Large Scale Learning with the Gaussian Process Latent Variable Model

Neil D. Lawrence

December 16, 2008

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. We briefly consider a GPR toy problem to highlight the strengths and weaknesses of the different approaches before studying the perfomance of these techniques on a benchmark visualisation data set.

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] can be adapted to work with the GP-LVM.

2 Gaussian Process Latent Variable Models

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 embedding 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 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 $\mathbf{W} = [\mathbf{w}_1 \dots \mathbf{w}_d]^{\mathrm{T}}$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 the embeddings 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 embeddings, $p\left(\mathbf{Y}|\mathbf{X}\right)$. If the mappings are linear,

$$y_{nj} = \mathbf{x}_n^{\mathrm{T}}\mathbf{w}_j,$$

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 $f\left(\mathbf{x}_i, \mathbf{w}\right)$ we can obtain non-linear mappings. An appropriate, and tractable, process prior is a Gaussian Process (GP). If the GP prior over each of the d functions is the same we obtain the following likelihood,

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

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 embeddings, $\mathbf{X}$, and the parameters of the kernel $\boldsymbol{\theta}$. We therefore consider the log-likelihood given by,

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

Gradients of $L\left(\mathbf{X}, \boldsymbol{\theta}\right)$ 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.

[1] A special case that is 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.

4 Sparse Approximations in Gaussian Process Regression

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 $O\left(k^2 N\right)$, 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) which 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

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

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

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

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, $\mathbf{u}$:

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

where

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

is the training conditional in QR-terminology and

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

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 note in itself entail any approximations: the approximations are introduced through assumptions about the form of these distributions.

[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.

4.1 Deterministic Training Conditional

The first approximation we consider in the context of the GP-LVM is known as the 'deterministic training conditional' 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 on-line 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

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

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

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

where we have used $\mathbf{X}_+ = \{\mathbf{X}_\mathbf{u}, \mathbf{X}\}$ to represent the augmented set of training inputs and 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 Appendix C. Having optimised with respect to these parameters predictions can be made for test points using

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

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

4.2 Fully and Partially Independent Training Conditionals

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 specified considered through the following form for the training conditional,

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

[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.

where the function $\mathbf{V} = \text{mask}(\mathbf{Z}, \text{M})$, 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

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

which may again be combined with $p(\mathbf{Y}|\mathbf{F}, \beta)$ to obtain the log-likelihood

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

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

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 subset 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 Inducing Variables in 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 devised a small series of experiments that allow us to compare the performance of the new sparse algorithm to the original, IVM-based sparse algorithm and the model without sparse approximation. First of all we will briefly study a toy regression model to gain a better intuition about how these models perform.

Approximation	DTC	FITC	PITC
$\beta^{-\frac{1}{2}}$	1.18	0.895	0.964
$\gamma^{-\frac{1}{2}}$	1.19	0.978	0.951

Table 1: Estimates of the standard deviation of the noise, $\beta^{-\frac{1}{2}}$, and the length scale, $\gamma^{-\frac{1}{2}}$, for the different approximations, the true noise level is $\beta^{-\frac{1}{2}} = 0.1$ and the true length scale is $\gamma^{-\frac{1}{2}} = \frac{1}{\sqrt{20}} = 0.224$. Below we show the estimated values for each method divided by the true value.

	DTC	FITC	PITC
$\mathrm{KL}(p\|\|q)$	42.9	12.3	7.50
$\mathrm{KL}(q\|\|p)$	1.31×10^4	46.7	51.7

Table 2: The Kullback-Leibler divergence between the true posterior and each of the approximations as calculated at each of the training inputs. Two KL divergences are presented, $\mathrm{KL}(p||q)$ is the divergence with the expectation under the true distribution, $\mathrm{KL}(q||p)$ is the divergence with the expectation taken under the approximating distribution.

5.1 1-D Regression Toy Problem

To illustrate the different models we first consider them in the context of a simple 1-dimensional regression problem. We sampled a 1-dimensional vector of 500 inputs uniformly from between -1 and 1. We then sampled a function from a Gaussian process with an RBF kernel,

$$k(x_i, x_j) = \exp\left(-\gamma (x_i - x_j)^2\right) + \beta^{-1}\delta_{ij},$$

with $\gamma = 20$ and $\beta = 100$ a vector of targets. We then optimised a Gaussian process model with each of the sparse approximations described above taking $k = 9$ in each case. The results of optimisation of each approximation through conjugate gradients are given in Figure 1. We note that there is a major difference between the result from the DTC (Figure 1(a)) and the other two approximations. The quality of the error bars for the DTC approximation is much worse than that of the FITC and PITC approximations. Another interesting difference is found between the level of noise and the length scale each method estimates (Table 1). The DTC overestimates both the noise and the length scale by close to 20 percentage points. The FITC underestimates the noise (by 11 percentage points) but obtains a much better estimate of the length scale. The PITC[5] improves over FITC on both the estimate of noise and has a slightly worse estimate of the length scale.

Finally we summarise the quality of each approximating method by considering the Kullback-Leibler (KL) divergence,

$$\mathrm{KL}(p||q) = \int p(\mathbf{f}) \log \frac{p(\mathbf{f})}{q(\mathbf{f})} d\mathbf{f}$$

between the posterior predictions of the function at the training points under the true model and that from each approximation. This measure takes into account correlations in the posterior covariance which cannot be shown in the plots in Figure 1.

The KL divergence is asymmetric, the expectation can taken either under the true distribution or the approximating distribution. It is often implicitly considered that taking the expectation under the true distribution is the 'right' way round, however this is a fallacy. The two KL divergences measure different things: $\mathrm{KL}(p||q)$ penalises stongly approximating distributions that fail to put probability mass where there is mass under the true distribution. $\mathrm{KL}(q||p)$ penalises strongly approximating distributions that put mass where there is no mass under the true distribution. In legal terms we wish the approximation to represent the truth, the *whole truth* and *nothing but the truth*, $\mathrm{KL}(p||q)$ asseses wether we have the whole truth; whereas $\mathrm{KL}(q||p)$ asseses whether we have

[5]For the PITC approximation, as suggested by Tresp [2000], Quiñonero Candela and Rasmussen [2005], we used block sizes of 9×9 which keeps the computational complexity of the algorithm at $O(k^2 N)$.

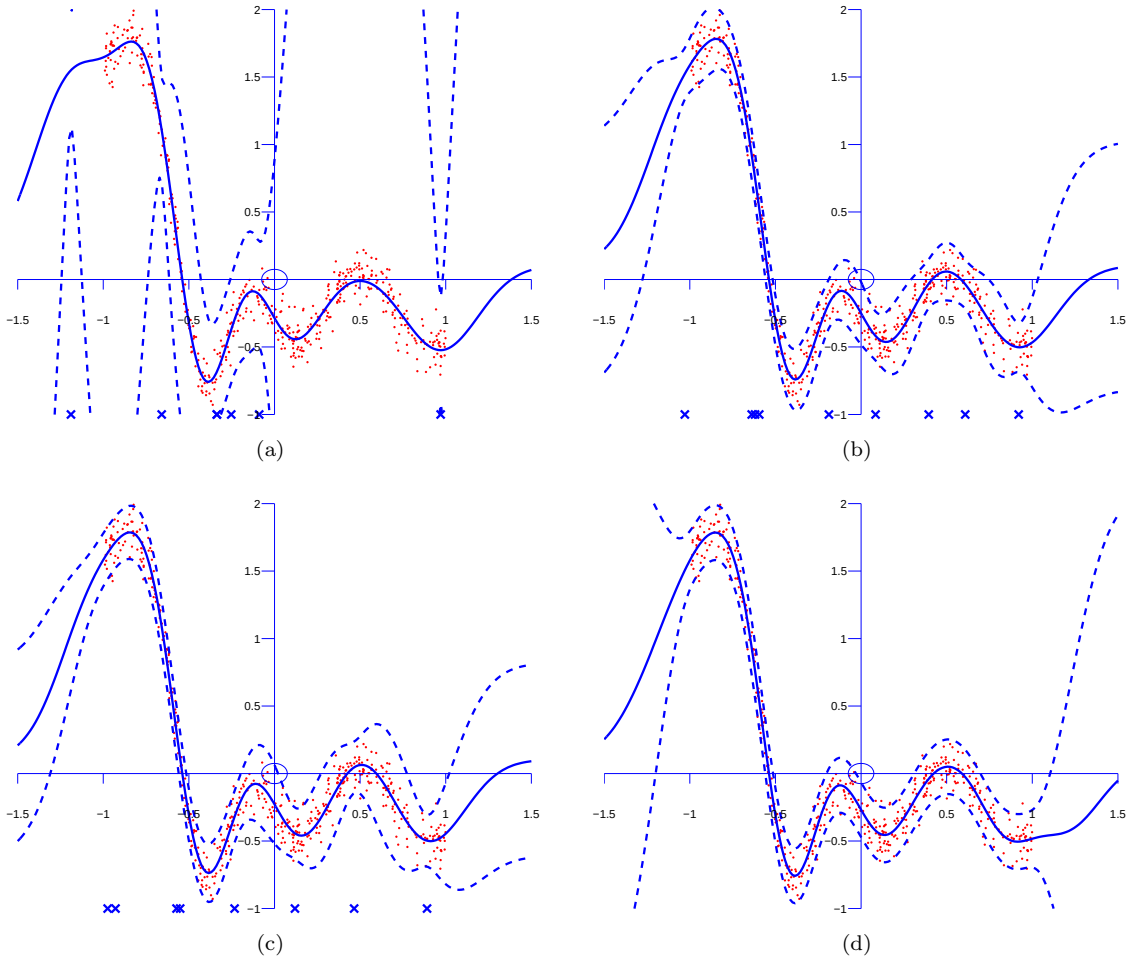

Figure 1: One dimensional regression toy problem with $N = 500$ and $k = 9$. (a) The DTC approximation, (b) the FITC approximation and (c) the PITC approximation. (d) The posterior prediction from the full Gaussian process with the correct hyperparameters. The data is marked as dots, the posterior mean prediction is given as a solid line and error bars at two standard deviations are given as dashed lines. The final locations of the inducing variables are show as crosses at the bottom of each plot (in each case not all of the inducing variables can be seen as some have drifted out of the range of the plot).

	DTC	FITC	PITC	IVM	Full	In Y
Errors	3	6	6	24	1	2

Table 3: 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').

nothing but the truth. With this in mind we can interpret Table 2 as follows, under both criteria DTC performs the worst; the PITC approximation appears better at telling the whole truth than the FITC approximation and the FITC approximation appears better at telling nothing but the truth than the PITC approximation. In conclusion, it seems that we should expect considerably better performance from the FITC and PITC approximations when they are used in the GP-LVM than the DTC approximation. In the next section we will attempt to assess if this is indeed the case using a well known benchmark data set.

5.2 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. The data is visualised by optimisation of GP-LVMs with two dimensional latent spaces using the three different approximations we have outlined in Figure 2. As well as the embeddings 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 the variances are low along spray-paint-like streaks across the latent space. This effect becomes less pronounced with the FITC approximation and is almost non-existent with the PITC approximation.

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 3. 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.

6 Discussion

In this report 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.

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 which provided support for the Gaussian Process Round Table where many of the ideas presented in this paper were first presented and discussed.

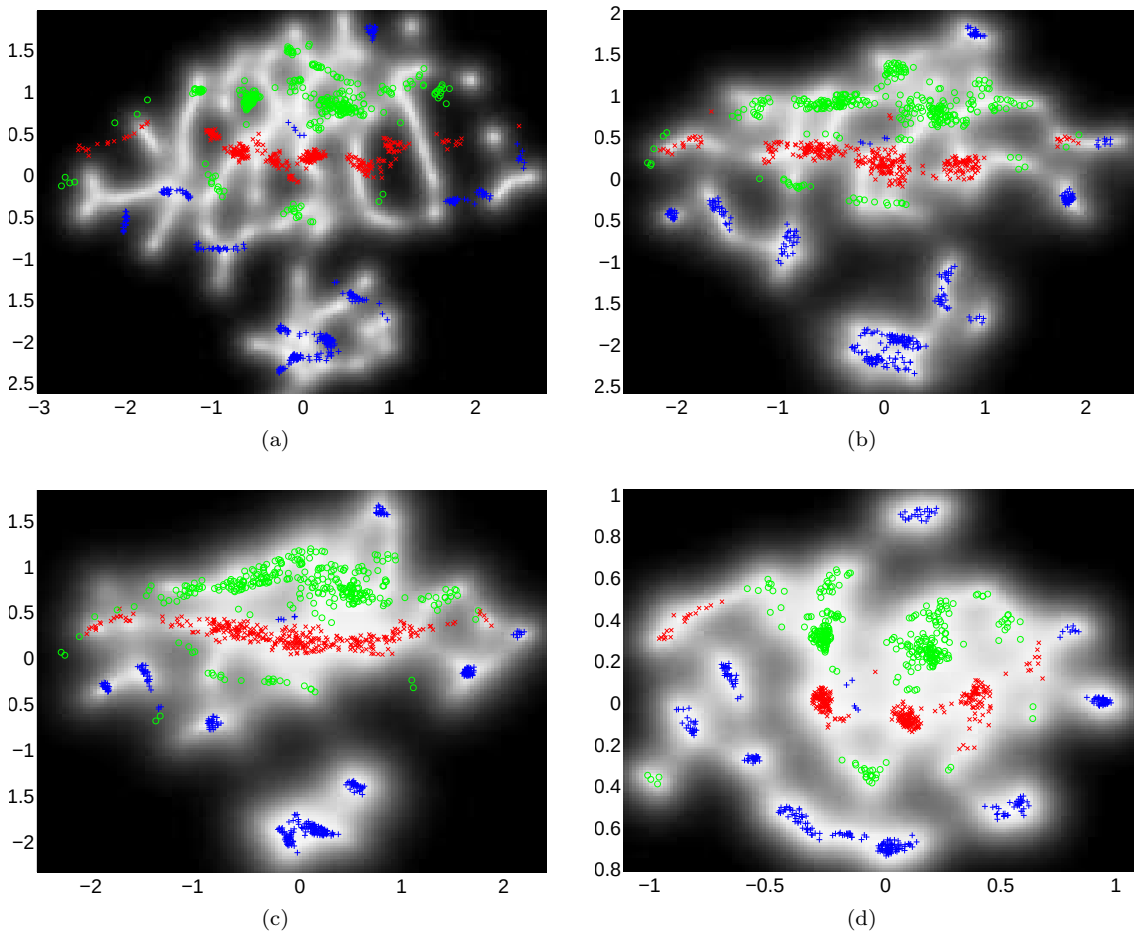

Figure 2: The oil data visualised using a GP-LVM with (a) the DTC approximation, (b) the FITC approximation, (c) the PITC approximation and (d) no approximation. Red crosses, green circles and blue plus signs represent stratified, annular and homogenous respectively. The greyscale background to the plots visualises the precision with which the posterior process is mapped in the data space.

References

C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995. ISBN 0-198-53864-2.

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. doi: 10.1016/0168-9002(93)90728-Z.

C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: the Generative Topographic Mapping. *Neural Computation*, 10(1):215–234, 1998. doi: 10.1162/089976698300017953.

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

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`.

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

K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popovic. Style-based inverse kinematics. In *ACM Transactions on Graphics (SIGGRAPH 2004)*, pages 522–531, 2004. doi: 10.1145/1186562.1015755.

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.

N. D. Lawrence. Probabilistic non-linear principal component analysis with Gaussian process latent variable models. *Journal of Machine Learning Research*, 6:1783–1816, 11 2005.

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: The informative vector machine. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 625–632, Cambridge, MA, 2003. MIT Press.

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

D. J. C. MacKay. Bayesian neural networks and density networks. *Nuclear Instruments and Methods in Physics Research, A*, 354(1):73–80, 1995. doi: 10.1016/0168-9002(94)00931-7.

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

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.

A. Schwaighofer and V. Tresp. Transductive and inductive methods for approximate Gaussian process regression. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15, pages 953–960, Cambridge, MA, 2003. MIT Press.

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*, Key West, FL, Jan 3–6 2003.

A. J. Smola and P. L. Bartlett. Sparse greedy Gaussian process regression. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 619–625, Cambridge, MA, 2001. MIT Press.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems*, volume 18, Cambridge, MA, 2006. MIT Press.

M. E. Tipping and C. M. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, B*, 6(3):611–622, 1999. doi: doi:10.1111/1467-9868.00196.

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

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. doi: 10.1109/ICCV.2005.193.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems*, volume 13, pages 682–688, Cambridge, MA, 2001. MIT Press.

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.

A On-line Source Code

Matlab source code for repeating the experiments described in Section 5 and **??** is available on-line from `http://www.dcs.shef.ac.uk/~neil/fgplvm`, the version number is 0.13. The three one dimensional regression problems may be re-run with the scripts `demSpgp1d1.m` for DTC, `demSpgp1d2.m` for FITC and `demSpgp1d3.m` for PITC. The ground truth is given by `demSpgp1d4.m` and the KL divergences are computed in `demSpgp1dKlDiv.m` .

The oil experiments were created using the scripts `demOil3.m` for DTC, `demOil1.m` for FITC and `demOil5.m` for PITC (`demOil2.m`, `demOil4.m` and `demOil6.m` are results that use back constraints and are not presented here). The full GP-LVM was created using C++ code available from `http://www.dcs.shef.ac.uk/~neil/gplvmcpp`, see instructions on that site for how to recreate it.

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

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

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

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

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 The Deterministic Training Conditional

Up to constant terms, the deterministic training conditional leads to a likelihood for the training data of the following form,

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

We deal with the inverse in the trace through the matrix inversion lemma giving,

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

Similarly the determinant can be re-expressed as

$$\begin{aligned}\left|\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}} + \beta^{-1}\mathbf{I}\right| &= \beta^{-N}\left|\mathbf{K}_{\mathbf{u},\mathbf{u}}\right|^{-1}\left|\mathbf{K}_{\mathbf{u},\mathbf{u}} + \beta\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right| \\ &= \beta^{-(N-k)}\left|\mathbf{K}_{\mathbf{u},\mathbf{u}}\right|^{-1}\left|\beta^{-1}\mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right|\end{aligned}$$

where k is the size of the active set. This leads to a more efficient-to-compute form of the likelihood,

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

where we have defined

$$\mathbf{A} = \beta^{-1}\mathbf{K}_{\mathbf{u},\mathbf{u}} + \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{K}_{\mathbf{f},\mathbf{u}}.$$

We wish to find the derivative of the likelihood with respect to $\mathbf{X}$, $\boldsymbol{\theta}$ and $\mathbf{X}_{\mathbf{u}}$. We will aim to recover these derivatives by first considering the derivatives with respect to $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{f}}$. The derivative with respect to $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ is

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

where we have defined

$$\mathbf{C} = d\mathbf{A}^{-1} + \beta\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{A}^{-1}.$$

Given our definition of $\mathbf{A}$ we have

$$\frac{d\mathbf{A}_{:}}{d\mathbf{K}_{\mathbf{u},\mathbf{u}:}} = \beta^{-1}\mathbf{I} \otimes \mathbf{I}$$

which, applying (2) and undoing the vec operation on both sides gives,

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

The next gradient of interest is that with respect to $\mathbf{K}_{\mathbf{u},\mathbf{f}}$. First we may write

$$\frac{dL\left(\mathbf{X}_+,\boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{u},\mathbf{f}:}} = \frac{dL_{\mathbf{A}}(\mathbf{X}_+,\boldsymbol{\theta})}{d\mathbf{K}_{\mathbf{u},\mathbf{f}}} + \beta\left(\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right):\right)^{\mathrm{T}}$$

where $L_{\mathbf{A}}\left(\mathbf{X}_+,\boldsymbol{\theta}\right)$ is those terms of $L\left(\mathbf{X}_+,\boldsymbol{\theta}\right)$ which depend on $\mathbf{A}$. Consider the gradient of $L_{\mathbf{A}}\left(\mathbf{X}_+,\boldsymbol{\theta}\right)$ with respect to $\mathbf{A}$,

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

The gradients of $\mathbf{A}$ with respect to $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ are given by

$$d\mathbf{A}:= \left(\mathbf{K}_{\mathbf{u},\mathbf{f}} \otimes \mathbf{I}\right) d\mathbf{K}_{\mathbf{u},\mathbf{f}}: + \left(\mathbf{I} \otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\right) d\mathbf{K}_{\mathbf{f},\mathbf{u}}:,$$

substituting into (3) we have

$$\begin{aligned} dL_{\mathbf{A}}\left(\mathbf{X}_+,\boldsymbol{\theta}\right) &= -\frac{1}{2}\left(\mathbf{C}:\right)^{\mathrm{T}}\left(\left(\mathbf{K}_{\mathbf{u},\mathbf{f}} \otimes \mathbf{I}\right) d\mathbf{K}_{\mathbf{u},\mathbf{f}}: + \left(\mathbf{I} \otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\right) d\mathbf{K}_{\mathbf{f},\mathbf{u}}:\right) \\ &= -\frac{1}{2}\left(\left(\mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}}\right):\right)^{\mathrm{T}} - \frac{1}{2}\left(\left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{C}\right):\right)^{\mathrm{T}} d\mathbf{K}_{\mathbf{f},\mathbf{u}}:, \end{aligned}$$

where we have made use of (2) to obtain the second line. Due to the left-hand-side being a scalar, we can remove the stack operators from the right hand side without loss of generality recovering

$$dL_{\mathbf{A}}\left(\mathbf{X}_+,\boldsymbol{\theta}\right) = -\frac{1}{2}\mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}} d\mathbf{K}_{\mathbf{u},\mathbf{f}} - \frac{1}{2}\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{C} d\mathbf{K}_{\mathbf{f},\mathbf{u}}$$

which implies that

$$\frac{dL_{\mathbf{A}}\left(\mathbf{X}_+,\boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{u},\mathbf{f}}} = -\mathbf{C}\mathbf{K}_{\mathbf{u},\mathbf{f}},$$

therefore the gradient with respect to $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ is given by

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

Finally we need the gradient with respect to β,

$$\frac{dL\left(\mathbf{X}_+,\boldsymbol{\theta}\right)}{d\beta} = \frac{d\left(N-k\right)}{2\beta} - \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\right) + \frac{dL_{\mathbf{A}}\left(\mathbf{X}_+,\boldsymbol{\theta}\right)}{d\beta}$$

where

$$\frac{dL_{\mathbf{A}}\left(\mathbf{X}_+,\boldsymbol{\theta}\right)}{d\beta} = \frac{1}{2}\mathrm{tr}\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right) - \frac{1}{2}\left(\mathbf{C}:\right)^{\mathrm{T}}\frac{d\mathbf{A}:}{d\beta}$$

and

$$\frac{d\mathbf{A}:}{d\beta} = -\beta^{-2}\mathbf{K}_{\mathbf{u},\mathbf{u}}$$

giving

$$\frac{dL\left(\mathbf{X}_+,\boldsymbol{\theta}\right)}{d\beta} = \frac{d\left(N-k\right)}{2\beta} - \frac{1}{2}\mathrm{tr}\left(\mathbf{Y}\mathbf{Y}^{\mathrm{T}} - \mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{K}_{\mathbf{f},\mathbf{u}}\right) + \frac{\beta^{-2}}{2}\left(\mathbf{C}:\right)^{\mathrm{T}}\mathbf{K}_{\mathbf{u},\mathbf{u}}: .$$

Each of these gradients can be computed in $O\left(k^2N\right)$. It remains to compute the gradients of $\mathbf{K}_{\mathbf{u},\mathbf{f}}$ and $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ with respect to $\mathbf{X}$, $\mathbf{X}_{\mathbf{u}}$and $\boldsymbol{\theta}$. Since the nature of these gradients is dependent on the choice of kernel we leave this for the reader.

D Fully and Partially Independent Training Conditional

For both the fully independent and partially independent training conditional, we construct a matrix $\mathbf{D}$ which is chosen such that it replaces portions of the DTC approximation with elements from the original covariance matrix. This complicates computation of the gradients slightly over those obtained for the DTC approximation as the matrix $\mathbf{D}$ depends on $\mathbf{K}_{\mathbf{u},\mathbf{f}}$, $\mathbf{K}_{\mathbf{u},\mathbf{u}}$ and some elements of $\mathbf{K}_{\mathbf{f},\mathbf{f}}$. The log likelihood of the training data is given by:

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

Once again we can apply the matrix inversion lemma to the term within the trace to obtain

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

and the determinant can be re-expressed as

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

We now define[6] $\mathbf{A}$ to be

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

leading to the following expression for the log likelihood of the training data

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

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

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

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

$$\frac{dL\left(\mathbf{X}_+, \boldsymbol{\theta}\right)}{d\mathbf{D}_{:}} = -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{H}\mathbf{D}^{-1}\right)_{:}\right)^{\mathrm{T}} - \frac{1}{2}\left(\mathbf{C}_{:}\right)^{\mathrm{T}}\frac{d\mathbf{A}_{:}}{d\mathbf{D}_{:}} \tag{4}$$

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

$$\begin{aligned} \frac{\partial\mathbf{A}_{:}}{\partial\mathbf{D}_{:}} &= -\left(\mathbf{K}_{\mathbf{u},\mathbf{f}} \otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\right)\left(\mathbf{D}^{-1} \otimes \mathbf{D}^{-1}\right) \\ &= -\left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1} \otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right). \end{aligned} \tag{5}$$

which allows us to combine (4) with (2) to write

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

[6] 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 implementation used this representation which is in line with the notation used by Quiñonero Candela and Rasmussen [2005]. However in implementation Snelson and Ghahramani [2006] used something more akin to the representation we gave here. We found this representation to be much more numerically stable.

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

$$\frac{dL\left(\mathbf{X}_+, \boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{u},\mathbf{u}:}} = \frac{d}{2}\left(\left(\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right):\right)^{\mathrm{T}} - \frac{1}{2}\left(\mathbf{C}:\right)^{\mathrm{T}}\frac{d\mathbf{A}:}{d\mathbf{K}_{\mathbf{u},\mathbf{u}:}} - \frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}\frac{d\mathbf{D}:}{d\mathbf{K}_{\mathbf{u},\mathbf{u}:}} \tag{6}$$

and

$$\begin{aligned}\frac{dL\left(\mathbf{X}_+, \boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{u},\mathbf{f}:}} &= -\frac{1}{2}\left(\mathbf{C}:\right)^{\mathrm{T}}\frac{d\mathbf{A}:}{d\mathbf{K}_{\mathbf{u},\mathbf{f}:}} - \frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}\frac{d\mathbf{D}:}{d\mathbf{K}_{\mathbf{u},\mathbf{f}:}} \\ &\quad +\beta\left(\left(\mathbf{A}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\mathbf{Y}\mathbf{Y}^{\mathrm{T}}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}\end{aligned} \tag{7}$$

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

$$\frac{dL\left(\mathbf{X}_+, \boldsymbol{\theta}\right)}{d\mathbf{K}_{\mathbf{f},\mathbf{f}:}} = -\frac{1}{2}\left(\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}\right):\right)^{\mathrm{T}}\frac{d\mathbf{D}:}{d\mathbf{K}_{\mathbf{f},\mathbf{f}:}} \tag{8}$$

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

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

Now we note that

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

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

$$d\mathbf{D}:= -\beta\mathrm{diag}\left(\mathbf{M}:\right)\left(\left(\mathbf{I}\otimes\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\right)d\mathbf{K}_{\mathbf{u},\mathbf{f}}: + \left(\mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\otimes\mathbf{I}\right)d\mathbf{K}_{\mathbf{f},\mathbf{u}:}\right) \tag{9}$$

and

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

with

$$\frac{\partial\mathbf{D}:}{\partial\mathbf{K}_{\mathbf{f},\mathbf{f}:}} = \beta\mathrm{diag}\left(\mathbf{M}:\right) \tag{10}$$

and

$$\frac{\partial\mathbf{D}:}{\partial\beta} = \mathrm{mask}\left(\mathbf{K}_{\mathbf{f},\mathbf{f}} - \mathbf{K}_{\mathbf{f},\mathbf{u}}\mathbf{K}_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{K}_{\mathbf{u},\mathbf{f}}, \mathbf{M}\right): . \tag{11}$$

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 (6). Combining the gradient of $\mathbf{A}$ and those of $\mathbf{D}$ with (6) 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 $\mathbf{Q} = \mathrm{mask}\left(\mathbf{D}^{-1}\mathbf{J}\mathbf{D}^{-1}, \mathbf{M}\right)$ and we have made use of the fact that

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

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

$$d\mathbf{A}: \quad = \quad \left(\mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1} \otimes \mathbf{I}\right) d\mathbf{K}_{\mathbf{u},\mathbf{f}} + \left(\mathbf{I} \otimes \mathbf{K}_{\mathbf{u},\mathbf{f}}\mathbf{D}^{-1}\right) d\mathbf{K}_{\mathbf{f},\mathbf{u}}. \tag{12}$$

Substituting (9) and (12) into (7) above we obtain

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

which leads to

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

By substituting (10) into (8) 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 (11) into (8) we have

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

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.