Efficient Nonparametric Bayesian Modelling with Sparse Gaussian Process Approximations

Matthias W. Seeger SEEGER@TUEBINGEN.MPG.DE
Max Planck Institute for Biological Cybernetics
Spemannstr. 38, Tübingen, Germany

Neil D. Lawrence NEIL@DCS.SHEF.AC.UK
Department of Computer Science, University of Sheffield
Regent Court, 211 Portobello St, Sheffield, UK

Ralf Herbrich RHERB@MICROSOFT.COM
Microsoft Research Ltd.
7 J J Thomson Ave, Cambridge, UK

Editor: ??

Abstract

Sparse approximations to Bayesian inference for nonparametric Gaussian Process models scale linearly in the number of training points, allowing for the application of powerful kernel-based models to large datasets. We present a general framework based on the *informative vector machine* (IVM) (Lawrence et al., 2003) and show how the complete Bayesian task of inference and learning of free hyperparameters can be performed in a practically efficient manner. Our framework allows for arbitrary likelihood and kernel functions, so that a large number of elementary models can be treated in a unified way. We present a range of experiments for our method applied to binary classification and regression tasks. Models based on a single latent function can be combined in order to address more complicated setups. We demonstrate this approach for a multi-way classification model.

Keywords: Gaussian Processes, Bayesian Learning, Sparse Approximations, Informative Vector Machine, Hyperparameter Learning, Multi-way Classification, Expectation Propagation

1. Introduction

Faced with the task of generalizing from noisy data whose source is imperfectly understood at best, *nonparametric smoothing* is a powerful and flexible way of analysis and prediction. The idea is to find a function which best satisfies a trade-off between a good *data fit* and exhibiting *regularity*, such as smoothness, periodicity, *etc.* When compared with the classical *parametric* approach to statistical modelling, namely to assume simple functional forms such as linearity or low-order polynomials, smoothing is often more flexible and does not suffer from bias which comes from assuming a fixed functional form. A probabilistic Bayesian formulation of smoothing with kernels is achieved by adopting the idea of *Gaussian process* (GP) priors (O'Hagan, 1978). By using a Bayesian viewpoint, we obtain a number of advantages over the traditional setup of finding a penalized best fit. For example, predictive uncertainties can be quantified, and free parameters can be adjusted automatically using

nonlinear optimization. Furthermore, the combination of basic models to address more complicated tasks is straightforward in principle.

A main problem with GP techniques is the unfavourable scaling with the number of datapoints, n, both in terms of computing (typically $O(n^3)$) and of memory usage ($O(n^2)$). Furthermore, even if learning can be accomplished, the final predictor requires at least $O(n)$ time for each prediction, and the whole (training) dataset has to be kept available for prediction. *Sparse approximations* to kernel methods (Tipping, 2001, Csató and Opper, 2002, Smola and Bartlett, 2001, Williams and Seeger, 2000, Lawrence et al., 2003, Tresp, 2000) are designed to overcome these scaling problems. For some $d \ll n$, learning with sparse approximations typically scales as $O(nd^2)$ time and $O(nd)$ memory, and the prediction cost scales with d only. In this paper we show how a complete Bayesian framework of inference and learning for general models featuring a single latent Gaussian process can be approximated efficiently (*i.e.* within the sparse constraints just stated). Apart from (Seeger, 2003), we are not aware of prior work on the same scale and generality. Earlier work has either concentrated on the particular GP single regression model with Gaussian noise (Seeger et al., 2003, Snelson and Ghahramani, 2006, Faul and Tipping, 2002), or has not dealt with the important problem of hyperparameter learning within the Bayesian context (Csató and Opper, 2002, Lawrence et al., 2003). The framework developed by Seeger (2003) differs from the one presented here, in that a variational approximation to the marginal likelihood is used for hyperparameter learning. We will give some arguments why our criterion here seems preferable. We also give an example of how a number of single process models can be combined into a multi-way classification model in a way that allows the inference techniques to be transferred from the single process case.

The structure of the paper is as follows. In Section 2, we introduce single Gaussian process models together with the main techniques we require in this paper, namely the expectation propagation technique and sparse approximations. In Section 3, we introduce the informative vector machine framework for conditional inference and learning in single GP models. In Section 4, we show how this framework can be used together with multivariate quadrature in order to obtain an IVM solution for multi-way classification. In Section 6, we present a range of experimental results. We close the paper with a discussion in Section 7.

The notation we use in this paper is explained in Appendix A.1.

2. Gaussian Process Models. Sparse Approximations. The Expectation Propagation Algorithm

In this Section, we introduce single Gaussian process models and show how inference and learning can be done in principle. The key problems of non-analytic computations and unfavourable scaling are addressed in turn by introducing sparse approximations and the expectation propagation technique for approximate inference.

2.1 Single Gaussian Process Models

Suppose we are given some data $D = \{(\boldsymbol{x}_i, y_i) \,|\, i = 1, \ldots, n\}$ assumed to be drawn independently and identically distributed (i.i.d.) from some unknown distribution over $\mathcal{X} \times \mathcal{Y}$. In order to predict y_* for further test points $\boldsymbol{x}_*$, we have to model the regularities that exist in D. In a single GP model, we do this by introducing a *latent function* $u : \mathcal{X} \to \mathbb{R}$ and

a likelihood distribution $P(y|u),\ y \in \mathcal{Y}, u \in \mathbb{R}$. We strive for a *conditional model* which explains the generation of the $\boldsymbol{y} = (y_i)_i$ given the inputs $\boldsymbol{X} = \{\boldsymbol{x}_i\}$,[1] and this is obtained by placing a *prior distribution* $P(u(\cdot))$ on the function u. Specifically, we assume that $u(\cdot)$ is *a priori* a *Gaussian process* (GP) with mean function 0 and covariance function $K(\boldsymbol{x}, \boldsymbol{x}')$. The covariance function is given by

$$K(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{E}[u(\boldsymbol{x})u(\boldsymbol{x}')]$$

and is positive semidefinite in the sense that for any $m \in \mathbb{N}$ and $\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_m$ we have that the matrix $\tilde{\boldsymbol{K}} = (K(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j))_{i,j}$ is positive semidefinite, *i.e.* $\boldsymbol{c}^T \tilde{\boldsymbol{K}} \boldsymbol{c} \geq 0$ for all $\boldsymbol{c}$. Intuitively, the GP is a consistent construction for inducing multivariate Gaussian random variables $\tilde{\boldsymbol{u}}$ with mean $\boldsymbol{0}$ and covariance matrix $\tilde{\boldsymbol{K}}$, for any finite set of input points. For more details on GPs for machine learning, the reader may consult (Seeger, 2004, Rasmussen and Williams, 2006).

Let us pause for a moment to see why this setup might achieve our goal of pushing inference from this model to prefer smooth, regular functions over very erratic ones. Many of the most frequently used covariance functions in practice are *isotropic*, in that $K(\boldsymbol{x}, \boldsymbol{x}') = K(\|\boldsymbol{x} - \boldsymbol{x}'\|)$. Here, $K(d)$ is a nonnegative even function which is monotonically decreasing on $[0, \infty)$. The correlation between $u(\boldsymbol{x})$ and $u(\boldsymbol{x}')$ is given by $K(d)/K(0)$, $d = \|\boldsymbol{x} - \boldsymbol{x}'\|$, which means that as $d \to 0$, the GP values become perfectly correlated. In mathematical terms, they converge to the same (random) value in mean-square.[2] Although the values $u(\boldsymbol{x})$ are all random and fluctuate as a whole, they are very strongly correlated at close points, which implies smoothness for functions drawn from the prior. We can also encode other properties of the latent function (which always hold in the mean-square sense) via the choice of the kernel, examples can be found in (Seeger, 2004).

Let $\boldsymbol{u} = (u(\boldsymbol{x}_i))_i \in \mathbb{R}^n$ be the variable of u evaluated at the datapoints. Note that $\boldsymbol{u} \sim P(\boldsymbol{u}) = N(\boldsymbol{0}, \boldsymbol{K})$ *a priori*, where $\boldsymbol{K} = (K(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j}$ is the data covariance (or kernel) matrix. The posterior distribution of $\boldsymbol{u}$ is obtained by Bayes' formula as $P(\boldsymbol{u}|D) \propto P(\boldsymbol{y}|\boldsymbol{u})P(\boldsymbol{u})$, where $P(\boldsymbol{y}|\boldsymbol{u}) = \prod_i P(y_i|u_i),\ u_i = u(\boldsymbol{x}_i)$. Since $P(y_i|u(\cdot)) = P(y_i|u_i)$, the posterior distribution of $u(\cdot)$ is defined through $P(u_*|D) = \int P(u_*|\boldsymbol{u})P(\boldsymbol{u}|D)\, d\boldsymbol{u}$ for points $\boldsymbol{x}_*$ and $u_* = u(\boldsymbol{x}_*)$. This posterior process is not Gaussian in general, but its mean and covariance function can be computed easily once we know mean and covariance matrix of the finite-dimensional posterior $P(\boldsymbol{u}|D)$. Furthermore, it is usually the case that prior and model come with so-called *hyperparameters* which have to be adjusted. For example, the likelihood may be a noise distribution whose variance is not known, or the covariance function K has free parameters. An empirical Bayesian method for adjusting such parameters $\boldsymbol{\theta}$ is to consider the *marginal likelihood*

$$P(D|\boldsymbol{\theta}) = \int P(D|\boldsymbol{u}, \boldsymbol{\theta})P(\boldsymbol{u}|\boldsymbol{\theta})\, d\boldsymbol{u} \tag{1}$$

1. All required input points will always be assumed to be known, and all distributions are always conditioned on all necessary input points. For simplicity, we do not make this explicit in the notation. We also use D and the targets $\boldsymbol{y}$ interchangeably.
2. One can construct GPs such that the convergence is with probability 1, namely the sample path $u(\cdot)$ is continuous almost surely.

which serves as a criterion to maximize for selecting $\boldsymbol{\theta}$. Note that again we make use of the fact that the function $u(\cdot)$ and the data $\boldsymbol{y}$ are independent, conditioned on the values $\boldsymbol{u}$, so that in order to obtain $P(D|\boldsymbol{\theta})$ we only need to integrate over the finite-dimensional $\boldsymbol{u}$.

In practice, there are at least two problems with this approach. First, the computations of the posterior $P(\boldsymbol{u}|D)$ and of $P(D|\boldsymbol{\theta})$ are typically analytically intractable. In the special case where the likelihood itself is Gaussian, the posterior and marginal likelihood are Gaussian and can be computed analytically. Much work on GP models for machine learning has concentrated on this case, which however is limited to regression models with Gaussian noise. If $P(y|u)$ is not Gaussian, a general strategy is to approximate $P(\boldsymbol{u}|D)$ by a Gaussian $Q(\boldsymbol{u}) = Q(\boldsymbol{u}; D)$. As mentioned above, in order to compute the predictive mean and covariance function of the posterior process $u(\cdot)|D$ (which is not Gaussian in general), all we need are the mean and covariance matrix of $P(\boldsymbol{u}|D)$, so we do not loose anything w.r.t. this task by making a Gaussian approximation. Several concrete methods along this line have been suggested. In this paper we use the *expectation propagation* (EP) technique (Minka, 2001) in order to compute $Q(\boldsymbol{u})$ and approximate $P(D|\boldsymbol{\theta})$. This method is introduced briefly in Section 2.3. Empirically, the EP technique seems to be more accurate than other variational or saddle-point approximations (Kuss and Rasmussen, 2005).

Second, even if $P(y|u)$ is Gaussian, the computation of $P(\boldsymbol{u}|D)$ scales as $O(n^3)$ with the number n of datapoints, which is prohibitive for large n. If $P(\boldsymbol{u}|D)$ is not tractable, methods for finding an approximate Gaussian $Q(\boldsymbol{u})$ scale as $O(n^3)$ as well. Sparse approximations are a general way of overcoming this scaling behaviour at the expense of more approximations. A general framework for sparse approximations has been given by Seeger (2003) and will be reviewed in Section 2.2.

2.2 Sparse Approximations

We are interested in a general way of approximating the posterior computation and representation and the marginal likelihood computation for the single GP models introduced in Section 2.1. Such a framework has been developed by Seeger (2003), and we will review some aspects here. Recall that $P(\boldsymbol{u}|D) \propto P(D|\boldsymbol{u})P(\boldsymbol{u})$, and the normalization constant is the marginal likelihood. If $P(D|\boldsymbol{u})$ is not Gaussian, it is approximated by a Gaussian function of $\boldsymbol{u}$ using a method such as EP (see Section 2.3). For the moment, we assume that $P(D|\boldsymbol{u})$ is Gaussian. We begin by developing a characterization of sparse GP approximations. Let us define some desirable characteristics of such techniques. First of all, we would like to obtain a predictive approximation $P(u_*|D)$ which is itself a proper Gaussian process. This desideratum already outrules several known methods such as subset of regressors (Silverman, 1985) or the relevance vector machine (Tipping, 2001). These have the serious flaw that predictive variance far from the training data is drastically underestimated. In fact, we restrict the procedure to modify the posterior $P(\boldsymbol{u}_J|D)$ on a finite set of input points, the corresponding outputs being denoted by $\boldsymbol{u}_J$. Exact inference uses $\boldsymbol{u}_J = \boldsymbol{u}$, the selected input points being the training set, but we allow for other choices. If $Q(\boldsymbol{u}_J|D)$ approximates $P(\boldsymbol{u}_J|D)$, the predictive process must be $Q(u_*|D) = \int P(u_*|\boldsymbol{u}_J)Q(\boldsymbol{u}_J|D)\, d\boldsymbol{u}_J$, where $P(u_*|\boldsymbol{u}_J)$ comes from the GP prior. Note that $Q(\boldsymbol{u}_J|D)$ must be a proper Gaussian. We do not know of any GP method resulting in a proper predictive GP, yet not conforming to this assumption. Next, for each prediction at a test point $\boldsymbol{x}_*$, we only allow for the evaluation

of $O(d)$ kernel values, namely for $K(\boldsymbol{x}_*, \tilde{\boldsymbol{x}}_i)$ with d fixed "active" points $\tilde{\boldsymbol{x}}_i$. A few methods have been proposed which do not adhere to this constraint, in that they allow for test points themselves to become active points (Tresp, 2000, Rasmussen and Quinonero Candela, 2005). While our reasoning here can be extended to these cases, we do not do so here. In this paper, we will be interested in the case that the active set is part of the training set, which seems reasonable given that the full method is supported on the training points and that observations y_i are available at these points only. However, our reasoning here is not restricted to this case. Denote the active variables by $\boldsymbol{u}_I = (u(\tilde{\boldsymbol{x}}_i))_i \in \mathbb{R}^d$. One can show (see Appendix A.2) that any sparse approximation conforming to our assumptions can be seen as *likelihood approximation*: the likelihood function $P(D|\boldsymbol{u})$ is effectively replaced by some Gaussian function $Q(D|\boldsymbol{u}_I)$ depending on $\boldsymbol{u}_I$ only. Note that $Q(D|\boldsymbol{u}_I)$ is not a distribution over D, and may be degenerate and unnormalized as Gaussian function. It is important to stress that most such methods do *not* strive for a functional approximation of $P(D|\boldsymbol{u})$ by some $Q(D|\boldsymbol{u}_I)$ w.r.t. some norm independent of the GP prior P. As such, the term "likelihood approximation" may be slightly misleading and has lead to suggestions to approximate the prior instead (Quinonero Candela and Rasmussen, 2005). However, the definition in (Seeger, 2003) states that $Q(D|\boldsymbol{u}_I)$ should in general be a nonnegative function of some process values $\boldsymbol{u}_I$ such that the *posterior* $P(u(\cdot)|D)$ is approximated well by $Q(u(\cdot)|D) \propto Q(D|\boldsymbol{u}_I)P(u(\cdot))$. The term "likelihood approximation" simply means that we get from $P(u(\cdot)|D)$ to $Q(u(\cdot)|D)$ by formally replacing the likelihood by $Q(D|\boldsymbol{u}_I)$.

Suppose now that $P(D|\boldsymbol{u}) = N^U(\boldsymbol{u}|\boldsymbol{b}, \boldsymbol{\Pi})$, where $\boldsymbol{\Pi}$ is diagonal. In this paper, the active set is part of the training set, *i.e.* $I \subset \{1, \ldots, n\}$. How shall we choose $Q(D|\boldsymbol{u}_I)$? The simplest way is to discard the factors corresponding to $i \notin I$, resulting in $Q(D|\boldsymbol{u}_I) = N^U(\boldsymbol{u}_I|\boldsymbol{b}_I, \boldsymbol{\Pi}_I)$. Note that just as the full likelihood $P(D|\boldsymbol{u})$, this approximation factorizes w.r.t. the variables u_i, $i \in I$. Also note that there are only $2d$ parameters $\boldsymbol{b}_I, \boldsymbol{\Pi}_I$. Adopting this likelihood approximation leads to the *informative vector machine* (IVM) (Lawrence et al., 2003, Seeger, 2003). In this paper we restrict ourselves to the IVM approximation, noting that it has clear computational advantages over other non-factorizing approximations, yet is also typically less accurate.

A more accurate likelihood approximation is obtained as $Q(D|\boldsymbol{u}_I) = N^U(\mathrm{E}_P[\boldsymbol{u}|\boldsymbol{u}_I]|\boldsymbol{b}, \boldsymbol{\Pi})$, where $\mathrm{E}_P[\boldsymbol{u}|\boldsymbol{u}_I] = \boldsymbol{K}_{\cdot,I}\boldsymbol{K}_I^{-1}\boldsymbol{u}_I$ is the conditional mean under the prior $P(\boldsymbol{u})$. This approximation has been used in (Csató and Opper, 2002, Seeger et al., 2003, Seeger, 2003) and can be shown to be optimal w.r.t. some relative entropy argument. Snelson and Ghahramani (2006) note that this likelihood approximation tends to underestimate predictive variance close to the training points. If $\boldsymbol{\lambda} = \mathrm{diag}(\boldsymbol{K} - \boldsymbol{K}_{\cdot,I}\boldsymbol{K}_I^{-1}\boldsymbol{K}_{I,\cdot})$, the likelihood approximation they propose has the form $N^U(\mathrm{E}_P[\boldsymbol{u}|\boldsymbol{u}_I]|\boldsymbol{z}, (\boldsymbol{\Pi}^{-1} + \mathrm{diag}\,\boldsymbol{\lambda})^{-1})$ for some $\boldsymbol{z}$, which (as compared to the previous suggestion) adds some variance except at the active points. This suggestion is optimal under a different relative entropy argument (at least for regression with Gaussian noise). The likelihood approximations of (Seeger et al., 2003, Snelson and Ghahramani, 2006) are more accurate than the factorized one of the IVM, but they also come at significant additional costs. Even in the case of regression with Gaussian noise, at least twice the amount of memory is needed. In case of a non-Gaussian likelihood, it is necessary to estimate the $2\,n$ free parameters of the likelihood approximation, rather than only $2\,d$ for the IVM. Strictly speaking, they need to be updated after each inclusion of a pattern into I. Moreover, if the active set I is determined using greedy forward selection (as

is done for the IVM, see Section 3), it becomes prohibitive to score all remaining points as candidates for each inclusion, while such a complete screening is possible for the IVM. We also cannot afford to compute the same criteria as used with the IVM for many candidates, but have to apply further approximations. Seeger (2003) discusses these issues in detail. We do not consider these more elaborate, costly, and accurate likelihood approximations in this paper, but an extension of our framework to these cases is straightforward in principle and will be addressed in future work.

2.3 Expectation Propagation

The *expectation propagation* (EP) scheme (Minka, 2001) is a general framework for approximate inference and learning in probabilistic graphical models. The basic idea and the application to GP models has been suggested already by Opper and Winther (2000) under the name of *adaptive TAP* approximation. We will only give an intuitive overview here, and refer to (Minka, 2001, Seeger, 2003, 2005) for details.

Suppose we are given the GP model introduced in Section 2.1, with likelihood factors $t_i(u_i) = P(y_i|u_i)$ (also called *sites* here), and recall that we would like to approximate the posterior $P(\boldsymbol{u}|D)$ and the marginal likelihood $P(D|\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ collects hyperparameters of the covariance function and the sites. We have already noted that a Gaussian approximation $Q(\boldsymbol{u})$ of $P(\boldsymbol{u}|D)$ leads to the fact that the approximate posterior process $u(\cdot)|D$ is Gaussian, so predictions at test points can be computed easily. EP is a way of obtaining such an approximation, by setting

$$Q(\boldsymbol{u}) \propto P(\boldsymbol{u}) \prod_{i=1}^{n} \tilde{t}_i(u_i|b_i, \pi_i). \tag{2}$$

Here, $\tilde{t}_i(u_i|b_i, \pi_i) = N^U(u_i|b_i, \pi_i)$ has the form of an (unnormalized) Gaussian (see Appendix A.1), and the *site parameters* $\boldsymbol{b} = (b_i)_i$, $\boldsymbol{\Pi} = \mathrm{diag}(\pi_i)_i$ have to be chosen such that $Q(\boldsymbol{u})$ is a proper Gaussian. EP starts with $\boldsymbol{b} = \boldsymbol{0}$, $\boldsymbol{\Pi} = \boldsymbol{0}$, *i.e.* $Q = P$, then updates the site parameters iteratively. For some i, the intuition behind an EP update is to remove the approximate contribution of i to Q and replace it by the real one, given by the true site $t_i(u_i)$. The resulting distribution is then projected back to a Gaussian Q' which is taken to be the new Q. Let $Q_{\backslash i}(\boldsymbol{u}) \propto Q(\boldsymbol{u})\tilde{t}_i(u_i)^{-1}$ be the cavity distribution (where $\tilde{t}_i(u_i)$ is removed), and define the tilted distribution $\hat{P}_i(\boldsymbol{u}) \propto t_i(u_i)Q_{\backslash i}(\boldsymbol{u})$. In a sense, $\hat{P}_i$ is a version of Q where the approximate effect of i is replaced by the true one. $\hat{P}_i$ is not Gaussian, but the fact that only a single non-Gaussian site is involved, this site depending on a single u_i only, allows us to compute mean and covariance of $\hat{P}_i$. Furthermore, we can choose b_i and π_i uniquely such that the new $Q(\boldsymbol{u})'$ has the same mean and covariance as $\hat{P}_i$. The latter computation is termed *moment matching*, another view is that the new Q' is the closest Gaussian to $\hat{P}_i$ in terms of relative entropy $\mathrm{D}[\hat{P}_i \,\|\, \cdot]$. This view and the fact that $\hat{P}_i$ is obtained from the old Q by multiplying with some $f(u_i)$ only, implies that the EP update, *i.e.* the re-computation of b_i, π_i requires knowledge of the current posterior *marginal* $Q(u_i)$ only. Details about the EP update are given in Appendix B.1. In practice, updates are run until convergence in $\boldsymbol{b}$, $\boldsymbol{\Pi}$. Convergence to some stationary point is not guaranteed in

general, although on GP models with a log-concave likelihood (see Appendix B.1) we have never experienced divergence.[3]

Note the similarity in form between EP and methods of approximate inference by belief propagation. EP uses current local posterior information in the form of the marginal $Q(u_i)$ together with local evidence information in form of the true site $t_i(u_i)$ in order to update the posterior belief. While the update seems local at first site, in that only the single b_i, π_i are updated, it actually has a global reach by affecting all other marginals $Q(u_j)$. In sparsely structured graphical networks, these marginals can be updated by message passing along the graph structure. However, if EP is applied to a GP model, the inverse prior covariance matrix $\boldsymbol{K}^{-1}$ is a dense matrix, so the model cannot be represented by a sparse graph. Updating the marginals $Q(u_j)$ after some EP update is a global operation which requires $O(n^2)$ in general.

EP comes with an approximation to the marginal likelihood $P(D|\boldsymbol{\theta})$ which works as follows. The approximation of $P(\boldsymbol{u}|D)$ by $Q(\boldsymbol{u})$ is done on the basis of matching moments of first and second order between Q and the $\hat{P}_i$ in turn. In order to approximate the normalization constant for the posterior, we simply extend this idea to matching the moments of zero order as well. Namely, we use the explicitly normalized site approximations $C_i\tilde{t}_i(u_i)$. If $Q_{\backslash i}(u_i)$ is the marginal cavity distribution, we require that the cavity expectation of the true site t_i and the site approximation $C_i\tilde{t}_i$ are the same:

$$Z_i = \mathrm{E}_{\backslash i}[t_i(u_i)] = \mathrm{E}_{\backslash i}[C_i\tilde{t}_i(u_i)] = C_i\tilde{Z}_i.$$

In practice, we run EP updates until the site parameters b_i, π_i converge, then do a final sweep over all sites in order to compute the C_i factors. We now obtain the approximate marginal likelihood by replacing $t_i(u_i)$ by $C_i\tilde{t}_i(u_i)$:

$$Q(D|\boldsymbol{\theta}) = \exp\left(\sum_i \log C_i + \Phi[Q] - \Phi[P]\right), \quad \Phi[N(\boldsymbol{\mu}, \boldsymbol{\Sigma})] = \frac{1}{2}\log|2\pi\boldsymbol{\Sigma}| + \frac{1}{2}\boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}. \quad (3)$$

Note that $\Phi[P]$ is the log partition function of the Gaussian P when written in exponential family form. In practice, we minimize $-\log Q(D|\boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$. Importantly, we can also compute the gradient of this criterion w.r.t. $\boldsymbol{\theta}$. It is shown by Seeger (2005) how this can be done by making use of the fixed point conditions of EP, namely that moments up to second order of $\hat{P}_i$ and Q do match for all i. The validity of these conditions lead to cancellations of terms depending on the mapping of $\boldsymbol{\theta}$ to $\boldsymbol{b}$, $\boldsymbol{\Pi}$ which in general could not be computed (see Section 3.3 for more on this point).

3. The Informative Vector Machine

In this Section we introduce the *informative vector machine* (IVM) framework (Lawrence et al., 2003) which combines EM and sparse approximations to achieve efficient inference and learning in single GP models. There are a multitude of such models, including univariate regression (with Gaussian or non-Gaussian noise), binary classification, ordered categorial, *etc.*. In fact, any *generalized linear model* (GLIM) (McCullach and Nelder, 1983) which

3. Divergence in such cases is usually due to numerical instabilities of the updates of the Q representation, but as shown in Appendix B.1 this is not a principled problem for log-concave likelihoods.

comes with a single linear function, gives rise to a single GP model if the linear function is replaced by the latent process. The IVM framework applies to all of these models.

The key features of the IVM as opposed to related approximations is the usage of a factorized likelihood approximation (see Section 2.2), *i.e.* the simplest and most efficient choice. This fact allows us to devise a particular representation of the posterior belief (the distribution $Q(\boldsymbol{u})$) which keeps the full set of marginals $Q(u_j)$ up-to-date at any time, at the total cost of only $O(nd^2)$. This is important because it allows to use powerful forward selection criteria based on these marginals, thus to screen *all* remaining points for usefulness as members in I during *each* inclusion. We do not know of any other sparse approximation framework (apart from the simple, yet often unsatisfactory, heuristic of choosing I completely at random, see Williams and Seeger (2001)) for which this can be done.

3.1 The IVM Representation. Approximate Inference

The Gaussian posterior approximation $Q(\boldsymbol{u})$ implied by EP has the form (Eq. 2). In the terminology of Section 2.2, $N^U(\boldsymbol{u}|\boldsymbol{b}, \boldsymbol{\Pi}) = \prod_{i=1}^n \tilde{t}_i(u_i)$ is a likelihood approximation to $P(D|\boldsymbol{u})$. A sparse (efficient) approximation requires that the likelihood approximation is a function of $\boldsymbol{u}_I$ only, where $I \subset \{1, \ldots, n\}$ is the active set of size d. Within the IVM scheme, this is achieved naturally by simply having $b_i = \pi_i = 0$ for all $i \notin I$, so the corresponding site approximations $\tilde{t}_i$ are constant and drop out. Thus, while a typical application of EP to GP inference requires updates being run until convergence of the site parameters, we restrict ourselves to selecting $i \in I$ in a data-driven (yet still efficient) manner, and limit EP updates to these *active* i. In this Section, we develop a representation for $Q(\boldsymbol{u})$ which satisfies the following criteria:

- Upon each inclusion of some j into I, the representation can be updated in a numerically stable and computationally efficiently manner, *i.e.* in $O(nd)$ time, requiring the evaluation of a single new column of the kernel matrix $\boldsymbol{K}$.
- The complete set of marginals $Q(u_j),\ j = 1, \ldots, n$ is a part of the representation.

We show how these objectives can be met with a representation of size $O(nd)$. Note that the requirement of having all marginals $Q(u_j)$ accessible at all times is what causes the cost of $O(nd)$ for each inclusion and the dominating size of the representation.[4] On the other hand, we need the marginals in order to perform an informed forward selection of I, namely so that we can screen all remaining points for each inclusion.

We simplify notation by denoting the current *active* site parameters by $\boldsymbol{b} \in \mathbb{R}^d$, $\boldsymbol{\Pi} \in \mathbb{R}^{d,d}$, where $\boldsymbol{\Pi}$ is diagonal. We also assume that the diagonal entries are nonnegative. In general applications of EP, negative $\boldsymbol{\Pi}$ elements are allowed, but may lead to numerical instabilities of the updates. For models we are interested in here, one can show that negative elements cannot occur (see Appendix B.1). The covariance matrix $\boldsymbol{A}$ of $Q(\boldsymbol{u})$ is

$$\begin{aligned} &\boldsymbol{A} = \left(\boldsymbol{K}^{-1} + \boldsymbol{I}_{\cdot,I}\boldsymbol{\Pi}\boldsymbol{I}_{I,\cdot}\right)^{-1} = \boldsymbol{K} - \boldsymbol{M}\boldsymbol{M}^T, \quad \boldsymbol{M} = \boldsymbol{K}_{\cdot,I}\boldsymbol{\Pi}^{1/2}\boldsymbol{L}^{-T}, \\ &\boldsymbol{B} = \boldsymbol{I} + \boldsymbol{\Pi}^{1/2}\boldsymbol{K}_I\boldsymbol{\Pi}^{1/2} = \boldsymbol{L}\boldsymbol{L}^T, \end{aligned} \tag{4}$$

4. We need a single matrix of size $n \times d$.

where $\boldsymbol{L} \in \mathbb{R}^{d,d}$ is the lower-triangular Cholesky factor of the symmetric positive definite $\boldsymbol{B}$. The validity of this representation follows from the Sherman-Morrison-Woodbury formula (Press et al., 1992). Note that this particular form is superior to most other ones suggested in the context of sparse approximations on the basis of numerical stability. $\boldsymbol{B}$ is typically well-conditioned, and the Cholesky decomposition is known to be stable (as opposed to matrix inversions which should be avoided whenever possible).

The stability of EP in the context of the IVM (or full GP inference) hinges on characteristics of the likelihood functions $t_i(u_i)$. We show in Appendix B.1 that IVM updates are numerically stable if the kernel matrix is not poorly conditioned, and the $t_i(u_i)$ are *log-concave* functions of u_i, *i.e.* $\log t_i(u_i)$ are concave.[5] The latter property is true for most frequently used likelihoods in Statistics. The most immediate consequence of using a log-concave likelihood together with a Gaussian prior is that the posterior distribution is unimodal. All exponential families (Gaussian, Gamma, Poisson, *etc*) have log-concave densities, the logit and probit classification models are log-concave as well (see Appendix B.1), as are their multivariate variants (see Appendix D). For regression models, very heavy-tailed noise models may fail to be log-concave, examples are the Student-t or the Cauchy distribution. In a sense, the Laplace distribution $P(y_i|u_i) \propto \exp(-|y_i - u_i|)$ is log-concave with the heaviest tails. Since log-concavity implies unimodality, multimodel distributions (such as Gaussian mixtures) are not log-concave either. The IVM framework has been used with multimodal sites (Lawrence and Jordan, 2005), but special care has to be taken to ensure stability.[6] We also conjecture (but do not have a proof) that the quality of approximation of the IVM is better for log-concave sites than for others. If the t_i are not log-concave, running full EP to convergence might still produce a good approximation, but this is expected to require several iterations over all sites, because the information of some of the t_i may be contradictory and has to be weighted carefully. For the IVM, we visit each site at most once, and most of the sites are not visited at all. This risky strategy seems somewhat less dangerous in the log-concave case.

The matrix $\boldsymbol{M} \in \mathbb{R}^{n,d}$ is called the *stub matrix*, it dominates the size requirements of the representation, and its update is $O(nd)$. The row $\boldsymbol{M}_{j,\cdot}^T$ of $\boldsymbol{M}$ is called stub of j, in that it contains all the information we need to store for j in order to be able to update the marginal $Q(u_j)$ efficiently after an inclusion. From (Eq. 4) we see that if $\boldsymbol{a} = \operatorname{diag} \boldsymbol{A}$, then $\boldsymbol{a} = \operatorname{diag} \boldsymbol{K} - \operatorname{diag} \boldsymbol{M}\boldsymbol{M}^T$. Furthermore, the mean $\boldsymbol{h}$ of $Q(\boldsymbol{u})$ can be written as

$$\boldsymbol{h} = \boldsymbol{M}\boldsymbol{\beta}, \quad \boldsymbol{\beta} = \boldsymbol{L}^{-1}\boldsymbol{\Pi}^{-1/2}\boldsymbol{b}.$$

We see that the maintenance of $\boldsymbol{M}$ in the representation is necessary for being able to update $\boldsymbol{h}$, $\boldsymbol{a}$ efficiently, and these are the required parameters of the marginals $Q(u_j)$. To conclude, the IVM representation consists of $\boldsymbol{b}$, $\boldsymbol{\pi} = \operatorname{diag} \boldsymbol{\Pi}$, $\boldsymbol{L}$, $\boldsymbol{M}$, $\boldsymbol{\beta}$, $\boldsymbol{h}$, $\boldsymbol{a}$. The update after an inclusion into I is detailed in Appendix C.1. Note that the dominating cost is

5. It has been observed empirically that log-concavity in the GP model case seems to have other beneficial consequences. For example, EP on the full GP model seems to converge reliably in this case. However, we do not know whether convergence of EP can be *guaranteed* under these assumptions.
6. Interestingly, IVM seems to behave more robustly than full EP with sites which are not log-concave. Possible reasons are that IVM does not run EP updates to convergence, but invokes them only once for each active site, and that troublesome patterns (w.r.t. inclusion into I) are also deemed uninformative by the selection criterion.

a $O(nd)$ matrix-vector multiplication which can be done efficiently by publicly available highly optimized matrix code (such as the ATLAS implementation of BLAS).

In order to predict u_* and y_* for some test point $\boldsymbol{x}_*$, we note that $Q(u_*|D) = \int P(u_*|\boldsymbol{u})Q(\boldsymbol{u})\,d\boldsymbol{u}$. The predictive distribution is therefore Gaussian, $Q(u_*|D) = N(u_*|\mu_*, \sigma_*^2)$, where

$$\mu_* = \boldsymbol{\beta}^T \boldsymbol{m}_*, \quad \sigma_*^2 = K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \|\boldsymbol{m}_*\|^2, \quad \boldsymbol{m}_* = \boldsymbol{L}^{-1}\boldsymbol{\Pi}^{1/2}(K(\boldsymbol{x}_i, \boldsymbol{x}_*))_{i\in I}.$$

Note that a prediction requires d kernel evaluations only, and the scaling does not depend on n, but only on d. If we only need to evaluate μ_*, we can also precompute $\boldsymbol{p} = \boldsymbol{\Pi}^{1/2}\boldsymbol{L}^{-T}\boldsymbol{\beta}$, after which μ_* can be computed in $O(d)$. We also have $Q(y_*|D) = \mathrm{E}[P(y_*|u_*)]$ with the expectation over $Q(u_*|D)$. Desired statistics of $Q(y_*|D)$ may in general be computed using Gaussian quadrature (see Appendix B.2).

It is important to clarify a point which is frequently misunderstood in the context of IVM. The IVM likelihood approximation does not simply mean that we throw away a large part of the data, namely the points not in I. This would be the case if we did the selection of I without screening all the datapoints, but in case of the IVM we actually spend the dominating effort to *screen all datapoints for every single inclusion*, so the selection of I is very dependent on *all* data. Readers may recognize a similarity with the support vectors in the support vector machine (SVM) (Schölkopf and Smola, 2002). In both architectures, only a subset of the data has to be available at prediction time, but the subset itself cannot be computed without looking at all the data. A very important *difference* between IVM and SVM is that for the former, we can *control* the degree of sparsity by explicitly choosing d, this not possible for the SVM (the choice of $\boldsymbol{\theta}$ has an indirect effect on sparsity in the SVM).

3.2 Greedy Forward Selection. The IVM Algorithm

The IVM representation developed in Section 3.1 keeps all posterior marginals $Q(u_j)$ accessible at all times (through $\boldsymbol{h}$, $\boldsymbol{a}$). The reason for this is that we intend to use this information in order to screen points $j \notin I$ as candidates for inclusion into I. For example, the marginal variance a_j quantifies the remaining uncertainty in the mean prediction h_j for u_j at pattern $\boldsymbol{x}_j$. Rather than inventing new heuristics, we can tap the field of *optimal design* (or *active learning*), where appropriate criteria for a very related task have been well-studied. In active learning, we are given a current posterior $Q(\boldsymbol{u})$ and a pool of candidate patterns $\boldsymbol{x}_j$ *without* any knowledge of the corresponding u_j or y_j. We have to select a subset of the patterns which are subsequently labeled (*i.e.* we obtain the y_j), and the goal is to select this set (of a fixed size) in order to gain the most knowledge about $u(\cdot)$ from the additional data. In the sequential variant, this subset is of size 1. The sequential approach can always be seen as approximation to the batch variant, in that a subset is selected one pattern at a time. This approximation is called *greedy forward selection*. Our situation is slightly different and in fact somewhat easier than active learning, in that we can access all y_j in the same way as the $\boldsymbol{x}_j$. Typical criteria used in sequential active learning have the form $c(\boldsymbol{x}_j) = \mathrm{E}_Q[f(y_j, u_j, \boldsymbol{x}_j)]$, *i.e.* the label y_j is predicted using the current belief $Q(\boldsymbol{u})$. If we know y_j (as in our case here), we obtain a stronger variant of the criterion by plugging in the true y_j instead of its prediction: $\tilde{c}(\boldsymbol{x}_j, y_j) = \mathrm{E}_Q[f(y_j, u_j, \boldsymbol{x}_j)]$, where E_Q is over u_j only.

Once we have specified a criterion, we select I by successively screening all remaining j and picking the one minimizing the criterion for the next inclusion. We update the IVM representation, and continue until I has reached the desired maximum size.[7] The selection criterion used by Lawrence et al. (2003) was the differential entropy $H[Q'(u_j)] - H[Q(u_j)]$, we selected the pattern for which the predictive variance a_j shrinks most with inclusion. In this paper, we focus on the *information gain score*

$$\Delta_j = -\mathrm{D}[Q'(u_j) \,\|\, Q(u_j)] = -\frac{1}{2}\left(\log m_j + m_j^{-1} + a_j^{-1}(h'_j - h_j)^2 - 1\right), \quad m_j = 1 + a_j\pi'_j, \tag{5}$$

where Q' is the posterior *after* inclusion of j into I. Note that this formulation is valid for $j \notin I$, thus $b_j = \pi_j = 0$. As opposed to the differential entropy, Δ_j does depend on the posterior mean change as well. For example, in the case of regression with Gaussian noise, the differential entropy score does not even depend on the target y_j, and in general the dependence on the actual target is not strong. Note that given h_j, a_j, computing Δ_j requires a single EP update which is $O(1)$ typically, so scoring all remaining patterns is $O(n)$. Note that we can also write $\Delta_j = -\mathrm{D}[Q'(\boldsymbol{u}) \,\|\, Q(\boldsymbol{u})]$, because $Q'(\boldsymbol{u}_{\setminus j}|u_j) = Q(\boldsymbol{u}_{\setminus j}|u_j)$ if j is included. The fact that Δ_j can be computed based on the marginal $Q(u_j)$ only is *not* an additional approximation.

We can now give a schematic overview of the IVM algorithm for approximate inference (Algorithm 1), given fixed hyperparameters $\boldsymbol{\theta}$. This basic scheme will be embedded below into a procedure for learning $\boldsymbol{\theta}$ (Section 3.3), and we give an example of how to use it as subroutine in more complicated models (Section 4).

Algorithm 1 Basic IVM Algorithm

Require: Desired sparsity $d \ll n$, threshold $\varepsilon > 0$ (numerical stability).
  $I = \emptyset$, $\boldsymbol{b} = \boldsymbol{0}$, $\boldsymbol{\Pi} = \mathrm{diag}(\boldsymbol{0})$, $\boldsymbol{a} = \mathrm{diag}\,\boldsymbol{K}$, $\boldsymbol{h} = \boldsymbol{0}$, $J = \{1, \ldots, n\}$.
  repeat
    for $j \in J$ **do**
      Compute Δ_j (Eq. 5).
    end for
    $i = \mathrm{argmax}_{j \in J}\{\Delta_j \,|\, \pi'_j > \varepsilon\}$
    Here, π'_j, b'_j are computed by an EP update (Appendix B.1).
    Site updates: $\pi_i \leftarrow \pi'_i$, $b_i \leftarrow b'_i$.
    Update representation as shown in Appendix C.1, notably $\boldsymbol{L}$, $\boldsymbol{M}$, $\boldsymbol{a}$, $\boldsymbol{h}$.
    $I \leftarrow I \cup \{i\}$, $J \leftarrow J \setminus \{i\}$.
  until $|I| = d$

It is important to note that rather than running EP on all site parameters until convergence, we leave most of the site parameters at 0 (otherwise we would not obtain a sparse IVM approximation), and we update the parameters for $i \in I$ only once, at the time they are included.

7. The maximum size of I is not a statistical variable, but a parameter of the algorithm. There is no sensible way of selecting an "appropriate size", since we expect that a larger size works better in general. The size must be chosen based on feasibility considerations.

We finally discuss a simple extension of the basic scheme which is useful if the storage of the complete $\boldsymbol{M}$ is not possible or will lead to performance degradation. In Algorithm 1, the selection set J of candidates for each inclusion is chosen to be $J = \{1, \ldots, n\} \setminus I$. We only really need $\boldsymbol{M}_{J,\cdot}$, so if we restrict J to a smaller set, we require less memory and update time. The problem is that it is not efficient to have J *grow* at a later stage: the update of $\boldsymbol{M}_{J,\cdot}$ when J is extended by j (say) is as expensive as having j in J from the beginning. We use the following strategy which shrinks J monotonically. We set some upper bound B on the total number of $\boldsymbol{M}_{J,\cdot}$ entries, so that $|J|d \leq B$ at all times. J remains constant over blocks of k inclusions.[8] At the beginning, $J = \{1, \ldots, n\}$. After each k inclusions, we pick a new size m' for J'. J is first replaced by $J \setminus I$. If J is still larger than m', we keep a certain fraction ρ of entries in J, namely the ones which achieved the best (lowest) scores. The remaining $1 - \rho$ part of J' is sampled at random from the rest of J. We re-arrange $\boldsymbol{M}_{J',\cdot}$ and continue. This procedure is called *randomized greedy selection*, a variant was used by Lawrence et al. (2003). The importance of randomized selection is to be able to apply the IVM approximation for active set sizes d even if nd is beyond our memory capacities. In light of the greedy nature of the selection of I, we expect that the most "important" points are included fairly early, which justifies the successive shrinkage of J during later stages.

3.2.1 Regression with Gaussian Noise

Let us illustrate our general abstract framework for the simple case of univariate regression with Gaussian noise. In this case, $P(y|u) = N(y|u, \sigma^2)$ is already Gaussian, so the EP approximation is not required. We do not have to write special code for this case, however. At the point of inclusion of j (say), we simply set $b_j = y_j\sigma^{-2}$, $\pi_j = \sigma^{-2}$. By doing so, we have $t_j(u_j) = \tilde{t}_j(u_j)$, *i.e.* site and site approximation are identical. Therefore, $Q(\boldsymbol{u})$ is identical to the true posterior $P(\boldsymbol{u}|\boldsymbol{y}_I)$. We need to stress again that this does not mean that the IVM blindly throws away much of the data, because the choice of I depends on all of $\boldsymbol{y}$.

We have that

$$\Delta_j = -\frac{1}{2}\left(\log m_j + m_j^{-1} + a_j\sigma^{-4}m_j^{-2}(y_j - h_j)^2 - 1\right), \quad m_j = 1 + a_j\sigma^{-2}$$

for the information gain score. In this example, we can see that the information gain score depends on the predictive mean h_j and the true target y_j, in that everything else equal, patterns with larger distance $|y_j - h_j|$ are preferred. This makes sense, as we are looking for patterns which come as "largest surprise" to the current model. As opposed to this, the differential entropy score used by Lawrence et al. (2003) does not depend on y_j.

3.3 The Marginal Likelihood Approximation

Recall from Section 2.1 that the marginal likelihood $P(D|\boldsymbol{\theta})$ (Eq. 1) can be used to select appropriate values for the hyperparameters $\boldsymbol{\theta}$ which maximize $P(D|\boldsymbol{\theta})$. This is an *empirical Bayesian* procedure. While the correct Bayesian way to deal with $\boldsymbol{\theta}$ would be to integrate them out, in practice the maximization of an approximation of $P(D|\boldsymbol{\theta})$ is often a useful surrogate. Justifications of this procedure can be found in (MacKay, 2003).

8. Changing J involves re-arrangements of the matrix $\boldsymbol{M}_{J,\cdot}$ which are themselves $O(nd)$, so should not be done after each inclusion.

We noted in Section 2.3 that an approximation to the marginal likelihood can be obtained within the EP framework (Eq. 3). Details about this approximate criterion and its gradient are given by Seeger (2005). In this Section, we show how to compute the criterion $\phi(\boldsymbol{\theta}) = -\log Q(D|\boldsymbol{\theta})$ within the IVM framework, and how to obtain an approximation to the gradient $\nabla_{\boldsymbol{\theta}}\phi$. Importantly this is possible within the same $O(nd^2)$ time constraint as the inference itself, in fact the latter will still dominate the cost. The details are given in Appendix C.2. ϕ is given in Eq. 11. This criterion is novel to our knowledge. Seeger (2003) considers learning $\boldsymbol{\theta}$ for non-regression models, but using a different variational criterion.

In order to efficiently optimize $\boldsymbol{\theta}$, we also need the gradient $\nabla_{\boldsymbol{\theta}}\phi$. First of all, the active set I depends on $\boldsymbol{\theta}$, yet this dependence has to be ignored for the gradient computation. However, even for fixed I, the computation of the gradient $\nabla_{\boldsymbol{\theta}}\phi$ requires us to know Jacobians such as $\nabla_{\boldsymbol{\theta}}\boldsymbol{b}$, $\nabla_{\boldsymbol{\theta}}\boldsymbol{\pi}$, and these cannot be computed analytically. Seeger (2005) shows that the problematic terms do actually cancel out if the EP algorithm is applied to full GP inference (without a sparse approximation). The reason is that the following fixed point condition holds for the site parameters once EP converges on them: $\hat{P}_i$ and Q have the same mean and covariance for all i. This is clearly not the case for the IVM approximation, where the site parameters for $i \notin I$ are kept at 0, and even for $i \in I$ we set the parameters only once. Note that for other more expensive sparse approximations, such as the ones mentioned in Section 2.2, the EP fixed point conditions do actually hold (for fixed I), and the exact gradient can be computed. We leave this as a direction for future work, we are not aware of work in that direction for non-Gaussian likelihood (Seeger (2003) computes the exact gradient for a different variational approximation to the marginal likelihood). In the case of IVM, we employ an approximation to the gradient, which is developed in Appendix C.2, and is given in Eq. 14 and Eq. 12.

We note that criterion and gradient computation can be unified with the further approximation of randomized greedy selection (see Section 3.2). Recall that this strategy ends up maintaining only the part $\boldsymbol{M}_{J,\cdot}$ of the dominating stub matrix, where $J \subset \{1, \ldots, n\} \setminus I$ is a selection index. Ideally, J contains patterns which carry most additional information, given $Q(\boldsymbol{u})$ and the current active set I, and our selection strategy is triggered towards that goal. Therefore, we may replace the parts in the marginal likelihood criterion summing over $i \notin I$ by the corresponding subparts summing over $i \in J$ only. In fact, the derivation of criterion and gradient in Appendix C.2 have been written in terms of J already, so they can be used directly with a selection index.

Our plan is to minimize ϕ w.r.t. $\boldsymbol{\theta}$, using some gradient-based optimizer which employs the conditional inference scheme of Section 3.2 as subroutine. We note that this is not a standard nonlinear optimization problem. First, we can only compute the gradient assuming that the active set I does not depend on the current $\boldsymbol{\theta}$, although strictly speaking it does. The same is true for the selection index J if a randomized greedy selection strategy is used (see Section 3.2). Next, even if these dependencies are ignored, we cannot compute the exact gradient $\nabla_{\boldsymbol{\theta}}\phi$ in general, because the EP fixed point conditions do not hold for most patterns i. Still, we observe empirically that the approximate gradient derived in Appendix C.2 is very useful for descending on ϕ. Our optimization strategy is similar to the one suggested by Seeger (2003). It is iterative, where each iteration step is either *major* or *minor*. Each step results in the computation of ϕ and a $\nabla_{\boldsymbol{\theta}}\phi$ approximation, followed by an update of $\boldsymbol{\theta}$. For a minor step, we assume that I (and J) are given, in fact they are retained from the

previous step. The IVM representation of Section 3.1 is computed directly, which scales as $O(nd^2)$, but in practice is much faster than doing conditional inference with a re-selection of I (and J). We then compute ϕ and the gradient based on the representation. In a major step, we start by calling the conditional inference subroutine of Section 3.2 from scratch, thereby re-selecting I (and J). We then do the same as during a minor step.

We face an interesting trade-off in the overall optimization. In practice, minor steps are much more efficient than major steps (although both are $O(nd^2)$). Furthermore, doing a major step may introduce discontinuities into the criterion curve $\phi(\boldsymbol{\theta})$. On the other hand, running many minor steps in between major ones is risky, because the IVM approximation as such, without the forward selection of I, is not a very good approximation to the full posterior, in the sense that the former is more prone to overfitting. Only a fairly frequent reselection of I with points deemed unusual under the current setting ensures a proper weight on complexity control. Our strategy is to use major steps to compute search directions, along which we descend using line searches with minor steps. We can also do major steps only for every k-th search direction, $k > 1$. With $k = 1$ and short line searches, we are on the safe side, but such a strategy requires many expensive major mode steps. Some researchers recommended selecting I early and sticking with it, only updating the hyperparameters later on, mainly because this strategy is easier to implement. Based on our experimental experience, we find that this strategy leads to suboptimal results in many cases.

Even within the inner optimization consisting of minor mode steps only, *i.e.* keeping I fixed, we have several options. Denoting the site parameters by $\boldsymbol{s}$, we note that ϕ is really a function $\phi(\boldsymbol{\theta}, \boldsymbol{s})$. The gradient given by Eq. 14 and Eq. 12 is $\nabla_{\boldsymbol{\theta}} \phi(\boldsymbol{\theta}, \boldsymbol{s})$ for fixed $\boldsymbol{s}$, neglecting the fact that $\boldsymbol{s}$ should really depend on $\boldsymbol{\theta}$. Namely, for fixed I we can define $\boldsymbol{s}$ as mapping of $\boldsymbol{\theta}$ through the site updates done in the ordering determined by I. Two strategies for the inner optimization are as follows. We can keep $\boldsymbol{s}$ constant throughout, updating it only together with I in major steps. The advantage is simplicity, since criterion and gradient are compatible and standard code can be used for the inner loops. On the other hand, the wrong criterion is minimized during inner loops. We can also update $\boldsymbol{s}$ with each criterion evaluation in the inner loops. This requires special "robust" line search code which neglects gradient information, but the correct criterion is minimized (for fixed I). A hybrid is possible as well, in that $\boldsymbol{s}$ is kept constant during line searches, but is recomputed for each new inner loop search direction. It is likely that an optimal strategy depends on the problem at hand and the resources we want to spend, and some fine-tuning is typically required for finding the best option. We present some experiments along these lines in Section 6.1.1.

A final note on the stability and converge properties of this method. Any type of "convergence" to high accuracy cannot be expected, even if I (and J) are fixed at some point. In practice, the best we can hope for are final small-scale oscillations around some point $\boldsymbol{\theta}$. However, at the moment we cannot provide a general convergence proof even guaranteeing such a limit behaviour. Snelson and Ghahramani (2006) propose a sparse approximation for GP regression with Gaussian noise, where the active input points are optimized over together with $\boldsymbol{\theta}$, always converging to a local minimum of their criterion. The advantages and drawbacks of this approach as compared to ours are discussed in Section 7.1.

4. Multi-Way Classification

In Section 3 we developed the IVM representation, conditional inference and learning techniques which can be applied to any single GP model. However, a range of important models do not fall into this category, because they require more than a single process to be represented. A natural approach to inference and learning in such a model is to approximate these tasks in a way which allows us to solve them using the single GP IVM framework as principal *subroutine*. In this Section, rather than trying to formalize another general framework for multi-process models, we concentrate on the concrete example of *multi-way classification*. Our approach, however, is general and may apply to other models as well. For example, it has been used with a multi-output regression model by Teh et al. (2005).

4.1 The Model. Inference Approximation

Suppose now that $\mathcal{Y} = \{1, \ldots, C\}$ for $C > 2$. The task therefore is multi-way classification with C classes. A common approach is to allocate a latent variable $u_c \in \mathbb{R}$ for every class, and to define a likelihood $P(y|\boldsymbol{u})$, $\boldsymbol{u} \in \mathbb{R}^C$. We will define a concrete classification likelihood below. We now require C latent processes $u_c(\cdot)$, which we assume *a priori* to be independent GPs with covariance functions $K^{(c)}$. A sensible likelihood will *couple* the classes, in that $P(y|\boldsymbol{u})$ does *not* factorize as a function of the u_c. For such a likelihood, we observe that the posterior processes $u_c(\cdot)$ are not independent anymore.

How can we implement sparse inference and learning for the multi-way GP model? A simple idea is to treat the model in the single GP context, by extending $\boldsymbol{x} \to (\boldsymbol{x}, c)$, $c = 1, \ldots, C$, with the joint covariance function

$$K((\boldsymbol{x}, c), (\boldsymbol{x}', c')) = \mathrm{I}_{\{c=c'\}} K^{(c)}(\boldsymbol{x}, \boldsymbol{x}').$$

The problem with this approach is that for n datapoints and a fixed degree of sparsity d/n, the IVM framework has a complexity of $O(C^3 n d^2)$, thus scales cubically in C, and we need $O(C^2 n d)$ memory. Such scaling is prohibitive. In fact, there is a very simple heuristic for dealing with the multi-way problem. Namely, we can learn C predictors independently, each of which discriminates one of the classes against the union of all others. This *one-against-rest* heuristic scales as $O(C n d^2)$ in the IVM framework. Since it works usually quite well in practice, it seems hard to justify a superlinear scaling in C of a proper multi-way method towards practitioners.

A linear scaling in C can be obtained by making the *factorization assumption* that the processes $u_c(\cdot)$ are independent under the approximate posterior Q:

$$Q(\boldsymbol{u}(\cdot)) = \prod_{c=1}^{C} Q_c(u_c(\cdot)).$$

Such factorization assumptions are fairly commonly used in variational approximations. While couplings in the true posterior are not represented in the factorized Q, the parameters of the Q_c factors are in general influenced by the presence of couplings. Seeger and Jordan (2004b) give a method which represents posterior couplings explicitly, but this is technically much more complicated than the factorized method presented here, and in fact does not lead to an improvement in predictions.

Under this factorization assumption, it becomes clear how the single GP IVM framework can be applied. Namely, each Q_c is represented by an IVM representation (see Section 3.1), where we allow for different active sets I_c of potentially different sizes d_c. The overall representation size is $O(n \sum_c d_c^2)$. Again, we can apply EP (see Section 2.3) in order to deal with the non-Gaussian sites $t_i(\boldsymbol{u}_i) = P(y_i|\boldsymbol{u}_i)$, $\boldsymbol{u}_i \in \mathbb{R}^C$. The EP updates work essentially in the same way as before, only that now we have $\boldsymbol{u}_i \in \mathbb{R}^C$ (recall Appendix B.1 for details of the univariate case). Site parameters have the form $\boldsymbol{b}_i, \boldsymbol{\pi}_i \in \mathbb{R}^C$ now, and $\tilde{t}_i(\boldsymbol{u}_i) = N^U(\boldsymbol{b}_i, \operatorname{diag} \boldsymbol{\pi}_i)$. Note that the site approximations have to be factorized, due to the independence assumptions for the Q_c. The cavity distribution $Q_{\backslash i}(\boldsymbol{u}_i)$ is factorized as well. For the moment matching, we require expectations of the form

$$\mathrm{E}_{\backslash i}\left[u_c^k t_i(\boldsymbol{u}_i)\right], \quad k = 0, 1, 2. \tag{6}$$

Note that $t_i(\boldsymbol{u}_i)$ does *not* factorize, therefore the tilted distribution $\hat{P}_i$ does not factorize either. However, because we need to retain independence of the Q_c, we have to project $\hat{P}_i$ onto a *factorized Gaussian* $Q'(\boldsymbol{u}_i)$, therefore all we need are means and variances of $\hat{P}_i$. If we formulate the projection as a relative entropy minimization problem onto factorized Gaussians, we obtain the update formulae

$$\pi_{j,c} = ({a'_{j,c}}^{-1} - a_{j,c}^{-1})_+, \quad b_{j,c} = a_{j,c}^{-1}(h'_{j,c} - h_{j,c}) + \pi_{j,c} h'_{j,c}, \tag{7}$$

where $h'_{j,c}, a'_{j,c}$ are mean and variance of $\hat{P}_j(u_{j,c})$, and $(x)_+ = x\mathrm{I}_{\{x>0\}}$. The quadrature part is challenging in general for all but small C. General rules such as Gaussian quadrature scale exponentially in C (see Appendix B.2). For a particular likelihood and large C, we either need to find a special method of doing the quadratures (we give an example below), or we need to use inaccurate rules together with corrections for the worst artefacts (Lerner (2002) gives an example for the latter).

4.2 The Multivariate Probit Likelihood. Greedy Forward Selection

The outcome of an EP update are new site parameters $\boldsymbol{b}_i$, $\boldsymbol{\pi}_i$. Based on these, the C independent IVM representations are updated, as is shown in Appendix C.1. We will address the implementation of greedy forward selection of the I_c below, but we first need to demonstrate how to compute the quadratures of Eq. 6 efficiently. To this end, we employ a particular likelihood for the multi-way setup: the *multivariate probit likelihood*. This likelihood is the natural generalization of the probit likelihood (see Appendix B.1) to C classes, namely as Gaussian perturbation of the indicator function of $\{v_y = \max_{y'} v_{y'}\}$:

$$P(y|\boldsymbol{u}) = \mathrm{E}_{\boldsymbol{n}\sim N(\boldsymbol{0},\boldsymbol{I})}\left[\mathrm{I}_{\{v_y = \max_{y'} v_{y'}\}}\right] = \mathrm{E}_{n_y}\left[\prod_{y'\neq y} \mathrm{E}_{n_{y'}}\left[\mathrm{I}_{\{v_y \geq v_{y'}\}}\right]\right], \quad \boldsymbol{v} = \boldsymbol{u} + \boldsymbol{p} + \boldsymbol{n}. \tag{8}$$

Here, $\boldsymbol{p} \in \mathbb{R}^C$ are bias parameters which are part of the hyperparameters $\boldsymbol{\theta}$. Expressions of the form of Eq. 6 have the same form as Eq. 8, where $\boldsymbol{v}$ still has a factorized Gaussian distribution (however, with a general diagonal covariance). Furthermore, we see that the coupling between the $\boldsymbol{v}$ components is mediated only through the single component v_y.

If we condition on v_y, the remaining components can be integrated out independently and analytically, resulting in terms involving the Gaussian c.d.f. $\Phi(z)$. Only the final integration over v_y has to be done using Gaussian quadrature. Therefore, for any C, the expression of Eq. 6 can be computed in $O(C)$. The details are slightly involved and are given in Appendix D. The usage of the multivariate probit likelihood in the context of sparse GP multi-way classification has been suggested by Seeger and Jordan (2004a) in unpublished work, and independently by Girolami and Rogers (2005).

We note that the multivariate probit likelihood is a log-concave function (this fact is shown in Appendix D). As discussed in Section 3.1, this has important consequences for the numerical stability of EP.

We now show how to generalize the greedy forward selection strategy of Section 3.2, thereby obtaining a conditional inference scheme similar to Algorithm 1, but for the multi-way model. We can score the candidate j for inclusion into I_c by

$$\Delta_{i,c} = -\mathrm{D}[Q'(u_{j,c}) \,\|\, Q(u_{j,c})] = -\frac{1}{2}\left(\log m_{j,c} + m_{j,c}^{-1} + a_{j,c}^{-1}(h'_{j,c} - h_{j,c})^2 - 1\right),$$
$$m_{j,c} = 1 + a_{j,c}\pi'_{j,c} = \max\{a_{j,c}/a'_{j,c}, 1\}.$$

In practice, the cost of computing a single $h'_{j,c}, a'_{j,c}$ is the same as computing all of $\boldsymbol{h}'_j, \boldsymbol{a}'_j$, therefore it makes sense to use a common selection index $J \subset \{1, \ldots, n\}$. For each inclusion, we compute the scores $\Delta_{j,c}$ for all c, $j \in J \setminus I_c$, then choose the top-scorer (j, c) for inclusion (namely j into I_c). This selection strategy is combined with randomization in order to shrink J during later inclusions. For this shrinkage, the entries j of J are ordered according to $\min_c\{\Delta_{j,c} \,|\, j \notin I_c\}$. Note that this means that J may (and should!) include such j for which $j \in I_c$ already for some c. Empirically, we observe that during early inclusions, we typically see that j is included into I_{y_j}, however during later stages the same j may be included into other I_c as well. Thus, while clearly $(\boldsymbol{x}_j, y_j)$ is usually most informative for $u_{y_j}(\cdot)$, other types of "informativeness" are considered as well.

We have completed the description of a factorized approximation for sparse conditional inference in the multi-way GP model. The essential steps in this method are calls to the IVM single process subroutines, as well as Gaussian quadratures. The factorization assumption relates our approach to variational mean field approximations, although the latter differ from our method in that the approximation Q minimizes a very different criterion.

5. Further Extensions

In Section 4, we gave a concrete example of how to design a multi-process method based on using the single-process IVM subroutines as major building blocks. In this Section, we provide further examples of how the basic IVM technology has been used to deal with more advanced situations.

Teh et al. (2005) are interested in a model for multi-output regression, where conditional dependencies between the outputs are represented by a linear mixture of latent Gaussian processes, akin to a supervised nonparametric variant of factor analysis. Their model is called *semiparametric latent factor model* (SLFM). In order to do inference efficiently in SLFMs, one can use a sequence of IVM representations to drive a nonparametric variant of the well-known belief propagation algorithm (Seeger et al., 2004, Seeger, 2006). This

marriage of structured graphical models inference with nonparametric Gaussian fields may have a range of other applications.

Both Seeger and Jordan (2004b) and Girolami and Rogers (2005) employ IVM representations in order to obtain multi-class methods. However, there have been many proposals to approach the multi-class problem using heuristic combinations of binary discriminants, c-against-rest being one of the simpler ones, and some of them perform quite well in practice. Any joint multi-class method has to compete in terms of running time and memory efficiency with these heuristics in order to become interesting to practitioners. Our simple factorizing method of Section 4 does so, while the methods of Seeger and Jordan (2004b), Girolami and Rogers (2005) are more complicated and expensive.

Lawrence and Platt (2004) use the IVM for multi-task learning (aka. hierarchical Bayesian learning). Lawrence and Jordan (2005) employ the IVM for semi-supervised learning, where unlabeled data is used to improve binary classification. The idea is to separate the real line (where the latent $u(\cdot)$ lives) in three categories, rather than two (positive, negative) as used with ordinary classification. The additional "null category" is centered at 0, and unlabeled data $\boldsymbol{x}_i$ is forced not to have their latent value u_i in this zone.

Lawrence et al. (2005) try to incorporate invariances w.r.t. transformations of $\boldsymbol{x}$ into a classifier by adopting the idea of virtual support vectors (see (Schölkopf and Smola, 2002), Sect. 11.3) to the IVM. For example in image classification, a bitmap can be rotated or translated to a certain extent without changing its class membership. The brute force approach to handling such invariances is simply to augment the original data set with virtual points generated by applying small-scale transformations to the data. It is more economical to apply the augmentation to the active points only (the "informative vectors"). In classification, points that are not active often lie far from the decision boundary, so the fitted method is invariant to small-scale transformations of them anyway. Some preliminary experimental results with virtual informative vectors are shown in Section 6.5.

6. Experiments

In this Section, we provide some experimental validation of our framework. These extend the studies which have already been obtained with the IVM in prior work (Lawrence et al., 2003, Seeger, 2003, Teh et al., 2005, Lawrence and Jordan, 2005). Since our framework is very general, we cannot hope to cover its potential applications in much depth here. The code we used for our experiments will be made publicly available for research purposes in the near future, and we hope to stimulate new applications beyond what we suggest in this paper.

6.1 Binary Classification (USPS)

In this Section, we present results on the USPS handwritten digits database (LeCun et al., 1989). The setup closely parallels the study in (Seeger, 2003), Sect. 4.8.4. USPS input points are gray-scale patterns of size 16×16, depicting the digits $0, \ldots, 9$. The training set contains $n = 7291$, the test set $m = 2007$ patterns. We use the corresponding c-against-rest

c	d	$terr\,(\%)$	tlh	c	d	$terr\,(\%)$	tlh	c	d	$terr\,(\%)$	tlh
0	50	3.64	-0.117	1	50	0.75	-0.088	2	50	4.24	-0.190
0	100	0.95	-0.081	1	100	0.70	-0.037	2	100	2.09	-0.110
0	200	0.80	-0.035	1	200	0.75	-0.032	2	200	1.25	-0.094
0	500	0.80	-0.029	1	500	0.70	-0.031	2	500	1.59	-0.052
3	50	2.84	-0.110	4	50	4.38	-0.220	5	50	4.98	-0.230
3	100	1.69	-0.087	4	100	2.74	-0.148	5	100	2.94	-0.117
3	200	1.15	-0.079	4	200	1.40	-0.091	5	200	1.00	-0.094
3	500	1.30	-0.045	4	500	1.49	-0.048	5	500	1.20	-0.041
6	50	2.24	-0.263	7	50	2.39	-0.089	8	50	2.39	-0.112
6	100	1.64	-0.097	7	100	0.90	-0.087	8	100	1.94	-0.104
6	200	0.70	-0.028	7	200	0.65	-0.029	8	200	1.69	-0.067
6	500	0.65	-0.020	7	500	0.60	-0.024	8	500	1.15	-0.054
9	50	1.44	-0.080								
9	100	1.15	-0.074								
9	200	0.95	-0.065								
9	500	0.80	-0.036								

Table 1: Results c-against-rest for USPS. d: final active set size. $terr$: test set error (in percent). tlh: test set average log likelihood.

tasks, $c = 0, \ldots, 9$. We employ the *Gaussian* kernel (also called *RBF* kernel)

$$K(\boldsymbol{x}, \boldsymbol{x}') = v \exp\left(-\frac{w}{2p}\|\boldsymbol{x} - \boldsymbol{x}'\|^2\right), \quad v, w > 0,\ \boldsymbol{x} \in \mathbb{R}^p. \tag{9}$$

We use a hyperprior $P(\boldsymbol{\theta})$, adding $-\log P(\boldsymbol{\theta})$ to the learning criterion. $\log v$ is $N(-1, 1)$, w^{-1} is Gamma with mean 1 and degrees of freedom 1. The likelihood is probit classification (see Appendix B.1), with hyperprior $N(0, 25)$ on the intercept parameter β. The optimization is started from $v = 10,\ w = 0.0166$, the latter being one over the average component variance. We do not employ randomized greedy selection, and we compare different final active set sizes $d = 50, 100, 200, 500$. For hyperparameter learning, we use the constant strategy described at the end of Section 3.3: the active set I and site parameters are reselected in major steps (outer iterations), but kept constant during the inner iterations (minor steps) (see Section 6.1.1). We allow for 15 outer iterations with a maximum of 8 inner iterations each.[9] Results are shown in Table 1.

These results are comparable to the ones obtained in (Seeger, 2003), Sect. 4.8.4, and they are state-of-the-art (see Schölkopf and Smola (2002), Chap. A.1). Note that the test set average log likelihood $m^{-1} \sum_i \log Q(y_{*,i}|\boldsymbol{x}_{*,i}, D)$ always improves with growing active set size, while the test error not always does (the differences in test error between $d = 200$ and $d = 500$ are not significant).

9. Judging from plotting the criterion curves, this setting is very conservative, and time can be saved by stopping earlier.

6.1.1 Inner Loop Optimization Strategies

Recall from the end of Section 3.3 that the learning criterion ϕ is really a function of the hyperparameters $\boldsymbol{\theta}$ and the site parameters $\boldsymbol{s}$, if the latter cannot be considered fixed due to a Gaussian likelihood, and that $\nabla_{\boldsymbol{\theta}}\phi$ is correct only if $\boldsymbol{s}$ is kept fixed. Possible strategies for running the inner optimizations (for fixed active set I) have been stated in Section 3.3, we call them *constant* (keep $\boldsymbol{s}$ constant during inner loop), *hybrid*, and *exact* (updating $\boldsymbol{s}$ for each criterion evaluation). In this Section, we present a simple comparison for these strategies on the USPS "2"-against-rest task and final active set size $d = 200$. To this end, we allow for 15 outer iterations of up to 8 inner steps, and protocol the learning criterion together with the test set error and average test set log likelihood. The results are shown in Figure 1.

The final test error and test average log likelihood (after a final major step) are: 0.013453; −0.099876 (constant), 0.016442; −0.064983 (hybrid), 0.014948; −0.091401 (exact). Note that the exact strategy is hampered by our inability of optimizing the criterion appropriately based on the incorrect gradient, in that more than half of the inner loops terminate with a failed line search. It cannot be recommended together with our present gradient-based optimization. Note also that the criterion curve (upper row) for the constant strategy is comparable to the others only at major steps, since between these $\boldsymbol{s}$ is kept constant and not reselected. There is an interesting antithetic behaviour between test error and test set log likelihood for the constant strategy: both increase during inner iterations and are decreased by major steps. This supports our hypothesis of a "runaway" effect during inner loops. However, the exploration leads to an overall trend of improvement. The hybrid strategy shows some failure patterns, in that the criterion increases during inner loops. This is because the line search optimizes $\phi(\cdot, \boldsymbol{s})$ for fixed $\boldsymbol{s}$, which is decreased properly indeed, but the subsequent update of $\boldsymbol{s}$ leads to an overall increase. Based on the results here, we favour the constant strategy over the hybrid one, since it is simpler to implement and behaves monotonically decreasing at least during inner iterations.

6.2 Binary Classification (MNIST)

In this Section, we present results on the MNIST handwritten digits database (available at `http://www.research.att.com/~yann/exdb/mnist/index.html`). This task has previously been used with IVM in (Lawrence et al., 2003, Seeger, 2003). MNIST input points are gray-scale patterns of size 28×28 which are sparse in that a significant fraction of pixels has value 0. The training set contains $n = 60000$, the test set $m = 10000$ patterns. We use the corresponding c-against-rest tasks, restricting our attention to the hardest ones $c = 5, 8, 9$. We employ the Gaussian kernel (Eq. 9) with the same hyperprior as in Section 6.1, starting the optimizations from $v = 10$, $w = 0.066$. We use final active set sizes $d = 3200$ ($c = 5$), $d = 3600$ ($c = 8$), and $d = 3900$ ($c = 9$). Since the task is much larger than USPS, and the active set sizes are large, we employ randomized greedy selection (see Section 3.2), limiting $\boldsymbol{M}_{\tilde{J},\cdot}$ to $3.6 \cdot 10^7$ entries. This means that for the first 100 inclusions, all remaining patterns are scored. After that, $\tilde{J}$ is shrunk subsequently. The final size of $\tilde{J}$ is $36000000/d$, which is 9230 for $c = 9$. We allow for 7 outer iterations (major steps), each consisting of 4 line searches. The results are shown in Table 2.

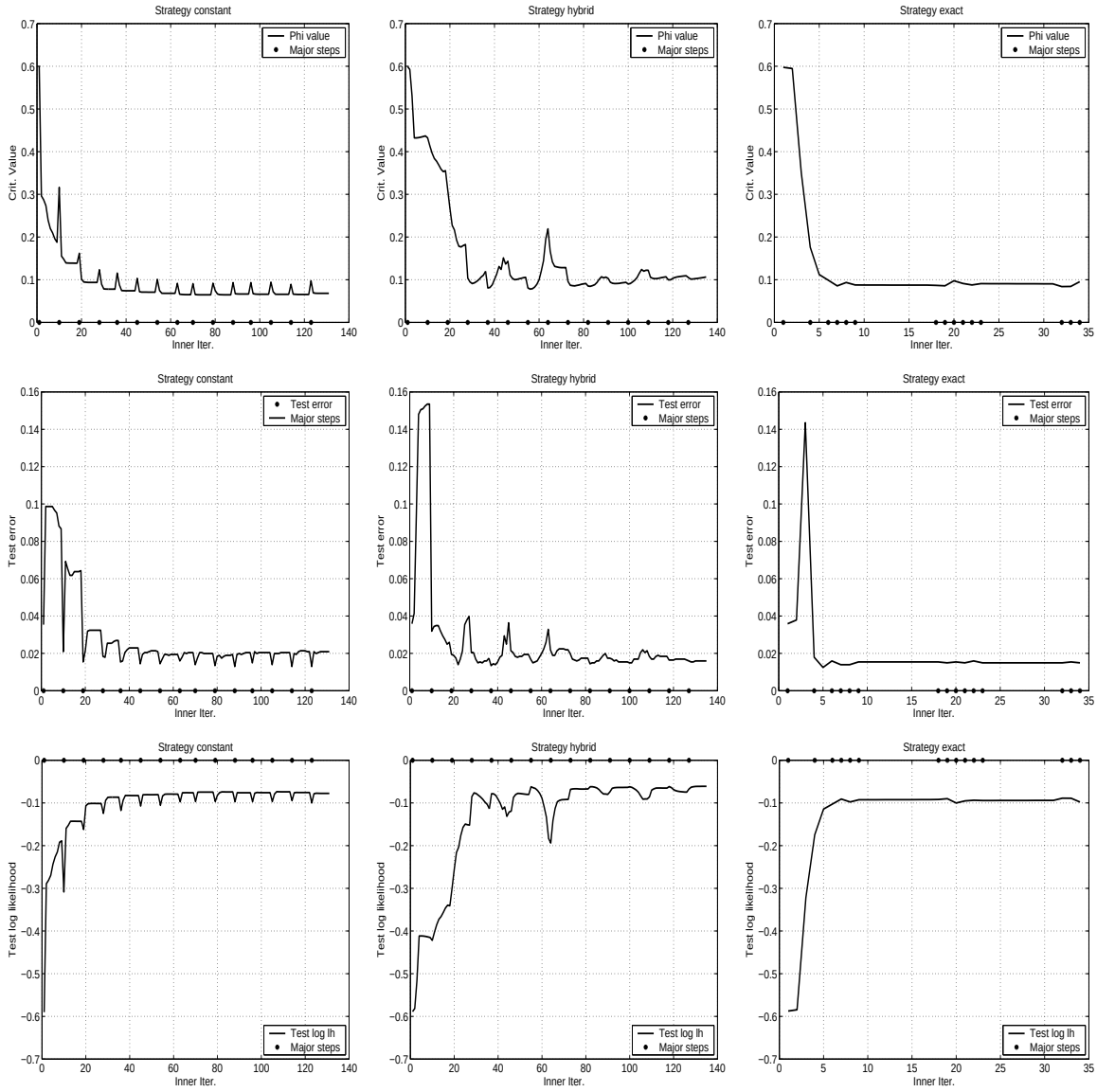

Figure 1: Comparison of inner optimization strategies *constant* (left column), *hybrid* (middle column), *exact* (right column). Criterion value (upper row), test set error (middle row), test set average log likelihood (bottom row).

The results are slightly worse than those quoted in (Lawrence et al., 2003, Seeger, 2003), which can be attributed to the hyperparameter learning procedure behaving suboptimally here. For the experiments in (Lawrence et al., 2003, Seeger, 2003), hyperparameters were selected by validation on a holdout dataset.

c	d	$terr\,(\%)$	tlh	v	w
5	3200	0.37	-0.026	22.91	11.38
8	3600	0.52	-0.043	18.39	14.25
9	3900	0.74	-0.037	19.45	15.55

Table 2: Results c-against-rest for MNIST. d: final active set size. $terr$: test set error (in percent). tlh: test set average log likelihood. v, w: final kernel parameter values.

6.3 Regression with Gaussian Noise (pumadyn-32nm)

In this Section, we present results on the *pumadyn-32nm* univariate regression dataset contained in the DELVE archive (see `www.cs.toronto/~delve`). This set, created using a robot arm simulator, is highly nonlinear and has fairly low noise. There are 32 real-valued attributes, and the database has 8192 cases. We fit a linear discriminant by least squares and subtract this effect off, normalizing the residuals to unit variance. We also normalize each input attribute to unit variance. We then use 10 random splits into training ($n = 7192$) and test set ($m = 1024$). Preprocessing and splits are the same as in the studies of Seeger et al. (2003), Seeger (2003). We use the *squared-exponential kernel* here, which is an anisotropic variant of the Gaussian:

$$K(\boldsymbol{x}, \boldsymbol{x}') = v \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}')^T (\operatorname{diag} \boldsymbol{w})(\boldsymbol{x} - \boldsymbol{x}')\right), \quad v, w_j > 0,\ \boldsymbol{x} \in \mathbb{R}^p. \tag{10}$$

As opposed to the Gaussian kernel, we have a separate inverse squared length scale parameter w_j for each dimension. Importantly, this parameterization implements a technique called *automatic relevance determination* (ARD) (Neal, 1996). Suppose variations in some of the attributes x_j are not relevant for explaining the variations in the response y. In this case, the "Occam's razor" effect embodied in Bayesian estimation techniques will lead to the corresponding w_j hyperparameters being driven close to 0. It is important to understand that ARD is about *conditional* variance (or dependence) rather than marginal one. All attributes in our dataset have marginal variance 1, yet only the attributes $5, 4, 16, 15$ are significantly relevant for predicting y, all others do not contain useful information. Due to our preprocessing, a linear least squares fit to the data results in a prediction which is constant 0 with mean square (MS) error about 1. If a method fails to identify the relevant attributes, it will in general not perform much better, even if it can represent nonlinear functions. For example, our own method fails if the Gaussian kernel is used, as do related techniques such as SVM regression. Note that in order to implement ARD on this task, we need to learn 34 hyperparameters: the w_j, the variance v, and the noise variance σ^2.

Our performance criterion here is average test mean square (MS) error. When comparing our figures here with the results of Seeger et al. (2003), Seeger (2003), note that the latter actually measure 1/2 MS error. Also note that MS error does not depend on the estimates of the predictive variance. Due to the special nature of the task, we can expect two different outcomes in principle. Either, the learning procedure fails to identify and suppress the irrelevant attributes and attains MS error close to 1, or learning succeeds with a MS error

two orders of magnitude less. We run our method with final active set sizes $d = 200, 300, 500$, allowing for 15 outer iterations of 8 inner steps each. We find that learning succeeds in all cases for all splits, with MS errors of 0.0464 ± 0.0047 for $d = 500$, 0.0478 ± 0.0059 for $d = 300$, and 0.0544 ± 0.0121 for $d = 200$. The average test set log likelihoods are 0.1067 ± 0.0691 for $d = 500$, 0.0991 ± 0.0650 for $d = 300$, and 0.0338 ± 0.1247 for $d = 200$. The positive result for $d = 200$ is in marked contrast to the findings in Seeger (2003), Sect. 4.8.3, where all ten runs failed to detect the relevant components. This indicates that the learning criterion we use here can behave superior to the variational criterion used in Seeger (2003).[10]

When comparing our results here with the ones of Seeger et al. (2003), Snelson and Ghahramani (2006), it is important to keep in mind that the IVM framework is based on the idea of using the simplest possible likelihood approximation (see Section 2.2) in order to attain an optimal scaling. It was designed for situations where (i) patterns do differ strongly in terms of "informativeness", and finding the most informative ones early is of central importance; and (ii) the main aim is to obtain a good estimate of the predictive mean. In fact, the study in (Lawrence et al., 2003) shows that IVM can be faster than the highly efficient and well-developed support vector machine (SVM). Binary classification is a good example, but regression with Gaussian noise is not. Other more involved likelihood approximations (see Section 2.2) result in a better posterior approximation for the same active set size d, which is especially reflected in better estimates of the predictive variances. However, these more advanced approximations are much more difficult to implement and costly to run for models with non-Gaussian likelihood, in fact all complete studies we know of have been done for Gaussian regression only. The performance and complexity gap between IVM and these methods is rather small for Gaussian regression, and we would in fact recommend using one of the latter techniques in this case, but it is not clear whether a completely general framework should be based on these advanced likelihood approximations.

6.4 Multi-Way Classification (satimage)

In this Section, we present multi-way classification experiments on the *satimage* task of the *statlog* repository (available at `http://www.niaad.liacc.up.pt/old/statlog/`). The data is from a remote sensing study, the input points being localized measurements of 3×3 neighborhoods in 4 spectral bands (36 integer-valued attributes). The aim is classification of soil type, and there are $C = 6$ soil classes. The training set has $n = 4435$ cases, there are $m = 2000$ test points. We did not apply any preprocessing to the data.

Our method presented in Section 4 does not come with a hyperparameter learning criterion at present, so the parameters of the kernels $K^{(c)}$ (one for each class; we use Gaussian kernels of Eq. 9 with different parameters, thus 12 hyperparameters in total) have to be selected by other conventional means such as cross-validation. For our purposes here, we simply use the same parameters as found by cross-validation optimization with the method *mc-sep* reported by Seeger and Chapelle (2006). This is a convenient, but certainly not optimal choice for our method here: the method of Seeger and Chapelle (2006) is an MAP approximation to Bayesian inference, it is not sparse, and the intercept parameters

10. It is maybe surprising that even in the case of a Gaussian likelihood, there are different possible marginal likelihood approximations for the IVM. This is because we want to avoid the "canonical" approximation $P(\boldsymbol{y}_I)$ obtained by ignoring all non-active points.

dtot	*terr* (%)	*tlh*
2000	8.7	-0.298
5000	8.2	-0.219
7500	8.4	-0.217
10000	8.65	-0.221

Table 3: Results multi-way factorized IVM for satimage task. *dtot*: sum of active set sizes d_c. *terr*: test set error (in percent). *tlh*: test set average log likelihood.

p_c are treated as primary rather than hyperparameters.[11] Since the hyperparameters are determined for a model with the multiple logistic (softmax) likelihood, we employ this likelihood here as well (not the multivariate probit likelihood of Section 4.2), using a Gauss-Hermite product quadrature rule with $3^C = 729$ evaluations (see Appendix B.2). The results are given in Table 3.

In comparison, the method *mc-sep* attained a test error of 7.55%. These results for the multi-way factorized IVM method are fairly discouraging, indicating that the factorized approximation may not be a very good one. Furthermore, the results do not improve with growing total active set size.

6.5 Virtual Informative Vectors (USPS)

In this Section, we present some preliminary results for the IVM virtual informative vector technique described in Section 5. We again make use of the USPS database (see Section 6.1). This study previously appeared in (Lawrence et al., 2005). The results are obtained using a different IVM implementation than was used for the other experiments quoted here.

We only consider translational invariances in our experiments, for which we use c-against-rest IVM classifiers, combining them by majority vote on the log predictive probabilities in order to obtain a multi-class method. We first apply the IVM classification method to the dataset, using a Gaussian kernel (Eq. 9) combined with a linear term, and learning the hyperparameters. For each c, we construct an augmented dataset by adding four translations of each active pattern, by one pixel up, down, left, and right. We then reselect an active set of size $d = 1000$ from this augmented dataset using IVM conditional inference for the hyperparameters learned already. The test results are shown in Table 4.

The resulting performance of the IVM with "virtual informative vectors" is very similar to that found through the use of "virtual support vectors" with the SVM.

7. Conclusions

We have presented a general framework for efficient sparse approximations of Bayesian inference and learning in single Gaussian process models which can be configured by arbitrary likelihood and covariance functions. Our method scales as $O(n\,d^2)$, where n is the dataset size, and $d \ll n$ is the active set size, a parameter which can be controlled. The active set

11. We fix p_c to the MAP values found by *mc-sep*.

0	**1**	**2**	**3**	**4**	
0.648 ± 0.00	0.389 ± 0.03	0.967 ± 0.06	0.683 ± 0.05	1.06 ± 0.02	
5	**6**	**7**	**8**	**9**	**Overall**
0.747 ± 0.06	0.523 ± 0.03	0.399 ± 0.00	0.638 ± 0.04	0.523 ± 0.04	3.30 ± 0.03

Table 4: Results for the virtual informative vector technique on the USPS digit task. Shown are the results for the individual binary c-against-rest tasks and the overall error computed from voting with these discriminants. We give mean and variance of test error over ten runs.

$I \subset \{1, \ldots, n\}$ can be seen as a bottleneck, separating the n observations from the posterior process. The central idea is to maintain an efficient representation of the posterior belief which allows to keep the posterior marginals at all datapoints up-to-date at all times, and which is also required to compute the learning criterion and its gradient. This representation can be updated in a numerically stable manner as I grows by new inclusions. The marginals represent our current belief of what each target should be, and by comparing this to the true targets we can select a maximally "informative" point for the next inclusion into I.

The IVM representation and forward selection technique for choosing I can be applied in situations where the single GP model is embedded in a larger context, possibly dealing with several dependent processes. We have presented an example in Section 4, treating a multi-way classification model. While the results for this method presented in Section 6.4 are not encouraging, the idea of a completely factorizing approximation with IVM representations for the single factors may be better suited for other multiple process models. The IVM technique has already been applied in a host of other situations, some of which go beyond single process models, this work is reviewed in Section 5.

The IVM framework consists of a combination of elementary techniques and is fairly well understood by now. By focussing on the main criteria of computational efficiency, generality, and numerical stability, we hope that it will be accepted as a "black box" technique for representing nonlinear univariate functions to be inferred from large datasets in a probabilistic manner, reaching application areas in machine learning which could not profit from nonparametric GP modelling so far due to resource constraints. We plan to release the code used in our experiments here into the public domain in the near future.

7.1 Related Work

A large number of different sparse approximations for GP inference and related kernel machines have been proposed so far. Seeger (2003) gives a detailed overview of many of them and their relations, see also Section 2.2. More recently, Quinonero Candela and Rasmussen (2005) give an overview which encompasses augmentation methods as well. While restricted to models for regression with Gaussian noise, they have a somewhat different view which complements the presentation in (Seeger, 2003) in a useful way. Both overviews taken together are fairly complete and provide relevant references which we omit here.

The framework presented here is novel, in that we show how to approximate the full Bayesian program of inference and hyperparameter learning for single process GP models with essentially arbitrary likelihood functions (some aspects of our implementation require that the sites are log-concave). Most previous work has concentrated on GP regression with Gaussian noise, for which exact inference is analytically tractable. Many issues such as site approximations, when and how to update the corresponding parameters, and how to best approximate the marginal likelihood for learning are not present in this special case. Conditional inference for the IVM has been proposed by Lawrence et al. (2003), but hyperparameters were adjusted by cross-validation there. Seeger (2003) gives a general framework such as ours here, applications thereof have been proposed by Teh et al. (2005), Seeger and Jordan (2004b). The difference to our framework here lies in the learning criterion, which in their case is a variational lower bound on the log marginal likelihood, while we propose a sparse variant of the EP marginal likelihood approximation here. While no systematic comparisons between the two have been done so far, we feel that our proposal here is more appealing, because EP (or ADF) is used all along in order to deal with non-Gaussian likelihoods. Also, our findings in Section 6.3 suggest that learning with the EP-based criterion may be more powerful. Other, more subtle advantages arise in the context of multi-way classification, where the quadrature terms required for our criterion here can be computed efficiently (see Section 4.2), while this is very awkward and inaccurate for the variational criterion.

The IVM was designed originally as direct Bayesian probabilistic competitor to the well-known support vector machine (SVM). Lawrence et al. (2003) present a study comparing IVM conditional inference (without hyperparameter learning) directly to the SVM, they find that both give comparable results on large scale binary classification tasks, with the IVM often being significantly faster. Since the SVM is not a proper probabilistic model, empirical Bayesian marginal likelihood maximization as exercised here for the IVM cannot be used for hyperparameter learning, and user-intensive methods such as cross-validation are typically used with the SVM. Valid predictive probability estimates can be obtained with IVM, but not with SVM.[12] Furthermore, more complex probabilistic models can be constructed by combining IVM representations, an example is given by Teh et al. (2005). We note that the large margin principle driving the SVM can also be generalized towards more advanced models (Taskar et al., 2004).

How does our framework relate to maybe the most popular sparse approximation method, the *relevance vector machine* (RVM) of Tipping (2001)? It has already been mentioned in Section 2.2 that the RVM does not meet one of our criteria for a sparse GP approximation, in that it does not result in a posterior distribution $Q(u(\cdot)|D)$ which is itself a Gaussian process. One could say that the RVM approximates the mean function of such a predictive process in a sparse way, but if it is used to obtain posterior uncertainties, serious shortcomings arise. For example, the predictive variance away from the data tends to zero rather than towards the (maximum) prior variance. One can "heal" the RVM through augmentation

12. This point is made clear in (Seeger, 2004), Sect. 7, and some empirical evidence is given in (Seeger, 2003), Sect. 4.7.2. Platt (1999) gives a heuristic for mapping SVM outputs to probabilities, but this is dismissed as inappropriate in (Seeger, 2003). We note that the IVM is a valid, but not a good method for estimating predictive uncertainties. If this estimation is of central importance for an analysis, we recommend using a more advanced likelihood approximation (see Section 2.2).

(Rasmussen and Quinonero Candela, 2005), but why not start with a proper GP approximation in the first place? Our IVM framework has similar, if not better scaling characteristics than the greedy RVM variant proposed by Faul and Tipping (2002). Furthermore, the RVM has rarely been applied to models with non-Gaussian likelihood, and in these cases a second order (Laplace) approximation has been used. Kuss and Rasmussen (2005) show that for binary classification, expectation propagation (the approximation underlying the IVM) leads to better predictive results than second order techniques.

Recently, Snelson and Ghahramani (2006) propose some interesting new ideas. First, as already discussed in Section 2.2, they criticize the likelihood approximation proposed by Seeger et al. (2003) (which is already an improvement in terms of accuracy on the IVM one) for underestimating predictive variances, and suggest another one which comes at the same cost. Compared to the IVM, either of these frameworks is significantly more expensive, especially in the case of non-Gaussian likelihoods. In fact, Snelson and Ghahramani (2006) only deal with the Gaussian likelihood case. Seeger (2003) shows how to generalize the work of Seeger et al. (2003) to non-Gaussian likelihoods, and a corresponding generalization of the method of Snelson and Ghahramani (2006) is subject to future work. Snelson and Ghahramani (2006) also suggest optimizing for the active points supporting $\boldsymbol{u}_I$. Recall from Section 2.2 that these do not necessarily have to be a subset of the training inputs, although good arguments can be made in favour of this restriction. On the positive side, their proposal does not require a combinatorial search for I, but rather a non-convex optimization over the active points themselves, which can be done using standard software if these points lie in some $\mathbb{R}^p$. They optimize the marginal likelihood approximation suggested by Seeger et al. (2003), both for the active points and the hyperparameters in turn. They always converge to a local optimum of this criterion.

While no direct comparison has been done between our IVM framework here and their method, we can still argue in favour of the IVM. First, the IVM uses a simpler likelihood approximation which renders conditional inference much more efficient. This comes at the cost of providing a less accurate approximation, which can be demonstrated especially well for regression with Gaussian noise, but which is much less of a disadvantage when considering classification tasks. A more interesting direct comparison would be between a method employing the likelihood approximation of Snelson and Ghahramani (2006) with our EP extension to non-Gaussian likelihood and our learning criterion to their method of choosing the active points by optimization. We will consider the former method in future work. We do think that forward selection with the informative criteria and the fact that all targets y_i are actually known and accessible, provides a good solution for the combinatorial and seemingly hard problem of choosing an optimal active set I. The method of Snelson and Ghahramani (2006) is restricted to input points coming from some $\mathbb{R}^p$ with fairly small p. They need heavy additional machinery, namely a good non-convex optimizer for a very large state space, even if there are only few hyperparameters. Moreover, it is not clear at all that their smooth optimization is in fact simpler, since they might easily get trapped in bad local minima. While the experiments shown in (Snelson and Ghahramani, 2006) show some improvement over the ones in (Seeger et al., 2003) for fixed hyperparameters, their results for *joint* optimization of active points *and* hyperparameters are fairly inconclusive.

Acknowledgments

The authors acknowledge very useful discussions with and sharing of ideas by Chris Williams, Yee-Whye Teh, Michael Jordan, Carl Rasmussen, Joaquin Quinonero Candela, and Lehel Csato. This work was supported in part by the IST Programme of the European Community, under the PASCAL Network of Excellence, IST-2002-506778. This publication only reflects the authors' views.

Appendix A. Notation. Miscalleneous

In this Section, we describe the notation used in this paper. We also collect details for some arguments made in the text.

A.1 Notation

We use a subindexing notation for matrices and vectors which is familiar in the linear algebra literature. Vectors and matrices are set in bold face. Subset indexing is defined as $\boldsymbol{A}_{I,J} := (\alpha_{i,j})_{i\in I, j\in J}$, I, J ordered index sets. $\cdot$ denotes the full index, i the singleton $\{i\}$, and $\boldsymbol{A}_I := \boldsymbol{A}_{I,I}$. $\boldsymbol{I}$ denotes the identity matrix, $\boldsymbol{I} = (\delta_{i,j})_{i,j}$, its columns are denoted as $\boldsymbol{\delta}_i = \boldsymbol{I}_{\cdot,i}$. The vector of all ones is $\boldsymbol{1} = (1)_i$, the vector of all zeros is $\boldsymbol{0} = (0)_i$. Note the special role of $\boldsymbol{I}_{I,\cdot}$ (selection) and $\boldsymbol{I}_{\cdot,I}$ (distribution), in that $\boldsymbol{A}_{I,J} = \boldsymbol{I}_{I,\cdot}\boldsymbol{A}\boldsymbol{I}_{\cdot,J}$, *etc.*. For matrices of the same size, $\boldsymbol{A} \circ \boldsymbol{B} = (a_{i,j}b_{i,j})_{i,j}$ denotes the Hadamard product.

The multivariate Gaussian distribution in $\mathbb{R}^n$ has the density

$$N(\boldsymbol{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = |2\pi\boldsymbol{\Sigma}|^{-1/2} \exp\left(-\frac{1}{2}(\boldsymbol{u}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{u}-\boldsymbol{\mu})\right).$$

We also need the following unnormalized form

$$N^U(\boldsymbol{u}|\boldsymbol{b}, \boldsymbol{B}) = \exp\left(\boldsymbol{b}^T\boldsymbol{u} - \frac{1}{2}\boldsymbol{u}^T\boldsymbol{B}\boldsymbol{u}\right).$$

If $N(\boldsymbol{u}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is a Gaussian, the corresponding unnormalized Gaussian is proportial to the density if $\boldsymbol{B} = \boldsymbol{\Sigma}^{-1}$, $\boldsymbol{b} = \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}$. The $\boldsymbol{b}$, $\boldsymbol{B}$ are also called *natural parameters*. Note that $N^U(\boldsymbol{u})$ may not be properly normalizable over $\mathbb{R}^n$, in fact we can do this iff $\boldsymbol{B}$ is positive semidefinite. $\Phi(z)$ denotes the *cumulative distribution function* (c.d.f.) of the standard scalar Gaussian $N(0,1)$, *i.e.* $\Phi(z) = (2\pi)^{-1/2}\int_{-\infty}^z e^{-x^2/2}\,dx$.

A.2 Likelihood Approximations

Recall from Section 2.2 that in the context of this paper, we assume that a sparse GP approximation still results in a proper Gaussian predictive process, and that for each test point $\boldsymbol{x}_*$, we only have to evaluate the kernels $K(\boldsymbol{x}_*, \tilde{\boldsymbol{x}}_i)$ for d fixed active points $\tilde{\boldsymbol{x}}_i$. Repeating the argument of Seeger (2003), we show that this implies that such an approximation can always be obtained by replacing the likelihood $P(D|\boldsymbol{u})$ (assumed Gaussian here for simplicity) by a Gaussian likelihood approximation $Q(D|\boldsymbol{u}_I)$ which depends on $\boldsymbol{u}_I = (u(\tilde{\boldsymbol{x}}_i))_i \in \mathbb{R}^d$ only, and is therefore parameterized in terms of at most $O(d^2)$ parameters.

Recall that the approximation is restricted to replace $P(\boldsymbol{u}_J|D)$ by a Gaussian $Q(\boldsymbol{u}_J|D) = N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, without doing other changes to the exact setup. We assume that $\boldsymbol{u}_J$ includes $\boldsymbol{u}$ (at the training points) and $\boldsymbol{u}_I$ (at the active points), which can trivially be obtained. Denote the points supporting $\boldsymbol{u}_J$ by $\boldsymbol{x}_j^{(J)}$. Now, $Q(u_*|D) = \int P(u_*|\boldsymbol{u}_J)Q(\boldsymbol{u}_J|D)\,d\boldsymbol{u}_J$, and standard formulae give $\mathrm{E}[u_*|D] = \boldsymbol{k}_*^T\boldsymbol{K}^{-1}\boldsymbol{\mu}$ and

$$\mathrm{Var}[u_*|D] = K(\boldsymbol{x}_*, \boldsymbol{x}_*) - \boldsymbol{k}_*^T\left(\boldsymbol{K}^{-1} - \boldsymbol{K}^{-1}\boldsymbol{\Sigma}\boldsymbol{K}^{-1}\right)\boldsymbol{k}_*,$$

where $\boldsymbol{k}_* = (K(\boldsymbol{x}_*, \boldsymbol{x}_j^{(J)}))_j$ and $\boldsymbol{K} = (K(\boldsymbol{x}_i^{(J)}, \boldsymbol{x}_j^{(J)}))_{i,j}$. Note that we slightly abuse the notation in this Section, denoting the kernel matrix at all $\boldsymbol{x}_j^{(J)}$ by $\boldsymbol{K}$. The kernel matrix

for the input datapoints is contained in $\boldsymbol{K}$. We only allow for the evaluation of $\boldsymbol{k}_{*,I} = (K(\boldsymbol{x}_*, \tilde{\boldsymbol{x}}_i))_i \in \mathbb{R}^d$. Therefore, we must have $\boldsymbol{\mu} = \boldsymbol{K}_{\cdot,I}\boldsymbol{v}$ and $\boldsymbol{\Sigma} = \boldsymbol{K} - \boldsymbol{K}_{\cdot,I}\boldsymbol{B}\boldsymbol{K}_{I,\cdot}$, with $\boldsymbol{v} \in \mathbb{R}^d$, $\boldsymbol{B} \in \mathbb{R}^{d,d}$. $\boldsymbol{B}$ is symmetric. Let $\boldsymbol{B} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$ be the spectral decomposition, where $\boldsymbol{D}$ is diagonal with nonzero entries. Now, any Gaussian posterior can be written as $Q(\boldsymbol{u}_J|D) \propto P(\boldsymbol{u}_J)N^U(\boldsymbol{u}_J|\boldsymbol{b}, \boldsymbol{\Pi})$, having inverse covariance matrix $\boldsymbol{K}^{-1} + \boldsymbol{\Pi}$. The Sherman-Morrison-Woodbury formula gives

$$\boldsymbol{K}^{-1} + \boldsymbol{I}_{\cdot,I}\boldsymbol{U}\left(\boldsymbol{D}^{-1} - \boldsymbol{U}^T\boldsymbol{K}_I\boldsymbol{U}\right)^{-1}\boldsymbol{U}^T\boldsymbol{I}_{I,\cdot} = \boldsymbol{\Sigma}^{-1} = \boldsymbol{K}^{-1} + \boldsymbol{\Pi}$$

(all terms exist because $\boldsymbol{\Sigma}$ is invertible), therefore $\boldsymbol{\Pi} = \boldsymbol{I}_{\cdot,I}\boldsymbol{\Pi}_I\boldsymbol{I}_{I,\cdot}$. Let $\boldsymbol{\Pi}_I = \boldsymbol{V}\boldsymbol{E}\boldsymbol{V}^T$ be the spectral decomposition. Next, using Sherman-Morrison-Woodbury once more, we have

$$\boldsymbol{K}_{\cdot,I}\boldsymbol{v} = \boldsymbol{\mu} = \left(\boldsymbol{K} - \boldsymbol{K}_{\cdot,I}\boldsymbol{U}\left(\boldsymbol{E}^{-1} + \boldsymbol{U}^T\boldsymbol{K}_I\boldsymbol{U}\right)^{-1}\boldsymbol{U}^T\boldsymbol{K}_{I,\cdot}\right)\boldsymbol{b} = \boldsymbol{K}\boldsymbol{b} - \boldsymbol{K}_{\cdot,I}\boldsymbol{z},$$

which shows that $\boldsymbol{b} = \boldsymbol{I}_{\cdot,I}\boldsymbol{b}_I$. Thus, $N^U(\boldsymbol{u}_J|\boldsymbol{b}, \boldsymbol{\Pi})$ is really only a function of $\boldsymbol{u}_I$. This completes the proof.

Appendix B. Expectation Propagation. Gaussian Quadrature

In this Section we describe the EP update in detail. An update requires the solution of Gaussian expectations of the likelihood function. The latter can be computed approximately using numerical quadrature, as explained here.

B.1 The EP Update

Recall the description of an EP update in the case of a single GP model from Section 2.3. We restrict ourselves to sites t_i which are functions of a single component u_i of $\boldsymbol{u}$, the extension to multivariate likelihoods can be found in Seeger (2003). Recall that $Q_{\backslash i}(\boldsymbol{u}) \propto Q(\boldsymbol{u})\tilde{t}_i(u_i)^{-1}$, and $\hat{P}_i(\boldsymbol{u}) \propto t_i(u_i)Q_{\backslash i}(\boldsymbol{u})$. The problem of computing mean and covariance of $\hat{P}_i$ is equivalent to minimizing the relative entropy $\mathrm{D}[\hat{P}_i \,\|\, Q']$ over all Gaussians $Q'(\boldsymbol{u})$. Since $\hat{P}_i(\boldsymbol{u}) \propto f(u_i)Q(\boldsymbol{u})$, we have that $\hat{P}_i(\boldsymbol{u}_{\backslash i}|u_i) = Q(\boldsymbol{u}_{\backslash i}|u_i)$, so that we have $Q'(\boldsymbol{u}_{\backslash i}|u_i) = Q(\boldsymbol{u}_{\backslash i}|u_i)$ for the minimizer. Now, $\mathrm{D}[\hat{P}_i(\boldsymbol{u}) \,\|\, Q'(\boldsymbol{u})] = \mathrm{D}[\hat{P}_i(u_i) \,\|\, Q'(u_i)]$, thus we just need mean h'_i and variance a'_i of the marginal $\hat{P}_i(u_i)$. Let $Q(u_i) = N(h_i, a_i)$. We have that

$$Q_{\backslash i}(u_i) = N(h_{\backslash i}, a_{\backslash i}), \quad a_{\backslash i} = \frac{a_i}{1 - \pi_i a_i},\ h_{\backslash i} = h_i + a_{\backslash i}(\pi_i h_i - b_i).$$

Note that if $b_i = \pi_i = 0$, then $h_{\backslash i} = h_i,\ a_{\backslash i} = a_i$. This is always the case if EP is used in the context of IVM. Next, define $Z_i = \mathrm{E}_{\backslash i}[t_i(u_i)]$, where $\mathrm{E}_{\backslash i}[\cdot]$ is w.r.t. $Q_{\backslash i}$. Let

$$\alpha_i = \frac{\partial}{\partial h_{\backslash i}} \log Z_i, \quad \nu_i = -\frac{\partial^2}{\partial h_{\backslash i}^2} \log Z_i.$$

The computation of Z_i, α_i, ν_i may not be analytically tractable (depending on t_i), but we can always obtain a very accurate approximation using Gaussian quadrature (see Appendix B.2). Now, it is easy to see that

$$h'_i = h_{\backslash i} + a_{\backslash i}\alpha_i, \quad a'_i = (1 - a_{\backslash i}\nu_i)a_{\backslash i}.$$

Furthermore, we can obtain Q' as new approximation Q by updating the site parameters as

$$\pi_i' = \frac{\nu_i}{1 - a_{\backslash i}\nu_i}, \quad b_i' = \pi_i'(h_{\backslash i} + \alpha_i/\nu_i) = \frac{h_{\backslash i}\nu_i + \alpha_i}{1 - a_{\backslash i}\nu_i}.$$

This completes the description of a single EP update. In the context of the IVM, we always start from $b_i = \pi_i = 0$ and never visit a site twice.

As an example, consider the binary classification probit noise model $P(y_i|u_i) = \Phi(y_i(u_i + \beta))$, Φ the c.d.f. of $N(0,1)$. We have

$$Z_i = \int \Phi(y_i(u_i + \beta))Q_{\backslash i}(u_i)\, du_i = \Phi\left(\frac{y_i(h_{\backslash i} + \beta)}{\sqrt{1 + a_{\backslash i}}}\right),$$

and

$$z_i = \frac{y_i(h_{\backslash i} + \beta)}{\sqrt{1 + a_{\backslash i}}}, \quad \alpha_i = \frac{y_i N(z_i|0,1)}{\Phi(z_i)\sqrt{1 + a_{\backslash i}}}, \quad \nu_i = \alpha_i\left(\alpha_i + \frac{h_{\backslash i} + \beta}{1 + a_{\backslash i}}\right).$$

In practice, some care has to be taken towards numerical error. First, we compute $\log Z_i$ instead of Z_i. For the probit noise, we use code to compute $\log \Phi(z)$ directly. We also need to take care of "0/0" situations, an example is $N(z_i|0,1)/\Phi(z_i) = (d/dz_i)\log\Phi(z_i)$ in the term for α_i above.

Note that in general we cannot guarantee that an EP update actually can be done. Depending on t_i and the current $Q(\boldsymbol{u})$, it may be that the cavity distribution or $\hat{P}_i$ are degenerate or cannot be normalized. In the remainder of this Section, we show that such a breakdown cannot occur for a large class of frequently used site functions t_i, and that EP is a numerically stable algorithm in such cases. Namely, assume that $t_i(u_i)$ is *log-concave*, in that $\log t_i(u_i)$ is concave in u_i. A powerful theorem states that if $f(\boldsymbol{x})$ is a log-concave function of $\boldsymbol{x} \in \mathbb{R}^m$, then any marginal of f (obtained by integrating out some components of $\boldsymbol{x}$) is log-concave again (Bogachev, 1998). Now, if $Z_i = \mathrm{E}_{\backslash i}[t_i(u_i)]$, as long as $Q_{\backslash i}$ is a proper Gaussian, then Z_i is log-concave as a function of $h_{\backslash i}$. Namely, in this case both t_i and $Q_{\backslash i}$ are log-concave in $(u_i, h_{\backslash i})$, and the product of log-concave functions is log-concave. This means that $\nu_i \geq 0$ (as second derivative of the convex function $-\log Z_i$), so that if $1 - a_{\backslash i}\nu_i > 0$, we have that $\pi_i' \geq 0$. We can give an argument why we should typically have $\nu_i < a_{\backslash i}^{-1}$. Namely, if t_i is Gaussian with variance σ^2, then $\nu_i = 1/(a_{\backslash i} + \sigma^2) < 1/a_{\backslash i}$. If t_i is a Delta distribution at some point, we have $\nu_i = 1/a_{\backslash i}$. Therefore, we would argue that if t_i is not concentrated infinitely, then $1 - a_{\backslash i}\nu_i > 0$. We see that if we start with all $\pi_i = 0$, they remain nonnegative throughout for log-concave t_i, which in turn means that the update of the EP representation is numerically stable. The implications of log-concavity are the same in the case of the t_i depending on more than a single $u(\cdot)$ value.

B.2 Gaussian Quadrature

For non-Gaussian likelihood factors $t_i(u_i)$, EP updates (as described in Appendix B.1) require the solution of Gaussian expectations of the functions $u_i^k t_i(u_i)$, $k = 0, 1, 2$. These are univariate integrals with a Gaussian weight function which are typically not analytically tractable. If t_i depends on q components of $\boldsymbol{u}$, we need to solve q-dimensional integrals of the same form.

In the univariate case, the method of choice to approximate these expressions is *Gaussian quadrature* (of the Gauss-Hermite type) (Davis and Rabinovitz, 1984). This method delivers approximations to integrals

$$\int f(u)e^{-u^2}\,du,$$

requiring the evaluation of $f(u)$ at a few chosen points only. In a nutshell, Gaussian quadrature constructs the unique set of polynomials orthonormal w.r.t. the inner product

$$(f,g) = \int f(u)g(u)e^{-u^2}\,du,$$

uses the zeros of the degree N member as abscissas and adjusts the weights such that the rule is exact for $f(u)$ being a polynomial of order $\leq N-1$. It turns out that the corresponding rule is exact for all polynomials of order $\leq 2N-1$, owing to the d.o.f.'s for the free choice of abscissas (a particular feature of Gaussian quadrature). Thus, its general accuracy hinges on $f(u)$ being smooth such that it can be well-approximated by a polynomial in the region where $|f(u)e^{-u^2}|$ is significantly different from 0. If this does not hold true for $f(u)$ (in our case a suitable linear transformation of $t_i(u_i)$) it may be necessary to split the integral into parts or deviate from the standard quadrature rules. Another important point in practice, applying to the computation of Z_i for example, is that if $f(u) > 0$ and has a large dynamic range, we compute

$$\log I \approx \log \sum_k w_k f(u^{(k)}) = \log \sum_k \exp(\log f(u^{(k)}) + \log w_k),$$

using a special implementation of the *logsumexp* function $\boldsymbol{x} \mapsto \log \mathbf{1}^T \exp(\boldsymbol{x})$ which is numerically stable: $m + \log \mathbf{1}^T \exp(\boldsymbol{x} - m\mathbf{1})$, $m = \max\{x_i\}$.

When EP is applied to models where the likelihood factors t_i depend on $q > 1$ components of $\boldsymbol{u}$, the ability of doing the quadratures becomes crucial. First of all, if q is large, the quadratures are intractable in general. Note that the whole idea of EP hinges on the fact that q is much smaller than the size of $\boldsymbol{u}$. For q of moderate size, we can use so-called *product rules*, applying Gauss-Hermite quadrature along each dimension. The drawback of this approach is that without exploiting useful structure in t_i, these rules scale exponentially in q, since they need to evaluate t_i on a grid in $\mathbb{R}^q$. The method of *exact monomials* (Davis and Rabinovitz, 1984) requires polynomially many evaluations, but can be quite inaccurate. A special problem with such inaccurate rules (which does not occur for product rules) is that they violate moment constraints when applied to moment calculations. For example, variances may become negative or covariance matrices may become indefinite. When using such rules, it is certainly important to correct for such artefacts, Lerner (2002) gives an example. Some researchers suggest to evaluate the integrals by simply sampling from the Gaussian weight function. This is however very inaccurate, and a well-designed quadrature rule typically behaves much superior. For special likelihoods, we may be able to find good quadrature rules which scale polynomially in q, an example is the multivariate probit likelihood for multi-way classification (see Section 4.1).

Appendix C. Informative Vector Machine

In this Section we provide details about the representation of the IVM, its update after an inclusion, the marginal likelihood criterion and its gradient.

C.1 Update of the Representation

Recall the representation of the posterior approximation $Q(\boldsymbol{u})$ of the IVM from Section 3.1. Suppose a new point i is to be included into I, and its site parameters b_i and π_i have already been computed using the EP update. We have to update $\boldsymbol{L} \to \boldsymbol{L}' \in \mathbb{R}^{d+1,d+1}$ and $\boldsymbol{M} \to \boldsymbol{M}' \in \mathbb{R}^{n,d+1}$. First, $\boldsymbol{L}'_{1\ldots d,1\ldots d} = \boldsymbol{L}$ and $\boldsymbol{l}_i = \boldsymbol{L}'_{1\ldots d,d+1} = \boldsymbol{L}^{-1}\pi_i^{1/2}\boldsymbol{\Pi}^{1/2}\boldsymbol{K}_{I,i} = \pi_i^{1/2}\boldsymbol{M}_{i,\cdot}^T$, furthermore $l_i = \boldsymbol{L}'_{d+1,d+1} = (1 + \pi_i\boldsymbol{K}_{i,i} - \boldsymbol{l}_i^T\boldsymbol{l}_i)^{1/2} = (1 + \pi_i a_i)^{1/2}$. Next, $\boldsymbol{M}'_{\cdot,1\ldots d} = \boldsymbol{M}$ and $\boldsymbol{\mu} = \boldsymbol{M}'_{\cdot,d+1} = l_i^{-1}(\pi_i^{1/2}\boldsymbol{K}_{\cdot,i} - \boldsymbol{M}\boldsymbol{l}_i)$, furthermore $\boldsymbol{a}' = \boldsymbol{a} - (\mu_j^2)_j$. For the update of $\boldsymbol{h}$, note that $\boldsymbol{h}' = \boldsymbol{h} + \beta'_{d+1}\boldsymbol{\mu}$, where $\boldsymbol{\beta}'_{1\ldots d} = \boldsymbol{\beta}$ and $\beta'_{d+1} = \alpha_i l_i \pi_i^{-1/2}$. Here, α_i is the value from the EP update (see Appendix B.1). The cost is dominated by the computation of $\boldsymbol{K}_{\cdot,i}$ and the computation of $\boldsymbol{\mu}$, the latter is $O(nd)$.

C.2 Learning Criterion and Gradient

In this Section we derive the EP marginal likelihood criterion $\phi = -\log Q(D|\boldsymbol{\theta})$, where $Q(D|\boldsymbol{\theta})$ is given in Eq. 3, and show how to approximate its gradient w.r.t. $\boldsymbol{\theta}$. Recall that I is the active set index, and the selection index J is such that $J \cap I = \emptyset$. J is a strict subset of $\{1, \ldots, n\} \setminus I$ if a randomized selection strategy is used (see Section 3.2). Denote $\tilde{J} = J \cup I$. The criterion does not depend on patterns outside $\tilde{J}$.

We start with the criterion. We have

$$\log C_i = \log Z_i - \log \tilde{Z}_i, \quad Z_i = \mathrm{E}_{\setminus i}[t_i(u_i)],\ \log \tilde{Z}_i = \Phi[Q(u_i)] - \Phi[Q_{\setminus i}(u_i)].$$

Here,

$$\log \tilde{Z}_i = \frac{1}{2}\left(\log \frac{a_i}{a_{\setminus i}} + a_i^{-1}h_i^2 - a_{\setminus i}^{-1}h_{\setminus i}^2\right), \quad \frac{a_i}{a_{\setminus i}} = 1 - \pi_i a_i.$$

Some algebra gives

$$\log \tilde{Z}_i = \frac{1}{2}\left(\log(1 - \pi_i a_i) - \frac{\pi_i h_i^2 - 2h_i b_i + a_i b_i^2}{1 - \pi_i a_i}\right).$$

If $i \in J$, then $\log \tilde{Z}_i = 0$, because $Q^{\setminus i} = Q$. Furthermore,

$$\Phi[Q] - \Phi[P] = \frac{1}{2}\left(\log|2\pi\boldsymbol{A}| + \boldsymbol{h}^T\boldsymbol{A}^{-1}\boldsymbol{h} - \log|2\pi\boldsymbol{K}|\right) = \frac{1}{2}\left(-\log|\boldsymbol{B}| + \boldsymbol{h}_I^T\boldsymbol{b}\right),$$

because $\boldsymbol{h} = \boldsymbol{A}_{\cdot,I}\boldsymbol{b}$. Therefore,

$$\phi = -\sum_{i\in\tilde{J}} \log Z_i + \sum_{i\in I} \log \tilde{Z}_i + \frac{1}{2}\left(\log|\boldsymbol{B}| - \boldsymbol{h}_I^T\boldsymbol{b}\right). \tag{11}$$

The computation of ϕ requires a sweep over all sites in order to compute the $\log Z_i$.

As noted in Section 3.3, the exact gradient $\nabla_{\boldsymbol{\theta}}\phi$ cannot be computed analytically, due to the presence of Jacobian terms. Seeger (2005) shows that these terms drop out if EP is run to convergence on the site parameters, which is clearly not the case for IVM. We approximate the gradient by dropping the Jacobian terms anyway, even though $\boldsymbol{b}$, $\boldsymbol{\pi}$ have not been set appropriately. The consequences of optimization with an approximate gradient are discussed in Section 3.3.

Seeger (2005) allows P and Q to live in an expontial family with sufficient statistics $\phi(\boldsymbol{u})$ and natural parameters $\boldsymbol{\theta}$ (this notational ambiguity will only be used within this paragraph). For the Gaussian, the natural parameters are the ones used in the $N^U(\boldsymbol{u})$ form (see Appendix A.1). The *moment parameters* for some P are defined as $\boldsymbol{\eta} = \mathrm{E}_P[\phi(\boldsymbol{u})]$. If $\boldsymbol{\theta}^{(0)}$ are the natural parameters of the prior P, Seeger (2005) shows that if the EP fixed point conditions hold, then

$$\nabla_{\boldsymbol{\theta}^{(0)}}\phi = \boldsymbol{\eta}^{(0)} - \boldsymbol{\eta},$$

where $\boldsymbol{\eta}^{(0)}$, $\boldsymbol{\eta}$ are the mean parameters for P, Q respectively. This means that the indirect dependence on $\boldsymbol{\theta}^{(0)}$ through the site parameters can be ignored. If the active set I were all the data we observe, and if we ran EP on this data until convergence, this gradient would be correct. For our approximation, we assume that the dependence of ϕ on the hyperparameters through the site parameters (on I) can be ignored. Let $\boldsymbol{\eta}^{(i)} = \mathrm{E}_{\hat{P}_i}[\phi(\boldsymbol{u})]$. For $i \in J$, we have $\nabla_{\boldsymbol{\theta}^{(0)}} \log Z_i = \boldsymbol{\eta}^{(i)} - \boldsymbol{\eta}$, noting that $\boldsymbol{\eta}^{\backslash i} = \boldsymbol{\eta}$, because $Q^{\backslash i} = Q$, and that $\nabla_{\boldsymbol{\theta}^{(\backslash i)}} = \nabla_{\boldsymbol{\theta}^{(0)}}$ because the site parameters are assumed constant. For $i \in I$, we have (ignoring the indirect dependence) $\nabla_{\boldsymbol{\theta}^{(0)}}(\log Z_i - \log \tilde{Z}_i) = \boldsymbol{\eta}^{(i)} - \boldsymbol{\eta}^{\backslash i} - (\boldsymbol{\eta} - \boldsymbol{\eta}^{\backslash i}) = \boldsymbol{\eta}^{(i)} - \boldsymbol{\eta}$. This term is $\mathbf{0}$ upon EP convergence, but this need not be the case for the IVM. Therefore,

$$\nabla_{\boldsymbol{\theta}^{(0)}}\phi = -\sum_{i\in\tilde{J}} \boldsymbol{\eta}^{(i)} + (|\tilde{J}| - 1)\boldsymbol{\eta} + \boldsymbol{\eta}^{(0)}.$$

Furthermore, if α is some parameter of the likelihood $P(y|u)$, independent of $\boldsymbol{\theta}^{(0)}$, considering the direct dependence only gives

$$\frac{\partial\phi}{\partial\alpha} = -\sum_{i\in\tilde{J}} Z_i^{-1}\mathrm{E}_{\backslash i}\left[\frac{\partial t_i(u_i)}{\partial\alpha}\right]. \tag{12}$$

In many practically relevant cases, we can bring this into a form which we already had to compute during EP updates. For example, if $t_i(u_i;\alpha) = t_i(u_i + \alpha)$, then $\partial t_i/\partial\alpha = \partial t_i/\partial u_i$, and integration by parts gives

$$\mathrm{E}_{\backslash i}\left[\frac{\partial t_i(u_i)}{\partial\alpha}\right] = -\mathrm{E}_{\backslash i}\left[t_i(u_i)\left(\frac{\partial}{\partial u_i}\log Q_{\backslash i}(u_i)\right)\right], \quad \frac{\partial}{\partial u_i}\log Q_{\backslash i}(u_i) = -a_{\backslash i}^{-1}(u_i - h_{\backslash i}). \tag{13}$$

If $t_i(u_i;\alpha) = t_i(\alpha u_i)$, $\alpha > 0$, we have $\partial t_i/\partial\alpha = (\partial t_i/\partial u_i)u_i/\alpha$, and integration by parts gives

$$\mathrm{E}_{\backslash i}\left[\frac{\partial t_i(u_i)}{\partial\alpha}\right] = -\alpha^{-1}\mathrm{E}_{\backslash i}\left[t_i(u_i)\left(1 - u_i a_{\backslash i}^{-1}(u_i - h_{\backslash i})\right)\right].$$

For the prior, $\boldsymbol{\theta}^{(0)}$ is equal to $\boldsymbol{K}^{-1}$ (since the prior mean is always $\mathbf{0}$), and the correspoding sufficient statistics are $\phi(\boldsymbol{u}) = -(1/2)\boldsymbol{u}\boldsymbol{u}^T$.[13] Ignoring the dependence of the site

13. A small technical issue arises if J is a proper subset of $\{1,\ldots,n\}\setminus I$. In this case, the natural parameters of the prior are $(\boldsymbol{K}_{\tilde{J}})^{-1}$, and everything goes through if $\cdot$ is replaced by $\tilde{J}$.

parameters on $\boldsymbol{K}$, we have

$$\nabla_{\boldsymbol{K}^{-1}}\left(\log Z_i - \log \tilde{Z}_i\right) = -\frac{1}{2}\left(\hat{\boldsymbol{A}}^{(i)} - \boldsymbol{A} + \hat{\boldsymbol{h}}^{(i)}\hat{\boldsymbol{h}}^{(i)T} - \boldsymbol{h}\boldsymbol{h}^T\right), \quad i \in \tilde{J},$$

where $\hat{\boldsymbol{h}}^{(i)}$, $\hat{\boldsymbol{A}}^{(i)}$ are obtained as posterior mean and covariance matrix EP-including the single pattern i starting from $Q^{\backslash i}$. Let $\tilde{b}_i$, $\tilde{\pi}_i$ be the site parameters of i if we included it into I.[14] Let $\phi_1 = -\sum_{i\in\tilde{J}}(\log Z_i - \log \tilde{Z}_i)$ be the first criterion part. We start with $d(\log Z_i - \log \tilde{Z}_i)$ for $i \in I$, the result must be the same for $i \in J$ if $\pi_i = b_i = 0$. Let $\Delta_i = \tilde{\pi}_i - \pi_i$ and $l_i^2 = 1 + \Delta_i a_i > 0$, $\Delta_{b,i} = \tilde{b}_i - b_i$. We have that $\hat{\boldsymbol{A}}^{(i)} - \boldsymbol{A} = -\Delta_i/l_i^2 \boldsymbol{A}_{\cdot,i}\boldsymbol{A}_{\cdot,i}^T$ by the Sherman-Morrison formula, and $\hat{\boldsymbol{h}}^{(i)} = \boldsymbol{h} + l_i^{-2}(\Delta_{b,i} - \Delta_i h_i)\boldsymbol{A}_{\cdot,i}$. Note that

$$\boldsymbol{K}^{-1}\boldsymbol{A}_{\cdot,i} =: \boldsymbol{v}^{(i)} = \boldsymbol{\delta}_i - \boldsymbol{I}_{\cdot,I}\boldsymbol{\Pi}^{1/2}\boldsymbol{L}^{-T}\boldsymbol{M}_{i,\cdot}^T$$

and $\boldsymbol{K}^{-1}\boldsymbol{h} = \boldsymbol{I}_{\cdot,I}\boldsymbol{v}$, $\boldsymbol{v} = \boldsymbol{\Pi}^{1/2}\boldsymbol{L}^{-T}\boldsymbol{\beta}$. With $\rho_i = l_i^{-2}(\Delta_{b,i} - \Delta_i h_i)$, some algebra gives

$$\boldsymbol{K}^{-1}\left(\hat{\boldsymbol{A}}^{(i)} - \boldsymbol{A} + \hat{\boldsymbol{h}}^{(i)}\hat{\boldsymbol{h}}^{(i)T} - \boldsymbol{h}\boldsymbol{h}^T\right)\boldsymbol{K}^{-1} = 2\,\mathrm{sym}\,\rho_i\boldsymbol{I}_{\cdot,I}\boldsymbol{v}\boldsymbol{v}^{(i)T} + \left(\rho_i^2 - l_i^{-2}\Delta_i\right)\boldsymbol{v}^{(i)}\boldsymbol{v}^{(i)T}.$$

Some tedious algebra (using Appendix B.1) gives

$$\rho_i = \frac{h_i\pi_i - b_i + \tilde{\alpha}_i}{1 - a_i\pi_i}, \quad \frac{\Delta_i}{l_i^2} = \frac{\tilde{\nu}_i - \pi_i(1 - a_i\pi_i)}{(1 - a_i\pi_i)^2}.$$

Let $\boldsymbol{E}^{(2)} = \mathrm{diag}(-(1/2)(\rho_i^2 - l_i^{-2}\Delta_i))_i$ and $\boldsymbol{e}^{(1)} = (\rho_i)_i$, furthermore

$$\boldsymbol{E}^{(1)} = \boldsymbol{M}_{\tilde{J},\cdot}\boldsymbol{L}^{-1}\boldsymbol{\Pi}^{1/2} \in \mathbb{R}^{|\tilde{J}|,d}.$$

Then, using that $d\boldsymbol{K}^{-1} = -\boldsymbol{K}^{-1}(d\boldsymbol{K})\boldsymbol{K}^{-1}$, we have

$$\begin{aligned} d\phi_1 &= \mathrm{tr}\,\boldsymbol{E}^{(2)}(d\,\mathrm{diag}\,\boldsymbol{K}_{\tilde{J}}) \\ &\quad + \mathrm{tr}\left(-2\boldsymbol{E}^{(2)}\boldsymbol{E}^{(1)} - \boldsymbol{e}^{(1)}\boldsymbol{v}^T + \boldsymbol{I}_{\tilde{J},I}\left(\boldsymbol{E}^{(1)T}\boldsymbol{E}^{(2)}\boldsymbol{E}^{(1)} + \boldsymbol{E}^{(1)T}\boldsymbol{e}^{(1)}\boldsymbol{v}^T\right)\right)^T(d\boldsymbol{K}_{\tilde{J},I}). \end{aligned}$$

Next, $\phi_2 = \Phi[P] - \Phi[Q]$, whence

$$d\phi_2 = -\frac{1}{2}\,\mathrm{tr}\left(\boldsymbol{K} - \boldsymbol{A} - \boldsymbol{h}\boldsymbol{h}^T\right)(d\boldsymbol{K}^{-1}).$$

Since $\boldsymbol{K} - \boldsymbol{A} = \boldsymbol{M}\boldsymbol{M}$, we have that

$$d\phi_2 = \frac{1}{2}\,\mathrm{tr}\,\boldsymbol{L}^{-1}\boldsymbol{\Pi}^{1/2}(d\boldsymbol{K}_I)\boldsymbol{\Pi}^{1/2}\boldsymbol{L}^{-T} - \frac{1}{2}\boldsymbol{v}^T(d\boldsymbol{K}_I)\boldsymbol{v}.$$

14. For $i \in J$, we have $Q^{\backslash i} = Q$, so the EP update for $\tilde{b}_i$, $\tilde{\pi}_i$ is done from Q. For $i \in I$, the EP update is done from $Q^{\backslash i} \neq Q$. In this case, $\tilde{b}_i$, $\tilde{\pi}_i$ can be different from b_i, π_i, because EP was not run to convergence. Note that if EP was run to convergence on I, we would have $\hat{\boldsymbol{A}}^{(i)} = \boldsymbol{A}$ in this case.

Let $\boldsymbol{E}^{(3)} = \boldsymbol{\Pi}^{1/2}\boldsymbol{L}^{-T}$. Altogether, we have

$$d\phi = \operatorname{tr} \boldsymbol{E}^{(2)}(d \operatorname{diag} \boldsymbol{K}_{\tilde{J}}) + \operatorname{tr}\Big(-2\boldsymbol{E}^{(2)}\boldsymbol{E}^{(1)} - \boldsymbol{e}^{(1)}\boldsymbol{v}^T + \boldsymbol{I}_{\tilde{J},I}\Big(\boldsymbol{E}^{(1)T}\boldsymbol{E}^{(2)}\boldsymbol{E}^{(1)} + (1/2)\boldsymbol{E}^{(3)}\boldsymbol{E}^{(3)T} + (\boldsymbol{E}^{(1)T}\boldsymbol{e}^{(1)} - (1/2)\boldsymbol{v})\boldsymbol{v}^T\Big)\Big)^T (d\boldsymbol{K}_{\tilde{J},I}). \tag{14}$$

This expression can be computed in $O(|\tilde{J}|d^2)$. The expressions $d\boldsymbol{K}_{\tilde{J},I}$ and $d \operatorname{diag} \boldsymbol{K}_{\tilde{J}}$ depend on the covariance function and its parameters (see Appendix C.2.1 for examples). In order to compute the gradient, we first precompute the two matrices in the trace expression above, the large one overwrites $\boldsymbol{M}$. We then compute the gradient component by component, which is $O(nd)$ per hyperparameter, in addition to the computation of the kernel matrix derivatives.

Recall from Section 3.3 that the minimization of ϕ proceeds in major and minor steps, and that I, J are kept fixed during the latter, yet the site parameters $\boldsymbol{b}, \boldsymbol{\Pi}$ are updated. We note that the gradient computed here would be exact if we also kept $\boldsymbol{b}, \boldsymbol{\Pi}$ fixed between major steps. This observation is useful in order to check the gradient implementation by comparing with finite differences.

C.2.1 Kernel Derivatives

In Appendix C.2 we give the gradient $\nabla_{\boldsymbol{\theta}}\phi$ in terms of derivatives of the kernel matrix. In this Section, we show how the latter are computed for classes of commonly used kernels.

Isotropic covariance functions have the property that $K(\boldsymbol{x}, \boldsymbol{x}') = K(d)$, $d = \|\boldsymbol{x} - \boldsymbol{x}'\|$. Let $\boldsymbol{X} \in \mathbb{R}^{n,d}$, $\boldsymbol{Y} \in \mathbb{R}^{m,d}$ be given matrices of input points. If K has the form $f(-(1/2)d^2)$, the kernel matrix $\boldsymbol{K} = K(\boldsymbol{X}, \boldsymbol{Y})$ is of the form

$$f(\boldsymbol{A}), \quad \boldsymbol{A} = \boldsymbol{X}\boldsymbol{Y}^T - \frac{1}{2}\boldsymbol{d}_X\boldsymbol{1}^T - \frac{1}{2}\boldsymbol{1}\boldsymbol{d}_Y^T,\ \boldsymbol{d}_X = \operatorname{diag} \boldsymbol{X}\boldsymbol{X}^T,\ \boldsymbol{d}_Y = \operatorname{diag} \boldsymbol{Y}\boldsymbol{Y}^T$$

For example, the Gaussian (RBF) kernel of Eq. 9 is obtained for $f(-(1/2)d^2) = v \exp(-(w/2p)d^2)$, $v, w > 0$. For this kernel, we have

$$\frac{\partial}{\partial \log v}\boldsymbol{K} = \boldsymbol{K}, \quad \frac{\partial}{\partial \log w}\boldsymbol{K} = w/p\boldsymbol{A} \circ \boldsymbol{K}.$$

Note that these expressions are pointwise functions of the squared distance matrix $\boldsymbol{A}$. This characteristics hold true for other isotropic kernels as well. Our implementation uses this fact by storing $\boldsymbol{A}$ during the kernel computation, so that the derivatives can then be computed in $O(nm)$ only.

We are also interested in *anisotropic* covariance functions of the form $K(\boldsymbol{x}, \boldsymbol{x}') = K(d)$, $d = [(\boldsymbol{x} - \boldsymbol{x}')^T\boldsymbol{W}(\boldsymbol{x} - \boldsymbol{x}')]^{1/2}$, where $\boldsymbol{W}$ is diagonal with positive entries. Isotropic kernels are a special case with $\boldsymbol{W} = w\boldsymbol{I}$. The significance of anisotropic kernels is that *length scale* parameters w_i can be learned individually for each dimension. If $K(d) = f(-(1/2)d^2)$, we have $\boldsymbol{K} = K(\boldsymbol{X}, \boldsymbol{Y}) = f(\boldsymbol{A})$ with

$$\boldsymbol{A} = \boldsymbol{X}\boldsymbol{W}\boldsymbol{Y}^T - \frac{1}{2}\boldsymbol{d}_X\boldsymbol{1}^T - \frac{1}{2}\boldsymbol{1}\boldsymbol{d}_Y^T,\ \boldsymbol{d}_X = \operatorname{diag} \boldsymbol{X}\boldsymbol{W}\boldsymbol{X}^T,\ \boldsymbol{d}_Y = \operatorname{diag} \boldsymbol{Y}\boldsymbol{W}\boldsymbol{Y}^T.$$

The squared-exponential kernel ofEq. 10 is obtained for $f(-(1/2)d^2) = v\exp(-(1/2)d^2)$, $v > 0$. For this kernel, we have

$$\frac{\partial}{\partial \log v}\boldsymbol{K} = \boldsymbol{K}, \quad \frac{\partial}{\partial \log w_i}\boldsymbol{K} = w_i\left(\boldsymbol{X}^{(j)}\boldsymbol{K}\boldsymbol{Y}^{(j)} - \frac{1}{2}\boldsymbol{X}^{(j)2}\boldsymbol{K} - \frac{1}{2}\boldsymbol{K}\boldsymbol{Y}^{(j)2}\right),$$
$$\boldsymbol{X}^{(j)} = \operatorname{diag}\boldsymbol{X}_{\cdot,j}, \quad \boldsymbol{Y}^{(j)} = \operatorname{diag}\boldsymbol{Y}_{\cdot,j}.$$

Again, maintaining $\boldsymbol{A}$ or $\boldsymbol{K}$ after the kernel matrix computation allows for an $O(nm)$ computation of the derivatives.

Appendix D. The Multivariate Probit Likelihood

In this Section, we show how the required expression of Eq. 6 can be computed efficiently (in $O(C)$) for the multivariate probit likelihood of Eq. 8. In this Section, we write $N(z) = N(z|0,1)$ for the standard normal density.

For a fixed $\boldsymbol{u}$, we can compute the likelihood of Eq. 8 by first conditioning on the single n_y:

$$P(y|\boldsymbol{u}) = \mathrm{E}_{n_y\sim N(0,1)}\left[\exp\left(\sum_{y'\neq y}\log\Phi\left(v_y - u_{y'} - p_{y'}\right)\right)\right],$$

where $\Phi(z)$ denotes the cumulative distribution function (c.d.f.) of $N(0,1)$. All but one of the intractable integrations can be done analytically.

Before we show how EP updates can be done for this site, we note that the multi-probit likelihood is a log-concave function, with the beneficial implications detailed in Appendix B.1. To prove this, we make use of the marginalization theorem again (Bogachev, 1998). The function

$$(\boldsymbol{u}, \boldsymbol{n}) \mapsto N(\boldsymbol{n}|\boldsymbol{0}, \boldsymbol{I})\mathrm{I}_{\{v_y\boldsymbol{1}\succeq\boldsymbol{v}\}}.$$

is log-concave for any $\boldsymbol{p}$, because the Gaussian density and indicator functions of convex sets are log-concave, and log-concavity is closed under multiplication. The multi-probit likelihood is obtained by integrating out $\boldsymbol{n}$, and is therefore log-concave itself.

For our application, we need to do expectations of $P(y|\boldsymbol{u})$ over the factorized $Q(\boldsymbol{u}) = N(\boldsymbol{h}, \operatorname{diag}\boldsymbol{a})$, which means taking an expectation over the single Gaussian variable $\boldsymbol{v} = \boldsymbol{u} + \boldsymbol{n} + \boldsymbol{p}$ which has independent components as well. Since the factors in the likelihood are coupled only through v_y, we end up with a one-dimensional Gaussian quadrature only. Namely, $v_{y'} \sim N(\mu_{y'}, \sigma^2_{y'})$ with $\mu_{y'} = h_{y'} + p_{y'}$, $\sigma^2_{y'} = a_{y'} + 1$. Let $z_{y'} = (v_y - \mu_{y'})/\sigma_{y'}$. Then, we have

$$\begin{aligned}\log Z = \log \mathrm{E}_Q\left[P(y|\boldsymbol{u})\right] &= \log \mathrm{E}_{v_y}\left[\exp\left(\sum_{y'\neq y}\log\Phi(z_{y'})\right)\right] \\ &= \log \mathrm{E}_{n_0\sim N(0,1)}\left[\exp\left(\sum_{y'\neq y}\log\Phi\left(\frac{\sigma_y n_0 + \mu_y - \mu_{y'}}{\sigma_{y'}}\right)\right)\right],\end{aligned}$$

which requires a one-dimensional Gauss-Hermite quadrature only, independent of C. Thus, $\log Z$ can be computed in $O(C)$ to high relative accuracy. As noted above, we use a stable *logsumexp* implementation.

Denote means and variances[15] of the tilted distribution $\hat{P}(\boldsymbol{u}) \propto P(y|\boldsymbol{u})Q(\boldsymbol{u})$ by $\boldsymbol{h}'$, $\boldsymbol{a}'$, and let $j \neq y$.

$$h'_j = \mathrm{E}_{v_y}\left[\exp\left(\sum_{y'\neq y,j} \log\Phi(z_{y'}) - \log Z\right)\mathrm{E}_{u_j}\left[u_j\,\Phi\left(v_y - u_j - p_j\right)\right]\right]. \tag{15}$$

Let $Z_j = \mathrm{E}_{u_j}[\Phi(v_y - u_j - p_j)] = \Phi(z_j)$. Then,

$$\mathrm{E}_{u_j}\left[u_j\,\Phi\left(v_y - u_j - p_j\right)\right] = Z_j(a_j\alpha_j + h_j), \quad \alpha_j = \frac{d\log Z_j}{dh_j} = \frac{-H(z_j)}{\sigma_j}.$$

Here, $H(z) = (d\log\Phi(z))/dz = N(z)/\Phi(z)$ is called *hazard function* in Statistics. This can be seen by noting that $dN(z)/dz = -zN(z)$, therefore integration by parts gives

$$\begin{aligned}\mathrm{E}_{n_0\sim N(0,1)}\left[n_0\Phi(An_0+B)\right] &= A\int N(n_0)N(An_0+B)\,dn_0 = AN(B|0,1+A^2)\\ &= \frac{A}{\sqrt{1+A^2}}N(C), \quad C = \frac{B}{\sqrt{1+A^2}}.\end{aligned}$$

Now,

$$\begin{aligned}h'_j &= \mathrm{E}_{v_y}\left[\exp\left(\sum_{y'\neq y}\log\Phi(z_{y'}) - \log Z\right)(a_j\alpha_j + h_j)\right]\\ &= h_j - \frac{a_j}{\sigma_j}\mathrm{E}_{v_y}\left[\exp\left(\sum_{y'\neq y,j}\log\Phi(z_{y'}) - \log Z + \log N(z_j)\right)\right].\end{aligned}$$

This requires a one-dimensional Gaussian quadrature. The quadrature terms have already been computed for $\log Z$. Note that the formula implies that $h'_j < h_j$ (for $j \neq y'$). For the variance we use the formula

$$\mathrm{E}_{n_0\sim N(0,1)}\left[n_0^2\Phi(An_0+B)\right] = \Phi(C)\left(1 - \frac{A^2}{1+A^2}CH(C)\right), \quad C = \frac{B}{\sqrt{1+A^2}},$$

obtained by integration by parts, noting that $n_0^2N(n_0) = N(n_0) - (d/dn_0)n_0N(n_0)$. We have

$$\begin{aligned}&\mathrm{E}_{u_j}\left[u_j^2\,\Phi\left(v_y - u_j - p_j\right)\right] = \mathrm{E}_{u_j}\left[(u_j - h_j + h_j)^2\,\Phi\left(v_y - u_j - p_j\right)\right]\\ &= Z_jh_j^2 + Z_ja_j\left(1 - \frac{a_j}{1+a_j}z_jH(z_j)\right) + 2h_jZ_ja_j\alpha_j\\ &= Z_j\left(h_j^2 + a_j\left(1 + \alpha_j\left(\frac{a_j}{\sigma_j}z_j + 2h_j\right)\right)\right).\end{aligned}$$

15. Covariances can be computed in the same way, but we do not need them in our application here.

Like in Eq. 15 we obtain

$$\mathrm{E}_{\hat{P}}\left[u_j^2\right] = h_j^2 + a_j - \frac{a_j}{\sigma_j}\mathrm{E}_{v_y}\left[\exp\left(\sum_{y'\neq y,j}\log\Phi(z_{y'}) - \log Z + \log N(z_j)\right)\left(2h_j + \frac{a_j}{\sigma_j}z_j\right)\right]$$
$$= h_j^2 + a_j + 2h_j(h_j' - h_j) - \frac{a_j^2}{\sigma_j^2}\mathrm{E}_{v_y}\left[\exp\left(\sum_{y'\neq y,j}\log\Phi(z_{y'}) - \log Z + \log N(z_j)\right)z_j\right].$$

Finally,

$$a_j' = a_j - (h_j' - h_j)^2 - \frac{a_j^2}{\sigma_j^2}\mathrm{E}_{v_y}\left[\exp\left(\sum_{y'\neq y,j}\log\Phi(z_{y'}) - \log Z + \log N(z_j|0,1)\right)z_j\right].$$

Again, this one-dimensional quadrature can use the terms computed for $\log Z$ already.

If $j = y$, we are left with

$$\mathrm{E}_{\hat{P}}\left[u_y^k\right] = \mathrm{E}_{u_y,n_y}\left[u_y^k\exp\left(\sum_{y'\neq y}\log\Phi\left(\frac{v_y - \mu_{y'}}{\sigma_{y'}}\right) - \log Z\right)\right],\ k = 1,2$$

which requires a two-dimensional quadrature (we use the Gauss-Hermite product rule).

References

V. Bogachev. *Gaussian Measures*. Mathematical Surveys and Monographs. American Mathematical Society, 1998.

Lehel Csató and Manfred Opper. Sparse online Gaussian processes. *Neural Computation*, 14:641–668, 2002.

P. Davis and P. Rabinovitz. *Methods of Numerical Integration*. Academic Press, 1984.

A. Faul and M. Tipping. Analysis of sparse Bayesian learning. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pages 383–389. MIT Press, 2002.

M. Girolami and S. Rogers. Variational bayesian multinomial probit regression with gaussian process priors. Technical report, University of Glasgow, 2005. To appear in Neural Computation.

M. Kuss and C. Rasmussen. Assessing approximate inference for binary Gaussian process classification. *Journal of Machine Learning Research*, 6:1679–1704, 2005.

N. Lawrence and M. Jordan. Semi-supervised learning via Gaussian processes. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems 17*. MIT Press, 2005.

N. Lawrence and J. Platt. Learning to learn with the informative vector machine. In C. Brodley, editor, *International Conference on Machine Learning 21*, pages 512–519. Morgan Kaufmann, 2004.

N. Lawrence, J. Platt, and M. Jordan. Extensions of the informative vector machine. In Winkler, Lawrence, and Niranjan, editors, *Deterministic and Statistical Methods in Machine Learning*. Springer, 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 15*, pages 609–616. MIT Press, 2003.

Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1: 541–551, 1989.

T. Leen, T. Dietterich, and V. Tresp, editors. *Advances in Neural Information Processing Systems 13*, 2001. MIT Press.

U. Lerner. *Hybrid Bayesian Networks*. PhD thesis, Stanford University, 2002.

D. MacKay. *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

P. McCullach and J.A. Nelder. *Generalized Linear Models*. Number 37 in Monographs on Statistics and Applied Probability. Chapman & Hall, 1st edition, 1983.

Thomas Minka. Expectation propagation for approximate Bayesian inference. In J. Breese and D. Koller, editors, *Uncertainty in Artificial Intelligence 17*. Morgan Kaufmann, 2001.

R. M. Neal. *Bayesian Learning for Neural Networks*. Number 118 in Lecture Notes in Statistics. Springer, 1996.

A. O'Hagan. Curve fitting and optimal design. *Journal of Roy. Stat. Soc. B*, 40(1):1–42, 1978.

Manfred Opper and Ole Winther. Gaussian processes for classification: Mean field algorithms. *Neural Computation*, 12(11):2655–2684, 2000.

J. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*. MIT Press, 1999.

William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C*. Cambridge University Press, 2nd edition, 1992.

J. Quinonero Candela and C. Rasmussen. A unifying view of sparse approximate Gaussian process regression. *Journal of Machine Learning Research*, 6:1935–1959, 2005.

C. Rasmussen and J. Quinonero Candela. Healing the relevance vector machine through augmentation. In L. De Raedt and S. Wrobel, editors, *International Conference on Machine Learning 22*. Morgan Kaufmann, 2005.

C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

Bernhard Schölkopf and Alexander Smola. *Learning with Kernels*. MIT Press, 1st edition, 2002.

M. Seeger. *Bayesian Gaussian Process Models: PAC-Bayesian Generalisation Error Bounds and Sparse Approximations*. PhD thesis, University of Edinburgh, July 2003. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

M. Seeger. Gaussian processes for machine learning. *International Journal of Neural Systems*, 14(2):69–106, 2004.

M. Seeger. Expectation propagation for exponential families. Technical report, University of California at Berkeley, 2005. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

M. Seeger. Gaussian process belief propagation. To appear in Machine Learning with Structured Outputs, 2006.

M. Seeger and O. Chapelle. Cross-validation optimization for structured hessian kernel methods. Submitted. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`., 2006.

M. Seeger and M. I. Jordan. Efficient sparse multi-way gaussian process classification. 2004a.

M. Seeger and M. I. Jordan. Sparse Gaussian process classification with multiple classes. Technical Report 661, Department of Statistics, University of California at Berkeley, 2004b. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

M. Seeger, M. I. Jordan, and Y.-W. Teh. Semiparametric latent factor models. Technical report, University of California at Berkeley, 2004. See `www.kyb.tuebingen.mpg.de/bs/people/seeger`.

M. Seeger, C. Williams, and N. Lawrence. Fast forward selection to speed up sparse Gaussian process regression. In C. Bishop and B. Frey, editors, *Workshop on Artificial Intelligence and Statistics 9*, pages 205–212, 2003. Electronic Proceedings (ISBN 0-9727358-0-1).

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

Alex Smola and Peter Bartlett. Sparse greedy Gaussian process regression. In Leen et al. (2001), pages 619–625.

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

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, 2004. To appear.

Y.-W. Teh, M. Seeger, and M. I. Jordan. Semiparametric latent factor models. In Z. Ghahramani and R. Cowell, editors, *Workshop on Artificial Intelligence and Statistics 10*, 2005.

Michael 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.

Christopher Williams and Matthias Seeger. The effect of the input density distribution on kernel-based classifiers. In P. Langley, editor, *International Conference on Machine Learning 17*, pages 1159–1166. Morgan Kaufmann, 2000.

Christopher K. I. Williams and Matthias Seeger. Using the Nyström method to speed up kernel machines. In Leen et al. (2001), pages 682–688.