Journal of Machine Learning Research 12 (2011) 1459-1500 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.

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 $\{u_q^i(\mathbf{x})\}_{i=1}^{R_q}$, $i = 1, \dots, R_q$, 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 $\mathbf{f} = [\mathbf{f}_1^\top, \dots, \mathbf{f}_D^\top]^\top$. 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

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

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 $q = 1, \dots, Q$ and $i = 1, \dots, R_q$ 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

$$k_q(\mathbf{x},\mathbf{x}') = \mathcal{N}(\mathbf{x}-\mathbf{x}'|\mathbf{0},\boldsymbol{\Lambda}_q^{-1}),$$

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

$$\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1,\mathbf{P}_1^{-1})\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_2,\mathbf{P}_2^{-1}) = \mathcal{N}(\boldsymbol{\mu}_1|\boldsymbol{\mu}_2,\mathbf{P}_1^{-1}+\mathbf{P}_2^{-1})\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_c,\mathbf{P}_c^{-1}), \tag{11}$$

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

$$k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = \sum_{q=1}^{Q} S_{d,q}S_{d',q}\int_{\mathcal{X}}\mathcal{N}(\mathbf{x}-\mathbf{z}|\mathbf{0},\mathbf{P}_d^{-1})\int_{\mathcal{X}}\mathcal{N}(\mathbf{x}'-\mathbf{z}'|\mathbf{0},\mathbf{P}_{d'}^{-1})\mathcal{N}(\mathbf{z}-\mathbf{z}'|\mathbf{0},\boldsymbol{\Lambda}_q^{-1})\mathrm{d}\mathbf{z}'\mathrm{d}\mathbf{z}.$$

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

$$k_{f_d,f_{d'}}(\mathbf{x},\mathbf{x}') = \sum_{q=1}^{Q} S_{d,q}S_{d',q}\mathcal{N}(\mathbf{x}-\mathbf{x}'|\mathbf{0},\mathbf{P}_d^{-1}+\mathbf{P}_{d'}^{-1}+\boldsymbol{\Lambda}_q^{-1}). \tag{12}$$

For a high value of the input dimension, p, the term $1/[(2\pi)^{p/2}|\mathbf{P}_d^{-1}+\mathbf{P}_{d'}^{-1}+\boldsymbol{\Lambda}_q^{-1}|^{1/2}]$ 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 $1/[(2\pi)^{p/4}|2\mathbf{P}_d^{-1}+\boldsymbol{\Lambda}_q^{-1}|^{1/4}]$ 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 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 $\operatorname{cov}[f_d(\mathbf{x}),u_q(\mathbf{x}')]$ in Equation (9), we obtain

$$k_{f_d,u_q}(\mathbf{x},\mathbf{x}') = S_{d,q}\mathcal{N}(\mathbf{x}-\mathbf{x}'|\mathbf{0},\mathbf{P}_d^{-1}+\boldsymbol{\Lambda}_q^{-1}).$$

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

$$p(\mathbf{y}|\mathbf{X},\boldsymbol{\theta}) = \mathcal{N}(\mathbf{y}|\mathbf{0},\mathbf{K}_{\mathbf{f},\mathbf{f}}+\boldsymbol{\Sigma}), \tag{13}$$

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 $\operatorname{cov}[f_d(\mathbf{x}),f_{d'}(\mathbf{x}')]$ in (8). The term $\boldsymbol{\Sigma}$ 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 $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_N\}$ 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

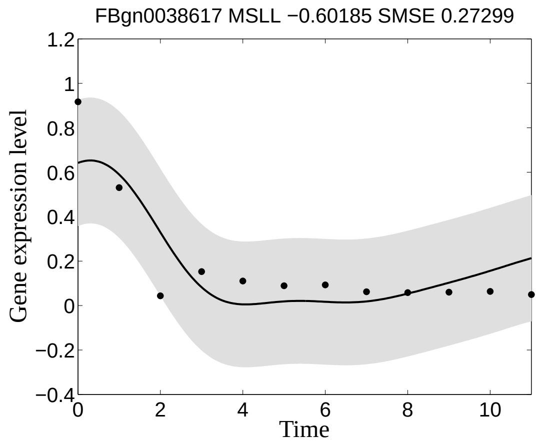
Table 1: Standardized mean square error (SMSE) and mean standardized log loss (MSLL) for the gene expression data for 50 outputs. CMOC stands for convolved multiple output covariance. The experiment was repeated ten times with a different set of 50 genes each time. Table includes the value of one standard 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.

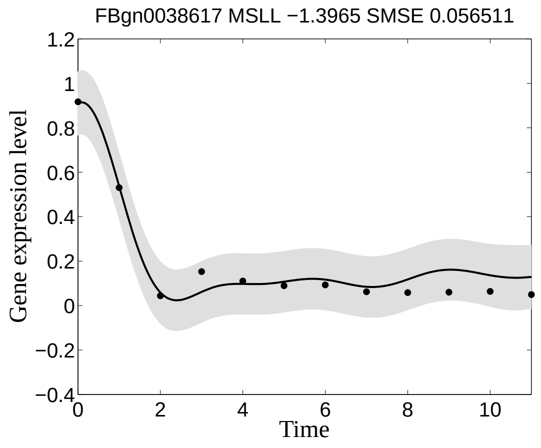
Figure 1 shows the prediction made over the test set (replica 2 in this case) by the two models for two particular genes, namely FBgn0038617 (Figure 1, first row) and FBgn0032216 (Figure 1, second row). The black dots in the figures represent the gene expression data of the particular genes. Figures 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.

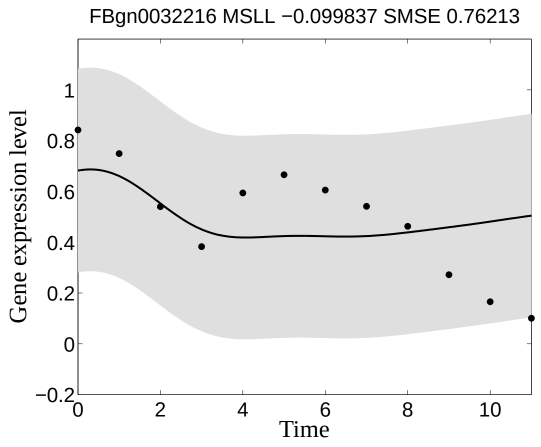
A similar analysis can be made for Figures 2(a), 2(b), 2(c) and 2(d). In this case, the test set is replica 1 and we have chosen two different genes, FBgn0010531 and FBgn0004907 with a similar behavior. Table 2 (last four rows) also highlights the performances of both models for the genes of Figure 2. Again, CMOC outperforms the linear model of coregionalization for both genes and in terms of SMSE and MSLL.



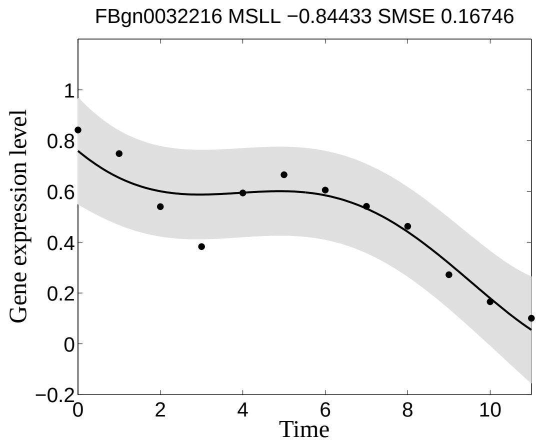
(a) LMC for a short length-scale output



(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.

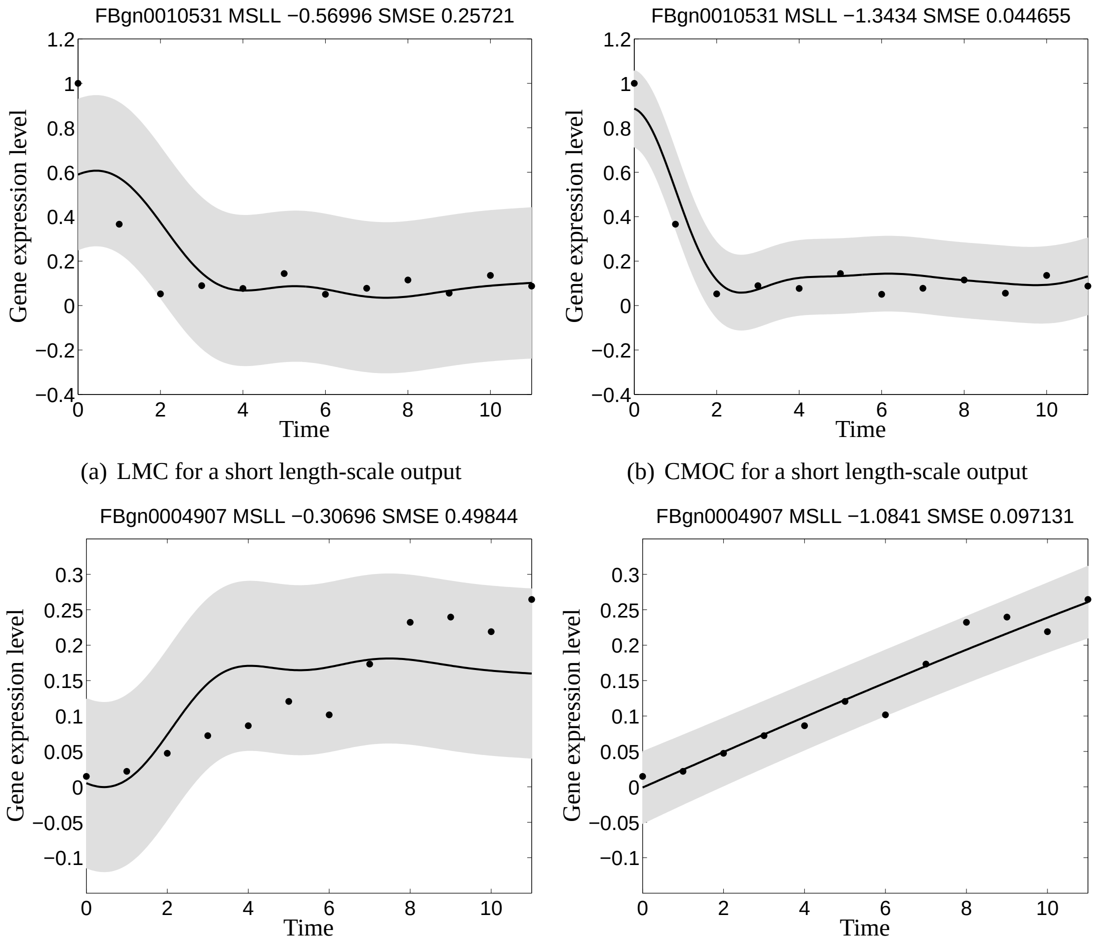
In practice though, we will see in the experimental section, that both the linear model of coregionalization and the convolved multiple output GPs can perform equally well in some data sets. However, the convolved covariance could offer an explanation of the data through a simpler model or converge to 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 $u(\mathbf{z})$, $\mathbf{u}(\mathbf{Z}) = [u(\mathbf{z}_1), \dots, u(\mathbf{z}_K)]^\top$, 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 $\operatorname{cov}\left[f_d(\mathbf{x}), u(\mathbf{z})\right]$ 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

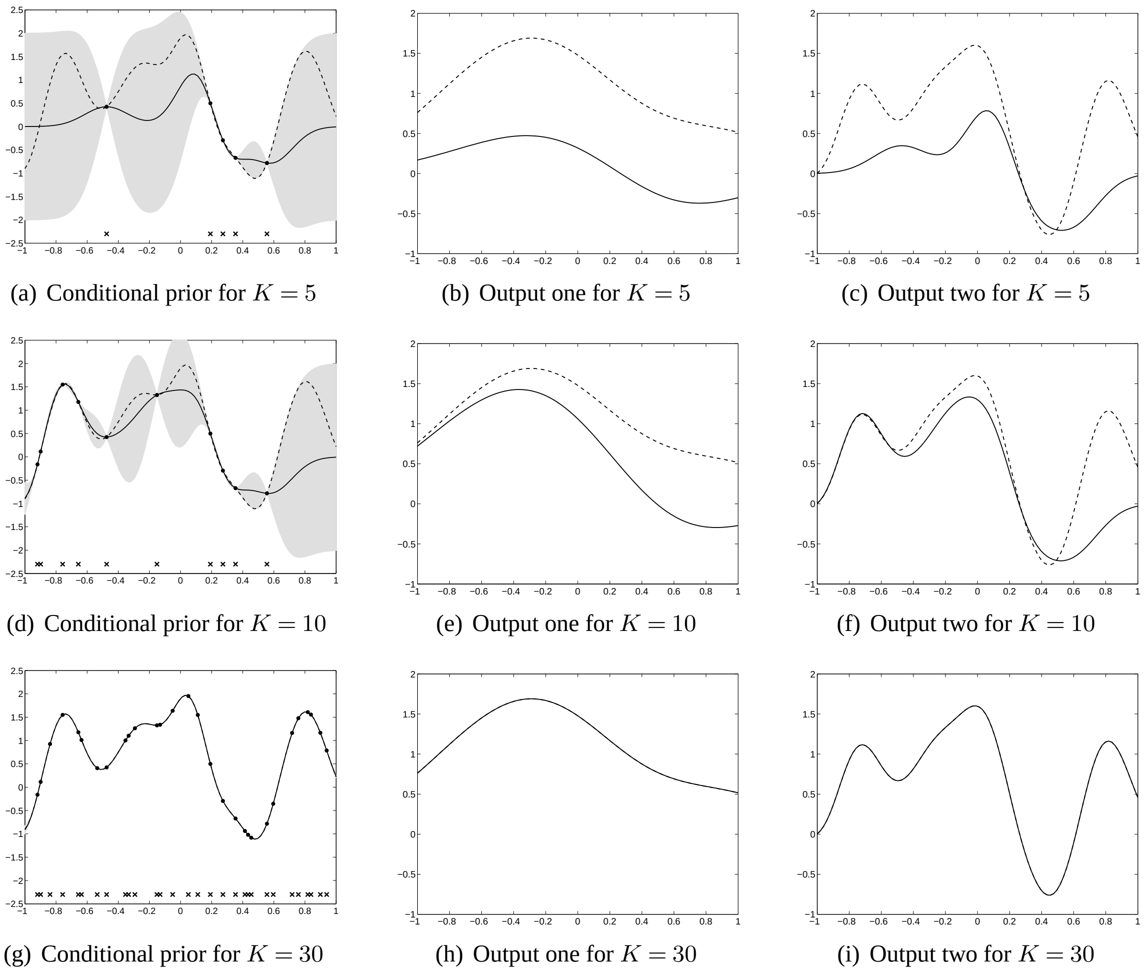
$$p(\mathbf{f}|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \prod_{d=1}^{D} p(\mathbf{f}_d|\mathbf{u},\mathbf{Z},\mathbf{X},\boldsymbol{\theta}) = \prod_{d=1}^{D} \mathcal{N}\left(\mathbf{f}|\mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{\mathbf{f}_d,\mathbf{f}_d}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

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 $\mathbf{D} = \operatorname{blockdiag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right]$, and we have used the notation $\operatorname{blockdiag}\left[\mathbf{G}\right]$ 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 $\mathbf{D} = \operatorname{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right] = \left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}^{\top}\right] \odot \mathbf{M}$, 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, 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 $\bar{\mathbf{f}}_n = [f_1(\mathbf{x}_n), f_2(\mathbf{x}_n), \dots, f_D(\mathbf{x}_n)]^{\top}$. 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 $\mathbf{D}_{d,d'} = \operatorname{diag}\left[\mathbf{K}_{\mathbf{f}_d,\mathbf{f}_{d'}} - \mathbf{K}_{\mathbf{f}_d,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}_{d'}}\right]$. 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 $\mathbf{D} = \operatorname{blockdiag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right]$; for full independence it is $\mathbf{D} = \operatorname{diag}\left[\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right]$ 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 $[-1, 1]$.

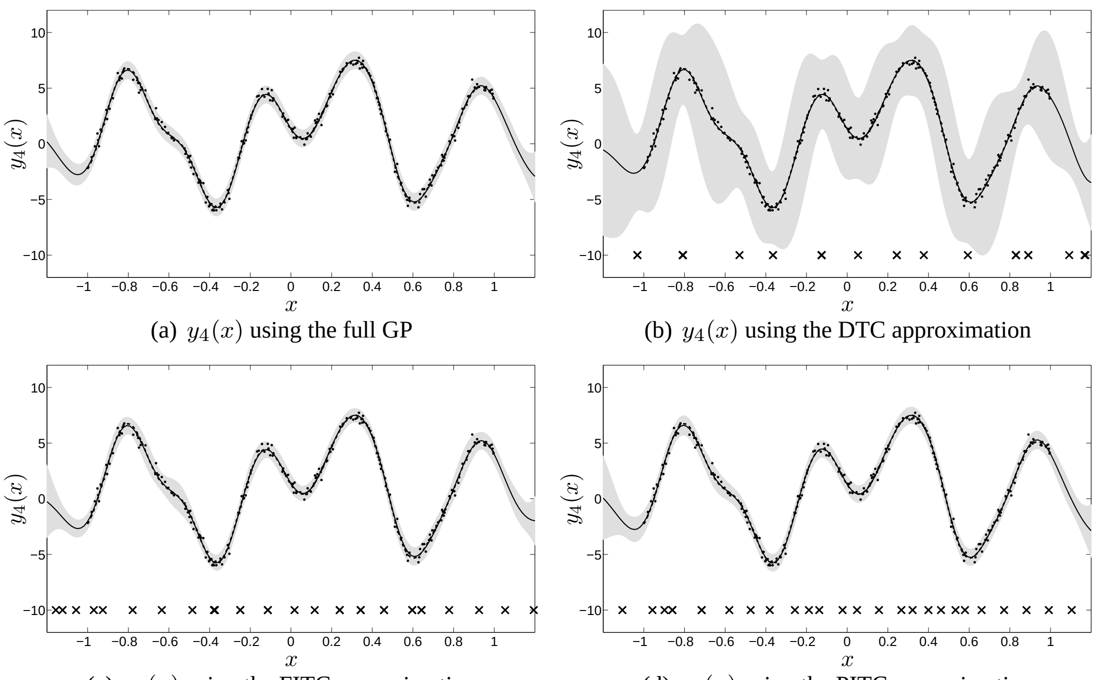
Figure 4 shows the training result of one of the ten repetitions. The predictions shown correspond to the full GP in Figure 4(a), the DTC approximation in Figure 4(b), the FITC approximation in Figure 4(c) and the PITC approximation in Figure 4(d).

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.

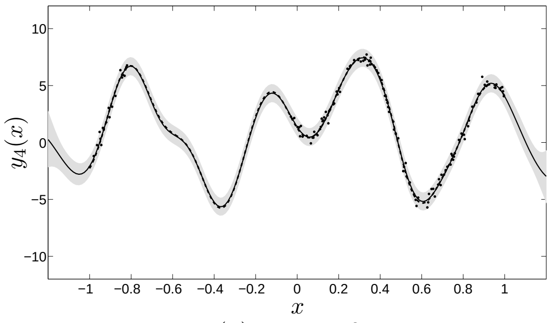
Due to the strong dependencies between the signals, our model is able to capture the correlations and predicts accurately the missing information.

6.2 Exam Score Prediction

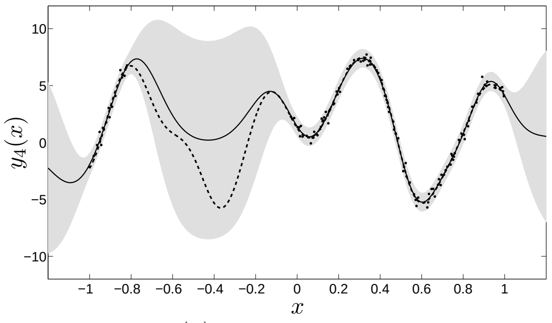
In the first experiment with real data that we consider, the goal is to predict the exam score obtained by a particular student belonging to a particular school. The data comes from the Inner London Education Authority (ILEA).[11] It consists of examination records from 139 secondary schools in years 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 **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

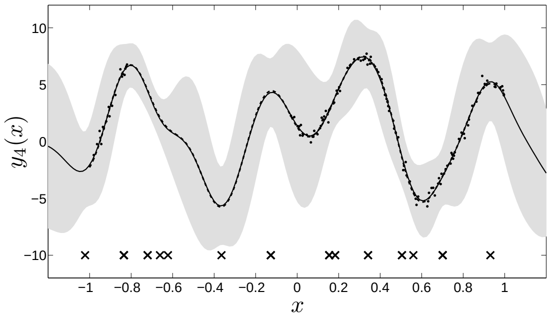
11. This data is available at `http://www.cmm.bristol.ac.uk/learning-training/multilevel-m-support/datasets.shtml`.
12. Performance in the verbal reasoning test was divided in three bands. Band 1 corresponds to the highest 25%, band 2 corresponds to the next 50% and band 3 the bottom 25% (Nuttall et 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).



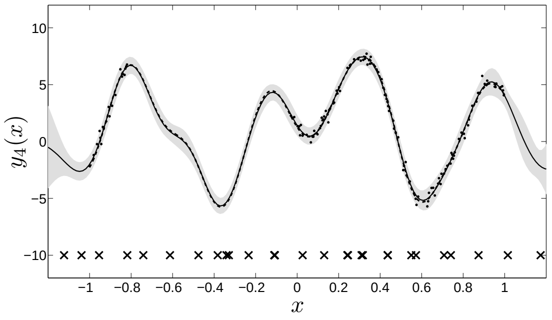
(a) $y_4(x)$ using the full GP



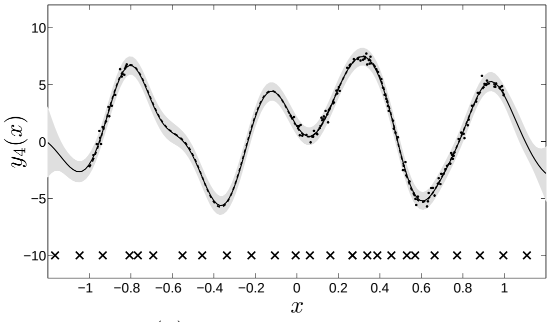
(b) $y_4(x)$ using an independent GP



(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

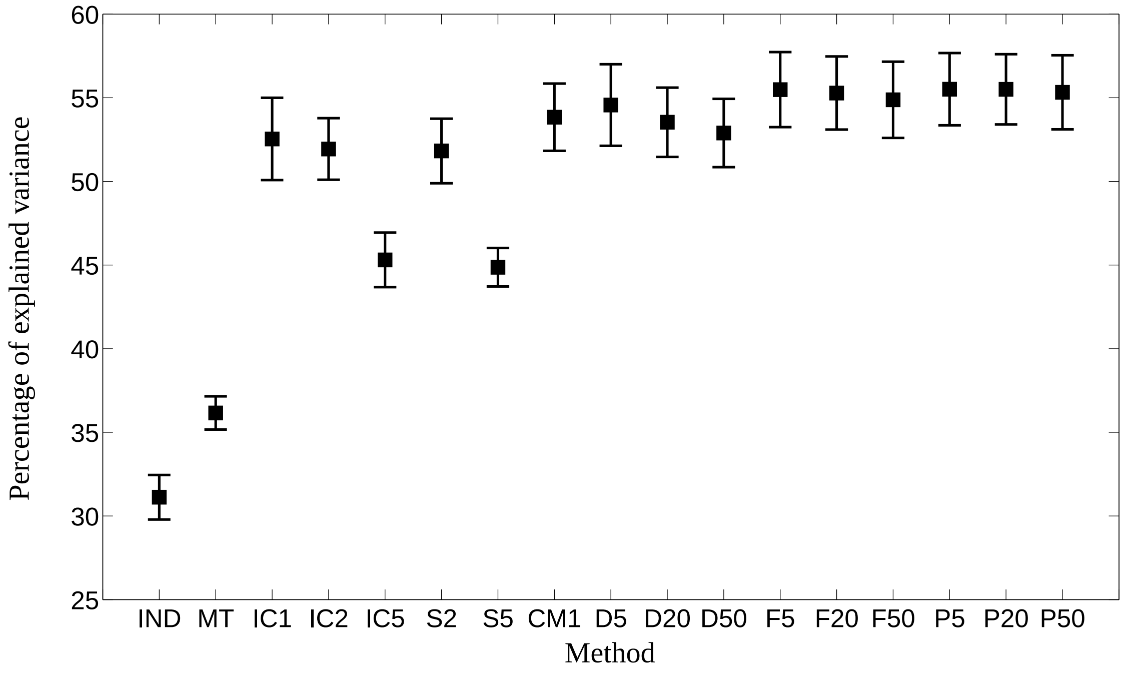


Figure 6: Mean and standard deviation of the percentage of explained variance for exam score prediction results on the ILEA data set. The experiment was repeated ten times. In the bottom of the figure, IND stands for independent GPs, MT stands for multi-task GPs, ICR_1 stands for intrinsic coregionalization 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/`.

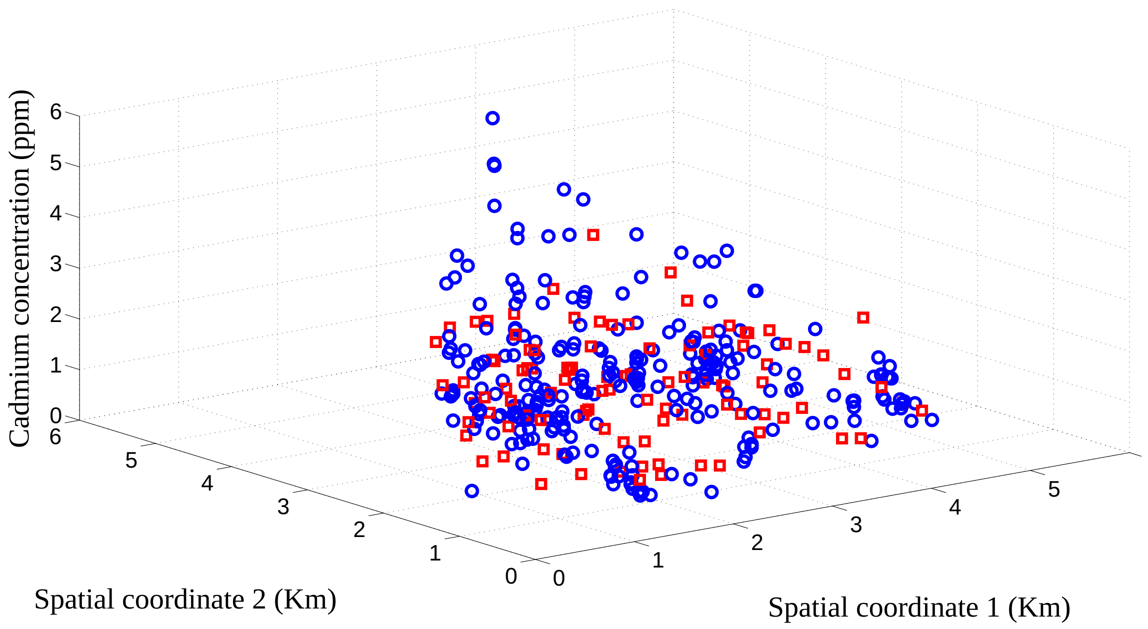


Figure 7: Cadmium concentration for the Swiss Jura example. The blue circles refer to the prediction set (training data for cadmium) and the red squares are the concentrations for the validation set (testing data for cadmium).

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.

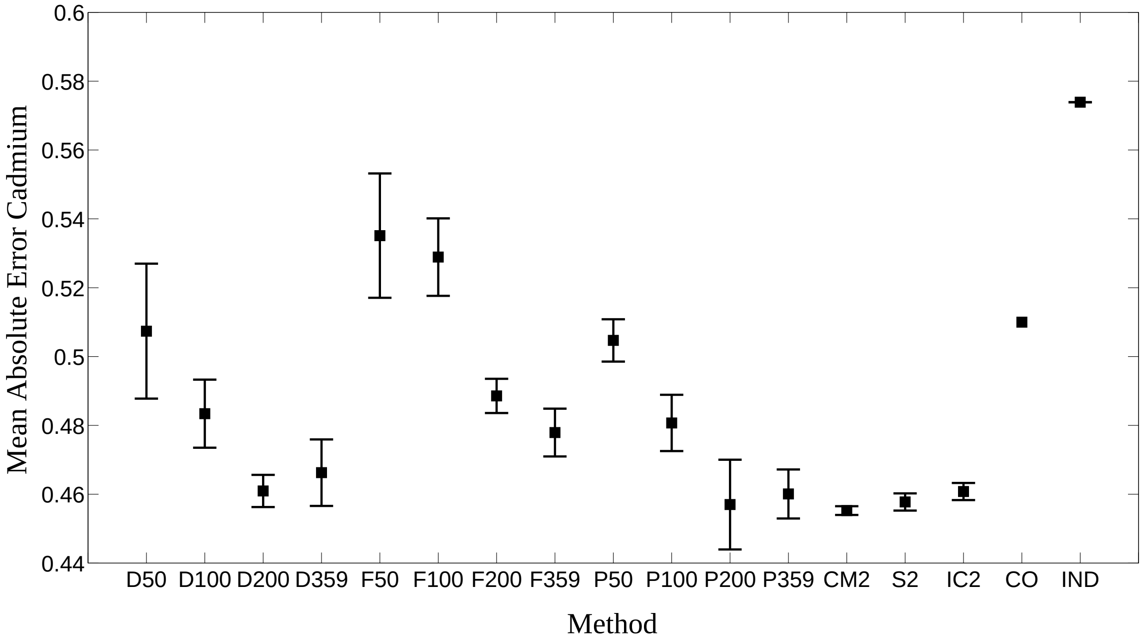


Figure 8: Average mean absolute error and standard deviation for prediction of the pollutant metal cadmium. The experiment was repeated ten times. In the bottom of the figure DK, FK, PK stands for DTC, FITC and PITC with K inducing values, CM2 stands for convolved multiple output covariance with $Q = 2$ 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.

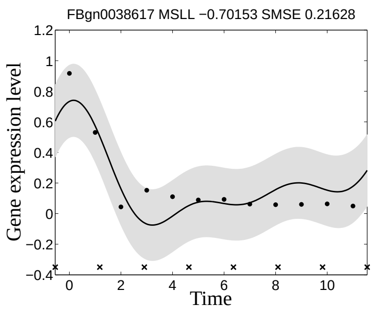
Train set	Test set	Method	Average SMSE	Average MSLL	Average TTPI
Replica 1	Replica 2	DTC	0.5421 ± 0.0085	-0.2493 ± 0.0183	2.04
		FITC	0.5469 ± 0.0125	-0.3124 ± 0.0200	2.31
		PITC	0.5537 ± 0.0136	-0.3162 ± 0.0206	2.59
Replica 2	Replica 1	DTC	0.5454 ± 0.0173	0.6499 ± 0.7961	2.10
		FITC	0.5565 ± 0.0425	-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.

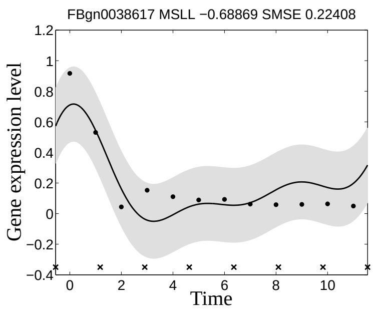
Table 9 shows the results of applying the approximations in terms of SMSE and MSLL (columns 4 and 5). DTC and FITC slightly outperforms PITC in terms of SMSE, but PITC outperforms both DTC and FITC in terms of MSLL. This pattern repeats itself when the training data comes from replica 1 or from replica 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.

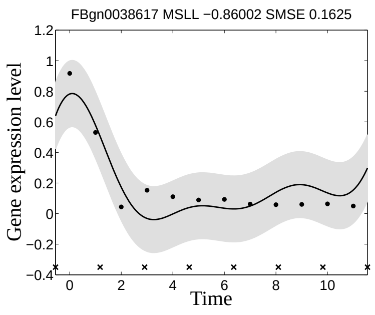
In Figure 10, we replicate the same exercise for the genes FBgn0010531 and FBgn0004907, that also appeared in Figure 2. Performances for DTC, FITC and PITC are shown in Table 10 (last six rows), which compare favourably with the performances for the linear model of coregionalization in Table 2 and close to the performances for the CMOC. In average, PITC outperforms the other methods for the specific set of genes in both figures above.



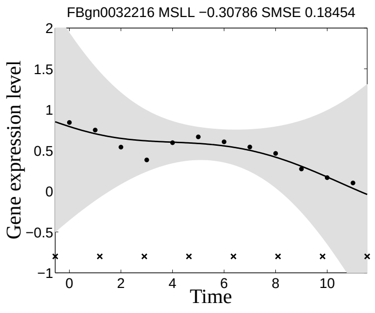
(a) DTC, short length scale



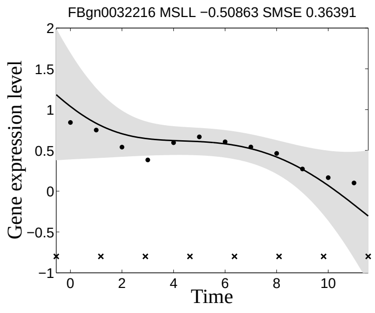
(b) FITC, short length scale



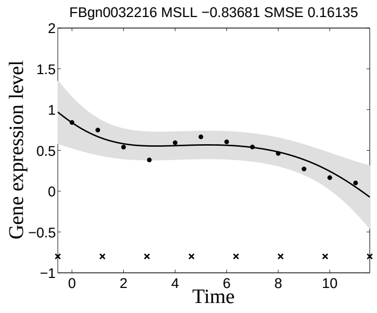
(c) PITC, short length scale



(d) DTC, long length scale



(e) FITC, long length scale



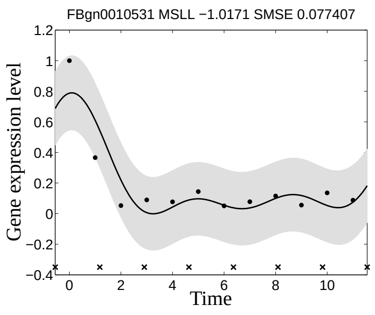
(f) PITC, long length scale

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.

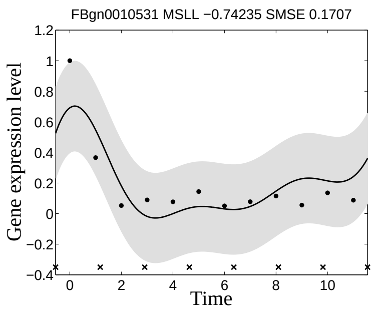
With respect to the training times, the Table 9 in the column 6 shows the average training time per iteration (average TTPI) for each approximation. To have an idea of the saving times, one iteration of the full GP model for the same 1000 genes would take around 4595.3 seconds. This gives a speed up factor of 1780, approximately.

7. Conclusions

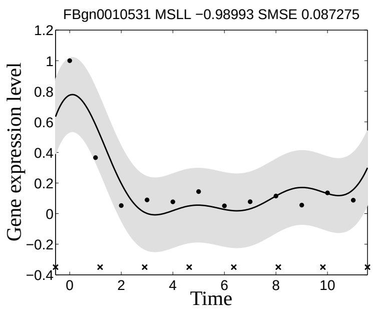
In this paper we first presented a review of different alternatives for multiple output regression grouped under a similar framework known as the linear model of coregionalization. Then we illustrated how the linear model of coregionalization can be interpreted as an instantaneous mixing of latent functions, in contrast to a convolved multiple output framework, where the mixing



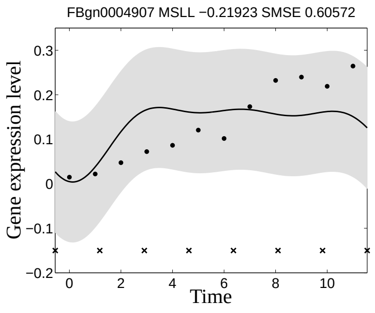
(a) DTC, short length scale



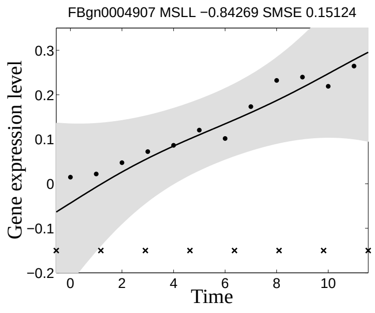
(b) FITC, short length scale



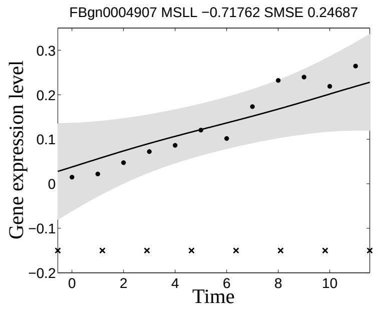
(c) PITC, short length scale



(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 $\mathbf{G:} = \operatorname{vec}\mathbf{G}$, 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 $\frac{\partial\mathcal{L}}{\partial\mathbf{G:}} = \left(\left(\frac{\partial\mathcal{L}}{\partial\mathbf{G}}\right)\mathbf{:}\right)^\top$. The obtain the hyperparameters, we maximize the following log-likelihood function,

$$\mathcal{L}(\mathbf{Z},\boldsymbol{\theta}) \propto -\frac{1}{2}\log|\mathbf{D}+\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}| - \frac{1}{2}\operatorname{trace}\left[\left(\mathbf{D}+\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right)^{-1}\mathbf{y}\mathbf{y}^\top\right] \tag{19}$$

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

$$\begin{aligned}\mathcal{L}(\mathbf{Z},\boldsymbol{\theta}) \propto & \frac{1}{2}\log|\mathbf{K}_{\mathbf{u},\mathbf{u}}| - \frac{1}{2}\log|\mathbf{A}| - \frac{1}{2}\log|\mathbf{D}| - \frac{1}{2}\operatorname{trace}\left[\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\right] \\ & + \frac{1}{2}\operatorname{trace}\left[\mathbf{D}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\right].\end{aligned}$$

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 $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ or $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ and δ_{GK} is zero if $\mathbf{G}$ is equal to $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ and one in other case. Next we present expressions for each partial derivative

$$\frac{\partial \mathcal{L}_{\mathbf{A}}}{\partial \mathbf{A:}} = -\frac{1}{2}(\mathbf{C:})^\top, \quad \frac{\partial \mathbf{A:}}{\partial \mathbf{D:}} = -\left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1} \otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right), \quad \frac{\partial \mathcal{L}_{\mathbf{D}}}{\partial \mathbf{D:}} = -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}\right)\mathbf{:}\right)^\top$$
$$\frac{\partial \mathbf{D:}}{\partial \mathbf{K}_{\mathbf{f},\mathbf{f}}\mathbf{:}} = \operatorname{diag}(\mathbf{M:}), \quad \frac{\partial \mathbf{D:}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{:}} = -\operatorname{diag}(\mathbf{M:})\left[\left(\mathbf{I}\otimes \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right) + \left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\otimes \mathbf{I}\right)\mathbf{T}_{\mathbf{D}}\right],$$
$$\frac{\partial \mathbf{D:}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{:}} = \operatorname{diag}(\mathbf{M:})\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\otimes \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right), \frac{\partial \mathbf{A:}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{:}} = \left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\otimes \mathbf{I}\right) + \left(\mathbf{I}\otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right)\mathbf{T}_{\mathbf{A}}$$
$$\frac{\partial \mathbf{A:}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{:}} = \mathbf{I}, \quad \frac{\partial \mathcal{L}_{\mathbf{K}_{\mathbf{u},\mathbf{f}}}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{:}} = \left(\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\mathbf{D}^{-1}\right)\mathbf{:}\right)^\top, \quad \frac{\partial \mathcal{L}_{\mathbf{K}_{\mathbf{u},\mathbf{u}}}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{u}}\mathbf{:}} = \frac{1}{2}\left(\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right)\mathbf{:}\right)^\top,$$

where $\mathbf{C} = \mathbf{A}^{-1} + \mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\mathbf{D}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}$, $\mathbf{T}_{\mathbf{D}}$ and $\mathbf{T}_{\mathbf{A}}$ are *vectorized transpose matrices* (see, e.g., Brookes, 2005) and $\mathbf{H} = \mathbf{D} - \mathbf{y}\mathbf{y}^\top + \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top + \left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\right)^\top$. We can replace the above expressions in (20) to find the corresponding derivatives, so

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{f},\mathbf{f}}\mathbf{:}} = \frac{1}{2}\left[((\mathbf{C})\mathbf{:})^\top\left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right) - \frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}\right)\mathbf{:}\right)^\top\right]\operatorname{diag}(\mathbf{M:}) \tag{21}$$
$$= -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right)\mathbf{:}\right)^\top\operatorname{diag}(\mathbf{M:}) = -\frac{1}{2}\left(\operatorname{diag}(\mathbf{M:})\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right)\mathbf{:}\right)^\top \tag{22}$$
$$= -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\odot \mathbf{M}\right)\mathbf{:}\right)^\top = -\frac{1}{2}(\mathbf{Q:})^\top \tag{23}$$

or simply

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{f},\mathbf{f}}} = -\frac{1}{2}\mathbf{Q},$$

where $\mathbf{J} = \mathbf{H} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}}$ 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

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{:}} &= \frac{1}{2}(\mathbf{Q:})^\top\left[\left(\mathbf{I}\otimes \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right) + \left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\otimes \mathbf{I}\right)\mathbf{T}_{\mathbf{D}}\right] - \frac{1}{2}(\mathbf{C:})^\top \\ &\quad \left[\left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\otimes \mathbf{I}\right) + \left(\mathbf{I}\otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right)\mathbf{T}_{\mathbf{A}}\right] + \left(\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\mathbf{D}^{-1}\right)\mathbf{:}\right)^\top \\ &= \left(\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Q} - \mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1} + \mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\mathbf{D}^{-1}\right)\mathbf{:}\right)^\top\end{aligned} \tag{24}$$

or simply

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{f}}} = \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Q} - \mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1} + \mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{y}\mathbf{y}^\top\mathbf{D}^{-1},$$

where in (24), $(\mathbf{Q:})^\top(\mathbf{F}\otimes \mathbf{I})\mathbf{T}_{\mathbf{D}} = (\mathbf{Q:})^\top\mathbf{T}_{\mathbf{D}}(\mathbf{I}\otimes \mathbf{F}) = \left(\mathbf{T}_{\mathbf{D}}^\top\mathbf{Q:}\right)^\top(\mathbf{I}\otimes \mathbf{F}) = (\mathbf{Q:})^\top(\mathbf{I}\otimes \mathbf{F})$. A similar analysis is formulated for the term involving $\mathbf{T}_{\mathbf{A}}$. Finally, results for $\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{f}}}$ and $\frac{\partial \mathcal{L}}{\partial \mathbf{\Sigma}}$ are obtained as

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K}_{\mathbf{u},\mathbf{u}}} = -\frac{1}{2}\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} - \mathbf{C} - \mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Q}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right), \quad \frac{\partial \mathcal{L}}{\partial \mathbf{\Sigma}} = -\frac{1}{2}\mathbf{Q}.$$

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.