Learning for Larger Datasets with the Gaussian Process Latent Variable Model

Neil D. Lawrence
School of Computer Science,
University of Manchester, Kilburn Building,
Oxford Road, Manchester, M13 9PL, U.K.

Abstract

In this paper we apply the latest techniques in sparse Gaussian process regression (GPR) to the Gaussian process latent variable model (GP-LVM). We review three techniques and discuss how they may be implemented in the context of the GP-LVM. Each approach is then implemented on a well known benchmark data set and compared with earlier attempts to sparsify the model.

1 Introduction

The Gaussian process latent variable model (GP-LVM) [Lawrence, 2004, 2005] is a flexible approach to probabilistic modelling in high dimensional spaces. It has been succesfully applied in a range of application domains including graphics [Grochow et al., 2004] and visual tracking [Urtasun et al., 2005]. A major advantage of the approach is its ability to effectively model probabilistically data of high dimensionality, however a major weakness with the approach is that computation of gradients and the likelihood are cubic in the number of data points. In the original GP-LVM paper the informative vector machine algorithm (IVM) [Lawrence et al., 2003] was used for obtaining a sparse representation. However, in the context of the GP-LVM, this approach suffers from several weakness. In this paper we show how recent developments in sparse Gaussian process regression [Snelson and Ghahramani, 2006, Quiñonero Candela and Rasmussen, 2005, Seeger et al., 2003] can be adapted to work with the GP-LVM.

2 GP-LVM

The Gaussian process latent variable model [Lawrence, 2004, 2005] is a flexible, non-linear dimensionality reduction technique which also provides a probabilistic representation of a data set. Given a data set $\mathbf{Y} \in \Re^{N\times d}$ containing N data points and d dimensions we seek a q-dimensional representation of the data given by $\mathbf{X} \in \Re^{N\times q}$. A standard probabilistic approach taken to this problem is to first define a mapping between $\mathbf{X}$ and $\mathbf{Y}$,

$$y_{nj} = f\left(\mathbf{x}_n, \mathbf{w}_j\right) + \epsilon_{nj},$$

where ϵ_n is a noise term, y_{nj} is the element from the nth row and jth column of $\mathbf{Y}$, $\mathbf{x}_n$ is a column vector taken from the nth row of $\mathbf{X}$ and the parameters of the mapping are given by the vectors $\{\mathbf{w}_j\}_{j=1}^{d}$. If the noise is drawn independently from a Gaussian distribution we can write down the condtional for $\mathbf{y}_n$ given $\mathbf{x}_n$ as,

$$p\left(\mathbf{y}_n|\mathbf{x}_n, \mathbf{W}\right) = \prod_{j=1}^{d} N\left(y_{nj}|f\left(\mathbf{x}_n, \mathbf{w}_j\right), \beta^{-1}\right),$$

where $\mathbf{W} = \left[\mathbf{w}_1 \ldots \mathbf{w}_d\right]^{\mathrm{T}}$ and we have introduced β for the precision (inverse variance) of the noise. The distribution of the full matrix $\mathbf{Y}$ given $\mathbf{X}$ and $\mathbf{W}$ is then

$$p\left(\mathbf{Y}|\mathbf{X}, \mathbf{W}\right) = \prod_{n=1}^{N}\prod_{j=1}^{d} N\left(y_{nj}|f\left(\mathbf{x}_n, \mathbf{w}_j\right), \beta^{-1}\right).$$

In most traditional approaches to this problem [Tipping and Bishop, 1999, MacKay, 1995, Bishop et al., 1998] the next step is to treat $\mathbf{X}$ as latent variables by selection of an appropriate prior distribution, $p\left(\mathbf{X}\right)$, and marginalisation. The model is then optimised by maximising the marginal likelihood $p\left(\mathbf{Y}|\mathbf{W}\right)$. A key innovation in the GP-LVM is to, instead, place a prior distribution over the mappings, $p\left(\mathbf{W}\right)$ and maximise the marginal likelihood with respect to the latent positions, $p\left(\mathbf{Y}|\mathbf{X}\right)$. If the mappings are linear,

$$f\left(\mathbf{x}_n, \mathbf{w}_j\right) = \mathbf{x}_n^{\mathrm{T}}\mathbf{w}_j,$$

and a Gaussian prior over $\mathbf{w}_j$ is used, the model is equivalent to principal component analysis. However by considering a process prior directly on the function $f\left(\cdot\right)$ we can obtain non-linear mappings. An appropriate, and tractable, process prior is a Gaussian Process (GP) [Rasmussen and

Williams, 2006]. If the GP prior over each of the d functions is the same we obtain the following likelihood,

$$p(\mathbf{Y}|\mathbf{X},\boldsymbol{\theta}) = \prod_{j=1}^{d} N\left(\mathbf{y}_{(j)}|\mathbf{0},\mathbf{K}\right)$$

where $\mathbf{y}_{(j)}$ is the jth column of $\mathbf{Y}$ and $\mathbf{K} \in \Re^{N\times N}$ is the covariance function or kernel of the Gaussian process which we assume is additionally parameterised by $\boldsymbol{\theta}$.

3 Learning in GP-LVMs

Learning in the GP-LVM consists of maximising the likelihood with respect to the positions of the latent variables, $\mathbf{X}$, and the parameters of the kernel $\boldsymbol{\theta}$. This leads to the following log-likelihood,

$$\begin{aligned} L(\mathbf{X},\boldsymbol{\theta}) &= -\frac{dN}{2}\log 2\pi - \frac{d}{2}\log|\mathbf{K}| \\ &\quad -\frac{1}{2}\mathrm{tr}\left(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right). \end{aligned}$$

Gradients of $L(\mathbf{X},\boldsymbol{\theta})$ are then easily obtained through combining gradients of $\frac{\partial L(\mathbf{X},\boldsymbol{\theta})}{\partial \mathbf{K}}$ with gradients given by $\frac{\partial \mathbf{K}}{\partial \mathbf{X}}$ and $\frac{\partial \mathbf{K}}{\partial \boldsymbol{\theta}}$. In general[1], it is not possible to obtain a fixed point solution for $\mathbf{X}$ and $\boldsymbol{\theta}$, to make progress we must turn to gradient based iterative optimisation of the log-likelihood. Such algorithms rely on multiple re-evaluations of the log-likelihood and its gradients. Each evaluation has $O\left(N^3\right)$ complexity due to the inverse of $\mathbf{K}$. In Lawrence [2004, 2005] a sparse approximation based on active set selection through the informative vector machine (IVM) was proposed. However, there are two key problems with this approach. Firstly, the resulting posterior distribution over mappings depends only on the active set. Secondly, the optimal active set changes as the optimisation proceeds. The active set must, therefore, be re-selected regularly causing fluctuations in the objective function. It can therefore be diffcult to determine when convergence has occured. In this paper we show how the latest methods for sparse Gaussian process regression can be used with the GP-LVM to reduce the complexity of gradient and likelihood evaluations to $O\left(k^2N\right)$ while retaining a convergent algorithm in which the posterior distributions over the mappings depend on the entire data set.

4 Sparse Approximations

By exploiting a sparse approximation to the full Gaussian process it is usually possible to reduce the computational complexity from an often prohibitive $O\left(N^3\right)$ to a more manageable $O\left(k^2N\right)$, where k is the number of points retained in the sparse representation. A large body of recent work has been focussed on approximating the covariance function with a low rank approximation [Smola and Bartlett, 2001, Williams and Seeger, 2001, Tresp, 2000, Schwaighofer and Tresp, 2003, Csató and Opper, 2002, Seeger et al., 2003]. Recently several of these approaches were unified by Quiñonero Candela and Rasmussen [2005]. The advantage of the unified viewpoint is that we can discuss approximations within the same context. Quiñonero Candela and Rasmussen [2005] also provide a consistent terminology for these approximations (which we will refer to as QR-terminology) that we will make use of in this paper. The approximations all involve augmenting the function values at the training points, $\mathbf{F} \in \Re^{N\times d}$, and the function values at the test points, $\mathbf{F}_* \in \Re^{\infty\times d}$, by an additional set of variables, $\mathbf{U} \in \Re^{k\times d}$. The number of these variables, k, can be specified by the user. The augmenting variables have been variously called the 'active points', 'pseudo-inputs' or 'support points'; in QR-terminology they are known as the inducing variables.

The factorisation of the likelihood across the columns[2] of $\mathbf{Y}$ allows us to focus on one column of $\mathbf{F}$ without loss of generality. We therefore consider function values at $\mathbf{f} \in \Re^{N\times 1}$, $\mathbf{f}_* \in \Re^{\infty\times 1}$ and $\mathbf{u} \in \Re^{k\times 1}$. These variables are considered to be jointly Gaussian distributed with $\mathbf{f}$ and $\mathbf{f}_*$ such that

$$p(\mathbf{f},\mathbf{f}_*) = \int p(\mathbf{f},\mathbf{f}_*|\mathbf{u})\,p(\mathbf{u})\,d\mathbf{u},$$

where the prior distribution over the inducing variables is given by a Gaussian process,

$$p(\mathbf{u}) = N(\mathbf{u}|\mathbf{0},\mathbf{K}_{\mathbf{u},\mathbf{u}}),$$

with a covariance function given by $\mathbf{K}_{\mathbf{u},\mathbf{u}}$. This covariance is constructed on a set of inputs[3] $\mathbf{X}_{\mathbf{u}}$ which may or may not be a subset of $\mathbf{X}$. For the full Gaussian process the presence or absence of these inducing variables is irrelevant, however through their introduction we can motivate most of the sparse approximations listed above. The key concept in unifying the different approximations [Quiñonero Candela and Rasmussen, 2005] is to consider that the variables associated with the training data, $\mathbf{f}$, are conditionally independent of those associated with the test data, $\mathbf{f}_*$, given the inducing variables, $\mathbf{u}$:

$$p(\mathbf{f},\mathbf{f}_*,\mathbf{u}) = p(\mathbf{f}|\mathbf{u})\,p(\mathbf{f}_*|\mathbf{u})\,p(\mathbf{u}),$$

where

$$p(\mathbf{f}|\mathbf{u}) = N\left(\mathbf{f}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right) \qquad (1)$$

[1]An exception is when $\mathbf{K} = \mathbf{X}\mathbf{X}^{\mathrm{T}} + \beta^{-1}\mathbf{I}$, *i.e.* when the process constrains the model to linear functions. In this case the principal component analysis solution is recovered.

[2]Much of our analysis may hold even if a factorisation assumption isn't made, but it simplifies the exposition if we constrain ourselves to the factorising case.

[3]There is nothing to prevent us from allowing a different set of inducing variables, $\mathbf{X}_{\mathbf{u}}^{(i)}$, for each of the d dimensions of the data, but we shall consider only one set for all data dimensions to keep the derivations simple.

is the training conditional in QR-terminology and

$$p(\mathbf{f}_*|\mathbf{u}) = N\left(\mathbf{f}_*|\mathbf{K}_{*,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},*}\right)$$

is the test conditional. $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ is the covariance function computed between the training inputs, $\mathbf{X}$, and the inducing variables, $\mathbf{X}_{\mathbf{u}}$, $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ is the symmetric covariance between the training inputs, $\mathbf{K}_{*,\mathbf{u}}$ is the covariance function between the test inputs and the inducing variables and $\mathbf{K}_{*,*}$ is the symmetric covariance function the test inputs. This decomposition does not in itself entail any approximations: the approximations are introduced through assumptions about the form of these distributions.

4.1 Deterministic Training Conditional

The first approximation we consider in the context of the GP-LVM is known as the 'deterministic training conditional' (DTC) approximation in QR-terminology. It is so called because it involves replacing the true training conditional (1) with a deterministic approximation of the form

$$q\left(\mathbf{f}_{(j)}|\mathbf{u}\right) = N\left(\mathbf{f}_{(j)}|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}, \mathbf{0}\right),$$

where we introduced the index on $\mathbf{f}$ to indicate the column of $\mathbf{F}$ from which it comes. This approximation was first proposed by Csató and Opper [2002] in the context of online learning of Gaussian processes and was further used by Seeger et al. [2003]. Re-introducing the prior over the inducing variables we find that the functional prior for this approximation is given by

$$q\left(\mathbf{f}_{(j)}\right) = N\left(\mathbf{f}_{(j)}|\mathbf{0}, \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right).$$

This prior may be combined with the likelihood,

$$p\left(\mathbf{y}_{(j)}|\mathbf{f}_{(j)}, \beta\right) = N\left(\mathbf{y}_{(j)}|\mathbf{f}_{(j)}, \beta^{-1}\mathbf{I}\right)$$

to give a marginal log likelihood of the form

$$\begin{aligned}\log p(\mathbf{Y}|\mathbf{X}_+, \boldsymbol{\theta}) = & -\frac{d}{2}\log(2\pi) \\ & -\frac{d}{2}\log\left|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}} + \beta^{-1}\mathbf{I}\right| \\ & -\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}} + \beta^{-1}\mathbf{I}\right)^{-1}\right) \\ = & \; L(\mathbf{X}_+, \boldsymbol{\theta}),\end{aligned}$$

where we have used $\mathbf{X}_+ = \{\mathbf{X}_{\mathbf{u}}, \mathbf{X}\}$ to represent the set of training inputs augmented by the inducing inputs. Gradients with respect to the $\mathbf{X}_+$ and $\boldsymbol{\theta}$ may all be determined through first seeking gradients with respect to $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ and combining them with gradients of a given kernel. Details of how these gradients may be arrived at are given in Lawrence [2006]. Having optimised with respect to these parameters predictions can be made for test points using

$$p\left(\mathbf{f}_{*(j)}|\mathbf{y}_{(j)}\right) = N\left(\mathbf{f}_{*(j)}|\bar{\mathbf{f}}_1, \Sigma_1\right),$$

where the mean is given by $\bar{\mathbf{f}}_1 = \mathbf{K}_{*,\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{y}_{(j)}$, the covariance is $\Sigma_1 = \mathbf{K}_{*,*} - \mathbf{K}_{*,\mathbf{u}}\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1} - \beta^{-1}\mathbf{A}^{-1}\right)\mathbf{K}_{\mathbf{u},*}$ and $\mathbf{A} = \left(\beta^{-1}\mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right)$.

4.2 Fully and Partially Indepence

Much of the recent insight into sparse Gaussian process regression was triggered by the work of Snelson and Ghahramani [2006]. Their paper proposed two orthogonal ideas. The first was to optimise with respect to the positions of the inducing variables (or pseudo-inputs as they term them)[4] and the second was a richer form of approximation to the training conditional. In QR-terminology it is refered to as the 'fully independent training conditional' (FITC) approximation and involves an independence assumption for the training conditional. Csató [2005] has pointed out that the Bayesian Committee Machine [Tresp, 2000, Schwaighofer and Tresp, 2003] makes similar *block* diagonal independence assumptions, in QR-terminology this is refered to as the 'partially independent training conditional'. Both approaches can be considered through the following form for the training conditional,

$$q(\mathbf{f}|\mathbf{u}) = N\left(\mathbf{f}_{(j)}|\bar{\mathbf{f}}_2, \Sigma_2\right),$$

where $\bar{\mathbf{f}}_2 = \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}$, $\Sigma_2 = \mathrm{mask}\left(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}, \mathbf{M}\right)$ and the function $\mathbf{V} = \mathrm{mask}(\mathbf{Z}, \mathbf{M})$, with $\mathbf{M}$ a matrix of ones and zeros, returns a matrix $\mathbf{V}$ of dimension matching that of $\mathbf{Z}$ with elements $v_{ij} = z_{ij}$ iff $m_{ij} = 1$ and zero otherwise. In the case that $\mathbf{M} = \mathbf{I}$ the training conditional has a *diagonal* covariance as is specified for the FITC approxixmation and if $\mathbf{M}$ is block diagonal the training conditional also has a *block diagonal* covariance structure as is specified for the PITC approximation.

Again we can reintroduce the prior over the inducing variables to recover a marginal distribution of the form

$$q(\mathbf{f}) = N(\mathbf{f}|\mathbf{0}, \Sigma_3),$$

where $\Sigma_3 = \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}} + \mathrm{mask}\left(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}, \mathbf{M}\right)$. This distribution may again be combined with $p(\mathbf{Y}|\mathbf{F}, \beta)$ to obtain the log-likelihood

$$\begin{aligned}\log p(\mathbf{Y}|\mathbf{X}_+, \boldsymbol{\theta}) = & -\frac{d}{2}\log(2\pi) \\ & -\frac{d}{2}\log\left|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}} + \mathbf{D}\right|\end{aligned}$$

[4] This idea is reminicent of adaptive basis models, such as multi-layer perceptrons and radial basis functions (see Bishop, 1995). It is indeed possible to represent such networks as Gaussian processes and to adapt the location of the basis by treating them as hyper parameters: however the resulting models would correspond to the Subset of Regressors approximation [Poggio and Girosi, 1990, Luo and Wahba, 1997, Williams et al., 2002, Quiñonero Candela and Rasmussen, 2005] rather than the more advanced approximations we are reviewing here.

$$
\begin{aligned}
& -\frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}+\mathbf{D}\right)^{-1}\right) \\
= \; & L\left(\mathbf{X}_{+},\boldsymbol{\theta}\right).
\end{aligned}
$$

where $\mathbf{D} = \mathrm{mask}\left(\beta^{-1}+\mathbf{K}_{\mathbf{f},\mathbf{f}}-\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}},\mathbf{M}\right)$. Derivatives of the likelihood with respect to $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ are given in Appendix C.

4.3 Application in the GP-LVM

As we mentioned in the previous section, Snelson and Ghahramani [2006] not only suggested the use of the FITC approximation but they also suggested optimisation of the inducing inputs jointly with the parameters of the covariance function. There are several reasons why this is an attractive idea, but in the context of the GP-LVM, perhaps foremost amoungst them is the fact that by jointly optimising over $\mathbf{X}$, $\mathbf{X}_{\mathbf{u}}$ and $\boldsymbol{\theta}$ convergence can be monitored in a straightforward manner. If, rather than optimising with respect to them, the inducing variables are being chosen as a sub-set of $\mathbf{X}$ then the likelihood fluctuates as they are reselected. This problem manifests itself in the IVM-based sparsification of the GP-LVM [Lawrence, 2005]. In this paper we will aim to improve performance through following Snelson and Ghahramani's suggestion of optimising the inducing variable inputs in the context of the approximations outlined above.

4.3.1 Multiple Output Regressions

Snelson and Ghahramani focus on Gaussian process regression with a single target vector (in the context of the GP-LVM, $d = 1$). They do not address the issue of whether to represent the inducing variables separately for each column of $\mathbf{Y}$ target or to share the same inducing variables across columns of $\mathbf{Y}$. Arguments could be made in favour of either approach, however allowing different sets of $\mathbf{X}_{\mathbf{u}}$ for each data dimension will cause the number of inducing variables to scale with d. For high dimensional data sets this could make the optimisation prohibitive. Furthermore, Snelson and Ghahramani have some concerns about overfitting when k is large: normally overfitting isn't a concern for Gaussian processes, but by using inducing variables we are introducing a large number of parameters into the system. The possibility of overfitting would be compounded by allowing different active sets for each column of d.

5 Experiments

To evaluate the new sparse GP approximations in the GP-LVM we considered two datasets, a benchmark data set visualising oil flow in a pipeline and a set of human motion capture data of a subject walking and running.

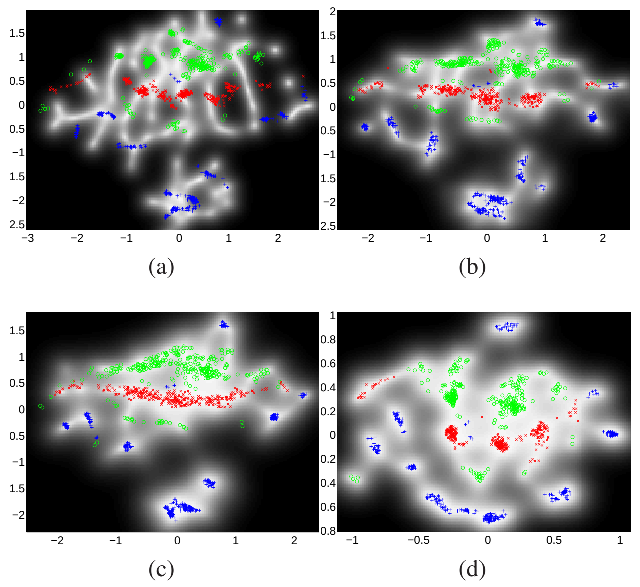

Figure 1: The oil data visualised using a GP-LVM with (a) the DTC approximation, (b) the FITC approximation, (c) the PITC approximation and (d) no approximation. Crosses (red), circules (green) circles and plus signs (blue) represent stratified, annular and homogenous respectively. The greyscale background to the plots visualises the precision with which the posterior process is mapped in the data space, for larger plots see Lawrence [2006].

5.1 Oil Data

In this section we present results on three phase oil flow data [Bishop and James, 1993] . The data consists of twelve dimensional measurements of oil flow within a pipeline. There are three phases of flow associated with the data: stratified, annular and homegenous and 1000 points in the data set. We compared the sparse GP-LVM using the different approximations with an IVM based approximation and a full GP-LVM trained on the data. In each case we projected the data onto two dimensions. We show the resulting projections for the new sparse approaches in Figure 1 as well as the result for the full GP-LVM. For the result from the IVM see Lawrence [2005]. As well as the locations of the data in the latent space we also show the uncertainty associated with the Gaussian processes as a function of the latent space. This is shown as a greyscale image with white representing low variance (or high precision) and black representing high variance (low precision).

There is a noticable difference in the pattern associated with the uncertainty for each plot. In particular for the DTC approximation (Figure 1 (a)) the variances are low along spray-paint-like streaks across the latent space. This effect becomes less pronounced with the FITC approximation (Figure 1 (b)) and is almost non-existent with the PITC approximation (Figure 1 (c)). In each case 100 inducing points were used to create these visualisations.

	DTC	FITC	PITC	IVM	FULL	IN Y
ERRORS	3	6	6	24	1	2

Table 1: Nearest neighbour errors in latent space for the oil data. For reference we include the result obtained when nearest neighbour classification is undertaken in the original high-dimensional data space (labelled 'In Y').

The quality of the models can be objectively assesed through computing the nearest neighbour classification error in the latent space. The results from doing so are shown in Table 1. Also included is a result obtained by using a sparse algorithm based on the IVM approximation from Lawrence [2005]. Each of the presented sparse algorithms strongly outperforms the IVM-based sparsification while not quite reaching the performance of the full algorithm.

5.2 Motion Capture

We followed Taylor et al. [2007] in considering a human motion capture data set. The data consisted of walking and running motions from subject 35 in the CMU Mocap data base. Motion numbers 1-17, 19-26, 28 and 30-34 were concatanated to form a training set. Each motion represents a separate sequence, we therefore made use of the dynamical refinement of the GP-LVM proposed by Wang et al. [2006] to handle the dynamics. Taylor et al. [2007] mapped the angles to be between -180 and 180, rescaled the data, removed dimensions which weren't varying and mapped the root node's rotations onto circles using sine and cosine. They created a data set for their model by forming each data point by concatanating two neighbouring frames. They applied their binary latent variable model to two missing data problems: the first where the right leg was removed from the test seqence and the second where the upper body was removed. The reconstruction obtained was compared to that given by nearest neighbour. We used a slightly modified presentation of the data set. We did not concatanate frames (as we wish to assess the ability of the dynamical model to represent temporal relationships in their entireity) and we removed the absolute value of the root node for position in the co-ordinates x and z and rotation around the y-axis, modelling instead differences between frames[5].

In learning we used the FITC approximation with 100 inducing points. However, rather than allowing these points to be moved freely, they were fixed to latent points that were uniformly sub-sampled from the data. The models were back constrained [Lawrence and Quiñonero Candela, 2006] using multi-layer perceptrons with 10 hidden units.

[5]The absolute root position in x and z and absolute rotation around y (the vertical axis) should normally be irrelevant in determining pose, but in this data the subject's path and start position were very similar in each sequence so they, unusually, carry information. We choose to discard this information because it would not normally be available.

DATA	CL	CB	L	L	B	B
ERR. TYP.	SC	SC	SC	RA	SC	RA
BLV	11.7	***8.8***	-	-	-	-
NN (S)	22.2	**20.5**	-	-	-	-
GP-LVM ($q = 3$)	-	-	11.4	3.40	***16.9***	**2.49**
GP-LVM ($q = 4$)	-	-	**9.7**	**3.38**	20.7	2.72
GP-LVM ($q = 5$)	-	-	13.4	4.25	23.4	2.78
NN (S)	-	-	13.5	4.44	20.8	2.62
NN	-	-	14.0	4.11	30.9	3.20

Table 2: Results from the missing data problem. Headings: CL and CB are the leg and body data sets as preprocessed by Taylor et al.. L and B are our pre-processings. The error types are SC, which is Taylor et al.'s cumulative error per joint in the *scaled* space, and RA, which is root mean square error in the angle space. Methods are: BLV: Taylor et al.'s binary latent variable model, NN: nearest neighbour, NN (S): nearest neighbour in scaled space, GP-LVM (latent dimension): the GP-LVM with different latent dimensions, q. Bold results are the best reported for a given column. Italicised results are the best reported in terms of cumulative error per joint across the data set (regardless of pre-processing).

The parameters of the dynamics model were learnt during optimisation[6]. The data set size was 2613 frames.

A summary of the results is given in Table 2. Note that the cumulative scaled error reported by Taylor is not always consistent with the angle error, in particular note that for the GP-LVM with $q = 3$ cumulative error is much higher than with $q = 4$, but the root mean square angle errors are very similar. The results show that the GP-LVM typically outperforms nearest neighbour.

6 Discussion

In this paper we have reviewed sparse approximations for Gaussian process regression from the perspective of a unifying framework proposed by Quiñonero Candela and Rasmussen [2005]. We combined each of the different approximations summarised by Quiñonero Candela and Rasmussen [2005] with the suggestion of Snelson and Ghahramani [2006] to optimise the locations of the inducing variables (described as pseudo inputs by Snelson and Ghahramani). The resulting approximations were combined with the Gaussian process latent variable model and results were then presented on a benchmark visualisation data set. The results show that much better quality results can be achieved with this family of approximations. The DTC approximation gave the best performance in a nearest neighbour experiment, but the quality of the error bars was better for the PITC and FITC approximations.

[6]This is possible due to the back constraints, if no back constraints are used the dynamics effectively learn to switch themselves off.

All the experiments detailed here may be recreated with code available on-line from `http://www.dcs.shef.ac.uk/~neil/fgplvm`.

Acknowledgements

We would like to acknowledge the European FP6 PASCAL Network of Excellence for support for the Gaussian Process Round Table and Pump Priming funding. The motion capture data used in the dynamics example was from the CMU Graphics Lab Motion Capture Database (`http://mocap.cs.cmu.edu`) which was created with funding from NSF EIA-0196217. Thanks to Graham Taylor for access to his code and help recreating the motion capture data set.

References

S. Becker, S. Thrun, and K. Obermayer, editors. *Advances in Neural Information Processing Systems*, volume 15, Cambridge, MA, 2003. MIT Press. 6

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995. ISBN 0-19-853864-2. 3

C. M. Bishop and G. D. James. Analysis of multiphase flows using dual-energy gamma densitometry and neural networks. *Nuclear Instruments and Methods in Physics Research*, A327: 580–593, 1993. 4

C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: the Generative Topographic Mapping. *Neural Computation*, 10 (1):215–234, 1998. 1

M. Brookes. The matrix reference manual. Available on-line., 2005. `http://www.ee.ic.ac.uk/hp/staff/dmb/matrix/intro.html`. 7

L. Csató. Sparsity in Gaussian processes: Questions. Talk at the Sheffield Gaussian Process Round Table., June 2005. Slides available from `http://www.dcs.shef.ac.uk/ml/gprt/slides/lehelcsato.pdf`. 3

L. Csató and M. Opper. Sparse on-line Gaussian processes. *Neural Computation*, 14(3):641–668, 2002. 2, 3

K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popovic. Style-based inverse kinematics. In *ACM Transactions on Graphics (SIGGRAPH 2004)*, 2004. 1

N. D. Lawrence. Gaussian process models for visualisation of high dimensional data. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems*, volume 16, pages 329–336, Cambridge, MA, 2004. MIT Press. 1, 2

N. D. Lawrence. Large scale learning with the Gaussian process latent variable model. Technical Report CS-06-05, University of Sheffield, 2006. 3, 4, 7

N. D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, Nov 2005. 1, 2, 4, 5

N. D. Lawrence and J. Quiñonero Candela. Local distance preservation in the GP-LVM through back constraints. In W. Cohen and A. Moore, editors, *Proceedings of the International Conference in Machine Learning*, volume 23, pages 513–520. Omnipress, 2006. ISBN 1-59593-383-2. 5, 6

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In Becker et al. [2003], pages 625–632. 1

T. K. Leen, T. G. Dietterich, and V. Tresp, editors. *Advances in Neural Information Processing Systems*, volume 13, Cambridge, MA, 2001. MIT Press. 6

Z. Luo and G. Wahba. Hybrid adaptive splines. *Journal of the American Statistical Association*, 92:107–116, 1997. 3

D. J. C. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, A*, 354 (1):73–80, 1995. 1

T. Poggio and F. Girosi. Networks for approximation and learning. *Proceedings of the IEEE*, 78(9):1481–1497, 1990. 3

J. Quiñonero Candela and C. E. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1939–1959, 2005. 1, 2, 3, 5, 7

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, Cambridge, MA, 2006. ISBN 026218253X. 1

A. Schwaighofer and V. Tresp. Transductive and inductive methods for approximate Gaussian process regression. In Becker et al. [2003], pages 953–960. 2, 3

M. Seeger, C. K. I. Williams, and N. D. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In C. M. Bishop and B. J. Frey, editors, *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*, Jan 3–6, Key West, FL, 2003. 1, 2, 3

A. J. Smola and P. L. Bartlett. Sparse greedy Gaussian process regression. In Leen et al. [2001], pages 619–625. 2

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Weiss et al. [2006]. 1, 3, 4, 5, 7

G. W. Taylor, G. E. Hinton, and S. Roweis. Modeling human motion using binary latent variables. In B. Schölkopf, J. C. Platt, and T. Hofmann, editors, *Advances in Neural Information Processing Systems*, volume 19, Cambridge, MA, 2007. MIT Press. 5, 7

M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 6(3): 611–622, 1999. 1

V. Tresp. A Bayesian committee machine. *Neural Computation*, 12(11):2719–2741, 2000. 2, 3

R. Urtasun, D. J. Fleet, A. Hertzmann, and P. Fua. Priors for people tracking from small training sets. In *IEEE International Conference on Computer Vision (ICCV)*, pages 403–410, Bejing, China, 17–21 Oct. 2005. IEEE Computer Society Press. 1

J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models. In Weiss et al. [2006]. 5

Y. Weiss, B. Schölkopf, and J. C. Platt, editors. *Advances in Neural Information Processing Systems*, volume 18, Cambridge, MA, 2006. MIT Press. 6

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In Leen et al. [2001], pages 682–688. 2

C. K. I. Williams, C. E. Rasmussen, A. Schwaighofer, and V. Tresp. Observations of the Nyström method for Gaussian process prediction. Technical report, University of Edinburgh, 2002. 3

A On-line Source Code

Matlab source code for repeating the experiments described in Section 5 is available on-line from `http://www.cs.man.ac.uk/~neill/fgplvm`. The experiments were created using the scripts `demOil3.m` for DTC, `demOil1.m` for FITC and `demOil5.m` for PITC (version 0.13 of the code, `demOil2.m`, `demOil4.m` and `demOil6.m` are results that use back constraints [Lawrence and Quiñonero Candela, 2006] and are not presented here). The full GP-LVM was created using C++ code available from `http://www.cs.man.ac.uk/~neill/gplvmcpp`, see instructions on that site for how to recreate it. The motion capture results were created using the scripts `demCmu35gplvm1.m`, `demCmu35gplvm2.m` and `demCmu35gplvm3.m`. The missing data results were created with `demCmu35gplvmReconstruct.m` and the nearest neighbour results from Taylor et al. are recreated in `demCmu35TaylorNearestNeighbour.m` (version 0.151 of the code).

B Matrix Derivatives

In what follows we will make use of matrix derivatives. Broadly speaking we follow the notation suggest in the The Matrix Reference Manual [Brookes, 2005]. We consider the derivative of one vector with respect to another as $\frac{d\mathbf{a}}{d\mathbf{b}} \in \Re^{m\times n}$ if $\mathbf{a} \in \Re^{m\times 1}$ and $\mathbf{b} \in \Re^{n\times 1}$. To obtain matrix-matrix derivatives we make use of $\mathbf{C}$: to indicate a vector formed from the matrix $\mathbf{C}$ by stacking the columns of $\mathbf{C}$ to form a $\Re^{mn\times 1}$ vector if $\mathbf{C} \in \Re^{m\times n}$. Under this notation we can write the derivative of a matrix $\mathbf{E} \in \Re^{p\times q}$ with respect to $\mathbf{C}$ as $\frac{d\mathbf{E}}{d\mathbf{C}} \in \Re^{pq\times mn}$. This notation makes it easier to apply the chain rule while maintaining matrix notation. This entails the use of Kronecker products, we denote the Kronecker product of $\mathbf{F}$ and $\mathbf{G}$ as $\mathbf{F}\otimes\mathbf{G}$. In most cases where they arise below they are later removed using this relationship

$$(\mathbf{E}:)^{\mathrm{T}}\,\mathbf{F}\otimes\mathbf{G} = \left(\left(\mathbf{G}^{\mathrm{T}}\mathbf{E}\mathbf{F}\right):\right)^{\mathrm{T}}, \qquad (2)$$

this form typically arises whenever the chain rule is applied,

$$\frac{dL}{d\mathbf{H}:}\frac{d\mathbf{H}:}{d\mathbf{J}:} = \frac{dL}{d\mathbf{J}:},$$

as we normally find that $\frac{d\mathbf{H}}{d\mathbf{J}}$ has the form of a Kronecker product, $\frac{d\mathbf{H}}{d\mathbf{J}} = \mathbf{F}\otimes\mathbf{G}$ and we expect the result of $\frac{dL}{d\mathbf{H}}$ and $\frac{dL}{d\mathbf{J}}$ to be in the form $(\mathbf{L}:)^{\mathrm{T}}$. The following two identities for Kronecker products will also prove useful.

$$\mathbf{F}^{\mathrm{T}}\otimes\mathbf{G}^{\mathrm{T}} = (\mathbf{F}\otimes\mathbf{G})^{\mathrm{T}}$$

and

$$(\mathbf{E}\otimes\mathbf{G})(\mathbf{F}\otimes\mathbf{H}) = \mathbf{E}\mathbf{F}\otimes\mathbf{G}\mathbf{H}.$$

In what we present below there are two ways of writing the derivative of a scalar with respect to a matrix, $\frac{dL}{d\mathbf{J}}$ and $\frac{dL}{d\mathbf{J}}$, the first being a row vector and the second is a matrix of the same dimension of $\mathbf{J}$. The second representation is more convenient for summarising the result, the first is easier to wield when computing the result. The equivalence of the representations is given by

$$\frac{dL}{d\mathbf{J}:} = \left(\left(\frac{dL}{d\mathbf{J}}\right):\right)^{\mathrm{T}}.$$

Any other results used for matrix differentiation and not explicitly given here may be found in Brookes [2005].

C FITC and PITC

In this appendix we consider the gradient required for the FITC and PITC approximations. The DTC gradients are more straightforward, we refer the reader to Lawrence [2006] for details. The log likelihood of the training data is given by:

$$\begin{aligned} L(\mathbf{X}_+,\boldsymbol{\theta}) &= -\frac{d}{2}\log\left|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}+\beta^{-1}\mathbf{D}\right| \\ &\quad -\frac{1}{2}\mathrm{tr}\left(\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}+\beta^{-1}\mathbf{D}\right)^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right).\end{aligned}$$

We can apply the matrix inversion lemma and re-express the determinant to obtain the following expression for the log likelihood of the training data

$$\begin{aligned} L(\mathbf{X},\mathbf{U},\boldsymbol{\theta}) &= -\frac{d}{2}\log|\mathbf{D}| - \frac{\beta}{2}\mathrm{tr}\left(\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right) \\ &\quad +\frac{d}{2}\log|\mathbf{K}_{\mathbf{u},\mathbf{u}}| - \frac{d}{2}\log|\mathbf{A}| \\ &\quad +\frac{\beta}{2}\mathrm{tr}\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right),\end{aligned}$$

where we define[7] $\mathbf{A}$ to be

$$\mathbf{A} = \left(\frac{1}{\beta}\mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right).$$

We first consider gradients with respect to $\mathbf{A}$.

$$\frac{dL(\mathbf{X}_+,\boldsymbol{\theta})}{d\mathbf{A}} = -\frac{1}{2}\mathbf{C}$$

where $\mathbf{C} = \mathbf{A}^{-1}d - \beta\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{D}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}$. Gradients with respect to $\mathbf{D}$ are given by

$$\frac{dL(\mathbf{X}_+,\boldsymbol{\theta})}{d\mathbf{D}:} = -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}} - \frac{1}{2}(\mathbf{C}:)^{\mathrm{T}}\frac{d\mathbf{A}:}{d\mathbf{D}:} \qquad (3)$$

where $\mathbf{H} = \left(\mathbf{D}^{-1}d - \beta\mathbf{Y}\mathbf{Y}^{\mathrm{T}} + 2\beta\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right)$. The gradient of $\mathbf{A}$: with respect to $\mathbf{D}$: is given by

$$\begin{aligned}\frac{\partial\mathbf{A}:}{\partial\mathbf{D}:} &= -\left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\otimes\mathbf{K}_{\mathbf{u},\mathbf{f}}\right)\left(\mathbf{D}^{-1}\otimes\mathbf{D}^{-1}\right) \\ &= -\left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\otimes\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right). \qquad (4)\end{aligned}$$

which allows us to combine (3) with (2) to write $\frac{dL(\mathbf{X}_+,\boldsymbol{\theta})}{d\mathbf{D}} = -\frac{1}{2}\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}$where $\mathbf{J} = \mathbf{H} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}}$. We are now in a position to write

$$\begin{aligned}\frac{dL(\mathbf{X}_+,\boldsymbol{\theta})}{d\mathbf{K}_{\mathbf{u},\mathbf{u}}:} &= \frac{d}{2}\left(\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right):\right)^{\mathrm{T}} - \frac{1}{2}(\mathbf{C}:)^{\mathrm{T}}\frac{d\mathbf{A}:}{d\mathbf{K}_{\mathbf{u},\mathbf{u}}:} \\ &\quad -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}\frac{d\mathbf{D}:}{d\mathbf{K}_{\mathbf{u},\mathbf{u}}:} \qquad (5)\end{aligned}$$

[7]Another possible definition for $\mathbf{A}$ would be $\hat{\mathbf{A}} = \left(\mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{u},\mathbf{f}}\hat{\mathbf{D}}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right)$ where $\hat{\mathbf{D}} = \beta^{-1}\mathbf{D}$, our original implementations (FGPLVM up to version 0.132) used this representation which is in line with the notation used by Quiñonero Candela and Rasmussen [2005]. However in their implementation Snelson and Ghahramani [2006] used something more akin to the representation we give here. We found this representation to be much more numerically stable (FGPLVM versions from 0.14).

and

$$\begin{aligned}\frac{dL\left(\mathbf{X}_{+},\boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{u},\mathbf{f}}:} &= -\frac{1}{2}\left(\mathbf{C}:\right)^{\mathrm{T}}\frac{d\mathbf{A}:}{d\mathbf{K}_{\mathbf{u},\mathbf{f}}:}\\ &\quad -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}\frac{d\mathbf{D}:}{d\mathbf{K}_{\mathbf{u},\mathbf{f}}:}\\ &\quad +\beta\left(\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}} \quad (6)\end{aligned}$$

gradients with respect to $\mathbf{K}_{\mathbf{f},\mathbf{f}}$ can be found through

$$\frac{dL\left(\mathbf{X}_{+},\boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{f},\mathbf{f}}:} = -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}\frac{d\mathbf{D}:}{d\mathbf{K}_{\mathbf{f},\mathbf{f}}:} \quad (7)$$

with a similar result holding for β. It remains to compute the derivatives of $\mathbf{D}$ and $\mathbf{A}$ with respect to $\mathbf{K}_{\mathbf{u},\mathbf{u}}$, $\mathbf{K}_{\mathbf{f},\mathbf{u}}$ and $\mathbf{K}_{\mathbf{f},\mathbf{f}}$.

We define $\mathbf{D}$ as

$$\mathbf{D} = \mathbf{I} + \beta\text{mask}\left(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}, \mathbf{M}\right).$$

Now we note that

$$\frac{d\text{mask}\left(\mathbf{Z},\mathbf{M}\right):}{d\mathbf{Z}:} = \text{diag}\left(\mathbf{M}:\right)$$

where the function diag $(\mathbf{z})$ takes a vector $\mathbf{z}$ and returns a diagonal matrix whose diagonal elements are given by $\mathbf{z}$. These definitions allow us to write

$$\begin{aligned}d\mathbf{D}: &= -\beta\text{diag}\left(\mathbf{M}:\right)\left(\left(\mathbf{I}\otimes\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right)d\mathbf{K}_{\mathbf{u},\mathbf{f}}:\right.\\ &\quad \left.+\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\otimes\mathbf{I}\right)d\mathbf{K}_{\mathbf{f},\mathbf{u}}:\right) \quad (8)\end{aligned}$$

and

$$\begin{aligned}\frac{\partial\mathbf{D}:}{\partial\mathbf{K}_{\mathbf{u},\mathbf{u}}:} &= \beta\text{diag}\left(\mathbf{M}:\right)\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\otimes\mathbf{K}_{\mathbf{f},\mathbf{u}}\right)\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\otimes\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right)\\ &= \beta\text{diag}\left(\mathbf{M}:\right)\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\otimes\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right)^{\mathrm{T}}\end{aligned}$$

with

$$\frac{\partial\mathbf{D}:}{\partial\mathbf{K}_{\mathbf{f},\mathbf{f}}:} = \beta\text{diag}\left(\mathbf{M}:\right) \quad (9)$$

and

$$\frac{\partial\mathbf{D}:}{\partial\beta} = \text{mask}\left(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}, \mathbf{M}\right):. \quad (10)$$

We can now use these equations in combination with gradients of $\mathbf{A}$ to obtain gradients of $L\left(\mathbf{X}_{+},\boldsymbol{\theta}\right)$ with respect to $\mathbf{K}_{\mathbf{u},\mathbf{u}}$, $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{f},\mathbf{f}}$, $\frac{d\mathbf{A}}{d\mathbf{K}_{\mathbf{u},\mathbf{u}}} = \frac{1}{\beta}\mathbf{I}$, where we have ignored the dependence of $\mathbf{D}$ on $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ as we already accounted for that in (5). Combining the gradient of $\mathbf{A}$ and those of $\mathbf{D}$ with (5) above we obtain

$$\frac{dL\left(\mathbf{X}_{+},\boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{u},\mathbf{u}}:} = \frac{1}{2}\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}d - \frac{1}{\beta}\mathbf{C} - \beta\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Q}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right),$$

where $\mathbf{Q} = \text{mask}\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1},\mathbf{M}\right)$ and we have made use of the fact that

$$\left(\mathbf{E}:\right)^{\mathrm{T}}\text{diag}\left(\mathbf{M}:\right)\left(\mathbf{F}\otimes\mathbf{G}\right)^{\mathrm{T}} = \left(\mathbf{G}^{\mathrm{T}}\text{mask}\left(\mathbf{E},\mathbf{M}\right)\mathbf{F}^{\mathrm{T}}\right):.$$

Similarly with respect to $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ we find

$$d\mathbf{A}: = \left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\otimes\mathbf{I}\right)d\mathbf{K}_{\mathbf{u},\mathbf{f}} + \left(\mathbf{I}\otimes\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right)d\mathbf{K}_{\mathbf{f},\mathbf{u}} \quad (11)$$

Substituting (8) and (11) into (6) above we obtain

$$\begin{aligned}dL\left(\mathbf{X}_{+},\boldsymbol{\theta}\right) &= -\frac{1}{2}\left(\left(\mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}d\mathbf{K}_{\mathbf{u},\mathbf{f}}\\ &\quad -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{C}\right):\right)^{\mathrm{T}}d\mathbf{K}_{\mathbf{f},\mathbf{u}}\\ &\quad +\frac{\beta}{2}\left(\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Q}\right):\right)^{\mathrm{T}}d\mathbf{K}_{\mathbf{u},\mathbf{f}}\\ &\quad +\frac{\beta}{2}\left(\left(\mathbf{Q}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right):\right)d\mathbf{K}_{\mathbf{f},\mathbf{u}}\\ &\quad +\beta\left(\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}d\mathbf{K}_{\mathbf{u},\mathbf{f}}\end{aligned}$$

which leads to

$$\begin{aligned}\frac{dL\left(\mathbf{X}_{+},\boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{u},\mathbf{f}}} &= -\mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1} + \beta\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Q}\\ &\quad +\beta\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{D}^{-1}.\end{aligned}$$

By substituting (9) into (7) we obtain

$$\frac{dL\left(\mathbf{X}_{+},\boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{f},\mathbf{f}}} = -\frac{\beta}{2}\mathbf{Q}$$

and finally by substituting (10) into (7) we have

$$\frac{dL\left(\mathbf{X}_{+},\boldsymbol{\theta}\right)}{d\beta} = \frac{1}{2}\text{tr}\left(\mathbf{Q}\right).$$

Once again these gradients may be combined with gradients of the kernel with respect to $\mathbf{X}$, $\mathbf{X}_{\mathbf{u}}$ and $\boldsymbol{\theta}$ to obtain the relevant gradients for their optimisation.