Journal of Machine Learning Research 6 (2005) 1939–1959 Submitted 10/05; Published 12/05

# A Unifying View of Sparse Approximate Gaussian Process Regression

**Joaquin Quiñonero-Candela** JQC@TUEBINGEN.MPG.DE
**Carl Edward Rasmussen** CARL@TUEBINGEN.MPG.DE
*Max Planck Institute for Biological Cybernetics*
*Spemannstraße 38*
*72076 Tübingen, Germany*

**Editor:** Ralf Herbrich

## Abstract

We provide a new unifying view, including all existing proper probabilistic sparse approximations for Gaussian process regression. Our approach relies on expressing the *effective prior* which the methods are using. This allows new insights to be gained, and highlights the relationship between existing methods. It also allows for a clear theoretically justified ranking of the closeness of the known approximations to the corresponding full GPs. Finally we point directly to designs of new better sparse approximations, combining the best of the existing strategies, within attractive computational constraints.

**Keywords:** Gaussian process, probabilistic regression, sparse approximation, Bayesian committee machine

Regression models based on Gaussian processes (GPs) are simple to implement, flexible, fully probabilistic models, and thus a powerful tool in many areas of application. Their main limitation is that memory requirements and computational demands grow as the square and cube respectively, of the number of training cases $n$, effectively limiting a direct implementation to problems with at most a few thousand cases. To overcome the computational limitations numerous authors have recently suggested a wealth of *sparse* approximations. Common to all these approximation schemes is that only a subset of the latent variables are treated exactly, and the remaining variables are given some approximate, but computationally cheaper treatment. However, the published algorithms have widely different motivations, emphasis and exposition, so it is difficult to get an overview (see Rasmussen and Williams, 2006, chapter 8) of how they relate to each other, and which can be expected to give rise to the best algorithms.

In this paper we provide a unifying view of sparse approximations for GP regression. Our approach is simple, but powerful: for each algorithm we analyze the posterior, and compute the *effective prior* which it is using. Thus, we reinterpret the algorithms as “exact inference with an approximated prior”, rather than the existing (ubiquitous) interpretation “approximate inference with the exact prior”. This approach has the advantage of directly expressing the approximations in terms of prior assumptions about the function, which makes the consequences of the approximations much easier to understand. While our view of the approximations is not the only one possible, it has the advantage of putting all existing probabilistic sparse approximations under one umbrella, thus enabling direct comparison and revealing the relation between them.

In Section 1 we briefly introduce GP models for regression. In Section 2 we present our unifying framework and write out the key equations in preparation for the unifying analysis of sparse algorithms in Sections 4-7. The relation of transduction and augmentation to our sparse framework is covered in Section 8. All our approximations are written in terms of a new set of *inducing variables*. The choice of these variables is itself a challenging problem, and is discussed in Section 9. We comment on a few special approximations outside our general scheme in Section 10 and conclusions are drawn at the end.

## 1. Gaussian Processes for Regression

Probabilistic regression is usually formulated as follows: given a training set $\mathcal{D} = \{(\mathbf{x}_i, y_i), i = 1, \ldots, n\}$ of $n$ pairs of (vectorial) inputs $\mathbf{x}_i$ and noisy (real, scalar) outputs $y_i$, compute the predictive distribution of the function values $f_*$ (or noisy $y_*$) at test locations $\mathbf{x}_*$. In the simplest case (which we deal with here) we assume that the noise is additive, independent and Gaussian, such that the relationship between the (latent) function $f(\mathbf{x})$ and the observed noisy targets $y$ are given by

$$y_i = f(\mathbf{x}_i) + \varepsilon_i \,, \quad \text{where} \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2_{\text{noise}}) \,, \tag{1}$$

where $\sigma^2_{\text{noise}}$ is the variance of the noise.

**Definition 1** *A Gaussian process (GP) is a collection of random variables, any finite number of which have consistent*[1] *joint Gaussian distributions.*

Gaussian process (GP) regression is a Bayesian approach which assumes a GP prior[2] over functions, i.e. assumes a priori that function values behave according to

$$p(\mathbf{f}|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \mathcal{N}(\mathbf{0}, K) \,, \tag{2}$$

where $\mathbf{f} = [f_1, f_2, \ldots, f_n]^\top$ is a vector of latent function values, $f_i = f(\mathbf{x}_i)$ and $K$ is a covariance matrix, whose entries are given by the *covariance function*, $K_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$. Note that the GP treats the latent function values $f_i$ as random variables, indexed by the corresponding input. In the following, for simplicity we will always neglect the explicit conditioning on the inputs; the GP model and all expressions are always conditional on the corresponding inputs. The GP model is concerned only with the conditional of the outputs given the inputs; we do not model anything about the inputs themselves.

**Remark 2** *Note, that to adhere to a strict Bayesian formalism, the GP covariance* function,[3] *which defines the prior, should not depend on the data (although it can depend on additional parameters).*

As we will see in later sections, some approximations are strictly equivalent to GPs, while others are not. That is, the implied prior may still be multivariate Gaussian, but the covariance function may be different for training and test cases.

**Definition 3** *A Gaussian process is called* degenerate *iff the covariance function has a finite number of non-zero eigenvalues.*

1. By consistency is meant simply that the random variables obey the usual rules of marginalization, etc.
2. For notational simplicity we exclusively use zero-mean priors.
3. The covariance *function* itself shouldn't depend on the data, though its value at a specific pair of inputs of course will.

Degenerate GPs (such as e.g. with polynomial covariance function) correspond to *finite* linear (-in-the-parameters) models, whereas non-degenerate GPs (such as e.g. with squared exponential or RBF covariance function) do not. The prior for a finite $m$ dimensional linear model only considers a universe of at most $m$ linearly independent functions; this may often be too restrictive when $n \gg m$. Note however, that non-degeneracy on its own doesn't guarantee the existence of the "right kind" of flexibility for a given particular modelling task. For a more detailed background on GP models, see for example that of Rasmussen and Williams (2006).

Inference in the GP model is simple: we put a joint GP prior on training and test latent values, $\mathbf{f}$ and $\mathbf{f}_*$[4], and combine it with the likelihood[5] $p(\mathbf{y}|\mathbf{f})$ using Bayes rule, to obtain the joint posterior

$$p(\mathbf{f},\mathbf{f}_*|\mathbf{y}) \;=\; \frac{p(\mathbf{f},\mathbf{f}_*)\,p(\mathbf{y}|\mathbf{f})}{p(\mathbf{y})}\,. \tag{3}$$

The final step needed to produce the desired posterior predictive distribution is to marginalize out the unwanted training set latent variables:

$$p(\mathbf{f}_*|\mathbf{y}) \;=\; \int p(\mathbf{f},\mathbf{f}_*|\mathbf{y})\mathrm{d}\mathbf{f} \;=\; \frac{1}{p(\mathbf{y})}\int p(\mathbf{y}|\mathbf{f})\,p(\mathbf{f},\mathbf{f}_*)\,\mathrm{d}\mathbf{f}\,, \tag{4}$$

or in words: the predictive distribution is the marginal of the renormalized joint prior times the likelihood. The joint GP prior and the independent likelihood are both Gaussian

$$p(\mathbf{f},\mathbf{f}_*) \;=\; \mathcal{N}\Big(\mathbf{0}, \begin{bmatrix} K_{\mathbf{f},\mathbf{f}} & K_{*,\mathbf{f}} \\ K_{\mathbf{f},*} & K_{*,*} \end{bmatrix}\Big), \quad \text{and} \quad p(\mathbf{y}|\mathbf{f}) \;=\; \mathcal{N}(\mathbf{f},\, \sigma^2_{\text{noise}} I)\,, \tag{5}$$

where $K$ is subscript by the variables between which the covariance is computed (and we use the asterisk $*$ as shorthand for $\mathbf{f}_*$) and $I$ is the identity matrix. Since both factors in the integral are Gaussian, the integral can be evaluated in closed form to give the Gaussian predictive distribution

$$p(\mathbf{f}_*|\mathbf{y}) \;=\; \mathcal{N}\big(K_{*,\mathbf{f}}(K_{\mathbf{f},\mathbf{f}}+\sigma^2_{\text{noise}}I)^{-1}\mathbf{y},\; K_{*,*}-K_{*,\mathbf{f}}(K_{\mathbf{f},\mathbf{f}}+\sigma^2_{\text{noise}}I)^{-1}K_{\mathbf{f},*}\big)\,, \tag{6}$$

see the relevant Gaussian identity in appendix A. The problem with the above expression is that it requires inversion of a matrix of size $n\times n$ which requires $\mathcal{O}(n^3)$ operations, where $n$ is the number of training cases. Thus, the simple exact implementation can handle problems with at most a few thousand training cases.

## 2. A New Unifying View

We now seek to modify the joint prior $p(\mathbf{f}_*,\mathbf{f})$ from (5) in ways which will reduce the computational requirements from (6). Let us first rewrite that prior by introducing an additional set of $m$ latent variables $\mathbf{u} = [u_1,\ldots,u_m]^\top$, which we call the *inducing variables*. These latent variables are values of the Gaussian process (as also $\mathbf{f}$ and $\mathbf{f}_*$), corresponding to a set of input locations $X_\mathbf{u}$, which we call the *inducing inputs*. Whereas the additional latent variables $\mathbf{u}$ are always marginalized out in the predictive distribution, the choice of inducing inputs *does* leave an imprint on the final solution.

4. We will mostly consider a vector of test cases $\mathbf{f}_*$ (rather than a single $f_*$).
5. You may have been expecting the likelihood written as $p(\mathbf{y}|\mathbf{f}_*,\mathbf{f})$ but since the likelihood is conditionally independent of everything else given $\mathbf{f}$, this makes no difference.

The inducing variables will turn out to be generalizations of variables which other authors have referred to variously as "support points", "active set" or "pseudo-inputs". Particular sparse algorithms choose the inducing variables in various different ways; some algorithms chose the inducing inputs to be a subset of the training set, others not, as we will discuss in Section 9. For now consider any arbitrary inducing variables.

Due to the *consistency* of Gaussian processes, we know that we can recover $p(\mathbf{f}_*, \mathbf{f})$ by simply integrating (marginalizing) out $\mathbf{u}$ from the joint GP prior $p(\mathbf{f}_*, \mathbf{f}, \mathbf{u})$

$$p(\mathbf{f}_*, \mathbf{f}) = \int p(\mathbf{f}_*, \mathbf{f}, \mathbf{u})\, d\mathbf{u} = \int p(\mathbf{f}_*, \mathbf{f}|\mathbf{u})\, p(\mathbf{u})\, d\mathbf{u}, \quad \text{where} \quad p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, K_{\mathbf{u},\mathbf{u}}) . \tag{7}$$

This is an exact expression. Now, we introduce the fundamental approximation which gives rise to almost all sparse approximations. We approximate the joint prior by assuming that $\mathbf{f}_*$ and $\mathbf{f}$ are *conditionally independent given* $\mathbf{u}$, see Figure 1, such that

$$p(\mathbf{f}_*, \mathbf{f}) \simeq q(\mathbf{f}_*, \mathbf{f}) = \int q(\mathbf{f}_*|\mathbf{u})\, q(\mathbf{f}|\mathbf{u})\, p(\mathbf{u})\, d\mathbf{u} . \tag{8}$$

The name *inducing* variable is motivated by the fact that $\mathbf{f}$ and $\mathbf{f}_*$ can only communicate though $\mathbf{u}$, and $\mathbf{u}$ therefore *induces* the dependencies between training and test cases. As we shall detail in the following sections, the different computationally efficient algorithms proposed in the literature correspond to different *additional assumptions* about the two approximate *inducing* conditionals $q(\mathbf{f}|\mathbf{u})$, $q(\mathbf{f}_*|\mathbf{u})$ of the integral in (8). It will be useful for future reference to specify here the exact expressions for the two conditionals

$$\textit{training}\text{ conditional:} \qquad p(\mathbf{f}|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\, K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}) , \tag{9a}$$

$$\textit{test}\text{ conditional:} \qquad p(\mathbf{f}_*|\mathbf{u}) = \mathcal{N}(K_{*,\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\, K_{*,*} - Q_{*,*}) , \tag{9b}$$

where we have introduced the shorthand notation[6] $Q_{\mathbf{a},\mathbf{b}} \triangleq K_{\mathbf{a},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}K_{\mathbf{u},\mathbf{b}}$. We can readily identify the expressions in (9) as special (noise free) cases of the standard predictive equation (6) with $\mathbf{u}$ playing the role of (noise free) observations. Note that the (positive semi-definite) covariance matrices in (9) have the form $K - Q$ with the following interpretation: the prior covariance $K$ minus a (non-negative definite) matrix $Q$ quantifying how much information $\mathbf{u}$ provides about the variables in question ($\mathbf{f}$ or $\mathbf{f}_*$). We emphasize that all the sparse methods discussed in the paper correspond simply to different approximations to the conditionals in (9), and throughout we use the exact likelihood and inducing prior

$$p(\mathbf{y}|\mathbf{f}) = \mathcal{N}(\mathbf{f}, \sigma^2_{\text{noise}} I) , \quad \text{and} \quad p(\mathbf{u}) = \mathcal{N}(\mathbf{0}, K_{\mathbf{u},\mathbf{u}}) . \tag{10}$$

## 3. The Subset of Data (SoD) Approximation

Before we get started with the more sophisticated approximations, we mention as a baseline method the simplest possible sparse approximation (which doesn't fall inside our general scheme): use only a subset of the data (SoD). The computational complexity is reduced to $O(m^3)$, where $m < n$. We would not generally expect SoD to be a competitive method, since it would seem impossible (even with fairly redundant data and a good choice of the subset) to get a realistic picture of the

6. Note, that $Q_{\mathbf{a},\mathbf{b}}$ depends on $\mathbf{u}$ although this is not explicit in the notation.

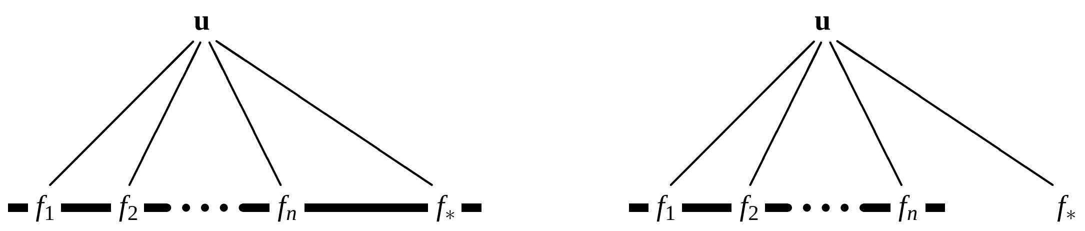

Figure 1: Graphical model of the relation between the inducing variables $\mathbf{u}$, the training latent functions values $\mathbf{f} = [f_1, \ldots, f_n]^\top$ and the test function value $f_*$. The thick horizontal line represents a set of fully connected nodes. The observations $y_1, \ldots, y_n, y_*$ (not shown) would dangle individually from the corresponding latent values, by way of the exact (factored) likelihood (5). **Left graph:** the fully connected graph corresponds to the case where no approximation is made to the full joint Gaussian process distribution between these variables. The inducing variables $\mathbf{u}$ are superfluous in this case, since all latent function values can communicate with all others. **Right graph:** assumption of *conditional independence* between training and test function values given $\mathbf{u}$. This gives rise to the separation between training and test conditionals from (8). Notice that having cut the communication path between training and test latent function values, information from $\mathbf{f}$ can only be transmitted to $f_*$ via the inducing variables $\mathbf{u}$.

uncertainties, when only a part of the training data is even considered. We include it here mostly as a baseline against which to compare better sparse approximations.

In Figure 5 top, left we see how the SoD method produces wide predictive distributions, when training on a randomly selected subset of 10 cases. A fair comparison to other methods would take into account that the computational complexity is independent of $n$ as opposed to other more advanced methods. These extra computational resources could be spent in a number of ways, e.g. larger $m$, or an active (rather than random) selection of the $m$ points. In this paper we will concentrate on understanding the theoretical foundations of the various approximations rather than investigating the necessary heuristics needed to turn the approximation schemes into actually practical algorithms.

## 4. The Subset of Regressors (SoR) Approximation

The Subset of Regressors (SoR) algorithm was given by Silverman (1985), and mentioned again by Wahba et al. (1999). It was then adapted by Smola and Bartlett (2001) to propose a sparse greedy approximation to Gaussian process regression. SoR models are finite linear-in-the-parameters models with a particular prior on the weights. For any input $\mathbf{x}_*$, the corresponding function value $f_*$ is given by:

$$f_* = K_{*,\mathbf{u}} \mathbf{w}_\mathbf{u} , \quad \text{with} \quad p(\mathbf{w}_\mathbf{u}) = \mathcal{N}(\mathbf{0}, K_{\mathbf{u},\mathbf{u}}^{-1}) , \tag{11}$$

where there is one weight associated to each inducing input in $X_\mathbf{u}$. Note that the covariance matrix for the prior on the weights is the *inverse* of that on $\mathbf{u}$, such that we recover the exact GP prior on $\mathbf{u}$, which is Gaussian with zero mean and covariance

$$\mathbf{u} \;=\; K_{\mathbf{u},\mathbf{u}}\,\mathbf{w_u} \;\;\Rightarrow\;\; \langle \mathbf{u}\,\mathbf{u}^\top \rangle \;=\; K_{\mathbf{u},\mathbf{u}} \langle \mathbf{w_u}\,\mathbf{w_u}^\top \rangle K_{\mathbf{u},\mathbf{u}} \;=\; K_{\mathbf{u},\mathbf{u}}\,. \tag{12}$$

Using the effective prior on $\mathbf{u}$ and the fact that $\mathbf{w_u} = K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}$ we can redefine the SoR model in an equivalent, more intuitive way:

$$\mathbf{f}_* \;=\; K_{*,\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u}\,, \quad \text{with} \quad \mathbf{u} \sim \mathcal{N}(\mathbf{0},\, K_{\mathbf{u},\mathbf{u}})\,. \tag{13}$$

We are now ready to integrate the SoR model in our unifying framework. Given that there is a *deterministic* relation between any $\mathbf{f}_*$ and $\mathbf{u}$, the approximate conditional distributions in the integral in eq. (8) are given by:

$$q_{\text{SoR}}(\mathbf{f}|\mathbf{u}) \;=\; \mathcal{N}(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\, \mathbf{0})\,, \quad \text{and} \quad q_{\text{SoR}}(\mathbf{f}_*|\mathbf{u}) \;=\; \mathcal{N}(K_{*,\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\, \mathbf{0})\,, \tag{14}$$

with zero conditional covariance, compare to (9). The effective prior implied by the SoR approximation is easily obtained from (8), giving

$$q_{\text{SoR}}(\mathbf{f},\mathbf{f}_*) \;=\; \mathcal{N}\Big(\mathbf{0},\, \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & Q_{*,*} \end{bmatrix}\Big)\,, \tag{15}$$

where we recall $Q_{\mathbf{a},\mathbf{b}} \triangleq K_{\mathbf{a},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} K_{\mathbf{u},\mathbf{b}}$. A more descriptive name for this method, would be the Deterministic Inducing Conditional (DIC) approximation. We see that this approximate prior is degenerate. There are only $m$ degrees of freedom in the model, which implies that only $m$ linearly independent functions can be drawn from the prior. The $m+1$-th one is a linear combination of the previous. For example, in a very low noise regime, the posterior could be severely constrained by only $m$ training cases.

The degeneracy of the prior causes unreasonable predictive distributions. Indeed, the approximate prior over functions is so restrictive, that given enough data only a very limited family of functions will be plausible under the posterior, leading to overconfident predictive variances. This is a general problem of finite linear models with small numbers of weights (for more details see Rasmussen and Quiñonero-Candela, 2005). Figure 5, top, right panel, illustrates the unreasonable predictive uncertainties of the SoR approximation on a toy dataset.[7]

The predictive distribution is obtained by using the SoR approximate prior (15) instead of the true prior in (4). For each algorithm we give two forms of the predictive distribution, one which is easy to interpret, and the other which is economical to compute with:

$$\begin{aligned} q_{\text{SoR}}(\mathbf{f}_*|\mathbf{y}) \;&=\; \mathcal{N}\big(Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\text{noise}} I)^{-1}\mathbf{y},\; Q_{*,*} - Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\text{noise}} I)^{-1} Q_{\mathbf{f},*}\big)\,, && \text{(16a)} \\ &=\; \mathcal{N}\big(\sigma^{-2} K_{*,\mathbf{u}} \Sigma K_{\mathbf{u},\mathbf{f}}\,\mathbf{y},\; K_{*,\mathbf{u}} \Sigma K_{\mathbf{u},*}\big)\,, && \text{(16b)} \end{aligned}$$

where we have defined $\Sigma = (\sigma^{-2} K_{\mathbf{u},\mathbf{f}} K_{\mathbf{f},\mathbf{u}} + K_{\mathbf{u},\mathbf{u}})^{-1}$. Equation (16a) is readily recognized as the regular prediction equation (6), except that the covariance $K$ has everywhere been replaced by $Q$, which was already suggested by (15). This corresponds to replacing the covariance function $k$ with $k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{u}) K_{\mathbf{u},\mathbf{u}}^{-1} k(\mathbf{u}, \mathbf{x}_j)$. The new covariance function has rank (at most) $m$. Thus we have the following

7. Wary of this fact, Smola and Bartlett (2001) propose using the predictive variances of the SoD, or a more accurate computationally costly alternative (more details are given by Quiñonero-Candela, 2004, Chapter 3).

**Remark 4** *The SoR approximation is equivalent to exact inference in the degenerate Gaussian process with covariance function* $k_{\text{SoR}}(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i, \mathbf{u}) K_{\mathbf{u},\mathbf{u}}^{-1} k(\mathbf{u}, \mathbf{x}_j)$.

The equivalent (16b) is computationally cheaper, and with (11) in mind, $\Sigma$ is the covariance of the posterior on the weights $\mathbf{w}_\mathbf{u}$. Note that as opposed to the subset of data method, all training cases are taken into account. The computational complexity is $O(nm^2)$ initially, and $O(m)$ and $O(m^2)$ per test case for the predictive mean and variance respectively.

## 5. The Deterministic Training Conditional (DTC) Approximation

Taking up ideas already contained in the work of Csató and Opper (2002), Seeger et al. (2003) recently proposed another sparse approximation to Gaussian process regression, which does not suffer from the nonsensical predictive uncertainties of the SoR approximation, but that interestingly leads to exactly the same predictive mean. Seeger et al. (2003), who called the method Projected Latent Variables (PLV), presented the method as relying on a *likelihood* approximation, based on the projection $\mathbf{f} = K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}$:

$$p(\mathbf{y}|\mathbf{f}) \simeq q(\mathbf{y}|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u},\, \sigma^2_{\text{noise}} I) \, . \tag{17}$$

The method has also been called the Projected Process Approximation (PPA) by Rasmussen and Williams (2006, Chapter 8). One way of obtaining an equivalent model is to retain the usual likelihood, but to impose a deterministic training conditional and the exact test conditional from eq. (9b)

$$q_{\text{DTC}}(\mathbf{f}|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u}, \mathbf{0}), \quad \text{and} \quad q_{\text{DTC}}(\mathbf{f}_*|\mathbf{u}) = p(\mathbf{f}_*|\mathbf{u}) \, . \tag{18}$$

This reformulation has the advantage of allowing us to stick to our view of exact inference (with exact likelihood) with approximate priors. Indeed, under this model the conditional distribution of $\mathbf{f}$ given $\mathbf{u}$ is identical to that of the SoR, given in the left of (14). A systematic name for this approximation is the Deterministic Training Conditional (DTC).

The fundamental difference with SoR is that DTC uses the exact test conditional (9b) instead of the deterministic relation between $\mathbf{f}_*$ and $\mathbf{u}$ of SoR. The joint prior implied by DTC is given by:

$$q_{\text{DTC}}(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}\right) , \tag{19}$$

which is surprisingly similar to the effective prior implied by the SoR approximation (15). The fundamental difference is that under the DTC approximation $\mathbf{f}_*$ has a prior variance of its own, given by $K_{*,*}$. This prior variance reverses the behaviour of the predictive uncertainties, and turns them into sensible ones, see Figure 5 for an illustration.

The predictive distribution is now given by:

$$\begin{aligned} q_{\text{DTC}}(\mathbf{f}_*|\mathbf{y}) &= \mathcal{N}(Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\text{noise}} I)^{-1} \mathbf{y},\, K_{*,*} - Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\text{noise}} I)^{-1} Q_{\mathbf{f},*} & \text{(20a)} \\ &= \mathcal{N}\big(\sigma^{-2} K_{*,\mathbf{u}} \Sigma K_{\mathbf{u},\mathbf{f}} \mathbf{y},\, K_{*,*} - Q_{*,*} + K_{*,\mathbf{u}} \Sigma K_{*,\mathbf{u}}^\top\big) \, , & \text{(20b)} \end{aligned}$$

where again we have defined $\Sigma = (\sigma^{-2} K_{\mathbf{u},\mathbf{f}} K_{\mathbf{f},\mathbf{u}} + K_{\mathbf{u},\mathbf{u}})^{-1}$ as in (16). The predictive mean for the DTC is identical to that of the SoR approximation (16), but the predictive variance replaces the $Q_{*,*}$ from SoR with $K_{*,*}$ (which is larger, since $K_{*,*} - Q_{*,*}$ is positive definite). This added term is the predictive variance of the posterior of $f_*$ conditioned on $\mathbf{u}$. It grows to the prior variance $K_{*,*}$ as $\mathbf{x}_*$ moves far from the inducing inputs in $X_\mathbf{u}$.

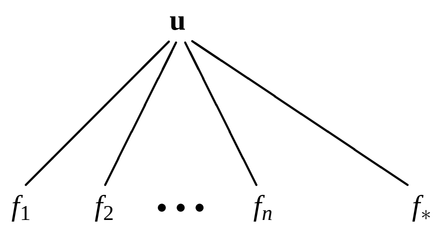

Figure 2: Graphical model for the FITC approximation. Compared to those in Figure 1, all edges between latent function values have been removed: the latent function values are conditionally fully independent given the inducing variables **u**. Although strictly speaking the SoR and DTC approximations could also be represented by this graph, note that both further assume a deterministic relation between **f** and **u**.

**Remark 5** *The only difference between the predictive distribution of DTC and SoR is the variance. The predictive variance of DTC is never smaller than that of SoR.*

Note, that since the covariances for training cases and test cases are computed differently, see (19), it follows that

**Remark 6** *The DTC approximation does not correspond exactly to a Gaussian process,*

as the covariance between latent values depends on whether they are considered training or test cases, violating consistency, see Definition 1. The computational complexity has the same order as for SoR.

## 6. The Fully Independent Training Conditional (FITC) Approximation

Recently Snelson and Ghahramani (2006) proposed another likelihood approximation to speed up Gaussian process regression, which they called Sparse Gaussian Processes using Pseudo-inputs (SGPP). While the DTC is based on the likelihood approximation given by (17), the SGPP proposes a more sophisticated likelihood approximation with a richer covariance

$$p(\mathbf{y}|\mathbf{f}) \simeq q(\mathbf{y}|\mathbf{u}) = \mathcal{N}(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u},\ \mathrm{diag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}] + \sigma^2_{\mathrm{noise}} I) \,, \tag{21}$$

where diag$[A]$ is a diagonal matrix whose elements match the diagonal of $A$. As we did in (18) for the DTC, we provide an alternative equivalent formulation called Fully Independent Training Conditional (FITC) based on the inducing conditionals:

$$q_{\mathrm{FITC}}(\mathbf{f}|\mathbf{u}) = \prod_{i=1}^{n} p(f_i|\mathbf{u}) = \mathcal{N}\big(K_{\mathbf{f},\mathbf{u}} K_{\mathbf{u},\mathbf{u}}^{-1} \mathbf{u},\ \mathrm{diag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}]\big) \,, \quad \text{and} \quad q_{\mathrm{FITC}}(f_*|\mathbf{u}) = p(f_*|\mathbf{u}) \,. \tag{22}$$

We see that as opposed to SoR and DTC, FITC does not impose a deterministic relation between **f** and **u**. Instead of ignoring the variance, FITC proposes an approximation to the training conditional distribution of **f** given **u** as a further independence assumption. In addition, the exact test conditional from (9b) is used in (22), although for reasons which will become clear towards the end of this section, we initially consider only a single test case, $f_*$. The corresponding graphical model is given in Figure 2. The effective prior implied by the FITC is given by

$$q_{\mathrm{FITC}}(\mathbf{f}, f_*) \;=\; \mathcal{N}\Big(\mathbf{0},\; \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} - \mathrm{diag}[Q_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}\Big)\,. \tag{23}$$

Note, that the sole difference between the DTC and FITC is that in the top left corner of the implied prior covariance, FITC replaces the approximate covariances of DTC by the exact ones on the diagonal. The predictive distribution is

$$\begin{aligned} q_{\mathrm{FITC}}(f_*|\mathbf{y}) \;&=\; \mathcal{N}\big(Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}}+\Lambda)^{-1}\mathbf{y},\; K_{*,*} - Q_{*,\mathbf{f}}(Q_{\mathbf{f},\mathbf{f}}+\Lambda)^{-1}Q_{\mathbf{f},*}\big) & \text{(24a)} \\ &=\; \mathcal{N}\big(K_{*,\mathbf{u}}\Sigma K_{\mathbf{u},\mathbf{f}}\Lambda^{-1}\mathbf{y},\; K_{*,*} - Q_{*,*} + K_{*,\mathbf{u}}\Sigma K_{\mathbf{u},*}\big)\,, & \text{(24b)} \end{aligned}$$

where we have defined $\Sigma = (K_{\mathbf{u},\mathbf{u}} + K_{\mathbf{u},\mathbf{f}}\Lambda^{-1}K_{\mathbf{f},\mathbf{u}})^{-1}$ and $\Lambda = \mathrm{diag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\mathrm{noise}}I]$. The computational complexity is identical to that of SoR and DTC.

So far we have only considered a single test case. There are two options for joint predictions, either 1) use the exact full test conditional from (9b), or 2) extend the additional factorizing assumption to the test conditional. Although Snelson and Ghahramani (2006) don't explicitly discuss joint predictions, it would seem that they probably intend the second option. Whereas the additional independence assumption for the test cases is not really necessary for computational reasons, it does affect the nature of the approximation. Under option 1) the training and test covariance are computed differently, and thus this does not correspond to our strict definition of a GP model, but

**Remark 7** *Iff the assumption of full independence is extended to the test conditional, the FITC approximation is equivalent to exact inference in a non-degenerate Gaussian process with covariance function* $k_{\mathrm{FIC}}(x_i,x_j) = k_{\mathrm{SoR}}(x_i,x_j) + \delta_{i,j}[k(x_i,x_j) - k_{\mathrm{SoR}}(x_i,x_j)]$,

where $\delta_{i,j}$ is Kronecker's delta. A logical name for the method where the conditionals (training and test) are always forced to be fully independent would be the Fully Independent Conditional (FIC) approximation. The effective prior implied by FIC is:

$$q_{\mathrm{FIC}}(\mathbf{f}, \mathbf{f}_*) \;=\; \mathcal{N}\Big(\mathbf{0},\; \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} - \mathrm{diag}[Q_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & Q_{*,*} - \mathrm{diag}[Q_{*,*} - K_{*,*}] \end{bmatrix}\Big)\,. \tag{25}$$

## 7. The Partially Independent Training Conditional (PITC) Approximation

In the previous section we saw how to improve the DTC approximation by approximating the training conditional with an independent distribution, i.e. one with a diagonal covariance matrix. In this section we will further improve the approximation (while remaining computationally attractive) by extending the training conditional to have a block diagonal covariance:

$$q_{\mathrm{PITC}}(\mathbf{f}|\mathbf{u}) \;=\; \mathcal{N}\big(K_{\mathbf{f},\mathbf{u}}K_{\mathbf{u},\mathbf{u}}^{-1}\mathbf{u},\; \mathrm{blockdiag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}]\big)\,, \quad \text{and} \quad q_{\mathrm{PITC}}(\mathbf{f}_*|\mathbf{u}) \;=\; p(\mathbf{f}_*|\mathbf{u})\,. \tag{26}$$

where blockdiag$[A]$ is a block diagonal matrix (where the blocking structure is not explicitly stated). We represent graphically the PITC approximation in Figure 3. Developing this analogously to the FITC approximation from the previous section, we get the joint prior

$$q_{\mathrm{PITC}}(\mathbf{f}, f_*) \;=\; \mathcal{N}\Big(\mathbf{0},\; \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} - \mathrm{blockdiag}[Q_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}\Big)\,, \tag{27}$$

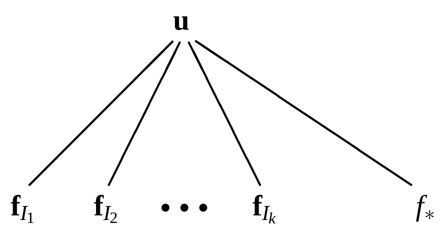

Figure 3: Graphical representation of the PITC approximation. The set of latent function values $\mathbf{f}_{I_i}$ indexed by the the set of indices $I_i$ is fully connected. The PITC differs from FITC (see graph in Fig. 2) in that conditional independence is now between the $k$ *groups* of training latent function values. This corresponds to the block diagonal approximation to the true training conditional given in (26).

and the predictive distribution is identical to (24), except for the alternative definition of $\Lambda = \text{blockdiag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\text{noise}} I]$. An identical expression was obtained by Schwaighofer and Tresp (2003, Sect. 3), developing from the original Bayesian committee machine (BCM) by Tresp (2000). The relationship to the FITC was pointed out by Lehel Csató. The BCM was originally proposed as a transductive learner (i.e. where the *test* inputs have to be known before training), and the inducing inputs $X_{\mathbf{u}}$ were chosen to be the test inputs. We discuss transduction in detail in the next section.

It is important to realize that the BCM proposes two orthogonal ideas: first, the block diagonal structure of the partially independent training conditional, and second setting the inducing inputs to be the test inputs. These two ideas can be used independently and in Section 8 we propose using the first without the second.

The computational complexity of the PITC approximation depends on the blocking structure imposed in (26). A reasonable choice, also recommended by Tresp (2000) may be to choose $k = n/m$ blocks, each of size $m \times m$. The computational complexity remains $O(nm^2)$. Since in the PITC model the covariance is computed differently for training and test cases

**Remark 8** *The PITC approximation does not correspond exactly to a Gaussian process.*

This is because computing covariances requires knowing whether points are from the training- or test-set, (27). One can obtain a Gaussian process from the PITC by extending the partial conditional independence assumption to the test conditional, as we did in Remark 7 for the FITC.

## 8. Transduction and Augmentation

The idea of transduction is that one should restrict the goal of learning to prediction on a pre-specified set of test cases, rather than trying to learn an entire function (induction) and then evaluate it at the test inputs. There may be no universally agreed upon definition of transduction. In this paper we use

**Definition 9** *Transduction occurs only if the predictive distribution depends on other test inputs.*

This operational definition excludes models for which there exist an equivalent inductive counterpart. According to this definition, it is irrelevant when the bulk of the computation takes place.

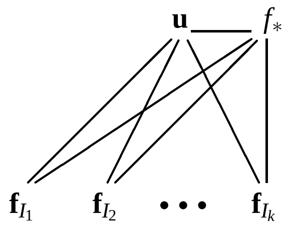

Figure 4: Two views on Augmentation. One view is to see that the test latent function value $f_*$ is now part of the inducing variables $\mathbf{u}$ and therefore has access to the training latent function values. An equivalent view is to consider that we have dropped the assumption of conditional independence between $f_*$ and the training latent function values. Even if $f_*$ has now direct access to each of the training $f_i$, these still need to go through $\mathbf{u}$ to talk to each other if they fall in conditionally independent blocks. We have in this figure decided to recycle the graph for PITC from Figure 3 to show that all approximations we have presented can be augmented, irrespective of what the approximation for the training conditional is.

There are several different possible motivations for transduction: 1) transduction is somehow easier than induction (Vapnik, 1995), 2) the test inputs may reveal important information, which should be used during training. This motivation drives models in semi-supervised learning (studied mostly in the context of classification) and 3) for approximate algorithms one may be able to limit the discrepancies of the approximation at the test points.

For exact GP models it seems that the first reason doesn't really apply. If you make predictions at the test points that are consistent with a GP, then it is trivial inside the GP framework to extend these to any other input points, and in effect we have done induction.

The second reason seems more interesting. However, in a standard GP setting, it is a consequence of the consistency property, see Remark 2, that predictions at one test input are independent of the location of any other test inputs. Therefore transduction can not be married with exact GPs:

**Remark 10** *Transduction can not occur in exact Gaussian process models.*

Whereas this holds for the usual setting of GPs, it could be different in non-standard situations where e.g. the covariance function depends on the empirical input densities.

Transduction can occur in the sparse approximation to GPs, by making the choice of inducing variables depend on the test inputs. The BCM from the previous section, where $X_{\mathbf{u}} = X_*$ (where $X_*$ are the test inputs) is an example of this. Since the inducing variables are connected to all other nodes (see Figure 3) we would expect the approximation to be good at $\mathbf{u} = \mathbf{f}_*$, which is what we care about for predictions, relating to reason 3) above. While this reasoning is sound, it is not necessarily a sufficient consideration for getting a good model. The model has to be able to simultaneously explain the training targets as well and if the choice of $\mathbf{u}$ makes this difficult, the posterior at the points of interest may be distorted. Thus, the choice of $\mathbf{u}$ should be governed by the ability to model the conditional of the latents given the inputs, and not solely by the density of the (test) inputs.

The main drawback of transduction is that by its nature it doesn't provide a predictive model in the way inductive models do. In the usual GP model one can do the bulk of the computation involved in the predictive distributions (e.g. matrix inversion) *before* seeing the test cases, enabling fast computation of test predictions.

It is interesting that whereas other methods spend much effort trying to optimize the inducing variables, the BCM simply uses the test set. The quality of the BCM approximation depends then on the particular location of the test inputs, upon which one usually does not have any control. We now see that there may be a better method, eliminating the drawback of transduction, namely use the PITC approximation, but choose the $\mathbf{u}$'s carefully (see Section 9), don't just use the test set.

## 8.1 Augmentation

An idea closely related to transduction, but not covered by our definition, is augmentation, which in contrast to transduction is done individually for each test case. Since in the previous sections, we haven't assumed anything about $\mathbf{u}$, we can simply augment the set of inducing variables by $f_*$ (i.e. have one additional inducing variable equal to the current test latent), and see what happens in the predictive distributions for the different methods. Let's first investigate the consequences for the test conditional from (9b). Note, the interpretation of the covariance matrix $K_{*,*} - Q_{*,*}$ was "the prior covariance minus the information which $\mathbf{u}$ provides about $f_*$". It is clear that the augmented $\mathbf{u}$ (with $f_*$) provides all possible information about $f_*$, and consequently $Q_{*,*} = K_{*,*}$. An equivalent view on augmentation is that the assumption of conditional independence between $f_*$ and $\mathbf{f}$ is dropped. This is seen trivially by adding edges between $f_*$ and the $f_i$ in the graphical model, Figure 4.

Augmentation was originally proposed by Rasmussen (2002), and applied in detail to the SoR with RBF covariance by Quiñonero-Candela (2004). Because the SoR is a finite linear model, and the basis functions are local (Gaussian bumps), the predictive distributions can be very misleading. For example, when making predictions far away from the center of any basis function, all basis functions have insignificant magnitudes, and the prediction (averaged over the posterior) will be close to zero, with very small error-bars; this is the opposite of the desired behaviour, where we would expect the error-bars to *grow* as we move away from the training cases. Here augmentation makes a particularly big difference turning the nonsensical predictive distribution into a reasonable one, by ensuring that there is always a basis function centered on the test case. Compare the non-augmented to the augmented SoR in Figure 5. An analogous Gaussian process based finite linear model that has recently been healed by augmentation is the relevance vector machine (Rasmussen and Quiñonero-Candela, 2005).

Although augmentation was initially proposed for a narrow set of circumstances, it is easily applied to any of the approximations discussed. Of course, augmentation doesn't make any sense for an exact, non-degenerate Gaussian process model (a GP with a covariance function that has a feature-space which is infinite dimensional, i.e. with basis functions *everywhere*).

**Remark 11** *A full non-degenerate Gaussian process cannot be augmented,*

since the corresponding $\mathbf{f}_*$ would already be connected to all other variables in the graphical model. But augmentation *does* make sense for sparse approximations to GPs.

The more general process view on augmentation has several advantages over the basis function view. It is not completely clear from the basis function view, which basis function should be used for augmentation. For example, Rasmussen and Quiñonero-Candela (2005) successfully apply augmentation using basis functions that have a zero contribution at the test location! In the process view however, it seems clear that one would chose the additional inducing variable to be $f_*$, to minimize the effects of the approximations.

Let us compute the effective prior for the *augmented* SoR. Given that $f_*$ is in the inducing set, the test conditional is not an approximation and we can rewrite the integral leading to the effective prior:

$$q_{\text{ASoR}}(\mathbf{f}_*, \mathbf{f}) \;=\; \int q_{\text{SoR}}(\mathbf{f}|\, f_*, \mathbf{u})\, p(f_*, \mathbf{u})\, \mathrm{d}\mathbf{u}\, . \tag{28}$$

It is interesting to notice that this is also the effective prior that would result from augmenting the DTC approximation, since $q_{\text{SoR}}(\mathbf{f}|\, f_*, \mathbf{u}) = q_{\text{DTC}}(\mathbf{f}|\, f_*, \mathbf{u})$.

**Remark 12** *Augmented SoR (ASoR) is equivalent to augmented DTC (ADTC).*

Augmented DTC only differs from DTC in the additional presence of $f_*$ among the inducing variables in the training conditional. We can only expect augmented DTC to be a more accurate approximation than DTC, since adding an additional inducing variable can only help capture information from $\mathbf{y}$. Therefore

**Remark 13** *DTC is a less accurate (but cheaper) approximation than augmented SoR.*

We saw previously in Section 5 that the DTC approximation does not suffer from the nonsensical predictive variances of the SoR. The equivalence between the augmented SoR and augmented DTC is another way of seeing how augmentation reverses the misbehaviour of SoR. The predictive distribution of the augmented SoR is obtained by adding $f_*$ to $\mathbf{u}$ in (20).

Prediction with an augmented sparse model comes at a higher computational cost, since now $f_*$ directly interacts with all of $\mathbf{f}$ and not just with $\mathbf{u}$. For each new test case, updating the augmented $\Sigma$ in the predictive equation (for example (20b) for DTC) implies computing the vector matrix product $K_{*,\mathbf{f}}K_{\mathbf{f},\mathbf{u}}$ with complexity $O(nm)$. This is clearly higher than the $O(m)$ for the mean, and $O(m^2)$ for the predictive distribution of all the non-augmented methods we have discussed.

Augmentation seems to be only really necessary for methods that make a severe approximation to the test conditional, like the SoR. For methods that make little or no approximation to the test conditional, it is difficult to predict the degree to which augmentation would help. However, one can see by giving $f_*$ access to all of the training latent function values in $\mathbf{f}$, one would expect augmentation to give less under-confident predictive distributions near the training data. Figure 5 clearly shows that augmented DTC (equivalent to augmented SoR) has a superior predictive distribution (both mean and variance) than standard DTC. Note however that in the figure we have purposely chosen a too short lengthscale to enhance visualization. Quantitatively, this superiority was experimentally assessed by Quiñonero-Candela (2004, Table 3.1). Augmentation hasn't been compared to the more advanced approximations FITC and PITC, and the figure would change in the more realistic scenario where the inducing inputs and hyperparameters are learnt (Snelson and Ghahramani, 2006).

Transductive methods like the BCM can be seen as joint augmentation, and one could potentially use it for any of the methods presented. It seems that the good performance of the BCM could essentially stem from augmentation, the presence of the *other* test inputs in the inducing set being probably of little benefit. Joint augmentation might bring some computational advantage, but won't change the scaling: note that augmenting $m$ times at a cost of $O(nm)$ apiece implies the same $O(nm^2)$ total cost as the jointly augmented BCM.

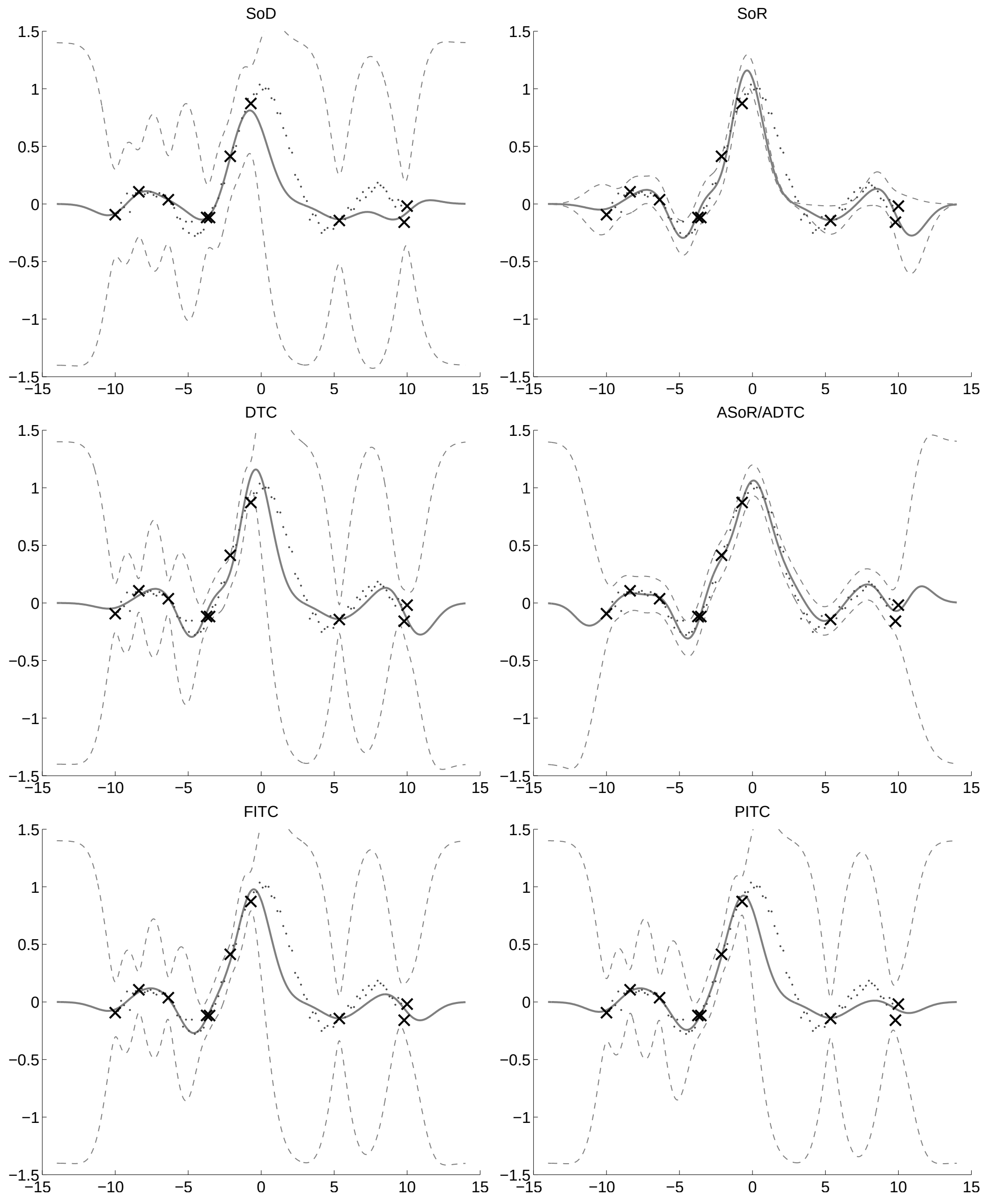

Figure 5: Toy example with identical covariance function and hyperparameters. The squared exponential covariance function is used, and a slightly too short lengthscale is chosen on purpose to emphasize the different behaviour of the predictive uncertainties. The dots are the training points, the crosses are the targets corresponding to the inducing inputs, randomly selected from the training set. The solid line is the mean of the predictive distribution, and the dotted lines show the 95% confidence interval of the predictions. Augmented DTC (ADTC) is equivalent to augmented SoR (ASoR), see Remark 12.

## 9. On the Choice of the Inducing Variables

We have until now assumed that the inducing inputs $X_{\mathbf{u}}$ were given. Traditionally, sparse models have very often been built upon a carefully chosen subset of the training inputs. This concept is probably best exemplified in the popular support vector machine (Cortes and Vapnik, 1995). In sparse Gaussian processes it has also been suggested to select the inducing inputs $X_{\mathbf{u}}$ from among the training inputs. Since this involves a prohibitive combinatorial optimization, greedy optimization approaches have been suggested using various selection criteria like online learning (Csató and Opper, 2002), greedy posterior maximization (Smola and Bartlett, 2001), maximum information gain (Seeger et al., 2003), matching pursuit (Keerthi and Chu, 2006), and probably more. As discussed in the previous section, selecting the inducing inputs from among the test inputs has also been considered in transductive settings. Recently, Snelson and Ghahramani (2006) have proposed to relax the constraint that the inducing variables must be a subset of training/test cases, turning the discrete selection problem into one of continuous optimization. One may hope that finding a good solution is easier in the continuous than the discrete case, although finding the global optimum is intractable in both cases. And perhaps the less restrictive choice can lead to better performance in very sparse models.

Which optimality criterion should be used to set the inducing inputs? Departing from a fully Bayesian treatment which would involve defining priors on $X_{\mathbf{u}}$, one could maximize the marginal likelihood (also called the evidence) with respect to $X_{\mathbf{u}}$, an approach also followed by Snelson and Ghahramani (2006). Each of the approximate methods proposed involves a different effective prior, and hence its own particular effective marginal likelihood conditioned on the inducing inputs

$$q(\mathbf{y}|X_{\mathbf{u}}) = \iint p(\mathbf{y}|\mathbf{f})\, q(\mathbf{f}|\mathbf{u})\, p(\mathbf{u}|X_{\mathbf{u}}) d\mathbf{u}\, d\mathbf{f} = \int p(\mathbf{y}|\mathbf{f})\, q(\mathbf{f}|X_{\mathbf{u}}) d\mathbf{f}\,, \tag{29}$$

which of course is independent of the test conditional. We have in the above equation explicitly conditioned on the inducing inputs $X_{\mathbf{u}}$. Using Gaussian identities, the effective marginal likelihood is very easily obtained by adding a ridge $\sigma^2_{\text{noise}} I$ (from the likelihood) to the covariance of effective prior on $\mathbf{f}$. Using the appropriate definitions of $\Lambda$, the log marginal likelihood becomes

$$\log q(\mathbf{y}|X_{\mathbf{u}}) \;=\; -\tfrac{1}{2}\log|Q_{\mathbf{f},\mathbf{f}} + \Lambda| - \tfrac{1}{2}\mathbf{y}^\top (Q_{\mathbf{f},\mathbf{f}} + \Lambda)^{-1}\mathbf{y} - \tfrac{n}{2}\log(2\pi)\,, \tag{30}$$

where $\Lambda_{\text{SoR}} = \Lambda_{\text{DTC}} = \sigma^2_{\text{noise}} I$, $\Lambda_{\text{FITC}} = \text{diag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}] + \sigma^2_{\text{noise}} I$, and $\Lambda_{\text{PITC}} = \text{blockdiag}[K_{\mathbf{f},\mathbf{f}} - Q_{\mathbf{f},\mathbf{f}}] + \sigma^2_{\text{noise}} I$. The computational cost of the marginal likelihood is $O(nm^2)$ for all methods, that of its gradient with respect to one element of $X_{\mathbf{u}}$ is $O(nm)$. This of course implies that the complexity of computing the gradient wrt. to the whole of $X_{\mathbf{u}}$ is $O(dnm^2)$, where $d$ is the dimension of the input space.

It has been proposed to maximize the effective posterior instead of the effective marginal likelihood (Smola and Bartlett, 2001). However this is potentially dangerous and can lead to overfitting. Maximizing the whole evidence instead is sound and comes at an identical computational cost (for a deeper analysis see Quiñonero-Candela, 2004, Sect. 3.3.5 and Fig. 3.2).

The marginal likelihood has traditionally been used to learn the hyperparameters of GPs in the non fully Bayesian treatment (see for example Williams and Rasmussen, 1996). For the sparse approximations presented here, once you are learning $X_{\mathbf{u}}$ it is straightforward to allow for learning hyperparameters (of the covariance function) during the same optimization, and there is no need to interleave optimization of $\mathbf{u}$ with learning of the hyperparameters as it has been proposed for example by Seeger et al. (2003).

## 10. Other Methods

In this section we briefly mention two approximations which don't fit in our unifying scheme, since one doesn't correspond to a proper probabilistic model, and the other one uses a particular construction for the covariance function, rather than allowing any general covariance function.

### 10.1 The Nyström Approximation

The Nyström Approximation for speeding up GP regression was originally proposed by Williams and Seeger (2001), and then questioned by Williams et al. (2002). Like SoR and DTC, the Nyström Approximation for GP regression approximates the prior covariance of $\mathbf{f}$ by $Q_{\mathbf{f},\mathbf{f}}$. However, unlike these methods, the Nyström Approximation is *not* based on a generative probabilistic model. The prior covariance between $f_*$ and $\mathbf{f}$ is taken to be exact, which is *inconsistent* with the prior covariance on $\mathbf{f}$:

$$q(\mathbf{f}, \mathbf{f}_*) = \mathcal{N}\Big(\mathbf{0}, \begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} & K_{\mathbf{f},*} \\ K_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}\Big) . \tag{31}$$

As a result we cannot derive this method from our unifying framework, nor represent it with a graphical model. Worse, the resulting prior covariance matrix is not even guaranteed to be positive definite, allowing the predictive variances to be negative. Notice that replacing $K_{\mathbf{f},*}$ by $Q_{\mathbf{f},*}$ in (31) is enough to make the prior covariance positive definite, and one obtains the DTC approximation.

**Remark 14** *The Nyström Approximation does not correspond to a well-formed probabilistic model.*

Ignoring any quibbles about positive definiteness, the predictive distribution of the Nyström Approximation is given by:

$$p(f_*|\mathbf{y}) = \mathcal{N}\big(K_{\mathbf{f},*}^\top [Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\text{noise}} I]^{-1} \mathbf{y},\, K_{*,*} - K_{\mathbf{f},*}^\top [Q_{\mathbf{f},\mathbf{f}} + \sigma^2_{\text{noise}} I]^{-1} K_{\mathbf{f},*}\big) , \tag{32}$$

but the predictive variance is not guaranteed to be positive. The computational cost is $O(nm^2)$.

### 10.2 The Relevance Vector Machine

The relevance vector machine, introduced by Tipping (2001), is a finite linear model with an independent Gaussian prior imposed on the weights. For any input $\mathbf{x}_*$, the corresponding function output is given by:

$$f_* = \boldsymbol{\phi}_* \mathbf{w} , \quad \text{with} \quad p(\mathbf{w}|A) = \mathcal{N}(0, A) , \tag{33}$$

where $\boldsymbol{\phi}_* = [\phi_1(\mathbf{x}), \ldots, \phi_m(\mathbf{x})]$ is the (row) vector of responses of the $m$ basis functions, and $A = \text{diag}(\alpha_1, \ldots, \alpha_m)$ is the diagonal matrix of joint prior precisions (inverse variances) of the weights. The $\alpha_i$ are learnt by maximizing the RVM evidence (obtained by also assuming Gaussian additive iid. noise, see (1)), and for the typical case of rich enough sets of basis functions many of the precisions go to infinity effectively pruning out the corresponding weights (for a very interesting analysis see Wipf et al., 2004). The RVM is thus a sparse method and the surviving basis functions are called *relevance vectors*.

Note that since the RVM is a finite linear model with Gaussian priors on the weights, it can be seen as a Gaussian process:

**Remark 15** *The RVM is equivalent to a degenerate Gaussian process with covariance function* $k_{\text{RVM}}(\mathbf{x}_i, \mathbf{x}_j) = \boldsymbol{\phi}_i A^{-1} \boldsymbol{\phi}_j^\top = \sum_{k=1}^m \alpha_k^{-1} \phi_k(\mathbf{x}_i)\, \phi_k(\mathbf{x}_j)$,

| Method | $q(\mathbf{f}_*\|\mathbf{u})$ | $q(\mathbf{f}\|\mathbf{u})$ | joint prior covariance | GP? |
|---|---|---|---|---|
| GP | exact | exact | $\begin{bmatrix} K_{\mathbf{f},\mathbf{f}} & K_{\mathbf{f},*} \\ K_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}$ | $\surd$ |
| SoR | determ. | determ. | $\begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & Q_{*,*} \end{bmatrix}$ | $\surd$ |
| DTC | exact | determ. | $\begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}$ | |
| FITC | (exact) | fully indep. | $\begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} - \mathrm{diag}[Q_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}$ | $(\surd)$ |
| PITC | exact | partially indep. | $\begin{bmatrix} Q_{\mathbf{f},\mathbf{f}} - \mathrm{blokdiag}[Q_{\mathbf{f},\mathbf{f}} - K_{\mathbf{f},\mathbf{f}}] & Q_{\mathbf{f},*} \\ Q_{*,\mathbf{f}} & K_{*,*} \end{bmatrix}$ | |

Table 1: Summary of the way approximations are built. All these methods are detailed in the previous sections. The initial cost and that of the mean and variance per test case are respectively $n^2$, $n$ and $n^2$ for the exact GP, and $nm^2$, $m$ and $m^2$ for all other methods. The "GP?" column indicates whether the approximation is equivalent to a GP. For FITC see Remark 7.

as was also pointed out by Tipping (2001, eq. (59)). Whereas all sparse approximations we have presented until now are totally independent of the choice of covariance function, for the RVM this choice is restricted to covariance functions that can be expressed as finite expansions in terms of some basis functions. Being degenerate GPs in exactly the same way as the SoR (presented in Section 4), the RVM does also suffer from unreasonable predictive variances. Rasmussen and Quiñonero-Candela (2005) show that the predictive distributions of RVMs can also be healed by augmentation, see Section 8. Once the $\alpha_i$ have been learnt, denoting by $m$ the number of surviving relevance vectors, the complexity of computing the predictive distribution of the RVM is $O(m)$ for mean and $O(m^2)$ for the variance.

RVMs are often used with radial basis functions centered on the training inputs. One potentially interesting extension to the RVM would be to *learn* the locations of the centers of the basis functions, in the same way as proposed by Snelson and Ghahramani (2006) for the FITC approximation, see Section 6. This is a curious reminiscence of learning the centers in RBF Networks.

## 11. Conclusions

We have provided a unifying framework for sparse approximations to Gaussian processes for regression. Our approach consists of two steps, first 1) we recast the approximation in terms of approximations to the prior, and second 2) we introduce inducing variables $\mathbf{u}$ and the idea of conditional independence given $\mathbf{u}$. We recover all existing sparse methods by making further simplifications of the covariances of the training and test conditionals, see Table 1 for a summary.

Previous methods were presented based on different approximation paradigms (e.g. likelihood approximations, projection methods, matrix approximations, minimization of Kullback-Leibler divergence, etc), making direct comparison difficult. Under our unifying view we deconstruct methods, making it clear which building blocks they are based upon. For example, the SGPP by Snelson and Ghahramani (2006) contains two ideas, 1) a likelihood approximation and 2) the idea of varying the inducing inputs continuously; these two ideas could easily be used independently, and incorporated in other methods. Similarly, the BCM by Tresp (2000) contains two independent ideas 1) a block diagonal assumption, and 2) the (transductive) idea of choosing the test inputs as the inducing variables. Finally we note that although all three ideas of 1) transductively setting $\mathbf{u} = \mathbf{f}_*$, 2) augmentation and 3) continuous optimization of $X_\mathbf{u}$ have been proposed in very specific settings, in fact they are completely general ideas, which can be applied to any of the approximation schemes considered.

We have ranked the approximation according to how close they are to the corresponding full GP. However, the performance in practical situations may not always follow this theoretical ranking since the approximations might exhibit properties (not present in the full GP) which may be particularly suitable for specific datasets. This may make the interpretation of empirical comparisons challenging. A further complication arises when adding the necessary heuristics for turning the theoretical constructs into practical algorithms. We have not described full algorithms in this paper, but are currently working on a detailed empirical study (in preparation, see also Rasmussen and Williams, 2006, chapter 8).

We note that the order of the computational complexity is identical for all the methods considered, $O(nm^2)$. This highlights that there is no computational excuse for using gross approximations, such as assuming deterministic relationships, in particular one should probably think twice before using SoR or even DTC. Although augmentation has attractive predictive properties, it is computationally expensive. It remains unclear whether augmentation could be beneficial on a fixed computational budget.

We have only considered the simpler case of regression in this paper, but sparseness is also commonly sought in classification settings. It should not be difficult to cast probabilistic approximation methods such as Expectation Propagation (EP) or the Laplace method (for a comparison, see Kuss and Rasmussen, 2005) into our unifying framework.

Our analysis suggests that a new interesting approximation would come from combining the best possible approximation (PITC) with the most powerful selection method for the inducing inputs. This would correspond to a non-transductive version of the BCM. We would evade the necessity of knowing the test set before doing the bulk of the computation, and we could hope to supersede the superior performance reported by Snelson and Ghahramani (2006) for very sparse approximations.

## Acknowledgments

Thanks to Neil Lawrence for arranging the 2005 Gaussian Process Round Table meeting in Sheffield, which provided much inspiration to this paper. Special thanks to Olivier Chapelle, Lehel Csató, Zoubin Ghahramani, Matthias Seeger, Ed Snelson and Chris Williams for helpful discussions, and to three anonymous reviewers. Both authors were supported by the German Research Council (DFG) through grant RA 1030/1. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778.

## Appendix A. Gaussian and Matrix Identities

In this appendix we provide identities used to manipulate matrices and Gaussian distributions throughout the paper. Let $\mathbf{x}$ and $\mathbf{y}$ be jointly Gaussian

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}\left( \begin{bmatrix} \boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y \end{bmatrix}, \begin{bmatrix} A & C \\ C^\top & B \end{bmatrix} \right) , \tag{34}$$

then the marginal and the conditional are given by

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, A) , \quad \text{and} \quad \mathbf{x}|\mathbf{y} \sim \mathcal{N}\big(\boldsymbol{\mu}_x + CB^{-1}(\mathbf{y} - \boldsymbol{\mu}_y), A - CB^{-1}C^\top\big) \tag{35}$$

Also, the product of a Gaussian in $\mathbf{x}$ with a Gaussian in a linear projection $P\mathbf{x}$ is again a Gaussian, although unnormalized

$$\mathcal{N}(\mathbf{x}|\mathbf{a},A)\,\mathcal{N}(P\mathbf{x}|\mathbf{b},B) \;=\; z_c\,\mathcal{N}(\mathbf{x}|\mathbf{c},C) , \tag{36}$$

where

$$C \;=\; \big(A^{-1} + P^\top B^{-1} P\big)^{-1} , \qquad c \;=\; C\big(A^{-1}\mathbf{a} + P^\top B^{-1}\mathbf{b}\big) .$$

The normalizing constant $z_c$ is gaussian in the means $\mathbf{a}$ and $\mathbf{b}$ of the two Gaussians:

$$z_c \;=\; (2\pi)^{-\frac{m}{2}} |B + PAP^\top|^{-\frac{1}{2}} \exp\Big( -\tfrac{1}{2}(\mathbf{b} - P\mathbf{a})^\top \big(B + PAP^\top\big)^{-1} (\mathbf{b} - P\mathbf{a})\Big) . \tag{37}$$

The matrix inversion lemma, also known as the Woodbury, Sherman & Morrison formula states that:

$$(Z + UWV^\top)^{-1} \;=\; Z^{-1} - Z^{-1}U(W^{-1} + V^\top Z^{-1} U)^{-1} V^\top Z^{-1} , \tag{38}$$

assuming the relevant inverses all exist. Here $Z$ is $n \times n$, $W$ is $m \times m$ and $U$ and $V$ are both of size $n \times m$; consequently if $Z^{-1}$ is known, and a low rank (ie. $m < n$) perturbation are made to $Z$ as in left hand side of eq. (38), considerable speedup can be achieved.

## References

Corinna Cortes and Vladimir Vapnik. Support-vector network. *Machine Learning*, 20(3):273–297, 1995.

Lehel Csató and Manfred Opper. Sparse online Gaussian processes. *Neural Computation*, 14(3): 641–669, 2002.

Sathiya Keerthi and Wei Chu. A Matching Pursuit approach to sparse Gaussian process regression. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, Cambridge, Massachussetts, 2006. The MIT Press.

Malte Kuss and Carl Edward Rasmussen. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, pages 1679–1704, 2005.

Joaquin Quiñonero-Candela. *Learning with Uncertainty – Gaussian Processes and Relevance Vector Machines*. PhD thesis, Technical University of Denmark, Lyngby, Denmark, 2004.

Carl Edward Rasmussen. Reduced rank Gaussian process learning. Technical report, Gatsby Computational Neuroscience Unit, UCL, 2002.

Carl Edward Rasmussen and Joaquin Quiñonero-Candela. Healing the relevance vector machine by augmentation. In *International Conference on Machine Learning*, 2005.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT press, 2006.

Anton Schwaighofer and Volker Tresp. Transductive and inductive methods for approximate Gaussian process regression. In Suzanna Becker, Sebastian Thrun, and Klaus Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 953–960, Cambridge, Massachussetts, 2003. The MIT Press.

Matthias Seeger, Christopher K. I. Williams, and Neil Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In Christopher M. Bishop and Brendan J. Frey, editors, *Ninth International Workshop on Artificial Intelligence and Statistics*. Society for Artificial Intelligence and Statistics, 2003.

Bernhard W. Silverman. Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *J. Roy. Stat. Soc. B*, 47(1):1–52, 1985. (with discussion).

Alexander J. Smola and Peter L. Bartlett. Sparse greedy Gaussian process regression. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 619–625, Cambridge, Massachussetts, 2001. The MIT Press.

Edward Snelson and Zoubin Ghahramani. Sparse Gaussian processes using pseudo-inputs. In Y. Weiss, B. Schölkopf, and J. Platt, editors, *Advances in Neural Information Processing Systems 18*, Cambridge, Massachussetts, 2006. The MIT Press.

Michael E. Tipping. Sparse Bayesian learning and the Relevance Vector Machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

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

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, 1995.

Grace Wahba, Xiwu Lin, Fangyu Gao, Dong Xiang, Ronald Klein, and Barbara Klein. The bias-variance tradeoff and the randomized GACV. In Michael S. Kerns, Sara A. Solla, and David A. Cohn, editors, *Advances in Neural Information Processing Systems 11*, pages 620–626, Cambridge, Massachussetts, 1999. The MIT Press.

Christopher K. I. Williams and Carl Edward Rasmussen. Gaussian processes for regression. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 514–520, Cambridge, Massachussetts, 1996. The MIT Press.

Christopher K. I. Williams, Carl Edward Rasmussen, Anton Schwaighofer, and Volker Tresp. Observations of the Nyström method for Gaussiam process prediction. Technical report, University of Edinburgh, Edinburgh, Scotland, 2002.

Christopher K. I. Williams and Mathias Seeger. Using the Nyström method to speed up kernel machines. In Todd K. Leen, Thomas G. Dietterich, and Volker Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 682–688, Cambridge, Massachussetts, 2001. The MIT Press.

David Wipf, Jason Palmer, and Bhaskar Rao. Perspectives on sparse Bayesian learning. In Sebastian Thrun, Lawrence Saul, and Bernhard Schölkopf, editors, *Advances in Neural Information Processing Systems 16*, Cambridge, Massachussetts, 2004. The MIT Press.