d t)(\alpha_d^2\omega_d^{-1} + \omega_d), \quad h_d(t) = -e^{-\alpha_d t}\left[\frac{\alpha_d}{\omega_d}\sin(\omega_d t) - \cos(\omega_d t)\right],$$
$$r_d(t) = \frac{S_d}{m_d\omega_d}\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_0^t G_d(t-\tau)u(\tau)\mathrm{d}\tau\right).$$

Given the parameters $\{\{m_d, v_d, \kappa_d\}_{d=1}^D, \{S_{d,q-1}\}_{d=1,q=1}^{D,Q}, \{\ell_{q-1}\}_{q=1}^Q\}$, the uncertainty in the outputs is induced by the prior over the initial conditions $y_d^q(t_{q-1}), \dot{y}_d^q(t_{q-1})$ for all values of t_{q-1} and the prior over the latent force $u_{q-1}(t)$ that is active during (t_{q-1}, t_q). We place Gaussian process priors over each of these latent forces $u_{q-1}(t)$, assuming independence between them.

For initial conditions, $y_d^q(t_{q-1})$ and $\dot{y}_d^q(t_{q-1})$, we could assume that they are either parameters to be estimated or random variables with uncertainty governed by independent Gaussian distributions with covariance matrices $\mathbf{K}_{\mathrm{IC}}^q$ as described in

to forces that act sequentially and in the latent force model, the Q forces act in parallel. Of course, both type of forces can act simultaneously, but to keep the notation uncluttered, here we assume there is only one force acting in parallel and Q refers to the forces acting in series.

the last section. However, for the class of applications we have in mind: mechanical systems, the outputs should be continuous across the switching points. We therefore assume that the uncertainty about the initial conditions for the interval q, is proscribed by the Gaussian process that describes the outputs $z_d(t)$ and velocities $\dot{z}_d(t)$ in the previous interval $q-1$. In particular, we assume that $y_d^q(t_{q-1})$ and $\dot{y}_d^q(t_{q-1})$ are Gaussian-distributed with mean values given by $y_d^{q-1}(t_{q-1} - t_{q-2})$ and $\dot{y}_d^{q-1}(t_{q-1} - t_{q-2})$, and covariances $k_{z_d,z_{d'}}(t_{q-1}, t_{q'-1}) = \text{cov}[y_d^{q-1}(t_{q-1} - t_{q-2}), y_{d'}^{q-1}(t_{q-1} - t_{q-2})]$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1}, t_{q'-1}) = \text{cov}[\dot{y}_d^{q-1}(t_{q-1} - t_{q-2}), \dot{y}_{d'}^{q-1}(t_{q-1} - t_{q-2})]$. We also consider covariances between $z_d(t_{q-1})$ and $\dot{z}_{d'}(t_{q'-1})$, this is, between positions and velocities for different values of q and d.

As an example, let us assume that we have one output ($D = 1$) and three switching intervals ($Q = 3$) with switching points t_0, t_1 and t_2. At t_0, we assume that $\mathbf{y}_{\text{IC}}$ follows a Gaussian distribution with mean zero and covariance $\mathbf{K}_{\text{IC}}$. From t_0 to t_1, the output $z(t)$ is described by

$$z(t) = y^1(t - t_0) = c^1(t - t_0)y^1(t_0) + e^1(t - t_0)\dot{y}^1(t_0) + f^1(t - t_0, u_0).$$

The initial condition for the position in the interval (t_1, t_2) is given by the last equation evaluated a t_1, this is, $z(t_1) = y^2(t_1) = y^1(t_1 - t_0)$. A similar analysis is used to obtain the initial condition associated to the velocity, $\dot{z}(t_1) = \dot{y}^2(t_1) = \dot{y}^1(t_1 - t_0)$. Then, from t_1 to t_2, the output $z(t)$ is

$$\begin{aligned} z(t) = y^2(t - t_1) &= c^2(t - t_1)y^2(t_1) + e^2(t - t_1)\dot{y}^2(t_1) + f^2(t - t_1, u_1), \\ &= c^2(t - t_1)y^1(t_1 - t_0) + e^2(t - t_1)\dot{y}^1(t_1 - t_0) + f^2(t - t_1, u_1). \end{aligned}$$

Following the same train of thought, the output $z(t)$ from t_2 is given as

$$z(t) = y^3(t - t_2) = c^3(t - t_2)y^3(t_2) + e^3(t - t_2)\dot{y}^3(t_2) + f^3(t - t_2, u_2),$$

where $y^3(t_2) = y^2(t_2 - t_1)$ and $\dot{y}^3(t_2) = \dot{y}^2(t_2 - t_1)$. Figure 5.1 shows an example of the switching dynamical latent force model scenario. To ensure the continuity of the outputs, the initial conditions are forced to be equal to the output of the last interval evaluated at the switching point.

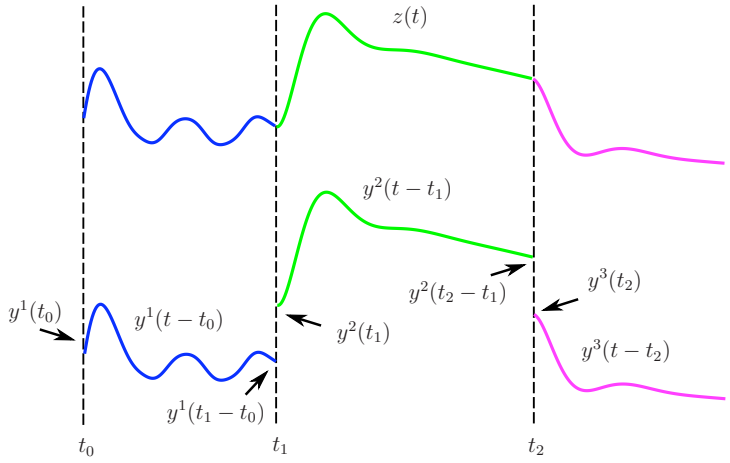
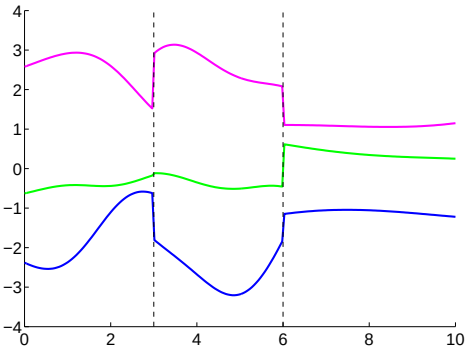
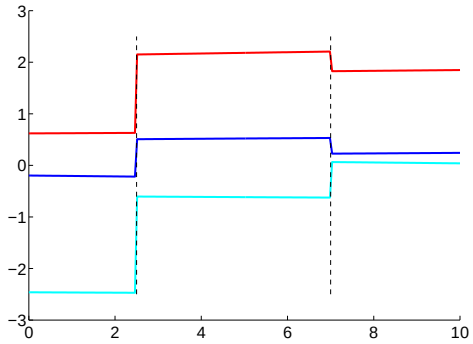


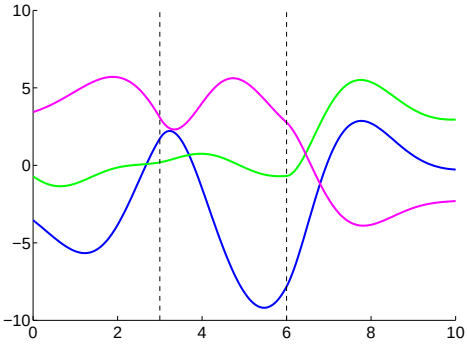
Figure 5.1: Representation of an output constructed through a switching dynamical latent force model with $Q = 3$. The initial conditions $y^q(t_{q-1})$ for each interval are matched to the value of the output in the last interval, evaluated at the switching point t_{q-1}, this is, $y^q(t_{q-1}) = y^{q-1}(t_{q-1} - t_{q-2})$.



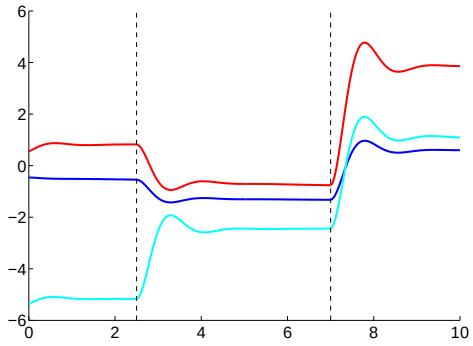
(a) System 1. Samples from the latent force.



(b) System 2. Samples from the latent force.



(c) System 1. Samples from the output.



(d) System 2. Samples from the output.

Figure 5.2: Joint samples of a switching dynamical LFM model with one output, $D = 1$, and three intervals, $Q = 3$, for two different systems. Dashed lines indicate the presence of switching points. While system 2 responds instantaneously to the input force, system 1 delays its reaction due to larger inertia.

5.2.2 The covariance function

The derivation of the covariance function for the switching model is rather involved. For continuous output signals, we must take into account constraints at each switching time. This causes initial conditions for each interval to be dependent on final conditions for the previous interval and induces correlations across the intervals. This effort is worthwhile though as the resulting model is very flexible and can take advantage of the switching dynamics to represent a range of signals.

As a taster, figure 5.2 shows samples from a covariance function of a switching dynamical latent force model with $D = 1$ and $Q = 3$. Note that while the latent forces (figures 5.2(a) and 5.2(b)) are discrete, the outputs (figures 5.2(c) and 5.2(d)) are continuous and have matching gradients at the switching points. The outputs are highly nonstationary. The switching times turn out to be parameters of the covariance function. They can be optimized along with the dynamical system parameters to match the location of the nonstationarities. We now give an overview of the covariance function derivation. Details are provided in the supplementary material, in publication **iv**.

In general, we need to compute the covariance $k_{z_d,z_{d'}}(t,t') = \text{cov}[z_d(t), z_{d'}(t')]$ for $z_d(t)$ in time interval (t_{q-1}, t_q) and $z_{d'}(t')$ in time interval $(t_{q'-1}, t_{q'})$. By definition, this covariance follows

$$\text{cov}[z_d(t), z_{d'}(t')] = \text{cov}\left[y_d^q(t - t_{q-1}), y_{d'}^{q'}(t - t_{q'-1})\right].$$

We assume independence between the latent forces $u_q(t)$ and independence between the initial conditions $\mathbf{y}_{\text{IC}}$ and the latent forces $u_q(t)$.[3] With these conditions, it can be shown[4] that the covariance function[5] for $q = q'$ is given as

$$\begin{aligned} c_d^q(t, t_{q-1}) c_{d'}^q(t', t_{q-1}) k_{z_d,z_{d'}}(t_{q-1}, t_{q-1}) + c_d^q(t, t_{q-1}) e_{d'}^q(t', t_{q-1}) k_{z_d,\dot{z}_{d'}}(t_{q-1}, t_{q-1}) \\ + e_d^q(t, t_{q-1}) c_{d'}^q(t', t_{q-1}) k_{\dot{z}_d,z_{d'}}(t_{q-1}, t_{q-1}) + e_d^q(t, t_{q-1}) e_{d'}^q(t', t_{q-1}) k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1}, t_{q-1}) \\ + k_{f_d,f_{d'}}^q(t, t'), \end{aligned} \tag{5.3}$$

[3] Derivations of these equations are rather involved. In the supplementary material of publication **iv**, section 2, we include a detailed description of how to obtain the equations (5.3) and (5.4)

[4] See supplementary material of publication **iv**, section 2.2.1.

[5] We will write $f_d^q(t - t_{q-1}, u_{q-1})$ as $f_d^q(t - t_{q-1})$, $c_d^q(t - t_{q-1})$ as $c_d^q(t, t_{q-1})$ and $e_d^q(t - t_{q-1})$ as $e_d^q(t, t_{q-1})$, for notational simplicity.

where

$$
\begin{aligned}
k_{z_d,z_{d'}}(t_{q-1},t_{q-1}) &= \mathrm{cov}[y_d^q(t_{q-1})y_{d'}^q(t_{q-1})],\\
k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1}) &= \mathrm{cov}[y_d^q(t_{q-1})\dot{y}_{d'}^q(t_{q-1})],\\
k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q-1}) &= \mathrm{cov}[\dot{y}_d^q(t_{q-1})y_{d'}^q(t_{q-1})],\\
k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1}) &= \mathrm{cov}[\dot{y}_d^q(t_{q-1})\dot{y}_{d'}^q(t_{q-1})],\\
k^q_{f_d,f_{d'}}(t,t') &= \mathrm{cov}[f_d^q(t-t_{q-1})f_{d'}^q(t'-t_{q-1})].
\end{aligned}
$$

In expression (5.3), $k_{z_d,z_{d'}}(t_{q-1},t_{q-1}) = \mathrm{cov}[y_d^{q-1}(t_{q-1}-t_{q-2}), y_{d'}^{q-1}(t_{q-1}-t_{q-2})]$. Values for $k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})$, $k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q-1})$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})$ can be obtained using similar definitions. The covariance $k^q_{f_d,f_{d'}}(t,t')$ follows an analogous expression to the one for $k_{f_d,f_{d'}}(t,t')$ in equation (5.2), now depending on the covariance $k_{u_{q-1},u_{q-1}}(t,t')$. We again assume that the covariances for the latent forces follow the squared-exponential form, with length-scale ℓ_q.

When $q > q'$, we have to take into account the correlation between the initial conditions $y_d^q(t_{q-1})$, $\dot{y}_d^q(t_{q-1})$ and the latent force $u_{q'-1}(t')$. This correlation appears because of the contribution of $u_{q'-1}(t')$ to the generation of the initial conditions. It can be shown[6] that the covariance function $\mathrm{cov}[z_d(t), z_{d'}(t')]$ for $q > q'$ follows

$$
\begin{aligned}
&c_d^q(t,t_{q-1})c_{d'}^{q'}(t',t_{q'-1})k_{z_d,z_{d'}}(t_{q-1},t_{q'-1}) + c_d^q(t,t_{q-1})e_{d'}^{q'}(t',t_{q'-1})k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1})\\
+&e_d^q(t,t_{q-1})c_{d'}^{q'}(t',t_{q'-1})k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q'-1}) + e_d^q(t,t_{q-1})e_{d'}^{q'}(t',t_{q'-1})k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1})\\
&+c_d^q(t,t_{q-1})\mathcal{X}_d^1 k^{q'}_{f_d,f_{d'}}(t_{q'-1},t') + c_d^q(t,t_{q-1})\mathcal{X}_d^2 k^{q'}_{r_d,f_{d'}}(t_{q'-1},t')\\
&+e_d^q(t,t_{q-1})\mathcal{X}_d^3 k^{q'}_{f_d,f_{d'}}(t_{q'-1},t') + e_d^q(t,t_{q-1})\mathcal{X}_d^4 k^{q'}_{r_d,f_{d'}}(t_{q'-1},t'),
\end{aligned}
\tag{5.4}
$$

where

$$
\begin{aligned}
k_{z_d,z_{d'}}(t_{q-1},t_{q'-1}) &= \mathrm{cov}[y_d^q(t_{q-1})y_{d'}^{q'}(t_{q'-1})],\\
k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1}) &= \mathrm{cov}[y_d^q(t_{q-1})\dot{y}_{d'}^{q'}(t_{q'-1})],\\
k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q'-1}) &= \mathrm{cov}[\dot{y}_d^q(t_{q-1})y_{d'}^{q'}(t_{q'-1})],\\
k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1}) &= \mathrm{cov}[\dot{y}_d^q(t_{q-1})\dot{y}_{d'}^{q'}(t_{q'-1})],\\
k^q_{r_d,f_{d'}}(t,t') &= \mathrm{cov}[r_d^q(t-t_{q-1})f_{d'}^q(t'-t_{q-1})],
\end{aligned}
$$

and $\mathcal{X}_d^1$, $\mathcal{X}_d^2$, $\mathcal{X}_d^3$ and $\mathcal{X}_d^4$ are functions of the form $\sum_{n=2}^{q-q'}\prod_{i=2}^{q-q'} x_d^{q-i+1}(t_{q-i+1}-t_{q-i})$,

[6] See supplementary material, section 2.2.2

with x_d^{q-i+1} being equal to c_d^{q-i+1}, e_d^{q-i+1}, g_d^{q-i+1} or h_d^{q-i+1}, depending on the values of q and q'.

An identical expression to (5.4) can be obtained for $q' > q$. Examples of these functions for specific values of q and q' and more details are also given in the supplementary material. We refer to this model as the switching dynamical latent force model (SDLFM).

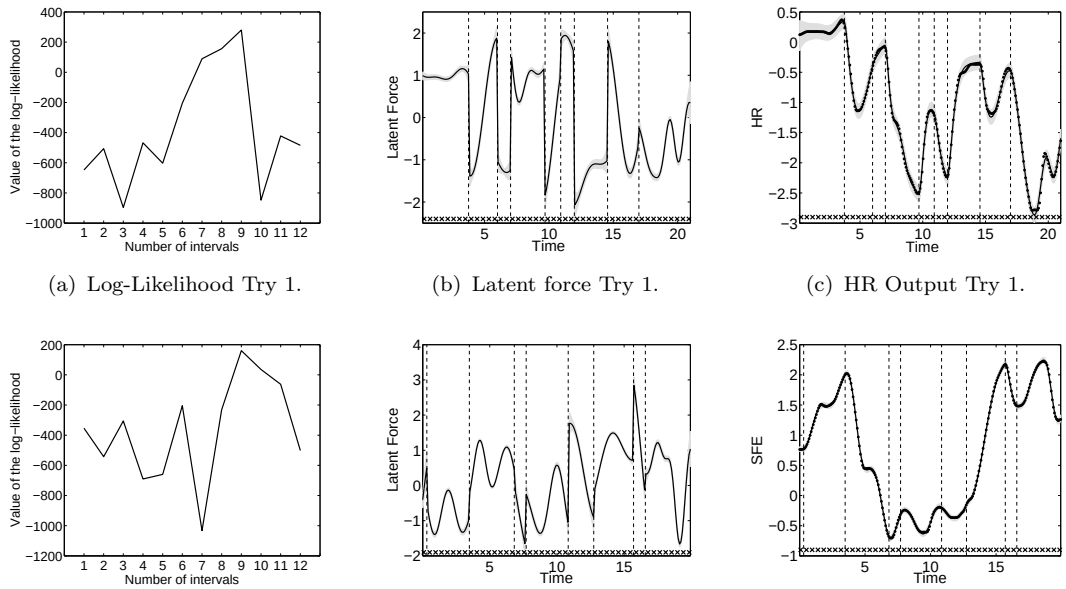
The covariance functions $k_{\dot{z}_d,z_{d'}}(t,t')$, $k_{z_d,\dot{z}_{d'}}(t,t')$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t,t')$ appearing in equations (5.3) and (5.4), are obtained by taking derivatives of $k_{z_d,z_{d'}}(t,t')$ with respect to t and t' (Solak et al., 2003).

The parameters in the SDLFM comprise the parameters of the mass-spring-damper systems, the sensitivity coefficients, the length-scales of the latent functions, the switching points and the covariance matrix $\mathbf{K}_{\text{IC}}^{q=0}$, this is $\boldsymbol{\theta}_{\text{SDLFM}} = \{\{m_d, v_d, \kappa_d\}_{d=1}^{D}, \{S_{d,q-1}\}_{d=1,q=1}^{D,Q}, \{\ell_{q-1}\}_{q=1}^{Q}, \{t_{q-1}\}_{q=1}^{Q}, \mathbf{K}_{\text{IC}}^{0}\}$. Given the number of outputs D and the number of intervals Q, we estimate the parameters $\boldsymbol{\theta}_{\text{SDLFM}}$ by maximizing the marginal likelihood of the joint Gaussian process $\{z_d(t)\}_{d=1}^{D}$ using gradient descent methods, as explained in section 2.2.1. Alternatively, we can use any of the efficient approximations of chapter 4.

5.3 Segmentation of human movement data

In this section, we evaluate the feasibility of the model for motion segmentation with possible applications in the analysis of human movement data and imitation learning. To do so, we had a human teacher take the robot by the hand and have him demonstrate striking movements in a cooperative game of table tennis with another human being as shown in figure 5.3. We recorded joint positions, angular velocities, and angular acceleration of the robot for two independent trials of the same table tennis exercise. For each trial, we selected four output positions and train several models for different values of Q, including the latent force model without switches ($Q = 1$). We evaluate the quality of the segmentation in terms of the lower bound

Figure 5.3: Data collection was performed using a Barrett WAM robot as haptic input device.



(a) Log-Likelihood Try 1. (b) Latent force Try 1. (c) HR Output Try 1.

(d) Log-Likelihood Try 2. (e) Latent force Try 2. (f) SFE Output Try 2.

Figure 5.4: Employing the switching dynamical LFM model on the human movement data collected as in figure 5.3 leads to plausible segmentations of the demonstrated trajectories. The first row corresponds to the lower bound, latent force and one of four outputs, humeral rotation (HR), for trial one. Second row shows the same quantities for trial two. In this case, the output corresponds to shoulder flexion and extension (SFE). Crosses in the bottom of the figure refer to the number of points used for the approximation of the Gaussian process, in this case $K = 50$.

for the marginal likelihood appearing in equation (4.8). For computing the entries in $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, we use the covariance describe in section 5.2.2, while the elements of covariance $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ are computed using expressions appearing in supplementary material in publication **iv**. Figure 5.4 shows the lower bound of the marginal likelihood, the inferred latent force and one output for trial one (first row) and the corresponding quantities for trial two (second row). Figures 5.4(a) and 5.4(d) show peaks for the lower bound of the marginal likelihood at $Q = 9$ for trial one and $Q = 10$ for trial two. As the movement has few gaps and the data has several output dimensions, it is hard even for a human being to detect the transitions between movements (unless it is visualized as in a movie). Nevertheless, the model found a maximum for the lower bound of the marginal likelihood at the correct instances in time where the human transits between two movements. At these instances the human usually reacts due to an external stimulus with a large jerk causing a jump in the forces. As a result, we obtained not only a segmentation of the movement but also a generative model for table tennis striking movements.

5.4 Related work

The related work can be divided in different areas, namely, alternative probabilistic methods to tackle the same segmentation problem, Gaussian processes for change-point detection and nonstationary covariance functions under a process convolution framework.

Peters (2007, chapter 2) provides a review of definitions and uses of motor primitives in classical robotics. More up-to-date approaches for segmentation of motor primitives include Williams et al. (2008) and Chiappa and Peters (2011). The idea in both methods is to represent the multivariate time series observations as noisy realizations of an underlying (usually Markov) process(es) with additional parameters that control the different durations of each primitive. In Williams et al. (2008), the probabilistic model employed is a *factorial Hidden Markov model*, while in Chiappa and Peters (2011) the probabilistic model corresponds to a *linear dynamical system* (see Bishop, 2006, for example). Chiappa et al. (2009) uses a mixture of linear dynamical systems to find similarities in a set of motion primitives that have been previously segmented. An important difference of these methodologies with the SDLFM is that those methods are parametric and driven only by data, while our approach is non-parametric and incorporates prior knowledge through the second order differential equation.

There has been a recent interest in employing Gaussian processes for detection of change points in time series analysis, an area of study that relates to some extent to our model. Some machine learning related papers include Garnett et al. (2010, 2009); Saatçi et al. (2010). Garnett et al. (2010, 2009) deals specifically with how to construct covariance functions in the presence of change points (see Garnett et al. (2010), section 4). The authors propose different alternatives according to the type of change point. From these alternatives, the closest ones to our work appear in subsections 4.2, 4.3 and 4.4. In subsection 4.2, a mechanism to keep continuity in a covariance function when there are two regimes described by different GPs, is proposed. The authors call this covariance continuous conditionally independent covariance function. In our switched latent force model, a more natural option is to use the initial conditions as the way to transit smoothly between different regimes. In subsections 4.3 and 4.4, the authors propose covariances that account for a sudden change in the input scale and a sudden change in the output scale. Both type of changes are automatically included in our model due to the latent force model construction: the changes in the input scale are

accounted by the different length-scales of the latent force GP process and the changes in the output scale are accounted by the different sensitivity parameters. Importantly, we are also concerned about multiple output systems. It is not clear how the methods in Garnett et al. (2010) could be extended to the multivariate case.
On the other hand, Saatçi et al. (2010) proposes an efficient inference procedure for Bayesian Online Change Point Detection (BOCPD) in which the underlying predictive model (UPM) is a GP. This reference is less concerned about the particular type of change that is represented by the model: in our application scenario, the continuity of the covariance function between two regimes must be assured beforehand.

In the context of single output GPs in the geostatistics literature, the process convolution construction has been used to develop nonstationary covariances for spatial domains, either by allowing the parameters of the smoothing kernel $\boldsymbol{\theta}_G$ to depend on the input location (Higdon, 1998; Higdon et al., 1998; Paciorek and Schervish, 2004) or by allowing convolutions with stationary Gaussian processes such that the covariance parameters $\boldsymbol{\psi}$ of such stationary processes depend of the input location (Fuentes, 2002a,b). A detailed description of both methods is given by Calder and Cressie (2007).

5.5 Summary

In this chapter, we introduced an extension of the latent force model that allows for discontinuous latent forces, while imposing continuity in the output functions that are modeled. The continuity in the outputs is accomplished by matching the initial conditions of positions and velocities of the current interval, with the final values of positions and velocities in the previous interval.
A different application for the SDLFM that we might consider as future work is the modeling of *walking* movements using motion capture data.

Chapter 6

Conclusions and Future Work

This final chapter summarizes the research work carried out in the thesis and outlines some ideas for future research.

Conclusions

In this thesis we have introduced a framework for developing covariance functions for multivariate Gaussian process regression. By establishing this multivariate prior, we have proposed a powerful probabilistic methodology for making simultaneous predictions providing also estimates for the uncertainty of several correlated variables. Important features of the model are the ability to incorporate prior knowledge through the specification of sensible smoothing kernels, and the non-parametric formulation in terms of kernel matrices.

In **chapter 2** we introduced the convolved multiple output covariance as the covariance obtained from convolving smoothing kernels specific to each output with covariance functions common to all outputs. Alternatives that lead to valid covariance functions for multiple outputs include the assumption of independence of the outputs, the linear model of coregionalization and process convolutions. All these alternatives can be seen as particular cases of the CMOC covariance. Publications **i**, **ii**, **iii** and **v** show experimental results that confirm that, in general, making predictions with Gaussian processes that employ the CMOC covariance, convey better performances or at least as good performances as the ones obtained by making predictions with Gaussian processes that employ alternative covariances.

In **chapter 3** we proposed the use of Green's functions associated to differential

equations as potential smoothing kernel functions in the CMOC construction. By knowing in advance the type of differential equation that might rule the behavior of the outputs, we can construct a sensible covariance function that can be applied to problem specific domains. In some real world problems, we actually have expert-knowledge about the dynamics that govern the outputs, like applications in systems biology, where interactions between genes and proteins can be modeled through first order differential equations (Barenco et al., 2006; Alon, 2006). In other scenarios, though, we just might want to include what we believe is the approximated dynamics of the system, like in motion capture data and robotics, where we assume the outputs follow second order differential equations. Notably, we have encoded dynamical systems in a multivariable covariance function that, embedded within the Gaussian process machinery, results in a generative model for the data.

In **chapter 4**, we provided different efficient approximations to make the multivariate Gaussian process regression methodology practical. In a set of approximations, we gain efficiency by modifying the marginal likelihood of the model and on a further approximation, by proposing a lower bound of the marginal likelihood. Experimental results in publications **i**, **iii** and **v** show that considerable speed-ups can be obtained by employing these approximations in the training phase as well as in the prediction stage.

Finally in **chapter 5**, we extended the CMOC to deal with nonstationary multivariate time series. Our motivation for this extension was twofold: on one hand, we wanted to provide the latent force model framework with the ability to handle discontinuous forces, an important application for generating natural-looking walking animated movements, and, on the other hand, we wanted to use the model for segmenting motor primitives as a first step towards a machine learning approach for imitation learning in robotics.

Future work

We envisage the future work following two complementary directions, the theoretical and the applied one.

Model selection. Free parameters in the CMOC include the number of latent processes Q, whose covariance functions have distinct parameter vectors $\{\boldsymbol{\psi}_q\}_{q=1}^{Q}$, and the number of latent functions R_q, which, for a fixed value of q, share the

same parameter vector $\boldsymbol{\psi}_q$. In some practical scenarios, we might know in advance how many latent functions should we use, because those functions represent physical quantities, like in a network of genes, for which the latent functions represent transcription factor proteins. However, in a black box problem, we might need to use cross-validation to assess the values of Q and R_q, which turns out to be an expensive procedure. One might think in proposing "information criteria" ideas, employing concepts like *Bayesian information criteria* or the *Akaike information criteria* (Bishop, 2006). A first attempt in that direction have been proposed by Chai et al. (2009) for the intrinsic coregionalization model. This is an area of research that requires further study.

Sparsity priors in latent force models. Having assumed a number of latent forces in the latent force model framework, one wonders if the influence of the latent forces is equal for all the outputs. The parameter that to some extent quantifies the relative strength of a force q over an output d is the sensitivity parameter $S_{d,q}$. In the way in which we estimate these parameters in the thesis, we implicitly assume that they follow a uniform prior. In many applications, though, this prior is inadequate. For example, it is well known that in a network of genes, only certain proteins interact with certain genes, this is, the network of interactions between proteins and genes is not dense. Therefore, we are interested in exploring priors over the sensitivity parameters that encourage sparsity. These priors could in principle be integrated in the variational approximation of chapter 4 and develop a practical system for performing sparse inference of gene networks involving thousands of genes and hundreds of transcription factors.

Human motion capture data. The latent force model of order two was developed in publication **ii** as a generative model for human movement. A thoughtful evaluation of this model is yet to be completed. We would like to push the boundaries of the model and test it in scenarios of transfer learning, for example, learning models for walking and running and then under certain restrictions for certain poses generate movements that look more realistic. We believe our model can provide a rich family of movements because it incorporates ideas from the dynamics of the movements. This is a hot topic of research in computer animation (Brubaker et al., 2009).

Motor primitives. Probabilistic models for description of motor primitives are attracting the attention from researches in the area of humanoid robotics. We provided an extension of the latent force model, that allows for segmenting motor primitives in a multivariate time series. Once the motor primitives are segmented, the next step is a process of labeling them so that they can be combined to generate more complex movements. A method that admits the identification of the motor primitives within the switching dynamical latent force model methodology constitutes a promising area of research.

Reprint of publication I

Mauricio A. Álvarez and Neil D. Lawrence (2008): Sparse Convolved Gaussian Processes for Multi-output Regression, in D. Koller and D. Schuurmans and Y. Bengio and L. Bottou (Eds), Advances in Neural Information Processing Systems 21, pp 57-64, 2009.

Sparse Convolved Gaussian Processes for Multi-output Regression

Mauricio Alvarez
School of Computer Science
University of Manchester, U.K.
alvarezm@cs.man.ac.uk

Neil D. Lawrence
School of Computer Science
University of Manchester, U.K.
neill@cs.man.ac.uk

Abstract

We present a sparse approximation approach for dependent output Gaussian processes (GP). Employing a latent function framework, we apply the convolution process formalism to establish dependencies between output variables, where each latent function is represented as a GP. Based on these latent functions, we establish an approximation scheme using a conditional independence assumption between the output processes, leading to an approximation of the full covariance which is determined by the locations at which the latent functions are evaluated. We show results of the proposed methodology for synthetic data and real world applications on pollution prediction and a sensor network.

1 Introduction

We consider the problem of modeling correlated outputs from a single Gaussian process (GP). Applications of modeling multiple outputs include multi-task learning (see *e.g.* [1]) and jointly predicting the concentration of different heavy metal pollutants [5]. Modelling multiple output variables is a challenge as we are required to compute cross covariances between the different outputs. In geostatistics this is known as *cokriging*. Whilst cross covariances allow us to improve our predictions of one output given the others because the correlations between outputs are modelled [6, 2, 15, 12] they also come with a computational and storage overhead. The main aim of this paper is to address these overheads in the context of convolution processes [6, 2].

One neat approach to account for non-trivial correlations between outputs employs convolution processes (CP). When using CPs each output can be expressed as the convolution between a smoothing kernel and a *latent* function [6, 2]. Let's assume that the latent function is drawn from a GP. If we also share the same latent function across several convolutions (each with a potentially different smoothing kernel) then, since a convolution is a linear operator on a function, the outputs of the convolutions can be expressed as a jointly distributed GP. It is this GP that is used to model the multi-output regression. This approach was proposed by [6, 2] who focussed on a white noise process for the latent function.

Even though the CP framework is an elegant way for constructing dependent output processes, the fact that the full covariance function of the joint GP must be considered results in significant storage and computational demands. For Q output dimensions and N data points the covariance matrix scales as QN leading to $O(Q^3N^3)$ computational complexity and $O(N^2Q^2)$ storage. Whilst other approaches to modeling multiple output regression are typically more constraining in the types of cross covariance that can be expressed [1, 15], these constraints also lead to structured covariances functions for which inference and learning are typically more efficient (typically for $N > Q$ these methods have $O(N^3Q)$ computation and $O(N^2Q)$ storage). We are interested in exploiting the richer class of covariance structures allowed by the CP framework, but without the additional computational overhead they imply.

We propose a sparse approximation for the full covariance matrix involved in the multiple output convolution process, exploiting the fact that each of the outputs is conditional independent of all others given the input process. This leads to an approximation for the covariance matrix which keeps intact the covariances of each output and approximates the cross-covariances terms with a low rank matrix. Inference and learning can then be undertaken with the same computational complexity as a set of independent GPs. The approximation turns out to be strongly related to the partially independent training conditional (PITC) [10] approximation for a single output GP. This inspires us to consider a further conditional independence function across data points that leads to an approximation which shares the form of the fully independent training conditional (FITC) approximation [13, 10] reducing computational complexity to $O(NQM^2)$ and storage to $O(NQM)$ with M representing a user specified value.

To introduce our sparse approximation some review of the CP framework is required (Section 2). Then in Section 3, we present sparse approximations for the multi-output GP. We discuss relations with other approaches in Section 4. Finally, in Section 5, we demonstrate the approach on both synthetic and real datasets.

2 Convolution Processes

Consider a set of Q functions $\{f_q(\mathbf{x})\}_{q=1}^{Q}$, where each function is expressed as the convolution between a smoothing kernel $\{k_q(\mathbf{x})\}_{q=1}^{Q}$, and a latent function $u(\mathbf{z})$,

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

More generally, we can consider the influence of more than one latent function, $\{u_r(\mathbf{z})\}_{r=1}^{R}$, and corrupt each of the outputs of the convolutions with an independent process (which could also include a noise term), $w_q(\mathbf{x})$, to obtain

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

The covariance between two different functions $y_q(\mathbf{x})$ and $y_s(\mathbf{x}')$ is then recovered as

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

where

$$\mathrm{cov}\left[f_q(\mathbf{x}), f_s(\mathbf{x}')\right] = \sum_{r=1}^{R}\sum_{p=1}^{R}\int_{-\infty}^{\infty} k_{qr}(\mathbf{x}-\mathbf{z})\int_{-\infty}^{\infty} k_{sp}(\mathbf{x}'-\mathbf{z}')\,\mathrm{cov}\left[u_r(\mathbf{z}), u_p(\mathbf{z}')\right]\mathrm{d}\mathbf{z}'\mathrm{d}\mathbf{z} \tag{2}$$

This equation is a general result; in [6, 2] the latent functions $u_r(\mathbf{z})$ are assumed as independent white Gaussian noise processes, *i.e.* $\mathrm{cov}\left[u_r(\mathbf{z}), u_p(\mathbf{z}')\right] = \sigma_{u_r}^2\delta_{rp}\delta_{\mathbf{z},\mathbf{z}'}$, so the expression (2) is simplified as

$$\mathrm{cov}\left[f_q(\mathbf{x}), f_s(\mathbf{x}')\right] = \sum_{r=1}^{R}\sigma_{u_r}^2\int_{-\infty}^{\infty} k_{qr}(\mathbf{x}-\mathbf{z})k_{sr}(\mathbf{x}'-\mathbf{z})\mathrm{d}\mathbf{z}.$$

We are going to relax this constraint on the latent processes, we assume that each inducing function is an independent GP, *i.e.* $\mathrm{cov}\left[u_r(\mathbf{z}), u_p(\mathbf{z}')\right] = k_{u_ru_p}(\mathbf{z},\mathbf{z}')\delta_{rp}$, where $k_{u_ru_r}(\mathbf{z},\mathbf{z}')$ is the covariance function for $u_r(\mathbf{z})$. With this simplification, (2) can be written as

$$\mathrm{cov}\left[f_q(\mathbf{x}), f_s(\mathbf{x}')\right] = \sum_{r=1}^{R}\int_{-\infty}^{\infty} k_{qr}(\mathbf{x}-\mathbf{z})\int_{-\infty}^{\infty} k_{sr}(\mathbf{x}'-\mathbf{z}')k_{u_ru_r}(\mathbf{z},\mathbf{z}')\mathrm{d}\mathbf{z}'\mathrm{d}\mathbf{z}. \tag{3}$$

As well as this correlation across outputs, the correlation between the latent function, $u_r(\mathbf{z})$, and any given output, $f_q(\mathbf{x})$, can be computed,

$$\mathrm{cov}\left[f_q(\mathbf{x}), u_r(\mathbf{z}))\right] = \int_{-\infty}^{\infty} k_{qr}(\mathbf{x}-\mathbf{z}')k_{u_ru_r}(\mathbf{z}',\mathbf{z})\mathrm{d}\mathbf{z}'. \tag{4}$$

3 Sparse Approximation

Given the convolution formalism, we can construct a full GP over the set of outputs. The likelihood of the model is given by

$$p(\mathbf{y}|\mathbf{X}, \boldsymbol{\phi}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}), \tag{5}$$

where $\mathbf{y} = \left[\mathbf{y}_1^\top, \ldots, \mathbf{y}_Q^\top\right]^\top$ is the set of output functions with $\mathbf{y}_q = [y_q(\mathbf{x}_1), \ldots, y_q(\mathbf{x}_N)]^\top$; $\mathbf{K}_{\mathbf{f},\mathbf{f}} \in \Re^{QN \times QN}$ is the covariance matrix relating all data points at all outputs, with elements $\mathrm{cov}\left[f_q(\mathbf{x}), f_s(\mathbf{x}')\right]$ in (3); $\boldsymbol{\Sigma} = \Sigma \otimes \mathbf{I}_N$, where Σ is a diagonal matrix with elements $\{\sigma_q^2\}_{q=1}^Q$; $\boldsymbol{\phi}$ is the set of parameters of the covariance matrix and $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ is the set of training input vectors at which the covariance is evaluated.

The predictive distribution for a new set of input vectors $\mathbf{X}_*$ is [11]

$$p(\mathbf{y}_*|\mathbf{y}, \mathbf{X}, \mathbf{X}_*, \boldsymbol{\phi}) = \mathcal{N}\left(\mathbf{K}_{\mathbf{f}_*,\mathbf{f}}(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma})^{-1}\mathbf{y}, \mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{f}}(\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma})^{-1}\mathbf{K}_{\mathbf{f},\mathbf{f}_*} + \boldsymbol{\Sigma}\right),$$

where we have used $\mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*}$ as a compact notation to indicate when the covariance matrix is evaluated at the inputs $\mathbf{X}_*$, with a similar notation for $\mathbf{K}_{\mathbf{f}_*,\mathbf{f}}$. Learning from the log-likelihood involves the computation of the inverse of $\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}$, which grows with complexity $\mathcal{O}((NQ)^3)$. Once the parameters have been learned, prediction is $\mathcal{O}(NQ)$ for the predictive mean and $\mathcal{O}((NQ)^2)$ for the predictive variance.

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, $u_r(\mathbf{z})$, then from (1) we see that each $y_q(\mathbf{x})$ *would* be independent, *i.e.* we can write,

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

where $\boldsymbol{\theta}$ are the parameters of the kernels and covariance functions. 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. The observed values of these M samples are then marginalized (as they are for the exact case) to obtain the approximation to the likelihood. Our intuition is that the approximation should be more accurate for larger M and smoother latent functions, as in this domain the latent function could be very well characterized from only a few samples.

We define $\mathbf{u} = \left[\mathbf{u}_1^\top, \ldots, \mathbf{u}_R^\top\right]^\top$ as the samples from the latent function with $\mathbf{u}_r = [u_r(\mathbf{z}_1), \ldots, u_r(\mathbf{z}_M)]^\top$; $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is then the covariance matrix between the samples from the latent functions $u_r(\mathbf{z})$, with elements given by $k_{u_r u_r}(\mathbf{z}, \mathbf{z}')$; $\mathbf{K}_{\mathbf{f},\mathbf{u}} = \mathbf{K}_{\mathbf{u},\mathbf{f}}^\top$ are the cross-covariance matrices between the latent functions $u_r(\mathbf{z})$ and the outputs $f_q(\mathbf{x})$, with elements $\mathrm{cov}\left[f_q(\mathbf{x}), u_r(\mathbf{z})\right]$ in (4) and $\mathbf{Z} = \{\mathbf{z}_1, \ldots, \mathbf{z}_M\}$ is the set of input vectors at which the covariance $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is evaluated.

We now make the conditional independence assumption given the samples from the latent functions,

$$p(\mathbf{y}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\theta}) = \prod_{q=1}^Q p(\mathbf{y}_q|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\theta}) = \prod_{q=1}^Q \mathcal{N}\left(\mathbf{K}_{\mathbf{f}_q,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f}_q,\mathbf{f}_q} - \mathbf{K}_{\mathbf{f}_q,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_q} + \sigma_q^2\mathbf{I}\right).$$

We rewrite this product as a single Gaussian with a block diagonal covariance matrix,

$$p(\mathbf{y}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{D} + \boldsymbol{\Sigma}\right) \tag{6}$$

where $\mathbf{D} = \mathrm{blockdiag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right]$, and we have used the notation $\mathrm{blockdiag}\left[\mathbf{G}\right]$ to indicate the block associated with each output of the matrix $\mathbf{G}$ should be retained, but all other elements should be set to zero. We can also write this as $\mathbf{D} = \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right] \odot \mathbf{M}$ where $\odot$ is the Hadamard product and $\mathbf{M} = \mathbf{I}_Q \otimes \mathbf{1}_N$, $\mathbf{1}_N$ being the $N \times N$ matrix of ones and $\otimes$ being the Kronecker product. We now marginalize the values of the samples from the latent functions by using their process priors, *i.e.* $p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}})$. This leads to the following marginal likelihood,

$$p(\mathbf{y}|\mathbf{Z}, \mathbf{X}, \boldsymbol{\theta}) = \int p(\mathbf{y}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\theta}) p(\mathbf{u}|\mathbf{Z}) \mathrm{d}\mathbf{u} = \mathcal{N}\left(\mathbf{0}, \mathbf{D} + \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}} + \boldsymbol{\Sigma}\right). \tag{7}$$

Notice that, compared to (5), the full covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ has been replaced by the low rank covariance $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}$ in all entries except in the diagonal blocks corresponding to $\mathbf{K}_{\mathbf{f}_q,\mathbf{f}_q}$. When using the marginal likelihood for learning, the computation load is associated to the calculation of the inverse of $\mathbf{D}$. The complexity of this inversion is $\mathcal{O}(N^3Q) + O(NQM^2)$, storage of the matrix is $O(N^2Q) + O(NQM)$. Note that if we set $M = N$ these reduce to $\mathcal{O}(N^3Q)$ and $O(N^2Q)$ respectively which matches the computational complexity of applying Q independent GPs to model the multiple outputs.

Combining eq. (6) with $p(\mathbf{u}|\mathbf{Z})$ using Bayes theorem, the posterior distribution over $\mathbf{u}$ is obtained as

$$p(\mathbf{u}|\mathbf{y},\mathbf{X},\mathbf{Z},\boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{y}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{u}}\right) \tag{8}$$

where $\mathbf{A} = \mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{u},\mathbf{f}}(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}$. The predictive distribution is expressed through the integration of (6), evaluated at $\mathbf{X}_*$, with (8), giving

$$\begin{aligned} p(\mathbf{y}_*|\mathbf{y},\mathbf{X},\mathbf{X}_*,\mathbf{Z},\boldsymbol{\theta}) &= \int p(\mathbf{y}_*|\mathbf{u},\mathbf{Z},\mathbf{X}_*,\boldsymbol{\theta})p(\mathbf{u}|\mathbf{y},\mathbf{X},\mathbf{Z},\boldsymbol{\theta})\mathrm{d}\mathbf{u} \\ &= \mathcal{N}\left(\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{y}, \mathbf{D}_* + \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_*} + \boldsymbol{\Sigma}\right) \end{aligned} \tag{9}$$

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

The functional form of (7) is almost identical to that of the PITC approximation [10], with the samples we retain from the latent function providing the same role as the *inducing values* in the partially independent training conditional (PITC) approximation. This is perhaps not surprising given that the nature of the conditional independence assumptions in PITC is similar to that we have made. A key difference is that in PITC it is not obvious which variables should be grouped together when making the conditional independence assumption, here it is clear from the structure of the model that each of the outputs should be grouped separately. However, the similarities are such that we find it convenient to follow the terminology of [10] and also refer to our approximation as a PITC approximation.

We have already noted that our sparse approximation reduces the computational complexity of multi-output regression with GPs to that of applying independent GPs to each output. For larger data sets the N^3 term in the computational complexity and the N^2 term in the storage is still likely to be prohibitive. However, we can be inspired by the analogy of our approach to the PITC approximation and consider a more radical factorization of the outputs. In the fully independent training conditional (FITC) [13, 14] a factorization across the data points is assumed. For us that would lead to the following expression for conditional distribution of the output functions given the inducing variables, $p(\mathbf{y}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \prod_{q=1}^{Q}\prod_{n=1}^{N} p(y_{qn}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta})$ which can be briefly expressed through (6) with $\mathbf{D} = \mathrm{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right] = \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right] \odot \mathbf{M}$, with $\mathbf{M} = \mathbf{I}_Q \otimes \mathbf{I}_N$. Similar equations are obtained for the posterior (8), predictive (9) and marginal likelihood distributions (7) leading to the Fully Independent Training Conditional (FITC) approximation [13, 10]. Note that the marginal likelihood might be optimized both with respect to the parameters associated with the covariance matrices and with respect to $\mathbf{Z}$. In supplementary material we include the derivatives of the marginal likelihood wrt the matrices $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$.

4 Related work

There have been several suggestions for constructing multiple output GPs [2, 15, 1]. Under the convolution process framework, the semiparametric latent factor model (SLFM) proposed in [15] corresponds to a specific choice for the smoothing kernel function in (1) namely, $k_{qr}(\mathbf{x}) = \phi_{qr}\delta(\mathbf{x})$. The latent functions are assumed to be independent GPs and in such a case, $\mathrm{cov}\left[f_q(\mathbf{x}), f_s(\mathbf{x}')\right] = \sum_r \phi_{qr}\phi_{sr}k_{u_r u_r}(\mathbf{x},\mathbf{x}')$. This can be written using matrix notation as $\mathbf{K}_{\mathbf{f},\mathbf{f}} = (\boldsymbol{\Phi}\otimes\mathbf{I})\mathbf{K}_{\mathbf{u},\mathbf{u}}(\boldsymbol{\Phi}^\top\otimes\mathbf{I})$. For computational speed up the informative vector machine (IVM) is employed [8].

In the multi-task learning model (MTLM) proposed in [1], the covariance matrix is expressed as $\mathbf{K}_{\mathbf{f},\mathbf{f}} = K^f \otimes k(\mathbf{x},\mathbf{x}')$, with K^f being constrained positive semi-definite and $k(\mathbf{x},\mathbf{x}')$ a covariance function over inputs. The Nyström approximation is applied to $k(\mathbf{x},\mathbf{x}')$. As stated in [1] with respect to SLFM, the convolution process is related with MTLM when the smoothing kernel function is

given again by $k_{qr}(\mathbf{x}) = \phi_{qr}\delta(\mathbf{x})$ and there is only one latent function with covariance $k_{uu}(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}')$. In this way, $\mathrm{cov}\left[f_q(\mathbf{x}), f_s(\mathbf{x}')\right] = \phi_q\phi_s k(\mathbf{x}, \mathbf{x}')$ and in matrix notation $\mathbf{K}_{\mathbf{f},\mathbf{f}} = \mathbf{\Phi}\mathbf{\Phi}^\top \otimes k(\mathbf{x}, \mathbf{x}')$. In [2], the latent processes correspond to white Gaussian noises and the covariance matrix is given by eq. (3). In this work, the complexity of the computational load is not discussed. Finally, [12] use a similar covariance function to the MTLM approach but use an IVM style approach to sparsification.

Note that in each of the approaches detailed above a δ function is introduced into the integral. In the dependent GP model of [2] it is introduced in the covariance function. Our approach considers the more general case when neither kernel nor covariance function is given by the δ function.

5 Results

For all our experiments we considered squared exponential covariance functions for the latent process of the form $k_{u_r u_r}(\mathbf{x}, \mathbf{x}') = \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top \mathbf{L}_r (\mathbf{x} - \mathbf{x}')\right]$, where $\mathbf{L}_r$ is a diagonal matrix which allows for different length-scales along each dimension. The smoothing kernel had the same form, $k_{qr}(\boldsymbol{\tau}) = \frac{S_{qr}|\mathbf{L}_{qr}|^{1/2}}{(2\pi)^{p/2}} \exp\left[-\frac{1}{2}\boldsymbol{\tau}^\top \mathbf{L}_{qr}\boldsymbol{\tau}\right]$, where $S_{qr} \in \mathbb{R}$ and $\mathbf{L}_{qr}$ is a symmetric positive definite matrix. For this kernel/covariance function combination the necessary integrals are tractable (see supplementary material).

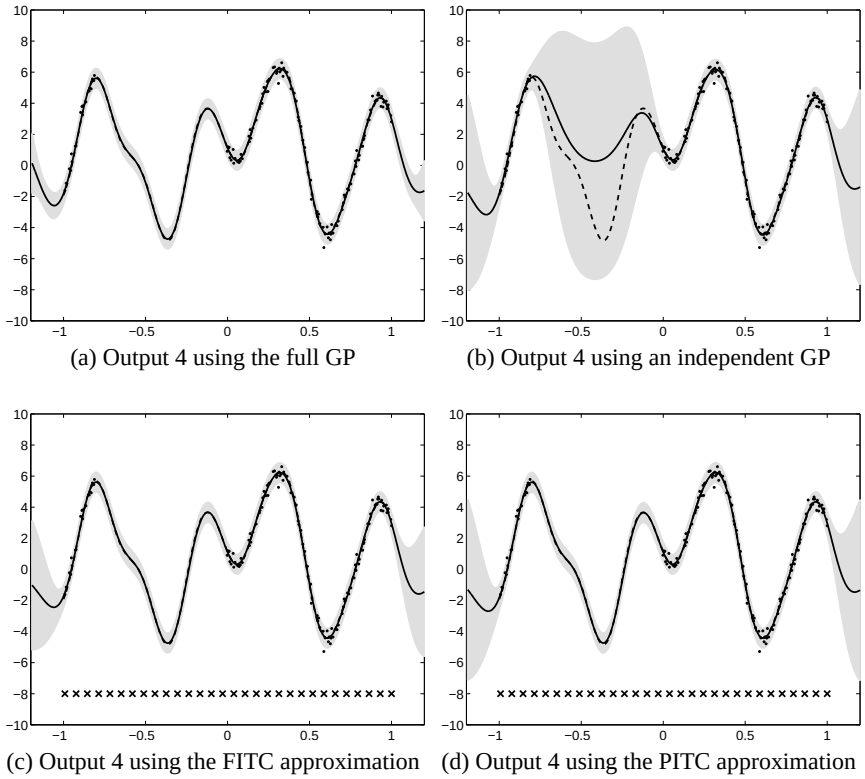
We first setup a toy problem in which we evaluate the quality of the prediction and the speed of the approximation. The toy problem consists of $Q = 4$ outputs, one latent function, $R = 1$, and $N = 200$ observation points for each output. The training data was sampled from the full GP with the following parameters, $S_{11} = S_{21} = 1$, $S_{31} = S_{41} = 5$, $L_{11} = L_{21} = 50$, $L_{31} = 300$, $L_{41} = 200$ for the outputs and $L_1 = 100$ for the latent function. For the independent processes, $w_q(\mathbf{x})$, we simply added white noise with variances $\sigma_1^2 = \sigma_2^2 = 0.0125$, $\sigma_3^2 = 1.2$ and $\sigma_4^2 = 1$. For the sparse approximations we used $M = 30$ fixed inducing points equally spaced between the range of the input and $R = 1$. We sought the kernel parameters through maximizing the marginal likelihood using a scaled conjugate gradient algorithm. For test data we removed a portion of one output as shown in Figure 1 (points in the interval $[-0.8, 0]$ were removed). The predictions shown correspond to the full GP (Figure 1(a)), an independent GP (Figure 1(b)), the FITC approximation (Figure 1(c)) and the PITC approximation (Figure 1(d)). Due to the strong dependencies between the signals, our model is able to capture the correlations and predicts accurately the missing information.

Table 1 shows prediction results over an independent test set. We used 300 points to compute the standarized mean square error (SMSE) [11] and ten repetitions of the experiment, so that we also included one standard deviation for the ten repetitions. The training times for iteration of each model are 1.45 ± 0.23 secs for the full GP, 0.29 ± 0.02 secs for the FITC and 0.48 ± 0.01 for the PITC. Table 1, shows that the SMSE of the sparse approximations is similar to the one obtained with the full GP with a considerable reduction of training times.

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 1: Standarized mean square error (SMSE) for the toy problem over an independent test set. All numbers are to be multiplied by 10^{-2}. The experiment was repeated ten times. Table included the value of one standard deviation over the ten repetitions.

We now follow a similar analysis for a dataset consisting of 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 [12]. We selected one of the sensors signals, tide height, and applied the PITC approximation scheme with an additional *squared exponential* independent kernel for each $w_q(\mathbf{x})$ [11]. Here $Q = 4$ and we chose $N = 1000$ of the 4320 for the training set, leaving the remaining points for testing. For comparison we also trained a set of independent GP models. We followed [12] in simulating sensor failure by introducing some missing ranges for these signals. In particular, we have a missing range



(a) Output 4 using the full GP (b) Output 4 using an independent GP

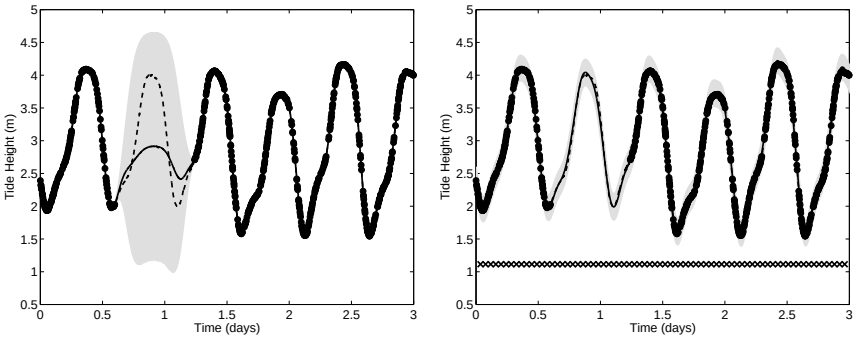
(c) Output 4 using the FITC approximation (d) Output 4 using the PITC approximation

Figure 1: Predictive mean and variance using the full multi-output GP, the sparse approximation and an independent GP for output 4. The solid line corresponds to the mean predictive, the shaded region corresponds to 2 standard deviations away from the mean and the dash line is the actual value of the signal without noise. The dots are the noisy training points. There is a range of missing data in the interval $[-0.8, 0.0]$. The crosses in figures 1(c) and 1(d) corresponds to the locations of the inducing inputs.

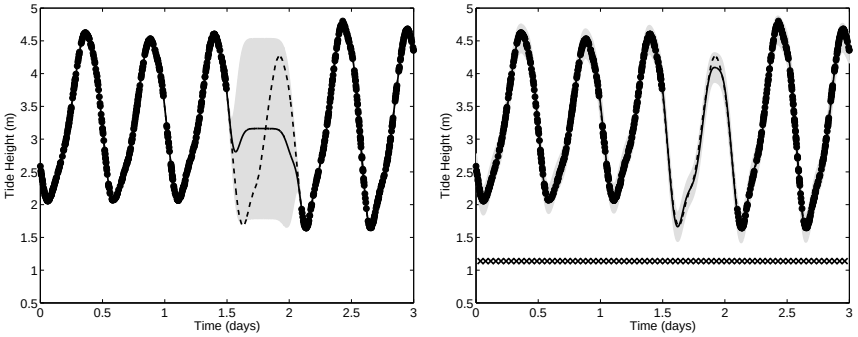
of $[0.6, 1.2]$ for the Bramblemet tide height sensor and $[1.5, 2.1]$ for the Cambermet. For the other two sensors we used all 1000 training observations. For the sparse approximation we took $M = 100$ equally spaced inducing inputs. We see from Figure 2 that the PITC approximation captures the dependencies and predicts closely the behavior of the signal in the missing range. This contrasts with the behavior of the independent model, which is not able to follow the original signal.

As another example we employ the Jura dataset, which consists of measurements of concentrations of several heavy metals collected in the topsoil of a 14.5 km^2 region of the Swiss Jura. The data is divided into a prediction set (259 locations) and a validation set (100 locations)[1]. In a typical situation, referred as *undersampled* or *heterotopic* case, a few expensive measurements of the attribute of interest are supplemented by more abundant data on correlated attributes that are cheaper to sample. We follow the experiments described in [5, p. 248,249] in which a *primary variable* (cadmium and copper) at prediction locations in conjunction with some *secondary variables* (nickel and zinc for cadmium; lead, nickel and zinc for copper) at prediction and validation locations, are employed to predict the concentration of the primary variable at validation locations. We compare results of independent GP, the PITC approximation, the full GP and ordinary co-kriging. For the PITC experiments, a *k-means* procedure is employed first to find the initial locations of the inducing values and then these locations are optimized in the same optimization procedure used for the parameters. Each experiment is repeated ten times. The results for ordinary co-kriging were obtained from [5, p. 248,249]. In this case, no values for standard deviation are reported. Figure 3 shows results of prediction for cadmium (Cd) and copper (Cu). From figure 3(a), it can be noticed that using 50 inducing values, the approximation exhibits a similar performance to the co-kriging method. As more

[1]This data is available at `http://www.ai-geostats.org/`



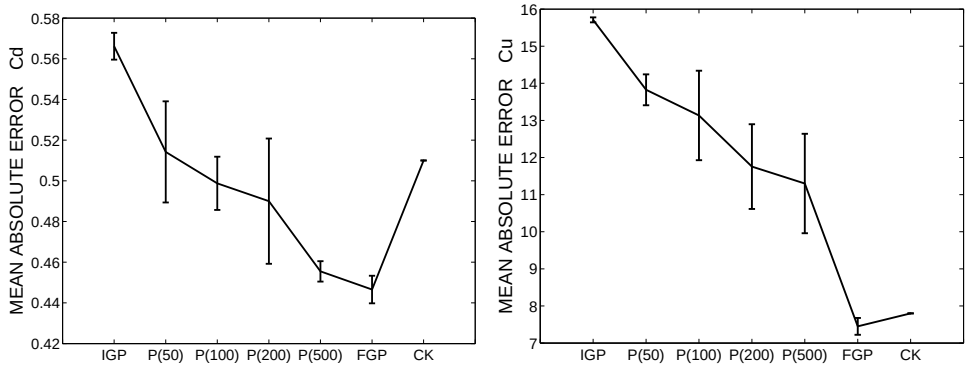
(a) Bramblemet using an independent GP (b) Bramblemet using PITC



(c) Cambermet using an independent GP (d) Cambermet using PITC

Figure 2: Predictive Mean and variance using independent GPs and the PITC approximation for the tide height signal in the sensor dataset. The dots indicate the training observations while the dash indicates the testing observations. We have emphasized the size of the training points to differentiate them from the testing points. The solid line corresponds to the mean predictive. The crosses in figures 2(b) and 2(d) corresponds to the locations of the inducing inputs.

inducing values are included, the approximation follows the performance of the full GP, as it would be expected. From figure 3(b), it can be observed that, although the approximation is better that the independent GP, it does not obtain similar results to the full GP. Summary statistics of the prediction data ([5, p. 15]) shows higher variability for the copper dataset than for the cadmium dataset, which explains in some extent the different behaviors.



(a) Cadmium (Cd) (b) Copper (Cu)

Figure 3: Mean absolute error and standard deviation for ten repetitions of the experiment for the Jura dataset In the bottom of each figure, IGP stands for independent GP, P(M) stands for PITC with M inducing values, FGP stands for full GP and CK stands for ordinary co-kriging (see [5] for detailed description).

6 Conclusions

We have presented a sparse approximation for multiple output GPs, capturing the correlated information among outputs and reducing the amount of computational load for prediction and optimization purposes. The reduction in computational complexity for the PITC approximation is from $O(N^3Q^3)$ to $O(N^3Q)$. This matches the computational complexity for modeling with independent GPs. However, as we have seen, the predictive power of independent GPs is lower.

Linear dynamical systems responses can be expressed as a convolution between the impulse response of the system with some input function. This convolution approach is an equivalent way of representing the behavior of the system through a linear differential equation. For systems involving high amounts of coupled differential equations [4], the approach presented here is a reasonable way of obtaining approximate solutions and incorporating prior domain knowledge to the model.

One could optimize with respect to positions of the values of the latent functions. As the input dimension grows, it might be more difficult to obtain an acceptable response. Some solutions to this problem have already been proposed [14].

Acknowledgments

We thank the authors of [12] who kindly made the sensor network database available.

References

[1] E. V. Bonilla, K. M. Chai, and C. K. I. Williams. Multi-task Gaussian process prediction. In J. C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *NIPS*, volume 20, Cambridge, MA, 2008. MIT Press. In press.

[2] P. Boyle and M. Frean. Dependent Gaussian processes. In L. Saul, Y. Weiss, and L. Bouttou, editors, *NIPS*, volume 17, pages 217–224, Cambridge, MA, 2005. MIT Press.

[3] M. Brookes. The matrix reference manual. Available on-line., 2005. `http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html`.

[4] P. Gao, A. Honkela, M. Rattray, and N. D. Lawrence. Gaussian process modelling of latent chemical species: Applications to inferring transcription factor activities. *Bioinformatics*, 24(16):i70–i75, 2008.

[5] P. Goovaerts. *Geostatistics For Natural Resources Evaluation*. Oxford University Press, 1997. ISBN 0-19-511538-4.

[6] D. M. Higdon. Space and space-time modelling using process convolutions. In C. Anderson, V. Barnett, P. Chatwin, and A. El-Shaarawi, editors, *Quantitative methods for current environmental issues*, pages 37–56. Springer-Verlag, 2002.

[7] N. D. Lawrence. Learning for larger datasets with the Gaussian process latent variable model. In Meila and Shen [9].

[8] N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, volume 15, pages 625–632, Cambridge, MA, 2003. MIT Press.

[9] M. Meila and X. Shen, editors. *AISTATS*, San Juan, Puerto Rico, 21-24 March 2007. Omnipress.

[10] J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *JMLR*, 6:1939–1959, 2005.

[11] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 0-262-18253-X.

[12] 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.

[13] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *NIPS*, volume 18, Cambridge, MA, 2006. MIT Press.

[14] E. Snelson and Z. Ghahramani. Local and global sparse Gaussian process approximations. In Meila and Shen [9].

[15] Y. W. Teh, M. Seeger, and M. I. Jordan. Semiparametric latent factor models. In R. G. Cowell and Z. Ghahramani, editors, *AISTATS 10*, pages 333–340, Barbados, 6-8 January 2005. Society for Artificial Intelligence and Statistics.

Reprint of publication II

Mauricio A. Álvarez, David Luengo and Neil D. Lawrence. Latent Force Models, in D. van Dyk and M. Welling (Eds.), Proceedings of The Twelfth International Conference on Artificial Intelligence and Statistics (AISTATS) 2009, JMLR: W&CP 5, pp. 9-16, Clearwater Beach, Florida, April 16-18, 2009.

Latent Force Models

Mauricio Alvarez
School of Computer Science
University of Manchester
Manchester, UK, M13 9PL
alvarezm@cs.man.ac.uk

David Luengo
Dep. Teoría de Señal y Comunicaciones
Universidad Carlos III de Madrid
28911 Leganés, Spain
luengod@ieee.org

Neil D. Lawrence
School of Computer Science
University of Manchester
Manchester, UK, M13 9PL
neill@cs.man.ac.uk

Abstract

Purely data driven approaches for machine learning present difficulties when data is scarce relative to the complexity of the model or when the model is forced to extrapolate. On the other hand, purely mechanistic approaches need to identify and specify all the interactions in the problem at hand (which may not be feasible) and still leave the issue of how to parameterize the system. In this paper, we present a hybrid approach using Gaussian processes and differential equations to combine data driven modelling with a physical model of the system. We show how different, physically-inspired, kernel functions can be developed through sensible, simple, mechanistic assumptions about the underlying system. The versatility of our approach is illustrated with three case studies from computational biology, motion capture and geostatistics.

1 Introduction

Traditionally, the main focus in machine learning has been model generation through a *data driven paradigm*. The usual approach is to combine a data set with a (typically fairly flexible) class of models and, through judicious use of regularization, make useful predictions on previously unseen data. There are two key problems with purely data driven approaches. Firstly, if data is scarce relative to the complexity of the system we may be unable to make accurate predictions on test data. Secondly, if the model is forced to extrapolate, *i.e.* make predictions in a regime in which data has not been seen yet, performance can be poor.

Purely *mechanistic models*, *i.e.* models which are inspired by the underlying physical knowledge of the system, are common in many areas such as chemistry, systems biology, climate modelling and geophysical sciences, *etc.* They normally make use of a fairly well characterized physical process that underpins the system, typically represented with a set of differential equations. The purely mechanistic approach leaves us with a different set of problems to those from the data driven approach. In particular, accurate description of a complex system through a mechanistic modelling paradigm may not be possible: even if all the physical processes can be adequately described, the resulting model could become extremely complex. Identifying and specifying all the interactions might not be feasible, and we would still be faced with the problem of identifying the parameters of the system.

Despite these problems, physically well characterized models retain a major advantage over purely data driven models. A mechanistic model can enable accurate prediction even in regions where there may be no available training data. For example, Pioneer space probes have been able to enter different extra terrestrial orbits despite the absence of data for these orbits.

In this paper we advocate an alternative approach. Rather than relying on an exclusively mechanistic or data driven approach we suggest a *hybrid system* which involves a (typically overly simplistic) mechanistic model of the system which can easily be augmented through machine learning techniques. We will start by considering two dynamical systems, both simple latent variable models, which incorporate first and second order differential equations. Our inspiration is the work of (Lawrence *et al.*, 2007; Gao *et al.*, 2008) who encoded a first order differential equation in a Gaussian process (GP). However, their aim was to construct an accurate model of transcriptional regulation, whereas ours is to make use of the mechanistic model to in-

Appearing in Proceedings of the 12^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2009, Clearwater Beach, Florida, USA. Volume 5 of JMLR: W&CP 5. Copyright 2009 by the authors.

corporate salient characteristics of the data (*e.g.* in a mechanical system *inertia*) without necessarily associating the components of our mechanistic model with actual physical components of the system. For example, for a human motion capture dataset we develop a mechanistic model of motion capture that does not exactly replicate the *physics* of human movement, but nevertheless captures salient features of the movement. Having shown how first and second order dynamical systems can be incorporated in a GP, we finally show how partial differential equations can also be incorporated for modelling systems with multiple inputs.

2 Latent Variables and Physical Systems

From the perspective of machine learning our approach can be seen as a type of latent variable model. In a latent variable model we may summarize a high dimensional data set with a reduced dimensional representation. For example, if our data consists of N points in a Q dimensional space we might seek a linear relationship between the data, $\mathbf{Y} \in \mathbb{R}^{N\times Q}$, and a reduced dimensional representation, $\mathbf{F} \in \mathbb{R}^{N\times R}$, where $R < Q$. From a probabilistic perspective this involves an assumption that we can represent the data as

$$\mathbf{Y} = \mathbf{F}\mathbf{W} + \mathbf{E}, \tag{1}$$

where $\mathbf{E}$ is a matrix-variate Gaussian noise: each column, $\boldsymbol{\epsilon}_{:,q}$ $(1 \leq q \leq Q)$, is a multi-variate Gaussian with zero mean and covariance $\boldsymbol{\Sigma}$, i.e. $\boldsymbol{\epsilon}_{:,q} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$. The usual approach, as undertaken in factor analysis and principal component analysis (PCA), to dealing with the unknowns in this model is to integrate out $\mathbf{F}$ under a Gaussian prior and optimize with respect to $\mathbf{W} \in \mathbb{R}^{R\times Q}$ (although it turns out that for a non-linear variant of the model it can be convenient to do this the other way around, see *e.g.* (Lawrence, 2005)). If the data has a temporal nature, then the Gaussian prior in the latent space could express a relationship between the rows of $\mathbf{F}$, $\mathbf{f}_{t_n} = \mathbf{f}_{t_{n-1}} + \boldsymbol{\eta}$, where $\boldsymbol{\eta} \sim \mathcal{N}(\mathbf{0}, \mathbf{C})$ and $\mathbf{f}_{t_n}$ is the n-th row of $\mathbf{F}$, which we associate with time t_n. This is known as the *Kalman filter/smoother*. Normally the times, t_n, are taken to be equally spaced, but more generally we can consider a joint distribution for $p(\mathbf{F}|\mathbf{t})$, $\mathbf{t} = [t_1 \dots t_N]^\top$, which has the form of a Gaussian process (GP),

$$p(\mathbf{F}|\mathbf{t}) = \prod_{r=1}^{R} \mathcal{N}\left(\mathbf{f}_{:,r}|\mathbf{0}, \mathbf{K}_{f_{:,r},f_{:,r}}\right),$$

where we have assumed zero mean and independence across the R dimensions of the latent space. The GP makes explicit the fact that the latent variables are functions, $\{f_r(t)\}_{r=1}^R$, and we have now described them with a process prior. The notation used, $\mathbf{f}_{:,r}$, indicates the r-th column of $\mathbf{F}$, and represents the values of that function for the r-th dimension at the times given by $\mathbf{t}$. The matrix $\mathbf{K}_{f_{:,r},f_{:,r}}$ is the covariance function associated to $f_r(t)$ computed at the times given in $\mathbf{t}$.

Such a GP can be readily implemented. Given the covariance functions for $\{f_r(t)\}$ the implied covariance functions for $\{y_q(t)\}$ are straightforward to derive. In (Teh *et al.*, 2005) this is known as a semi-parametric latent factor model (SLFM), although their main focus is not the temporal case. Historically the Kalman filter approach has been preferred, perhaps because of its linear computational complexity in N. However, recent advances in sparse approximations have made the general GP framework practical (see (Quiñonero Candela and Rasmussen, 2005) for a review).

So far the model described relies on the latent variables to provide the dynamic information. Our main contribution is to include a further dynamical system with a *mechanistic* inspiration. We now use a mechanical analogy to introduce it. Consider the following physical interpretation of (1): the latent functions, $f_r(t)$, are R forces and we observe the displacement of Q springs, $y_q(t)$, to the forces. Then we can reinterpret (1) as the force balance equation, $\mathbf{Y}\mathbf{D} = \mathbf{F}\mathbf{S} + \tilde{\mathbf{E}}$. Here we have assumed that the forces are acting, for example, through levers, so that we have a matrix of sensitivities, $\mathbf{S} \in \mathbb{R}^{R\times Q}$, and a diagonal matrix of spring constants, $\mathbf{D} \in \mathbb{R}^{Q\times Q}$. The original model is recovered by setting $\mathbf{W} = \mathbf{S}\mathbf{D}^{-1}$ and $\tilde{\boldsymbol{\epsilon}}_{:,q} \sim \mathcal{N}\left(\mathbf{0}, \mathbf{D}^\top\boldsymbol{\Sigma}\mathbf{D}\right)$. The model can be extended by assuming that the spring is acting in parallel with a damper and that the system has mass, allowing us to write,

$$\mathbf{F}\mathbf{S} = \ddot{\mathbf{Y}}\mathbf{M} + \dot{\mathbf{Y}}\mathbf{C} + \mathbf{Y}\mathbf{D} + \boldsymbol{\epsilon}, \tag{2}$$

where $\mathbf{M}$ and $\mathbf{C}$ are diagonal matrices of masses and damping coefficients respectively, $\dot{\mathbf{Y}} \in \mathbb{R}^{N\times Q}$ is the first derivative of $\mathbf{Y}$ w.r.t. time and $\ddot{\mathbf{Y}}$ is the second derivative. The second order mechanical system that this model describes will exhibit several characteristics which are impossible to represent in the simpler latent variable model given by (1), such as inertia and resonance. This model is not only appropriate for data from mechanical systems. There are many analogous systems which can also be represented by second order differential equations, e.g. Resistor-Inductor-Capacitor circuits. A unifying characteristic for all these models is that the system is beign forced by latent functions, $\{f_r(t)\}_{r=1}^R$. Hence, we refer to them as *latent force models* (LFMs).

One way of thinking of our model is to consider puppetry. A marionette is a representation of a human (or animal) controlled by a limited number of inputs through strings (or rods) attached to the character.

This limited number of inputs can lead to a wide range of character movements. In our model, the data is the movements of the marionette, and the latent forces are the inputs to the system from the puppeteer.

Finally, note that it is of little use to include dynamical models of the type specified in (2) if their effects cannot be efficiently incorporated into the inference process. Fortunately, as we will see in the case studies, for an important class of covariance functions it is *analytically* tractable to compute the implied covariance functions for $\{y_q(t)\}_{q=1}^{Q}$. Then, given the data a conjugate gradient descent algorithm can be used to obtain the hyperparameters of the model which minimize the minus log-likelihood, and inference is performed based on standard GP regression techniques.

3 First Order Dynamical System

A single input module is a biological network motif where the transcription of a number of genes is driven by a single transcription factor. In (Barenco *et al.*, 2006) a simple first order differential equation was proposed to model this situation. Then (Lawrence *et al.*, 2007; Gao *et al.*, 2008) suggested that inference of the latent transcription factor concentration should be handled using GPs. In effect their model can be seen as a latent force model based on a first order differential equation with a single latent force. Here we consider the extension of this model to multiple latent forces. As a mechanistic model, this is a severe over simplification of the physical system: transcription factors are known to interact in a non linear manner. Despite this we will be able to uncover useful information. Our model is based on the following differential equation,

$$\frac{\mathrm{d}y_q(t)}{\mathrm{d}t} + D_q y_q(t) = B_q + \sum_{r=1}^{R} S_{rq} f_r(t). \qquad (3)$$

Here the latent forces, $f_r(t)$, represent protein concentration (which is difficult to observe directly), the outputs, $y_q(t)$, are the mRNA abundance levels for different genes, B_q and D_q are respectively the basal transcription and the decay rates of the q-th gene, and S_{rq} are coupling constants that quantify the influence of the r-th input on the q-th output (*i.e.* the sensitivity of gene q to the concentration of protein r). Solving (3) for $y_q(t)$, we obtain

$$y_q(t) = \frac{B_q}{D_q} + \sum_{r=1}^{R} L_{rq}[f_r](t),$$

where we have ignored transient terms, which are easily included, and the linear operator is given by the following linear convolution operator,

$$L_{rq}[f_r](t) = S_{rq} \exp(-D_q t) \int_0^t f_r(\tau) \exp(D_q \tau) \mathrm{d}\tau.$$

If each latent force is taken to be independent with a covariance function given by

$$k_{f_r,f_r}(t,t') = \exp\left(-\frac{(t-t')^2}{\ell_r^2}\right),$$

then we can compute the covariance of the outputs analytically, obtaining (Lawrence *et al.*, 2007)

$$k_{y_p y_q}(t,t') = \sum_{r=1}^{R} \frac{S_{rp} S_{rq} \sqrt{\pi} \ell_r}{2} [h_{qp}(t',t) + h_{pq}(t,t')],$$

where

$$h_{qp}(t',t) = \frac{\exp(\nu_{rq}^2)}{D_p + D_q} \exp(-D_q t') \Bigg\{ \exp(D_q t) \times \left[\operatorname{erf}\left(\frac{t'-t}{\ell_r} - \nu_{rq}\right) + \operatorname{erf}\left(\frac{t}{\ell_r} + \nu_{rq}\right) \right] - \exp(-D_p t) \left[\operatorname{erf}\left(\frac{t'}{\ell_r} - \nu_{rq}\right) + \operatorname{erf}(\nu_{rq}) \right] \Bigg\},$$

here $\operatorname{erf}(x)$ is the real valued error function, $\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-y^2) \mathrm{d}y$, and $\nu_{rq} = \ell_r D_q / 2$.

Additionally, we can compute the cross-covariance between the inputs and outputs,

$$k_{y_q f_r}(t,t') = \frac{S_{rq} \sqrt{\pi} \ell_r}{2} \exp(\nu_{rq}^2) \exp(-D_q(t-t')) \times \left[\operatorname{erf}\left(\frac{t'-t}{\ell_r} - \nu_{rq}\right) + \operatorname{erf}\left(\frac{t'}{\ell_r} + \nu_{rq}\right) \right].$$

3.1 p53 Data

Our data is from (Barenco *et al.*, 2006), where leukemia cell lines were bombarded with radiation to induce activity of the transcription factor p53. This transcription factor repairs DNA damage and triggers a mechanism which pauses the cell-cycle and potentially terminates the cell. In (Barenco *et al.*, 2006) microarray gene expression levels of known targets of p53 were used to fit a first order differential equation model to the data. The model was then used to provide a ranked list of 50 genes identified as regulated by p53.

Our aim is to determine if there are additional "latent forces" which could better explain the activity of some of these genes. The experimental data consists of measurements of expression levels of 50 genes for three different replicas. Within each replica, there are measurements at seven different time instants. We constructed a latent force model with six latent forces, assuming that each replica was independently produced but fixing the hyperparameters of the kernel across the

replicas[1]. We employed a sparse approximation, as proposed in (Alvarez and Lawrence, 2009), with ten inducing points for speeding up computation.

Of the six latent functions, two were automatically switched off by the model. Two further latent functions, shown in Figure 1 as latent forces 1 & 2, were consistent across all replicas: their shapes were time translated versions of the p53 profile as identified by (Barenco *et al.*, 2006; Lawrence *et al.*, 2007; Gao *et al.*, 2008). This time translation allows genes to experience different *transcriptional delays*, a mechanism not included explicitly in our model, but mimicked by linear mixing of an early and a late signal. The remaining two latent functions were inconsistent across the replicas (see e.g. latent force 3 in Figure 1). They appear to represent processes not directly related to p53. This was backed up by the sensitivity parameters found in the model. The known p53 targets DDB2, p21, SESN1/hPA26, BIK and TNFRSF10b were found to respond to latent forces 1 & 2. Conversely, the genes that were most responsive to latent force 3 were MAP4K4, a gene involved in environmental stress signalling, and FDXR, an electron transfer protein.

4 Second Order Dynamical System

In Section 1 we introduced the analogy of a marionette's motion being controlled by a reduced number of forces. Human motion capture data consists of a skeleton and multivariate time courses of angles which summarize the motion. This motion can be modelled with a set of second order differential equations which, due to variations in the centers of mass induced by the movement, are non-linear. The simplification we consider for the latent force model is to linearize these differential equations, resulting in the following second order dynamical system,

$$\frac{\mathrm{d}^2 y_q(t)}{\mathrm{d}t^2} + C_q \frac{\mathrm{d}y_q(t)}{\mathrm{d}t} + D_q y_q(t) = B_q + \sum_{r=1}^{R} S_{rq} f_r(t), \quad (4)$$

where the mass of the system, without loss of generality, is normalized to 1. Whilst (4) is not the correct physical model for our system, it will still be helpful when extrapolating predictions across different motions, as we shall see in the next section. Note also that, although similar to (3), the dynamic behavior of this system is much richer than that of the first order system, since it can exhibit inertia and resonance.

[1]The decay rates were asssumed equal within replicas. Although this might be an important restriction for this experiment, our purpose in this paper is to expose a general methodology without delving into the details of each experimental setup.

For the motion capture data $y_q(t)$ corresponds to a given observed angle over time, and its derivatives represent angular velocity and acceleration. The system is summarized by the undamped natural frequency, $\omega_{0q} = \sqrt{D_q}$, and the damping ratio, $\zeta_q = \frac{1}{2} C_q / \sqrt{D_q}$. Systems with a damping ratio greater than one are said to be overdamped, whereas underdamped systems exhibit resonance and have a damping ratio less than one. For critically damped systems $\zeta_q = 1$, and finally, for undamped systems (i.e. no friction) $\zeta_q = 0$.

Ignoring the initial conditions once more, the solution of (4) is again given by a convolution, with the linear operator now being

$$\begin{aligned} L_{rq}[f_r](t) &= \frac{S_{rq}}{\omega_q} \exp(-\alpha_q t) \\ &\quad \times \int_0^t f_r(\tau) \exp(\alpha_q \tau) \sin(\omega_q (t-\tau)) \mathrm{d}\tau, \end{aligned} \quad (5)$$

where $\omega_q = \sqrt{4D_q - C_q^2}/2$ and $\alpha_q = C_q/2$.

Once again, if we consider a latent force governed by a GP with the RBF covariance function we can solve (5) analytically, obtaining a closed-form expression for the covariance matrix of the outputs,

$$k_{y_p y_q}(t,t') = \sum_{r=1}^{R} \frac{S_{rp} S_{rq} \sqrt{\pi \ell_r^2}}{8 \omega_p \omega_q} k^{(r)}_{y_p y_q}(t,t').$$

Here $k^{(r)}_{y_p y_q}(t,t')$ can be considered the cross-covariance between the p-th and q-th outputs under the effect of the r-th latent force, and is given by

$$\begin{aligned} k^{(r)}_{y_p y_q}(t,t') &= h_r(\widetilde{\gamma}_q, \gamma_p, t, t') + h_r(\gamma_p, \widetilde{\gamma}_q, t', t) \\ &\quad + h_r(\gamma_q, \widetilde{\gamma}_p, t, t') + h_r(\widetilde{\gamma}_p, \gamma_q, t', t) \\ &\quad - h_r(\widetilde{\gamma}_q, \widetilde{\gamma}_p, t, t') - h_r(\widetilde{\gamma}_p, \widetilde{\gamma}_q, t', t) \\ &\quad - h_r(\gamma_q, \gamma_p, t, t') - h_r(\gamma_p, \gamma_q, t', t), \end{aligned}$$

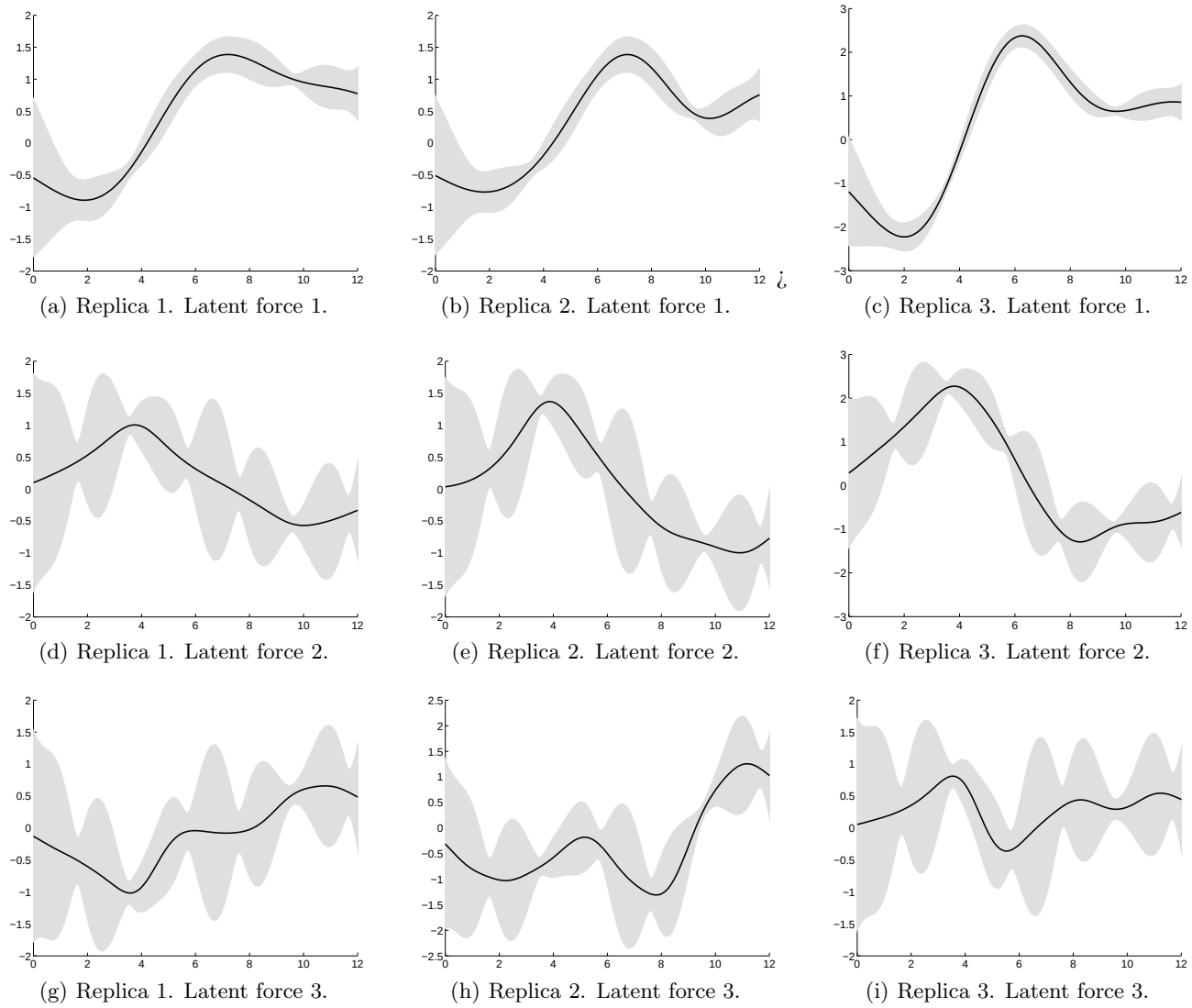
where $\gamma_p = \alpha_p + j\omega_p$, $\widetilde{\gamma}_p = \alpha_p - j\omega_p$, and

$$h_r(\gamma_q, \gamma_p, t, t') = \frac{\Upsilon_r(\gamma_q, t', t) - \exp(-\gamma_p t) \Upsilon_r(\gamma_q, t', 0)}{\gamma_p + \gamma_q},$$

with

$$\begin{aligned} \Upsilon_r(\gamma_q, t, t') &= 2 \exp\left(\tfrac{\ell_r^2 \gamma_q^2}{4}\right) \exp(-\gamma_q (t-t')) \\ &\quad - \exp\left(-\tfrac{(t-t')^2}{\ell_r^2}\right) \mathrm{w}(j z_{rq}(t)) - \exp\left(-\tfrac{(t')^2}{\ell_r^2}\right) \\ &\quad \times \exp(-\gamma_q t) \mathrm{w}(-j z_{rq}(0)), \end{aligned} \quad (6)$$

and $z_{rq}(t) = (t-t')/\ell_r - (\ell_r \gamma_q)/2$. Note that $z_{rq}(t) \in \mathbb{C}$, and $\mathrm{w}(jz)$ in (6), for $z \in \mathbb{C}$, denotes Faddeeva's function $\mathrm{w}(jz) = \exp(z^2)\mathrm{erfc}(z)$, where $\mathrm{erfc}(z)$ is the complex version of the complementary error function,



(a) Replica 1. Latent force 1. (b) Replica 2. Latent force 1. (c) Replica 3. Latent force 1.

(d) Replica 1. Latent force 2. (e) Replica 2. Latent force 2. (f) Replica 3. Latent force 2.

(g) Replica 1. Latent force 3. (h) Replica 2. Latent force 3. (i) Replica 3. Latent force 3.

Figure 1: (a)-(c) and (d)-(f) the two latent forces associated with p53 activity. p53 targets are sensitive to a combination of these functions allowing them to account for transcriptional delays. (g)-(h) a latent force that was inconsistent across the replicas. It may be associated with cellular processes not directly related to p53.

$\mathrm{erfc}(z) = 1 - \mathrm{erf}(z) = \frac{2}{\sqrt{\pi}} \int_z^\infty \exp(-v^2)\mathrm{d}v$. Faddeeva's function is usually considered the complex equivalent of the error function, since $|\mathrm{w}(jz)|$ is bounded whenever the imaginary part of jz is greater or equal than zero, and is the key to achieving a good numerical stability when computing (6) and its gradients.

Similarly, the cross-covariance between latent functions and outputs is given by

$$k_{y_q f_r}(t, t') = \frac{\ell_r S_{rq} \sqrt{\pi}}{j4\omega_q} [\Upsilon_r(\widetilde{\gamma}_q, t, t') - \Upsilon_r(\gamma_q, t, t')],$$

A visualization of a covariance matrix with a latent force and three different outputs (overdamped, underdamped and critically damped) is given in Figure 2.

4.1 Motion Capture data

Our motion capture data set is from the CMU motion capture data base[2]. We considered 3 balancing motions (18, 19, 20) from subject 49. The subject starts in a standing position with arms raised, then, over about 10 seconds, he raises one leg in the air and lowers his arms to an outstretched position. Of interest to us was the fact that, whilst motions 18 and 19 are relatively similar, motion 20 contains more dramatic movements. We were interested in training on motions 18 and 19 and testing on the more dramatic movement

[2]The CMU Graphics Lab Motion Capture Database was created with funding from NSF EIA-0196217 and is available at `http://mocap.cs.cmu.edu`.

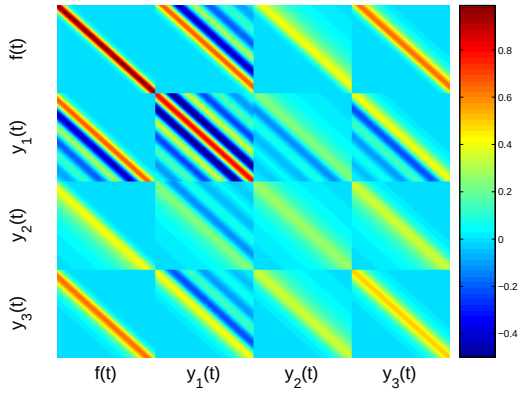


Figure 2: Visualization of the covariance matrix associated with the second order kernel. Three outputs and their correlation with the latent function are shown. Output 1 is underdamped and the natural frequency is observable through the bars of alternating correlation and anti correlation in the associated portions of the covariance matrix. Output 2 is overdamped, note the more diffuse covariance in comparison to Output 3 which is critically damped.

to assess the model's ability to extrapolate. The data was down-sampled by 32 (from 120 frames per second to just 3.75) and we focused on the subject's left arm. Our objective was to reconstruct the motion of this arm for motion 20 given the angles of the shoulder and the parameters learned from motions 18 and 19 using two latent functions. First, we train the second order differential equation latent force model on motions 18 and 19, treating the sequences as independent but sharing parameters (i.e. the damping coefficients and natural frequencies of the two differential equations associated with each angle were constrained to be the same). Then, for the test data, we condition on the observations of the shoulder's orientation to make predictions for the rest of the arm's angles.

For comparison, we considered a regression model that directly predicts the angles of the arm given the orientation of the shoulder using standard independent GPs with RBF covariance functions. Results are summarized in Table 1, with some example plots of the tracks of the angles given in Figure 3.

Table 1: Root mean squared (RMS) angle error for prediction of the left arm's configuration in the motion capture data. Prediction with the latent force model outperforms the prediction with regression for all apart from the radius's angle.

Angle	Latent Force Error	Regression Error
Radius	4.11	4.02
Wrist	6.55	6.65
Hand X rotation	1.82	3.21
Hand Z rotation	2.76	6.14
Thumb X rotation	1.77	3.10
Thumb Z rotation	2.73	6.09

5 Partial Differential Equations and Latent Forces

So far we have considered dynamical latent force models based on ordinary differential equations, leading to multioutput Gaussian processes which are functions of a single variable: time. However, the methodology can also be applied in the context of partial differential equations in order to recover multioutput Gaussian processes which are functions of several inputs.

5.1 Diffusion in the Swiss Jura

The Jura data is a set of measurements of concentrations of several heavy metal pollutants collected from topsoil in a 14.5 km^2 region of the Swiss Jura. We consider a latent function that represents how the pollutants were originally laid down. As time passes, we assume that the pollutants diffuse at different rates resulting in the concentrations observed in the data set. We therefore consider a simplified version of the diffusion equation, known also as the heat equation,

$$\frac{\partial y_q(\mathbf{x},t)}{\partial t} = \sum_{j=1}^{d} \kappa_q \frac{\partial^2 y_q(\mathbf{x},t)}{\partial x_j^2},$$

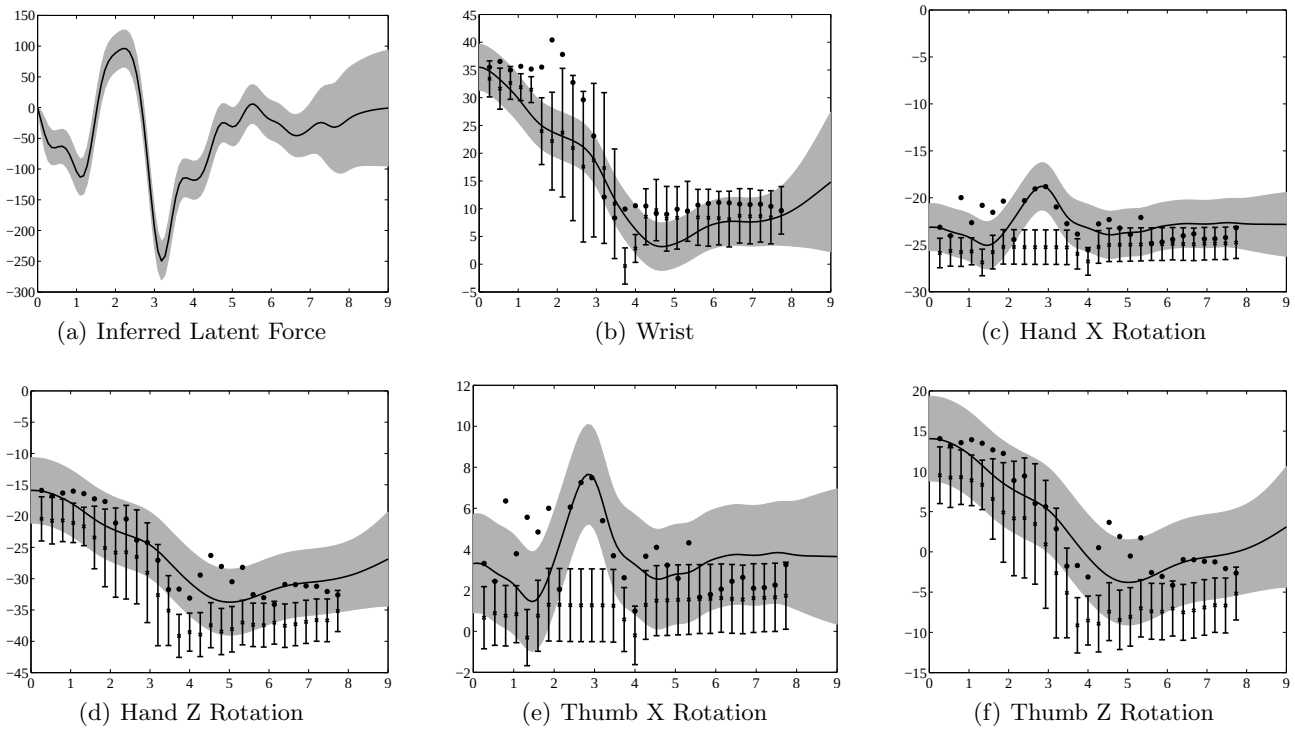
where $d = 2$ is the dimension of $\mathbf{x}$, the measured concentration of each pollutant over space and time is given by $y_q(\mathbf{x},t)$, and the latent function $f_r(\mathbf{x})$ now represents the concentration of pollutants at time zero (i.e. the system's initial condition). The solution to the system (Polyanin, 2002) is then given by

$$y_q(\mathbf{x},t) = \sum_{r=1}^{R} S_{rq} \int_{\mathbb{R}^d} f_r(\mathbf{x}') G_q(\mathbf{x},\mathbf{x}',t) \mathrm{d}\mathbf{x}'$$

where $G_q(\mathbf{x},\mathbf{x}',t)$ is the Green's function given as

$$G_q(\mathbf{x},\mathbf{x}',t) = \frac{1}{2^d \pi^{d/2} T_q^{d/2}} \exp\left[-\sum_{j=1}^{d} \frac{(x_j - x_j')^2}{4T_q}\right],$$

with $T_q = \kappa_q t$. Again, if we take the latent function to be given by a GP with the RBF covariance function we can compute the multiple output covariance functions analytically. The covariance function between



(a) Inferred Latent Force (b) Wrist (c) Hand X Rotation

(d) Hand Z Rotation (e) Thumb X Rotation (f) Thumb Z Rotation

Figure 3: (a) Inferred latent force for the motion capture data. The force shown is the weighted sum of the two forces that drive the system. (b)-(f) Predictions from the latent force model (solid line, grey error bars) and from direct regression from the shoulder angles (crosses with stick error bars). For these examples noise is high due to the relatively small length of the bones. Despite this the latent force model does a credible job of capturing the angle, whereas direct regression with independent GPs fails to capture the trends.

the output functions is obtained as

$$k_{y_p y_q}(\mathbf{x}, \mathbf{x}', t) = \sum_{r=1}^{R} \frac{S_{rp} S_{rq} |\mathbf{L}_r|^{1/2}}{|\mathbf{L}_{rp} + \mathbf{L}_{rq} + \mathbf{L}_r|^{1/2}} \times \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top (\mathbf{L}_{rp} + \mathbf{L}_{rq} + \mathbf{L}_r)^{-1} (\mathbf{x} - \mathbf{x}')\right],$$

where $\mathbf{L}_{rp}, \mathbf{L}_{rq}$ and $\mathbf{L}_r$ are diagonal isotropic matrices with entries $2\kappa_p t$, $2\kappa_q t$ and $1/\ell_r^2$ respectively. The covariance function between the output and latent functions is given by

$$k_{y_q f_r}(\mathbf{x}, \mathbf{x}', t) = \frac{S_{rq} |\mathbf{L}_r|^{1/2}}{|\mathbf{L}_{rq} + \mathbf{L}_r|^{1/2}} \times \exp\left[-\frac{1}{2}(\mathbf{x} - \mathbf{x}')^\top (\mathbf{L}_{rq} + \mathbf{L}_r)^{-1} (\mathbf{x} - \mathbf{x}')\right].$$

5.2 Prediction of Metal Concentrations

We used our model to replicate the experiments described in (Goovaerts, 1997, pp. 248,249) in which a *primary variable* (cadmium, copper, lead and cobalt) is predicted in conjunction with some *secondary variables* (nickel and zinc for cadmium; lead, nickel and zinc for copper; copper, nickel and zinc for lead; nickel and zinc for cobalt).[3] By conditioning on the values of the secondary variables we can improve the prediction of the primary variables. We compare results for the diffusion kernel with results from prediction using independent GPs for the metals and "ordinary co-kriging" (as reported by (Goovaerts, 1997, pp. 248,249)). For our experiments we made use of 10 repeats to report standard deviations. Mean absolute errors and standard deviations are shown in Table 2 ((Goovaerts, 1997) does not report standard deviations for the co-kriging method). Our diffusion model outperforms co-kriging for all but one example.

6 Discussion

We have proposed a hybrid approach for the use of simple mechanistic models with Gaussian processes which allows for the creation of new kernels with physically meaningful parameters. We have shown how these kernels can be applied to a range of data sets for the analysis of microarray data, motion capture data and

[3] Data available at `http://www.ai-geostats.org/`.

Table 2: Mean absolute error and standard deviation for ten repetitions of the experiment for the Jura dataset. IGPs stands for independent GPs, GPDK stands for GP diffusion kernel, OCK for ordinary cokriging, Cd for Cadmium, Cu for Copper, Pb for lead and Co for Cobalt. For the Gaussian process with diffusion kernel, we learn the diffusion coefficients and the length-scale of the covariance of the latent function.

Metals	IGPs	GPDK	OCK
Cd	0.5823±0.0133	0.4505±0.0126	0.5
Cu	15.9357±0.0907	7.1677±0.2266	7.8
Pb	22.9141±0.6076	10.1097±0.2842	10.7
Co	2.0735±0.1070	1.7546±0.0895	1.5

geostatistical data. To do this we proposed a range of linear differential equation models: first order, second order and a partial differential equation. The solutions to all these differential equations are in the form of convolutions. When applied to a Gaussian process latent function they result in a joint GP over the latent functions and the observed outputs which provides a general framework for multi-output GP regression.

We are not the first to suggest the use of convolution processes for multi-output regression, they were proposed by (Higdon, 2002) and built on by (Boyle and Frean, 2005) — the ideas in these papers have also recently been made more computationally practical through sparse approximations suggested by (Alvarez and Lawrence, 2009). However, whilst (Boyle and Frean, 2005) was motivated by the general idea of constructing multi-output GPs, our aims are different. Our focus has been embodying GPs with the characteristics of mechanistic models so that our data driven models can exhibit well understood characteristics of these physical systems. To maintain tractability these mechanistic models are necessarily over simplistic, but our results have shown that they can lead to significant improvements on a wide range of data sets.

Acknowledgements

DL has been partly financed by Comunidad de Madrid (project PRO-MULTIDIS-CM, S-0505/TIC/0233), and by the Spanish government (CICYT project TEC2006-13514-C02-01 and researh grant JC2008-00219). MA and NL have been financed by a Google Research Award and EPSRC Grant No EP/F005687/1 "Gaussian Processes for Systems Identification with Applications in Systems Biology".

References

Mauricio Alvarez and Neil D. Lawrence. Sparse convolved gaussian processes for multi-output regression. In *Advances in Neural Information Processing Systems 21*, pages 57–64. MIT Press, 2009.

Martino Barenco, Daniela Tomescu, Daniel Brewer, Robin Callard, Jaroslav Stark, and Michael Hubank. Ranked prediction of p53 targets using hidden variable dynamic modeling. *Genome Biology*, 7(3):R25, 2006.

Phillip Boyle and Marcus Frean. Dependent Gaussian processes. In Lawrence Saul, Yair Weiss, and Léon Bouttou, editors, *Advances in Neural Information Processing Systems*, volume 17, pages 217–224, Cambridge, MA, 2005. MIT Press.

Pei Gao, Antti Honkela, Magnus Rattray, and Neil D. Lawrence. Gaussian process modelling of latent chemical species: Applications to inferring transcription factor activities. *Bioinformatics*, 24:i70–i75, 2008.

Pierre Goovaerts. *Geostatistics For Natural Resources Evaluation*. Oxford University Press, 1997.

David M. Higdon. Space and space-time modelling using process convolutions. In C. Anderson, V. Barnett, P. Chatwin, and A. El-Shaarawi, editors, *Quantitative methods for current environmental issues*, pages 37–56. Springer-Verlag, 2002.

Neil D. Lawrence, Guido Sanguinetti, and Magnus Rattray. Modelling transcriptional regulation using Gaussian processes. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems*, volume 19, pages 785–792, Cambridge, MA, 2007. MIT Press.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, Nov. 2005.

Andrei D. Polyanin. *Handbook of Linear Partial Differential Equations for Engineers and Scientists*. Chapman & Hall/CRC Press, 2002.

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.

Yee Whye Teh, Matthias Seeger, and Michael I. Jordan. Semiparametric latent factor models. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 333–340, Barbados, 6-8 January 2005. Society for Artificial Intelligence and Statistics.

Reprint of publication III

Mauricio A. Álvarez, David Luengo, Michalis K. Titsias and Neil D. Lawrence (2010): Efficient Multioutput Gaussian Processes through Variational Inducing Kernels, in Y. Whye Teh and M. Titterington (Eds.), Proceedings of The Thirteenth International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, JMLR: W&CP 9, pp. 25-32, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010.

Efficient Multioutput Gaussian Processes through Variational Inducing Kernels

Mauricio A. Álvarez
School of Computer Science
University of Manchester
Manchester, UK, M13 9PL
alvarezm@cs.man.ac.uk

David Luengo
Depto. Teoría de Señal y Comunicaciones
Universidad Carlos III de Madrid
28911 Leganés, Spain
luengod@ieee.org

Michalis K. Titsias, Neil D. Lawrence
School of Computer Science
University of Manchester
Manchester, UK, M13 9PL
{mtitsias,neill}@cs.man.ac.uk

Abstract

Interest in multioutput kernel methods is increasing, whether under the guise of multitask learning, multisensor networks or structured output data. From the Gaussian process perspective a multioutput Mercer kernel is a covariance function over correlated output functions. One way of constructing such kernels is based on convolution processes (CP). A key problem for this approach is efficient inference. Álvarez and Lawrence recently presented a sparse approximation for CPs that enabled efficient inference. In this paper, we extend this work in two directions: we introduce the concept of variational inducing functions to handle potential non-smooth functions involved in the kernel CP construction and we consider an alternative approach to approximate inference based on variational methods, extending the work by Titsias (2009) to the multiple output case. We demonstrate our approaches on prediction of school marks, compiler performance and financial time series.

1 Introduction

In this paper we are interested in developing priors over multiple functions in a Gaussian processes (GP) framework. While such priors can be trivially specified by considering the functions to be independent, our focus is on priors which specify correlations between the functions. Most attempts to apply such priors (Teh *et al.*, 2005; Rogers *et al.*, 2008; Bonilla *et al.*, 2008) have focused on what is known in the geostatistics community as "linear model of coregionalization" (LMC) (Goovaerts, 1997). In these models the different outputs are assumed to be linear combinations of a set of one or more "latent functions". GP priors are placed, independently, over each of the latent functions inducing a correlated covariance function over the D outputs $\{f_d(\mathbf{x})\}_{d=1}^{D}$.

We wish to go beyond the LMC framework, in particular, our focus is on *convolution processes* (CPs). Using CPs for multi-output GPs was proposed by Higdon (2002) and introduced to the machine learning audience by Boyle and Frean (2005). Convolution processes allow the integration of prior information from physical models, such as ordinary differential equations, into the covariance function. Álvarez *et al.* (2009a), inspired by Lawrence *et al.* (2007), have demonstrated how first and second order differential equations, as well as partial differential equations, can be accommodated in a covariance function. They interpret the set of latent functions as a set of *latent forces*, and they term the resulting models "latent force models" (LFM). The covariance functions for these models are derived through convolution processes. In the CP framework, output functions are generated by convolving R independent latent processes $\{u_r\}_{r=1}^{R}$ with smoothing kernel functions $G_{d,r}(\mathbf{x})$, for each output d and latent force r,

$$f_d(\mathbf{x}) = \sum_{r=1}^{R} \int_{\mathcal{Z}} G_{d,r}(\mathbf{x}-\mathbf{z})\, u_r(\mathbf{z})\, \mathrm{d}\mathbf{z}. \qquad (1)$$

The LMC can be seen as a particular case of the CP, in which the kernel functions $G_{d,r}(\mathbf{x})$ correspond to a scaled Dirac δ-function $G_{d,r}(\mathbf{x}-\mathbf{z}) = a_{d,r}\delta(\mathbf{x}-\mathbf{z})$.

A practical problem associated with the CP framework is that in these models inference has computational complexity $O(N^3D^3)$ and storage requirements $O(N^2D^2)$. Recently Álvarez and Lawrence (2009) introduced an efficient approximation for inference in this multi-output GP model. Their idea was to exploit a conditional independence assumption over the output functions $\{f_d(\mathbf{x})\}_{d=1}^{D}$: if the latent functions are fully observed then the output functions *are* conditionally independent of one another (as can be seen in (1)). Furthermore, if the latent processes are sufficiently smooth, the conditional independence assumption

Appearing in Proceedings of the 13^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, Chia Laguna Resort, Sardinia, Italy. Volume 9 of JMLR: W&CP 9. Copyright 2010 by the authors.

will hold approximately even for a finite number of observations of the latent functions $\left\{\{u_r(\mathbf{z}_k)\}_{k=1}^K\right\}_{r=1}^R$, where the variables $\{\mathbf{z}_k\}_{k=1}^K$ are usually referred to as the inducing inputs. These assumptions led to approximations that were very similar in spirit to the PITC and FITC approximations of Snelson and Ghahramani (2006); Quiñonero Candela and Rasmussen (2005).

In this paper we build on the work of Álvarez and Lawrence and extend it in two ways. First, we notice that if the locations of the inducing points are close relative to the length scale of the latent function, the PITC approximation will be accurate enough. However, if the length scale becomes small the approximation requires very many inducing points. In the worst case, the latent process could be white noise (as suggested by Higdon (2002) and implemented by Boyle and Frean (2005)). In this case the approximation will fail completely. To deal with such type of latent functions, we develop the concept of an *inducing function*, a generalization of the traditional concept of *inducing variable* commonly employed in several sparse GP methods. As we shall see, an inducing function is an artificial construction generated from a convolution operation between a smoothing kernel or *inducing kernel* and the latent functions u_r. The artificial nature of the inducing function is based on the fact that its construction is immersed in a variational-like inference procedure that does not modify the marginal likelihood of the true model. This leads us to the second extension of the paper: a problem with the FITC and PITC approximations can be their tendency to overfit when inducing inputs are optimized. A solution to this problem was given in a recent work by Titsias (2009) who provided a sparse GP approximation that has an associated variational bound. In this paper we show how the ideas of Titsias can be extended to the multiple output case. Our variational approximation is developed through the inducing functions and the quality of the approximation can be controlled through the inducing kernels and the number and location of the inducing inputs. Our approximation allows us to consider latent force models with a large number of states, D, and data points N. The use of inducing kernels also allows us to extend the inducing variable approximation of the latent force model framework to systems of *stochastic differential equations* (SDEs). We apply the approximation to different real world datasets, including a multivariate financial time series example.

A similar idea to the inducing function one introduced in this paper, was simultaneously proposed by Lázaro-Gredilla and Figueiras-Vidal (2010). Lázaro-Gredilla and Figueiras-Vidal (2010) introduced the concept of inducing feature to improve performance over the pseudo-inputs approach of Snelson and Ghahramani (2006) in sparse GP models. Our use of inducing functions and inducing kernels is motivated by the necessity to deal with non-smooth latent functions in the CP model of multiple outputs.

2 Multioutput GPs (MOGPs)

Let $\mathbf{y}_d \in \mathbb{R}^N$, where $d = 1, \ldots, D$, be the observed data associated with the output function $y_d(\mathbf{x})$. For simplicity, we assume that all the observations associated with different outputs are evaluated at the same inputs $\mathbf{X}$ (although this assumption is easily relaxed). We will often use the stacked vector $\mathbf{y} = (\mathbf{y}_1, \ldots, \mathbf{y}_D)$ to collectively denote the data of all the outputs. Each observed vector $\mathbf{y}_d$ is assumed to be obtained by adding independent Gaussian noise to a vector of function values $\mathbf{f}_d$ so that the likelihood is $p(\mathbf{y}_d|\mathbf{f}_d) = \mathcal{N}(\mathbf{y}_d|\mathbf{f}_d, \sigma_d^2 I)$, where $\mathbf{f}_d$ is defined via (1). More precisely, the assumption in (1) is that a function value $f_d(\mathbf{x})$ (the noise-free version of $y_d(\mathbf{x})$) is generated from a common pool of R independent latent functions $\{u_r(\mathbf{x})\}_{r=1}^R$, each having a covariance function (Mercer kernel) given by $k_r(\mathbf{x}, \mathbf{x}')$. Notice that the outputs share the same latent functions, but they also have their own set of parameters $(\{\boldsymbol{\alpha}_{dr}\}_{r=1}^R, \sigma_d^2)$ where $\boldsymbol{\alpha}_{dr}$ are the parameters of the smoothing kernel $G_{d,r}(\cdot)$. Because convolution is a linear operation, the covariance between any pair of function values $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x}')$ is given by $k_{f_d,f_{d'}}(\mathbf{x}, \mathbf{x}') = \text{Cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')] = \sum_{r=1}^R \int_{\mathcal{Z}} G_{d,r}(\mathbf{x} - \mathbf{z}) \int_{\mathcal{Z}} G_{d',r}(\mathbf{x}' - \mathbf{z}') k_r(\mathbf{z}, \mathbf{z}') \mathrm{d}\mathbf{z}\mathrm{d}\mathbf{z}'$. This covariance function is used to define a fully-coupled GP prior $p(\mathbf{f}_1, \ldots, \mathbf{f}_D)$ over all the function values associated with the different outputs. The joint probability distribution of the multioutput GP model can be written as $p(\{\mathbf{y}_d, \mathbf{f}_d\}_{d=1}^D) = \prod_{d=1}^D p(\mathbf{y}_d|\mathbf{f}_d) p(\mathbf{f}_1, \ldots, \mathbf{f}_D)$. The GP prior $p(\mathbf{f}_1, \ldots, \mathbf{f}_D)$ has a zero mean vector and a $(ND) \times (ND)$ covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, where $\mathbf{f} = (\mathbf{f}_1, \ldots, \mathbf{f}_D)$, which consists of $N \times N$ blocks of the form $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$. Elements of each block are given by $k_{f_d,f_{d'}}(\mathbf{x}, \mathbf{x}')$ for all possible values of $\mathbf{x}$. Each such block is a cross-covariance (or covariance) matrix of pairs of outputs.

Prediction using the above GP model, as well as the maximization of the marginal likelihood $p(\mathbf{y}) = N(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2\mathbf{I}, \ldots, \sigma_D^2\mathbf{I})$, requires $O(N^3D^3)$ time and $O(N^2D^2)$ storage which rapidly becomes infeasible even when only a few hundred outputs and data points are considered. Efficient approximations are needed in order to make the above multioutput GP model more practical.

3 PITC-like approximation for MOGPs

Before we propose our variational sparse inference method for multioutput GP regression in Section 4, we review the sparse method proposed by Álvarez and Lawrence (2009). This method is based on a likelihood approximation. More precisely, each output function $y_d(\mathbf{x})$ is independent from the other output functions given the full-length of each latent function $u_r(\mathbf{x})$. This means, that the likelihood of the data factorizes according to $p(\mathbf{y}|u) =$

$\prod_{d=1}^{D} p(\mathbf{y}_d|u) = \prod_{d=1}^{D} p(\mathbf{y}_d|\mathbf{f}_d)$, with $u = \{u_r\}_{r=1}^{R}$ the set of latent functions. The sparse method in Álvarez and Lawrence (2009) makes use of this factorization by assuming that it remains valid even when we are only allowed to exploit the information provided by a finite set of function values, $\mathbf{u}_r$, instead of the full-length function $u_r(\mathbf{x})$ (which involves uncountably many points). Let $\mathbf{u}_r$, for $r = 1, \ldots, R$, be a K-dimensional vector of values from the function $u_r(\mathbf{x})$ which are evaluated at the inputs $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^{K}$. The vector $\mathbf{u} = (\mathbf{u}_1, \ldots, \mathbf{u}_R)$ denotes all these variables. The sparse method approximates the exact likelihood function $p(\mathbf{y}|u)$ with the likelihood $p(\mathbf{y}|\mathbf{u}) = \prod_{d=1}^{D} p(\mathbf{y}_d|\mathbf{u}) = \prod_{d=1}^{D} \mathcal{N}(\mathbf{y}_d|\boldsymbol{\mu}_{\mathbf{f}_d|\mathbf{u}}, \boldsymbol{\Sigma}_{\mathbf{f}_d|\mathbf{u}} + \sigma_d^2 I)$, where $\boldsymbol{\mu}_{\mathbf{f}_d|\mathbf{u}} = \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}$ and $\boldsymbol{\Sigma}_{\mathbf{f}_d|\mathbf{u}} = \mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_d}$ are the mean and covariance matrices of the conditional GP priors $p(\mathbf{f}_d|\mathbf{u})$. The matrix $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is a block diagonal covariance matrix where the rth block $\mathbf{K}_{\mathbf{u}_r,\mathbf{u}_r}$ is obtained by evaluating $k_r(\mathbf{z}, \mathbf{z}')$ at the inducing inputs $\mathbf{Z}$. Further, the matrix $\mathbf{K}_{\mathbf{f}_d,\mathbf{u}}$ is defined by the cross-covariance function $\text{Cov}[f_d(\mathbf{x}), u_r(\mathbf{z})] = \int_{\mathcal{Z}} G_{d,r}(\mathbf{x} - \mathbf{z}')k_r(\mathbf{z}', \mathbf{z})d\mathbf{z}'$. The variables $\mathbf{u}$ follow the GP prior $p(\mathbf{u}) = N(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}})$ and can be integrated out to give the following approximation to the exact marginal likelihood:

$$p(\mathbf{y}|\boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}|\mathbf{0}, \mathbf{D} + \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}} + \boldsymbol{\Sigma}). \quad (2)$$

Here, $\mathbf{D}$ is a block-diagonal matrix, where each block is given by $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_d}$ for all d. This approximate marginal likelihood represents exactly each diagonal (output-specific) block $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d}$ while each off diagonal (cross-output) block $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}}$ is approximated by the Nyström matrix $\mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_{d'}}$.

The above sparse method has a similar structure to the PITC approximation introduced for single-output regression (Quiñonero Candela and Rasmussen, 2005). Because of this similarity, Álvarez and Lawrence (2009) call their multioutput sparse approximation PITC as well. Two of the properties of this PITC approximation (which may sometimes be seen as limitations) are:

1. It assumes that all latent functions u are smooth.
2. It is based on a modification of the initial full GP model. This implies that the inducing inputs $\mathbf{Z}$ are extra kernel hyparameters in the modified GP model.

Because of point 1, the method is not applicable when the latent functions are white noise processes. An important class of problems where we have to deal with white noise processes arise in linear SDEs where the above sparse method is currently not applicable there. Because of 2, the maximization of the marginal likelihood in eq. (2) with respect to $(\mathbf{Z}, \boldsymbol{\theta})$, where $\boldsymbol{\theta}$ are model hyperparameters, may be prone to overfitting especially when the number of variables in $\mathbf{Z}$ is large. Moreover, fitting a modified sparse GP model implies that the full GP model is not approximated in a systematic and rigorous way since there is no distance or divergence between the two models that is minimized.

In the next section, we address point 1 above by introducing the concept of variational inducing kernels that allow us to efficiently sparsify multioutput GP models having white noise latent functions. Further, these inducing kernels are incorporated into the variational inference method of Titsias (2009) (thus addressing point 2) that treats the inducing inputs $\mathbf{Z}$ as well as other quantities associated with the inducing kernels as variational parameters. The whole variational approach provides us with a very flexible, robust to overfitting, approximation framework that overcomes the limitations of the PITC approximation.

4 Sparse variational approximation

In this section, we introduce the concept of variational inducing kernels (VIKs). VIKs give us a way to define more general inducing variables that have larger approximation capacity than the $\mathbf{u}$ inducing variables used earlier and importantly allow us to deal with white noise latent functions. To motivate the idea, we first explain why the $\mathbf{u}$ variables can work when the latent functions are smooth and fail when these functions become white noises.

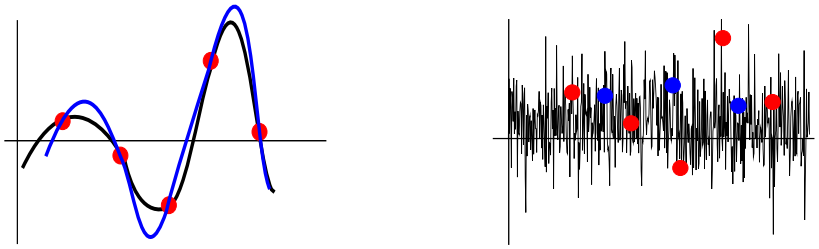
In PITC, we assume each latent function $u_r(\mathbf{x})$ is smooth and we sparsify the GP model through introducing, $\mathbf{u}_r$, inducing variables which are direct observations of the latent function, $u_r(\mathbf{x})$, at particular input points. Because of the latent function's smoothness, the $\mathbf{u}_r$ variables also carry information about other points in the function through the imposed prior over the latent function. So, having observed $\mathbf{u}_r$ we can reduce the uncertainty of the whole function.

With the vector of inducing variables $\mathbf{u}$, if chosen to be sufficiently large relative to the length scales of the latent functions, we can efficiently represent the functions $\{u_r(\mathbf{x})\}_{r=1}^{R}$ and subsequently variables $\mathbf{f}$ which are just convolved versions of the latent functions.[1] When the reconstruction of $\mathbf{f}$ from $\mathbf{u}$ is perfect, the conditional prior $p(\mathbf{f}|\mathbf{u})$ becomes a delta function and the sparse PITC approximation becomes exact. Figure 1(a) shows a cartoon description of a summarization of $u_r(\mathbf{x})$ by $\mathbf{u}_r$.

In contrast, when some of the latent functions are *white noise* processes the sparse approximation will fail. If $u_r(\mathbf{z})$ is white noise[2] it has a covariance function $\delta(\mathbf{z} - \mathbf{z}')$. Such processes naturally arise in the application of *stochastic differential equations* (see section 6) and are the ultimate non-

[1] This idea is like a "soft version" of the Nyquist-Shannon sampling theorem. If the latent functions were bandlimited, we could compute exact results given a high enough number of inducing points. In general they won't be bandlimited, but for smooth functions low frequency components will dominate over high frequencies, which will quickly fade away.

[2] Such a process can be thought as the "time derivative" of the Wiener process.



(a) Latent function is smooth (b) Latent function is noise

Figure 1: With a smooth latent function as in (a), we can use some inducing variables $\mathbf{u}_r$ (red dots) from the complete latent process $u_r(\mathbf{x})$ (in black) to generate smoothed versions (for example the one in blue), with uncertainty described by $p(u_r|\mathbf{u}_r)$. However, with a white noise latent function as in (b), choosing inducing variables $\mathbf{u}_r$ (red dots) from the latent process (in black) does not give us a clue about other points (for example the blue dots).

smooth processes where two values $u_r(\mathbf{z})$ and $u_r(\mathbf{z}')$ are uncorrelated when $\mathbf{z} \neq \mathbf{z}'$. When we apply the sparse approximation a vector of "white-noise" inducing variables $\mathbf{u}_r$ does not carry information about $u_r(\mathbf{z})$ at any input $\mathbf{z}$ that differs from all inducing inputs $\mathbf{Z}$. In other words there is no additional information in the conditional prior $p(u_r(\mathbf{z})|\mathbf{u}_r)$ over the unconditional prior $p(u_r(\mathbf{z}))$. Figure 1(b) shows a pictorial representation. The lack of structure makes it impossible to exploit the correlations in the standard sparse methods like PITC.[3]

Our solution to this problem is the following. We will define a more powerful form of inducing variable, one based not around the latent function at a point, but one given by the convolution of the latent function with a smoothing kernel. More precisely, let us replace each inducing vector $\mathbf{u}_r$ with variables $\boldsymbol{\lambda}_r$ which are evaluated at the inputs $\mathbf{Z}$ and are defined according to

$$\lambda_r(\mathbf{z}) = \int T_r(\mathbf{z} - \mathbf{v}) u_r(\mathbf{v}) \mathrm{d}\mathbf{v}, \tag{3}$$

where $T_r(\mathbf{x})$ is a smoothing kernel (e.g. Gaussian) which we call the *inducing kernel* (IK). This kernel is not necessarily related to the model's smoothing kernels. These newly defined inducing variables can carry information about $u_r(\mathbf{z})$ not only at a single input location but from the entire input space. We can even allow a separate IK for each inducing point, this is, if the set of inducing points is $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^K$, then $\lambda_r(\mathbf{z}_k) = \int T_{r,k}(\mathbf{z}_k - \mathbf{v}) u_r(\mathbf{v}) \mathrm{d}\mathbf{v}$, with the advantage of associating to each inducing point $\mathbf{z}_k$ its own set of adaptive parameters in $T_{r,k}$. For the PITC approximation, this adds more hyperparameters to the likelihood, perhaps leading to overfitting. However, in the variational approximation we define all these new parameters as variational parameters and therefore they do not cause the model to overfit.

If $u_r(\mathbf{z})$ has a white noise[4] GP prior the covariance function for $\lambda_r(\mathbf{x})$ is

$$\mathrm{Cov}[\lambda_r(\mathbf{x}), \lambda_r(\mathbf{x}')] = \int T_r(\mathbf{x} - \mathbf{z}) T_r(\mathbf{x}' - \mathbf{z}) \mathrm{d}\mathbf{z} \tag{4}$$

and the cross-covariance between $f_d(\mathbf{x})$ and $\lambda_r(\mathbf{x}')$ is

$$\mathrm{Cov}[f_d(\mathbf{x}), \lambda_r(\mathbf{x}')] = \int G_{d,r}(\mathbf{x} - \mathbf{z}) T_r(\mathbf{x}' - \mathbf{z}) d\mathbf{z}. \tag{5}$$

Notice that this cross-covariance function, unlike the case of $\mathbf{u}$ inducing variables, maintains a weighted integration over the whole input space. This implies that a single inducing variable $\lambda_r(\mathbf{x})$ can properly propagate information from the full-length process $u_r(\mathbf{x})$ into $\mathbf{f}$.

It is possible to combine the IKs defined above with the PITC approximation of Álvarez and Lawrence (2009), but in this paper our focus will be on applying them within the variational framework of Titsias (2009). We therefore refer to the kernels as variational inducing kernels (VIKs).

Variational inference

We now extend the variational inference method of Titsias (2009) to deal with multiple outputs and incorporate them into the VIK framework.

We compactly write the joint probability model $p(\{\mathbf{y}_d, \mathbf{f}_d\}_{d=1}^D)$ as $p(\mathbf{y}, \mathbf{f}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f})$. The first step of the variational method is to augment this model with inducing variables. For our purpose, suitable inducing variables are defined through VIKs. More precisely, let $\boldsymbol{\lambda} = (\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_R)$ be the whole vector of inducing variables where each $\boldsymbol{\lambda}_r$ is a K-dimensional vector of values obtained according to eq. (3). $\boldsymbol{\lambda}_r$'s role is to carry information about the latent function $u_r(\mathbf{z})$. Each $\boldsymbol{\lambda}_r$ is evaluated at the inputs $\mathbf{Z}$ and has its own VIK, $T_r(\mathbf{x})$, that depends on parameters $\boldsymbol{\theta}_{T_r}$. The $\boldsymbol{\lambda}$ variables augment the GP model according to $p(\mathbf{y}, \mathbf{f}, \boldsymbol{\lambda}) = p(\mathbf{y}|\mathbf{f})p(\mathbf{f}|\boldsymbol{\lambda})p(\boldsymbol{\lambda})$. Here, $p(\boldsymbol{\lambda}) = \mathcal{N}(\boldsymbol{\lambda}|\mathbf{0}, \mathbf{K}_{\boldsymbol{\lambda},\boldsymbol{\lambda}})$ and $\mathbf{K}_{\boldsymbol{\lambda},\boldsymbol{\lambda}}$ is a block diagonal

[3] Returning to our sampling theorem analogy, the white noise process has infinite bandwidth. It is therefore impossible to represent it by observations at a few fixed inducing points.

[4] It is straightforward to generalize the method for rough latent functions that are not white noise or to combine smooth latent functions with white noise.

matrix where each block $\mathbf{K}_{\boldsymbol{\lambda}_r,\boldsymbol{\lambda}_r}$ is obtained by evaluating the covariance function in eq. (4) at the inputs $\mathbf{Z}$. Additionally, $p(\mathbf{f}|\boldsymbol{\lambda}) = \mathcal{N}(\mathbf{f}|\mathbf{K}_{\mathbf{f},\boldsymbol{\lambda}}\mathbf{K}_{\boldsymbol{\lambda},\boldsymbol{\lambda}}^{-1}\boldsymbol{\lambda}, \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\boldsymbol{\lambda}}\mathbf{K}_{\boldsymbol{\lambda},\boldsymbol{\lambda}}^{-1}\mathbf{K}_{\boldsymbol{\lambda},\mathbf{f}})$ where the cross-covariance $\mathbf{K}_{\mathbf{f},\boldsymbol{\lambda}}$ is computed through eq. (5). Because of the consistency condition $\int p(\mathbf{f}|\boldsymbol{\lambda})p(\boldsymbol{\lambda})\mathrm{d}\boldsymbol{\lambda} = p(\mathbf{f})$, performing exact inference in the above augmented model is equivalent to performing exact inference in the initial GP model. Crucially, this holds for any values of the *augmentation* parameters $(\mathbf{Z}, \{\boldsymbol{\theta}_{T_r}\}_{r=1}^R)$. This is the key property that allows us to turn these augmentation parameters into variational parameters by applying approximate sparse inference.

Our method now proceeds along the lines of Titsias (2009). We introduce the variational distribution $q(\mathbf{f}, \boldsymbol{\lambda}) = p(\mathbf{f}|\boldsymbol{\lambda})\phi(\boldsymbol{\lambda})$, where $p(\mathbf{f}|\boldsymbol{\lambda})$ is the conditional GP prior defined earlier and $\phi(\boldsymbol{\lambda})$ is an arbitrary variational distribution. By minimizing the KL divergence between $q(\mathbf{f}, \boldsymbol{\lambda})$ and the true posterior $p(\mathbf{f}, \boldsymbol{\lambda}|\mathbf{y})$, we can compute the following Jensen's lower bound on the true log marginal likelihood (a detailed derivation of the bound is available in Álvarez *et al.* (2009b)):

$$F_V = \log \mathcal{N}\left(\mathbf{y}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\boldsymbol{\lambda}}\mathbf{K}_{\boldsymbol{\lambda},\boldsymbol{\lambda}}^{-1}\mathbf{K}_{\boldsymbol{\lambda},\mathbf{f}} + \boldsymbol{\Sigma}\right) - \frac{1}{2}\operatorname{tr}\left(\boldsymbol{\Sigma}^{-1}\widetilde{\mathbf{K}}\right),$$

where $\boldsymbol{\Sigma}$ is the covariance function associated with the additive noise process and $\widetilde{\mathbf{K}} = \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\boldsymbol{\lambda}}\mathbf{K}_{\boldsymbol{\lambda},\boldsymbol{\lambda}}^{-1}\mathbf{K}_{\boldsymbol{\lambda},\mathbf{f}}$. Note that this bound consists of two parts. The first part is the log of a GP prior with the only difference that now the covariance matrix has a particular low rank form. This form allows the inversion of the covariance matrix to take place in $O(NDK^2)$ time rather than $O(N^3D^3)$. The second part can be seen as a penalization term that regularizes the estimation of the parameters. Notice also that only the diagonal of the exact covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ needs to be computed. Overall, the computation of the bound can be done efficiently in $O(NDK^2)$ time.

The bound can be maximized with respect to all parameters of the covariance function; both model parameters and variational parameters. The variational parameters are the inducing inputs $\mathbf{Z}$ and the parameters $\boldsymbol{\theta}_{T_r}$ of each VIK which are rigorously selected so that the KL divergence is minimized. In fact each VIK is also a variational quantity and one could try different forms of VIKs in order to choose the one that gives the best lower bound.

The form of the bound is very similar to the projected process approximation, also known as DTC (Csató and Opper, 2001; Seeger *et al.*, 2003; Rasmussen and Williams, 2006). However, the bound has an additional trace term that penalizes the movement of inducing inputs away from the data. This term converts the DTC approximation to a lower bound and prevents overfitting. In what follows, we refer to this approximation as DTCVAR, where the VAR suffix refers to the variational framework.

5 Experiments

We present results of applying the method proposed for two real-world datasets that will be described in short. We compare the results obtained using PITC, the intrinsic coregionalization model (ICM)[5] employed in Bonilla *et al.* (2008) and the method using variational inducing kernels. For PITC we estimate the parameters through the maximization of the approximated marginal likelihood of equation (2) using the scaled-conjugate gradient method. We use one latent function and both the covariance function of the latent process, $k_r(\mathbf{x}, \mathbf{x}')$, and the kernel smoothing function, $G_{d,r}(\mathbf{x})$, follow a Gaussian form, this is $k(\mathbf{x}, \mathbf{x}') = \mathcal{N}(\mathbf{x} - \mathbf{x}'|\mathbf{0}, \mathbf{C})$, where $\mathbf{C}$ is a diagonal matrix. For the DTCVAR approximations, we maximize the variational bound F_V. Optimization is also performed using scaled conjugate gradient. We use one white noise latent function and a corresponding inducing function. The inducing kernels and the model kernels follow the same Gaussian form. Using this form for the covariance or kernel, all convolution integrals are solved analytically.

5.1 Exam score prediction

In this experiment the goal is to predict the exam score obtained by a particular student belonging to a particular school. The data comes from the Inner London Education Authority (ILEA).[6] It consists of examination records from 139 secondary schools in years 1985, 1986 and 1987. It is a random 50% sample with 15362 students. The input space consists of features related to each student and features related to each school. From the multiple output point of view, each school represents one output and the exam score of each student a particular instantiation of that output.

We follow the same preprocessing steps employed in Bonilla *et al.* (2008). The only features used are the student-dependent ones (year in which each student took the exam, gender, VR band and ethnic group), which are categorical variables. Each of them is transformed to a binary representation. For example, the possible values that the variable year of the exam can take are 1985, 1986 or 1987 and are represented as 100, 010 or 001. The transformation is also applied to the variables gender (two binary variables), VR band (four binary variables) and ethnic group (eleven binary variables), ending up with an input space with dimension 20. The categorical nature of data restricts the input space to 202 unique input feature vectors. However, two students represented by the same input vector $\mathbf{x}$ and belonging both to the same school d, can obtain different exam scores. To reduce this noise in the

[5] The ICM is a particular case of the LMC with one latent function (Goovaerts, 1997).

[6] Data is available at `http://www.cmm.bristol.ac.uk/learning-training/multilevel-m-support/datasets.shtml`

data, we follow Bonilla *et al.* (2008) in taking the mean of the observations that, within a school, share the same input vector and use a simple heteroskedastic noise model in which the variance for each of these means is divided by the number of observations used to compute it. The performance measure employed is the percentage of unexplained variance defined as the sum-squared error on the test set as a percentage of the total data variance.[7] The performance measure is computed for ten repetitions with 75% of the data in the training set and 25% of the data in the test set.

Figure 5.1 shows results using PITC, DTCVAR with one smoothing kernel and DTCVAR with as many inducing kernels as inducing points (DTCVARS in the figure). For 50 inducing points all three alternatives lead to approximately the same results. PITC keeps a relatively constant performance for all values of inducing points, while the DTCVAR approximations increase their performance as the number of inducing points increases. This is consistent with the expected behaviour of the DTCVAR methods, since the trace term penalizes the model for a reduced number of inducing points. Notice that all the approximations outperform independent GPs and the best result of the ICM presented in Bonilla *et al.* (2008).

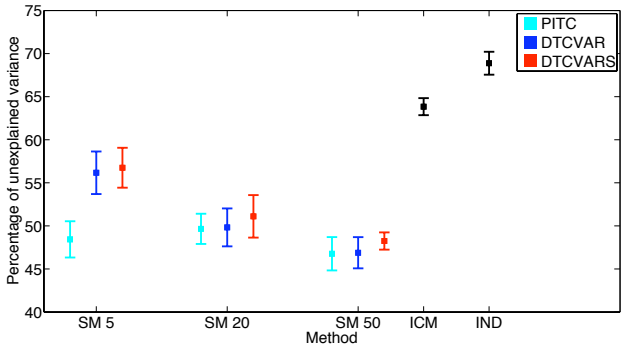


Figure 2: Exam score prediction results for the school dataset. Results include the mean of the percentage of unexplained variance of ten repetitions of the experiment, together with one standard deviation. In the bottom, SM X stands for sparse method with X inducing points, DTCVAR refers to the DTC variational approximation with one smoothing kernel and DTCVARS to the same approximation using as many inducing kernels as inducing points. Results using the ICM model and independent GPs (appearing as IND in the figure) have also been included.

5.2 Compiler prediction performance

In this dataset the outputs correspond to the speed-up of 11 C programs after some transformation sequence has been applied to them. The speed-up is defined as the execution time of the original program divided by the execution time of the transformed program. The input space consists of 13-dimensional binary feature vectors, where the presence of a one in these vectors indicates that the program has received that particular transformation. The dataset contains 88214 observations for each output and the same number of input vectors. All the outputs share the same input space. Due to technical requirements, it is important that the prediction of the speed-up for the particular program is made using few observations in the training set. We compare our results to the ones presented in Bonilla *et al.* (2008) and use $N = 16, 32, 64$ and 128 for the training set. The remaining $88214 - N$ observations are used for testing, employing as performance measure the mean absolute error. The experiment is repeated ten times and standard deviations are also reported. We only include results for the average performance over the 11 outputs.

Figure 3 shows the results of applying independent GPs (IND in the figure), the intrinsic coregionalization model (ICM in the figure), PITC, DTCVAR with one inducing kernel (DTCVAR in the figure) and DTCVAR with as many inducing kernels as inducing points (DTCVARS in the figure). Since the training sets are small enough, we also include results of applying the GP generated using the full covariance matrix of the convolution construction (see FULL GP in the figure). We repeated the experiment for different values of K, but show results only for $K = N/2$. Results for ICM and IND were obtained from Bonilla *et al.* (2008). In general, the DTCVAR variants outperform the

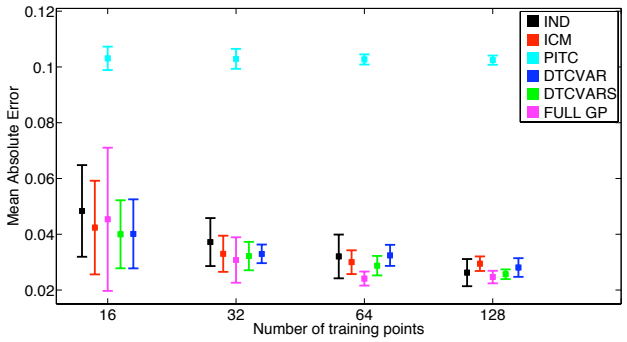


Figure 3: Mean absolute error and standard deviation over ten repetitions of the compiler experiment as a function of the training points. IND stands for independent GPs, ICM stands for intrinsic coregionalization model, DTCVAR refers to the DTCVAR approximation using one inducing kernel, DTCVARS refers to the DTCVAR approximation using as many inducing kernels as inducing points and FULL GP stands for the GP for the multiple outputs without any approximation.

ICM method, and the independent GPs for $N = 16,\ 32$ and 64. In this case, using as many inducing kernels as inducing points improves on average the performance. All methods, including the independent GPs are better than PITC. The size of the test set encourages the application of the sparse methods: for $N = 128$, making the prediction of the whole dataset using the full GP takes in average 22 minutes while the prediction with DTCVAR takes 0.65 minutes. Using more inducing kernels improves the performance, but also

[7] In Bonilla *et al.* (2008), results are reported in terms of explained variance.

makes the evaluation of the test set more expensive. For DTCVARS, the evaluation takes in average 6.8 minutes. Time results are average results over the ten repetitions.

6 Stochastic Latent Force Models

The starting point of stochastic differential equations is a stochastic version of the equation of motion, which is called Langevin's equation:

$$\frac{\mathrm{d}f(t)}{\mathrm{d}t} = -Cf(t) + Su(t), \tag{6}$$

where $f(t)$ is the velocity of the particle, $-Cf(t)$ is a systematic friction term, $u(t)$ is a random fluctuation external force, i.e. white noise, and S indicates the sensitivity of the ouput to the random fluctuations. In the mathematical probability literature, the above is written more rigorously as $\mathrm{d}f(t) = -Cf(t)\mathrm{d}t + S\mathrm{d}W(t)$ where $W(t)$ is the Wiener process (standard Brownian motion). Since $u(t)$ is a GP and the equation is linear, $f(t)$ must be also a GP which turns out to be the Ornstein-Uhlenbeck (OU) process.

Here, we are interested in extending the Langevin equation to model multivariate time series. The way that the model in (6) is extended is by adding more output signals and more external forces. The forces can be either smooth (systematic or drift-type) forces or white noise forces. Thus,

$$\frac{\mathrm{d}f_d(t)}{\mathrm{d}t} = -D_d f_d(t) + \sum_{r=1}^{R} S_{d,r} u_r(t), \tag{7}$$

where $f_d(t)$ is the dth output signal. Each $u_r(t)$ can be either a smooth latent force that is assigned a GP prior with covariance function $k_r(t,t')$ or a white noise force that has a GP prior with covariance function $\delta(t-t')$. That is, we have a composition of R latent forces, where R_s of them correspond to smooth latent forces and R_o correspond to white noise processes. The intuition behind this combination of input forces is that the smooth part can be used to represent medium/long term trends that cause a departure from the mean of the output processes, whereas the stochastic part is related to short term fluctuations around the mean. A model with $R_s = 1$ and $R_o = 0$ was proposed by Lawrence *et al.* (2007) to describe protein transcription regulation in a single input motif (SIM) gene network.

Solving the differential equation (7), we obtain

$$f_d(t) = e^{-D_d t} f_{d0} + \sum_{r=1}^{R} S_{d,r} \int_0^t e^{-D_d(t-z)} u_r(z) dz,$$

where f_{d0} arises from the initial condition. This model now is a special case of the multioutput regression model discussed in sections 1 and 2 where each output signal $y_d(t) = f_d(t) + \epsilon$ has a mean function $e^{-D_d t} f_{d0}$ and each model kernel $G_{d,r}(\mathbf{x})$ is equal to $S_{d,r} e^{-D_d(t-z)}$. The above model can be viewed as a stochastic latent force model (SLFM) following the work of Álvarez *et al.* (2009a).

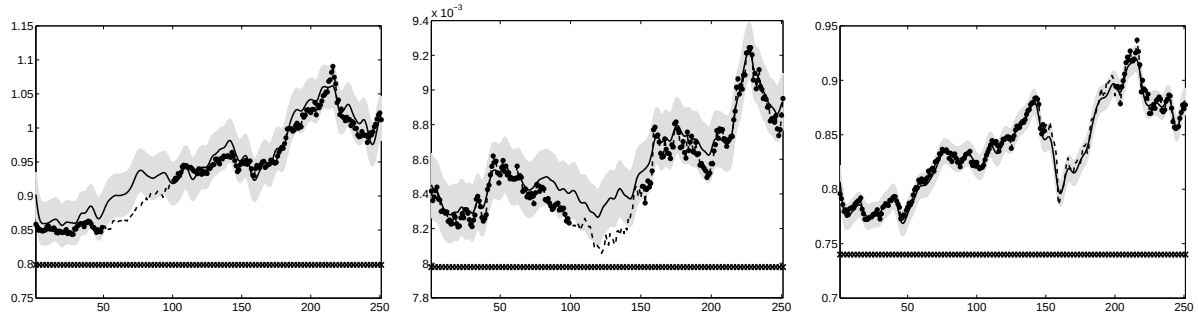
Latent market forces

The application considered is the inference of missing data in a multivariate financial data set: the foreign exchange rate w.r.t. the dollar of 10 of the top international currencies (Canadian Dollar [CAD], Euro [EUR], Japanese Yen [JPY], Great British Pound [GBP], Swiss Franc [CHF], Australian Dollar [AUD], Hong Kong Dollar [HKD], New Zealand Dollar [NZD], South Korean Won [KRW] and Mexican Peso [MXN]) and 3 precious metals (gold [XAU], silver [XAG] and platinum [XPT]).[8] We considered all the data available for the calendar year of 2007 (251 working days). In this data there are several missing values: XAU, XAG and XPT have 9, 8 and 42 days of missing values respectively. On top of this, we also introduced artificially long sequences of missing data. Our objective is to model the data and test the effectiveness of the model by imputing these missing points. We removed a test set from the data by extracting contiguous sections from 3 currencies associated with very different geographic locations: we took days 50–100 from CAD, days 100–150 from JPY and days 150–200 from AUD. The remainder of the data comprised the training set, which consisted of 3051 points, with the test data containing 153 points. For preprocessing we removed the mean from each output and scaled them so that they all had unit variance.

It seems reasonable to suggest that the fluctuations of the 13 correlated financial time-series are driven by a smaller number of latent market forces. We therefore modelled the data with up to six latent forces which could be noise or smooth GPs. The GP priors for the smooth latent forces are assumed to have a Gaussian covariance function, $k_{u_r u_r}(t,t') = (1/\sqrt{2\pi\ell_r^2}) \exp(-((t-t')^2)/2\ell_r^2)$, where the hyperparameter ℓ_r is known as the lengthscale.

We present an example with $R = 4$. For this value of R, we consider all the possible combinations of R_o and R_s. The training was performed in all cases by maximizing the variational bound using the scale conjugate gradient algorithm until convergence was achieved. The best performance in terms of achieving the highest value for F_V was obtained for $R_s = 1$ and $R_o = 3$. We compared against the LMC model for different values of the latent functions in that framework. While our best model resulted in an standardized mean square error of 0.2795, the best LMC (with R=2) resulted in 0.3927. We plotted predictions from the latent market force model to characterize the performance when filling in missing data. In figure 4 we show the output signals obtained using the model with the highest bound ($R_s = 1$ and $R_o = 3$) for CAD, JPY and AUD. Note that the model performs better at capturing the deep drop in AUD than it does for the fluctuations in CAD and JPY.

[8] Data is available at `http://fx.sauder.ubc.ca/data.html`.



(a) CAD: Real data and prediction (b) JPY: Real data and prediction (c) AUD: Real data and prediction

Figure 4: Predictions from the model with $R_s = 1$ and $R_o = 3$ are shown as solid lines for the mean and grey bars for error bars at 2 standard deviations. For CAD, JPY and AUD the data was artificially held out. The true values are shown as a dotted line. Crosses on the x-axes of all plots show the locations of the inducing inputs.

7 Conclusions

We have presented a variational approach to sparse approximations in convolution processes. Our main focus was to provide efficient mechanisms for learning in multiple output Gaussian processes when the latent function is fluctuating rapidly. In order to do so, we have introduced the concept of inducing function, which generalizes the idea of inducing point, traditionally employed in sparse GP methods. The approach extends the variational approximation of Titsias (2009) to the multiple output case. Using our approach we can perform efficient inference on latent force models which are based around SDEs, but also contain a smooth driving force. Our approximation is flexible enough and has been shown to be applicable to a wide range of data sets, including high-dimensional ones.

Acknowledgements

We would like to thank Edwin Bonilla for his valuable feedback with respect to the datasets in section 5. Also to the authors of Bonilla *et al.* (2008) who kindly made the compiler dataset available. DL has been partly financed by the Spanish government through CICYT projects TEC2006-13514-C02-01 and TEC2009-14504-C02-01, and the CONSOLIDER-INGENIO 2010 Program (Project CSD2008-00010). MA and NL have been financed by a Google Research Award "Mechanistically Inspired Convolution Processes for Learning" and MA, NL and MT have been financed by EPSRC Grant No EP/F005687/1 "Gaussian Processes for Systems Identification with Applications in Systems Biology".

References

Mauricio A. Álvarez and Neil D. Lawrence. Sparse convolved Gaussian processes for multi-output regression. In *NIPS 21*, pages 57–64. MIT Press, 2009.

Mauricio A. Álvarez, David Luengo, and Neil D. Lawrence. Latent Force Models. In *JMLR: W&CP 5*, pages 9–16, 2009.

Mauricio A. Álvarez, David Luengo, Michalis K. Titsias, and Neil D. Lawrence. Variational inducing kernels for sparse convolved multiple output Gaussian processes. Technical report, School of Computer Science, University of Manchester, 2009. Available at `http://arxiv.org/abs/0912.3268`.

Edwin V. Bonilla, Kian Ming Chai, and Christopher K. I. Williams. Multi-task Gaussian process prediction. In *NIPS 20*, pages 153–160. MIT Press, 2008.

Phillip Boyle and Marcus Frean. Dependent Gaussian processes. In *NIPS 17*, pages 217–224. MIT Press, 2005.

Lehel Csató and Manfred Opper. Sparse representation for Gaussian process models. In *NIPS 13*, pages 444–450. MIT Press, 2001.

Pierre Goovaerts. *Geostatistics For Natural Resources Evaluation*. Oxford University Press, 1997.

David M. Higdon. Space and space-time modelling using process convolutions. In *Quantitative methods for current environmental issues*, pages 37–56. Springer-Verlag, 2002.

Neil D. Lawrence, Guido Sanguinetti, and Magnus Rattray. Modelling transcriptional regulation using Gaussian processes. In *NIPS 19*, pages 785–792. MIT Press, 2007.

Miguel Lázaro-Gredilla and Aníbal Figueiras-Vidal. Inter-domain Gaussian processes for sparse inference using inducing features. In *NIPS 22*, pages 1087–1095. MIT press, 2010.

Joaquin Quiñonero Candela and Carl Edward Rasmussen. A unifying view of sparse approximate Gaussian process regression. *JMLR*, 6:1939–1959, 2005.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

Alex 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 *Proc. Int. Conf. on Information Proc. in Sensor Networks (IPSN 2008)*, 2008.

Matthias Seeger, Christopher K. I. Williams, and Neil D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In *Proc. 9th Int. Workshop on Artificial Intelligence and Statistics*, Key West, FL, 3–6 January 2003.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *NIPS 18*. MIT Press, 2006.

Yee Whye Teh, Matthias Seeger, and Michael I. Jordan. Semiparametric latent factor models. In *AISTATS 10*, pages 333–340, 2005.

Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *JMLR: W&CP 5*, pages 567–574, 2009.

Reprint of publication IV

Mauricio A. Álvarez, Jan Peters, Bernhard Schölkopf and Neil D. Lawrence (2011): Switched latent force models for movement segmentation, in J. S.e.Taylor, R. Zemel, C. Williams and J. Lafferty (Eds), Advances in Neural Information Processing Systems 23, pp 55-63, 2011.

Switched Latent Force Models for Movement Segmentation

Mauricio A. Álvarez [1], **Jan Peters** [2], **Bernhard Schölkopf** [2], **Neil D. Lawrence** [3,4]
[1] School of Computer Science, University of Manchester, Manchester, UK M13 9PL
[2] Max Planck Institute for Biological Cybernetics, Tübingen, Germany 72076
[3] School of Computer Science, University of Sheffield, Sheffield, UK S1 4DP
[4] The Sheffield Institute for Translational Neuroscience, Sheffield, UK S10 2HQ

Abstract

Latent force models encode the interaction between multiple related dynamical systems in the form of a kernel or covariance function. Each variable to be modeled is represented as the output of a differential equation and each differential equation is driven by a weighted sum of latent functions with uncertainty given by a Gaussian process prior. In this paper we consider employing the latent force model framework for the problem of determining robot motor primitives. To deal with discontinuities in the dynamical systems or the latent driving force we introduce an extension of the basic latent force model, that switches between different latent functions and potentially different dynamical systems. This creates a versatile representation for robot movements that can capture discrete changes and non-linearities in the dynamics. We give illustrative examples on both synthetic data and for striking movements recorded using a Barrett WAM robot as haptic input device. Our inspiration is robot motor primitives, but we expect our model to have wide application for dynamical systems including models for human motion capture data and systems biology.

1 Introduction

Latent force models [1] are a new approach for modeling data that allows combining dimensionality reduction with systems of differential equations. The basic idea is to assume an observed set of D correlated functions to arise from an unobserved set of R forcing functions. The assumption is that the R forcing functions drive the D observed functions through a set of differential equation models. Each differential equation is driven by a weighted mix of latent forcing functions. Sets of coupled differential equations arise in many physics and engineering problems particularly when the temporal evolution of a system needs to be described. Learning such differential equations has important applications, e.g., in the study of human motor control and in robotics [6]. A latent force model differs from classical approaches as it places a probabilistic process prior over the latent functions and hence can make statements about the uncertainty in the system. A joint Gaussian process model over the latent forcing functions and the observed data functions can be recovered using a Gaussian process prior in conjunction with linear differential equations [1]. The resulting latent force modeling framework allows the combination of the knowledge of the systems dynamics with a data driven model. Such generative models can be used to good effect, for example in ranked target prediction for transcription factors [5].

If a single Gaussian process prior is used to represent each latent function then the models we consider are limited to smooth driving functions. However, discontinuities and segmented latent forces are omnipresent in real-world data. For example, impact forces due to contacts in a mechanical dynamical system (when grasping an object or when the feet touch the ground) or a switch in an electrical circuit result in discontinuous latent forces. Similarly, most non-rhythmic natural mo-

tor skills consist of a sequence of segmented, discrete movements. If these segments are separate time-series, they should be treated as such and *not* be modeled by the same Gaussian process model.

In this paper, we extract a sequence of dynamical systems motor primitives modeled by second order linear differential equations in conjunction with forcing functions (as in [1, 6]) from human movement to be used as demonstrations of elementary movements for an anthropomorphic robot. As human trajectories have a large variability: both due to planned uncertainty of the human's movement policy, as well as due to motor execution errors [7], a probabilistic model is needed to capture the underlying motor primitives. A set of second order differential equations is employed as mechanical systems are of the same type and a temporal Gaussian process prior is used to allow probabilistic modeling [1]. To be able to obtain a sequence of dynamical systems, we augment the latent force model to include discontinuities in the latent function and change dynamics. We introduce discontinuities by switching between different Gaussian process models (superficially similar to a mixture of Gaussian processes; however, the switching times are modeled as parameters so that at any instant a single Gaussian process is driving the system). Continuity of the observed functions is then ensured by constraining the relevant state variables (for example in a second order differential equation velocity and displacement) to be continuous across the switching points. This allows us to model highly non stationary multivariate time series. We demonstrate our approach on synthetic data and real world movement data.

2 Review of Latent force models (LFM)

Latent force models [1] are hybrid models that combine mechanistic principles and Gaussian processes as a flexible way to introduce prior knowledge for data modeling. A set of D functions $\{y_d(t)\}_{d=1}^D$ is modeled as the set of output functions of a series of coupled differential equations, whose common input is a linear combination of R latent functions, $\{u_r(t)\}_{r=1}^R$. Here we focus on a second order ordinary differential equation (ODE). We assume the output $y_d(t)$ is described by

$$A_d \frac{\mathrm{d}^2 y_d(t)}{\mathrm{d}t^2} + C_d \frac{\mathrm{d}y_d(t)}{\mathrm{d}t} + \kappa_d y_d(t) = \textstyle\sum_{r=1}^R S_{d,r} u_r(t),$$

where, for a mass-spring-damper system, A_d would represent the mass, C_d the damper and κ_d, the spring constant associated to the output d. We refer to the variables $S_{d,r}$ as the sensitivity parameters. They are used to represent the relative strength that the latent force r exerts over the output d. For simplicity we now focus on the case where $R = 1$, although our derivations apply more generally. Note that models that learn a forcing function to drive a linear system have proven to be well-suited for imitation learning for robot systems [6]. The solution of the second order ODE follows

$$y_d(t) = y_d(0)c_d(t) + \dot{y}_d(0)e_d(t) + f_d(t,u), \tag{1}$$

where $y_d(0)$ and $\dot{y}_d(0)$ are the output and the velocity at time $t = 0$, respectively, known as the initial conditions (IC). The angular frequency is given by $\omega_d = \sqrt{(4A_d\kappa_d - C_d^2)/(4A_d^2)}$ and the remaining variables are given by

$$c_d(t) = e^{-\alpha_d t}\Big[\cos(\omega_d t) + \frac{\alpha_d}{\omega_d}\sin(\omega_d t)\Big], \quad e_d(t) = \frac{e^{-\alpha_d t}}{\omega_d}\sin(\omega_d t),$$
$$f_d(t,u) = \frac{S_d}{A_d\omega_d}\int_0^t G_d(t-\tau)u(\tau)\mathrm{d}\tau = \frac{S_d}{A_d\omega_d}\int_0^t e^{-\alpha_d(t-\tau)}\sin[(t-\tau)\omega_d]u(\tau)\mathrm{d}\tau,$$

with $\alpha_d = C_d/(2A_d)$. Note that $f_d(t,u)$ has an implicit dependence on the latent function $u(t)$. The uncertainty in the model of Eq. (1) is due to the fact that the latent force $u(t)$ and the initial conditions $y_d(0)$ and $\dot{y}_d(0)$ are not known. We will assume that the latent function $u(t)$ is sampled from a zero mean Gaussian process prior, $u(t) \sim \mathcal{GP}(0, k_{u,u}(t,t'))$, with covariance function $k_{u,u}(t,t')$.
If the initial conditions, $\mathbf{y}_{IC} = [y_1(0), y_2(0), \ldots, y_D(0), v_1(0), v_2(0), \ldots, v_D(0)]^\top$, are independent of $u(t)$ and distributed as a zero mean Gaussian with covariance K_{IC} the covariance function between any two output functions, d and d' at any two times, t and t', $k_{y_d,y_{d'}}(t,t')$ is given by

$$c_d(t)c_{d'}(t')\sigma_{y_d,y_{d'}} + c_d(t)e_{d'}(t')\sigma_{y_d,v_{d'}} + e_d(t)c_{d'}(t')\sigma_{v_d,y_{d'}} + e_d(t)e_{d'}(t')\sigma_{v_d,v_{d'}} + k_{f_d,f_{d'}}(t,t'),$$

where $\sigma_{y_d,y_{d'}}$, $\sigma_{y_d,v_{d'}}$, $\sigma_{v_d,y_{d'}}$ and $\sigma_{v_d,v_{d'}}$ are entries of the covariance matrix K_{IC} and

$$k_{f_d,f_{d'}}(t,t') = K_0 \textstyle\int_0^t G_d(t-\tau)\int_0^{t'} G_{d'}(t'-\tau')k_{u,u}(t,t')\mathrm{d}\tau'\mathrm{d}\tau, \tag{2}$$

where $K_0 = S_d S_{d'}/(A_d A_{d'} \omega_d \omega_{d'})$. So the covariance function $k_{f_d,f_{d'}}(t,t')$ depends on the covariance function of the latent force $u(t)$. If we assume the latent function has a radial basis function (RBF) covariance, $k_{u,u}(t,t') = \exp[-(t-t')^2/\ell^2]$, then $k_{f_d,f_{d'}}(t,t')$ can be computed analytically [1] (see also supplementary material). The latent force model induces a joint Gaussian process model across all the outputs. The parameters of the covariance function are given by the parameters of the differential equations and the length scale of the latent force. Given a multivariate time series data set these parameters may be determined by maximum likelihood.
The model can be thought of as a set of mass-spring-dampers being driven by a function sampled from a Gaussian process. In this paper we look to extend the framework to the case where there can be discontinuities in the latent functions. We do this through switching between different Gaussian process models to drive the system.

3 Switching dynamical latent force models (SDLFM)

We now consider switching the system between different latent forces. This allows us to change the dynamical system and the driving force for each segment. By constraining the displacement and velocity at each switching time to be the same, the output functions remain continuous.

3.1 Definition of the model

We assume that the input space is divided in a series of non-overlapping intervals $[t_{q-1}, t_q]_{q=1}^{Q}$. During each interval, only one force $u_{q-1}(t)$ out of Q forces is active, that is, there are $\{u_{q-1}(t)\}_{q=1}^{Q}$ forces. The force $u_{q-1}(t)$ is activated after time t_{q-1} (switched on) and deactivated (switched off) after time t_q. We can use the basic model in equation (1) to describe the contribution to the output due to the sequential activation of these forces. A particular output $z_d(t)$ at a particular time instant t, in the interval (t_{q-1}, t_q), is expressed as

$$z_d(t) = y_d^q(t - t_{q-1}) = c_d^q(t - t_{q-1}) y_d^q(t_{q-1}) + e_d^q(t - t_{q-1}) \dot{y}_d^q(t_{q-1}) + f_d^q(t - t_{q-1}, u_{q-1}).$$

This equation is assummed to be valid for describing the output only inside the interval (t_{q-1}, t_q). Here we highlighted this idea by including the superscript q in $y_d^q(t - t_{q-1})$ to represent the interval q for which the equation holds, although later we will omit it to keep the notation uncluttered. Note that for $Q = 1$ and $t_0 = 0$, we recover the original latent force model given in equation (1). We also define the velocity $\dot{z}_d(t)$ at each time interval (t_{q-1}, t_q) as

$$\dot{z}_d(t) = \dot{y}_d^q(t - t_{q-1}) = g_d^q(t - t_{q-1}) y_d^q(t_{q-1}) + h_d^q(t - t_{q-1}) \dot{y}_d^q(t_{q-1}) + m_d^q(t - t_{q-1}, u_{q-1}),$$

where $g_d(t) = -e^{-\alpha_d t} \sin(\omega_d t)(\alpha_d^2 \omega_d^{-1} + \omega_d)$ and

$$h_d(t) = -e^{-\alpha_d t}\left[\frac{\alpha_d}{\omega_d}\sin(\omega_d t) - \cos(\omega_d t)\right], \quad m_d(t) = \frac{S_d}{A_d \omega_d}\frac{\mathrm{d}}{\mathrm{d}t}\left(\int_0^t G_d(t-\tau) u(\tau) \mathrm{d}\tau\right).$$

Given the parameters $\boldsymbol{\theta} = \{\{A_d, C_d, \kappa_d, S_d\}_{d=1}^{D}, \{\ell_{q-1}\}_{q=1}^{Q}\}$, the uncertainty in the outputs is induced by the prior over the initial conditions $y_d^q(t_{q-1}), \dot{y}_d^q(t_{q-1})$ for all values of t_{q-1} and the prior over latent force $u_{q-1}(t)$ that is active during (t_{q-1}, t_q). We place independent Gaussian process priors over each of these latent forces $u_{q-1}(t)$, assuming independence between them.

For initial conditions $y_d^q(t_{q-1}), \dot{y}_d^q(t_{q-1})$, we could assume that they are either parameters to be estimated or random variables with uncertainty governed by independent Gaussian distributions with covariance matrices K_{IC}^q as described in the last section. However, for the class of applications we will consider: mechanical systems, the outputs should be continuous across the switching points. We therefore assume that the uncertainty about the initial conditions for the interval q, $y_d^q(t_{q-1}), \dot{y}_d^q(t_{q-1})$ are proscribed by the Gaussian process that describes the outputs $z_d(t)$ and velocities $\dot{z}_d(t)$ in the previous interval $q-1$. In particular, we assume $y_d^q(t_{q-1}), \dot{y}_d^q(t_{q-1})$ are Gaussian-distributed with mean values given by $y_d^{q-1}(t_{q-1} - t_{q-2})$ and $\dot{y}_d^{q-1}(t_{q-1} - t_{q-2})$ and covariances $k_{z_d,z_{d'}}(t_{q-1}, t_{q'-1}) = \mathrm{cov}[y_d^{q-1}(t_{q-1} - t_{q-2}), y_{d'}^{q-1}(t_{q-1} - t_{q-2})]$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1}, t_{q'-1}) = \mathrm{cov}[\dot{y}_d^{q-1}(t_{q-1} - t_{q-2}), \dot{y}_{d'}^{q-1}(t_{q-1} - t_{q-2})]$. We also consider covariances between $z_d(t_{q-1})$ and $\dot{z}_{d'}(t_{q'-1})$, this is, between positions and velocities for different values of q and d.

Example 1. Let us assume we have one output ($D = 1$) and three switching intervals ($Q = 3$) with switching points t_0, t_1 and t_2. At t_0, we assume that $\mathbf{y}_{IC}$ follows a Gaussian distribution with

mean zero and covariance K_{IC}. From t_0 to t_1, the output $z(t)$ is described by

$$z(t) = y^1(t-t_0) = c^1(t-t_0)y^1(t_0) + e^1(t-t_0)\dot{y}^1(t_0) + f^1(t-t_0, u_0).$$

The initial condition for the position in the interval (t_1, t_2) is given by the last equation evaluated a t_1, this is, $z(t_1) = y^2(t_1) = y^1(t_1 - t_0)$. A similar analysis is used to obtain the initial condition associated to the velocity, $\dot{z}(t_1) = \dot{y}^2(t_1) = \dot{y}^1(t_1 - t_0)$. Then, from t_1 to t_2, the output $z(t)$ is

$$\begin{aligned} z(t) = y^2(t-t_1) &= c^2(t-t_1)y^2(t_1) + e^2(t-t_1)\dot{y}^2(t_1) + f^2(t-t_1, u_1), \\ &= c^2(t-t_1)y^1(t_1-t_0) + e^2(t-t_1)\dot{y}^1(t_1-t_0) + f^2(t-t_1, u_1). \end{aligned}$$

Following the same train of thought, the output $z(t)$ from t_2 is given as

$$z(t) = y^3(t-t_2) = c^3(t-t_2)y^3(t_2) + e^3(t-t_2)\dot{y}^3(t_2) + f^3(t-t_2, u_2),$$

where $y^3(t_2) = y^2(t_2 - t_1)$ and $\dot{y}^3(t_2) = \dot{y}^2(t_2 - t_1)$. Figure 1 shows an example of the switching dynamical latent force model scenario. To ensure the continuity of the outputs, the initial condition is forced to be equal to the output of the last interval evaluated at the switching point.

3.2 The covariance function

The derivation of the covariance function for the switching model is rather involved. For continuous output signals, we must take into account constraints at each switching time. This causes initial conditions for each interval to be dependent on final conditions for the previous interval and induces correlations across the intervals. This effort is worthwhile though as the resulting model is very flexible and can take advantage of the switching dynamics to represent a range of signals.

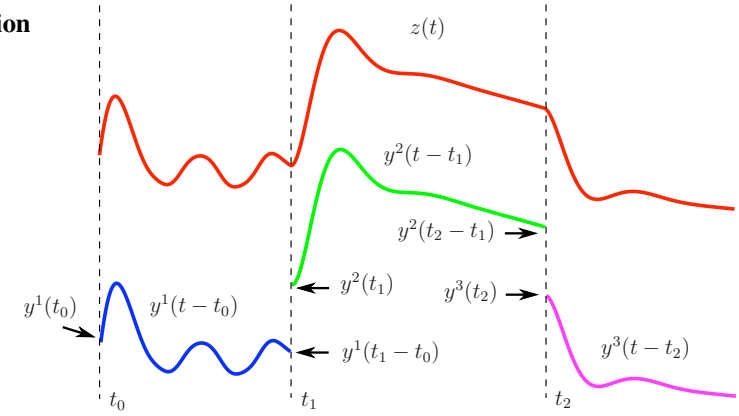


Figure 1: Representation of an output constructed through a switching dynamical latent force model with $Q = 3$. The initial conditions $y^q(t_{q-1})$ for each interval are matched to the value of the output in the last interval, evaluated at the switching point t_{q-1}, this is, $y^q(t_{q-1}) = y^{q-1}(t_{q-1} - t_{q-2})$.

As a taster, Figure 2 shows samples from a covariance function of a switching dynamical latent force model with $D = 1$ and $Q = 3$. Note that while the latent forces (a and c) are discrete, the outputs (b and d) are continuous and have matching gradients at the switching points. The outputs are highly nonstationary. The switching times turn out to be parameters of the covariance function. They can be optimized along with the dynamical system parameters to match the location of the nonstationarities. We now give an overview of the covariance function derivation. Details are provided in the supplementary material.

(a) System 1. Samples from the latent force.

(b) System 1. Samples from the output.

(c) System 2. Samples from the latent force.

(d) System 2. Samples from the output.

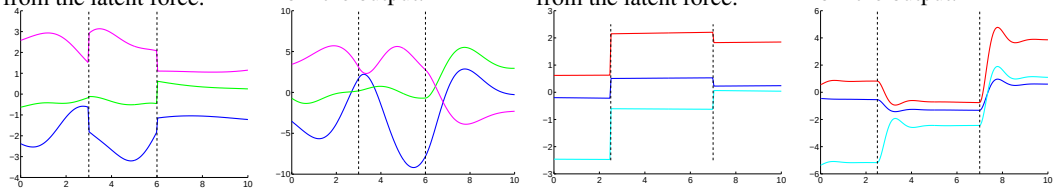


Figure 2: Joint samples of a switching dynamical LFM model with one output, $D = 1$, and three intervals, $Q = 3$, for two different systems. Dashed lines indicate the presence of switching points. While system 2 responds instantaneously to the input force, system 1 delays its reaction due to larger inertia.

In general, we need to compute the covariance $k_{z_d,z_{d'}}(t,t') = \text{cov}[z_d(t), z_{d'}(t')]$ for $z_d(t)$ in time interval (t_{q-1}, t_q) and $z_{d'}(t')$ in time interval $(t_{q'-1}, t_{q'})$. By definition, this covariance follows

$$\text{cov}[z_d(t), z_{d'}(t')] = \text{cov}\left[y_d^q(t-t_{q-1}), y_{d'}^{q'}(t-t_{q'-1}))\right].$$

We assumme independence between the latent forces $u_q(t)$ and independence between the initial conditions $\mathbf{y}_{IC}$ and the latent forces $u_q(t)$.[1] With these conditions, it can be shown[2] that the covariance function[3] for $q = q'$ is given as

$$\begin{aligned}
&c_d^q(t-t_{q-1})c_{d'}^q(t'-t_{q-1})k_{z_d,z_{d'}}(t_{q-1},t_{q-1}) + c_d^q(t-t_{q-1})e_{d'}^q(t'-t_{q-1})k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})\\
&+e_d^q(t-t_{q-1})c_{d'}^q(t'-t_{q-1})k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q-1}) + e_d^q(t-t_{q-1})e_{d'}^q(t'-t_{q-1})k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})\\
&+k_{f_d,f_{d'}}^q(t,t'),
\end{aligned} \tag{3}$$

where

$$\begin{aligned}
k_{z_d,z_{d'}}(t_{q-1},t_{q-1}) &= \text{cov}[y_d^q(t_{q-1})y_{d'}^q(t_{q-1})], \quad k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1}) = \text{cov}[y_d^q(t_{q-1})\dot{y}_{d'}^q(t_{q-1})],\\
k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q-1}) &= \text{cov}[\dot{y}_d^q(t_{q-1})y_{d'}^q(t_{q-1})], \quad k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1}) = \text{cov}[\dot{y}_d^q(t_{q-1})\dot{y}_{d'}^q(t_{q-1})].\\
k_{f_d,f_{d'}}^q(t,t') &= \text{cov}[f_d^q(t-t_{q-1})f_{d'}^q(t'-t_{q-1})].
\end{aligned}$$

In expression (3), $k_{z_d,z_{d'}}(t_{q-1},t_{q-1}) = \text{cov}[y_d^{q-1}(t_{q-1}-t_{q-2}), y_{d'}^{q-1}(t_{q-1}-t_{q-2})]$ and values for $k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})$, $k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q-1})$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})$ can be obtained by similar expressions. The covariance $k_{f_d,f_{d'}}^q(t,t')$ follows a similar expression that the one for $k_{f_d,f_{d'}}(t,t')$ in equation (2), now depending on the covariance $k_{u_{q-1},u_{q-1}}(t,t')$. We will assume that the covariances for the latent forces follow the RBF form, with length-scale ℓ_q.

When $q > q'$, we have to take into account the correlation between the initial conditions $y_d^q(t_{q-1})$, $\dot{y}_d^q(t_{q-1})$ and the latent force $u_{q'-1}(t')$. This correlation appears because of the contribution of $u_{q'-1}(t')$ to the generation of the initial conditions, $y_d^q(t_{q-1})$, $\dot{y}_d^q(t_{q-1})$. It can be shown[4] that the covariance function $\text{cov}[z_d(t), z_{d'}(t')]$ for $q > q'$ follows

$$\begin{aligned}
&c_d^q(t-t_{q-1})c_{d'}^{q'}(t'-t_{q'-1})k_{z_d,z_{d'}}(t_{q-1},t_{q'-1}) + c_d^q(t-t_{q-1})e_{d'}^{q'}(t'-t_{q'-1})k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1})\\
&+e_d^q(t-t_{q-1})c_{d'}^{q'}(t'-t_{q'-1})k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q'-1}) + e_d^q(t-t_{q-1})e_{d'}^{q'}(t'-t_{q'-1})k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1})\\
&+c_d^q(t-t_{q-1})\mathcal{X}_d^1 k_{f_d,f_{d'}}^{q'}(t_{q'-1},t') + c_d^q(t-t_{q-1})\mathcal{X}_d^2 k_{m_d,f_{d'}}^{q'}(t_{q'-1},t')\\
&+e_d^q(t-t_{q-1})\mathcal{X}_d^3 k_{f_d,f_{d'}}^{q'}(t_{q'-1},t') + e_d^q(t-t_{q-1})\mathcal{X}_d^4 k_{m_d,f_{d'}}^{q'}(t_{q'-1},t'),
\end{aligned} \tag{4}$$

where

$$\begin{aligned}
k_{z_d,z_{d'}}(t_{q-1},t_{q'-1}) &= \text{cov}[y_d^q(t_{q-1})y_{d'}^{q'}(t_{q'-1})], \quad k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1}) = \text{cov}[y_d^q(t_{q-1})\dot{y}_{d'}^{q'}(t_{q'-1})],\\
k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q'-1}) &= \text{cov}[\dot{y}_d^q(t_{q-1})y_{d'}^{q'}(t_{q'-1})], \quad k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q'-1}) = \text{cov}[\dot{y}_d^q(t_{q-1})\dot{y}_{d'}^{q'}(t_{q'-1})],\\
k_{m_d,f_{d'}}^q(t,t') &= \text{cov}[m_d^q(t-t_{q-1})f_{d'}^q(t'-t_{q-1})],
\end{aligned}$$

and $\mathcal{X}_d^1$, $\mathcal{X}_d^2$, $\mathcal{X}_d^3$ and $\mathcal{X}_d^4$ are functions of the form $\sum_{n=2}^{q-q'}\prod_{i=2}^{q-q'} x_d^{q-i+1}(t_{q-i+1}-t_{q-i})$, with x_d^{q-i+1} being equal to c_d^{q-i+1}, e_d^{q-i+1}, g_d^{q-i+1} or h_d^{q-i+1}, depending on the values of q and q'.

A similar expression to (4) can be obtained for $q' > q$. Examples of these functions for specific values of q and q' and more details are also given in the supplementary material.

4 Related work

There has been a recent interest in employing Gaussian processes for detection of change points in time series analysis, an area of study that relates to some extent to our model. Some machine learning related papers include [3, 4, 9]. [3, 4] deals specifically with how to construct covariance functions

[1] Derivations of these equations are rather involved. In the supplementary material, section 2, we include a detailed description of how to obtain the equations (3) and (4)

[2] See supplementary material, section 2.2.1.

[3] We will write $f_d^q(t-t_{q-1}, u_{q-1})$ as $f_d^q(t-t_{q-1})$ for notational simplicity.

[4] See supplementary material, section 2.2.2

in the presence of change points (see [3], section 4). The authors propose different alternatives according to the type of change point. From these alternatives, the closest ones to our work appear in subsections 4.2, 4.3 and 4.4. In subsection 4.2, a mechanism to keep continuity in a covariance function when there are two regimes described by different GPs, is proposed. The authors call this covariance continuous conditionally independent covariance function. In our switched latent force model, a more natural option is to use the initial conditions as the way to transit smoothly between different regimes. In subsections 4.3 and 4.4, the authors propose covariances that account for a sudden change in the input scale and a sudden change in the output scale. Both type of changes are automatically included in our model due to the latent force model construction: the changes in the input scale are accounted by the different length-scales of the latent force GP process and the changes in the output scale are accounted by the different sensitivity parameters. Importantly, we also concerned about multiple output systems.

On the other hand, [9] proposes an efficient inference procedure for Bayesian Online Change Point Detection (BOCPD) in which the underlying predictive model (UPM) is a GP. This reference is less concerned about the particular type of change that is represented by the model: in our application scenario, the continuity of the covariance function between two regimes must be assured beforehand.

5 Implementation

In this section, we describe additional details on the implementation, i.e., covariance function, hyperparameters, sparse approximations.

Additional covariance functions. The covariance functions $k_{\dot{z}_d,z_{d'}}(t,t')$, $k_{z_d,\dot{z}_{d'}}(t,t')$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t,t')$ are obtained by taking derivatives of $k_{z_d,z_{d'}}(t,t')$ with respect to t and t' [10].

Estimation of hyperparameters. Given the number of outputs D and the number of intervals Q, we estimate the parameters $\boldsymbol{\theta}$ by maximizing the marginal-likelihood of the joint Gaussian process $\{z_d(t)\}_{d=1}^{D}$ using gradient-descent methods. With a set of input points, $\mathbf{t} = \{t_n\}_{n=1}^{N}$, the marginal-likelihood is given as $p(\mathbf{z}|\boldsymbol{\theta}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{K}_{\mathbf{z},\mathbf{z}} + \boldsymbol{\Sigma})$, where $\mathbf{z} = [\mathbf{z}_1^\top, \ldots, \mathbf{z}_D^\top]^\top$, with $\mathbf{z}_d = [z_d(t_1), \ldots, z_d(t_N)]^\top$, $\mathbf{K}_{\mathbf{z},\mathbf{z}}$ is a $D \times D$ block-partitioned matrix with blocks $\mathbf{K}_{\mathbf{z}_d,\mathbf{z}_{d'}}$. The entries in each of these blocks are evaluated using $k_{z_d,z_{d'}}(t,t')$. Furthermore, $k_{z_d,z_{d'}}(t,t')$ is computed using the expressions (3), and (4), according to the relative values of q and q'.

Efficient approximations Optimizing the marginal likelihood involves the inversion of the matrix $\mathbf{K}_{\mathbf{z},\mathbf{z}}$, inversion that grows with complexity $\mathcal{O}(D^3N^3)$. We use a sparse approximation based on variational methods presented in [2] as a generalization of [11] for multiple output Gaussian processes. The approximations establish a lower bound on the marginal likelihood and reduce computational complexity to $\mathcal{O}(DNK^2)$, being K a reduced number of points used to represent $u(t)$.

6 Experimental results

We now show results with artificial data and data recorded from a robot performing a basic set of actions appearing in table tennis.

6.1 Toy example

Using the model, we generate samples from the GP with covariance function as explained before. In the first experiment, we sample from a model with $D = 2$, $R = 1$ and $Q = 3$, with switching points $t_0 = -1, t_1 = 5$ and $t_2 = 12$. For the outputs, we have $A_1 = A_2 = 0.1$, $C_1 = 0.4, C_2 = 1$, $\kappa_1 = 2, \kappa_2 = 3$. We restrict the latent forces to have the same length-scale value $\ell_0 = \ell_1 = \ell_2 = 1e-3$, but change the values of the sensitivity parameters as $S_{1,1} = 10, S_{2,1} = 1, S_{1,2} = 10, S_{2,2} = 5, S_{1,3} = -10$ and $S_{2,3} = 1$, where the first subindex refers to the output d and the second subindex refers to the force in the interval q. In this first experiment, we wanted to show the ability of the model to detect changes in the sensitivities of the forces, while keeping the length scales equal along the intervals. We sampled 5 times from the model with each output having 500 data points and add some noise with variance equal to ten percent of the variance of each sampled output. In each of the five repetitions, we took $N = 200$ data points for training and the remaining 300 for testing.

		$Q=1$	$Q=2$	$Q=3$	$Q=4$	$Q=5$
1	SMSE	76.27 ± 35.63	14.66 ± 11.74	$\mathbf{0.30\pm0.02}$	$\mathbf{0.31\pm0.03}$	$\mathbf{0.72\pm0.56}$
	MSLL	-0.98 ± 0.46	-1.79 ± 0.26	$\mathbf{-2.90\pm0.03}$	$\mathbf{-2.87\pm0.04}$	$\mathbf{-2.55\pm0.41}$
2	SMSE	7.27 ± 6.88	$\mathbf{1.08\pm0.05}$	$\mathbf{1.10\pm0.05}$	$\mathbf{1.06\pm0.05}$	$\mathbf{1.10\pm0.09}$
	MSLL	-1.79 ± 0.28	$\mathbf{-2.26\pm0.02}$	$\mathbf{-2.25\pm0.02}$	$\mathbf{-2.27\pm0.03}$	$\mathbf{-2.26\pm0.06}$

Table 1: Standarized mean square error (SMSE) and mean standardized log loss (MSLL) using different values of Q for both toy examples. The figures for the SMSE must be multiplied by 10^{-2}. See the text for details.

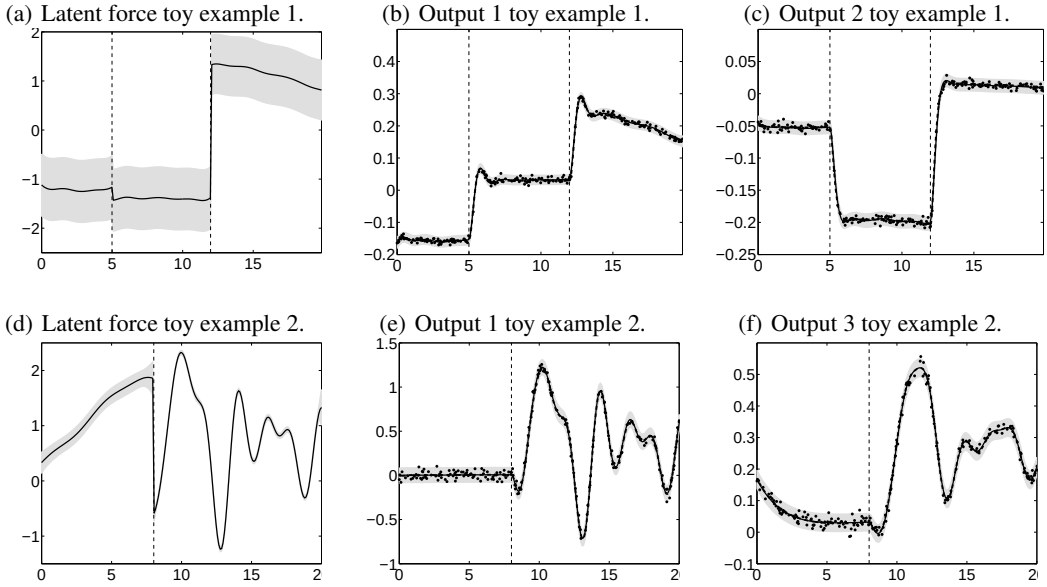


Figure 4: Mean and two standard deviations for the predictions over the latent force and two of the three outputs in the test set. Dashed lines indicate the final value of the swithcing points after optimization. Dots indicate training data.

Optimization of the hyperparameters (including t_1 and t_2) is done by maximization of the marginal likelihood through scaled conjugate gradient. We train models for $Q = 1, 2, 3, 4$ and 5 and measure the mean standarized log loss (MSLL) and the mean standarized mean square error (SMSE) [8] over the test set for each value of Q. Table 1, first two rows, show the corresponding average results over the 5 repetitions together with one standard deviation. Notice that for $Q = 3$, the model gets by the first time the best performance, performance that repeats again for $Q = 4$. The SMSE performance remains approximately equal for values of Q greater than 3. Figures 4(a), 4(b) and 4(c) shows the kind of predictions made by the model for $Q = 3$.

Figure 3: Data collection was performed using a Barrett WAM robot as haptic input device.

We generate also a different toy example, in which the length-scales of the intervals are different. For the second toy experiment, we assume $D = 3$, $Q = 2$ and switching points $t_0 = -2$ and $t_1 = 8$. The parameters of the outputs are $A_1 = A_2 = A_3 = 0.1$, $C_1 = 2, C_2 = 3, C_3 = 0.5$, $\kappa_1 = 0.4, \kappa_2 = 1, \kappa_3 = 1$ and length scales $\ell_0 = 1e-3$ and $\ell_1 = 1$. Sensitivities in this case are $S_{1,1} = 1, S_{2,1} = 5, S_{3,1} = 1, S_{1,2} = 5, S_{2,2} = 1$ and $S_{3,2} = 1$. We follow the same evaluation setup as in toy example 1. Table 1, last two rows, show the performance again in terms of MLSS and SMSE. We see that for values of $Q > 2$, the MLSS and SMSE remain similar. In figures 4(d), 4(e) and 4(f), the inferred latent force and the predictions made for two of the three outputs.

6.2 Segmentation of human movement data for robot imitation learning

In this section, we evaluate the feasibility of the model for motion segmentation with possible applications in the analysis of human movement data and imitation learning. To do so, we had a human teacher take the robot by the hand and have him demonstrate striking movements in a cooperative game of table tennis with another human being as shown in Figure 3. We recorded joint positions,

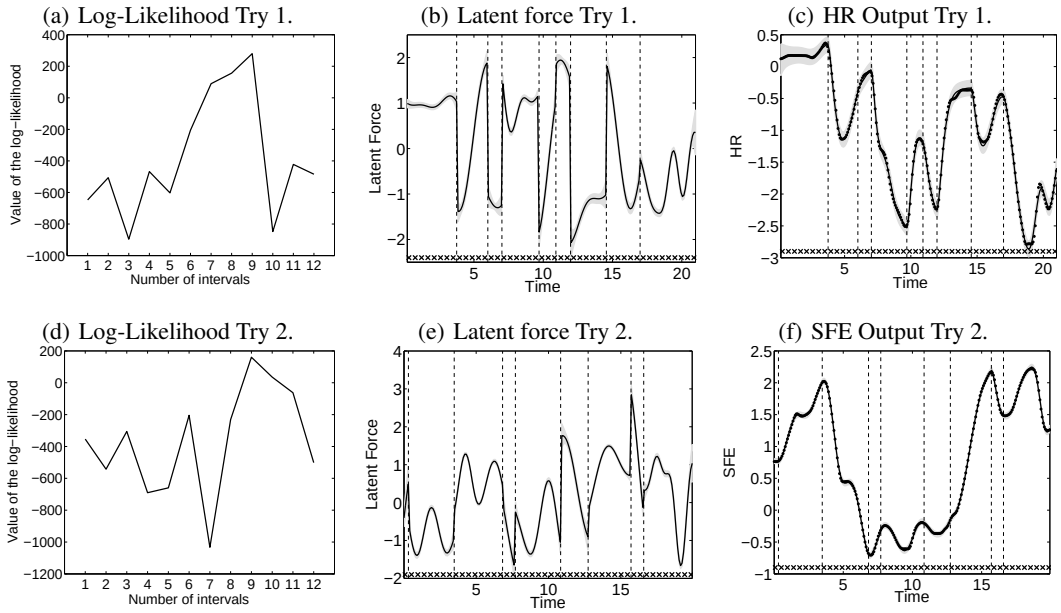


Figure 5: Employing the switching dynamical LFM model on the human movement data collected as in Fig.3 leads to plausible segmentations of the demonstrated trajectories. The first row corresponds to the log-likelihood, latent force and one of four outputs for trial one. Second row shows the same quantities for trial two. Crosses in the bottom of the figure refer to the number of points used for the approximation of the Gaussian process, in this case $K = 50$.

angular velocities, and angular acceleration of the robot for two independent trials of the same table tennis exercise. For each trial, we selected four output positions and train several models for different values of Q, including the latent force model without switches ($Q = 1$). We evaluate the quality of the segmentation in terms of the log-likelihood. Figure 5 shows the log-likelihood, the inferred latent force and one output for trial one (first row) and the corresponding quantities for trial two (second row). Figures 5(a) and 5(d) show peaks for the log-likelihood at $Q = 9$ for trial one and $Q = 10$ for trial two. As the movement has few gaps and the data has several output dimensions, it is hard even for a human being to detect the transitions between movements (unless it is visualized as in a movie). Nevertheless, the model found a maximum for the log-likelihood at the correct instances in time where the human transits between two movements. At these instances the human usually reacts due to an external stimulus with a large jerk causing a jump in the forces. As a result, we obtained not only a segmentation of the movement but also a generative model for table tennis striking movements.

7 Conclusion

We have introduced a new probabilistic model that develops the latent force modeling framework with switched Gaussian processes. This allows for discontinuities in the latent space of forces. We have shown the application of the model in toy examples and on a real world robot problem, in which we were interested in finding and representing striking movements. Other applications of the switching latent force model that we envisage include modeling human motion capture data using the second order ODE and a first order ODE for modeling of complex circuits in biological networks. To find the order of the model, this is, the number of intervals, we have used cross-validation. Future work includes proposing a less expensive model selection criteria.

Acknowledgments

MA and NL are very grateful for support from a Google Research Award "Mechanistically Inspired Convolution Processes for Learning" and the EPSRC Grant No EP/F005687/1 "Gaussian Processes for Systems Identification with Applications in Systems Biology". MA also thanks PASCAL2 Internal Visiting Programme. We also thank to three anonymous reviewers for their helpful comments.

References

[1] Mauricio Álvarez, David Luengo, and Neil D. Lawrence. Latent Force Models. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pages 9–16, Clearwater Beach, Florida, 16-18 April 2009. JMLR W&CP 5.

[2] Mauricio A. Álvarez, David Luengo, Michalis K. Titsias, and Neil D. Lawrence. Efficient multioutput Gaussian processes through variational inducing kernels. In *JMLR: W&CP 9*, pages 25–32, 2010.

[3] Roman Garnett, Michael A. Osborne, Steven Reece, Alex Rogers, and Stephen J. Roberts. Sequential Bayesian prediction in the presence of changepoints and faults. *The Computer Journal*, 2010. Advance Access published February 1, 2010.

[4] Roman Garnett, Michael A. Osborne, and Stephen J. Roberts. Sequential Bayesian prediction in the presence of changepoints. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 345–352, 2009.

[5] Antti Honkela, Charles Girardot, E. Hilary Gustafson, Ya-Hsin Liu, Eileen E. M. Furlong, Neil D. Lawrence, and Magnus Rattray. Model-based method for transcription factor target identification with limited data. *PNAS*, 107(17):7793–7798, 2010.

[6] A. Ijspeert, J. Nakanishi, and S. Schaal. Learning attractor landscapes for learning motor primitives. In *Advances in Neural Information Processing Systems 15*, 2003.

[7] T. Oyama, Y. Uno, and S. Hosoe. Analysis of variability of human reaching movements based on the similarity preservation of arm trajectories. In *International Conference on Neural Information Processing (ICONIP)*, pages 923–932, 2007.

[8] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006.

[9] Yunus Saatçi, Ryan Turner, and Carl Edward Rasmussen. Gaussian Process change point models. In *Proceedings of the 27th Annual International Conference on Machine Learning*, pages 927–934, 2010.

[10] E. Solak, R. Murray-Smith W. E. Leithead, D. J. Leith, and C. E. Rasmussen. Derivative observations in Gaussian process models of dynamic systems. In Sue Becker, Sebastian Thrun, and Klaus Obermayer, editors, *NIPS*, volume 15, pages 1033–1040, Cambridge, MA, 2003. MIT Press.

[11] Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In *JMLR: W&CP 5*, pages 567–574, 2009.

Supplementary material
Switched Latent Force Models for Movement Segmentation

Mauricio A. Álvarez [1], **Jan Peters** [2], **Bernhard Schölkopf** [2], **Neil D. Lawrence** [3,4]
[1] School of Computer Science, University of Manchester, Manchester, UK M13 9PL
[2] Max Planck Institute for Biological Cybernetics, Tübingen, Germany 72076
[3] School of Computer Science, University of Sheffield, Sheffield, UK S1 4DP
[4] The Sheffield Institute for Translational Neuroscience, Sheffield, UK S10 2HQ

1 Introduction

The motor primitive idea is similar to the latent force model one. We want to use a set of templates for basic motions in order to generate more complex ones. The analogy we can think of is the generation of speech, in which phonemes are used to generate words and sentences.

Motor primitive model

Motor primitives employ the concept of autonomous dynamical system in which the independent variable is first parameterized by a first order homogenous dynamical system. The output of this system is used as the independent variable of the inducing force of a second order differential equation [2]. The first system is known as the *canonical system* and its form depends on the type of movement that is to be represented: point attractive and limit cycle behaviors are the two most basic behaviors of nonlinear dynamical systems. In motor control these correspond to discrete and rythmic movements.

Latent force model

The latent force model was first introduced in [1]. A set of coupled second order ordinary differential equations was employed for human-balancing movement representation. Here we only review the basic form for the covariance function in the Gaussian process formulation of the Latent force model. More details and applications can be found in [1].

A set of D outputs $\{f_d(t)\}_{d=1}^D$ (where each of them describes the relative position of a particle wrt to a set of reference points in a spring-damper-mass system) is represented by a Gaussian process with covariance function,

$$k_{f_d f_{d'}}(t,t') = \sum_{q=1}^{Q} \frac{S_{qd} S_{qd'} \sqrt{\pi \ell_q^2}}{8 A_d A_{d'} \omega_d \omega_{d'}} k^{(q)}_{f_d f_{d'}}(t,t'),$$

with A_d the mass of system d, ω_d the angular frequency, S_{qd} the relative strength of latent force q over output d, ℓ_q the length-scale of the RBF covariance for the Gaussian process that describes the latent force q and $k^{(q)}_{f_d f_{d'}}(t,t')$, the cross-covariance between the d-th and d'-th outputs under the effect of the q-th latent force, and is given by

$$\begin{aligned} k^{(q)}_{f_d f_{d'}}(t,t') = {} & h^q(\widetilde{\gamma}_{d'}, \gamma_d, t, t') + h^q(\gamma_d, \widetilde{\gamma}_{d'}, t', t) + \; h^q(\gamma_{d'}, \widetilde{\gamma}_d, t, t') + h^q(\widetilde{\gamma}_d, \gamma_{d'}, t', t) \\ & - \; h^q(\widetilde{\gamma}_{d'}, \widetilde{\gamma}_d, t, t') - h^q(\widetilde{\gamma}_d, \widetilde{\gamma}_{d'}, t', t) - \; h^q(\gamma_{d'}, \gamma_d, t, t') - h^q(\gamma_d, \gamma_{d'}, t', t), \end{aligned}$$

where $\gamma_d = \alpha_d + j\omega_d$, $\widetilde{\gamma}_d = \alpha_d - j\omega_d$, and

$$h^q(\gamma_{d'}, \gamma_d, t, t') = \frac{1}{\gamma_d + \gamma_{d'}} \left[\Upsilon^q(\gamma_{d'}, t', t) - \exp(-\gamma_d t)\Upsilon^q(\gamma_{d'}, t', 0)\right].$$

$$\begin{aligned}\Upsilon^q(\gamma_{d'}, t, t') = &\underbrace{2\exp\left(\frac{\ell_q^2\gamma_{d'}^2}{4}\right)\exp(-\gamma_{d'}(t-t'))}_{\psi^q(\gamma_{d'},t,t')} - \underbrace{\exp\left(-\frac{(t-t')^2}{\ell_q^2}\right)\mathrm{w}(jz(t,t'))}_{\upsilon^q(\gamma_{d'},t,t')} \\ &- \underbrace{\exp\left(-\frac{(t')^2}{\ell_q^2}\right)\exp(-\gamma_{d'}t)\mathrm{w}(-jz(0,t'))}_{\varphi^q(\gamma_{d'},t,t')} \\ &= \psi^q(\gamma_{d'}, t, t') - \upsilon^q(\gamma_{d'}, t, t') - \varphi^q(\gamma_{d'}, t, t'),\end{aligned}$$

and $z(t,t') = (t-t')/\ell_q - (\ell_q\gamma_{d'})/2$. Note that $z(t,t') \in \mathbb{C}$, and $\mathrm{w}(jz)$ in the above equation, for $z \in \mathbb{C}$, denotes Faddeeva's function $\mathrm{w}(jz) = \exp(z^2)\mathrm{erfc}(z)$, where $\mathrm{erfc}(z)$ is the complex version of the complementary error function, $\mathrm{erfc}(z) = 1 - \mathrm{erf}(z) = \frac{2}{\sqrt{\pi}}\int_z^\infty \exp(-v^2)\mathrm{d}v$. Faddeeva's function is usually considered the complex equivalent of the error function, since $|\mathrm{w}(jz)|$ is bounded whenever the imaginary part of jz is greater or equal than zero, and is the key to achieving a good numerical stability when computing $\Upsilon^q(\gamma_{d'}, t, t')$ and its gradients.

2 Switching forces

Figure 1 shows a cartoon representation of output $z_d(t)$ switching its behavior between points t_0, t_1, t_2 and t_3. For each interval (t_{i-1}, t_i), only the latent force $u_{i-1}(t)$ is active.

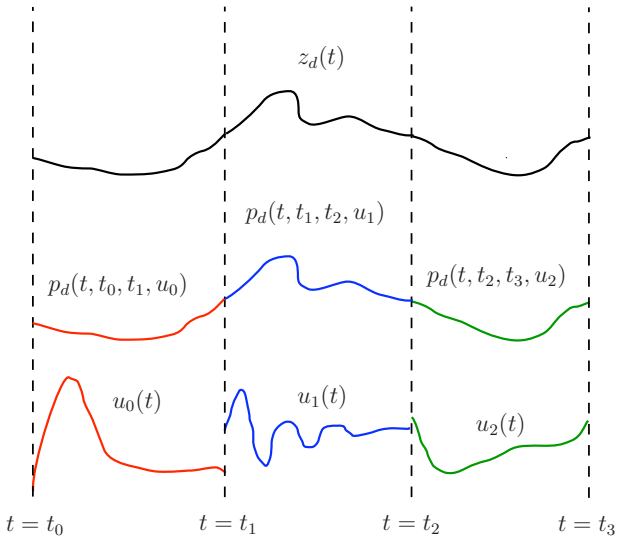


Figure 1: A pictorial representation of the switching scenario for $z_d(t)$

2.1 Definition of the model

Taking into account the initial conditions, the solution to the second order model is given as

$$\begin{aligned}y_d(t) = &y_d(0)e^{-\alpha_d t}\left[\cos(\omega_d t) + \frac{\alpha_d}{\omega_d}\sin(\omega_d t)\right] + \dot{y}_d(0)\left[\frac{e^{-\alpha_d t}}{\omega_d}\sin(\omega_d t)\right] \\ &+ \frac{S_d}{A_d\omega_d}\int_0^t e^{-\alpha_d(t-\tau)}\sin[(t-\tau)\omega_d]u(\tau)\mathrm{d}\tau,\end{aligned}$$

where $y_d(0)$ and $\dot{y}_d(0)$ are the initial conditions. This is the basic equation we need to use to express the covariance function for the switching model. The uncertainty in this model is due to the latent force $u(t)$ and the initial conditions $y_d(0)$ and $\dot{y}_d(0)$. For simplicity, we write the above equation as

$$y_d(t) = c_d(t)y_d(0) + e_d(t)\dot{y}_d(0) + f_d(t), \tag{1}$$

with

$$\begin{aligned}
c_d(t) &= e^{-\alpha_d t}\left[\cos(\omega_d t) + \frac{\alpha_d}{\omega_d}\sin(\omega_d t)\right] \\
e_d(t) &= \left[\frac{e^{-\alpha_d t}}{\omega_d}\sin(\omega_d t)\right] \\
f_d(t) &= \frac{S_d}{A_d\omega_d}\int_0^t e^{-\alpha_d(t-\tau)}\sin[(t-\tau)\omega_d]u(\tau)\mathrm{d}\tau = \int_0^t G_d(t-\tau)u(\tau)\mathrm{d}\tau.
\end{aligned}$$

We'll need also the velocity $v_d(t)$, which is is given as

$$v_d(t) = \frac{\mathrm{d}y_d(t)}{\mathrm{d}t} = g_d(t)y_d(0) + h_d(t)\dot{y}_d(0) + m_d(t), \tag{2}$$

with

$$\begin{aligned}
g_d(t) &= \frac{\mathrm{d}c_d(t)}{\mathrm{d}t} = -e^{-\alpha_d t}\sin(\omega_d t)\left(\frac{\alpha_d^2}{\omega_d} + \omega_d\right) \\
h_d(t) &= \frac{\mathrm{d}e_d(t)}{\mathrm{d}t} = -e^{-\alpha_d t}\left[\frac{\alpha_d}{\omega_d}\sin(\omega_d t) - \cos(\omega_d t)\right] \\
m_d(t) &= \frac{\mathrm{d}}{\mathrm{d}t}\left(\int_0^t G_d(t-\tau)u(\tau)\mathrm{d}\tau\right).
\end{aligned}$$

Furthermore, we also need the acceleration, given as

$$a_d(t) = \frac{\mathrm{d}v_d(t)}{\mathrm{d}t} = r_d(t)y_d(0) + b_d(t)\dot{y}_d(0) + w_d(t), \tag{3}$$

with

$$\begin{aligned}
r_d(t) &= \frac{\mathrm{d}h_d(t)}{\mathrm{d}t} = e^{-\alpha_d t}\left(\frac{\alpha_d^2}{\omega_d} + \omega_d\right)\left[\alpha_d\sin(\omega_d t) - \omega_d\cos(\omega_d t)\right] \\
b_d(t) &= \frac{\mathrm{d}g_d(t)}{\mathrm{d}t} = e^{-\alpha_d t}\left[\left(\frac{\alpha_d^2}{\omega_d} - \omega_d\right)\sin(\omega_d t) - 2\alpha_d\cos(\omega_d t)\right] \\
w_d(t) &= \frac{\mathrm{d}^2}{\mathrm{d}t^2}\left(\int_0^t G_d(t-\tau)u(\tau)\mathrm{d}\tau\right).
\end{aligned}$$

The input space is divided in non-overlapping intervals $[t_{q-1}, t_q]_{q=1}^Q$ and for each one of these intervals, only one force $u_{q-1}(t)$ out of Q forces is active, this is, there are $\{u_{q-1}\}_{q=1}^Q$ forces. The force $u_{q-1}(t)$ is activated after time t_{q-1} and desactivated after time t_q. We can use the basic model in the equation before to describe the contribution to the output due to the sequential activation of these forces. An output $z_d(t)$ at a particular time instant t, in the interval (t_{q-1}, t_q), is expressed as

$$z_d(t, t_{q-1}, t_q) = p_d(t, t_{q-1}, t_q, u_{q-1}), \quad \text{for} \quad 1 \leq d \leq D,$$

where $p_d(t, t_{q-1}, t_q, u_{q-1})$ uses the model for $y_d(t)$ in equation (1) as

$$\begin{aligned}
p_d(t, t_{q-1}, t_q, u_{q-1}) = y_d(t)\big|_{t_{q-1}} &= c_d(t-t_{q-1})y_d(t_{q-1}) + e_d(t-t_{q-1})\dot{y}_d(t_{q-1}) \\
&\quad + f_d(t, t_{q-1}, t_q, u_{q-1}).
\end{aligned}$$

Notice that there are as many intervals $\{(t_{q-1}, t_q)\}_{q=1}^Q$ as latent forces $\{u_q(t)\}_{q=1}^Q$. For simplicity, we write $z_d(t, t_{q-1}, t_q)$ as $z_d(t)$. In the above equation, $y_d(t)\big|_{t_{q-1}}$ expresses that $y_d(t)$ has to be evaluated with the initial condition specified at t_{q-1} and

$$f_d(t, t_{q-1}, t_q, u_{q-1}) = \int_0^{t-t_{q-1}} G_d(t - t_{q-1} - \tau)u_{q-1}(\tau)\mathrm{d}\tau. \tag{4}$$

Expression $f_d(t, t_{q-1}, t_q, u_{q-1})$ is a function of four arguments: the first argument, t, refers to the independent variable inside the kernel smoothing function $G_d(t-\tau)$ and in the upper limit of the convolution transform; the second argument, t_{q-1}, and third argument t_q specify the lower and upper limits of the time interval for which the convolution is being computed and the fourth argument, u_{q-1}, specifies the latent force acting in this interval. Additionally, we define a similar function for the velocity $\dot{z}_d(t)$ as

$$\dot{z}_d(t, t_q, t_{q-1}) = \xi_d(t, t_{q-1}, t_q, u_{q-1}), \quad \text{for} \quad 1 \leq d \leq D,$$

where

$$\begin{aligned}\xi_d(t, t_{q-1}, t_q, u_{q-1}) = v_d(t)\big|_{t_{q-1}} &= g_d(t-t_{q-1})y_d(t_{q-1}) + h_d(t-t_{q-1})\dot{y}_d(t_{q-1}) \\ &\quad + m_d(t, t, t_{q-1}, t_q, u_{q-1}),\end{aligned}$$

and $m_d(t, t_{q-1}, t_q, u_{q-1})$ follows

$$m_d(t, t_{q-1}, t_q, u_{q-1}) = \frac{\mathrm{d}}{\mathrm{d}t}\left(\int_0^{t-t_{q-1}} G_d(t-t_{q-1}-\tau)u_{q-1}(\tau)\mathrm{d}\tau\right). \tag{5}$$

Again, for simplicity, we write $\dot{z}_d(t, t_q, t_{q-1})$ as $\dot{z}_d(t)$. The initial conditions $y_d(t_{q-1})$ and $\dot{y}_d(t_{q-1})$ can be defined again in terms of $z_d(t)$ and $\dot{z}_d(t)$

$$\begin{aligned}y_d(t_{q-1}) &= z_d(t_{q-1}) = p_d(t_{q-1}, t_{q-2}, t_{q-1}, u_{q-2}), \\ \dot{y}_d(t_{q-1}) &= \dot{z}_d(t_{q-1}) = \xi_d(t_{q-1}, t_{q-2}, t_{q-1}, u_{q-2}).\end{aligned}$$

Without loss of generality, we assume that the initial conditions at $t = t_0$ for all d, are parameters of the model. This is $y_d(t_0)$ and $\dot{y}_d(t_0)$ are parameters that need to be estimated. Eventually, we might need to put a prior over them. A similar expression is obtained for the acceleration $\ddot{z}_d(t)$.

Example 1. Suppose we have $Q = 3$ as in figure 1. Then, the outputs $z_d(t)$ will be given as $z_d(t, t_0, t_1) = p_d(t, t_0, t_1, u_0)$, $z_d(t, t_1, t_2) = p_d(t, t_1, t_2, u_1)$ and $z_d(t, t_2, t_3) = p_d(t, t_2, t_3, u_2)$. Equally, the velocities $\dot{z}_d(t)$ will follow $\dot{z}_d(t, t_0, t_1) = \xi_d(t, t_0, t_1, u_0)$, $\dot{z}_d(t, t_1, t_2) = \xi_d(t, t_1, t_2, u_1)$ and $\dot{z}_d(t, t_2, t_3) = \xi_d(t, t_2, t_3, u_2)$. We also have the initial conditions. For t_0, the initial conditions are parameters $y_d(t_0)$ and $\dot{y}_d(t_0)$. For the intervals starting at t_1 and t_2, the initial conditions are given as $y_d(t_1) = p_d(t_1, t_0, t_1, u_0)$ and $y_d(t_2) = p_d(t_2, t_1, t_2, u_1)$. And for the velocities $\dot{y}_d(t_1) = \xi_d(t_1, t_1, t_0, t_1, u_0)$ and $\dot{y}_d(t_2) = \xi_d(t_2, t_1, t_2, u_1)$.

2.2 Covariance for the outputs

In general, we need to compute the covariance $\mathrm{cov}[z_d(t), z_{d'}(t')]$ for every time interval (t_{q-1}, t_q) and for intervals (t_{q-1}, t_q) and $(t_{q'-1}, t'_q)$. The covariance $\mathrm{cov}[z_d(t), z_{d'}(t')]$ for time interval (t_{q-1}, t_q) is given as

$$\mathrm{cov}[z_d(t), z_{d'}(t')] = \mathrm{cov}\left[p_d(t, t_{q-1}, t_q, u_{q-1}), p_{d'}(t', t_{q-1}, t_q, u_{q-1})\right]. \tag{6}$$

And the covariance $\mathrm{cov}[z_d(t), z_{d'}(t')]$ for time intervals (t_{q-1}, t_q) and $(t_{q'-1}, t'_q)$ is given as

$$\mathrm{cov}[z_d(t), z_{d'}(t')] = \mathrm{cov}\left[p_d(t, t, t_{q-1}, t_q, u_{q-1}), p_{d'}(t', t_{q'-1}, t_{q'}, u_{q'-1})\right]. \tag{7}$$

2.2.1 Covariance for interval (t_{q-1}, t_q)

The covariance in equation (6), follows

$$\begin{aligned}&\mathrm{cov}\{[c_d(t-t_{q-1})y_d(t_{q-1}) + e_d(t-t_{q-1})\dot{y}_d(t_{q-1}) + f_d(t, t, t_{q-1}, t_q, u_{q-1})] \\ &\quad [c_{d'}(t'-t_{q-1})y_{d'}(t_{q-1}) + e_{d'}(t'-t_{q-1})\dot{y}_{d'}(t_{q-1}) + f_{d'}(t', t', t_{q-1}, t_q, u_{q-1})\} \\ &= c_d(t-t_{q-1})c_{d'}(t'-t_{q-1})\,\mathrm{cov}\{y_d(t_{q-1})y_{d'}(t_{q-1})\} + c_d(t-t_{q-1})e_{d'}(t'-t_{q-1})\,\mathrm{cov}\{y_d(t_{q-1})\dot{y}_{d'}(t_{q-1})\} \\ &+c_d(t-t_{q-1})\,\mathrm{cov}\{y_d(t_{q-1})f_{d'}(t', t', t_{q-1}, t_q, u_{q-1})\} + e_d(t-t_{q-1})c_{d'}(t'-t_{q-1})\,\mathrm{cov}\{\dot{y}_d(t_{q-1})y_{d'}(t_{q-1})\} \\ &+e_d(t-t_{q-1})e_{d'}(t'-t_{q-1})\,\mathrm{cov}\{\dot{y}_d(t_{q-1})\dot{y}_{d'}(t_{q-1})\} + e_d(t-t_{q-1})\,\mathrm{cov}\{\dot{y}_d(t_{q-1})f_{d'}(t', t', t_{q-1}, t_q, u_{q-1})\} \\ &+c_{d'}(t'-t_{q-1})\,\mathrm{cov}\{f_d(t, t, t_{q-1}, t_q, u_{q-1})y_{d'}(t_{q-1})\} + e_{d'}(t'-t_{q-1})\,\mathrm{cov}\{f_d(t, t, t_{q-1}, t_q, u_{q-1})\dot{y}_{d'}(t_{q-1})\} \\ &\quad + \mathrm{cov}\{f_d(t, t, t_{q-1}, t_q, u_{q-1})f_{d'}(t', t', t_{q-1}, t_q, u_{q-1})\}.\end{aligned}$$

The terms $\text{cov}\{y_d(t_{q-1})y_{d'}(t_{q-1})\}$, $\text{cov}\{y_d(t_{q-1})\dot{y}_{d'}(t_{q-1})\}$, $\text{cov}\{\dot{y}_d(t_{q-1})y_{d'}(t_{q-1})\}$ and $\text{cov}\{\dot{y}_d(t_{q-1})\dot{y}_{d'}(t_{q-1})\}$ are obtained from the covariance already computed. These terms are equivalent as $k_{z_d,z_{d'}}(t_{q-1},t_{q-1}) = \text{cov}\{y_d(t_{q-1})y_{d'}(t_{q-1})\}$, $k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1}) = \text{cov}\{y_d(t_{q-1})\dot{y}_{d'}(t_{q-1})\}$, $k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q-1}) = \text{cov}\{\dot{y}_d(t_{q-1})y_{d'}(t_{q-1})\}$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1}) = \text{cov}\{\dot{y}_d(t_{q-1})\dot{y}_{d'}(t_{q-1})\}$. The expressions $\text{cov}\{y_d(t_{q-1})f_{d'}(t',t',t_{q-1},t_q,u_{q-1})\}$, $\text{cov}\{\dot{y}_d(t_{q-1})f_{d'}(t',t',t_{q-1},t_q,u_{q-1})\}$, $\text{cov}\{f_d(t,t,t_{q-1},t_q,u_{q-1})y_{d'}(t_{q-1})\}$ and $\text{cov}\{f_d(t,t,t_{q-1},t_q,u_{q-1})\dot{y}_{d'}(t_{q-1})\}$ are zero. This can be seen from the fact that terms like $y_d(t_{q-1})$ are a obtained as a result of terms $y_d(t_{k-1})$ and $f_d(t_{k-1},t_{k-1},t_k,u_k)$, for $k<q$, and the covariance between those terms with $f_d(t,t_{q-1},t_q,u_{q-1})$ is zero. Finally, the term $\text{cov}\{f_d(t,t_{q-1},t_q,u_{q-1})f_{d'}(t',t_{q-1},t_q,u_{q-1})\}$ is denoted as $k^{(q-1)}_{f_d,f_{d'}}(t,t')$.

In this way the covariance $\text{cov}\left[p_d(t,t,t_{q-1},t_q,u_{q-1}),p_{d'}(t',t',t_{q-1},t_q,u_{q-1})\right]$ is equal to

$$\begin{aligned}
&c_d(t-t_{q-1})c_{d'}(t'-t_{q-1})k_{z_d,z_{d'}}(t_{q-1},t_{q-1}) + c_d(t-t_{q-1})e_{d'}(t'-t_{q-1})k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})\\
&+e_d(t-t_{q-1})c_{d'}(t'-t_{q-1})k_{\dot{z}_d,z_d}(t_{q-1},t_{q-1}) + e_d(t-t_{q-1})e_{d'}(t'-t_{q-1})k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})\\
&\qquad + k^{(q-1)}_{f_d,f_{d'}}(t,t'). \qquad (8)
\end{aligned}$$

The term $k_{z_d,z_{d'}}(t_{q-1},t_{q-1})$ is equal to $\text{cov}[z_d(t_{q-1},t_{q-2},t_{q-1}),z_{d'}(t_{q-1},t_{q-2},t_{q-1})]$ and analog expressions are obtained for $k_{z_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})$, $k_{\dot{z}_d,z_{d'}}(t_{q-1},t_{q-1})$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1},t_{q-1})$.

Example 1 (Continued). We continue with the example in figure 1. We need to compute the covariance $k_{z_d,z_{d'}}(t,t')$ in the intervals $(t_0,t_1]$, $(t_1,t_2]$ and $(t_2,t_3]$. For the covariance in the interval $(t_0,t_1]$, we have

$$\begin{aligned}
\text{cov}[z_d(t),z_{d'}(t')] &= \text{cov}[p_d(t,t_0,t_1,u_0),p_{d'}(t,t_0,t_1,u_0)]\\
&= c_d(t-t_0)c_{d'}(t'-t_0)k_{z_d,z_{d'}}(t_0,t_0) + c_d(t-t_0)e_{d'}(t'-t_0)k_{z_d,\dot{z}_{d'}}(t_0,t_0)\\
&+ e_d(t-t_0)c_{d'}(t'-t_0)k_{\dot{z}_d,z_d}(t_0,t_0) + e_d(t-t_0)e_{d'}(t'-t_0)k_{\dot{z}_d,\dot{z}_{d'}}(t_0,t_0)\\
&+ k^{(0)}_{f_d,f_{d'}}(t,t').
\end{aligned}$$

We assume the terms $k_{z_d,z_{d'}}(t_0,t_0)$, $k_{z_d,\dot{z}_{d'}}(t_0,t_0)$, $k_{\dot{z}_d,z_d}(t_0,t_0)$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_0,t_0)$ are parameters that have to be estimated in the inference process. We also have access to $\text{cov}[z_d(t),\dot{z}_{d'}(t')]$, $\text{cov}[\dot{z}_d(t),z_{d'}(t')]$ and $\text{cov}[\dot{z}_d(t),\dot{z}_{d'}(t')]$. With these expressions we compute $k_{z_d,z_{d'}}(t_1,t_1) = \text{cov}[z_d(t_1),z_{d'}(t_1)]$, $k_{z_d,\dot{z}_{d'}}(t_1,t_1) = \text{cov}[z_d(t_1),\dot{z}_{d'}(t_1)]$, $k_{\dot{z}_d,z_d}(t_1,t_1) = \text{cov}[\dot{z}_d(t_1),z_{d'}(t_1)]$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_1,t_1) = \text{cov}[\dot{z}_d(t_1),\dot{z}_{d'}(t_1)]$, that are needed to compute the covariance in the next interval.

For the covariance in the interval $(t_1,t_2]$, we have

$$\text{cov}[z_d(t),z_{d'}(t')] = \text{cov}[p_d(t,t_1,t_2,u_1),p_{d'}(t',t_1,t_2,u_1)], \qquad (9)$$

which follows the same form that equation (8)

$$\begin{aligned}
&c_d(t-t_1)c_{d'}(t'-t_1)k_{z_d,z_{d'}}(t_1,t_1) + c_d(t-t_1)e_{d'}(t'-t_1)k_{z_d,\dot{z}_{d'}}(t_1,t_1)\\
&+ e_d(t-t_1)c_{d'}(t'-t_1)k_{\dot{z}_d,z_d}(t_1,t_1) + e_d(t-t_1)e_{d'}(t'-t_1)k_{\dot{z}_d,\dot{z}_{d'}}(t_1,t_1) + k^{(1)}_{f_d,f_{d'}}(t,t').
\end{aligned}$$

With the final expression for $\text{cov}[z_d(t,t_1,t_2),z_{d'}(t',t_1,t_2)]$, we compute $k_{z_d,z_{d'}}(t_2,t_2) = \text{cov}[z_d(t_2),z_{d'}(t_2)]$, $k_{z_d,\dot{z}_{d'}}(t_2,t_2) = \text{cov}[z_d(t_2),\dot{z}_{d'}(t_2)]$, $k_{\dot{z}_d,z_d}(t_2,t_2) = \text{cov}[\dot{z}_d(t_2),z_{d'}(t_2)]$ and $k_{\dot{z}_d,\dot{z}_{d'}}(t_2,t_2) = \text{cov}[\dot{z}_d(t_2),\dot{z}_{d'}(t_2)]$, that are needed to compute the covariance in the next interval.

We finally need the covariance for the interval $(t_2,t_3]$. This covariance is computed as

$$\text{cov}[z_d(t),z_{d'}(t')] = \text{cov}[p_d(t,t_2,t_3,u_2),p_{d'}(t',t_2,t_3,u_2)], \qquad (10)$$

given as

$$\begin{aligned}
&c_d(t-t_2)c_{d'}(t'-t_2)k_{z_d,z_{d'}}(t_2,t_2) + c_d(t-t_2)e_{d'}(t'-t_2)k_{z_d,\dot{z}_{d'}}(t_2,t_2)\\
&+ e_d(t-t_2)c_{d'}(t'-t_2)k_{\dot{z}_d,z_d}(t_2,t_2) + e_d(t-t_2)e_{d'}(t'-t_2)k_{\dot{z}_d,\dot{z}_{d'}}(t_2,t_2) + k^{(2)}_{f_d,f_{d'}}(t,t').
\end{aligned}$$

2.2.2 Covariance for intervals (t_{q-1}, t_q) and $(t_{q'-1}, t'_q)$

For the covariance in equation (7), we have two regimes

1. $q > q'$.
2. $q < q'$.

The case for which $q = q'$ was analized in the subsection before this one. We are interested in computing the term $\operatorname{cov}\left[p_d(t,t,t_{q-1},t_q,u_{q-1}), p_{d'}(t',t',t_{q'-1},t_{q'},u_{q'-1})\right]$, for $q > q'$ and $q < q'$. For $q > q'$, we have

$$
\begin{aligned}
&c_d(t-t_{q-1})c_{d'}(t'-t_{q'-1})\operatorname{cov}\{y_d(t_{q-1})y_{d'}(t_{q'-1})\} + c_d(t-t_{q-1})e_{d'}(t'-t_{q'-1})\operatorname{cov}\{y_d(t_{q-1})\dot{y}_{d'}(t_{q'-1})\}\\
&+c_d(t-t_{q-1})\operatorname{cov}\{y_d(t_{q-1})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\} + e_d(t-t_{q-1})c_{d'}(t'-t_{q'-1})\operatorname{cov}\{\dot{y}_d(t_{q-1})y_{d'}(t_{q'-1})\}\\
&+e_d(t-t_{q-1})e_{d'}(t'-t_{q'-1})\operatorname{cov}\{\dot{y}_d(t_{q-1})\dot{y}_{d'}(t_{q'-1})\} + e_d(t-t_{q-1})\operatorname{cov}\{\dot{y}_d(t_{q-1})f_{d'}(t',t_{q'-1},t_q,u_{q'-1})\}\\
&+c_{d'}(t'-t_{q-1})\operatorname{cov}\{f_d(t,t_{q-1},t_q,u_{q-1})y_{d'}(t_{q'-1})\} + e_{d'}(t'-t_{q-1})\operatorname{cov}\{f_d(t,t_{q-1},t_q,u_{q-1})\dot{y}_{d'}(t_{q'-1})\}\\
&+\operatorname{cov}\{f_d(t,t_{q-1},t_q,u_{q-1})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}.
\end{aligned}
$$

The terms $\operatorname{cov}\{y_d(t_{q-1})y_{d'}(t_{q'-1})\}$, $\operatorname{cov}\{y_d(t_{q-1})\dot{y}_{d'}(t_{q'-1})\}$, $\operatorname{cov}\{\dot{y}_d(t_{q-1})y_{d'}(t_{q'-1})\}$ and $\operatorname{cov}\{\dot{y}_d(t_{q-1})\dot{y}_{d'}(t_{q'-1})\}$ are obtained from the covariance already computed. The term $\operatorname{cov}\{f_d(t,t_{q-1},t_q,u_{q-1})f_{d'}(t',t_{q'-1},t'_q,u_{q'-1})\}$ is equal to zero, because there is no correlation between u_{q-1} and $u_{q'-1}$. Also, the covariances $c_{d'}(t'-t_{q-1})\operatorname{cov}\{f_d(t,t_{q-1},t_q,u_{q-1})y_{d'}(t_{q'-1})\}$ and $e_{d'}(t'-t_{q-1})\operatorname{cov}\{f_d(t,t_{q-1},t_q,u_{q-1})\dot{y}_{d'}(t_{q'-1})\}$ are zero, since $q > q'$, there is no correlation between force u_{q-1} and any force u_{k-1} for $k <= q'-2$. We can rewrite the above expression as

$$
\begin{aligned}
&c_d(t-t_{q-1})c_{d'}(t'-t_{q'-1})\operatorname{cov}\{y_d(t_{q-1})y_{d'}(t_{q'-1})\} + c_d(t-t_{q-1})e_{d'}(t'-t_{q'-1})\operatorname{cov}\{y_d(t_{q-1})\dot{y}_{d'}(t_{q'-1})\}\\
&+e_d(t-t_{q-1})c_{d'}(t'-t_{q'-1})\operatorname{cov}\{\dot{y}_d(t_{q-1})y_{d'}(t_{q'-1})\} + e_d(t-t_{q-1})e_{d'}(t'-t_{q'-1})\operatorname{cov}\{\dot{y}_d(t_{q-1})\dot{y}_{d'}(t_{q'-1})\}\\
&+c_d(t-t_{q-1})\operatorname{cov}\{y_d(t_{q-1})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\} + e_d(t-t_{q-1})\operatorname{cov}\{\dot{y}_d(t_{q-1})f_{d'}(t',t_{q'-1},t_q,u_{q'-1})\}
\end{aligned}
$$

Terms like $\operatorname{cov}\{y_d(t_{q-1})f_{d'}(t',t',t_{q'-1},t_{q'},u_{q'-1})\}$ and $\operatorname{cov}\{\dot{y}_d(t_{q-1})f_{d'}(t',t',t_{q'-1},t_q,u_{q'-1})\}$ requiere further analysis.

Let's look in detail the term $\operatorname{cov}\{y_d(t_{q-1})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}$. This term is equal to

$$
\begin{aligned}
\operatorname{cov}\{y_d(t_{q-1})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\} &= \operatorname{cov}\left\{p_d(t_{q-1},t_{q-2},t_{q-1},u_{q-2})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\right\}\\
&= \operatorname{cov}\left\{\left[c_d(t_{q-1}-t_{q-2})y_d(t_{q-2}) + e_d(t_{q-1}-t_{q-2})\dot{y}_d(t_{q-2})\right.\right.\\
&\quad \left.\left.+ f_d(t_{q-1},t_{q-2},t_{q-1},u_{q-2})\right]f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\right\}\\
&= \underbrace{c_d(t_{q-1}-t_{q-2})\operatorname{cov}\{y_d(t_{q-2})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}}_{A}\\
&\quad + \underbrace{e_d(t_{q-1}-t_{q-2})\operatorname{cov}\{\dot{y}_d(t_{q-2})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}}_{B}\\
&\quad + \operatorname{cov}\{f_d(t_{q-1},t_{q-2},t_{q-1},u_{q-2})f_{d'}(t',t',t_{q'-1},t_{q'},u_{q'-1})\}.
\end{aligned}
$$

The term $\operatorname{cov}\{f_d(t_{q-1},t_{q-2},t_{q-1},u_{q-2})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}$ is only different from zero for $q = q'+1$ and it would reduce to $\widehat{k}^{(q'-1)}_{f_d,f_{d'}}(t_{q-1},t')$. For A and B, if $q < q'+1$, the terms in the are zero because there is no correlation between forces $u_{q'-1}$ and forces u_{q-2}, for $q < q'+1$. For $q > q'+1$, the term in A is equal to

$$
\begin{aligned}
&c_d(t_{q-1}-t_{q-2})\operatorname{cov}\left\{\left[c_d(t_{q-2}-t_{q-3})y_d(t_{q-3}) + e_d(t_{q-2}-t_{q-3})\dot{y}_d(t_{q-3})\right.\right.\\
&\left.\left.+ f_d(t_{q-2},t_{q-3},t_{q-2},u_{q-3})\right]f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\right\}\\
&= \underbrace{c_d(t_{q-1}-t_{q-2})c_d(t_{q-2}-t_{q-3})\operatorname{cov}\{y_d(t_{q-3})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}}_{A'}\\
&+ \underbrace{c_d(t_{q-1}-t_{q-2})e_d(t_{q-2}-t_{q-3})\operatorname{cov}\{\dot{y}_d(t_{q-3})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}}_{B'}\\
&+ c_d(t_{q-1}-t_{q-2})\operatorname{cov}\{f_d(t_{q-2},t_{q-3},t_{q-2},u_{q-3})f_{d'}(t',t_{q'-1},t_{q'},u_{q'-1})\}.
\end{aligned}
$$

The last term in the above equation is different from zero for $q = q' + 2$. Thus, this last term follows

$$c_d(t_{q-1} - t_{q-2})k_{f_d,f_{d'}}^{(q'-1)}(t_{q-2}, t').$$

The terms A' and B' follow the same form that the terms A and B. Again, if $q < q' + 2$, then the particular terms in are zeros. If, $q > q' + 2$, the recursion repeats until the most inner term in $\text{cov}\{y_d(t_{q-n})f_{d'}(t', t_{q'-1}, t_{q'}, u_{q'-1})\}$ is such that $q = q' + n$. A similar expression can analysis can be made for the term B. The final covariance would then be equal to

$$\begin{aligned}
&c_d(t - t_{q-1})c_{d'}(t' - t_{q'-1})k_{z_d,z_{d'}}(t_{q-1}, t_{q'-1}) + c_d(t - t_{q-1})e_{d'}(t' - t_{q'-1})k_{z_d,\dot{z}_{d'}}(t_{q-1}, t_{q'-1}) \\
+&e_d(t - t_{q-1})c_{d'}(t' - t_{q'-1})k_{\dot{z}_d,z_{d'}}(t_{q-1}, t_{q'-1}) + e_d(t - t_{q-1})e_{d'}(t' - t_{q'-1})k_{\dot{z}_d,\dot{z}_{d'}}(t_{q-1}, t_{q'-1}) \\
&+c_d(t - t_{q-1})f_1(t_{q-1}, t_{q-1}, \ldots, t_{q-n})k_{f_d,f'_d}^{(q'-1)}(t_{q-n}, t') \\
&+c_d(t - t_{q-1})f_2(t_{q-1}, t_{q-1}, \ldots, t_{q-n})k_{m_d,f'_d}^{(q'-1)}(t_{q-n}, t') \\
&+e_d(t - t_{q-1})f_3(t_{q-1}, t_{q-1}, \ldots, t_{q-n})k_{f_d,f'_d}^{(q'-1)}(t_{q-n}, t') \\
&+e_d(t - t_{q-1})f_4(t_{q-1}, t_{q-1}, \ldots, t_{q-n})k_{m_d,f'_d}^{(q'-1)}(t_{q-n}, t'),
\end{aligned}$$

where $f_1(\cdot)$, $f_2(\cdot)$, $f_3(\cdot)$ and $f_4(\cdot)$ are functions of the form

$$\sum x(t_{q-1} - t_{q-2})x(t_{q-2} - t_{q-3}) \ldots x(t_{q-n+1} - t_{q-n}),$$

with x being equal to c_d, e_d, g_d or h_d, depending on the case. To compute the exact form of the expression $f_1(\cdot)$, $f_2(\cdot)$, $f_3(\cdot)$ and $f_4(\cdot)$ we use the following set of rules

- After a $c_d(\cdot)$ term, only $c_d(\cdot)$ and $e_d(\cdot)$ terms follow.
- After a $e_d(\cdot)$ term, only $g_d(\cdot)$ and $h_d(\cdot)$ terms follow.
- After a $g_d(\cdot)$ term, only $c_d(\cdot)$ and $e_d(\cdot)$ terms follow.
- After a $h_d(\cdot)$ term, only $h_d(\cdot)$ and $g_d(\cdot)$ terms follow.

Figures 2, 3, 4 and 5 show examples of the kind of recursions that are generated. In all figures, red indicates a term like $c_d(\cdot)$, blue indicates a term like $e_d(\cdot)$, green indicates a term like $g_d(\cdot)$ and purple indicates $h_d(\cdot)$.

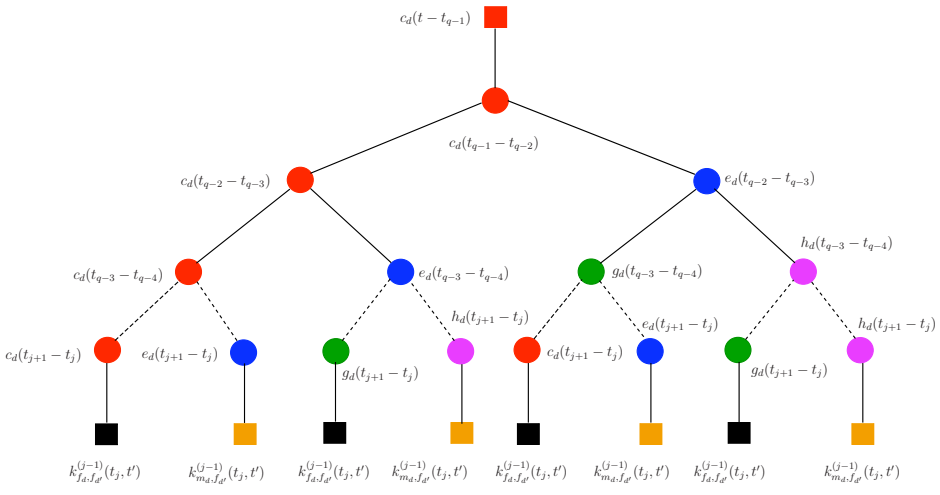


Figure 2: This figure represents the innermost covariances involved when computing the term A'

For $q' > q$ we can make a similar analysis (not presented here).

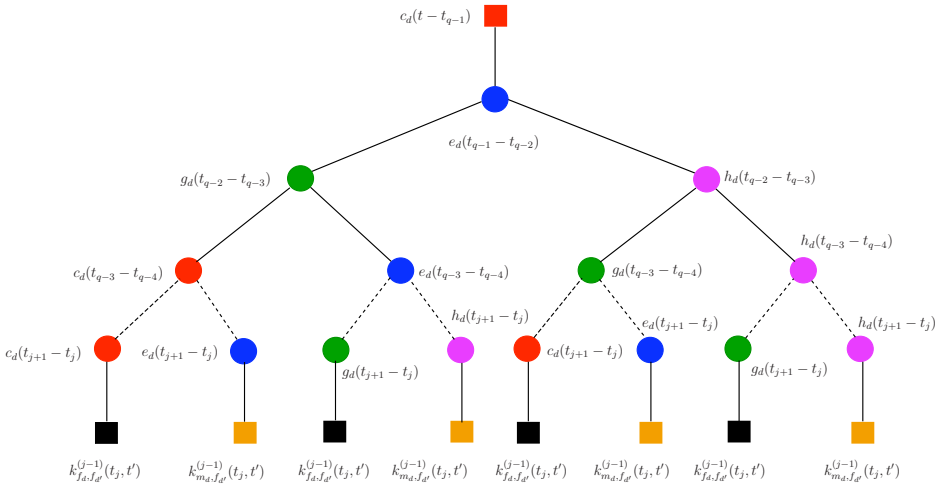


Figure 3: This figure represents the innermost covariances involved when computing the term B'

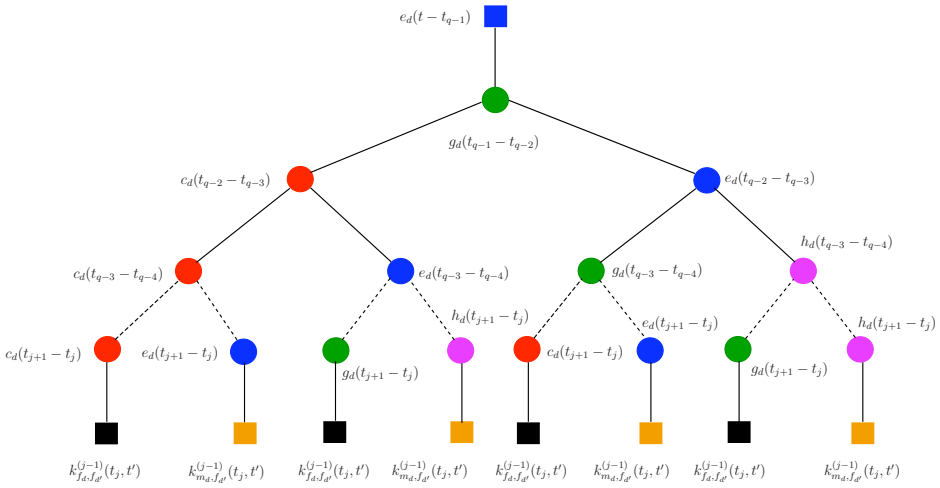


Figure 4: This figure represents the innermost covariances involved when computing the term C'

Example 1 (Continued). We continue with the example in figure 1. First, we compute the covariance between intervals $(t_1, t_2]$ and $(t_0, t_1]$. For this covariance we have

$$\operatorname{cov}[z_d(t), z_{d'}(t')] = \operatorname{cov}\left[p_d(t, t_1, t_2, u_1), p_{d'}(t', t_0, t_1, u_0)\right].$$

This covariance is equal to

$$\begin{aligned}
&c_d(t-t_1)c_{d'}(t'-t_0)\operatorname{cov}\{y_d(t_1)y_{d'}(t_0)\} + c_d(t-t_1)e_{d'}(t'-t_0)\operatorname{cov}\{y_d(t_1)\dot{y}_{d'}(t_0)\} \\
&+e_d(t-t_1)c_{d'}(t'-t_0)\operatorname{cov}\{\dot{y}_d(t_1)y_{d'}(t_0)\} + e_d(t-t_1)e_{d'}(t'-t_0)\operatorname{cov}\{\dot{y}_d(t_1)\dot{y}_{d'}(t_0)\} \\
&+c_d(t-t_1)\operatorname{cov}\{y_d(t_1)f_{d'}(t', t_0, t_1, u_0)\} + e_d(t-t_1)\operatorname{cov}\{\dot{y}_d(t_1)f_{d'}(t', t_0, t_1, u_0)\},
\end{aligned}$$

which reduces to

$$\begin{aligned}
&c_d(t-t_1)c_{d'}(t'-t_0)k_{z_d,z_{d'}}(t_1,t_0) + c_d(t-t_1)e_{d'}(t'-t_0)k_{z_d,\dot{z}_{d'}}(t_1,t_0) \\
&+e_d(t-t_1)c_{d'}(t'-t_0)k_{\dot{z}_d,z_{d'}}(t_1,t_0) + e_d(t-t_1)e_{d'}(t'-t_0)k_{\dot{z}_d,\dot{z}_{d'}}(t_1,t_0) \\
&+c_d(t-t_1)k^{(0)}_{f_d,f_{d'}}(t_1,t') + e_d(t-t_1)k^{(0)}_{m_d,f_{d'}}(t_1,t').
\end{aligned}$$

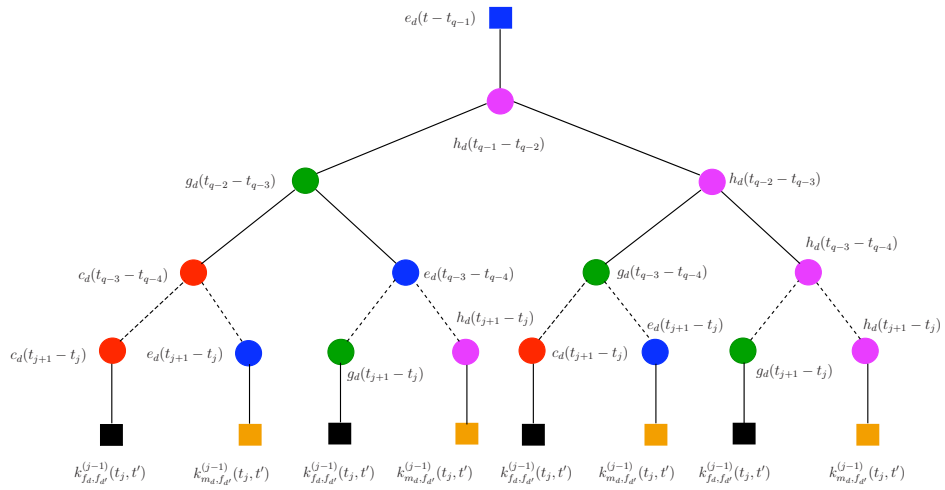


Figure 5: This figure represents the innermost covariances involved when computing the term D'

Now we compute the covariance between intervals $(t_2, t_3]$ and $(t_0, t_1]$. For this covariance we have

$$\operatorname{cov}[z_d(t), z_{d'}(t')] = \operatorname{cov}\left[p_d(t, t, t_2, t_3, u_2), p_{d'}(t', t', t_0, t_1, u_0)\right].$$

We then have

$$\begin{aligned}
&c_d(t-t_2)c_{d'}(t'-t_0)\operatorname{cov}\{y_d(t_2)y_{d'}(t_0)\} + c_d(t-t_2)e_{d'}(t'-t_0)\operatorname{cov}\{y_d(t_2)\dot{y}_{d'}(t_0)\} \\
&+e_d(t-t_2)c_{d'}(t'-t_0)\operatorname{cov}\{\dot{y}_d(t_2)y_{d'}(t_0)\} + e_d(t-t_2)e_{d'}(t'-t_0)\operatorname{cov}\{\dot{y}_d(t_2)\dot{y}_{d'}(t_0)\} \\
&+c_d(t-t_2)\operatorname{cov}\{y_d(t_2)f_{d'}(t', t_0, t_1, u_0)\} + e_d(t-t_2)\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t', t_0, t_1, u_0)\}.
\end{aligned}$$

Using $y_d(t_2) = z_d(t_2, t_1, t_2)$ and $\dot{y}_d(t_2) = \dot{z}_d(t_2.t_1, t_2)$,

$$\begin{aligned}
y_d(t_2) &= z_d(t_2, t_1, t_2) = p_d(t_2, t_1, t_2, u_1) = c_d(t_2-t_1)y_d(t_1) + e_d(t_2-t_1)\dot{y}_d(t_1) + f_d(t_2, t_1, t_2, u_1) \\
\dot{y}_d(t_2) &= \dot{z}_d(t_2, t_1, t_2) = \xi_d(t_2, t_1, t_2, u_1) = g_d(t_2-t_1)y_d(t_1) + h_d(t_2-t_1)\dot{y}_d(t_1) + m_d(t_2, t_1, t_2, u_1),
\end{aligned}$$

we have for $\operatorname{cov}\{y_d(t_2)f_{d'}(t', t', t_0, t_1, u_0)\}$ and $\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t', t', t_0, t_1, u_0)\}$

$$\begin{aligned}
\operatorname{cov}\{y_d(t_2)f_{d'}(t', t_0, t_1, u_0)\} &= c_d(t_2-t_1)\operatorname{cov}\{y_d(t_1)f_{d'}(t', t_0, t_1, u_0)\} \\
&\quad + e_d(t_2-t_1)\operatorname{cov}\{\dot{y}_d(t_1)f_{d'}(t', t_0, t_1, u_0)\} \\
\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t', t', t_0, t_1, u_0)\} &= g_d(t_2-t_1)\operatorname{cov}\{y_d(t_1)f_{d'}(t', t_0, t_1, u_0)\} \\
&\quad + h_d(t_2-t_1)\operatorname{cov}\{\dot{y}_d(t_1)f_{d'}(t', t_0, t_1, u_0)\}.
\end{aligned}$$

Furthermore,

$$\begin{aligned}
y_d(t_1) &= z_d(t_1, t_0, t_1) = p_d(t_1, t_0, t_1, u_0) = c_d(t_1-t_0)y_d(t_0) + e_d(t_1-t_0)\dot{y}_d(t_0) + f_d(t_1, t_0, t_1, u_0) \\
\dot{y}_d(t_1) &= \dot{z}_d(t_2, t_1, t_2) = \xi_d(t_1, t_0, t_1, u_0) = g_d(t_1-t_0)y_d(t_0) + h_d(t_1-t_0)\dot{y}_d(t_0) + m_d(t_1, t_0, t_1, u_0).
\end{aligned}$$

Then we get $\operatorname{cov}\{y_d(t_1)f_{d'}(t', t_0, t_1, u_0)\} = k^{(0)}_{f_d,f_{d'}}(t_1, t')$ and $\operatorname{cov}\{\dot{y}_d(t_1)f_{d'}(t', t_0, t_1, u_0)\} = k^{(0)}_{m_d,f_{d'}}(t_1, t')$. Putting all these expressions together, we get

$$\begin{aligned}
&c_d(t-t_2)c_{d'}(t'-t_0)k_{z_d,z_{d'}}(t_2, t_0) + c_d(t-t_2)e_{d'}(t'-t_0)k_{z_d,\dot{z}_{d'}}(t_2, t_0) \\
&+e_d(t-t_2)c_{d'}(t'-t_0)k_{\dot{z}_d,z_{d'}}(t_2, t_0) + e_d(t-t_2)e_{d'}(t'-t_0)k_{\dot{z}_d,\dot{z}_{d'}} \\
&+c_d(t-t_2)c_d(t_2-t_1)k^{(0)}_{f_d,f_{d'}}(t_1, t') + c_d(t-t_2)e_d(t_2-t_1)k^{(0)}_{m_d,f_{d'}}(t_1, t') \\
&+e_d(t-t_2)g_d(t_2-t_1)k^{(0)}_{f_d,f_{d'}}(t_1, t') + e_d(t-t_2)h_d(t_2-t_1)k^{(0)}_{m_d,f_{d'}}(t_1, t')
\end{aligned}$$

Next we compute the covariance between intervals $(t_2, t_3]$ and $(t_1, t_2]$. For this covariance we have

$$\operatorname{cov}[z_d(t), z_{d'}(t')] = \operatorname{cov}\left[p_d(t, t_2, t_3, u_2), p_{d'}(t', t_1, t_2, u_1)\right]$$

We have

$$c_d(t-t_2)c_{d'}(t'-t_1)k_{z_d,z_{d'}}(t_2,t_1)+c_d(t-t_2)e_{d'}(t'-t_1)k_{z_d,\dot{z}_{d'}}(t_2,t_1)$$
$$+e_d(t-t_2)c_{d'}(t'-t_1)k_{\dot{z}_d,z_{d'}}(t_2,t_1)+e_d(t-t_2)e_{d'}(t'-t_1)k_{\dot{z}_d,\dot{z}_{d'}}(t_2,t_1)$$
$$+c_d(t-t_2)\operatorname{cov}\{y_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}+e_d(t-t_2)\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}.$$

The covariance $\operatorname{cov}\{y_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}=k^{(1)}_{f_d,f'_d}(t_2,t')$ and $\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}=k^{(1)}_{m_d,f'_d}(t_2,t')$. Then, the complete covariance would be equal to

$$c_d(t-t_2)c_{d'}(t'-t_1)k_{z_d,z_{d'}}(t_2,t_1)+c_d(t-t_2)e_{d'}(t'-t_1)k_{z_d,\dot{z}_{d'}}(t_2,t_1)$$
$$+e_d(t-t_2)c_{d'}(t'-t_1)k_{\dot{z}_d,z_{d'}}(t_2,t_1)+e_d(t-t_2)e_{d'}(t'-t_1)k_{\dot{z}_d,\dot{z}_{d'}}(t_2,t_1)$$
$$+c_d(t-t_2)k^{(1)}_{f_d,f'_d}(t_2,t')+e_d(t-t_2)k^{(1)}_{m_d,f'_d}(t_2,t').$$

Suppose we need to compute the covariance between the intervals $(t_4,t_5]$ and $(t_1,t_2]$. For this $q=4$ and $q'=1$. The covariance is given as

$$\operatorname{cov}\{[c_d(t-t_4)y_d(t_4)+e_d(t-t_4)\dot{y}_d(t_4)+f_d(t,t_4,t_5,u_4)]$$
$$[c_{d'}(t'-t_1)y_{d'}(t_1)+e_{d'}(t'-t_1)\dot{y}_{d'}(t_1)+f_{d'}(t',t_1,t_2,u_1)\}$$
$$=c_d(t-t_4)c_{d'}(t'-t_1)\operatorname{cov}\{y_d(t_4)y_{d'}(t_1)\}+c_d(t-t_4)e_{d'}(t'-t_1)\operatorname{cov}\{y_d(t_4)\dot{y}_{d'}(t_1)\}$$
$$+e_d(t-t_4)c_{d'}(t'-t_1)\operatorname{cov}\{\dot{y}_d(t_4)y_{d'}(t_1)\}+e_d(t-t_4)e_{d'}(t'-t_1)\operatorname{cov}\{\dot{y}_d(t_4)\dot{y}_{d'}(t_1)\}$$
$$+c_d(t-t_4)\operatorname{cov}\{y_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}+e_d(t-t_4)\operatorname{cov}\{\dot{y}_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}$$

We need to compute the covariances $\operatorname{cov}\{y_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}$ and $\operatorname{cov}\{\dot{y}_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}$. The expression for $y_d(t_4)$ is

$$y_d(t_4)=z_d(t_4,t_3,t_4)=p_d(t_4,t_3,t_4,u_3)=c_d(t_4-t_3)y_d(t_3)+e_d(t_4-t_3)\dot{y}_d(t_3)+f_d(t_4,t_3,t_4,u_3)$$
$$\dot{y}_d(t_4)=\dot{z}_d(t_4,t_3,t_4)=\xi_d(t_4,t_3,t_4,u_3)=g_d(t_4-t_3)y_d(t_3)+h_d(t_4-t_3)\dot{y}_d(t_3)+m_d(t_4,t_3,t_4,u_3)$$

Then the covariances $\operatorname{cov}\{y_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}$ and $\operatorname{cov}\{\dot{y}_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}$ are equal to

$$c_d(t_4-t_3)\operatorname{cov}\{y_d(t_3)f_{d'}(t',t_1,t_2,u_1)\}+e_d(t_4-t_3)\operatorname{cov}\{\dot{y}_d(t_3)f_{d'}(t',t_1,t_2,u_1)\},$$
$$g_d(t_4-t_3)\operatorname{cov}\{y_d(t_3)f_{d'}(t',t_1,t_2,u_1)\}+h_d(t_4-t_3)\operatorname{cov}\{\dot{y}_d(t_3)f_{d'}(t',t_1,t_2,u_1)\}.$$

At the same time, in the above expression, we have that $y_d(t_2)$ and $\dot{y}_d(t_2)$ follow

$$y_d(t_3)=c_d(t_3-t_2)y_d(t_2)+e_d(t_3-t_2)\dot{y}_d(t_2)+f_d(t_3,t_2,t_3,u_2)$$
$$\dot{y}_d(t_3)=g_d(t_3-t_2)y_d(t_2)+h_d(t_3-t_2)\dot{y}_d(t_2)+m_d(t_3,t_2,t_3,u_2)$$

Then, we can write the expression for $\operatorname{cov}\{y_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}$ as

$$c_d(t_4-t_3)\big[c_d(t_3-t_2)\operatorname{cov}\{y_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}+e_d(t_3-t_2)\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}\big]$$
$$+e_d(t_4-t_3)\big[g_d(t_3-t_2)\operatorname{cov}\{y_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}+h_d(t_3-t_2)\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}\big].$$

The expression for $\operatorname{cov}\{\dot{y}_d(t_4)f_{d'}(t',t_1,t_2,u_1)\}$ would follow

$$g_d(t_4-t_3)\big[c_d(t_3-t_2)\operatorname{cov}\{y_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}+e_d(t_3-t_2)\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}\big]$$
$$+h_d(t_4-t_3)\big[g_d(t_3-t_2)\operatorname{cov}\{y_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}+h_d(t_3-t_2)\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}\big].$$

From the expression for $y_d(t_2)$ and $\dot{y}_d(t_2)$, we get $\operatorname{cov}\{y_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}=k^{(1)}_{f_d,f_{d'}}(t_2,t')$ and $\operatorname{cov}\{\dot{y}_d(t_2)f_{d'}(t',t_1,t_2,u_1)\}=k^{(1)}_{m_d,f_{d'}}(t_2,t')$. The total covariance then would be equal to

$$c_d(t-t_4)c_{d'}(t'-t_1)k_{z_d,z_{d'}}(t_4,t_1)+c_d(t-t_4)e_{d'}(t'-t_1)k_{z_d,\dot{z}_{d'}}(t_4,t_1)$$
$$+e_d(t-t_4)c_{d'}(t'-t_1)k_{\dot{z}_d,z_{d'}}(t_4,t_1)+e_d(t-t_4)e_{d'}(t'-t_1)k_{\dot{z}_d,\dot{z}_{d'}}(t_4,t_1)$$
$$+c_d(t-t_4)\big[c_d(t_4-t_3)\big[c_d(t_3-t_2)k^{(1)}_{f_d,f_{d'}}(t_2,t')+e_d(t_3-t_2)k^{(1)}_{m_d,f_{d'}}(t_2,t')\big]$$
$$+e_d(t_4-t_3)\big[g_d(t_3-t_2)k^{(1)}_{f_d,f_{d'}}(t_2,t')+h_d(t_3-t_2)k^{(1)}_{m_d,f_{d'}}(t_2,t')\big]\big]$$
$$+e_d(t-t_4)\big[g_d(t_4-t_3)\big[c_d(t_3-t_2)k^{(1)}_{f_d,f_{d'}}(t_2,t')+e_d(t_3-t_2)k^{(1)}_{m_d,f_{d'}}(t_2,t')\big]$$
$$+h_d(t_4-t_3)\big[g_d(t_3-t_2)k^{(1)}_{f_d,f_{d'}}(t_2,t')+h_d(t_3-t_2)k^{(1)}_{m_d,f_{d'}}(t_2,t')\big]\big].$$

Reorganizing, we get

$$
\begin{aligned}
&c_d(t-t_4)c_{d'}(t'-t_1)k_{z_d,z_{d'}}(t_4,t_1) + c_d(t-t_4)e_{d'}(t'-t_1)k_{z_d,\dot{z}_{d'}}(t_4,t_1)\\
&+ e_d(t-t_4)c_{d'}(t'-t_1)k_{\dot{z}_d,z_{d'}}(t_4,t_1) + e_d(t-t_4)e_{d'}(t'-t_1)k_{\dot{z}_d,\dot{z}_{d'}}(t_4,t_1)\\
&+ c_d(t-t_4)\big[c_d(t_4-t_3)c_d(t_3-t_2) + e_d(t_4-t_3)g_d(t_3-t_2)\big]k^{(1)}_{f_d,f_{d'}}(t_2,t')\\
&+ c_d(t-t_4)\big[c_d(t_4-t_3)e_d(t_3-t_2) + e_d(t_4-t_3)h_d(t_3-t_2)\big]k^{(1)}_{m_d,f_{d'}}(t_2,t')\\
&+ e_d(t-t_4)\big[g_d(t_4-t_3)c_d(t_3-t_2) + h_d(t_4-t_3)g_d(t_3-t_2)\big]k^{(1)}_{f_d,f_{d'}}(t_2,t')\\
&+ e_d(t-t_4)\big[g_d(t_4-t_3)e_d(t_3-t_2) + h_d(t_4-t_3)h_d(t_3-t_2)\big]k^{(1)}_{m_d,f_{d'}}(t_2,t').
\end{aligned}
$$

Or in a more familiar expression,

$$
\begin{aligned}
&c_d(t-t_4)c_{d'}(t'-t_1)k_{z_d,z_{d'}}(t_4,t_1) + c_d(t-t_4)e_{d'}(t'-t_1)k_{z_d,\dot{z}_{d'}}(t_4,t_1)\\
&+ e_d(t-t_4)c_{d'}(t'-t_1)k_{\dot{z}_d,z_{d'}}(t_4,t_1) + e_d(t-t_4)e_{d'}(t'-t_1)k_{\dot{z}_d,\dot{z}_{d'}}(t_4,t_1)\\
&+ c_d(t-t_4)f_1(t_4,t_3,t_2)k^{(1)}_{f_d,f_{d'}}(t_2,t') + c_d(t-t_4)f_2(t_4,t_3,t_2)k^{(1)}_{m_d,f_{d'}}(t_2,t')\\
&+ e_d(t-t_4)f_3(t_4,t_3,t_2)k^{(1)}_{f_d,f_{d'}}(t_2,t') + e_d(t-t_4)f_4(t_4,t_3,t_2)k^{(1)}_{m_d,f_{d'}}(t_2,t').
\end{aligned}
$$

where, $f_1(t_4,t_3,t_2) = c_d(t_4-t_3)c_d(t_3-t_2) + e_d(t_4-t_3)g_d(t_3-t_2)$, $f_2(t_4,t_3,t_2) = c_d(t_4-t_3)e_d(t_3-t_2)+e_d(t_4-t_3)h_d(t_3-t_2)$, $f_3(t_4,t_3,t_2) = g_d(t_4-t_3)c_d(t_3-t_2)+h_d(t_4-t_3)g_d(t_3-t_2)$ and $f_4(t_4,t_3,t_2) = g_d(t_4-t_3)e_d(t_3-t_2) + h_d(t_4-t_3)h_d(t_3-t_2)$.

2.3 Covariances between outputs and latent functions

For inference purposes, we'll also need the cross-covariances between the ouputs $z_d(t,t_{q-1},t_q)$ and the latent forces $u_{q'-1}(t')$. If $q' > q$, then this covariance is zero. We are left with the cases $q' = q$ and $q' < q$.

2.3.1 Covariance between $z_d(t,t_{q-1},t_q)$ and $u_{q'-1}(t')$, with $q' = q$

We have

$$
\operatorname{cov}[z_d(t,t_{q-1},t_q),u_{q-1}(t')] = \operatorname{cov}\left[p_d(t,t_{q-1},t_q,u_{q-1})u_{q-1}(t')\right],
$$

which is given as

$$
\begin{aligned}
c_d(t-t_{q-1})\operatorname{cov}\left[y_d(t_{q-1})u_{q-1}(t)\right] + e_d(t-t_{q-1})\operatorname{cov}\left[\dot{y}_d(t_{q-1})u_{q-1}(t)\right]\\
+ \operatorname{cov}\left[f_d(t,t_{q-1},t_q,u_{q-1})u_{q-1}(t')\right].
\end{aligned}
$$

From the above equation, the only term different from zero is $\operatorname{cov}\left[f_d(t,t_{q-1},t_q,u_{q-1})u_{q-1}(t')\right] = k_{f_d,u_{q-1}}(t,t')$. Then, we have $\operatorname{cov}\left[p_d(t,t_{q-1},t_q,u_{q-1})u_{q-1}(t')\right] = k_{f_d,u_{q-1}}(t,t')$.

2.3.2 Covariance between $z_d(t,t_{q-1},t_q)$ and $u_{q'-1}(t')$, with $q' < q$

We have

$$
\operatorname{cov}[z_d(t,t_{q-1},t_q),u_{q'-1}(t')] = \operatorname{cov}\left[p_d(t,t_{q-1},t_q,u_{q-1})u_{q'-1}(t')\right].
$$

It would be

$$
\begin{aligned}
c_d(t-t_{q-1})\operatorname{cov}\left[y_d(t_{q-1})u_{q'-1}(t')\right] + e_d(t-t_{q-1})\operatorname{cov}\left[\dot{y}_d(t_{q-1})u_{q'-1}(t')\right]\\
+ \operatorname{cov}\left[f_d(t,t_{q-1},t_q,u_{q-1})u_{q'-1}\right].
\end{aligned}
$$

Being q strictly greater than q', we only need to compute $\operatorname{cov}\left[y_d(t_{q-1})u_{q'-1}(t')\right]$ and $\operatorname{cov}\left[\dot{y}_d(t_{q-1})u_{q'-1}(t')\right]$. For the first term, we have

$$
\begin{aligned}
\operatorname{cov}\left[y_d(t_{q-1})u_{q'-1}(t')\right] &= \operatorname{cov}\left[\big(c_d(t_{q-1}-t_{q-2})y_d(t_{q-2}) + e_d(t_{q-1}-t_{q-2})\dot{y}_d(t_{q-2})\right.\\
&\quad \left.+ f_d(t_{q-1},t_{q-2},t_{q-1},u_{q-2})\big)u_{q'-1}(t')\right]\\
&= \underbrace{c_d(t_{q-1}-t_{q-2})\operatorname{cov}\left[y_d(t_{q-2})u_{q'-1}(t')\right]}_{A}\\
&\quad + \underbrace{e_d(t_{q-1}-t_{q-2})\operatorname{cov}\left[\dot{y}_d(t_{q-2})u_{q'-1}(t')\right]}_{B}\\
&\quad + \operatorname{cov}\left[f_d(t_{q-1},t_{q-2},t_{q-1},u_{q-2})u_{q'-1}(t')\right].
\end{aligned}
$$

The terms A and B, repeat again in a recursion similar to the ones in section 2.2.2. The final expression is then equal to

$$\begin{aligned}
&c_d(t-t_{q-1})f_1(t_{q-1},t_{q-1},\ldots,t_{q-n})k^{(q'-1)}_{f_d,u_{q'-1}}(t_{q-n},t')\\
+&c_d(t-t_{q-1})f_2(t_{q-1},t_{q-1},\ldots,t_{q-n})k^{(q'-1)}_{m_d,u_{q'-1}}(t_{q-n},t')\\
+&e_d(t-t_{q-1})f_3(t_{q-1},t_{q-1},\ldots,t_{q-n})k^{(q'-1)}_{f_d,u_{q'-1}}(t_{q-n},t')\\
+&e_d(t-t_{q-1})f_4(t_{q-1},t_{q-1},\ldots,t_{q-n})k^{(q'-1)}_{m_d,u_{q'-1}}(t_{q-n},t'),
\end{aligned}$$

where $f_1(\cdot)$, $f_2(\cdot)$, $f_3(\cdot)$ and $f_4(\cdot)$ are again functions of the form

$$\sum x(t_{q-1}-t_{q-2})x(t_{q-2}-t_{q-3})\ldots x(t_{q-n+1}-t_{q-n}),$$

with x being equal to c_d, e_d, g_d or h_d, depending on the case.

Example 1 (continued). We continue with the example. We want to compute the following terms

$$\begin{array}{lll}
\operatorname{cov}[z_d(t,t_0,t_1),u_0(t')] & \operatorname{cov}[z_d(t,t_0,t_1),u_1(t')] & \operatorname{cov}[z_d(t,t_0,t_1),u_2(t')]\\
\operatorname{cov}[z_d(t,t_1,t_2),u_0(t')] & \operatorname{cov}[z_d(t,t_1,t_2),u_1(t')] & \operatorname{cov}[z_d(t,t_1,t_2),u_2(t')]\\
\operatorname{cov}[z_d(t,t_2,t_3),u_0(t')] & \operatorname{cov}[z_d(t,t_2,t_3),u_1(t')] & \operatorname{cov}[z_d(t,t_2,t_3),u_2(t')]
\end{array}$$

From the above analysis, the terms $\operatorname{cov}[z_d(t,t_0,t_1),u_1(t')]$, $\operatorname{cov}[z_d(t,t_0,t_1),u_2(t')]$ and $\operatorname{cov}[z_d(t,t_1,t_2),u_2(t')]$ are zero. Furthermore, the terms $\operatorname{cov}[z_d(t,t_0,t_1),u_0(t')]$, $\operatorname{cov}[z_d(t,t_1,t_2),u_1(t')]$ and $\operatorname{cov}[z_d(t,t_2,t_3),u_2(t')]$ are

$$\begin{aligned}
\operatorname{cov}[z_d(t,t_0,t_1),u_0(t')] &= k_{f_d,u_0}(t,t')\\
\operatorname{cov}[z_d(t,t_1,t_2),u_1(t')] &= k_{f_d,u_1}(t,t')\\
\operatorname{cov}[z_d(t,t_2,t_3),u_2(t')] &= k_{f_d,u_2}(t,t').
\end{aligned}$$

We are left with the terms $\operatorname{cov}[z_d(t,t_1,t_2),u_0(t')]$, $\operatorname{cov}[z_d(t,t_2,t_3),u_0(t')]$ and $\operatorname{cov}[z_d(t,t_2,t_3),u_1(t')]$. The term $\operatorname{cov}[z_d(t,t_1,t_2),u_0(t')]$ follows as

$$\begin{aligned}
\operatorname{cov}[z_d(t,t_1,t_2),u_0(t')] &= \operatorname{cov}\{[p_d(t,t_1,t_2,u_1)]u_0(t')\}\\
&= \operatorname{cov}\{[c_d(t-t_1)y_d(t_1)+e_d(t-t_1)\dot{y}_d(t_1)+f_d(t,t_1,t_2,u_1)]\,u_0(t')\}\\
&= c_d(t-t_1)\operatorname{cov}\left[y_d(t_1)u_0(t')\right]+e_d(t-t_1)\operatorname{cov}\left[\dot{y}_d(t_1)u_0(t')\right].
\end{aligned}$$

The terms $\operatorname{cov}\left[y_d(t_1)u_0(t')\right]$ and $\operatorname{cov}\left[\dot{y}_d(t_1)u_0(t')\right]$ are

$$\begin{aligned}
\operatorname{cov}\left[y_d(t_1)u_0(t')\right] &= \operatorname{cov}\left[(c_d(t_1-t_0)y_d(t_0)+e_d(t_1-t_0)\dot{y}_d(t_0)+f_d(t_1,t_0,t_1,u_0))u_0(t')\right]\\
&= k_{f_d,u_0}(t_1,t')\\
\operatorname{cov}\left[\dot{y}_d(t_1)u_0(t')\right] &= \operatorname{cov}\left[(g_d(t_1-t_0)y_d(t_0)+h_d(t_1-t_0)\dot{y}_d(t_0)+m_d(t_1,t_0,t_1,u_0))u_0(t')\right]\\
&= k_{m_d,u_0}(t_1,t').
\end{aligned}$$

The final covariance is then

$$\operatorname{cov}[z_d(t,t_1,t_2),u_0(t')] = c_d(t-t_1)k_{f_d,u_0}(t_1,t')+e_d(t-t_1)k_{m_d,u_0}(t_1,t').$$

Now, we compute the term $\operatorname{cov}[z_d(t,t_2,t_3),u_0(t')]$, which will be given as

$$\begin{aligned}
\operatorname{cov}[z_d(t,t_2,t_3),u_0(t')] &= \operatorname{cov}\{[p_d(t,t,t_2,t_3,u_2)]u_0(t')\}\\
&= \operatorname{cov}\{[c_d(t-t_2)y_d(t_2)+e_d(t-t_2)\dot{y}_d(t_2)+f_d(t,t_2,t_3,u_2)]\,u_0(t')\}\\
&= c_d(t-t_2)\operatorname{cov}\left[y_d(t_2)u_0(t')\right]+e_d(t-t_2)\operatorname{cov}\left[\dot{y}_d(t_2)u_0(t')\right].
\end{aligned}$$

The term $\operatorname{cov}\left[y_d(t_2)u_0(t')\right]$ follows

$$\begin{aligned}
\operatorname{cov}[y_d(t_2),u_0(t')] &= \operatorname{cov}\{p_d(t_2,t_1,t_2,u_1)u_0(t')\}\\
&= c_d(t_2-t_1)\operatorname{cov}[y_d(t_1)u_0(t')]+e_d(t_2-t_1)\operatorname{cov}[\dot{y}_d(t_1)u_0(t')].
\end{aligned}$$

The term $\mathrm{cov}\left[\dot{y}_d(t_2)u_0(t')\right]$ follows

$$\begin{aligned}\mathrm{cov}[\dot{y}_d(t_2), u_0(t')] &= \mathrm{cov}\{\xi_d(t_2, t_1, t_2, u_1)u_0(t')\} \\ &= g_d(t_2 - t_1)\,\mathrm{cov}[y_d(t_1)u_0(t')] + h_d(t_2 - t_1)\,\mathrm{cov}[\dot{y}_d(t_1)u_0(t')]\end{aligned}$$

Putting together all these terms, the covariance $\mathrm{cov}[z_d(t, t_2, t_3), u_0(t')]$ is given as

$$\begin{aligned}\mathrm{cov}[z_d(t, t_2, t_3), u_0(t')] = c_d(t - t_2)\,&[c_d(t_2 - t_1)k_{f_d,u_0}(t_1, t') + e_d(t_2 - t_1)k_{m_d,u_0}(t_1, t')] \\ &+ e_d(t - t_2)\,[g_d(t_2 - t_1)k_{f_d,u_0}(t_1, t') + h_d(t_2 - t_1)k_{m_d,u_0}(t_1, t')]\,.\end{aligned}$$

Or in a more familiar form

$$\begin{aligned}\mathrm{cov}[z_d(t, t_2, t_3), u_0(t')] = {} & c_d(t - t_2)f_1(t_2, t_1)k_{f_d,u_0}(t_1, t') + c_d(t - t_2)f_2(t_2, t_1)k_{m_d,u_0}(t_1, t') \\ & + e_d(t - t_2)f_3(t_2, t_1)k_{f_d,u_0}(t_1, t') + e_d(t - t_2)f_4(t_2, t_1)k_{m_d,u_0}(t_1, t'),\end{aligned}$$

where $f_1(t_2, t_1) = c_d(t_2 - t_1)$, $f_2(t_2, t_1) = e_d(t_2 - t_1)$, $f_3(t_2, t_1) = g_d(t_2 - t_1)$ and $f_4(t_2, t_1) = h_d(t_2 - t_1)$.
Finally, we compute $\mathrm{cov}[z_d(t, t_2, t_3), u_1(t')]$ as

$$\begin{aligned}\mathrm{cov}[z_d(t, t_2, t_3), u_1(t')] &= \mathrm{cov}\{p_d(t, t_2, t_3, u_2)u_1(t')\} \\ &= \mathrm{cov}\{[c_d(t - t_2)y_d(t_2) + e_d(t - t_2)\dot{y}_d(t_2) + f_d(t, t_2, t_3, u_2)]\,u_1(t')\} \\ &= c_d(t - t_2)k_{f_d,u_1}(t_2, t') + e_d(t - t_2)k_{m_d,u_1}(t_2, t').\end{aligned}$$

3 Covariance for the velocities and accelerations

To get expressions for the covariances $\mathrm{cov}\left[z_d(t), \dot{z}_{d'}(t')\right]$ (Position - Velocity), $\mathrm{cov}\left[\dot{z}_d(t), z_{d'}(t')\right]$ (Velocity - Position), $\mathrm{cov}\left[\dot{z}_d(t), \dot{z}_{d'}(t')\right]$ (Velocity - Velocity), $\mathrm{cov}\left[z_d(t), \ddot{z}_{d'}\right]$ (Position - Acceleration), $\mathrm{cov}\left[\ddot{z}_d(t), z_{d'}(t')\right]$ (Acceleration - Position), $\mathrm{cov}\left[\dot{z}_d(t), \ddot{z}_{d'}\right]$ (Velocity - Acceleration), $\mathrm{cov}\left[\ddot{z}_d(t), \dot{z}_{d'}(t')\right]$ (Acceleration - Velocity) and $\mathrm{cov}\left[\ddot{z}_d(t), \ddot{z}_{d'}(t')\right]$ (Acceleration - Acceleration), we take the appropiate number of derivatives with respect to t and t' [3].

References

[1] Mauricio Álvarez, David Luengo, and Neil D. Lawrence. Latent Force Models. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pages 9–16, Clearwater Beach, Florida, 16-18 April 2009. JMLR W&CP 5.

[2] Stefan Schaal, Peyman Mohajerian, and Auke Ijspeert. *Progress in Brain Research*, chapter Dynamics systems vs. optimal control - a unifying view, pages 425–445. Elsevier B.V., 2007.

[3] E. Solak, R. Murray-Smith W. E. Leithead, D. J. Leith, and C. E. Rasmussen. Derivative observations in Gaussian process models of dynamic systems. In Sue Becker, Sebastian Thrun, and Klaus Obermayer, editors, *NIPS*, volume 15, pages 1033–1040, Cambridge, MA, 2003. MIT Press.

Reprint of publication V

Mauricio A. Álvarez and Neil D. Lawrence (2011): "Computationally Efficient Convolved Multiple Output Gaussian Processes", *Journal of Machine Learning Research 12*, pp 1425–1466.

Journal of Machine Learning Research 12 (2011) 1425-1466 Submitted 11/09; Revised 10/10; Published 5/11

Computationally Efficient Convolved Multiple Output Gaussian Processes

Mauricio A. Álvarez* MALVAREZ@UTP.EDU.CO
School of Computer Science
University of Manchester
Manchester, UK, M13 9PL

Neil D. Lawrence† N.LAWRENCE@SHEFFIELD.AC.UK
School of Computer Science
University of Sheffield
Sheffield, S1 4DP

Editor: Carl Edward Rasmussen

Abstract

Recently there has been an increasing interest in regression methods that deal with multiple outputs. This has been motivated partly by frameworks like multitask learning, multisensor networks or structured output data. From a Gaussian processes perspective, the problem reduces to specifying an appropriate covariance function that, whilst being positive semi-definite, captures the dependencies between all the data points and across all the outputs. One approach to account for non-trivial correlations between outputs employs convolution processes. Under a latent function interpretation of the convolution transform we establish dependencies between output variables. The main drawbacks of this approach are the associated computational and storage demands. In this paper we address these issues. We present different efficient approximations for dependent output Gaussian processes constructed through the convolution formalism. We exploit the conditional independencies present naturally in the model. This leads to a form of the covariance similar in spirit to the so called PITC and FITC approximations for a single output. We show experimental results with synthetic and real data, in particular, we show results in school exams score prediction, pollution prediction and gene expression data.

Keywords: Gaussian processes, convolution processes, efficient approximations, multitask learning, structured outputs, multivariate processes

1. Introduction

Accounting for dependencies between model outputs has important applications in several areas. In sensor networks, for example, missing signals from failing sensors may be predicted due to correlations with signals acquired from other sensors (Osborne et al., 2008). In geostatistics, prediction of the concentration of heavy pollutant metals (for example, Copper), that are expensive to measure, can be done using inexpensive and oversampled variables (for example, pH) as a proxy (Goovaerts, 1997). Within the machine learning community this approach is sometimes known as multitask learning. The idea in multitask learning is that information shared between the tasks leads to im-

∗. Also in Faculty of Engineering, Universidad Tecnológica de Pereira, Pereira, Colombia, 660003.
†. Also at the Sheffield Institute for Translational Neuroscience, Sheffield, UK, S10 2HQ.

©2011 Mauricio A. Álvarez and Neil D. Lawrence.

proved performance in comparison to learning the same tasks individually (Caruana, 1997; Bonilla et al., 2008).

In this paper, we consider the problem of modeling related outputs in a Gaussian process (GP). A Gaussian process specifies a prior distribution over functions. When using a GP for multiple related outputs, our purpose is to develop a prior that expresses correlation between the outputs. This information is encoded in the covariance function. The class of valid covariance functions is the same as the class of reproducing kernels.[1] Such kernel functions for single outputs are widely studied in machine learning (see, for example, Rasmussen and Williams, 2006). More recently the community has begun to turn its attention to covariance functions for multiple outputs. One of the paradigms that has been considered (Teh et al., 2005; Osborne et al., 2008; Bonilla et al., 2008) is known in the geostatistics literature as *the linear model of coregionalization* (LMC) (Journel and Huijbregts, 1978; Goovaerts, 1997). In the LMC, the covariance function is expressed as the sum of Kronecker products between *coregionalization matrices* and a set of underlying covariance functions. The correlations across the outputs are expressed in the coregionalization matrices, while the underlying covariance functions express the correlation between different data points.

Multitask learning has also been approached from the perspective of *regularization theory* (Evgeniou and Pontil, 2004; Evgeniou et al., 2005). These *multitask kernels* are obtained as generalizations of the regularization theory to vector-valued functions. They can also be seen as examples of LMCs applied to linear transformations of the input space.

In the linear model of coregionalization each output can be thought of as an instantaneous mixing of the underlying signals/processes. An alternative approach to constructing covariance functions for multiple outputs employs *convolution processes* (CP). To obtain a CP in the single output case, the output of a given process is convolved with a smoothing kernel function. For example, a white noise process may be convolved with a smoothing kernel to obtain a covariance function (Barry and Ver Hoef, 1996; Ver Hoef and Barry, 1998). Ver Hoef and Barry (1998) and then Higdon (2002) noted that if a single input process was convolved with different smoothing kernels to produce different outputs, then correlation between the outputs could be expressed. This idea was introduced to the machine learning audience by Boyle and Frean (2005). We can think of this approach to generating multiple output covariance functions as a non-instantaneous mixing of the base processes.

The convolution process framework is an elegant way for constructing dependent output processes. However, it comes at the price of having to consider the full covariance function of the joint GP. For D output dimensions and N data points the covariance matrix scales as DN leading to $O(N^3D^3)$ computational complexity and $O(N^2D^2)$ storage. We are interested in exploiting the richer class of covariance structures allowed by the CP framework, but reducing the additional computational overhead they imply.

In this paper, we propose different efficient approximations for the full covariance matrix involved in the multiple output convolution process. We exploit the fact that, in the convolution framework, each of the outputs is conditional independent of all others if the input process is fully observed. This leads to an approximation that turns out to be strongly related to the partially independent training conditional (PITC) (Quiñonero-Candela and Rasmussen, 2005) approximation for a single output GP. This analogy inspires us to consider a further conditional independence

1. In this paper we will use kernel to refer to both reproducing kernels and smoothing kernels. Reproducing kernels are those used in machine learning that conform to Mercer's theorem. Smoothing kernels are kernel functions which are convolved with a signal to create a smoothed version of that signal.

assumption across data points. This leads to an approximation which shares the form of the fully independent training conditional (FITC) approximation (Snelson and Ghahramani, 2006; Quiñonero-Candela and Rasmussen, 2005). This reduces computational complexity to $O(NDK^2)$ and storage to $O(NDK)$ with K representing a user specified value for the number of inducing points in the approximation.

The rest of the paper is organized as follows. First we give a more detailed review of related work, with a particular focus on relating multiple output work in machine learning to other fields. Despite the fact that there are several other approaches to multitask learning (see for example Caruana, 1997, Heskes, 2000, Bakker and Heskes, 2003, Xue et al., 2007 and references therein), in this paper, we focus our attention to those that address the problem of constructing the covariance or kernel function for multiple outputs, so that it can be employed, for example, together with Gaussian process prediction. Then we review the convolution process approach in Section 3 and Section 4. We demonstrate how our conditional independence assumptions can be used to reduce the computational load of inference in Section 5. Experimental results are shown in Section 6 and finally some discussion and conclusions are presented in Section 7.

2. Related Work

In geostatistics, multiple output models are used to model the co-occurrence of minerals or pollutants in a spatial field. Many of the ideas for constructing covariance functions for multiple outputs have first appeared within the geostatistical literature, where they are known as linear models of coregionalization (LMC). We present the LMC and then review how several models proposed in the machine learning literature can be seen as special cases of the LMC.

2.1 The Linear Model of Coregionalization

The term linear model of coregionalization refers to models in which the outputs are expressed as *linear* combinations of independent random functions. If the independent random functions are Gaussian processes then the resulting model will also be a Gaussian process with a positive semi-definite covariance function. Consider a set of D output functions $\{f_d(\mathbf{x})\}_{d=1}^{D}$ where $\mathbf{x} \in \Re^p$ is the input domain. In a LMC each output function, $f_d(\mathbf{x})$, is expressed as (Journel and Huijbregts, 1978)

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} a_{d,q} u_q(\mathbf{x}). \tag{1}$$

Under the GP interpretation of the LMC, the functions $\{u_q(\mathbf{x})\}_{q=1}^{Q}$ are taken (without loss of generality) to be drawn from a zero-mean GP with $\mathrm{cov}[u_q(\mathbf{x}), u_{q'}(\mathbf{x}')] = k_q(\mathbf{x}, \mathbf{x}')$ if $q = q'$ and zero otherwise. Some of these base processes might have the same covariance, this is $k_q(\mathbf{x}, \mathbf{x}') = k_{q'}(\mathbf{x}, \mathbf{x}')$, but they would still be independently sampled. We can group together the base processes that share latent functions (Journel and Huijbregts, 1978; Goovaerts, 1997), allowing us to express a given output as

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} a_{d,q}^{i} u_q^{i}(\mathbf{x}), \tag{2}$$

where the functions $\left\{u_q^i(\mathbf{x})\right\}_{i=1}^{R_q}$, $i = 1, \ldots, R_q$, represent the latent functions that share the same covariance function $k_q(\mathbf{x}, \mathbf{x}')$. There are now Q groups of functions, each member of a group shares the same covariance, but is sampled independently.

In geostatistics it is common to simplify the analysis of these models by assuming that the processes $f_d(\mathbf{x})$ are stationary and ergodic (Cressie, 1993). The stationarity and ergodicity conditions are introduced so that the prediction stage can be realized through an optimal linear predictor using a single realization of the process (Cressie, 1993). Such linear predictors receive the general name of *cokriging*. The cross covariance between any two functions $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x})$ is given in terms of the covariance functions for $u_q^i(\mathbf{x})$

$$\operatorname{cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')] = \sum_{q=1}^{Q}\sum_{q'=1}^{Q}\sum_{i=1}^{R_q}\sum_{i'=1}^{R_q} a_{d,q}^i a_{d',q'}^{i'} \operatorname{cov}[u_q^i(\mathbf{x}), u_{q'}^{i'}(\mathbf{x}')].$$

Because of the independence of the latent functions $u_q^i(\mathbf{x})$, the above expression can be reduced to

$$\operatorname{cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')] = \sum_{q=1}^{Q}\sum_{i=1}^{R_q} a_{d,q}^i a_{d',q}^i k_q(\mathbf{x}, \mathbf{x}') = \sum_{q=1}^{Q} b_{d,d'}^q k_q(\mathbf{x}, \mathbf{x}'), \tag{3}$$

with $b_{d,d'}^q = \sum_{i=1}^{R_q} a_{d,q}^i a_{d',q}^i$.

For a number N of input vectors, let $\mathbf{f}_d$ be the vector of values from the output d evaluated at $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$. If each output has the same set of inputs the system is known as *isotopic*. In general, we can allow each output to be associated with a different set of inputs, $\mathbf{X}^{(d)} = \{\mathbf{x}_n^{(d)}\}_{n=1}^{N_d}$, this is known as *heterotopic*.[2] For notational simplicity, we restrict ourselves to the isotopic case, but our analysis can also be completed for heterotopic setups. The covariance matrix for $\mathbf{f}_d$ is obtained expressing Equation (3) as

$$\operatorname{cov}[\mathbf{f}_d, \mathbf{f}_{d'}] = \sum_{q=1}^{Q}\sum_{i=1}^{R_q} a_{d,q}^i a_{d',q}^i \mathbf{K}_q = \sum_{q=1}^{Q} b_{d,d'}^q \mathbf{K}_q,$$

where $\mathbf{K}_q \in \Re^{N\times N}$ has entries given by computing $k_q(\mathbf{x}, \mathbf{x}')$ for all combinations from $\mathbf{X}$. We now define $\mathbf{f}$ to be a stacked version of the outputs so that $\mathbf{f} = [\mathbf{f}_1^\top, \ldots, \mathbf{f}_D^\top]^\top$. We can now write the covariance matrix for the joint process over $\mathbf{f}$ as

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \sum_{q=1}^{Q} \mathbf{A}_q \mathbf{A}_q^\top \otimes \mathbf{K}_q = \sum_{q=1}^{Q} \mathbf{B}_q \otimes \mathbf{K}_q, \tag{4}$$

where the symbol $\otimes$ denotes the Kronecker product, $\mathbf{A}_q \in \Re^{D\times R_q}$ has entries $a_{d,q}^i$ and $\mathbf{B}_q = \mathbf{A}_q\mathbf{A}_q^\top \in \Re^{D\times D}$ has entries $b_{d,d'}^q$ and is known as the *coregionalization matrix*. The covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ is positive semi-definite as long as the coregionalization matrices $\mathbf{B}_q$ are positive semi-definite and $k_q(\mathbf{x}, \mathbf{x}')$ is a valid covariance function. By definition, coregionalization matrices $\mathbf{B}_q$ fulfill the positive semi-definiteness requirement. The covariance functions for the latent processes, $k_q(\mathbf{x}, \mathbf{x}')$, can simply be chosen from the wide variety of covariance functions (reproducing kernels) that are

2. These names come from geostatistics.

used for the single output case. Examples include the squared exponential (sometimes called the Gaussian kernel or RBF kernel) and the Matérn class of covariance functions (see Rasmussen and Williams, 2006, Chapter 4).

The linear model of coregionalization represents the covariance function as a product of the contributions of two covariance functions. One of the covariance functions models the dependence between the functions independently of the input vector $\mathbf{x}$, this is given by the coregionalization matrix $\mathbf{B}_q$, whilst the other covariance function models the input dependence independently of the particular set of functions $f_d(\mathbf{x})$, this is the covariance function $k_q(\mathbf{x}, \mathbf{x}')$.

We can understand the LMC by thinking of the functions having been generated as a two step process. Firstly we sample a set of independent processes from the covariance functions given by $k_q(\mathbf{x}, \mathbf{x}')$, taking R_q independent samples for each $k_q(\mathbf{x}, \mathbf{x}')$. We now have $R = \sum_{q=1}^{Q} R_q$ independently sampled functions. These functions are *instantaneously mixed*[3] in a linear fashion. In other words the output functions are derived by application of a scaling and a rotation to an output space of dimension D.

2.1.1 INTRINSIC COREGIONALIZATION MODEL

A simplified version of the LMC, known as the intrinsic coregionalization model (ICM) (Goovaerts, 1997), assumes that the elements $b^q_{d,d'}$ of the coregionalization matrix $\mathbf{B}_q$ can be written as $b^q_{d,d'} = \upsilon_{d,d'} b_q$. In other words, as a scaled version of the elements b_q which do not depend on the particular output functions $f_d(\mathbf{x})$. Using this form for $b^q_{d,d'}$, Equation (3) can be expressed as

$$\text{cov}[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')] = \sum_{q=1}^{Q} \upsilon_{d,d'} b_q k_q(\mathbf{x}, \mathbf{x}') = \upsilon_{d,d'} \sum_{q=1}^{Q} b_q k_q(\mathbf{x}, \mathbf{x}').$$

The covariance matrix for $\mathbf{f}$ takes the form

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \boldsymbol{\Upsilon} \otimes \mathbf{K}, \tag{5}$$

where $\boldsymbol{\Upsilon} \in \Re^{D \times D}$, with entries $\upsilon_{d,d'}$, and $\mathbf{K} = \sum_{q=1}^{Q} b_q \mathbf{K}_q$ is an equivalent valid covariance function.

The intrinsic coregionalization model can also be seen as a linear model of coregionalization where we have $Q = 1$. In such case, Equation (4) takes the form

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \mathbf{A}_1 \mathbf{A}_1^\top \otimes \mathbf{K}_1 = \mathbf{B}_1 \otimes \mathbf{K}_1, \tag{6}$$

where the coregionalization matrix $\mathbf{B}_1$ has elements $b^1_{d,d'} = \sum_{i=1}^{R_1} a^i_{d,1} a^i_{d',1}$. The value of R_1 determines the rank of the matrix $\mathbf{B}_1$.

As pointed out by Goovaerts (1997), the ICM is much more restrictive than the LMC since it assumes that each basic covariance $k_q(\mathbf{x}, \mathbf{x}')$ contributes equally to the construction of the autocovariances and cross covariances for the outputs.

3. The term instantaneous mixing is taken from blind source separation. Of course, if the underlying processes are not temporal but spatial, instantaneous is not being used in its original sense. However, it allows us to distinguish this mixing from convolutional mixing.

2.1.2 LINEAR MODEL OF COREGIONALIZATION IN MACHINE LEARNING

Several of the approaches to multiple output learning in machine learning based on kernels can be seen as examples of the linear model of coregionalization.

Semiparametric latent factor model. The semiparametric latent factor model (SLFM) proposed by Teh et al. (2005) turns out to be a simplified version of Equation (4). In particular, if $R_q = 1$ (see Equation 1), we can rewrite Equation (4) as

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \sum_{q=1}^{Q} \mathbf{a}_q \mathbf{a}_q^\top \otimes \mathbf{K}_q,$$

where $\mathbf{a}_q \in \Re^{D\times 1}$ with elements $a_{d,q}$. With some algebraic manipulations that exploit the properties of the Kronecker product[4] we can write

$$\mathbf{K}_{\mathbf{f},\mathbf{f}} = \sum_{q=1}^{Q} (\mathbf{a}_q \otimes \mathbf{I}_N)\mathbf{K}_q(\mathbf{a}_q^\top \otimes \mathbf{I}_N) = (\widetilde{\mathbf{A}} \otimes \mathbf{I}_N)\widetilde{\mathbf{K}}(\widetilde{\mathbf{A}}^\top \otimes \mathbf{I}_N),$$

where $\mathbf{I}_N$ is the N-dimensional identity matrix, $\widetilde{\mathbf{A}} \in \Re^{D\times Q}$ is a matrix with columns $\mathbf{a}_q$ and $\widetilde{\mathbf{K}} \in \Re^{QN\times QN}$ is a block diagonal matrix with blocks given by $\mathbf{K}_q$.

The functions $u_q(\mathbf{x})$ are considered to be latent factors and the semiparametric name comes from the fact that it is combining a nonparametric model, this is a Gaussian process, with a parametric linear mixing of the functions $u_q(\mathbf{x})$. The kernel for each basic process q, $k_q(\mathbf{x},\mathbf{x}')$, is assumed to be of Gaussian type with a different length scale per input dimension. For computational speed up the informative vector machine (IVM) is employed (Lawrence et al., 2003).

Multi-task Gaussian processes. The intrinsic coregionalization model has been employed in Bonilla et al. (2008) for multitask learning. We refer to this approach as multi-task Gaussian processes (MTGP). The covariance matrix is expressed as $\mathbf{K}_{\bar{\mathbf{f}}(\mathbf{x}),\bar{\mathbf{f}}(\mathbf{x}')} = K^f \otimes k(\mathbf{x},\mathbf{x}')$, with $\bar{\mathbf{f}}(\mathbf{x}) = [f_1(\mathbf{x}),\ldots,f_D(\mathbf{x})]^\top$, K^f being constrained positive semi-definite and $k(\mathbf{x},\mathbf{x}')$ a covariance function over inputs. It can be noticed that this expression has is equal to the one in (5), when it is evaluated for $\mathbf{x},\mathbf{x}' \in \mathbf{X}$. In Bonilla et al. (2008), K^f (equal to $\boldsymbol{\Upsilon}$ in Equation 5 or $\mathbf{B}_1$ in Equation 6) expresses the correlation between tasks or inter-task dependencies and it is represented through a probabilistic principal component analysis (PPCA) model. In turn, the spectral factorization in the PPCA model is replaced by an incomplete Cholesky decomposition to keep numerical stability, so that $K^f \approx \widetilde{\mathbf{L}}\widetilde{\mathbf{L}}^\top$, where $\widetilde{\mathbf{L}} \in \Re^{D\times R_1}$. An application of MTGP for obtaining the inverse dynamics of a robotic manipulator was presented in Chai et al. (2009).

It can be shown that if the outputs are considered to be noise-free, prediction using the intrinsic coregionalization model under an isotopic data case is equivalent to independent prediction over each output (Helterbrand and Cressie, 1994). This circumstance is also known as autokrigeability (Wackernagel, 2003) and it can also be seen as the cancellation of inter-task transfer (Bonilla et al., 2008).

Multi-output Gaussian processes. The intrinsic coregionalization model has been also used in Osborne et al. (2008). Matrix $\boldsymbol{\Upsilon}$ in Expression (5) is assumed to be of the spherical parametrisation kind, $\boldsymbol{\Upsilon} = \mathrm{diag}(\mathbf{e})\mathbf{S}^\top\mathbf{S}\,\mathrm{diag}(\mathbf{e})$, where $\mathbf{e}$ gives a description for the length scale of each output variable and $\mathbf{S}$ is an upper triangular matrix whose i-th column is associated with particular spherical

4. See Brookes (2005) for a nice overview.

coordinates of points in $\Re^i$ (for details see Osborne and Roberts, 2007, Section 3.4). Function $k(\mathbf{x}, \mathbf{x}')$ is represented through a Mátern kernel, where different parametrisations of the covariance allow the expression of periodic and non-periodic terms. Sparsification for this model is obtained using an IVM style approach.

Multi-task kernels in regularization theory. Kernels for multiple outputs have also been studied in the context of regularization theory. The approach is based mainly on the definition of kernels for multitask learning provided in Evgeniou and Pontil (2004) and Evgeniou et al. (2005), derived based on the theory of kernels for vector-valued functions. Let $\mathcal{D} = \{1, \ldots, D\}$. According to Evgeniou et al. (2005), the following lemma can be used to construct multitask kernels,

Lemma 1 *If G is a kernel on $\mathcal{T} \times \mathcal{T}$ and, for every $d \in \mathcal{D}$ there are prescribed mappings $\Phi_d : \mathcal{X} \to \mathcal{T}$ such that*

$$k_{d,d'}(\mathbf{x}, \mathbf{x}') = k((\mathbf{x}, d), (\mathbf{x}', d')) = G(\Phi_d(\mathbf{x}), \Phi_{d'}(\mathbf{x}')), \quad \mathbf{x}, \mathbf{x}' \in \Re^p,\ d, d' \in \mathcal{D},$$

then $k(\cdot)$ is a multitask or multioutput kernel.

A linear multitask kernel can be obtained if we set $\mathcal{T} = \Re^m$, $\Phi_d(\mathbf{x}) = \mathbf{C}_d\mathbf{x}$ with $\Phi_d \in \Re^m$ and $G : \Re^m \times \Re^m \to \Re$ as the polynomial kernel $G(\mathbf{z}, \mathbf{z}') = (\mathbf{z}^\top \mathbf{z}')^n$ with $n = 1$, leading to $k_{d,d'}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{C}_d^\top \mathbf{C}_{d'} \mathbf{x}'$. The lemma above can be seen as the result of applying kernel properties to the mapping $\Phi_d(\mathbf{x})$ (see Genton, 2001, p. 2). Notice that this corresponds to a generalization of the semiparametric latent factor model where each output is expressed through its own basic process acting over the linear transformation $\mathbf{C}_d\mathbf{x}$, this is, $u_d(\Phi_d(\mathbf{x})) = u_d(\mathbf{C}_d\mathbf{x})$. In general, it can be obtained from $f_d(\mathbf{x}) = \sum_{q=1}^{D} a_{d,q} u_q(\Phi_q(\mathbf{x}))$, where $a_{d,q} = 1$ if $d = q$ or zero, otherwise.

A more detailed analysis of the LMC and more connections with other methods in statistics and machine learning can be found in Álvarez et al. (2011b).

3. Convolution Processes for Multiple Outputs

The approaches introduced above all involve some form of instantaneous mixing of a series of independent processes to construct correlated processes. Instantaneous mixing has some limitations. If we wanted to model two output processes in such a way that one process was a blurred version of the other, we cannot achieve this through instantaneous mixing. We can achieve blurring through convolving a base process with a smoothing kernel. If the base process is a Gaussian process, it turns out that the convolved process is also a Gaussian process. We can therefore exploit convolutions to construct covariance functions (Barry and Ver Hoef, 1996; Ver Hoef and Barry, 1998; Higdon, 1998, 2002). A recent review of several extensions of this approach for the single output case is presented in Calder and Cressie (2007). Applications include the construction of nonstationary covariances (Higdon, 1998; Higdon et al., 1998; Fuentes, 2002a,b; Paciorek and Schervish, 2004) and spatiotemporal covariances (Wikle et al., 1998; Wikle, 2002, 2003).

Ver Hoef and Barry (1998) first, and Higdon (2002) later, suggested using convolutions to construct multiple output covariance functions. The approach was introduced to the machine learning community by Boyle and Frean (2005). Consider again a set of D functions $\{f_d(\mathbf{x})\}_{d=1}^{D}$. Now each function could be expressed through a convolution integral between a smoothing kernel, $\{G_d(\mathbf{x})\}_{d=1}^{D}$, and a latent function $u(\mathbf{x})$,

$$f_d(\mathbf{x}) = \int_{\mathcal{X}} G_d(\mathbf{x} - \mathbf{z}) u(\mathbf{z}) \mathrm{d}\mathbf{z}. \tag{7}$$

More generally, and in a similar way to the linear model of coregionalization, we can consider the influence of more than one latent function, $u_q^i(\mathbf{z})$, with $q = 1, \ldots, Q$ and $i = 1, \ldots, R_q$ to obtain

$$f_d(\mathbf{x}) = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}) u_q^i(\mathbf{z}) \mathrm{d}\mathbf{z}.$$

As in the LMC, there are Q groups of functions, each member of the group has the same covariance $k_q(\mathbf{x}, \mathbf{x}')$, but is sampled independently. Under the same independence assumptions used in the LMC, the covariance between $f_d(\mathbf{x})$ and $f_{d'}(\mathbf{x}')$ follows

$$\operatorname{cov}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right] = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}) \int_{\mathcal{X}} G_{d',q}^i(\mathbf{x}' - \mathbf{z}') k_q(\mathbf{z}, \mathbf{z}') \mathrm{d}\mathbf{z}' \mathrm{d}\mathbf{z}. \tag{8}$$

Specifying $G_{d,q}^i(\mathbf{x} - \mathbf{z})$ and $k_q(\mathbf{z}, \mathbf{z}')$ in (8), the covariance for the outputs $f_d(\mathbf{x})$ can be constructed indirectly. Note that if the smoothing kernels are taken to be the Dirac delta function such that,

$$G_{d,q}^i(\mathbf{x} - \mathbf{z}) = a_{d,q}^i \delta(\mathbf{x} - \mathbf{z}),$$

where $\delta(\cdot)$ is the Dirac delta function, the double integral is easily solved and the linear model of coregionalization is recovered. This matches to the concept of *instantaneous mixing* we introduced to describe the LMC. In a convolutional process the mixing is more general, for example the latent process could be smoothed for one output, but not smoothed for another allowing correlated output functions of different length scales.

The traditional approach to convolution processes in statistics and signal processing is to assume that the latent functions $u_q(\mathbf{z})$ are independent white Gaussian noise processes, $k_q(\mathbf{z}, \mathbf{z}') = \sigma_q^2 \delta(\mathbf{z} - \mathbf{z}')$. This allows us to simplify (8) as

$$\operatorname{cov}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right] = \sum_{q=1}^{Q} \sum_{i=1}^{R_q} \sigma_q^2 \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}) G_{d',q}^i(\mathbf{x}' - \mathbf{z}) \mathrm{d}\mathbf{z}.$$

In general, though, we can consider any type of latent process, for example, we could assume GPs for the latent functions with general covariances $k_q(\mathbf{z}, \mathbf{z}')$.

As well as this covariance across outputs, the covariance between the latent function, $u_q^i(\mathbf{z})$, and any given output, $f_d(\mathbf{x})$, can be computed,

$$\operatorname{cov}\left[f_d(\mathbf{x}), u_q^i(\mathbf{z})\right] = \int_{\mathcal{X}} G_{d,q}^i(\mathbf{x} - \mathbf{z}') k_q(\mathbf{z}', \mathbf{z}) \mathrm{d}\mathbf{z}'. \tag{9}$$

Additionally, we can corrupt each of the outputs of the convolutions with an independent process (which could also include a noise term), $w_d(\mathbf{x})$, to obtain

$$y_d(\mathbf{x}) = f_d(\mathbf{x}) + w_d(\mathbf{x}). \tag{10}$$

The covariance between two different outputs $y_d(\mathbf{x})$ and $y_{d'}(\mathbf{x}')$ is then recovered as

$$\operatorname{cov}\left[y_d(\mathbf{x}), y_{d'}(\mathbf{x}')\right] = \operatorname{cov}\left[f_d(\mathbf{x}), f_{d'}(\mathbf{x}')\right] + \operatorname{cov}\left[w_d(\mathbf{x}), w_{d'}(\mathbf{x}')\right] \delta_{d,d'},$$

where $\delta_{d,d'}$ is the Kronecker delta function.[5]

As mentioned before, Ver Hoef and Barry (1998) and Higdon (2002) proposed the direct use of convolution processes for constructing multiple output Gaussian processes. Lawrence et al. (2007) arrive at a similar construction from solving a physical model: a first order differential equation (see also Gao et al., 2008). This idea of using physical models to inspire multiple output systems has been further extended in Álvarez et al. (2009) who give examples using the heat equation and a second order system. A different approach using Kalman Filtering ideas has been proposed in Calder (2003, 2007). Calder proposed a model that incorporates dynamical systems ideas to the process convolution formalism. Essentially, the latent processes are of two types: random walks and independent cyclic second-order autoregressions. With this formulation, it is possible to construct a multivariate output process using convolutions over these latent processes. Particular relationships between outputs and latent processes are specified using a special transformation matrix ensuring that the outputs are invariant under invertible linear transformations of the underlying factor processes (this matrix is similar in spirit to the sensitivity matrix of Lawrence et al. (2007) and it is given a particular form so that not all latent processes affect the whole set of outputs).

Bayesian kernel methods. The convolution process is closely related to the Bayesian kernel method (Pillai et al., 2007; Liang et al., 2009) for constructing reproducible kernel Hilbert spaces (RKHS), assigning priors to signed measures and mapping these measures through integral operators. In particular, define the following space of functions,

$$\mathcal{F} = \left\{ f \middle| f(x) = \int_{\mathcal{X}} G(x,z)\gamma(\mathrm{d}z),\, \gamma \in \Gamma \right\},$$

for some space $\Gamma \subseteq \mathcal{B}(\mathcal{X})$ of signed Borel measures. In Pillai et al. (2007, Proposition 1), the authors show that for $\Gamma = \mathcal{B}(\mathcal{X})$, the space of all signed Borel measures, $\mathcal{F}$ corresponds to a RKHS. Examples of these measures that appear in the form of stochastic processes include Gaussian processes, Dirichlet processes and Lévy processes. This framework can be extended for the multiple output case, expressing the outputs as

$$f_d(x) = \int_{\mathcal{X}} G_d(x,z)\gamma(\mathrm{d}z).$$

The analysis of the mathematical properties of such spaces of functions is beyond the scope of this paper and is postponed for future work.

Other connections of the convolution process approach with methods in statistics and machine learning are further explored in Álvarez et al. (2011b).

A general purpose convolution kernel for multiple outputs. A simple general purpose kernel for multiple outputs based on the convolution integral can be constructed assuming that the kernel smoothing function, $G_{d,q}(\mathbf{x})$, and the covariance for the latent function, $k_q(\mathbf{x},\mathbf{x}')$, follow both a Gaussian form. A similar construction using a Gaussian form for $G(\mathbf{x})$ and a white noise process for $u(\mathbf{x})$ has been used in Paciorek and Schervish (2004) to propose a nonstationary covariance function in single output regression. It has also been used in Boyle and Frean (2005) as an example of constructing dependent Gaussian processes.

The kernel smoothing function is given as

$$G_{d,q}(\mathbf{x}) = S_{d,q}\mathcal{N}(\mathbf{x}|\mathbf{0}, \mathbf{P}_d^{-1}),$$

5. We have slightly abused of the delta notation to indicate the Kronecker delta for discrete arguments and the Dirac function for continuous arguments. The particular meaning should be understood from the context.

where $S_{d,q}$ is a variance coefficient that depends both on the output d and the latent function q and $\mathbf{P}_d$ is the precision matrix associated to the particular output d. The covariance function for the latent process is expressed as

$$k_q(\mathbf{x},\mathbf{x}') = \mathcal{N}(\mathbf{x}-\mathbf{x}'|\mathbf{0},\mathbf{\Lambda}_q^{-1}),$$

with $\mathbf{\Lambda}_q$ the precision matrix of the latent function q.

Expressions for the kernels are obtained applying systematically the identity for the product of two Gaussian distributions. Let $\mathcal{N}(\mathbf{x}|\boldsymbol{\mu},\mathbf{P}^{-1})$ denote a Gaussian for $\mathbf{x}$, then

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1,\mathbf{P}_1^{-1})\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_2,\mathbf{P}_2^{-1}) = \mathcal{N}(\boldsymbol{\mu}_1|\boldsymbol{\mu}_2,\mathbf{P}_1^{-1}+\mathbf{P}_2^{-1})\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c,\mathbf{P}_c^{-1}), \tag{11}$$

where $\boldsymbol{\mu}_c = (\mathbf{P}_1+\mathbf{P}_2)^{-1}(\mathbf{P}_1\boldsymbol{\mu}_1+\mathbf{P}_2\boldsymbol{\mu}_2)$ and $\mathbf{P}_c^{-1} = (\mathbf{P}_1+\mathbf{P}_2)^{-1}$. For all integrals we assume that $\mathcal{X} = \Re^p$. Using these forms for $G_{d,q}(\mathbf{x})$ and $k_q(\mathbf{x},\mathbf{x}')$, expression (8) (with $R_q = 1$) can be written as

$$k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = \sum_{q=1}^{Q} S_{d,q}S_{d',q}\int_{\mathcal{X}}\mathcal{N}(\mathbf{x}-\mathbf{z}|\mathbf{0},\mathbf{P}_d^{-1})\int_{\mathcal{X}}\mathcal{N}(\mathbf{x}'-\mathbf{z}'|\mathbf{0},\mathbf{P}_{d'}^{-1})\mathcal{N}(\mathbf{z}-\mathbf{z}'|\mathbf{0},\mathbf{\Lambda}_q^{-1})\mathrm{d}\mathbf{z}'\mathrm{d}\mathbf{z}.$$

Since the Gaussian covariance is stationary, we can write it as $\mathcal{N}(\mathbf{x}-\mathbf{x}'|\mathbf{0},\mathbf{P}^{-1}) = \mathcal{N}(\mathbf{x}'-\mathbf{x}|\mathbf{0},\mathbf{P}^{-1}) = \mathcal{N}(\mathbf{x}|\mathbf{x}',\mathbf{P}^{-1}) = \mathcal{N}(\mathbf{x}'|\mathbf{x},\mathbf{P}^{-1})$. Using the identity in Equation (11) twice, we get

$$k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = \sum_{q=1}^{Q} S_{d,q}S_{d',q}\mathcal{N}(\mathbf{x}-\mathbf{x}'|\mathbf{0},\mathbf{P}_d^{-1}+\mathbf{P}_{d'}^{-1}+\mathbf{\Lambda}_q^{-1}). \tag{12}$$

For a high value of the input dimension, p, the term $1/[(2\pi)^{p/2}|\mathbf{P}_d^{-1}+\mathbf{P}_{d'}^{-1}+\mathbf{\Lambda}_q^{-1}|^{1/2}]$ in each of the Gaussian's normalization terms will dominate, making values go quickly to zero. We can fix this problem, by scaling the outputs using the factors $1/[(2\pi)^{p/4}|2\mathbf{P}_d^{-1}+\mathbf{\Lambda}_q^{-1}|^{1/4}]$ and $1/[(2\pi)^{p/4}|2\mathbf{P}_{d'}^{-1}+\mathbf{\Lambda}_q^{-1}|^{1/4}]$. Each of these scaling factors correspond to the standard deviation associated to $k_{f_d,f_d}(\mathbf{x},\mathbf{x})$ and $k_{f_{d'},f_{d'}}(\mathbf{x},\mathbf{x})$.

Equally for the covariance $\mathrm{cov}[f_d(\mathbf{x}),u_q(\mathbf{x}'))]$ in Equation (9), we obtain

$$k_{f_d,u_q}(\mathbf{x},\mathbf{x}') = S_{d,q}\mathcal{N}(\mathbf{x}-\mathbf{x}'|\mathbf{0},\mathbf{P}_d^{-1}+\mathbf{\Lambda}_q^{-1}).$$

Again, this covariance must be standardized when working in higher dimensions.

4. Hyperparameter Learning

Given the convolution formalism, we can construct a full GP over the set of outputs. The likelihood of the model is given by

$$p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}|\mathbf{0},\mathbf{K}_{\mathbf{f},\mathbf{f}}+\mathbf{\Sigma}), \tag{13}$$

where $\mathbf{y} = \left[\mathbf{y}_1^\top,\ldots,\mathbf{y}_D^\top\right]^\top$ is the set of output functions with $\mathbf{y}_d = [y_d(\mathbf{x}_1),\ldots,y_d(\mathbf{x}_N)]^\top$; $\mathbf{K}_{\mathbf{f},\mathbf{f}} \in \Re^{DN\times DN}$ is the covariance matrix arising from the convolution. It expresses the covariance of each data point at every other output and data point and its elements are given by $\mathrm{cov}[f_d(\mathbf{x}),f_{d'}(\mathbf{x}')]$ in (8). The term $\mathbf{\Sigma}$ represents the covariance associated with the independent processes in (10), $w_d(\mathbf{x})$. It could contain structure, or alternatively could simply represent noise that is independent across

the data points. The vector $\boldsymbol{\theta}$ refers to the hyperparameters of the model. For exposition we will focus on the isotopic case (although our implementations allow heterotopic modeling), so we have a matrix $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ which is the common set of training input vectors at which the covariance is evaluated.

The predictive distribution for a new set of input vectors $\mathbf{X}_*$ is (Rasmussen and Williams, 2006)

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

where we have used $\mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*}$ as a compact notation to indicate when the covariance matrix is evaluated at the inputs $\mathbf{X}_*$, with a similar notation for $\mathbf{K}_{\mathbf{f}_*,\mathbf{f}}$. Learning from the log-likelihood involves the computation of the inverse of $\mathbf{K}_{\mathbf{f},\mathbf{f}} + \boldsymbol{\Sigma}$ giving the problematic complexity of $O(N^3D^3)$. Once the parameters have been learned, prediction is $O(ND)$ for the predictive mean and $O(N^2D^2)$ for the predictive variance.

As we have mentioned before, the main focus of this paper is to present some efficient approximations for the multiple output convolved Gaussian Process. Given the methods presented before, we now show an application that benefits from the non-instantaneous mixing element brought by the convolution process framework.

Comparison between instantaneous mixing and non-instantaneous mixing for regression in genes expression data. Microarray studies have made the simultaneous measurement of mRNA from thousands of genes practical. Transcription is governed by the presence or absence of transcription factor (TF) proteins that act as switches to turn on and off the expression of the genes. Most of these methods are based on assuming that there is an instantaneous linear relationship between the gene expression and the protein concentration. We compare the performance of the intrinsic coregionalization model (Section 2.1.1) and the convolved GPs for two independent time series or replicas of 12 time points collected hourly throughout Drosophila embryogenesis in wild-type embryos (Tomancak et al., 2002). For preprocessing the data, we follow Honkela et al. (2010). We concentrate on a particular transcription factor protein, namely `twi`, and the genes associated with it. The information about the network connections is obtained from the ChIP-chip experiments. This particular TF is key regulator of mesoderm and muscle development in Drosophila (Zinzen et al., 2009).

After preprocessing the data, we end up with a data set of 1621 genes with expression data for $N = 12$ time points. It is believed that this set of genes are regulated by at least the `twi` transcription factor. For each one of these genes, we have access to 2 replicas. We randomly select $D = 50$ genes from replica 1 for training a full multiple output GP model based on either the LMC framework or the convolved GP framework. The corresponding 50 genes of replica 2 are used for testing and results are presented in terms of the standardized mean square error (SMSE) and the mean standardized log loss (MSLL) as defined in Rasmussen and Williams (2006).[6] The parameters of both the LMC and the convolved GPs are found through the maximization of the marginal likelihood in Equation (13). We repeated the experiment 10 times using a different set of 50 genes each time. We also repeated the experiment selecting the 50 genes for training from replica 2 and the corresponding 50 genes of replica 1 for testing.

6. The definitions for the SMSE and the MSLL we have used here are slightly different from the ones provided in Rasmussen and Williams (2006). Instead of comparing against a Gaussian with a global mean and variance computed from all the outputs in the training data, we compare against a Gaussian with local means and local variances computed from the training data associated to each output.

We are interested in a reduced representation of the data so we assume that $Q = 1$ and $R_q = 1$, for the LMC and the convolved multiple output GP in Equations (2) and (8), respectively. For the LMC model, we follow Bonilla et al. (2008) and assume an incomplete Cholesky decomposition for $\mathbf{B}_1 = \widetilde{L}\widetilde{L}^\top$, where $\widetilde{L} \in \Re^{50\times 1}$ and as the basic covariance $k_q(\mathbf{x}, \mathbf{x}')$ we assume the squared exponential covariance function (p. 83, Rasmussen and Williams, 2006). For the convolved multiple output GP we employ the covariance described in Section 3, Equation (12), with the appropriate scaling factors.

Train set	Test set	Method	Average SMSE	Average MSLL
Replica 1	Replica 2	LMC	0.6069 ± 0.0294	-0.2687 ± 0.0594
		CMOC	0.4859 ± 0.0387	-0.3617 ± 0.0511
Replica 2	Replica 1	LMC	0.6194 ± 0.0447	-0.2360 ± 0.0696
		CMOC	0.4615 ± 0.0626	-0.3811 ± 0.0748

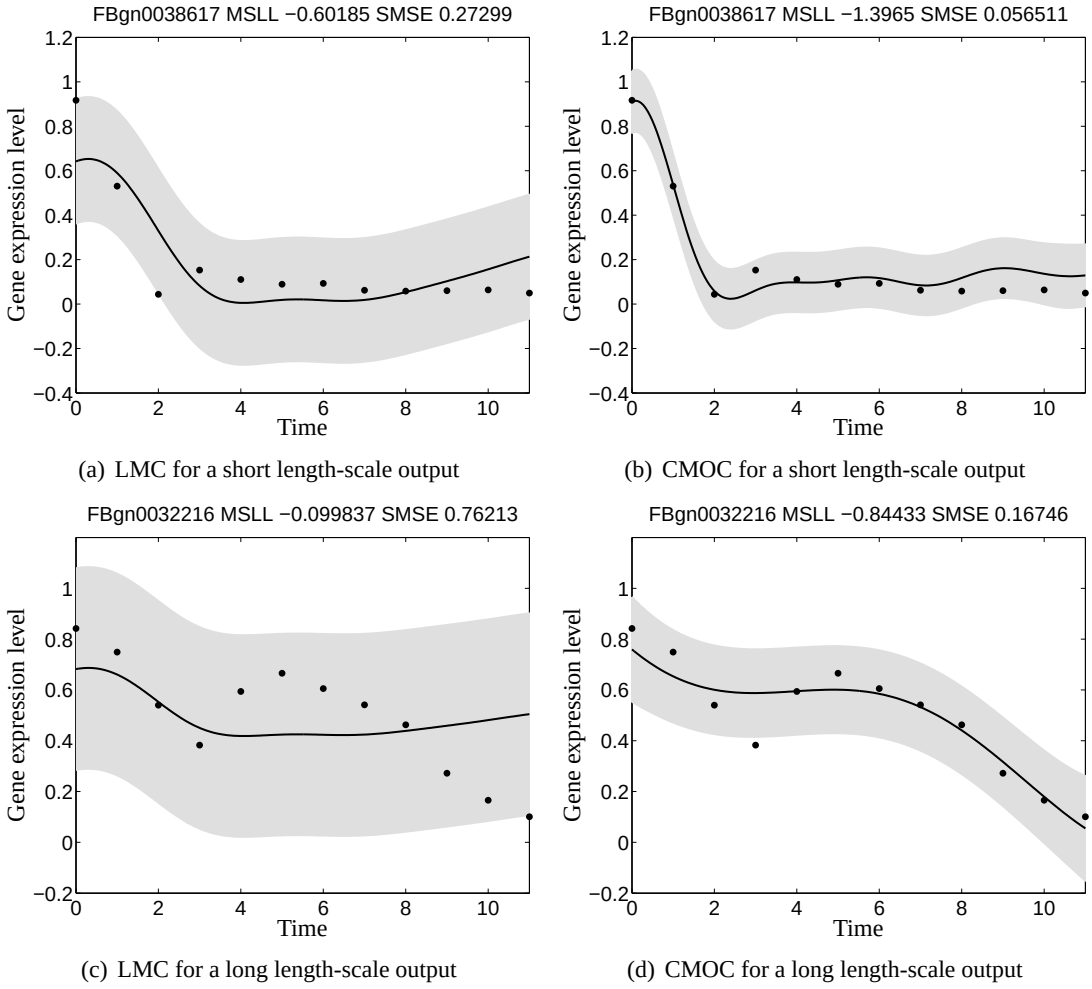
Table 1: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the gene expression data for 50 outputs. CMOC stands for convolved multiple output covariance. The experiment was repeated ten times with a different set of 50 genes each time. Table includes the value of one standard deviation over the ten repetitions. More negative values of MSLL indicate better models.

Table 1 shows the results of both methods over the test set for the two different replicas. It can be seen that the convolved multiple output covariance (appearing as CMOC in the table), outperforms the LMC covariance both in terms of SMSE and MSLL.

Figure 1 shows the prediction made over the test set (replica 2 in this case) by the two models for two particular genes, namely FBgn0038617 (Figure 1, first row) and FBgn0032216 (Figure 1, second row). The black dots in the figures represent the gene expression data of the particular genes. Figures 1(a) and 1(c) show the response of the LMC and Figures 1(b) and 1(d) show the response of the convolved multiple output covariance. It can be noticed from the data that the two genes differ in their responses to the action of the transcription factor, that is, while gene FBgn0038617 has a rapid decay around time 2 and becomes relatively constant for the rest of the time interval, gene FBgn0032216 has a smoother response within the time frame. The linear model of coregionalization is driven by a latent function with a length-scale that is shared across the outputs. Notice from Figures 1(a) and 1(c) that the length-scale for both responses is the same. On the other hand, due-to the non-instantaneous mixing of the latent function, the convolved multiple output framework, allows the description of each output using its own length-scale, which gives an added flexibility for describing the data.

Table 2 (first four rows) shows the performances of both models for the genes of Figure 1. CMOC outperforms the linear model of coregionalization for both genes in terms of SMSE and MSLL.

A similar analysis can be made for Figures 2(a), 2(b), 2(c) and 2(d). In this case, the test set is replica 1 and we have chosen two different genes, FBgn0010531 and FBgn0004907 with a similar behavior. Table 2 (last four rows) also highlights the performances of both models for the genes of Figure 2. Again, CMOC outperforms the linear model of coregionalization for both genes and in terms of SMSE and MSLL.



(a) LMC for a short length-scale output

(b) CMOC for a short length-scale output

(c) LMC for a long length-scale output

(d) CMOC for a long length-scale output

Figure 1: Predictive mean and variance for genes FBgn0038617 (first row) and FBgn0032216 (second row) using the linear model of coregionalization in Figures 1(a) and 1(c) and the convolved multiple-output covariance in Figures 1(b) and 1(d), with $Q = 1$ and $R_q = 1$. The training data comes from replica 1 and the testing data from replica 2. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure and appear also in Table 2. The adjectives "short" and "long" given to the length-scales in the captions of each figure, must be understood like relative to each other.

Having said this, we can argue that the performance of the LMC model can be improved by either increasing the value of Q or the value R_q, or both. For the intrinsic coregionalization model, we would fix the value of $Q = 1$ and increase the value of R_1. Effectively, we would be increasing the rank of the coregionalization matrix $\mathbf{B}_1$, meaning that more latent functions sampled from the same covariance function are being used to explain the data. In a extreme case in which each output has its own length scale, this translates into equating the number of latent functions to the number

Test replica	Test genes	Method	SMSE	MSLL
Replica 2	FBgn0038617	LMC	0.2729	−0.6018
		CMOC	0.0565	−1.3965
	FBgn0032216	LMC	0.7621	−0.0998
		CMOC	0.1674	−0.8443
Replica 1	FBgn0010531	LMC	0.2572	−0.5699
		CMOC	0.0446	−1.3434
	FBgn0004907	LMC	0.4984	−0.3069
		CMOC	0.0971	−1.0841

Table 2: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the genes in Figures 1 and 2 for LMC and CMOC. Genes FBgn0038617 and FBgn0010531 have a shorter length-scale when compared to genes FBgn0032216 and FBgn0004907.

of outputs, or in other words assuming a full rank for the matrix $\mathbf{B}_1$. This leads to the need of estimating the matrix $\mathbf{B}_1 \in \Re^{D\times D}$, that might be problematic if D is high. For the semiparametric latent factor model, we would fix the value of $R_q = 1$ and increase Q, the number of latent functions sampled from Q different GPs. Again, in the extreme case of each output having its own length-scale, we might need to estimate a matrix $\widetilde{\mathbf{A}} \in \Re^{D\times D}$, which could be problematic for a high value of outputs. In a more general case, we could also combine values of $Q > 1$ and $R_q > 1$. We would need then, to find values of Q and R_q that fit the different outputs with different length scales.

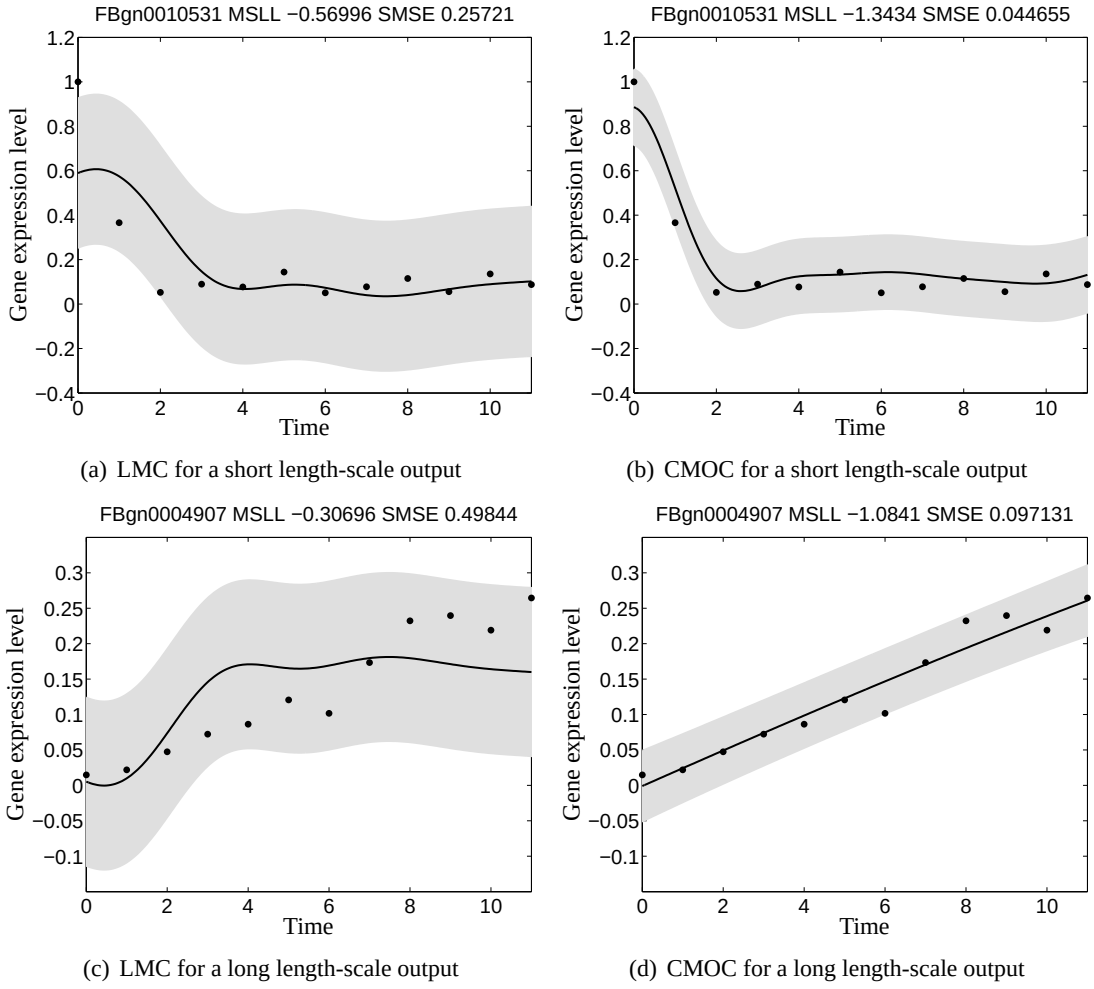
In practice though, we will see in the experimental section, that both the linear model of coregionalization and the convolved multiple output GPs can perform equally well in some data sets. However, the convolved covariance could offer an explanation of the data through a simpler model or converge to the LMC, if needed.

5. Efficient Approximations for Convolutional Processes

Assuming that the double integral in Equation (8) is tractable, the principle challenge for the convolutional framework is computing the inverse of the covariance matrix associated with the outputs. For D outputs, each having N data points, the inverse has computational complexity $O(D^3N^3)$ and associated storage of $O(D^2N^2)$. We show how through making specific conditional independence assumptions, inspired by the model structure (Álvarez and Lawrence, 2009), we arrive at a efficient approximation similar in form to the partially independent training conditional model (PITC, see Quiñonero-Candela and Rasmussen, 2005). The relationship with PITC then inspires us to make further conditional independence assumptions.

5.1 Latent Functions as Conditional Means

For notational simplicity, we restrict the analysis of the approximations to one latent function $u(\mathbf{x})$. The key to all approximations is based on the form we assume for the latent functions. From the perspective of a generative model, Equation (7) can be interpreted as follows: first we draw a sample from the Gaussian process prior $p(u(\mathbf{z}))$ and then solve the integral for each of the outputs $f_d(\mathbf{x})$ involved. Uncertainty about $u(\mathbf{z})$ is also propagated through the convolution transform.



(a) LMC for a short length-scale output

(b) CMOC for a short length-scale output

(c) LMC for a long length-scale output

(d) CMOC for a long length-scale output

Figure 2: Predictive mean and variance for genes FBgn0010531 (first row) and FBgn0004907 (second row) using the linear model of coregionalization in Figures 2(a) and 2(c), and the convolved multiple-output covariance in Figures 2(b) and 2(d), with $Q = 1$ and $R_q = 1$. The difference with Figure 1 is that now the training data comes from replica 2 while the testing data comes from replica 1. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure.

For the set of approximations, instead of drawing a sample from $u(\mathbf{z})$, we first draw a sample from a finite representation of $u(\mathbf{z})$, $\mathbf{u}(\mathbf{Z}) = [u(\mathbf{z}_1), \ldots, u(\mathbf{z}_K)]^\top$, where $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^K$ is the set of input vectors at which $u(\mathbf{z})$ is evaluated. Due to the properties of a Gaussian process, $p(\mathbf{u}(\mathbf{Z}))$ follows a multivariate Gaussian distribution. Conditioning on $\mathbf{u}(\mathbf{Z})$, we next sample from the conditional prior $p(u(\mathbf{z})|\mathbf{u}(\mathbf{Z}))$ and use this function to solve the convolution integral for each $f_d(\mathbf{x})$.[7] Under

7. For simplicity in the notation, we just write $\mathbf{u}$ to refer to $\mathbf{u}(\mathbf{Z})$.

this generative approach, we can approximate each function $f_d(\mathbf{x})$ using

$$f_d(\mathbf{x}) \approx \int_{\mathcal{X}} G_d(\mathbf{x}-\mathbf{z})\,\mathrm{E}\left[u(\mathbf{z})|\mathbf{u}\right]\mathrm{d}\mathbf{z}. \tag{14}$$

Replacing $u(\mathbf{z})$ for $\mathrm{E}\left[u(\mathbf{z})|\mathbf{u}\right]$ is a reasonable approximation as long as $u(\mathbf{z})$ is a smooth function so that the infinite dimensional object $u(\mathbf{z})$ can be summarized by $\mathbf{u}$. Figure 3 shows a cartoon example of the quality of the approximations for two outputs as the size of the set $\mathbf{Z}$ increases. The first column represents the conditional prior $p\left(u(\mathbf{z})|\mathbf{u}\right)$ for a particular choice of $u(\mathbf{z})$. The second and third columns represent the outputs $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ obtained when using Equation (14).

Using expression (14), the likelihood function for $\mathbf{f}$ follows

$$p(\mathbf{f}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{f}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right), \tag{15}$$

where $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is the covariance matrix between the samples from the latent function $\mathbf{u}(\mathbf{Z})$, with elements given by $k_{u,u}(\mathbf{z},\mathbf{z}')$ and $\mathbf{K}_{\mathbf{f},\mathbf{u}} = \mathbf{K}_{\mathbf{u},\mathbf{f}}^{\top}$ is the cross-covariance matrix between the latent function $u(\mathbf{z})$ and the outputs $f_d(\mathbf{x})$, with elements $\mathrm{cov}\left[f_d(\mathbf{x}), u(\mathbf{z})\right]$ in (9).

Given the set of points $\mathbf{u}$, we can have different assumptions about the uncertainty of the outputs in the likelihood term. For example, we could assume that the outputs are independent or uncorrelated, keeping only the uncertainty involved for each output in the likelihood term. Another approximation assumes that the outputs are deterministic, this is $\mathbf{K}_{\mathbf{f},\mathbf{f}} = \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$. The only uncertainty left would be due to the prior $p(\mathbf{u})$. Next, we present different approximations of the covariance of the likelihood that lead to a reduction in computational complexity.

5.1.1 PARTIAL INDEPENDENCE

We assume that the individual outputs in $\mathbf{f}$ are independent given the latent function $\mathbf{u}$, leading to the following expression for the likelihood

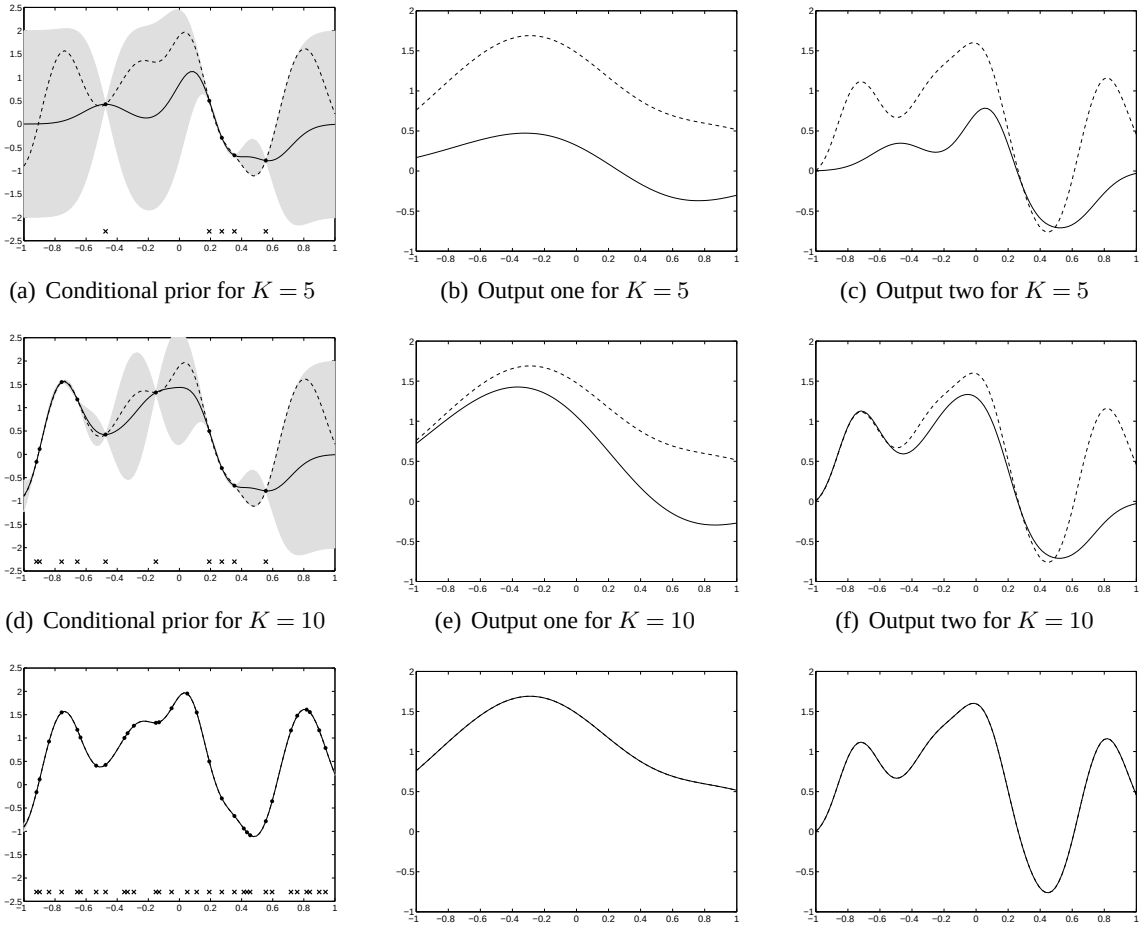
$$p(\mathbf{f}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \prod_{d=1}^{D} p(\mathbf{f}_d|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \prod_{d=1}^{D} \mathcal{N}\left(\mathbf{f}|\mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_d}\right).$$

We rewrite this product of multivariate Gaussians as a single Gaussian with a block diagonal covariance matrix, including the uncertainty about the independent processes

$$p(\mathbf{y}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{y}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{D}+\boldsymbol{\Sigma}\right) \tag{16}$$

where $\mathbf{D} = \mathrm{blockdiag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right]$, and we have used the notation $\mathrm{blockdiag}\left[\mathbf{G}\right]$ to indicate that the block associated with each output of the matrix $\mathbf{G}$ should be retained, but all other elements should be set to zero. We can also write this as $\mathbf{D} = \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right] \odot \mathbf{M}$ where $\odot$ is the Hadamard product and $\mathbf{M} = \mathbf{I}_D \otimes \mathbf{1}_N$, $\mathbf{1}_N$ being the $N \times N$ matrix of ones. We now marginalize the values of the samples from the latent function by using its process prior, this means $p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{u}|\mathbf{0},\mathbf{K}_{\mathbf{u},\mathbf{u}})$. This leads to the following marginal likelihood,

$$p(\mathbf{y}|\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \int p(\mathbf{y}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta})p(\mathbf{u}|\mathbf{Z})\mathrm{d}\mathbf{u} = \mathcal{N}\left(\mathbf{y}|\mathbf{0},\mathbf{D}+\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}+\boldsymbol{\Sigma}\right). \tag{17}$$



(a) Conditional prior for $K = 5$ (b) Output one for $K = 5$ (c) Output two for $K = 5$

(d) Conditional prior for $K = 10$ (e) Output one for $K = 10$ (f) Output two for $K = 10$

(g) Conditional prior for $K = 30$ (h) Output one for $K = 30$ (i) Output two for $K = 30$

Figure 3: Conditional prior and two outputs for different values of K. The first column, Figures 3(a), 3(d) and 3(g), shows the mean and confidence intervals of the conditional prior distribution using one input function and two output functions. The dashed line represents one sample from the prior. Conditioning over a few points of this sample, shown as black dots, the conditional mean and conditional covariance are computed. The solid line represents the conditional mean and the shaded region corresponds to 2 standard deviations away from the mean. The second column, 3(b), 3(e) and 3(h), shows the solution to Equation (7) for output one using the sample from the prior (dashed line) and the conditional mean (solid line), for different values of K. The third column, 3(c), 3(f) and 3(i), shows the solution to Equation (7) for output two, again for different values of K.

Notice that, compared to (13), the full covariance matrix $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ has been replaced by the low rank covariance $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}$ in all entries except in the diagonal blocks corresponding to $\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d}$. Depending on our choice of K, the inverse of the low rank approximation to the covariance is either dominated by a $O(DN^3)$ term or a $O(K^2DN)$ term. Storage of the matrix is $O(N^2D) + O(NDK)$.

Note that if we set $K = N$ these reduce to $O(N^3D)$ and $O(N^2D)$ respectively. Rather neatly this matches the computational complexity of modeling the data with D independent Gaussian processes across the outputs.

The functional form of (17) is almost identical to that of the partially independent training conditional (PITC) approximation (Quiñonero-Candela and Rasmussen, 2005) or the partially independent conditional (PIC) approximation (Quiñonero-Candela and Rasmussen, 2005; Snelson and Ghahramani, 2007), with the samples we retain from the latent function providing the same role as the *inducing values* in the PITC or PIC.[8] This is perhaps not surprising given that the PI(T)C approximations are also derived by making conditional independence assumptions. A key difference is that in PI(T)C it is not obvious which variables should be grouped together when making these conditional independence assumptions; here it is clear from the structure of the model that each of the outputs should be grouped separately.

5.1.2 FULL INDEPENDENCE

We can be inspired by the analogy of our approach to the PI(T)C approximation and consider a more radical factorization of the likelihood term. In the fully independent training conditional (FITC) approximation or the fully independent conditional (FIC) approximation (Snelson and Ghahramani, 2006, 2007), a factorization across the data points is assumed. For us that would lead to the following expression for the conditional distribution of the output functions given the inducing variables,

$$p(\mathbf{f}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\theta}) = \prod_{d=1}^{D}\prod_{n=1}^{N} p(f_{n,d}|\mathbf{u}, \mathbf{Z}, \mathbf{X}, \boldsymbol{\theta}),$$

which can be expressed through (16) with $\mathbf{D} = \text{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right] = \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right] \odot \mathbf{M}$, with $\mathbf{M} = \mathbf{I}_D \otimes \mathbf{I}_N$ or simply $\mathbf{M} = \mathbf{I}_{DN}$. The marginal likelihood, including the uncertainty about the independent processes, is given by Equation (17) with the diagonal form for $\mathbf{D}$. Training with this approximated likelihood reduces computational complexity to $O(K^2DN)$ and the associated storage to $O(KDN)$.

5.1.3 DETERMINISTIC LIKELIHOOD

In Quiñonero-Candela and Rasmussen (2005), the relationship between the projected process approximation (Csató and Opper, 2001; Seeger et al., 2003) and the FI(T)C and PI(T)C approximations is elucidated. They show that if, given the set of values $\mathbf{u}$, the outputs are assumed to be deterministic, the likelihood term of Equation (15) can be simplified as

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

Marginalizing with respect to the latent function using $p(\mathbf{u}|\mathbf{Z}) = \mathcal{N}(\mathbf{u}|\mathbf{0}, \mathbf{K}_{\mathbf{u},\mathbf{u}})$ and including the uncertainty about the independent processes, we obtain the marginal likelihood as

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

8. We refer to both PITC and PIC by PI(T)C.

In other words, we can approximate the full covariance $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ using the low rank approximation $\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}$. Using this new marginal likelihood to estimate the parameters $\boldsymbol{\theta}$ reduces computational complexity to $O(K^2DN)$. The approximation obtained has similarities with the projected latent variables (PLV) method also known as the projected process approximation (PPA) or the deterministic training conditional (DTC) approximation (Csató and Opper, 2001; Seeger et al., 2003; Quiñonero-Candela and Rasmussen, 2005; Rasmussen and Williams, 2006).

5.1.4 ADDITIONAL INDEPENDENCE ASSUMPTIONS

As mentioned before, we can consider different conditional independence assumptions for the likelihood term. One further assumption that is worth mentioning considers conditional independencies across data points and dependence across outputs. This would lead to the following likelihood term

$$p(\mathbf{f}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\bar{\mathbf{f}}_n|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}),$$

where $\bar{\mathbf{f}}_n = [f_1(\mathbf{x}_n), f_2(\mathbf{x}_n), \ldots, f_D(\mathbf{x}_n)]^{\top}$. We can use again Equation (16) to express the likelihood. In this case, though, the matrix $\mathbf{D}$ is a partitioned matrix with blocks $\mathbf{D}_{d,d'} \in \Re^{N\times N}$ and each block $\mathbf{D}_{d,d'}$ would be given as $\mathbf{D}_{d,d'} = \text{diag}\left[\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_{d'}}\right]$. For cases in which $D > N$, that is, the number of outputs is greater than the number of data points, this approximation may be more accurate than the one obtained with the partial independence assumption. For cases where $D < N$ it may be less accurate, but faster to compute.[9]

5.2 Posterior and Predictive Distributions

Combining the likelihood term for each approximation with $p(\mathbf{u}|\mathbf{Z})$ using Bayes' theorem, the posterior distribution over $\mathbf{u}$ is obtained as

$$p(\mathbf{u}|\mathbf{y},\mathbf{X},\mathbf{Z},\boldsymbol{\theta}) = \mathcal{N}\left(\mathbf{u}|\mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{y}, \mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{u}}\right), \tag{18}$$

where $\mathbf{A} = \mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}$ and $\mathbf{D}$ follows a particular form according to the different approximations: for partial independence it equals $\mathbf{D} = \text{blockdiag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right]$; for full independence it is $\mathbf{D} = \text{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right]$ and for the deterministic likelihood, $\mathbf{D} = \mathbf{0}$.

For computing the predictive distribution we have two options, either use the posterior for $\mathbf{u}$ and the approximated likelihoods or the posterior for $\mathbf{u}$ and the likelihood of Equation (15), that corresponds to the likelihood of the model without any approximations. The difference between both options is reflected in the covariance for the predictive distribution. Quiñonero-Candela and Rasmussen (2005) proposed a taxonomy of different approximations according to the type of likelihood used for the predictive distribution, in the context of single output Gaussian processes.

In this paper, we opt for the posterior for $\mathbf{u}$ and the likelihood of the model without any approximations. If we choose the exact likelihood term in Equation (15) (including the noise term), the

9. Notice that if we work with the block diagonal matrices $\mathbf{D}_{d,d'}$, we would need to invert the full matrix $\mathbf{D}$. However, since the blocks $\mathbf{D}_{d,d'}$ are diagonal matrices themselves, the inversion can be done efficiently using, for example, a block Cholesky decomposition. Furthermore, we would be restricted to work with isotopic input spaces. Alternatively, we could rearrange the elements of the matrix $\mathbf{D}$ so that the blocks of the main diagonal are the covariances associated with the vectors $\bar{\mathbf{f}}_n$.

predictive distribution is expressed through the integration of the likelihood term evaluated at $\mathbf{X}_*$, with (18), giving

$$p(\mathbf{y}_*|\mathbf{y},\mathbf{X},\mathbf{X}_*,\mathbf{Z},\boldsymbol{\theta}) = \int p(\mathbf{y}_*|\mathbf{u},\mathbf{Z},\mathbf{X}_*,\boldsymbol{\theta})p(\mathbf{u}|\mathbf{y},\mathbf{X},\mathbf{Z},\boldsymbol{\theta})\mathrm{d}\mathbf{u} = \mathcal{N}\left(\mathbf{y}_*|\boldsymbol{\mu}_{\mathbf{y}_*},\mathbf{K}_{\mathbf{y}_*,\mathbf{y}_*}\right),$$

where

$$\begin{aligned}\boldsymbol{\mu}_{\mathbf{y}_*} &= \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^\top(\mathbf{D}+\boldsymbol{\Sigma})^{-1}\mathbf{y},\\ \mathbf{K}_{\mathbf{y}_*,\mathbf{y}_*} &= \mathbf{K}_{\mathbf{f}_*,\mathbf{f}_*} - \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top + \mathbf{K}_{\mathbf{f}_*,\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{f}_*,\mathbf{u}}^\top + \boldsymbol{\Sigma}_*.\end{aligned}$$

For the single output case, the assumption of the deterministic likelihood is equivalent to the deterministic training conditional (DTC) approximation, the full independence approximation leads to the fully independent training conditional (FITC) approximation (Quiñonero-Candela and Rasmussen, 2005) and the partial independence leads to the partially independent training conditional (PITC) approximation (Quiñonero-Candela and Rasmussen, 2005). The similarities of our approximations for multioutput GPs with respect to approximations presented in Quiñonero-Candela and Rasmussen (2005) for single output GPs are such, that we find it convenient to follow the same terminology and also refer to our approximations as DTC, FITC and PITC approximations for multioutput Gaussian processes.

5.3 Discussion: Model Selection in Approximated Models

The marginal likelihood approximation for the PITC, FITC and DTC variants is a function of both the hyperparameters of the covariance function and the location of the inducing variables. For estimation purposes, there seems to be a consensus in the GP community that hyperparameters for the covariance function can be obtained by maximization of the marginal likelihood. For selecting the inducing variables, though, there are different alternatives that can in principle be used. Simpler methods include fixing the inducing variables to be the same set of input data points or grouping the input data using a clustering method like K-means and then use the K resulting vectors as inducing variables. More sophisticated alternatives consider that the set of inducing variables must be restricted to be a subset of the input data (Csató and Opper, 2001; Williams and Seeger, 2001). This set of methods require a criteria for choosing the optimal subset of the training points (Smola and Bartlett, 2001; Seeger et al., 2003). Such approximations are truly sparse in the sense that only few data points are needed at the end for making predictions. Recently, Snelson and Ghahramani (2006) suggested using the marginal likelihood not only for the optimization of the hyperparameters in the covariance function, but also for the optimization of the location of these inducing variables. Although, using such procedure to find the optimal location of the inducing inputs might look in principle like an overwhelming optimization problem (inducing points usually appear non-linearly in the covariance function), in practice it has been shown that performances close to the full GP model can be obtained in a fraction of the time that it takes to train the full model. In that respect, the inducing points that are finally found are optimal in the same optimality sense that the hyperparameters of the covariance function.

Essentially, it would be possible to use any of the methods just mentioned above together with the multiple-output GP regression models presented in Sections 2.1, 2.1.2 and 3. In this paper, though, we follow Snelson and Ghahramani (2006) and optimize the locations of the inducing variables using the approximated marginal likelihoods and leave the comparison between the different model selection methods for inducing variables for future work.

In appendix A we include the derivatives of the marginal likelihood wrt the matrices $\mathbf{K}_{\mathbf{f},\mathbf{f}}, \mathbf{K}_{\mathbf{u},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$.

6. Experimental Evaluation

In this section we present results of applying the approximations in exam score prediction, pollutant metal prediction and the prediction of gene expression behavior in a gene-network. When possible, we first compare the convolved multiple output GP method against the intrinsic model of coregionalization and the semiparametric latent factor model. Then, we compare the different approximations in terms of accuracy and training times. First, though, we illustrate the performance of the approximation methods in a toy example.[10]

6.1 A Toy Example

For the toy experiment, we employ the kernel constructed as an example in Section 3. The toy problem consists of $D = 4$ outputs, one latent function, $Q = 1$ and $R_q = 1$ and one input dimension. The training data was sampled from the full GP with the following parameters, $S_{1,1} = S_{2,1} = 1$, $S_{3,1} = S_{4,1} = 5$, $P_{1,1} = P_{2,1} = 50$, $P_{3,1} = 300, P_{4,1} = 200$ for the outputs and $\Lambda_1 = 100$ for the latent function. For the independent processes, $w_d(\mathbf{x})$, we simply added white noise separately to each output so we have variances $\sigma_1^2 = \sigma_2^2 = 0.0125$, $\sigma_3^2 = 1.2$ and $\sigma_4^2 = 1$. We generate $N = 500$ observation points for each output and use 200 observation points (per output) for training the full and the approximated multiple output GP and the remaining 300 observation points for testing. We repeated the same experiment setup ten times and compute the standardized mean square error and the mean standardized log loss. For the approximations we use $K = 30$ inducing inputs. We sought the kernel parameters and the positions of the inducing inputs through maximizing the marginal likelihood using a scaled conjugate gradient algorithm. Initially the inducing inputs are equally spaced between the interval $[-1, 1]$.

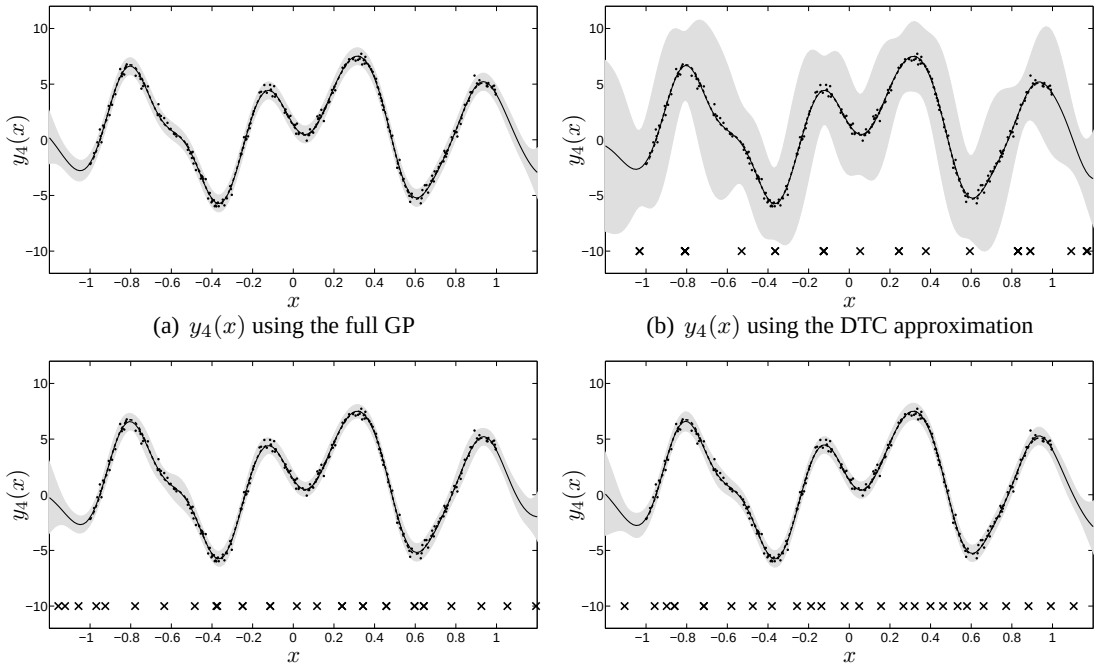
Figure 4 shows the training result of one of the ten repetitions. The predictions shown correspond to the full GP in Figure 4(a), the DTC approximation in Figure 4(b), the FITC approximation in Figure 4(c) and the PITC approximation in Figure 4(d).

Tables 3 and 4 show the average prediction results over the test set. Table 3 shows that the SMSE of the approximations is similar to the one obtained with the full GP. However, there are important differences in the values of the MSLL shown in Table 4. DTC offers the worst performance. It gets better for FITC and PITC since they offer a more precise approximation to the full covariance.

The training times for iteration of each model are 1.97 secs for the full GP, 0.20 secs for DTC, 0.41 for FITC and 0.59 for the PITC, on average.

As we have mentioned before, one important feature of multiple output prediction is that we can exploit correlations between outputs to predict missing observations. We used a simple example to illustrate this point. We removed a portion of one output between $[-0.8, 0]$ from the training data in the experiment before (as shown in Figure 5) and train the different models to predict the behavior of $y_4(x)$ for the missing information. The predictions shown correspond to the full GP in Figure 5(a), an independent GP in Figure 5(b), the DTC approximation in Figure 5(c), the FITC approximation in

10. Code to run all simulations in this section is available at `http://staffwww.dcs.shef.ac.uk/people/N.Lawrence/multigp/`.



(a) $y_4(x)$ using the full GP

(b) $y_4(x)$ using the DTC approximation

(c) $y_4(x)$ using the FITC approximation

(d) $y_4(x)$ using the PITC approximation

Figure 4: Predictive mean and variance using the full multi-output GP and the approximations for output 4. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. The dashed line corresponds to the ground truth signal, that is, the sample from the full GP model without noise. In these plots the predictive mean overlaps almost exactly with the ground truth. The dots are the noisy training points. The crosses in Figures 4(b), 4(c) and 4(d) correspond to the locations of the inducing inputs after convergence. Notice that the DTC approximation in Figure 4(b) captures the predictive mean correctly, but fails in reproducing the correct predictive variance.

Method	SMSE $y_1(x)$	SMSE $y_2(x)$	SMSE $y_3(x)$	SMSE $y_4(x)$
Full GP	1.06 ± 0.08	0.99 ± 0.06	1.10 ± 0.09	1.05 ± 0.09
DTC	1.06 ± 0.08	0.99 ± 0.06	1.12 ± 0.09	1.05 ± 0.09
FITC	1.06 ± 0.08	0.99 ± 0.06	1.10 ± 0.08	1.05 ± 0.08
PITC	1.06 ± 0.08	0.99 ± 0.06	1.10 ± 0.09	1.05 ± 0.09

Table 3: Standardized mean square error (SMSE) for the toy problem over the test set. All numbers are to be multiplied by 10^{-2}. The experiment was repeated ten times. Table includes the value of one standard deviation over the ten repetitions.

Figure 5(d) and the PITC approximation in Figure 5(e). The training of the approximation methods is done in the same way than in the experiment before.

Method	MSLL $y_1(x)$	MSLL $y_2(x)$	MSLL $y_3(x)$	MSLL $y_4(x)$
Full GP	-2.27 ± 0.04	-2.30 ± 0.03	-2.25 ± 0.04	-2.27 ± 0.05
DTC	-0.98 ± 0.18	-0.98 ± 0.18	-1.25 ± 0.16	-1.25 ± 0.16
FITC	-2.26 ± 0.04	-2.29 ± 0.03	-2.16 ± 0.04	-2.23 ± 0.05
PITC	-2.27 ± 0.04	-2.30 ± 0.03	-2.23 ± 0.04	-2.26 ± 0.05

Table 4: Mean standardized log loss (MSLL) for the toy problem over the test set. More negative values of MSLL indicate better models. The experiment was repeated ten times. Table includes the value of one standard deviation over the ten repetitions.

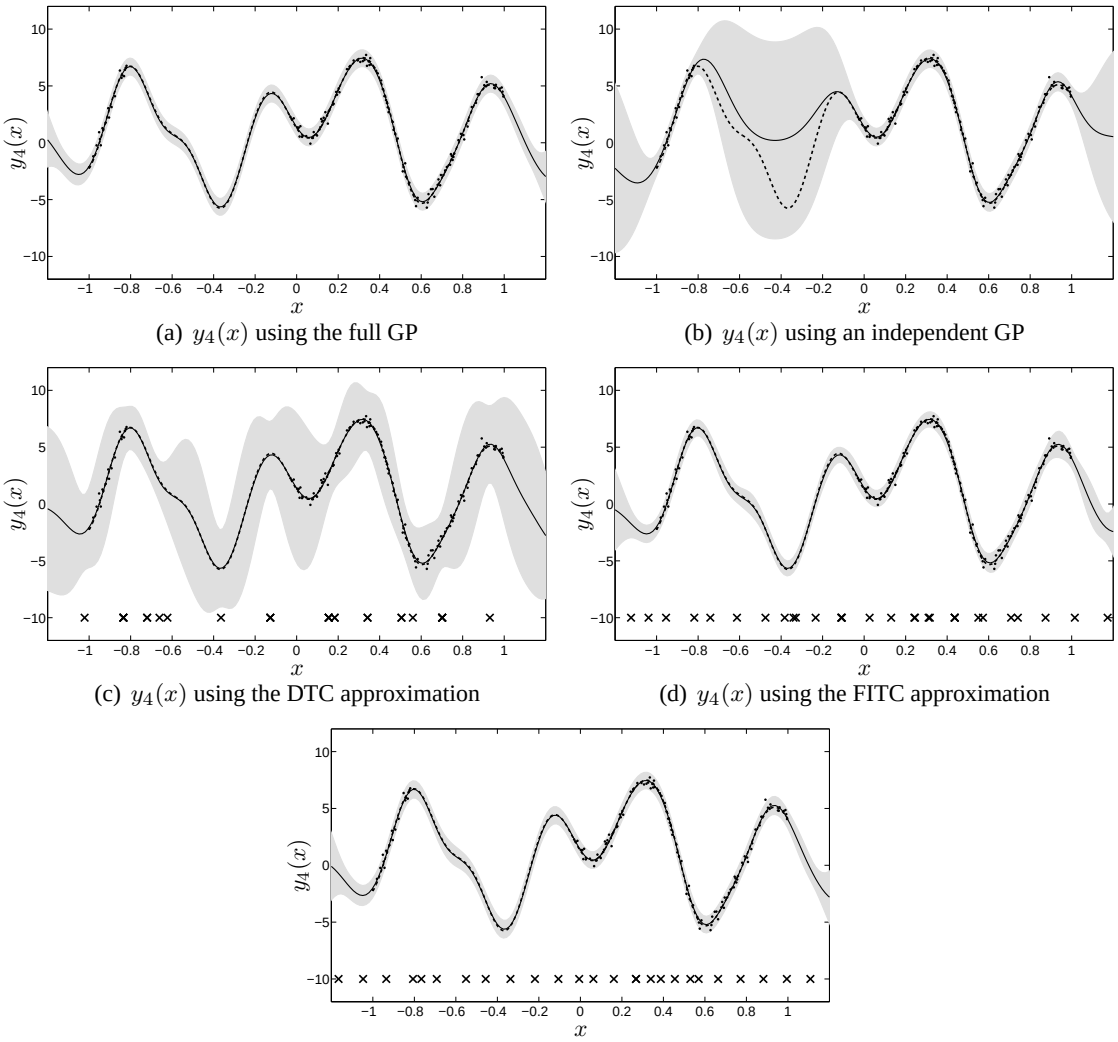
Due to the strong dependencies between the signals, our model is able to capture the correlations and predicts accurately the missing information.

6.2 Exam Score Prediction

In the first experiment with real data that we consider, the goal is to predict the exam score obtained by a particular student belonging to a particular school. The data comes from the Inner London Education Authority (ILEA).[11] It consists of examination records from 139 secondary schools in years 1985, 1986 and 1987. It is a random 50% sample with 15362 students. The input space consists of four features related to each student (year in which each student took the exam, gender, performance in a verbal reasoning (VR) test[12] and ethnic group) and four features related to each school (percentage of students eligible for free school meals, percentage of students in VR band one, school gender and school denomination). From the multiple output point of view, each school represents one output and the exam score of each student a particular instantiation of that output or $D = 139$.

We follow the same preprocessing steps employed in Bonilla et al. (2008). The only features used are the student-dependent ones, which are categorial variables. Each of them is transformed to a binary representation. For example, the possible values that the variable year of the exam can take are 1985, 1986 or 1987 and are represented as 100, 010 or 001. The transformation is also applied to the variables gender (two binary variables), VR band (four binary variables) and ethnic group (eleven binary variables), ending up with an input space with 20 dimensions. The categorial nature of the data restricts the input space to $N = 202$ unique input feature vectors. However, two students represented by the same input vector $\mathbf{x}$, and belonging both to the same school, d, can obtain different exam scores. To reduce this noise in the data, we take the mean of the observations that, within a school, share the same input vector and use a simple heteroskedastic noise model in which the variance for each of these means is divided by the number of observations used to compute it.[13] The performance measure employed is the percentage of explained variance defined as the total variance of the data minus the sum-squared error on the test set as a percentage of the total data variance. It can be seen as the percentage version of the coefficient of determination between the

11. This data is available at `http://www.cmm.bristol.ac.uk/learning-training/multilevel-m-support/datasets.shtml`.
12. Performance in the verbal reasoning test was divided in three bands. Band 1 corresponds to the highest 25%, band 2 corresponds to the next 50% and band 3 the bottom 25% (Nuttall et al., 1989; Goldstein, 1991).
13. Different noise models can be used. However, we employed this one so that we can compare directly to the results presented in Bonilla et al. (2008).



(a) $y_4(x)$ using the full GP

(b) $y_4(x)$ using an independent GP

(c) $y_4(x)$ using the DTC approximation

(d) $y_4(x)$ using the FITC approximation

(e) $y_4(x)$ using the PITC approximation

Figure 5: Predictive mean and variance using the full multi-output GP, the approximations and an independent GP for output 4 with a range of missing observations in the interval $[-0.8, 0.0]$. The solid line corresponds to the mean predictive, the shaded region corresponds to 2 standard deviations away from the mean and the dash line is the actual value of the signal without noise. The dots are the noisy training points. The crosses in Figures 5(c), 5(d) and 5(e) correspond to the locations of the inducing inputs after convergence.

test targets and the predictions. The performance measure is computed for ten repetitions with 75% of the data in the training set and 25% of the data in the testing set.

We first compare different methods without including the efficient approximations. These methods are independent GPs, multi-task GPs (Bonilla et al., 2008), the intrinsic coregionalization model, the semiparametric latent factor model and convolved multiple output GPs. Results are

Method	Explained variance (%)
Independent GPs (Bonilla et al., 2008)	31.12 ± 1.33
Multi-task GP (Nyström, $R_1 = 2$) (Bonilla et al., 2008)	36.16 ± 0.99
Intrinsic coregionalization model ($R_1 = 1$)	52.54 ± 2.46
Intrinsic coregionalization model ($R_1 = 2$)	51.94 ± 1.84
Intrinsic coregionalization model ($R_1 = 5$)	45.31 ± 1.63
Semiparametric latent factor model ($Q = 2$)	51.82 ± 1.93
Semiparametric latent factor model ($Q = 5$)	44.87 ± 1.15
Convolved Multiple Outputs GPs ($Q = 1$, $R_q = 1$)	$\mathbf{53.84 \pm 2.01}$

Table 5: Average percentage of explained variance and standard deviation for the exam score prediction on the ILEA data set computed over 10 repetitions. The independent GP result and the multi-task GP result were taken from Bonilla et al. (2008). The value of R_1 in the multi-task GP and in the intrinsic coregionalization model indicates the rank of the matrix $\mathbf{B}_1$ in Equation (6). The value of Q in the semiparametric latent factor model indicates the number of latent functions. The value of R_q in the convolved multiple output GP refers to the number of latent functions that share the same number of parameters (see Equation 8). Refer to the text for more details.

presented in Table 5. The results for the independent GPs and the multi-task GPs were taken from Bonilla et al. (2008). The multi-task GP result uses a matrix $\mathbf{B}_1$ with rank $R_1 = 2$. For the intrinsic model of coregionalization, we use an incomplete Cholesky decomposition $\mathbf{B}_1 = \widetilde{L}\widetilde{L}^\top$, and include results for different values of the rank R_1. The basic covariance $k_q(\mathbf{x}, \mathbf{x}')$ in the ICM is assumed to follow a Gaussian form. For the semiparametric latent factor model, all the latent functions use covariance functions with Gaussian forms. For SLFM, we include results for different values of the number of latent functions ($Q = 2$ and $Q = 5$). Note that SLFM with $Q = 1$ is equivalent to ICM with $R_1 = 1$. For the convolved multiple output covariance result, the kernel employed was introduced in Section 3. For all the models we estimate the parameters maximizing the likelihood through scaled conjugate gradient and run the optimization algorithm for a maximum of 1000 iterations. Table 5 shows that all methods outperform the independent GPs. Even though multi-task GPs with $R_1 = 2$ and ICM with $R_1 = 2$ are equivalent methods, the difference of results might be explained because the multi-task GP method uses a Nyström approximation for the matrix $\mathbf{K}_1$ in Equation (6). Results for ICM with $R_1 = 1$, SLFM with $Q = 2$ and the convolved covariance are similar within the standard deviations. The convolved GP was able to recover the best performance using only one latent function ($Q = 1$). This data set was also employed to evaluate the performance of the multitask kernels in Evgeniou and Pontil (2004). The best result presented in this work was 34.37 ± 0.3. However, due to the averaging of the observations that we employed here, it is not fair to compare directly against those results.

We present next the results of using the efficient approximations for the exam school prediction example. In Figure 6, we have included the results of Table 5 alongside the results of using DTC, FITC and PITC for 5, 20 and 50 inducing points. The initial positions of the inducing points are selected using the *k-means* algorithm with the training data points as inputs to the algorithm. The positions of these points are optimized in a scaled conjugate gradient procedure together with the parameters of the model. We notice that using the approximations we obtain similar performances

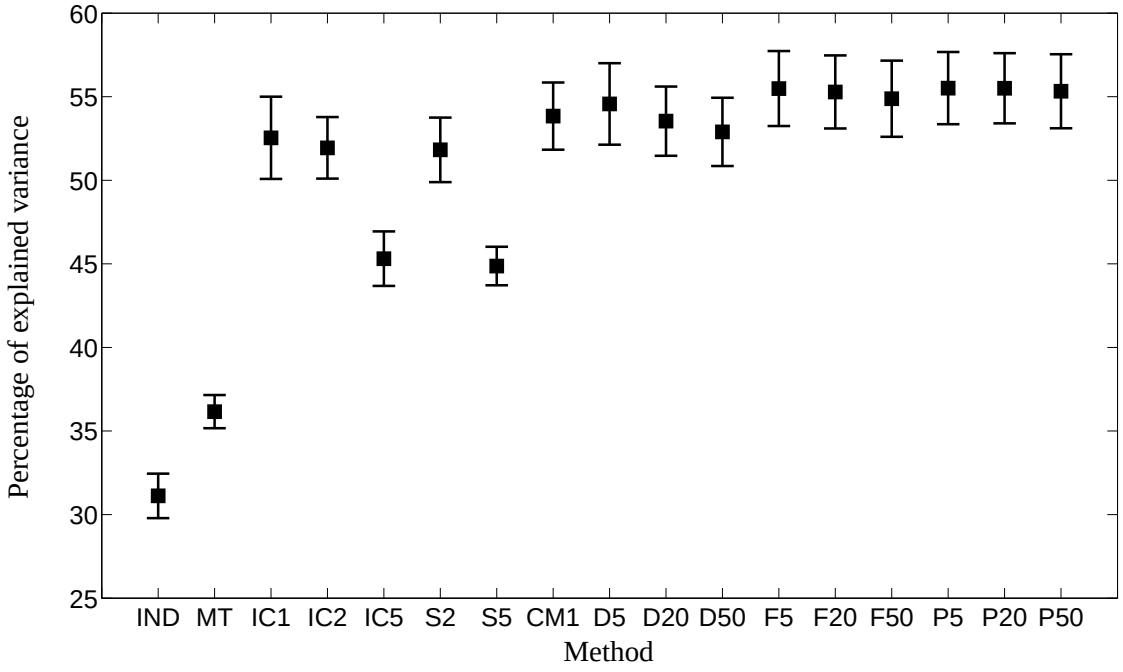


Figure 6: Mean and standard deviation of the percentage of explained variance for exam score prediction results on the ILEA data set. The experiment was repeated ten times. In the bottom of the figure, IND stands for independent GPs, MT stands for multi-task GPs, ICR_1 stands for intrinsic coregionalization model with rank R_1, SQ stands for semiparametric latent factor model with Q latent functions, CM1 stands for convolved multiple output covariance with $Q = 1$ and $R_q = 1$ and DK, FK, PK stands for DTC, FITC and PITC with K inducing points, respectively. The independent GPs and multi-task GPs results were obtained from Bonilla et al. (2008).

to the full models with as few as 5 inducing points. FITC and PITC slightly outperform the DTC method, although results are within the standard deviation.

Table 6 shows the training times for the different methods.[14] Clearly, the efficient approximations are faster than the full methods. This is particularly true when comparing the training times per iteration (second column). The approximations were run over 1000 iterations, but the results for 100 iterations were pretty much the same. For the ICM and SLFM results, definitely more than 100 iterations were needed. With 1000 iterations DTC with 5 inducing points offers a speed up factor of 24 times over the ICM with $R_1 = 1$ and a speed up factor of 137 over the full convolved multiple output method.[15] On the other hand, with 1000 iterations, PITC with 50 inducing points offers a speed up of 9.8 over ICM with $R_1 = 1$ and a speed up of 55 over the full convolved GP method.

14. All experiments with real data were run in workstations with 2.59 GHz, AMD Opteron's and up to 16 GHz of RAM. Only one processor was used on each run.
15. The speed up factor is computed as the relation between the slower method and the faster method, using the training times of the third column in Table 6.

Method	Time per iter. (secs)	Training time (secs)
ICM ($R_1 = 1$)	83.60	16889
ICM ($R_1 = 2$)	85.61	47650
ICM ($R_1 = 5$)	88.02	64535
SLFM ($Q = 2$)	97.00	58564
SLFM ($Q = 5$)	130.23	130234
CMOGP ($Q = 1$, $R_q = 1$)	95.55	95510
DTC 5 ($Q = 1$, $R_q = 1$)	0.69	694
DTC 20 ($Q = 1$, $R_q = 1$)	0.80	804
DTC 50 ($Q = 1$, $R_q = 1$)	1.04	1046
FITC 5 ($Q = 1$, $R_q = 1$)	0.94	947
FITC 20 ($Q = 1$, $R_q = 1$)	1.02	1026
FITC 50 ($Q = 1$, $R_q = 1$)	1.27	1270
PITC 5 ($Q = 1$, $R_q = 1$)	1.13	1132
PITC 20 ($Q = 1$, $R_q = 1$)	1.24	1248
PITC 50 ($Q = 1$, $R_q = 1$)	1.71	1718

Table 6: Training times for the exam score prediction example. In the table, CMOGP stands for convolved multiple outputs GP. The first column indicates the training time per iteration of each method while the second column indicates the total training time. All the numbers presented are average results over the ten repetitions.

As mentioned before, the approximations reach similar performances using 100 iterations, increasing the speed up factors by ten.

To summarize this example, we have shown that the convolved multiple output GP offers a similar performance to the ICM and SLFM methods. We also showed that the efficient approximations can offer similar performances to the full methods and by a fraction of their training times. Moreover, this example involved a relatively high-input high-output dimensional data set, for which the convolved covariance has not been used before in the literature.

6.3 Heavy Metals in the Swiss Jura

The second example with real data that we consider is the prediction of the concentration of several metal pollutants in a region of the Swiss Jura. This is a relatively low-input low-output dimensional data set that we use to illustrate the ability of the PITC approximation to reach the performance of the full GP if the enough amount of inducing points is used. The data consist of measurements of concentrations of several heavy metals collected in the topsoil of a 14.5 km^2 region of the Swiss Jura. The data is divided into a prediction set (259 locations) and a validation set (100 locations).[16] In a typical situation, referred to as undersampled or heterotopic case, a few expensive measurements of the attribute of interest are supplemented by more abundant data on correlated attributes that are cheaper to sample. We follow the experiment described in Goovaerts (1997, p. 248, 249) in which a *primary variable* (cadmium) at prediction locations in conjunction with some *secondary variables* (nickel and zinc) at prediction and validation locations, are employed to predict the con-

16. This data is available at `http://www.ai-geostats.org/`.

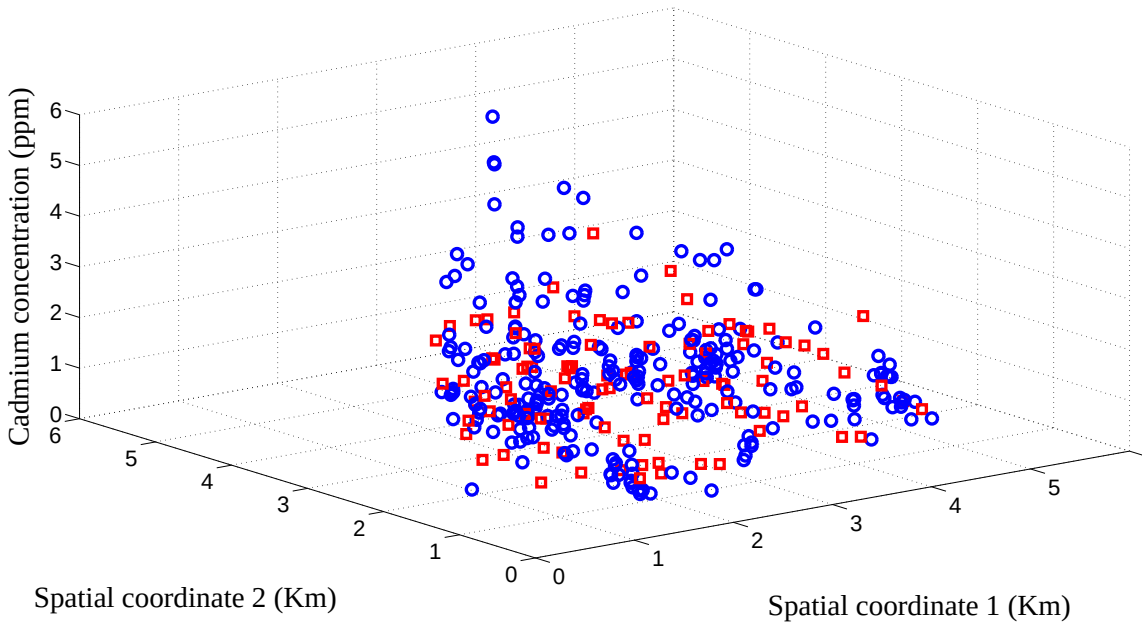


Figure 7: Cadmium concentration for the Swiss Jura example. The blue circles refer to the prediction set (training data for cadmium) and the red squares are the concentrations for the validation set (testing data for cadmium).

centration of the primary variable at validation locations. Figure 7 shows the cadmium concentration for the particular set of input locations of the prediction set (blue circles) and the particular set of input locations of the validation set (red squares). As in the exam score prediction example, we first compare the performances of the full GP methods and later we introduce the performances of the approximations. We compare results of independent GPs, ordinary cokriging, the intrinsic coregionalization model, the semiparametric latent factor model and the convolved multiple output covariance. For independent GPs we use Gaussian covariances with different length-scales for each input dimension. Before describing the particular setup for the other methods appearing in Table 7, we first say a few lines about the cokriging method. The interested reader can find details in several geostatistics books (see Cressie, 1993; Goovaerts, 1997; Wackernagel, 2003).

Cokriging is the generalization of kriging to multiple outputs. It is an unbiased linear predictor that minimizes the error variance between the data and the predicted values. Different cokriging methods assume that each output can be decomposed as a sum of a residual component with zero mean and non-zero covariance function and a trend component. The difference between the cokriging estimators is based on the assumed model for the trend component. While in simple cokriging the mean is assumed to be constant and known, in ordinary cokriging it is assumed to be constant, but unknown, leading to a different set of equations for the predictor. Whichever cokriging method is used implies using the values of the covariance for the residual component in the equations for the prediction, making explicit the need for a positive semidefinite covariance function. In the geostatistics literature, the usual practice is to use the linear model of coregionalization to construct a valid covariance function for the residual component and then use any of the cokriging estimators

Method	Average Mean absolute error
Independent GPs	0.5739 ± 0.0003
Ordinary cokriging (p. 248, 249 Goovaerts, 1997)	0.51
Intrinsic coregionalization model ($R_1 = 2$)	0.4608 ± 0.0025
Semiparametric latent factor model ($Q = 2$)	0.4578 ± 0.0025
Convolved Multiple Outputs GPs ($Q = 2$, $R_q = 1$)	$\mathbf{0.4552 \pm 0.0013}$

Table 7: Average mean absolute error and standard deviation for predicting the concentration of metal cadmium with the full dependent GP model and different forms for the covariance function. The result for ordinary cokriging was obtained from Goovaerts (p. 248, 249 1997) and it is explained in the text. For the intrinsic coregionalization model and the semiparametric latent factor model we use a Gaussian covariance with different length-scales along each input dimension. For the convolved multiple output covariance, we use the covariance described in Section 3. See the text for more details.

for making predictions. A common algorithm to fit the linear model of coregionalization minimizes some error measure between a sample or experimental covariance matrix obtained from the data and the particular matrix obtained from the form chosen for the linear model of coregionalization (Goulard and Voltz, 1992).

Let us go back to the results shown in Table 7. The result that appears as ordinary cokriging was obtained with the ordinary cokriging predictor and a LMC with $Q = 3$ and $R_q = 3$ (p. 119 Goovaerts, 1997). Two of the basic covariances $k_q(\mathbf{x}, \mathbf{x}')$ have a particular polynomial form, while the other corresponds to a bias term.[17] For the prediction stage, only the closest 16 data locations in the primary and secondary variables are employed. Also in Table 7, we present results using the intrinsic coregionalization with a rank two ($R_1 = 2$) for $\mathbf{B}_1$, the semiparametric latent factor model with two latent functions ($Q = 2$) and the convolved multiple output covariance with two latent functions ($Q = 2$ and $R_q = 1$). The choice of either $R_1 = 2$ or $Q = 2$ for the methods was due to the cokriging setup for which two polynomial-type covariances were used. The basic covariances for ICM and SLFM have a Gaussian form with different length scales in each input dimension. For the CMOC, we employ the covariance from Section 3. Parameters for independent GPs, ICM, SLFM and CMOC are learned maximizing the marginal likelihood in Equation (13), using a scaled conjugate gradient procedure. We run the optimization algorithm for up to 200 iterations. Since the prediction and location sets are fixed, we repeat the experiment ten times changing the initial values of the parameters.

Table 7 shows that all methods, including ordinary cokriging, outperform independent GPs. ICM, SLFM and CMOC outperform cokriging. Results for SLFM and CMOC are similar, although CMOC outperformed ICM in every trial of the ten repetitions. The better performance for the SLFM and the CMOC over the ICM would indicate the need for a second latent function with different parameters to the first one. Using a non-instantaneous approach may slightly increase the performance. However, results overlap within one standard deviation.

17. In fact, the linear model of coregionalization employed is constructed using variograms as basic tools that account for the dependencies in the input space. Variograms and covariance functions are related tools used in the geostatistics literature to describe dependencies between variables. A precise definition of the concept of variogram is out of the scope of this paper.

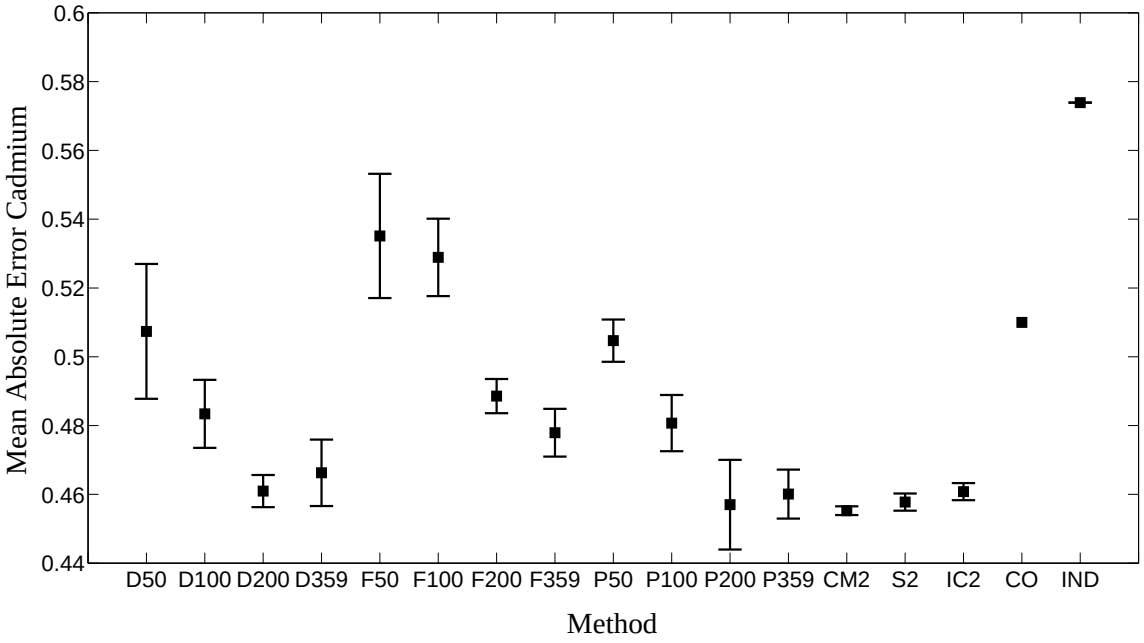


Figure 8: Average mean absolute error and standard deviation for prediction of the pollutant metal cadmium. The experiment was repeated ten times. In the bottom of the figure DK, FK, PK stands for DTC, FITC and PITC with K inducing values, CM2 stands for convolved multiple output covariance with $Q = 2$ and $R_q = 1$, S2 stands for semiparametric latent factor model with $Q = 2$ latent functions, IC2 stands for intrinsic coregionalization model with rank $R_1 = 2$, CO stands for the cokriging method explained in the text and IND stands for independent GPs.

We next include the performances for the efficient approximations. For the results of the approximations, a *k-means* procedure is employed first to find the initial locations of the inducing values and then these locations are optimized in the same optimization procedure used for the parameters. Each experiment is repeated ten times changing the initial value of the parameters. Figure 8 shows the results of prediction for cadmium for the different approximations with varying number of inducing points (this is, different values of K). We also include in the figure the results for the convolved multiple output GP (CM2), semiparametric latent factor model (S2), intrinsic coregionalization model (IC2), ordinary cokriging (CO) and independent GPs (IND).

Notice that DTC and PITC outperform cokriging and independent GPs for any value of K. Also for $K = 200$ and $K = 359$, DTC and PITC reach the performance of the full GP methods, either in average (for $K = 200$) or within one standard deviation (for $K = 359$). $K = 200$ might be a considerable amount of inducing points when compared to the total amount of input training data (359 for nickel and zinc and 259 for cadmium). The need of that amount of inducing points could be explained due to the high variability of the data: mean values for the concentration of pollutant metals are 1.30, 20.01 and 75.88 for cadmium, nickel and zinc, while standard deviations are 0.91,

Method	Time per iter. (secs)	Training time (secs)
ICM	3.84	507
SLFM	4.14	792
CMOGP	4.47	784
DTC 50	0.28	20
DTC 100	0.80	64
DTC 200	1.95	185
DTC 359	4.24	551
FITC 50	0.81	69
FITC 100	1.14	159
FITC 200	2.12	244
FITC 359	5.76	691
PITC 50	1.78	268
PITC 100	2.46	320
PITC 200	4.06	385
PITC 359	7.94	1191

Table 8: Training times for the prediction of the cadmium pollutant metal. In the table, CMOGP stands for convolved multiple outputs GP. The first column indicates the training time per iteration of each method and the second column indicates the total training time. All the numbers presented are average results over the ten repetitions.

8.09 and 30.81 giving coefficients of variation of 70.00%, 40.42% and 40.60%.[18] Variability in cadmium can be observed intuitively from Figure 7. Notice also that FITC outperforms cokriging and independent GPs for $K = 200$ and $K = 359$. The figure also shows that DTC outperforms FITC for all values of K. However, the measure of performance employed, the mean absolute error, does not take into account the predictive variance of the approximated GPs. Using as measures the standardized mean absolute error and the mean standardized log-likelihood, that take into account the predictive variance, FITC outperforms DTC: DTC in average has a MSLL of 0.4544 and a SMSE of 0.9594 while FITC in average has a MSLL of -0.0637 with a SMSE of 0.9102. PITC in average has a MSLL of -0.1226 and SMSE 0.7740. Averages were taken over the different values of K.

Finally, Table 8 shows the timing comparisons for the pollutant example. The training times for DTC with 200 inducing points and PITC with 200 inducing points, which are the first methods that reach the performance of the full GP, are less than any of the times of the full GP methods. For DTC with 200 inducing points, the speed up factor is about 2.74 when compared to ICM and 4.23 when compared to CMOGP. For PITC with 200 inducing points, the speed up factor is 1.31 when compared to ICM and 2.03 when compared to CMOGP. Notice also that all methods are less or equally expensive than the different full GP variants, except for PITC with 359 inducing variables. For this case, however, 4 out of the 10 repetitions reached the average performance in 100 iterations, given a total training time of approximately 794.12 secs., a time much closer to CMOGP and SLFM.

18. The coefficient of variation is defined as the standard deviation over the mean. It could be interpreted also as the inverse of the signal-to-noise ratio.

6.4 Regression Over Gene Expression Data

We now present a third example with real data. This time we only include the performances for the approximations. The goal is to do multiple output regression over gene expression data. The setup was described in Section 4. The difference with that example, is that instead of using $D = 50$ outputs, here we use $D = 1000$ outputs. We do multiple output regression using DTC, FITC and PITC fixing the number of inducing points to $K = 8$ equally spaced in the interval $[-0.5, 11.5]$. Since it is a 1-dimensional input data set, we do not optimize the location of the inducing points, but fix them to the equally spaced initial positions. As for the full GP model in example of Section 4, we make $Q = 1$ and $R_q = 1$. Again we use scaled conjugate gradient to find the parameters that maximize the marginal likelihood in each approximation. The optimization procedure runs for 100 iterations.

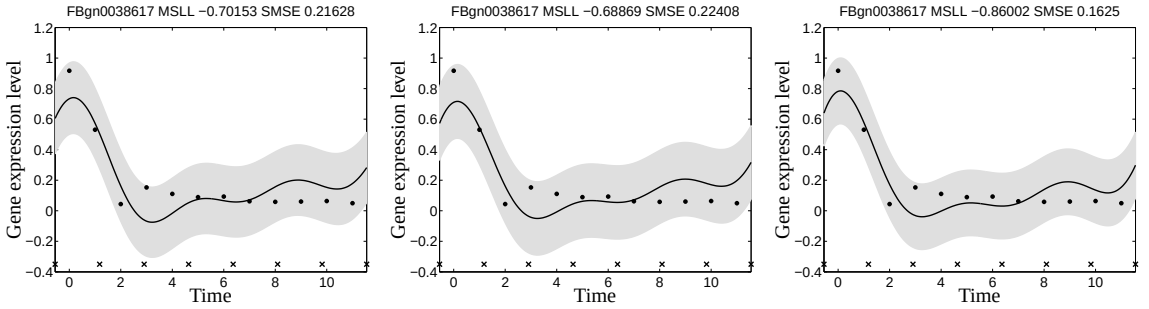
Train set	Test set	Method	Average SMSE	Average MSLL	Average TTPI
Replica 1	Replica 2	DTC	0.5421 ± 0.0085	-0.2493 ± 0.0183	2.04
		FITC	0.5469 ± 0.0125	-0.3124 ± 0.0200	2.31
		PITC	0.5537 ± 0.0136	-0.3162 ± 0.0206	2.59
Replica 2	Replica 1	DTC	0.5454 ± 0.0173	0.6499 ± 0.7961	2.10
		FITC	0.5565 ± 0.0425	-0.3024 ± 0.0294	2.32
		PITC	0.5713 ± 0.0794	-0.3128 ± 0.0138	2.58

Table 9: Standardized mean square error (SMSE), mean standardized log loss (MSLL) and training time per iteration (TTPI) for the gene expression data for 1000 outputs using the efficient approximations for the convolved multiple output GP. The experiment was repeated ten times with a different set of 1000 genes each time. Table includes the value of one standard deviation over the ten repetitions.

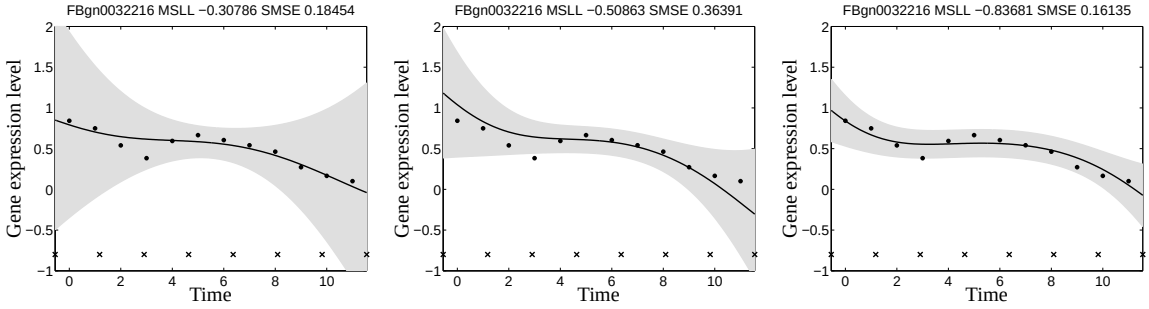
Table 9 shows the results of applying the approximations in terms of SMSE and MSLL (columns 4 and 5). DTC and FITC slightly outperforms PITC in terms of SMSE, but PITC outperforms both DTC and FITC in terms of MSLL. This pattern repeats itself when the training data comes from replica 1 or from replica 2.

In Figure 9 we show the performance of the approximations over the same two genes of Figure 1, these are FBgn0038617 and FBgn0032216. The non-instantaneous mixing effect of the model can still be observed. Performances for these particular genes are highlighted in Table 10. Notice that the performances are between the actual performances for the LMC and the CMOC appearing in Table 2. We include these figures only for illustrative purposes, since both experiments use a different number of outputs. Figures 1 and 2 were obtained as part of multiple output regression problem of $D = 50$ outputs, while Figures 9 and 10 were obtained in a multiple output regression problem with $D = 1000$ outputs.

In Figure 10, we replicate the same exercise for the genes FBgn0010531 and FBgn0004907, that also appeared in Figure 2. Performances for DTC, FITC and PITC are shown in Table 10 (last six rows), which compare favourably with the performances for the linear model of coregionalization in Table 2 and close to the performances for the CMOC. In average, PITC outperforms the other methods for the specific set of genes in both figures above.



(a) DTC, short length scale (b) FITC, short length scale (c) PITC, short length scale



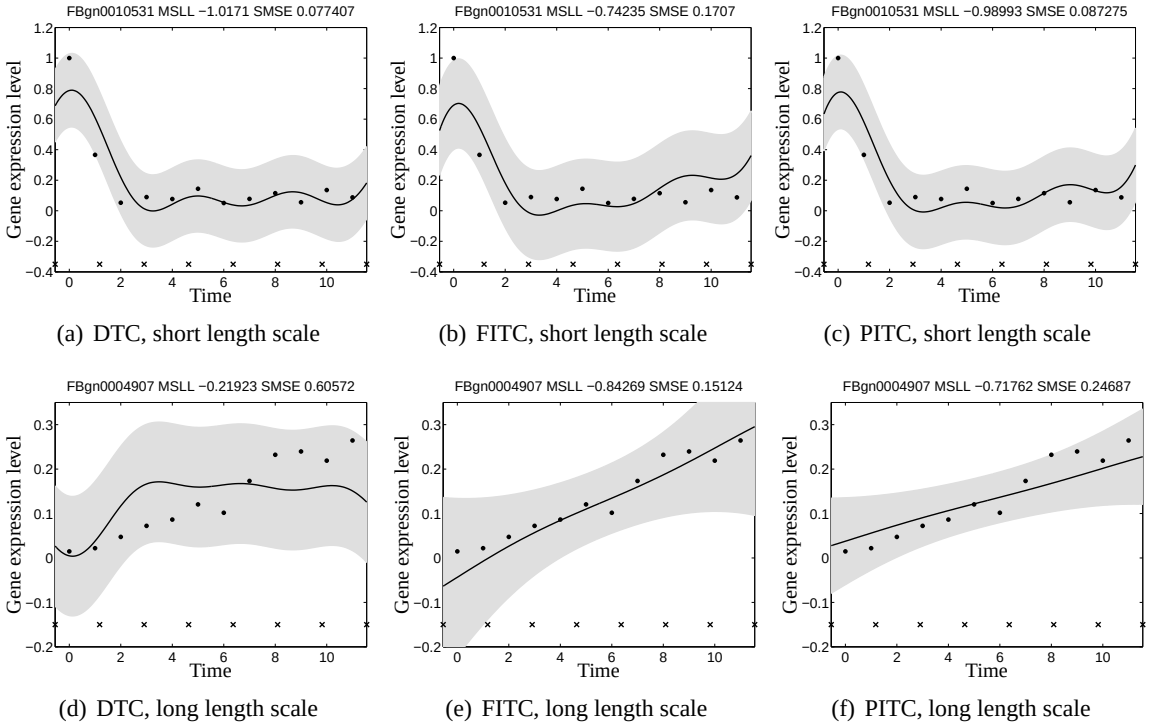
(d) DTC, long length scale (e) FITC, long length scale (f) PITC, long length scale

Figure 9: Predictive mean and variance for genes FBgn0038617 (first row) and FBgn0032216 (second row) using the different approximations. In the first column DTC in Figures 9(a) and 9(d), second column FITC in Figures 9(b) and 9(e), and in the third column PITC in Figures 9(c) and 9(f). The training data comes from replica 1 and the testing data from replica 2. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure. The adjectives "short" and "long" given to the length-scales in the captions of each figure, must be understood like relative to each other. The crosses in the bottom of each figure indicate the positions of the inducing points, which remain fixed during the training procedure.

With respect to the training times, the Table 9 in the column 6 shows the average training time per iteration (average TTPI) for each approximation. To have an idea of the saving times, one iteration of the full GP model for the same 1000 genes would take around 4595.3 seconds. This gives a speed up factor of 1780, approximately.

7. Conclusions

In this paper we first presented a review of different alternatives for multiple output regression grouped under a similar framework known as the linear model of coregionalization. Then we illustrated how the linear model of coregionalization can be interpreted as an instantaneous mixing of latent functions, in contrast to a convolved multiple output framework, where the mixing



(a) DTC, short length scale (b) FITC, short length scale (c) PITC, short length scale

(d) DTC, long length scale (e) FITC, long length scale (f) PITC, long length scale

Figure 10: Predictive mean and variance for genes FBgn0010531 (first row) and FBgn0004907 (second row) using the different approximations. In the first column DTC in Figures 10(a) and 10(d), second column FITC in Figures 10(b) and 10(e), and in the third column PITC in Figures 10(c) and 10(f). The training data comes now from replica 2 and the testing data from replica 1. The solid line corresponds to the predictive mean, the shaded region corresponds to 2 standard deviations of the prediction. Performances in terms of SMSE and MSLL are given in the title of each figure. The crosses in the bottom of each figure indicate the positions of the inducing points, which remain fixed during the training procedure.

is not necessarily instantaneous. Experimental results showed that in systems with a presence of some dynamics (for example, the gene expression data set), having this additional element of non-instantaneous mixing can lead to simpler explanations of the data. While, in systems for which the dynamics is not so obvious (for example, the exam score prediction data set), the benefit of using the non-instantaneous mixing was less noticeable.

We have also presented different efficient approximations for multiple output GPs, in the context of convolution processes. Using these approximations we can capture the correlated information among outputs while reducing the amount of computational load for prediction and optimization purposes. The computational complexity for the DTC and the FITC approximations is $O(NDK^2)$. The reduction in computational complexity for the PITC approximation is from $O(N^3D^3)$ to $O(N^3D)$. This matches the computational complexity for modeling with independent GPs. However, as we have seen, the predictive power of independent GPs is lower. Also, since

Test replica	Test genes	Method	SMSE	MSLL
Replica 2	FBgn0038617	DTC	0.2162	−0.7015
		FITC	0.2240	−0.6886
		PITC	0.1625	−0.8600
	FBgn0032216	DTC	0.1845	−0.3078
		FITC	0.3639	−0.5086
		PITC	0.1613	−0.8368
Replica 1	FBgn0010531	DTC	0.0774	−1.0171
		FITC	0.1707	−0.7423
		PITC	0.0872	−0.9899
	FBgn0004907	DTC	0.6057	−0.2192
		FITC	0.1512	−0.8426
		PITC	0.2468	−0.7176

Table 10: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the genes in Figures 9 and 10 for DTC, FITC and PITC with $K = 8$. Genes FBgn0038617 and FBgn0010531 have a shorter length-scale when compared to genes FBgn0032216 and FBgn0004907.

PITC makes a better approximation of the likelihood, the variance of the results is usually lower and approaches closely to the performance of the full GP, when compared to DTC and FITC. As a byproduct of seeing the linear model of coregionalization as a particular case of the convolved GPs, we can extend all the approximations to work under the linear model of coregionalization regime.

With an appropriate selection of the kernel smoothing function we have an indirect way to generate different forms for the covariance function in the multiple output setup. We showed an example with Gaussian kernels, for which a suitable standardization of the kernels can be made, leading to competitive results in high-dimensional input regression problems, as seen in the school exam score prediction problem. The authors are not aware of other work in which this convolution process framework has been applied in problems with high input dimensions.

As shown with the Swiss Jura experiment, we might need a considerable amount of inducing points compared to the amount of training data, when doing regression over very noisy outputs. This agrees to some extent with our intuition in Section 5, where we conditioned the validity of the approximations to the smoothness of the latent functions. However, even for this case, we can obtain the same performances in a fraction of the time that takes to train a full GP. Moreover, the approximations allow multiple output regression over a large amount of outputs, in scenarios where training a full GP become extremely expensive. We showed an example of this type with the multiple output regression over the gene expression data.

Linear dynamical systems responses can be expressed as a convolution between the impulse response of the system with some input function. This convolution approach is an equivalent way of representing the behavior of the system through a linear differential equation. For systems involving high amounts of coupled differential equations (Álvarez et al., 2009; Álvarez et al., 2011a; Honkela et al., 2010), the approach presented here is a reasonable way of obtaining approximate solutions and incorporating prior domain knowledge to the model.

Recently, Titsias (2009) highlighted how optimizing inducing variables can be problematic as they introduce many hyperparameters in the likelihood term. Titsias (2009) proposed a variational method with an associated lower bound where inducing variables are *variational parameters*. Following the ideas presented here, we can combine easily the method of Titsias (2009) and propose a lower bound for the multiple output case. We have followed a first attempt in that direction and some results have been presented in Álvarez et al. (2010).

Acknowledgments

The authors would like to thank Edwin Bonilla for his valuable feedback with respect to the exam score prediction example. The work has benefited greatly from discussions with David Luengo, Michalis Titsias, and Magnus Rattray. We also thank to three anonymous reviewers for their helpful comments. The authors are very grateful for support from a Google Research Award "Mechanistically Inspired Convolution Processes for Learning" and the EPSRC Grant No EP/F005687/1 "Gaussian Processes for Systems Identification with Applications in Systems Biology". MA also acknowledges the support from the Overseas Research Student Award Scheme (ORSAS), from the School of Computer Science of the University of Manchester and from the Universidad Tecnológica de Pereira, Colombia.

Appendix A. Derivatives for the Approximations

In this appendix, we present the derivatives needed to apply the gradient methods in the optimization routines. We present the first order derivatives of the log-likelihood with respect to $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$. These derivatives can be combined with the derivatives of $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ with respect to $\boldsymbol{\theta}$ and employ these expressions in a gradient-like optimization procedure.

We follow the notation of Brookes (2005) obtaining similar results to Lawrence (2007). This notation allows us to apply the chain rule for matrix derivation in a straight-forward manner. Let's define $\mathbf{G}_{:} = \mathrm{vec}\,\mathbf{G}$, where vec is the vectorization operator over the matrix $\mathbf{G}$. For a function $\mathcal{L}$ the equivalence between $\frac{\partial \mathcal{L}}{\partial \mathbf{G}}$ and $\frac{\partial \mathcal{L}}{\partial \mathbf{G}_{:}}$ is given through $\frac{\partial \mathcal{L}}{\partial \mathbf{G}_{:}} = \left(\left(\frac{\partial \mathcal{L}}{\partial \mathbf{G}}\right)_{:}\right)^{\top}$. The obtain the hyperparameters, we maximize the following log-likelihood function,

$$\mathcal{L}(\mathbf{Z},\boldsymbol{\theta}) \propto -\frac{1}{2}\log|\mathbf{D}+\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}| - \frac{1}{2}\,\mathrm{trace}\left[\left(\mathbf{D}+\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right)^{-1}\mathbf{y}\mathbf{y}^{\top}\right] \tag{19}$$

where we have redefined $\mathbf{D}$ as $\mathbf{D} = \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right] \odot \mathbf{M} + \boldsymbol{\Sigma}$, to keep a simpler notation. Using the matrix inversion lemma and its equivalent form for determinants, expression (19) can be written as

$$\begin{aligned}\mathcal{L}(\mathbf{Z},\boldsymbol{\theta}) \propto & \frac{1}{2}\log|\mathbf{K}_{\mathbf{u},\mathbf{u}}| - \frac{1}{2}\log|\mathbf{A}| - \frac{1}{2}\log|\mathbf{D}| - \frac{1}{2}\,\mathrm{trace}\left[\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\right] \\ & + \frac{1}{2}\,\mathrm{trace}\left[\mathbf{D}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\right].\end{aligned}$$

We can find $\frac{\partial \mathcal{L}}{\partial \boldsymbol{\theta}}$ and $\frac{\partial \mathcal{L}}{\partial \mathbf{Z}}$ applying the chain rule to $\mathcal{L}$ obtaining expressions for $\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{f},\mathbf{f}}}$, $\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{f},\mathbf{u}}}$ and $\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{u}}}$ and combining those with the derivatives of the covariances wrt $\boldsymbol{\theta}$ and $\mathbf{Z}$,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{G}_{:}} = \frac{\partial \mathcal{L}_{\mathbf{A}}}{\partial \mathbf{A}_{:}}\frac{\partial \mathbf{A}_{:}}{\partial \mathbf{D}_{:}}\frac{\partial \mathbf{D}_{:}}{\partial \mathbf{G}_{:}} + \frac{\partial \mathcal{L}_{\mathbf{D}}}{\partial \mathbf{D}_{:}}\frac{\partial \mathbf{D}_{:}}{\partial \mathbf{G}_{:}} + \left[\frac{\partial \mathcal{L}_{\mathbf{A}}}{\partial \mathbf{A}_{:}}\frac{\partial \mathbf{A}_{:}}{\partial \mathbf{G}_{:}} + \frac{\partial \mathcal{L}_{\mathbf{G}}}{\partial \mathbf{G}_{:}}\right]\delta_{GK}, \tag{20}$$

where the subindex in $\mathcal{L}_{\mathbf{E}}$ stands for those terms of $\mathcal{L}$ which depend on $\mathbf{E}$, $\mathbf{G}$ is either $\mathbf{K_{f,f}}$, $\mathbf{K_{u,f}}$ or $\mathbf{K_{u,u}}$ and δ_{GK} is zero if $\mathbf{G}$ is equal to $\mathbf{K_{f,f}}$ and one in other case. Next we present expressions for each partial derivative

$$\frac{\partial \mathcal{L}_{\mathbf{A}}}{\partial \mathbf{A:}} = -\frac{1}{2}(\mathbf{C:})^{\top}, \quad \frac{\partial \mathbf{A:}}{\partial \mathbf{D:}} = -\left(\mathbf{K_{u,f}}\mathbf{D}^{-1} \otimes \mathbf{K_{u,f}}\mathbf{D}^{-1}\right), \quad \frac{\partial \mathcal{L}_{\mathbf{D}}}{\partial \mathbf{D:}} = -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}\right)\mathbf{:}\right)^{\top}$$
$$\frac{\partial \mathbf{D:}}{\partial \mathbf{K_{f,f}:}} = \operatorname{diag}(\mathbf{M:}), \quad \frac{\partial \mathbf{D:}}{\partial \mathbf{K_{u,f}:}} = -\operatorname{diag}(\mathbf{M:})\left[\left(\mathbf{I} \otimes \mathbf{K_{f,u}}\mathbf{K_{u,u}}^{-1}\right) + \left(\mathbf{K_{f,u}}\mathbf{K_{u,u}}^{-1} \otimes \mathbf{I}\right)\mathbf{T_D}\right],$$
$$\frac{\partial \mathbf{D:}}{\partial \mathbf{K_{u,u}:}} = \operatorname{diag}(\mathbf{M:})\left(\mathbf{K_{f,u}}\mathbf{K_{u,u}}^{-1} \otimes \mathbf{K_{f,u}}\mathbf{K_{u,u}}^{-1}\right), \frac{\partial \mathbf{A:}}{\partial \mathbf{K_{u,f}:}} = \left(\mathbf{K_{u,f}}\mathbf{D}^{-1} \otimes \mathbf{I}\right) + \left(\mathbf{I} \otimes \mathbf{K_{u,f}}\mathbf{D}^{-1}\right)\mathbf{T_A}$$
$$\frac{\partial \mathbf{A:}}{\partial \mathbf{K_{u,u}:}} = \mathbf{I}, \quad \frac{\partial \mathcal{L}_{\mathbf{K_{u,f}}}}{\partial \mathbf{K_{u,f}:}} = \left(\left(\mathbf{A}^{-1}\mathbf{K_{u,f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\mathbf{D}^{-1}\right)\mathbf{:}\right)^{\top}, \quad \frac{\partial \mathcal{L}_{\mathbf{K_{u,u}}}}{\partial \mathbf{K_{u,u}:}} = \frac{1}{2}\left(\left(\mathbf{K_{u,u}}^{-1}\right)\mathbf{:}\right)^{\top},$$

where $\mathbf{C} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{K_{u,f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\mathbf{D}^{-1}\mathbf{K_{f,u}}\mathbf{A}^{-1}$, $\mathbf{T_D}$ and $\mathbf{T_A}$ are *vectorized transpose matrices* (see, e.g., Brookes, 2005) and $\mathbf{H} = \mathbf{D} - \mathbf{y}\mathbf{y}^{\top} + \mathbf{K_{f,u}}\mathbf{A}^{-1}\mathbf{K_{u,f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top} + \left(\mathbf{K_{f,u}}\mathbf{A}^{-1}\mathbf{K_{u,f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\right)^{\top}$. We can replace the above expressions in (20) to find the corresponding derivatives, so

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K_{f,f}:}} = \frac{1}{2}\left[((\mathbf{C})\mathbf{:})^{\top}\left(\mathbf{K_{u,f}}\mathbf{D}^{-1} \otimes \mathbf{K_{u,f}}\mathbf{D}^{-1}\right) - \frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}\right)\mathbf{:}\right)^{\top}\right]\operatorname{diag}(\mathbf{M:}) \tag{21}$$
$$= -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right)\mathbf{:}\right)^{\top}\operatorname{diag}(\mathbf{M:}) = -\frac{1}{2}\left(\operatorname{diag}(\mathbf{M:})\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right)\mathbf{:}\right)^{\top} \tag{22}$$
$$= -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1} \odot \mathbf{M}\right)\mathbf{:}\right)^{\top} = -\frac{1}{2}(\mathbf{Q:})^{\top} \tag{23}$$

or simply

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K_{f,f}}} = -\frac{1}{2}\mathbf{Q},$$

where $\mathbf{J} = \mathbf{H} - \mathbf{K_{f,u}}\mathbf{C}\mathbf{K_{u,f}}$ and $\mathbf{Q} = \left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1} \odot \mathbf{M}\right)$. We have used the property $(\mathbf{B:})^{\top}(\mathbf{F} \otimes \mathbf{P}) = \left(\left(\mathbf{P}^{\top}\mathbf{B}\mathbf{F}\right)\mathbf{:}\right)^{\top}$ in (21) and the property $\operatorname{diag}(\mathbf{B:})\mathbf{F:} = (\mathbf{B} \odot \mathbf{F})\mathbf{:}$, to go from (22) to (23). We also have

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{K_{u,f}:}} = &\frac{1}{2}(\mathbf{Q:})^{\top}\left[\left(\mathbf{I} \otimes \mathbf{K_{f,u}}\mathbf{K_{u,u}}^{-1}\right) + \left(\mathbf{K_{f,u}}\mathbf{K_{u,u}}^{-1} \otimes \mathbf{I}\right)\mathbf{T_D}\right] - \frac{1}{2}(\mathbf{C:})^{\top} \\ &\left[\left(\mathbf{K_{u,f}}\mathbf{D}^{-1} \otimes \mathbf{I}\right) + \left(\mathbf{I} \otimes \mathbf{K_{u,f}}\mathbf{D}^{-1}\right)\mathbf{T_A}\right] + \left(\left(\mathbf{A}^{-1}\mathbf{K_{u,f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\mathbf{D}^{-1}\right)\mathbf{:}\right)^{\top} \\ = &\left(\left(\mathbf{K_{u,u}}^{-1}\mathbf{K_{u,f}}\mathbf{Q} - \mathbf{C}\mathbf{K_{u,f}}\mathbf{D}^{-1} + \mathbf{A}^{-1}\mathbf{K_{u,f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\mathbf{D}^{-1}\right)\mathbf{:}\right)^{\top}\end{aligned} \tag{24}$$

or simply

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K_{u,f}}} = \mathbf{K_{u,u}}^{-1}\mathbf{K_{u,f}}\mathbf{Q} - \mathbf{C}\mathbf{K_{u,f}}\mathbf{D}^{-1} + \mathbf{A}^{-1}\mathbf{K_{u,f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^{\top}\mathbf{D}^{-1},$$

where in (24), $(\mathbf{Q:})^{\top}(\mathbf{F} \otimes \mathbf{I})\mathbf{T_D} = (\mathbf{Q:})^{\top}\mathbf{T_D}(\mathbf{I} \otimes \mathbf{F}) = \left(\mathbf{T_D}^{\top}\mathbf{Q:}\right)^{\top}(\mathbf{I} \otimes \mathbf{F}) = (\mathbf{Q:})^{\top}(\mathbf{I} \otimes \mathbf{F})$. A similar analysis is formulated for the term involving $\mathbf{T_A}$. Finally, results for $\frac{\partial \mathcal{L}}{\partial \mathbf{K_{u,f}}}$ and $\frac{\partial \mathcal{L}}{\partial \mathbf{\Sigma}}$ are obtained as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K_{u,u}}} = -\frac{1}{2}\left(\mathbf{K_{u,u}}^{-1} - \mathbf{C} - \mathbf{K_{u,u}}^{-1}\mathbf{K_{u,f}}\mathbf{Q}\mathbf{K_{f,u}}\mathbf{K_{u,u}}^{-1}\right), \quad \frac{\partial \mathcal{L}}{\partial \mathbf{\Sigma}} = -\frac{1}{2}\mathbf{Q}.$$

References

Mauricio A. Álvarez and Neil D. Lawrence. Sparse convolved Gaussian processes for multi-output regression. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 57–64. MIT Press, Cambridge, MA, 2009.

Mauricio A. Álvarez, David Luengo, and Neil D. Lawrence. Latent Force Models. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pages 9–16. JMLR W&CP 5, Clearwater Beach, Florida, 16-18 April 2009.

Mauricio A. Álvarez, David Luengo, Michalis K. Titsias, and Neil D. Lawrence. Efficient multioutput Gaussian processes through variational inducing kernels. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 25–32. JMLR W&CP 9, Chia Laguna, Sardinia, Italy, 13-15 May 2010.

Mauricio A. Álvarez, Jan Peters, Bernhard Schölkopf, and Neil D. Lawrence. Switched latent force models for movement segmentation. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 55–63. MIT Press, Cambridge, MA, 2011a.

Mauricio A. Álvarez, Lorenzo Rosasco, and Neil D. Lawrence. Kernels for vector-valued functions: a review, 2011b. Universidad Tecnológica de Pereira, Massachusetts Institute of Technology and University of Sheffield. In preparation.

Bart Bakker and Tom Heskes. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research*, 4:83–99, 2003.

Ronald Paul Barry and Jay M. Ver Hoef. Blackbox kriging: spatial prediction without specifying variogram models. *Journal of Agricultural, Biological and Environmental Statistics*, 1(3):297–322, 1996.

Edwin V. Bonilla, Kian Ming Chai, and Christopher K. I. Williams. Multi-task Gaussian process prediction. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20*, pages 153–160. MIT Press, Cambridge, MA, 2008.

Phillip Boyle and Marcus Frean. Dependent Gaussian processes. In Lawrence K. Saul, Yair Weiss, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 17*, pages 217–224. MIT Press, Cambridge, MA, 2005.

Michael Brookes. The matrix reference manual. Available on-line., 2005. `http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html`.

Catherine A. Calder. *Exploring Latent Structure in Spatial Temporal Processes Using Process Convolutions*. PhD thesis, Institute of Statistics and Decision Sciences, Duke University, Durham, NC, USA, 2003.

Catherine A. Calder. Dynamic factor process convolution models for multivariate space-time data with application to air quality assessment. *Environmental and Ecological Statistics*, 14(3):229–247, 2007.

Catherine A. Calder and Noel Cressie. Some topics in convolution-based spatial modeling. In *Proceedings of the 56th Session of the International Statistics Institute*, August 2007.

Rich Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

Kian Ming A. Chai, Christopher K. I. Williams, Stefan Klanke, and Sethu Vijayakumar. Multi-task Gaussian process learning of robot inverse dynamics. In D. Koller, D. Schuurmans, Y. Bengio, and L. Bottou, editors, *Advances in Neural Information Processing Systems 21*, pages 265–272. MIT Press, Cambridge, MA, 2009.

Noel A. C. Cressie. *Statistics for Spatial Data*. John Wiley & Sons (Revised edition), USA, 1993.

Lehel Csató and Manfred Opper. Sparse representation for Gaussian process models. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 444–450. MIT Press, Cambridge, MA, 2001.

Theodoros Evgeniou and Massimiliano Pontil. Regularized Multi-task Learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117, 2004.

Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

Montserrat Fuentes. Interpolation of nonstationary air pollution processes: a spatial spectral approach. *Statistical Modelling*, 2:281–298, 2002a.

Montserrat Fuentes. Spectral methods for nonstationary spatial processes. *Biometrika*, 89(1):197–210, 2002b.

Pei Gao, Antti Honkela, Magnus Rattray, and Neil D. Lawrence. Gaussian process modelling of latent chemical species: Applications to inferring transcription factor activities. *Bioinformatics*, 24:i70–i75, 2008. doi: 10.1093/bioinformatics/btn278.

Marc G. Genton. Classes of kernels for machine learning: A statistics perspective. *Journal of Machine Learning Research*, 2:299–312, 2001.

Harvey Goldstein. Multilevel modelling of survey data. *The Statistician*, 40(2):235–244, 1991.

Pierre Goovaerts. *Geostatistics For Natural Resources Evaluation*. Oxford University Press, USA, 1997.

Michel Goulard and Marc Voltz. Linear coregionalization model: Tools for estimation and choice of cross-variogram matrix. *Mathematical Geology*, 24(3):269–286, 1992.

Jeffrey D. Helterbrand and Noel Cressie. Universal cokriging under intrinsic coregionalization. *Mathematical Geology*, 26(2):205–226, 1994.

Tom Heskes. Empirical Bayes for learning to learn. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning 17*, pages 367–374. Morgan Kaufmann, San Francisco, CA, June 29-July 2 2000.

David M. Higdon. A process-convolution approach to modeling temperatures in the north atlantic ocean. *Journal of Ecological and Environmental Statistics*, 5:173–190, 1998.

David M. Higdon. Space and space-time modelling using process convolutions. In C. Anderson, V. Barnett, P. Chatwin, and A. El-Shaarawi, editors, *Quantitative Methods for Current Environmental Issues*, pages 37–56. Springer-Verlag, 2002.

David M. Higdon, Jenise Swall, and John Kern. Non-stationary spatial modeling. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 6*, pages 761–768. Oxford University Press, 1998.

Antti Honkela, Charles Girardot, E. Hilary Gustafson, Ya-Hsin Liu, Eileen E. M. Furlong, Neil D. Lawrence, and Magnus Rattray. Model-based method for transcription factor target identification with limited data. *Proc. Natl. Acad. Sci.*, 107(17):7793–7798, 2010.

Andre G. Journel and Charles J. Huijbregts. *Mining Geostatistics*. Academic Press, London, 1978. ISBN 0-12391-050-1.

Neil D. Lawrence. Learning for larger datasets with the Gaussian process latent variable model. In Marina Meila and Xiaotong Shen, editors, *AISTATS 11*, pages 243–250. Omnipress, San Juan, Puerto Rico, 21-24 March 2007.

Neil D. Lawrence, Matthias Seeger, and Ralf Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In Sue Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 625–632. MIT Press, Cambridge, MA, 2003.

Neil D. Lawrence, Guido Sanguinetti, and Magnus Rattray. Modelling transcriptional regulation using Gaussian processes. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems 19*, pages 785–792. MIT Press, Cambridge, MA, 2007.

Feng Liang, Kai Mao, Ming Liao, Sayan Mukherjee, and Mike West. Non-parametric Bayesian kernel models. Department of Statistical Science, Duke University, Discussion Paper 07-10. (Submitted for publication), 2009.

Desmond L. Nuttall, Harvey Goldstein, Robert Prosser, and Jon Rasbash. Differential school effectiveness. *International Journal of Educational Research*, 13(7):769–776, 1989.

Michael A. Osborne and Stephen J. Roberts. Gaussian processes for prediction. Technical report, Department of Engineering Science, University of Oxford, 2007.

Michael A. Osborne, Alex Rogers, Sarvapali D. Ramchurn, Stephen J. Roberts, and Nicholas 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.

Christopher J. Paciorek and Mark J. Schervish. Nonstationary covariance functions for Gaussian process regression. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

Natesh S. Pillai, Qiang Wu, Feng Liang, Sayan Mukherjee, and Robert L. Wolpert. Characterizing the function space for Bayesian kernel models. *Journal of Machine Learning Research*, 8:1769–1797, 2007.

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.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 0-262-18253-X.

Matthias Seeger, Christopher K. I. Williams, and Neil D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In Christopher M. Bishop and Brendan J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, FL, 3–6 Jan 2003.

Alexander J. Smola and Peter L. Bartlett. Sparse greedy Gaussian process regression. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 619–625. MIT Press, Cambridge, MA, 2001.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Yair Weiss, Bernhard Schölkopf, and John C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 1257–1264. MIT Press, Cambridge, MA, 2006.

Edward Snelson and Zoubin Ghahramani. Local and global sparse Gaussian process approximations. In Marina Meila and Xiaotong Shen, editors, *AISTATS 11*, pages 524–531, San Juan, Puerto Rico, 21-24 March 2007. Omnipress.

Yee Whye Teh, Matthias Seeger, and Michael I. Jordan. Semiparametric latent factor models. In Robert G. Cowell and Zoubin Ghahramani, editors, *AISTATS 10*, pages 333–340. Society for Artificial Intelligence and Statistics, Barbados, 6-8 January 2005.

Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In David van Dyk and Max Welling, editors, *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pages 567–574. JMLR W&CP 5, Clearwater Beach, Florida, 16-18 April 2009.

Pavel Tomancak, Amy Beaton, Richard Weiszmann, Elaine Kwan, ShengQiang Shu, Suzanna E Lewis, Stephen Richards, Michael Ashburner, Volker Hartenstein, Susan E Celniker, and Gerald M Rubin. Systematic determination of patterns of gene expression during drosophila embryogenesis. *Genome Biology*, 3(12):research0088.1–0088.14, 2002.

Jay M. Ver Hoef and Ronald Paul Barry. Constructing and fitting models for cokriging and multivariable spatial prediction. *Journal of Statistical Plannig and Inference*, 69:275–294, 1998.

Hans Wackernagel. *Multivariate Geostatistics*. Springer-Verlag Heidelberg New York, 2003.

Christopher K. Wikle. A kernel-based spectral model for non-Gaussian spatio-temporal processes. *Statistical Modelling*, 2:299–314, 2002.

Christopher K. Wikle. Hierarchical Bayesian models for predicting the spread of ecological processes. *Ecology*, 84(6):1382–1394, 2003.

Christopher K. Wikle, L. Mark Berliner, and Noel Cressie. Hierarchical Bayesian space-time models. *Environmental and Ecological Statistics*, 5:117–154, 1998.

Christopher K. I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688. MIT Press, Cambridge, MA, 2001.

Ya Xue, Xuejun Liao, and Lawrence Carin. Multi-task learning for classification with Dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

Robert P. Zinzen, Charles Girardot, Julien Gagneur, Martina Braun, and Eileen E. M. Furlong. Combinatorial binding predicts spatio-temporal cis-regulatory activity. *Nature*, 462:65–70, 2009.

Bibliography

Uri Alon. *An Introduction to Systems Biology: Design Principles of Biological Circuits*. Chapman and Hall/CRC, London, 2006. ISBN 1-58488-642-0.

Mauricio A. Álvarez, David Luengo, and Neil D. Lawrence. Latent Force Models. In van Dyk and Welling (2009), pages 9–16.

Mauricio A. Álvarez, David Luengo, Michalis K. Titsias, and Neil D. Lawrence. Variational inducing kernels for sparse convolved multiple output Gaussian processes. Technical report, School of Computer Science, University of Manchester, UK and Departamento de Teoría de la Señal y Comunicaciones, Universidad Carlos III, Spain, 2009. Available at `http://arxiv.org/pdf/0912.3268`.

Mauricio A. Álvarez, David Luengo, Michalis K. Titsias, and Neil D. Lawrence. Efficient multioutput Gaussian processes through variational inducing kernels. In Teh and Titterington (2010), pages 25–32.

Bart Bakker and Tom Heskes. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research*, 4:83–99, 2003.

Yonathan Bard. *Nonlinear Parameter Estimation*. Academic Press, first edition, 1974.

Martino Barenco, Daniela Tomescu, Daniel Brewer, Robin Callard, Jaroslav Stark, and Michael Hubank. Ranked prediction of p53 targets using hidden variable dynamic modeling. *Genome Biology*, 7(3):R25, 2006.

Ronald Paul Barry and Jay M. Ver Hoef. Blackbox kriging: spatial prediction without specifying variogram models. *Journal of Agricultural, Biological and Environmental Statistics*, 1(3):297–322, 1996.

Sue Becker, Sebastian Thrun, and Klaus Obermayer, editors. *NIPS*, volume 15, Cambridge, MA, 2003. MIT Press.

Yoshua Bengio, Dale Schuurmans, John Laferty, Chris Williams, and Aron Culotta, editors. *NIPS*, volume 22, Cambridge, MA, 2010. MIT Press.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

Edwin V. Bonilla, Kian Ming Chai, and Christopher K. I. Williams. Multi-task Gaussian process prediction. In Platt et al. (2008).

Phillip Boyle. *Gaussian Processes for Regression and Optimisation*. PhD thesis, Victoria University of Wellington, Wellington, New Zealand, 2007.

Phillip Boyle and Marcus Frean. Dependent Gaussian processes. In Saul et al. (2005), pages 217–224.

Phillip Boyle and Marcus Frean. Multiple output Gaussian process regression. Technical Report CS-TR-05/2, School of Mathematical and Computing Sciences, Victoria University, New Zealand, 2005b.

Daniel Brewer, Martino Barenco, Robin Callard, Michael Hubank, and Jaroslav Stark. Fitting ordinary differential equations to short time course data. *Philosophical Transactions of the Royal Society A*, 366:519–544, 2008.

Michael Brookes. The matrix reference manual. Available on-line., 2005. `http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html`.

Marcus A. Brubaker, Leonid Sigal, and David J. Fleet. Physics-based human motion modeling for people tracking: A short tutorial. Technical report, Department of Computer Science, University of Toronto and Disney Research, 2009.

Catherine A. Calder. Dynamic factor process convolution models for multivariate space-time data with application to air quality assessment. *Environmental and Ecological Statistics*, 14(3):229–247, 2007.

Catherine A. Calder. A dynamic process convolution approach to modeling ambient particulate matter concentrations. *Environmetrics*, 19:39–48, 2008.

Catherine A. Calder. *Exploring latent structure in spatial temporal processes using process convolutions*. PhD thesis, Institute of Statistics and Decision Sciences, Duke University, Durham, NC, USA, 2003.

Catherine A. Calder and Noel Cressie. Some topics in convolution-based spatial modeling. In *Proceedings of the 56th Session of the International Statistics Institute*, August 2007.

Rich Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997.

Kian Ming Chai. Generalization errors and learning curves for regression with multi-task Gaussian processes. In Bengio et al. (2010), pages 279–287.

Kian Ming A. Chai, Christopher K. I. Williams, Stefan Klanke, and Sethu Vijayakumar. Multi-task Gaussian process learning of robot inverse dynamics. In Koller et al. (2009), pages 265–272.

Silvia Chiappa and Jan Peters. Movement extraction by detecting dynamics switches and repetitions. In *NIPS*, volume 24, pages 388–396. MIT Press, Cambridge, MA, 2011.

Silvia Chiappa, Jens Kober, and Jan Peters. Using Bayesian dynamical systems for motion template libraries. In Koller et al. (2009), pages 297–304.

Stefano Conti and Anthony O'Hagan. Bayesian emulation of complex multi-output and dynamic computer models. *Journal of Statistical Planning and Inference*, 140(3):640–651, 2010.

Noel A. C. Cressie. *Statistics for Spatial Data*. John Wiley & Sons (Revised edition), USA, 1993.

Lehel Csató and Manfred Opper. Sparse representation for Gaussian process models. In Leen et al. (2001), pages 444–450.

Marc Peter Deisenroth, Marco F. Huber, and Uwe D. Hanebeck. Analytic moment-based Gaussian process filtering. In *Proceedings of the 26th International Conference on Machine Learning (ICML 2009)*, pages 225–232. Omnipress, 2009.

Theodoros Evgeniou and Massimiliano Pontil. Regularized Multi-task Learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117, 2004.

Theodoros Evgeniou, Charles A. Micchelli, and Massimiliano Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6: 615–637, 2005.

Tamar Flash and Binyamin Hochner. Motor primitives in vertebrates and invertebrates. *Current Opinion in Neurobiology*, 15:660–666, 2005.

Montserrat Fuentes. Interpolation of nonstationary air pollution processes: a spatial spectral approach. *Statistical Modelling*, 2:281–298, 2002a.

Montserrat Fuentes. Spectral methods for nonstationary spatial processes. *Biometrika*, 89(1):197–210, 2002b.

Pei Gao, Antti Honkela, Magnus Rattray, and Neil D. Lawrence. Gaussian process modelling of latent chemical species: Applications to inferring transcription factor activities. *Bioinformatics*, 24:i70–i75, 2008. doi: 10.1093/bioinformatics/btn278.

Roman Garnett, Michael A. Osborne, and Stephen J. Roberts. Sequential Bayesian prediction in the presence of changepoints. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 345–352, 2009.

Roman Garnett, Michael A. Osborne, Steven Reece, Alex Rogers, and Stephen J. Roberts. Sequential Bayesian prediction in the presence of changepoints and faults. *The Computer Journal*, 2010. Advance Access published February 1, 2010.

Marc G. Genton. Classes of kernels for machine learning: A statistics perspective. *Journal of Machine Learning Research*, 2:299–312, 2001.

Agathe Girard, Carl Edward Rasmussen, Joaquin Quiñonero Candela, and Roderick Murray-Smith. Gaussian process priors with uncertain inputs – application to multiple-step ahead time series forecasting. In Becker et al. (2003), pages 529–536.

Tilmann Gneiting, Zoltán Sasvári, and Martin Schlather. Analogies and correspondences between variograms and covariance functions. *Advances in Applied Probability*, 33(3):617–630, 2001.

Pierre Goovaerts. *Geostatistics For Natural Resources Evaluation*. Oxford University Press, USA, 1997.

Michel Goulard and Marc Voltz. Linear coregionalization model: Tools for estimation and choice of cross-variogram matrix. *Mathematical Geology*, 24(3): 269–286, 1992.

Thore Graepel. Solving noisy linear operator equations by Gaussian processes: Application to ordinary and partial differential equations. In Tom Fawcett and Nina Mishra, editors, *Proceedings of the Twentieth International Conference on Machine Learning*, pages 234–241, Washington, DC, USA, August 21-24 2003. AAAI Press.

D. H. Griffel. *Applied Functional Analysis*. Dover publications Inc., Mineola, New York, reprinted edition, 2002.

Jeffrey D. Helterbrand and Noel Cressie. Universal cokriging under intrinsic coregionalization. *Mathematical Geology*, 26(2):205–226, 1994.

Dave Higdon, Jim Gattiker, Brian Williams, and Maria Rightley. Computer model calibration using high dimensional output. *Journal of the American Statistical Association*, 103(482):570–583, 2008.

David M. Higdon. Space and space-time modelling using process convolutions. In C. Anderson, V. Barnett, P. Chatwin, and A. El-Shaarawi, editors, *Quantitative methods for current environmental issues*, pages 37–56. Springer-Verlag, 2002.

David M. Higdon. A process-convolution approach to modeling temperatures in the north atlantic ocean. *Journal of Ecological and Environmental Statistics*, 5:173–190, 1998.

David M. Higdon, Jenise Swall, and John Kern. Non-stationary spatial modeling. In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics 6*, pages 761–768. Oxford University Press, 1998.

Antti Honkela, Charles Girardot, E. Hilary Gustafson, Ya-Hsin Liu, Eileen E. M. Furlong, Neil D. Lawrence, and Magnus Rattray. Model-based method for transcription factor target identification with limited data. *Proc. Natl. Acad. Sci.*, 107(17):7793–7798, 2010.

Lars Hörmander. *The analysis of Linear Partial Differential Operators I.* Springer-Verlag, Berlin Hiedelberg, first edition, 1983.

Auke Jan Ijspeert, Jun Nakanishi, and Stefan Schaal. Learning attractor landscapes for learning motor primitives. In Becker et al. (2003), pages 1523–1530.

Andre G. Journel and Charles J. Huijbregts. *Mining Geostatistics.* Academic Press, London, 1978. ISBN 0-12391-050-1.

Jonathan Ko, Daniel J. Klein, Dieter Fox, and Dirk Haehnel. GP-UKF: Unscented Kalman filters with Gaussian process prediction and observation models. In *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1901–1907, San Diego, CA, USA, 2007.

Juš Kocijan, Agathe Girard, Blaž Banko, and Roderick Murray-Smith. Dynamic systems identification with Gaussian processes. *Mathematical and Computer Modelling of Dynamical Systems*, 11(4):411–424, 2005.

Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors. *NIPS*, volume 21, Cambridge, MA, 2009. MIT Press.

H.R. Künsch, A. Papritz, and F. Bassi. Generalized cross-covariances and their estimation. *Mathematical Geology*, 29(6):779–799, 1997.

R.M. Lark and A. Papritz. Fitting a linear model of coregionalization for soil properties using simulated annealing. *Geoderma*, 115:245–260, 2003.

Neil D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, Nov. 2005.

Neil D. Lawrence and John C. Platt. Learning to learn with the informative vector machine. In *Proceedings of the 21st International Conference on Machine Learning (ICML 2004)*, pages 512–519, 2004.

Neil D. Lawrence, Matthias Seeger, and Ralf Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In Becker et al. (2003), pages 625–632.

Neil D. Lawrence, Guido Sanguinetti, and Magnus Rattray. Modelling transcriptional regulation using Gaussian processes. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *NIPS*, volume 19, pages 785–792. MIT Press, Cambridge, MA, 2007.

Miguel Lázaro-Gredilla and Aníbal Figueiras-Vidal. Inter-domain Gaussian processes for sparse inference using inducing features. In Bengio et al. (2010), pages 1087–1095.

Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors. *NIPS*, volume 13, Cambridge, MA, 2001. MIT Press.

Feng Liang, Kai Mao, Ming Liao, Sayan Mukherjee, and Mike West. Nonparametric Bayesian kernel models. Department of Statistical Science, Duke University, Discussion Paper 07-10. (Submitted for publication), 2009.

Lennart Ljung. *System Identification: Theory for the User*. Prentice Hall PTR, Upper Saddle River, New Jersey, second edition, 1999.

Anandamayee Majumdar and Alan E. Gelfand. Multivariate spatial modeling for geostatistical data using convolved covariance functions. *Mathematical Geology*, 39(2):225–244, 2007.

Georges Matheron. The intrinsic random functions and their applications. *Advances in Applied Probability*, 5(3):439–468, 1973.

Charles A. Micchelli and Massimiliano Pontil. Kernels for multi–task learning. In Saul et al. (2005), pages 921–928.

Thomas P. Minka and Rosalind W. Picard. Learning how to learn is learning with point sets, 1999. Revised version 1999 available at `http://research.microsoft.com/en-us/um/people/minka/papers/point-sets.html`.

Toby J. Mitchell and Max Morris. Asymptotically optimum experimental designs for prediction of deterministic functions given derivative information. *Journal of Statistical Planning and Inference*, 41:377–389, 1994.

Max D. Morris, Toby J. Mitchell, and Donald Ylvisaker. Bayesian design and analysis of computer experiments: use of derivatives in surface prediction. *Technometrics*, 35(3):243–255, 1993.

Roderick Murray-Smith and Barak A. Pearlmutter. Transformation of Gaussian process priors. In Joab Winkler, Mahesan Niranjan, and Neil Lawrence, editors, *Deterministic and Statistical Methods in Machine Learning*, pages 110–123. LNAI 3635, Springer-Verlag, 2005.

Anthony O'Hagan. Bayesian analysis of computer code outputs: A tutorial. *Reliability Engineering and System Safety*, 91:1290–1300, 2006.

Bert Øksendal. *Stochastic Differential Equations: An Introduction with Applications*. Springer, Germany, sixth edition, 2003.

Michael A. Osborne and Stephen J. Roberts. Gaussian processes for prediction. Technical report, Department of Engineering Science, University of Oxford, 2007.

Michael A. Osborne, Alex Rogers, Sarvapali D. Ramchurn, Stephen J. Roberts, and Nicholas 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.

Christopher J. Paciorek and Mark J. Schervish. Nonstationary covariance functions for Gaussian process regression. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

Bernard Pelletier, Pierre Dutilleul, Guillaume Larocque, and James W. Fyles. Fitting the linear model of coregionalization by generalized least squares. *Mathematical Geology*, 36(3):323–343, 2004.

Theodore J. Perkins, Johannes Jaeger, John Reinitz, and Leon Glass. Reverse engineering the gap gene network of Drosophila melanogaster. *PLOS Comput Biol*, 2(5):417–427, 2006.

Jan Peters. *Machine Learning of Motor Skills for Robotics*. PhD thesis, Department of Computer Science, University of Southern California, California, USA, 2007.

Natesh S. Pillai, Qiang Wu, Feng Liang, Sayan Mukherjee, and Robert L. Wolpert. Characterizing the function space for Bayesian kernel models. *Journal of Machine Learning Research*, 8:1769–1797, 2007.

John C. Platt, Daphne Koller, Yoram Singer, and Sam Roweis, editors. *NIPS*, volume 20, Cambridge, MA, 2008. MIT Press.

Andrei D. Polyanin. *Handbook of Linear Partial Differential Equations for Engineers and Scientists*. Chapman & Hall/CRC Press, 2002.

A.A. Poyton, M. Saeed Varziri, Kim B. McAuley, James McLellan, and Jim O. Ramsay. Parameter estimation in continuous-time dynamic models using principal differential analysis. *Computers and Chemical Engineering*, 30:698–708, 2006.

Joaquin Quiñonero-Candela and Carl Edward Rasmussen. Analysis of some methods for reduced rank Gaussian process regression. In R. Murray-Smith and R. Shorten, editors, *Lecture Notes in Computer Science*, volume 3355, pages 98–127. Springer, 2005a.

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

Jim O. Ramsay. Principal differential analysis: Data reduction by differential operators. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(3):495–508, 1996.

Jim O. Ramsay and Bernard W. Silverman. *Functional Data Analysis*. Springer Series in Statistics, New York, NY, (USA), second edition, 2005.

Jim O. Ramsay, G. Hooker, D. Campbell, and J. Cao. Parameter estimation for differential equations: a generalized smoothing approach. *Journal of the Royal Statistical Society. Series B (Methodological)*, 69(5):741–796, 2007.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 0-262-18253-X.

Linda E. Reichl. *A modern course in statistical physics*. John Wiley & Sons, United States of America, second edition, 1998.

Gary F. Roach. *Green's functions*. Cambridge University Press, Cambridge, UK, second edition, 1982.

Jonathan Rougier. Efficient emulators for multivariate deterministic functions. *Journal of Computational and Graphical Statistics*, 17(4):827–834, 2008.

Bryan P. Rynne and Martin A. Youngson. *Linear Functional Analysis*. Springer-Verlag London, second edition, 2008.

Yunus Saatçi, Ryan Turner, and Carl Edward Rasmussen. Gaussian process change point models. In *Proceedings of the 27th Annual International Conference on Machine Learning*, pages 927–934, 2010.

Lawrence Saul, Yair Weiss, and Léon Bouttou, editors. *NIPS*, volume 17, Cambridge, MA, 2005. MIT Press.

Stefan Schaal, Auke Ijspeert, and Aude Billard. Computational approaches to motor learning by imitation. *Philosophical Transactions of the Royal Society B, Biological Sciences*, 358:537–547, 2003.

Stefan Schaal, Peyman Mohajerian, and Auke Ijspeert. *Progress in Brain Research*, chapter Dynamics systems vs. optimal control - a unifying view, pages 425–445. Elsevier B.V., 2007.

Matthias Seeger, Christopher K. I. Williams, and Neil D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In Christopher M. Bishop and Brendan J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Key West, FL, 3–6 Jan 2003.

Sam K. Shanmugan and Arthur M. Breipohl. *Random signals: detection, estimation and data analysis*. John Wiley & Sons, United States of America, first edition, 1988.

Jian Qing Shi, Roderick Murray-Smith, D.M. Titterington, and Barak Pearlmutter. Learning with large data sets using filtered Gaussian process priors. In R. Murray-Smith and R. Shorten, editors, *Proceedings of the Hamilton Summer School on Switching and Learning in Feedback systems*, pages 128–139. LNCS 3355, Springer-Verlag, 2005.

Alexander J. Smola and Peter L. Bartlett. Sparse greedy Gaussian process regression. In Leen et al. (2001), pages 619–625.

Edward Snelson and Zoubin Ghahramani. Local and global sparse Gaussian process approximations. In Marina Meila and Xiaotong Shen, editors, *AISTATS 11*, San Juan, Puerto Rico, 21-24 March 2007. Omnipress.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Yair Weiss, Bernhard Schölkopf, and John C. Platt, editors, *NIPS*, volume 18, Cambridge, MA, 2006. MIT Press.

Ercan Solak, Roderick Murray-Smith, William E. Leithead, Douglas J. Leith, and Carl E. Rasmussen. Derivative observations in Gaussian process models of dynamic systems. In Becker et al. (2003), pages 1033–1040.

Michael L. Stein. *Interpolation of Spatial Data*. Springer-Verlag New York, Inc., first edition, 1999.

Yee Whye Teh and Mike Titterington, editors. *AISTATS*, Chia Laguna, Sardinia, Italy, 13-15 May 2010. JMLR W&CP 9.

Yee Whye Teh, Matthias Seeger, and Michael I. Jordan. Semiparametric latent factor models. In Robert G. Cowell and Zoubin Ghahramani, editors, *AISTATS 10*, pages 333–340, Barbados, 6-8 January 2005. Society for Artificial Intelligence and Statistics.

Keith R. Thompson. *Implementation of Gaussian Process Models for nonlinear system Identification*. PhD thesis, Department of Electronics and Electrical Engineering, University of Glasgow, Glasgow, UK, 2009.

Sebastian Thrun. Is learning the n-thing any easier than learning the first? In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *NIPS*, volume 08, pages 640–646. MIT Press, Cambridge, MA, 1996.

Michalis Titsias, Neil D Lawrence, and Magnus Rattray. Efficient sampling for Gaussian process inference using control variables. In Koller et al. (2009), pages 1681–1688.

Michalis K. Titsias. Variational learning of inducing variables in sparse Gaussian processes. In van Dyk and Welling (2009), pages 567–574.

Pavel Tomancak, Amy Beaton, Richard Weiszmann, Elaine Kwan, ShengQiang Shu, Suzanna E Lewis, Stephen Richards, Michael Ashburner, Volker Hartenstein, Susan E Celniker, and Gerald M Rubin. Systematic determination of patterns of gene expression during drosophila embryogenesis. *Genome Biology*, 3(12):research0088.1–0088.14, 2002.

Ryan Turner, Marc Peter Deisenroth, and Carl Edward Rasmussen. State-space inference and learning with Gaussian processes. In Teh and Titterington (2010), pages 868–875.

David van Dyk and Max Welling, editors. *AISTATS*, Clearwater Beach, Florida, 16-18 April 2009. JMLR W&CP 5.

Domitilla Del Vecchio, Richard M. Murray, and Pietro Perona. Decomposition of human motion into dynamics-based primitives with application to drawing tasks. *Automatica*, pages 2085–2098, 2003.

Jay M. Ver Hoef and Ronald Paul Barry. Constructing and fitting models for cokriging and multivariable spatial prediction. *Journal of Statistical Plannig and Inference*, 69:275–294, 1998.

Jay M. Ver Hoef, Noel Cressie, and Ronald Paul Barry. Flexible spatial models for kriging and cokriging using moving averages and the Fast Fourier Transform (FFT). *Journal of Computational and Graphical Statistics*, 13(2):265–282, 2004.

Hans Wackernagel. *Multivariate Geostatistics*. Springer-Verlag Heidelberg New york, 2003.

Christian Walder, Kwang In Kim, and Bernhard Schölkopf. Sparse multiscale Gaussian process regression. In *Proceedings of the 25th international conference on Machine learning*, ICML 2008, pages 1112–1119, 2008.

Christopher K. Wikle. Hierarchical Bayesian models for predicting the spread of ecological processes. *Ecology*, 84(6):1382–1394, 2003.

Christopher K. Wikle. A kernel-based spectral model for non-Gaussian spatio-temporal processes. *Statistical Modelling*, 2:299–314, 2002.

Christopher K. Wikle, L. Mark Berliner, and Noel Cressie. Hierarchical Bayesian space-time models. *Environmental and Ecological Statistics*, 5:117–154, 1998.

Ben Williams, Marc Toussaint, and Amos Storkey. Modelling motion primitives and their timing in biologically executed movements. In Platt et al. (2008), pages 1609–1616.

Christopher K. I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In Leen et al. (2001), pages 682–688.

Ian Woodward, Mark R. Lomas, and Richard A. Betts. Vegetation-climate feedbacks in a greenhouse world. *Philosophical Transactions: Biological Sciences*, 353(1365):29–39, 1998.

Ya Xue, Xuejun Liao, and Lawrence Carin. Multi-task learning for classification with Dirichlet process priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

Kai Yu, Volker Tresp, and Anton Schwaighofer. Learning Gaussian processes from multiple tasks. In *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, pages 1012–1019, 2005.

Hao Zhang. Maximum-likelihood estimation for multivariate spatial linear coregionalization models. *Environmetrics*, 18:125–139, 2007.

Robert P. Zinzen, Charles Girardot, Julien Gagneur, Martina Braun, and Eileen E. M. Furlong. Combinatorial binding predicts spatio-temporal cis-regulatory activity. *Nature*, 462:65–70, 2009.