Journal of Machine Learning Research 12 (2011) 111-139 Submitted 11/09; Revised 10/10; Published 1/11

Bayesian Generalized Kernel Mixed Models

Zhihua Zhang ZHZHANG@ZJU.EDU.CN
Guang Dai GUANG.GDAI@GMAIL.COM
College of Computer Science and Technology
Zhejiang University
Hangzhou, Zhejiang 310027, China

Michael I. Jordan JORDAN@CS.BERKELEY.EDU
Computer Science Division and Department of Statistics
University of California
Berkeley, CA 94720-1776, USA

Editor: Neil Lawrence

Abstract

We propose a fully Bayesian methodology for generalized kernel mixed models (GKMMs), which are extensions of generalized linear mixed models in the feature space induced by a reproducing kernel. We place a mixture of a point-mass distribution and Silverman's *g*-prior on the regression vector of a generalized kernel model (GKM). This mixture prior allows a fraction of the components of the regression vector to be zero. Thus, it serves for sparse modeling and is useful for Bayesian computation. In particular, we exploit data augmentation methodology to develop a Markov chain Monte Carlo (MCMC) algorithm in which the reversible jump method is used for model selection and a Bayesian model averaging method is used for posterior prediction. When the feature basis expansion in the reproducing kernel Hilbert space is treated as a stochastic process, this approach can be related to the Karhunen-Loève expansion of a Gaussian process (GP). Thus, our sparse modeling framework leads to a flexible approximation method for GPs.

Keywords: reproducing kernel Hilbert spaces, generalized kernel models, Silverman's g-prior, Bayesian model averaging, Gaussian processes

1. Introduction

Supervised learning based on reproducing kernel Hilbert spaces (RKHSs) has become increasingly popular since the support vector machine (SVM) (Vapnik, 1998) and its variants such as penalized kernel logistic regression models (Zhu and Hastie, 2005) have been proposed. Sparseness has also emerged as a significant theme generally associated with RKHS methods. The SVM naturally embodies sparseness due to its use of the hinge loss function. Penalized kernel logistic regression models, however, are not naturally sparse. Thus, Zhu and Hastie (2005) proposed a methodology that they refer to as the *import vector machine* (IVM), where a fraction of the training data—called *import vectors* by analogy to the support vectors of the SVM—are used to index kernel basis functions.

Kernel supervised learning methods can be unified using the tools of regularization theory (Hastie et al., 2001). The regularization term is usually defined as the L_1 or L_2 norm of the vector of regression coefficients. From a Bayesian standpoint, this term is obtained from assigning a Gaussian or Laplacian prior to the regression vector. Moreover, using logarithmic scoring rules (Bernardo and

Smith, 1994), a loss function can often be viewed as the negative conditional log-likelihood. This perspective leads to interpreting regularization methods in terms of maximum *a posteriori* (MAP) estimation, and has motivated recent Bayesian interpretations of kernel methods (Tipping, 2001; Sollich, 2001; Mallick et al., 2005; Chakraborty et al., 2005; Zhang and Jordan, 2006; Pillai et al., 2007; Liang et al., 2009; MacLehose and Dunson, 2009).

Although the use of either the hinge loss function or L_1 regularization is an effective tool for achieving sparsity in the frequentist paradigm (Vapnik, 1998; Tibshirani, 1996), in the Bayesian setting the corresponding prior yields posteriors that cannot be computed in closed form. In the Bayesian methods of Mallick et al. (2005), for example, since conjugate priors for the regression vector do not exist, a sampling methodology based on data augmentation was employed to update the regression vector. In the Bayesian lasso (Park and Casella, 2008) or the Bayesian elastic net (Li and Lin, 2010), Gibbs sampling was used, based on assumptions of normality and independence. Given that an appeal to sampling methods must be made, it is not clear that mimicking frequentist methods is the best way to achieve sparsity within the Bayesian paradigm. Indeed, explicit support-vector selection or variable selection is not straightforward for these existing Bayesian approaches, and sparsity is often enforced in an ad hoc manner via Bayesian credible intervals (Park and Casella, 2008; Li and Lin, 2010).

In this paper we propose *generalized kernel models* (GKMs) as a framework in which sparsity can be given an explicit treatment and in which a fully Bayesian methodology can be carried out. The GKM is derived from generalized linear models (GLMs) (McCullagh and Nelder, 1989) in the RKHS. We define *active vectors* to be those input vectors that are indexed by the nonzero components of the regression vector in GKMs.[1] We assign to the regression vector a mixture of the point-mass distribution and a prior which we refer to as the *Silverman g-prior* (Silverman, 1985). Our use of this prior is based on three facts. First, the Silverman *g*-prior can induce an empirical RKHS norm on the training data (see Section 2.2). Second, posterior consistency results are available for Bayesian estimation procedures based on the Silverman *g*-prior (Zhang et al., 2008). Third, the mixture of the point-mass prior and the Silverman *g*-prior allows a fraction of regression coefficients in question to be zero and thus provides an explicit Bayesian approach to the selection of active vectors.

It is worth noting that the Silverman *g*-prior is related to the Zellner *g*-prior (Zellner, 1986), which has been widely applied to Bayesian variable selection and Bayesian model selection (Smith and Kohn, 1996; George and McCulloch, 1997; Kohn et al., 2001; Nott and Green, 2004; Sha et al., 2004) because of its computational tractability in evaluating marginal likelihoods.

We develop Bayesian approaches to parameter estimation, model selection and response prediction for the GKM. In particular, motivated by the use of the data augmentation methodology in Bayesian GLMs (Albert and Chib, 1993; Holmes and Held, 2006), we exploit this methodology to devise an MCMC algorithm for our Bayesian GKMs. The algorithm uses a reversible jump procedure (Green, 1995) for the automatic selection of active vectors and a Bayesian model averaging method (Raftery et al., 1997) for the posterior prediction of future observations. We show that our algorithm is amenable to low-rank matrix update techniques (see Section 3.2) that make it computationally feasible even for large data sets.

Another development in Bayesian kernel methods is based on Gaussian processes (GPs), which provide a general approach to assigning prior distributions to functions for nonparametric modeling.

1. Our "active vectors" are the analogs of import vectors for the IVM and support vectors for the SVM.

In geostatistics, GPs have been seen numerous applications to spatial statistical analysis under the name of "kriging." Diggle et al. (1998) broadened the scope of kriging by exploiting the combination of kriging and GLMs. In the machine learning community, ideas related to kriging and its extensions have been widely exploited in Bayesian treatments of classification and regression problems (Williams and Barber, 1998; Neal, 1999; Rasmussen and Williams, 2006). In these problems the data in question are not necessarily spatial. A major concern with GPs is the computational burden for large data sets. Thus, sparse approximations, such as the "subset of regressors," the Nyström method, the informative vector machine, the "subset of data" and the "data squashing" technique, are generally used to mitigate the computational burden (Williams and Seeger, 2001; Smola and Bartlett, 2001; Lawrence et al., 2003; Snelson, 2007).

Building on existing connections between kernel methods and GP-based models (see, e.g., Pillai et al., 2007), we use the Karhunen-Loève expansion of the Gaussian process to explore relationships between our Bayesian GKMs and GP-based classification. In particular, we show that our reversible jump method can be used to implement a "subset of regressors" approximation method for GP-based classification.

The rest of this paper is organized as follows. Section 2 presents a Bayesian framework for kernel supervised learning. Sections 3 and 4 present the MCMC algorithm for fully Bayesian GKMs and sparse GP classifiers, respectively. The experimental analysis is then presented in Section 5. Two extensions and some conclusions are given in Sections 6 and 7, respectively.

2. A Bayesian Approach for Kernel Supervised Learning

We start with a supervised learning problem over a set of training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$ where $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^p$ is an input vector and y_i is a univariate continuous output for the regression problem or binary output for the classification problem. Our current concern is to learn a predictive function $f(\mathbf{x})$ from the training data.

Suppose $f = u + h \in (\{1\} + \mathcal{H}_K)$ where $\mathcal{H}_K$ is an RKHS. Estimating $f(\mathbf{x})$ from data is formulated as a regularization problem of the form

$$\min_{f \in \mathcal{H}_K} \left\{ \frac{1}{n} \sum_{i=1}^{n} L(y_i, f(\mathbf{x}_i)) + \frac{g}{2} \|h\|^2_{\mathcal{H}_K} \right\}, \tag{1}$$

where $L(y, f(\mathbf{x}))$ is a loss function, $\|h\|^2_{\mathcal{H}_K}$ is the RKHS norm and $g > 0$ is the regularization parameter. By the representer theorem (Wahba, 1990), the solution for (1) is of the form

$$f(\mathbf{x}) = u + \sum_{j=1}^{n} \beta_j K(\mathbf{x}, \mathbf{x}_j), \tag{2}$$

where u is called an offset term, $K(\cdot,\cdot)$ is the kernel function and the β_j are referred to as regression coefficients. Noticing that $\|h\|^2_{\mathcal{H}_K} = \sum_{i,j=1}^n K(\mathbf{x}_i, \mathbf{x}_j)\beta_i\beta_j$ and substituting (2) into (1) we obtain the minimization problem with respect to (w.r.t.) the u and β_j as

$$\min_{u,\beta} \left\{ \frac{1}{n} \sum_{i=1}^{n} L(y_i, u + \mathbf{k}_i'\beta) + \frac{g}{2} \beta' \mathbf{K} \beta \right\}, \tag{3}$$

where $\beta = (\beta_1, \ldots, \beta_n)'$ is an $n \times 1$ regression vector and $\mathbf{K} = [\mathbf{k}_1, \ldots, \mathbf{k}_n]$ is the $n \times n$ kernel matrix with $\mathbf{k}_i = (K(\mathbf{x}_i, \mathbf{x}_1), \ldots, K(\mathbf{x}_i, \mathbf{x}_n))'$. Since $\mathbf{K}$ is symmetric and positive semidefinite, the term $\beta'\mathbf{K}\beta$ is in fact an empirical RKHS norm w.r.t. the training data.

The predictive function $f(\mathbf{x})$ in (2) is based on a basis expansion of kernel functions. We now show that the predictive function can also be expressed by a basis expansion of feature functions. Given a Mercer reproducing kernel $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, there exists a corresponding mapping (say ψ) from the input space $\mathcal{X}$ to a feature space (say $\mathcal{F} \subset \mathbb{R}^r$). That is, we have a vector-valued function $\psi(\mathbf{x}) = (\psi_1(\mathbf{x}), \ldots, \psi_r(\mathbf{x}))'$, which is called the *feature vector* of $\mathbf{x}$, such that $K(\mathbf{x}_i, \mathbf{x}_j) = \psi(\mathbf{x}_i)'\psi(\mathbf{x}_j)$. By the *Mercer-Hilbert-Schmidt Theorem* (Wahba, 1990), we know that there exists an orthogonal sequence of continuous eigenfunctions $\{\phi_j\}$ in the square integrable Hilbert functional space $L_2(\mathcal{X})$ and eigenvalues $l_1 \geq l_2 \geq \ldots \geq 0$. Furthermore, we have a definition of the feature functions $\psi : \mathcal{X} \to L_2(\mathcal{X})$ as $\psi(\mathbf{x}) = \left\{\sqrt{l_j}\phi_j(\mathbf{x})\right\}_{j=1}^r$. That is, $\psi_j(\mathbf{x}) = \sqrt{l_j}\phi_j(\mathbf{x})$. Thus the $\psi_j(\mathbf{x})$ constitute a set of basis functions of $L_2(\mathcal{X})$. Consequently, they can be used to express the predictive function as follows:

$$f(\mathbf{x}) = u + \sum_{k=1}^{r} b_k \psi_k(\mathbf{x}) = u + \psi(\mathbf{x})'\mathbf{b}, \tag{4}$$

where $\mathbf{b} = (b_1, \ldots, b_r)'$. There are possibly infinitely many basis functions in (4) because r is possibly infinite. In the case that r is infinite, one may use a finite-dimensional approximation to $f(\mathbf{x})$ by keeping the first n $\psi_j(\mathbf{x})$'s and setting the remaining b_j, $j > n$ to zero (Zhang et al., 2007). Now letting $\mathbf{b} = \Psi'\beta$, we re-derive (2) from (4) due to $\mathbf{K} = \Psi\Psi'$ where $\Psi = [\psi(\mathbf{x}_1), \ldots, \psi(\mathbf{x}_n)]'$.

2.1 Generalized Kernel Models

Using the logarithmic scoring rule (Bernardo and Smith, 1994), the loss $L(y, f(\mathbf{x}))$ can be viewed as a negative conditional log-likelihood. This motivates us to construct the following model

$$y \sim p(y|\mu) \quad \text{with} \quad \mu = \tau(u + \mathbf{k}'\beta), \tag{5}$$

where $\tau(\cdot)$ is a known link function and $\mathbf{k} = (K(\mathbf{x}, \mathbf{x}_1), \ldots, K(\mathbf{x}, \mathbf{x}_n))'$. This model can be obtained from the model

$$y \sim p(y|\mu) \quad \text{with} \quad \mu = \tau(u + \psi(\mathbf{x})'\mathbf{b}) \tag{6}$$

by using the transformation $\mathbf{b} = \Psi'\beta$. Since the model in (6) is a GLM in the feature space, we call model (5) the *generalized kernel model* (GKM).

GKMs provide a unifying framework for kernel-based regression and classification. With different $p(y|\mu)$ and τ, we have different kernel models. In the regression problem, $p(y|\mu)$ is usually normal and τ is the identity function.

In this paper we are mainly concerned with the classification problem where y is encoded as a binary value, that is, $y \in \{0, 1\}$. We thus model $p(y|\mu)$ as Bernoulli distribution:

$$p(y|\mu) = \mu^y(1-\mu)^{1-y} = [\tau(u + \mathbf{k}'\beta)]^y[1 - \tau(u + \mathbf{k}'\beta)]^{1-y}.$$

Typically, τ is either the logistic link $\tau(z) = \frac{\exp(z)}{1+\exp(z)}$ or the probit link $\tau(z) = \Phi(z)$, the cumulative distribution function of a standard normal variable. The probit link is widely used in Bayesian GLMs due to its tractability in calculating the marginal likelihood. In our fully Bayesian GKMs in Section 3, we will use this link.

2.2 Silverman's g-prior

Assume that the b_k are independent Gaussian variables with $E(b_k) = 0$ and $E(b_k^2) = g^{-1}$, that is, $\mathbf{b} \sim N_r(\mathbf{0},\, g^{-1}\mathbf{I}_r)$. Here and later, we denote by $\mathbf{I}_m$ the $m \times m$ identity matrix, by $\mathbf{1}_m$ the $m \times 1$ vector

of ones, and by $\mathbf{0}$ the zero vector or matrix with appropriate size. Because of $\mathbf{b} = \Psi'\beta$, we have $\beta = \mathbf{K}^{-1}\Psi\mathbf{b}$. As a result, the prior for β is $\beta \sim N_n\left(\mathbf{0},\, g^{-1}\mathbf{K}^{-1}\right)$ due to $\mathbf{K}^{-1}\Psi\Psi'\mathbf{K}^{-1} = \mathbf{K}^{-1}$. It is possible that the kernel matrix $\mathbf{K}$ is singular. For such a $\mathbf{K}$, we use its Moore-Penrose inverse $\mathbf{K}^{+}$ instead and still have $\mathbf{K}^{+}\mathbf{K}\mathbf{K}^{+} = \mathbf{K}^{+}$. The prior distribution for β becomes a singular normal distribution (Mardia et al., 1979). In either case, we use $\mathbf{K}^{-1}$ for notational simplicity.

The prior $N_n\left(\mathbf{0},\, \mathbf{K}^{-1}\right)$ for β was first proposed by Silverman (1985) in his Bayesian formulation of spline smoothing. Thus, Zhang et al. (2008) referred to the prior $\beta \sim N_n\left(\mathbf{0},\, g^{-1}\mathbf{K}^{-1}\right)$ as the *Silverman g-prior* because it is related to the Zellner *g*-prior (Zellner, 1986). Since the prior density of β is proportional to $\exp(-g\beta'\mathbf{K}\beta/2)$, the Silverman *g*-prior is design-dependent. Moreover, the regularization term $g\beta'\mathbf{K}\beta/2$ in (3) is readily derived from this prior.

When $\mathbf{K}$ is singular, by analogy to the *generalized singular g-prior* (*gsg*-prior) (West, 2003) we call $N_n\left(\mathbf{0},\, g^{-1}\mathbf{K}^{-1}\right)$ a *generalized Silverman g-prior*. It is worth pointing out that Green (1985) argued that the definition of Silverman's prior is implicit. We have presented an explicit derivation of this prior. Like the Zellner *g*-prior (Zellner, 1986; Liang et al., 2008), the Silverman *g*-prior has only a single shared global scaling parameter *g*. Thus, the prior induces a global shrinkage rule.

2.3 Sparse Models

Recall that the number of active vectors is equal to the number of nonzero components of β. That is, if $\beta_j = 0$, the jth input vector is excluded from the basis expansion in (2), otherwise the jth input vector is an active vector. We are thus interested in a prior for β which allows some components of β to be zero. In particular, we assign a point-mass mixture prior to β built on the Silverman *g*-prior.

We introduce an indicator binary vector $\gamma = (\gamma_1, \ldots, \gamma_n)'$ such that $\gamma_j = 1$ if $\mathbf{x}_j$ is an active vector and $\gamma_j = 0$ if it is not. Let $n_\gamma = \sum_{j=1}^{n}\gamma_j$ be the number of active vectors, and let $\mathbf{K}_\gamma$ be the $n \times n_\gamma$ submatrix of $\mathbf{K}$ consisting of those columns of $\mathbf{K}$ for which $\gamma_j = 1$. We further let $\mathbf{K}_{\gamma\gamma}$ be the $n_\gamma \times n_\gamma$ submatrix of $\mathbf{K}_\gamma$ consisting of those rows of $\mathbf{K}_\gamma$ for which $\gamma_j = 1$, and β_γ and $\mathbf{k}_\gamma$ be the corresponding $n_\gamma \times 1$ subvectors of β and $\mathbf{k}$. Based on GKMs in (5) and the Silverman *g*-prior, we thus obtain the following sparse model

$$y \sim p(y|\tau(f(\mathbf{x}))) \quad \text{with} \quad f(\mathbf{x}) = u + \mathbf{k}'_\gamma\beta_\gamma \text{ and } \beta_\gamma \sim N_{n_\gamma}(\mathbf{0},\, g^{-1}\mathbf{K}_{\gamma\gamma}^{-1}). \tag{7}$$

In the existing literature for Bayesian sparse classification and regression (Tipping, 2001; Figueiredo, 2003; Park and Casella, 2008; Hans, 2009; Li and Lin, 2010; Carvalho et al., 2010), a typical choice of the prior on β is the class of multivariate scale mixtures of normals. The resulting shrinkage rule is derived by mixing over a set of local scaling parameters. This differ from our global shrinkage rule. See Carvalho et al. (2010) for further discussion of sparsity priors.

3. Methodology

In this section we present a fully Bayesian GKM (FBGKM) based on (7). Since $p(y|\tau(f(\mathbf{x})))$ is non-normal for the classification problem, conjugate priors for β usually do not exist. In order to facilitate the implementation of Bayesian inference in this setting, we make use of the data augmentation methodology which has been used by Albert and Chib (1993) for Bayesian GLMs and by Mallick et al. (2005) for their Bayesian SVMs. The basic idea is to introduce auxiliary variables linking y and the model parameters. We apply this methodology to our FBGKM.

3.1 Hierarchical Models

Let $\mathbf{s} = (s_1, \ldots, s_n)'$ be a vector of auxiliary variables corresponding to the training data $\{(\mathbf{x}_i, y_i)\}_{i=1}^n$. We in particular define

$$\mathbf{s} = u\mathbf{1}_n + \mathbf{K}_\gamma \beta_\gamma + \varepsilon \text{ with } \varepsilon \sim N_n(\mathbf{0}, \sigma^2 \mathbf{I}_n).$$

Since τ is defined as the probit link in our FBGKM, we have $\sigma^2 = 1$ and

$$y_i = \begin{cases} 1 & \text{if } s_i > 0 \\ 0 & \text{otherwise.} \end{cases}$$

Given $\mathbf{s}$, $\mathbf{y} = (y_1, \ldots, y_n)'$ is independent of u, β and γ. Consequently, we can assign conjugate priors for these parameters and perform an efficient Bayesian inference.

Firstly, we assume $u \sim N(0, \eta^{-1})$ and $g \sim Ga(a_g/2, b_g/2)$ where $Ga(a,b)$ represents a gamma distribution. Let $\widetilde{\beta}_\gamma = (u, \beta_\gamma')'$. We thus have

$$\widetilde{\beta}_\gamma \sim N_{n_\gamma+1}(\mathbf{0}, \Sigma_\gamma^{-1}) \text{ with } \Sigma_\gamma = \begin{bmatrix} \eta & \mathbf{0} \\ \mathbf{0} & g\mathbf{K}_{\gamma\gamma} \end{bmatrix}.$$

By integrating out $\widetilde{\beta}_\gamma$, the marginal distribution of $\mathbf{s}$ conditional on γ is normal, namely,

$$p(\mathbf{s}|\gamma) = N_n(\mathbf{0}, \mathbf{Q}_\gamma) \tag{8}$$

with $\mathbf{Q}_\gamma = \mathbf{I}_n + \widetilde{\mathbf{K}}_\gamma \Sigma_\gamma^{-1} \widetilde{\mathbf{K}}_\gamma'$ where $\widetilde{\mathbf{K}}_\gamma = [\mathbf{1}_n, \mathbf{K}_\gamma]$ $(n \times (n_\gamma + 1))$. Bayes theorem yields the following distribution of $\widetilde{\beta}_\gamma$ conditional on $\mathbf{s}$ and γ:

$$[\widetilde{\beta}_\gamma | \mathbf{s}, \gamma] \sim N_{n_\gamma+1}(\Upsilon_\gamma^{-1} \widetilde{\mathbf{K}}_\gamma' \mathbf{s}, \Upsilon_\gamma^{-1}), \tag{9}$$

where $\Upsilon_\gamma = \widetilde{\mathbf{K}}_\gamma' \widetilde{\mathbf{K}}_\gamma + \Sigma_\gamma$.

Secondly, the kernel function K is assumed to be indexed by hyperparameters θ (see, e.g., Mallick et al., 2005). For example, the Gaussian kernel $K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\theta^2)$ is a function of the width parameter θ. For simplicity, the dependence of K on θ will be left implicit henceforth. If θ is p-dimensional, we take a uniform prior for each element of θ on $[a_{\theta_j}, b_{\theta_j}]$. Namely,

$$\theta \sim \prod_{j=1}^p U(a_{\theta_j}, b_{\theta_j}).$$

Thirdly, as in Kohn et al. (2001) and Nott and Green (2004), we assign an independent Bernoulli prior to each component of γ, namely,

$$p(\gamma|\alpha) = \prod_{j=1}^n \alpha^{\gamma_j} (1-\alpha)^{1-\gamma_j} = \alpha^{n_\gamma} (1-\alpha)^{n-n_\gamma},$$

where $\alpha \in (0,1)$. It is natural to place a Beta prior on α, $\alpha \sim B(a_\alpha, b_\alpha)$. Marginalizing out α results in the following prior on γ:

$$p(\gamma) = \frac{Be(n_\gamma + a_\alpha, n - n_\gamma + b_\alpha)}{Be(a_\alpha, b_\alpha)}, \tag{10}$$

where $Be(\cdot,\cdot)$ is the Beta function. Kohn et al. (2001) proposed a method of selecting the hyperparameters a_α and b_α by controlling the value of n_γ. In the following experiments, we use the uninformative fixed specification $a_\alpha = 1$ and $b_\alpha = 1$.

Finally, we assume that η follows $Ga(a_\eta/2,\ b_\eta/2)$ and we shall keep the hyperparameters a_η, b_η, a_g and b_g fixed in this paper. In summary, we form a hierarchical model in which the joint density of all variables mentioned takes the form

$$p(\mathbf{y},\mathbf{s},\gamma,u,\beta,\theta,\eta,g) = p(\eta)p(g)p(\gamma)p(\theta)p(u|\eta)p(\beta|g,\gamma,\theta)p(\mathbf{s}|u,\beta,\theta,\gamma)p(\mathbf{y}|\mathbf{s}).$$

The corresponding directed acyclic graph is shown in Figure 1.

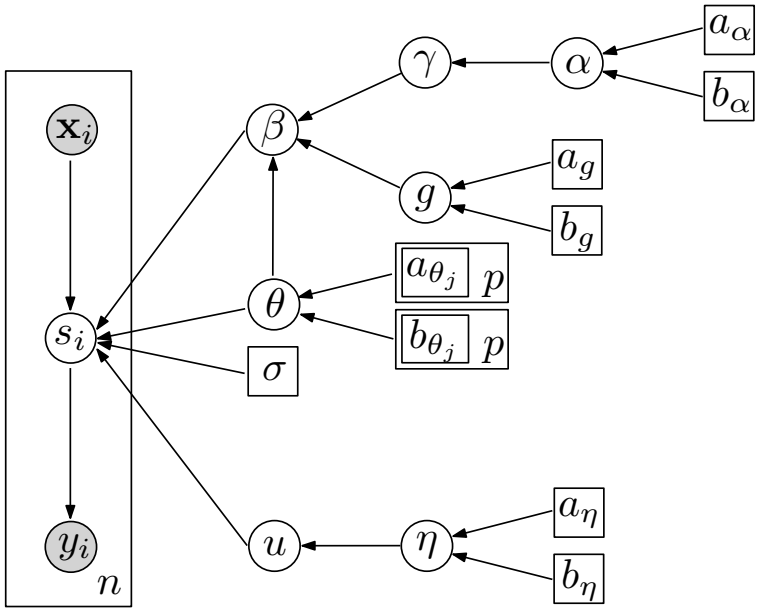


Figure 1: A graphical representation for the hierarchical model.

3.2 Inference

Our goal is to generate realizations of parameters from the conditional joint density $p(\mathbf{s},u,\beta,\gamma,g|\mathbf{y})$ via an MCMC algorithm. In order to speed up mixing of the MCMC, we use marginal posterior distributions whenever possible. Our MCMC algorithm consists of the following steps.

Start Give a_η, b_η, a_g and b_g, and initialize $\mathbf{s}$, γ, g, η, u and β_γ.

Step (a) Impute each s_i from $p(s_i|y_i,u,\beta_\gamma)$.

Step (b) Update η, g, $\widetilde{\beta}_\gamma$ and θ according to $p(\eta|u)$, $p(g|\beta_\gamma)$, $p(\widetilde{\beta}_\gamma|\mathbf{s},\gamma,\eta,g)$ and $p(\theta|\mathbf{s},\gamma)$, respectively.

Step (c) Update γ from $p(\gamma|\mathbf{s})$.

Step (a) is to draw $\mathbf{s}$ from $p(\mathbf{s}|\mathbf{y}, u, \beta_\gamma)$. We perform this step by using a technique which was proposed by Holmes and Held (2006) for the conventional probit regression. In particular, $\mathbf{s}$ is updated from its marginal distribution having integrated over $\tilde{\beta}_\gamma$; that is, s_i is generated from $p(s_i|\mathbf{s}_{-i}, y_i, \gamma)$ where $\mathbf{s}_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)'$. The details of this procedure is given in Appendix A. Please also refer to Holmes and Held (2006).

We now consider the updates of $\widetilde{\beta}_\gamma$, η and g. Given $\mathbf{s}$, these parameters are independent of $\mathbf{y}$, so their updates are based on $p(\widetilde{\beta}_\gamma, \eta, g|\mathbf{s}, \gamma)$. Hence, we update $\widetilde{\beta}_\gamma$ from $[\widetilde{\beta}_\gamma|\mathbf{s}, \gamma, \eta, g] \sim N_{n_\gamma+1}(\Upsilon_\gamma^{-1}\widetilde{\mathbf{K}}_\gamma'\mathbf{s},\ \Upsilon_\gamma^{-1})$. Since g is only dependent on β_γ and the prior is conjugate, we use the Gibbs sampler to update g from its conditional distribution, which is given by

$$[g|\beta_\gamma] \sim Ga\Big(\frac{a_g+n_\gamma}{2}, \frac{b_g+\beta_\gamma'\mathbf{K}_{\gamma\gamma}\beta_\gamma}{2}\Big).$$

The update of η is obtained from its conditional distribution as

$$[\eta|u] \sim Ga\Big(\frac{a_\eta+1}{2}, \frac{b_\eta+u^2}{2}\Big).$$

In order to update θ, we need to use an MH sampler. We write the marginal conditional distribution of θ as

$$p(\theta|\mathbf{s}, \gamma) \propto p(\mathbf{s}|\gamma, \eta, g, \theta)p(\theta),$$

where $p(\mathbf{s}|\gamma, \eta, g, \theta)$ is given by (8). In the following experiments (see Section 5.3), the proposal distribution is specified as a Gaussian distribution with the current value of θ as mean and 0.2 as variance. Let θ^* denote the proposed move from the current θ. Then this move is accepted with probability

$$\min\Big\{1,\ \frac{p(\mathbf{s}|\gamma, \eta, g, \theta^*)}{p(\mathbf{s}|\gamma, \eta, g, \theta)}\Big\}.$$

This acceptance probability involves the calculations of the inverses and determinants of both $\mathbf{Q}_\gamma$ and $\mathbf{Q}_\gamma^*$, where $\mathbf{Q}_\gamma^*$ is obtained from $\mathbf{Q}_\gamma$ with θ^* replacing θ. To reduce computational costs, we employ the formulas in (11) which is given below for computing these inverses and determinants. Our Bayesian estimation method for the kernel parameter θ is more efficient than that given in BSVM and CSVM (Mallick et al., 2005), in which computing the inverses and determinants of two consecutive full kernel matrices $\mathbf{K}$ and $\mathbf{K}^*$ is required at each sweep of MCMC sampling.

Step (c) is used for the automatic choice of active vectors. To implement this step, we borrow a method devised by Nott and Green (2004). This method was derived from the reversible jump methodology of Green (1995). Specifically, we generate a proposal γ^* from the current value of γ by one of three possible moves:

Birth move randomly choose a 0 in γ and change it to 1;

Death move randomly choose a 1 in γ and change it to 0;

Swap move randomly choose a 0 and a 1 in γ and switch them.

The acceptance probability for each move is

$$\min\{1, \text{likelihood ratio} \times \text{prior ratio} \times \text{proposal ratio}\}.$$

Letting $k = n_\gamma$, we denote the probabilities of birth, death and swap by b_k, d_k and $1-b_k-d_k$, respectively. For birth, death and swap moves, the acceptance probabilities are

$$\min\left\{1, \frac{p(\mathbf{s}|\gamma^*)p(\gamma^*)d_{k+1}(n-k)}{p(\mathbf{s}|\gamma)p(\gamma)b_k(k+1)}\right\},$$

$$\min\left\{1, \frac{p(\mathbf{s}|\gamma^*)p(\gamma^*)b_{k-1}k}{p(\mathbf{s}|\gamma)p(\gamma)d_k(n-k+1)}\right\},$$

$$\min\left\{1, \frac{p(\mathbf{s}|\gamma^*)p(\gamma^*)}{p(\mathbf{s}|\gamma)p(\gamma)}\right\},$$

where $p(\mathbf{s}|\gamma)$ and $p(\gamma)$ are given in (8) and (10). In our experiments we set $b_0 = 1$ and $d_0 = 0$, $b_k = d_k = 0.3$ for $1 \le k \le k_{\max}-1$, and $d_k = 1$ and $b_k = 0$ for $k_{\max} \le k \le n$. Here, $k_{\max}$ is a specified maximum number of active vectors such that $k_{\max} \le n$.

An alternative to this approach is the stochastic search method of George and McCulloch (1997). This method also employs birth, death and swap moves; it differs from the reversible jump procedure because it does not incorporate the probabilities of birth, death and swap into its acceptance probabilities.

Recall that the main computational burden of our MCMC algorithm comes from the calculations of the determinant and inverse of $\mathbf{Q}_\gamma$ ($\mathbf{Q}_{\gamma^*}$) during the MCMC sweeps. It is worth noting that when n is relatively large, we can reduce the computational burden by giving $k_{\max}$ a value far less than n, that is, $k_{\max} \ll n$, and then computing:

$$\mathbf{Q}_\gamma^{-1} = \mathbf{I}_n - \widetilde{\mathbf{K}}_\gamma \Upsilon_\gamma^{-1} \widetilde{\mathbf{K}}_\gamma' \quad \text{and} \quad |\mathbf{Q}_\gamma| = |\Upsilon_\gamma||\Sigma_\gamma|^{-1} = \eta^{-1} g^{-n_\gamma} |\mathbf{K}_{\gamma\gamma}|^{-1} |\Upsilon_\gamma|. \tag{11}$$

For example, for both the USPS and NewsGroups data sets used in our experiments, we set $k_{\max} = 200 \ll n$. In this setting, we always have $n_\gamma \le k_{\max} \ll n$. Since Υ_γ and $\mathbf{K}_{\gamma\gamma}$ are $(n_\gamma+1)\times(n_\gamma+1)$ and $n_\gamma \times n_\gamma$, these formulas for $\mathbf{Q}_\gamma^{-1}$ and $|\mathbf{Q}_\gamma|$ are feasible computationally. This is an advantage over the stochastic search method of George and McCulloch (1997). Finally, in the reversible jump method, the matrices obtained before and after each move only change a column and a row. Thus, it is possible to exploit rank-one matrix update techniques to make the method still more efficient.

3.3 Prediction

Given a new input vector $\mathbf{x}_*$, we need to predict its label y_*. The posterior predictive distribution of y_* is

$$p(y_*|\mathbf{x}_*, \mathbf{y}) = \int p(y_*|\mathbf{x}_*, \tilde{\beta}_\gamma, \mathbf{y}) p(\tilde{\beta}_\gamma|\mathbf{y}) d\tilde{\beta}_\gamma.$$

We know that this integral cannot be computed in closed form. Moreover, it is intractable to select the model which is parameterized by β_γ for prediction. An intuitive approach is to choose a model with a value of γ having the highest posterior probability among those γ that appear during the MCMC sweeps. However, this is expensive in terms of memory because γ takes 2^n possible distinct values. To deal with this problem, we use a Bayesian model averaging method (Raftery et al., 1997) for posterior prediction.

The Bayesian model averaging method is based on the MCMC sampling process. Specifically, we have

$$p(y_* = 1|\mathbf{x}_*, \mathbf{y}) \approx \frac{1}{T}\sum_{t=1}^{T} p\left(y_* = 1 \middle| \mathbf{y}, \mathbf{x}_*, u^{(t)}, \beta_\gamma^{(t)}\right).$$

Here $(\cdot)^{(t)}$ is the tth MCMC realization of $(\cdot)$, which is taken at every Mth sweep after the burn-in of the MCMC algorithm. In the following experiments, we run the MCMC algorithm for $10,000$ sweeps, discard the first $5,000$ as the burn-in, and retain every 5th (i.e., $M = 5$) realization of parameters after the burn-in for inference and prediction. This implies that the Bayesian model averaging method uses $1,000$ ($T = (10,000-5,000)/5$) active sets for prediction.

We should point out that our Bayesian model does not treat the training and test as two separate procedures. In fact, our reversible jump MCMC algorithm deals with parameter estimation, model selection and posterior prediction jointly in a single paradigm. Moreover, the reversible jump method is a sequential approach for model selection and posterior prediction. This implies that after the burn-in the selection of active vectors and the prediction of responses are simultaneously implemented. Thus, the MCMC algorithm does not require extra computational complexity for the prediction of responses.

4. Sparse Gaussian Processes for Classification

In nonparametric Bayesian methods for regression and classification, $f(\mathbf{x})$ is directly regarded as a stochastic function; in particular, $f(\mathbf{x})$ is often modeling as a Gaussian process. There has been much discussion of the relationships between RKHS-based methods and GP-based methods (see, e.g., Rasmussen and Williams, 2006; Pillai et al., 2007). In this section we further investigate this relationship and then propose an effective and efficient GP-based classification method.

4.1 Gaussian Process Priors

The following proposition summarizes the connection between the Gaussian process and the feature basis expansion $\sum_{k=1}^{r} b_k \psi_k(\mathbf{x})$ given in (4).

Proposition 1 *Given a Gaussian process $\zeta(\mathbf{x})$ over $\mathcal{X}$, with zero mean and covariance function $g^{-1}K(\cdot,\cdot)$, where $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a Mercer reproducing kernel, there exists a vector-valued function $\psi(\mathbf{x}) = (\psi_1(\mathbf{x}), \ldots, \psi_r(\mathbf{x}))'$ from $\mathcal{X}$ to $\mathbb{R}^r$ (r is possibly infinite) such that $K(\mathbf{x}_i, \mathbf{x}_j) = \psi(\mathbf{x}_i)'\psi(\mathbf{x}_j)$ for $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$ and*

$$\zeta(\mathbf{x}) = \sum_{k=1}^{r} b_k \psi_k(\mathbf{x}) \;\; \textit{with} \;\; b_k \overset{i.i.d}{\sim} N(0,\, g^{-1}). \tag{12}$$

Conversely, given a function $\zeta : \mathcal{X} \to \mathbb{R}$ in (12), then ζ is a Gaussian process with zero mean and covariance function $g^{-1}K(\mathbf{x}_i, \mathbf{x}_j)$ where $K(\mathbf{x}_i, \mathbf{x}_j) = \psi(\mathbf{x}_i)'\psi(\mathbf{x}_j)$.

If the feature expansion in (12) is regarded as a stochastic process, it is known as the Karhunen-Loève expansion. Proposition 1 provides a direct connection between GKMs and GP classifiers (GPCs) (Neal, 1999; Girolami and Rogers, 2006), and between GKMs and model-based geostatistics (Diggle et al., 1998). We see that $\mathbf{b} = (b_1, \ldots, b_r)'$ behaves as a regression vector in GKMs, whereas it plays the role of a latent vector in GPCs. Consequently, the feature function $\psi(\mathbf{x})$ defines the fixed-effect part of GKMs and the random-effect part of GPCs. In parallel with the fact that GKMs are GLMs in the feature space induced by the reproducing kernel K, we see that GPCs are generalized linear mixed models (Harville, 1977) in the feature space.

As discussed in Section 2, the Karhunen-Loève expansion can also be approximated by a finite-dimensional expansion over the training data set; that is,

$$\zeta(\mathbf{x}) = \sum_{i=1}^{n} \beta_i K(\mathbf{x}, \mathbf{x}_i) \text{ with } \beta = (\beta_1, \ldots, \beta_n)' \sim N_n(\mathbf{0},\, g^{-1}\mathbf{K}^{-1}).$$

Let $\zeta = (\zeta(\mathbf{x}_1), \ldots, \zeta(\mathbf{x}_n))'$ be the vector of n realizations of ζ over the training data. We then have $\zeta = \mathbf{K}\beta \sim N_n(\mathbf{0}, g^{-1}\mathbf{K})$. In our sparse treatment, some of the β_i are set to zero and the subvector β_γ of the nonzero elements is modeled as $N_{n_\gamma}(\mathbf{0}, g^{-1}\mathbf{K}_{\gamma\gamma}^{-1})$. In this case, $\zeta = \mathbf{K}_\gamma\beta_\gamma$ follows a singular normal distribution, that is, $\zeta \sim N_n(\mathbf{0}, g^{-1}\mathbf{K}_\gamma\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma')$. This sparse technique is called the "subset of regressors" (Rasmussen and Williams, 2006). In the following sections we investigate this sparsity-inducing approach to GP-based classification as an alternative to the FBGKM introduced in Section 3.

4.2 The MCMC Algorithm

By analogy with on the hierarchical model for our FBGKM in Section 3.1, we model the auxiliary variable s_i as

$$s_i = s(\mathbf{x}_i) = u + \zeta(\mathbf{x}_i) + \varepsilon_i \text{ with } \varepsilon_i \sim N(0,\, 1)$$

and keep other settings unchanged. Here $\zeta(\mathbf{x})$ is the Gaussian process with $\mathrm{E}(\zeta(\mathbf{x})) = 0$ and $\mathrm{Cov}(\zeta(\mathbf{x}_i), \zeta(\mathbf{x}_j)) = g^{-1}K(\mathbf{x}_i, \mathbf{x}_j)$. Applying $\zeta(\mathbf{x})$ to the training data, we have

$$\mathbf{s} = u\mathbf{1}_n + \zeta + \varepsilon \text{ with } \varepsilon \sim N_n(\mathbf{0},\, \mathbf{I}_n).$$

The inverses of $n \times n$ matrices are also required during Bayesian inference and prediction for GPCs. In order to reduce the computational costs, we use $\mathbf{K}_\gamma\beta_\gamma$ with $\beta_\gamma \sim N_{n_\gamma}(\mathbf{0}, g^{-1}\mathbf{K}_{\gamma\gamma}^{-1})$ to approximate ζ as in Section 4.1. This yields a *sparse GPC* (SGPC) model. The MCMC algorithm for SGPC is immediately obtained from that for FBGKM by simply removing the update of β_γ in Section 3.2, because β_γ is now the latent vector and it is not used for prediction. In particular, GPCs use the expectation of y_* w.r.t. $p(y_*|\mathbf{x}_*, \mathbf{y})$ as the predictor. We thus need to insert a step, which is to sample $s_* = s(\mathbf{x}_*)$ from $p(s_*|\mathbf{s}, \gamma, u, \mathbf{y})$, into the MCMC algorithm for prediction. This step is only necessary at every Mth sweep after the burn-in of the MCMC algorithm (see Diggle et al., 1998).

Now the marginal distribution of $\mathbf{s}$ is $[\mathbf{s}|u, \gamma] \sim N_n(u\mathbf{1}_n,\, \mathbf{M}_\gamma)$, where $\mathbf{M}_\gamma = \mathbf{I}_n + g^{-1}\mathbf{K}_\gamma\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma'$. Since s_* is conditionally independent of $\mathbf{y}$, given $\mathbf{s}$, we have

$$p(s_*|\mathbf{s}, \gamma, u) = N\big(u + g^{-1}\mathbf{k}_\gamma(\mathbf{x}_*)'\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma'\mathbf{M}_\gamma^{-1}(\mathbf{s} - u\mathbf{1}_n),\, v\big),$$

where $v = g^{-1}\mathbf{k}_\gamma(\mathbf{x}_*)'\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{k}_\gamma(\mathbf{x}_*) + 1 - g^{-2}\mathbf{k}_\gamma(\mathbf{x}_*)'\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{K}_\gamma'\mathbf{M}_\gamma^{-1}\mathbf{K}_\gamma\mathbf{K}_{\gamma\gamma}^{-1}\mathbf{k}_\gamma(\mathbf{x}_*)$ and $\mathbf{k}_\gamma(\mathbf{x}_*)$ is the subvector of $(K(\mathbf{x}_*, \mathbf{x}_1), \ldots, K(\mathbf{x}_*, \mathbf{x}_n))'$ corresponding to $\gamma_i = 1$. Since we have $\mathsf{E}(y_*|s_*) = \tau(s_*)$ and the used probit link τ is a monotonically increasing function on $(-\infty, \infty)$, we allocate $y_* = 1$ if $s_* > 0$ and $y_* = 0$ otherwise.

Let $I = \{\mathbf{x}_i : \gamma_i = 1, i = 1, \ldots, n\}$ be the set of active vectors. If the kernel function is stationary, then v is near zero and the posterior predictive mean of s_* reverts to u when $\mathbf{x}_*$ is far from points in the set I. Thus the sparse technique will give poor predictions, especially underestimates of the predictive variance. However, this problem is mitigated in our SGPC methodology since it uses Bayesian model averaging for prediction. That is, the prediction is based on the average over T active sets. In the following experiments the average is taken on $1,000$ active sets.

It is again worth noting that the reversible jump MCMC algorithm devised in this paper deals with parameter estimation and posterior prediction jointly in a single paradigm. Moreover, the reversible jump methodology is a sequential approach to model selection and posterior prediction. The main computational burden of the MCMC algorithm comes from the sampling procedure for parameter estimation, and the MCMC algorithm does not require extra computational complexity for the selection of active vectors and the prediction of responses.

The MCMC algorithms used in Neal (1999) and Diggle et al. (1998) is less efficient than ours because they do not use data augmentation or exploit sparsity. Girolami and Rogers (2006) proposed a Bayesian multinomial probit regression model, using variational methods for inference. For additional discussion of sparse approaches to GPs, interested readers should refer to Quiñonero-Candela and Rasmussen (2005), Rasmussen and Williams (2006) and Snelson (2007) and references therein.

5. Experimental Evaluations

In this section we conduct several experiments to evaluate the performance of our proposed Bayesian classification methods: FBGKM and SGPC. We compare the methods with various closely related Bayesian and non-Bayesian classification methods, including the Bayesian SVM (BSVM) (Mallick et al., 2005), the complete SVM (CSVM) (Mallick et al., 2005), sparse Gaussian processes (SGP+FIC) (Snelson and Ghahramani, 2006), and the conventional IVM and SVM.

We also implement our Bayesian GKM without Step (c) of the MCMC algorithm in Section 3.2. That is, we implement an MCMC algorithm that consists of Steps (a)-(b) by fixing $n_\gamma = n$. We denote the resulting model by BGKM to distinguish it from FBGKM. We could also implement a full (non-sparse) GPC, but since such a full GPC would have almost the same computational complexity as the GBKM, we do not implement the non-sparse GPC. All experiments have been implemented in Matlab on a Pentium 4 with a 2.80GHz CPU and 2.00GB of RAM.

5.1 Setup

We perform the experiments on several benchmark data sets: `BCI`, `g241d`, `Digit1`, $\texttt{COIL}_2$, `USPS digits` {`(0 vs. 1)`, `(0 vs. 9)`}, `Letters` {`(A vs. B)`, `(A vs. C)`}, `NewsGroups corpora`, $\texttt{Adult}_1$, $\texttt{Adult}_2$, `Mushrooms`, `Splice`, `Astroparticle`, `Ringnorm`, `Thyroid`, `Twonorm`, and `Waveform`. We first present a brief review of these data sets.

The `BCI` data set contains data obtained from project in brain-computer interfaces in which a single subject performs 400 trials in which he imagines movements with either the left or right hand. The `g241d` data set is an artificial data set which is generated by two unit-variance isotropic Gaussians with potentially misleading cluster structure. The `Digit1` data set is generated by applying a sequence of transformations to digit images, leading to a low-dimensional manifold geometrical structure embedded into a high-dimensional space. The $\texttt{COIL}_2$ data set is derived from the Columbia object image library (COIL-100) under a sequence of transformations, for example, rescaling, adding noise, and masking dimensions. Note that the `BCI`, `g241d`, `Digit1` and $\texttt{COIL}_2$ data sets are available at `http://www.kyb.tuebingen.mpg.de/ssl-book/`.

The USPS database is a handwritten digits data set which contains the digits from 0 to 9 automatically scanned from envelopes by the U.S. Postal Service. In our experiments, two digit pairs {`(0 vs. 1)`, `(0 vs. 9)`} data sets are randomly constituted from the USPS database, and the dimensionality of each digit image and the number of digits in each digit class of each data set are 256 and 1000, respectively. The `Letters` data set consists of images of the 26 capital letters from

"A" to "Z," and two letter pairs {(A vs. B), (A vs. C)} are randomly constituted from "A," "B" and "C" with 789, 766, and 736 cases, respectively.

The 20 NewsGroups data set is organized into 20 different newsgroups, each corresponding to a different topic, and we randomly select the alt.atheism and comp.graphics topics for the binary classification problem. The total vocabulary size is 1390. Based on the information gain, 893 features are employed.

The Adult data set is originally extracted from the 1994 Census database with 14 features, of which six features are continuous and eight are categorical. Further, the Adult data set is processed with dimensionality of 123, that is, each continuous feature is discretized into a binary feature and each categorical feature with q categories is converted to q binary features. Here, the $\texttt{Adult}_1$ and $\texttt{Adult}_2$ data sets are constituted according to different training and test sizes.

The Mushrooms data set is originally drawn from the Audubon Society Field Guide to North American Mushrooms with 22 features. Similar to the Adult data set, the Mushrooms data set is processed into the binary feature representations, leading to 123 dimensions for each instance. The Splice data set is based on the biological process whereby intronic DNA is removed during protein translation. The Astrop (Astroparticle) data set is obtained from Jan Conrad of Uppsala University, Sweden. The Adult, Mushrooms, Splice, and Astrop data sets are available at http://www.csie.ntu.edu.tw/~cjlin.

The Ringnorm data set is artificially generated from two multivariate Gaussian distributions for the binary classification problem. That is, the instances within each class are obtained from a 20-variate Gaussian distribution. The Thyroid is collected from several databases of thyroid disease records. We use this data set to conduct a binary classification experiment in which the class euthyroidism is considered as the normal class and the classes hypothyroidism and hyperthyroidism are considered as an abnormal class.

The Twonorm data set is also an artificial 20-dimensional two-class classification example, which consists of 7400 instances. The Waveform data set is generated from a combination of 2 of 3 "base" waves in a 21-dimensional space. The Ringnorm, Thyroid, Twonorm, and Waveform data sets are widely used for the classification benchmarking, and they are available at http://ida.first.gmd.de/~raetsch/data/benchmarks.htm.

Table 1 gives a summary of these data sets. In our experiments, each data set is randomly partitioned into two disjoint subsets as the training and test. Twenty random partitions are generated for each data set. Based on these partitions, several evaluation criteria, including the average classification error rate, standard deviation and average computational time, are reported.

All of the methods that we implement are based on a Gaussian RBF kernel with a single width parameter; that is, $K(\mathbf{x}_i,\mathbf{x}_j) = \exp\left(-\|\mathbf{x}_i-\mathbf{x}_j\|_2^2/\theta^2\right)$. In Section 5.3 we present experiments in which this hyperparameter is estimated from data based on the ideas discussed in Section 3.2. In the remaining sections, however, we use a simpler procedure in which the value of θ is set to the mean Euclidean distance between training data points. We found this setting to be effective empirically in our applications. The gain in computational complexity is significant, particularly for the full GP methods, BSVM and CSVM, whose calculations involve two full kernel matrices. In particular, for each new value of θ, it is necessary to recalculate the kernel matrix $\mathbf{K}$ for each sweep of the MCMC algorithms.

In addition, we set the hyperparameters in both FBGKM and SGPC as follows: $a_\eta = 1$, $b_\eta = 0.1$, $a_g = 4$ and $b_g = 0.1$. For all of the Bayesian classification methods, we run each MCMC algorithm for $10,000$ sweeps, discard the first $5,000$ as the burn-in, and retain every 5th realization

Data Set	n	m	p	$k_{\max}$
BCI	300	100	117	100
g241d	300	1200	241	200
Digit1	300	1200	241	200
COIL_2	300	1200	241	200
USPS (0 vs.1)	500	1500	256	200
USPS (0 vs.9)	500	1500	256	200
Letters (A vs.B)	300	1255	16	100
Letters (A vs.C)	300	1225	16	100
NewsGroups	500	1485	893	200
Splice	2000	1175	60	200
Astrop(article)	4000	3089	4	200
Mushrooms	4000	4124	112	200
Adult_1	6000	10000	123	200
Adult_2	20000	12500	123	200
Ringnorm	400	7000	20	400
Thyroid	140	75	5	140
Twonorm	400	7000	20	400
Waveform	400	4600	21	400

Table 1: Summary of the Benchmark Data Sets: n—the size of the training data set; m—the size of the test data set; p—the dimension of the input vector; $k_{\max}$—the maximum number of active vectors.

of parameters after the burn-in for inference and prediction. These settings are empirically validated to be sufficient for these methods to achieve convergence. Recall that the test is implemented after the burn-in of the MCMC sampling. This implies that the Bayesian model averaging component of our Bayesian methods uses $1,000$ ($T = (10,000-5,000)/5$) active sets for test.

5.2 Evaluation 1

In the first evaluation, we compare BGKM, FBGKM and SGPC with BSVM and CSVM, because they are the two existing Bayesian kernel methods most closely related to our Bayesian classification methods.

We conduct this evaluation on the first nine data sets in Table 1, randomly partitioning the data into disjoint training and test data sets according to the corresponding settings of n and m. All the inputs are normalized to have zero mean and unit variance. Tables 2 and 3 report the performance of the five Bayesian methods on the nine different data sets in terms of the average classification error rate (%), the standard deviation and the corresponding average computational time (s).

From Tables 2 and 3, we can see that our FBGKM, SGPC and BGKM methods based on the Silverman g-prior achieve slightly lower classification error rates than the BSVM and CSVM methods on the whole. Moreover, our methods have roughly similar classification error rates on the nine data sets. In addition, FBGKM and SGPC are more efficient computationally than BGKM the other methods; this is due to their exploitation of sparsity.

Data Set	BSVM		CSVM		BGKM		SGPC		FBGKM	
	err	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)	*err*	($\pm std$)
BCI	28.15	($\pm$2.15)	29.40	($\pm$2.58)	29.35	($\pm$2.82)	27.10	($\pm$1.85)	29.83	($\pm$2.36)
g241d	17.15	($\pm$1.68)	17.63	($\pm$1.15)	16.37	($\pm$1.11)	16.55	($\pm$1.22)	16.30	($\pm$0.89)
Digit1	4.86	($\pm$0.74)	4.88	($\pm$0.75)	4.87	($\pm$0.65)	5.51	($\pm$0.66)	4.85	($\pm$0.67)
COIL_2	9.71	($\pm$0.81)	9.86	($\pm$0.71)	9.16	($\pm$0.99)	9.83	($\pm$0.97)	9.797	($\pm$0.32)
USPS(0 vs. 1)	0.40	($\pm$0.30)	0.35	($\pm$0.11)	0.28	($\pm$0.05)	0.31	($\pm$0.14)	0.28	($\pm$0.06)
USPS(0 vs. 9)	1.36	($\pm$0.36)	1.40	($\pm$0.29)	1.36	($\pm$0.28)	1.21	($\pm$0.19)	1.37	($\pm$0.24)
Letters(A vs. B)	0.92	($\pm$0.59)	0.95	($\pm$0.45)	0.75	($\pm$0.24)	0.53	($\pm$0.19)	0.77	($\pm$0.24)
Letters(A vs. C)	0.83	($\pm$0.15)	0.93	($\pm$0.27)	0.87	($\pm$0.15)	0.65	($\pm$0.20)	0.84	($\pm$0.15)
NewsGroups	5.62	($\pm$0.80)	5.08	($\pm$0.33)	4.92	($\pm$0.28)	4.66	($\pm$0.38)	4.83	($\pm$0.25)

Table 2: Experimental results for the five methods on different data sets: $err-$ the test error rates (%); $std-$ the corresponding standard deviation.

Data Set	BSVM	CSVM	BGKM	SGPC	FBGKM
BCI	2.615×10^3	2.596×10^3	1.063×10^3	0.756×10^3	0.688×10^3
g241d	4.339×10^3	4.365×10^3	1.819×10^3	1.227×10^3	1.451×10^3
Digit1	5.248×10^3	5.210×10^3	2.459×10^3	1.738×10^3	2.011×10^3
COIL_2	4.988×10^3	4.996×10^3	2.454×10^3	1.357×10^3	1.502×10^3
USPS(0 vs. 1)	2.133×10^4	2.047×10^4	6.013×10^3	2.464×10^3	2.700×10^3
USPS(0 vs. 9)	2.239×10^4	2.230×10^4	6.479×10^3	2.868×10^3	2.974×10^3
Letters(A vs. B)	2.009×10^3	2.007×10^3	0.914×10^3	0.568×10^3	0.593×10^3
Letters(A vs. C)	2.026×10^3	2.042×10^3	0.896×10^3	0.604×10^3	0.596×10^3
NewsGroups	2.286×10^4	2.291×10^4	6.270×10^3	2.675×10^3	2.910×10^3

Table 3: The computational times (s) for the five methods on different data sets.

In the following experiments, we attempt to analyze the performance of the methods with respect to different values of the training size n and the maximum number k_{max} of active vectors. For the sake of simplicity, we only report results on the NewsGroups data set.

Tables 4 and 5 show the experimental results when changing the training size n and fixing the maximum number of active vectors to $k_{max} = 200$. As can be seen, all the five methods obtain a lower classification error rate and have greater computational costs as the training size n increases. Furthermore, FBGKM, SGPC and BGKM slightly outperform BSVM and CSVM in both classification error rate and computational cost. The FBGKM and SGPC methods are relatively more efficient for the data sets of large training size n.

Table 6 shows the experimental results for our FBGKM and SGPC methods with respect to different values of the maximum number k_{max} of active vectors and for a fixed training size of $n = 800$. The performance of these two methods is roughly similar for each setting k_{max}; that is, they are insensitive to k_{max}. However, their computational costs tend to slightly increase as the maximum number k_{max} of active vectors increases.

Additionally, in order to study the MCMC mixing performance of our FBGKM and SGPC methods we report the numbers of active vectors over different data sets. In particular, Figure 2

Training size n	BSVM	CSVM	BGKM	SGPC	FBGKM
	err $(\pm std)$	err $(\pm std)$	err $(\pm std)$	err $(\pm std)$	err $(\pm std)$
n=300	5.99 ($\pm$1.44)	5.84 ($\pm$0.80)	5.37 ($\pm$0.52)	5.34 ($\pm$0.40)	5.08 ($\pm$0.49)
n=400	5.65 ($\pm$0.98)	5.83 ($\pm$0.93)	5.10 ($\pm$0.35)	5.03 ($\pm$0.55)	5.05 ($\pm$0.39)
n=500	5.62 ($\pm$0.80)	5.08 ($\pm$0.33)	4.92 ($\pm$0.28)	4.66 ($\pm$0.38)	4.83 ($\pm$0.25)
n=600	5.77 ($\pm$0.61)	5.13 ($\pm$0.20)	4.92 ($\pm$0.43)	4.35 ($\pm$0.47)	4.74 ($\pm$0.28)
n=700	5.63 ($\pm$0.82)	4.82 ($\pm$0.21)	4.44 ($\pm$0.36)	4.12 ($\pm$0.22)	4.61 ($\pm$0.52)
n=800	5.14 ($\pm$0.59)	5.10 ($\pm$0.16)	4.49 ($\pm$0.47)	4.13 ($\pm$0.51)	4.56 ($\pm$0.34)

Table 4: Experimental results for the five methods corresponding to different training sizes n on the NewsGroups data set with $k_{max} = 200$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation.

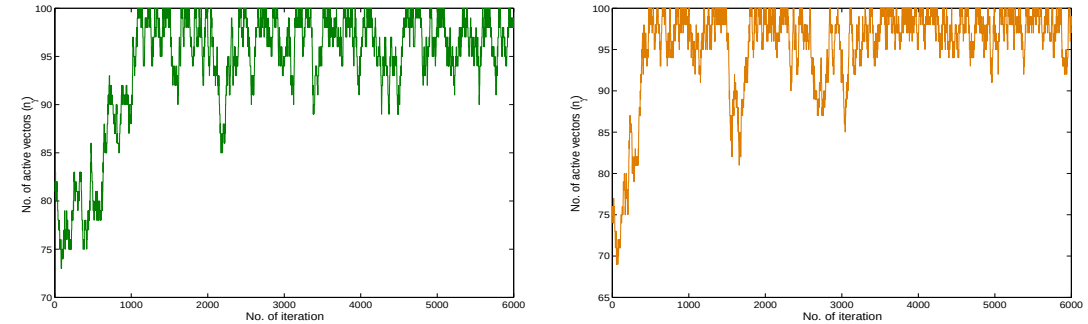
Training size n	BSVM	CSVM	BGKM	SGPC	FBGKM
n=300	5.949×10^3	5.830×10^3	2.467×10^3	1.862×10^3	2.085×10^3
n=400	1.173×10^4	1.171×10^4	4.674×10^3	2.555×10^3	2.804×10^3
n=500	2.286×10^4	2.291×10^4	6.270×10^3	2.675×10^3	2.910×10^3
n=600	3.458×10^4	3.461×10^4	8.340×10^3	2.748×10^3	2.973×10^3
n=700	5.195×10^4	5.186×10^4	1.207×10^4	3.279×10^3	3.610×10^3
n=800	7.754×10^4	7.757×10^4	1.673×10^4	3.885×10^3	4.327×10^3

Table 5: The computational times (s) for the five methods corresponding to different training sizes n on the NewsGroups data set with $k_{max} = 200$.

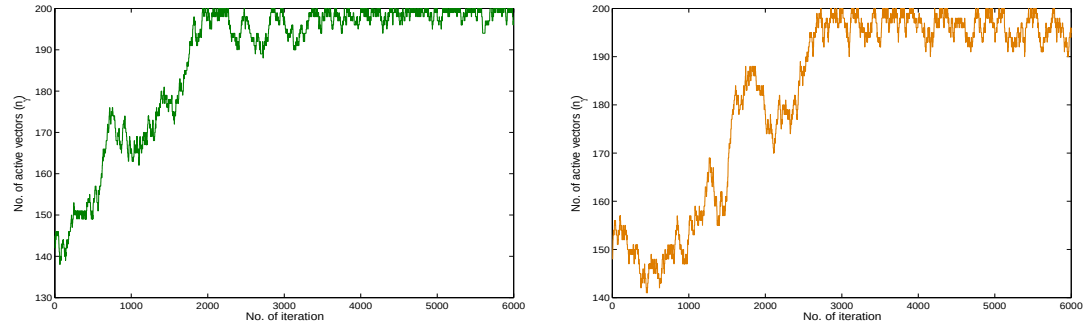
k_{max} of active vectors	FBGKM		SGPC	
	err $(\pm std)$	$time$	err $(\pm std)$	$time$
$k_{max} = 300$	4.55 ($\pm$0.46)	6.522×10^3	4.27 ($\pm$0.47)	5.592×10^3
$k_{max} = 400$	4.62 ($\pm$0.45)	7.189×10^3	4.80 ($\pm$0.49)	6.808×10^3
$k_{max} = 500$	4.64 ($\pm$0.37)	8.536×10^3	4.75 ($\pm$0.38)	7.704×10^3
$k_{max} = 600$	4.75 ($\pm$0.48)	1.033×10^4	4.57 ($\pm$0.43)	9.469×10^3
$k_{max} = 700$	4.72 ($\pm$0.28)	1.170×10^4	4.74 ($\pm$0.31)	1.057×10^4

Table 6: Experimental results for FBGKM and SGPC corresponding to different maximum numbers k_{max} of active vectors on the NewsGroups data set with $n = 800$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation; $time-$ the corresponding computational time (s).

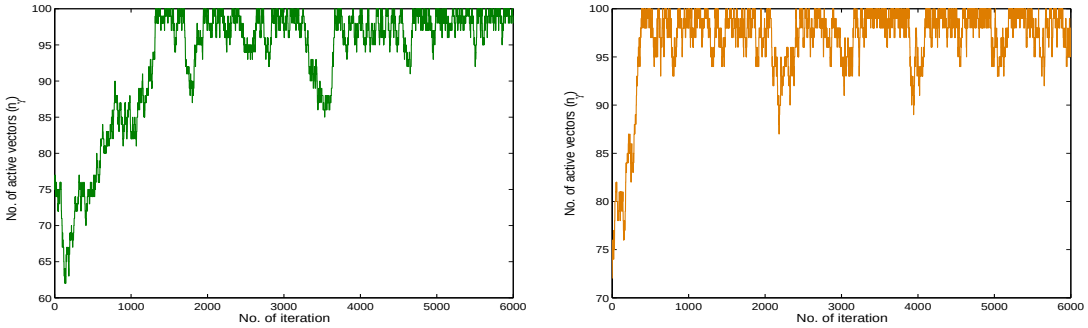
depicts the output of the numbers n_γ of active vectors corresponding to the first 6000 sweeps in the MCMC inference procedure on BCI, Digit1, Letters {(A vs.B)} and NewsGroups. The results in Figure 2 clearly show that the FBGKM and SGPC methods mix rapidly in these experiments, yielding reliable estimates after the first 3000 sweeps.



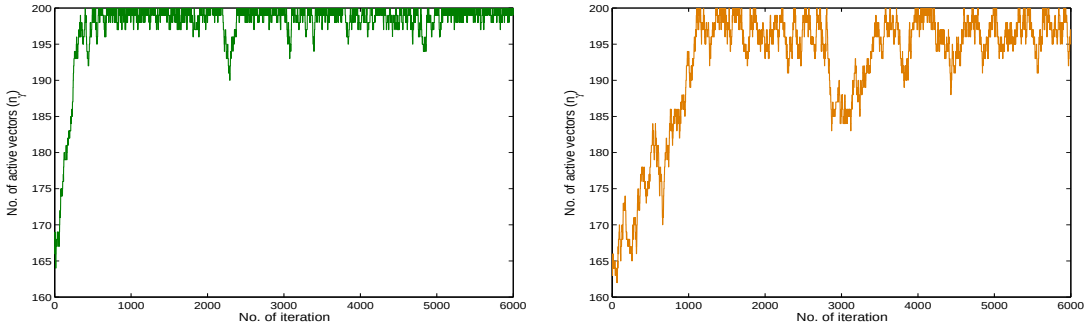
(a) FBGKM, $k_{max} = 100$ (b) SGPC, $k_{max} = 100$



(c) FBGKM, $k_{max} = 200$ (d) SGPC, $k_{max} = 200$



(e) FBGKM, $k_{max} = 100$ (f) SGPC, $k_{max} = 100$



(g) FBGKM, $k_{max} = 200$ (h) SGPC, $k_{max} = 200$

Figure 2: MCMC Output for the numbers n_γ of active vectors of our FBGKM and SGPC methods on the four data sets: (a/b) `BCI`; (c/d) `Digit1`; (e/f) `Letters (A vs. B)`; (g/h) `NewsGroups`.

In probit-type models, since the posterior distribution of each s_i is truncated normal, we are able to update the s_i using the Gibbs sampler. Recall that we employ an efficient auxiliary variable approach proposed by Holmes and Held (2006) for the implementation of the Gibbs sampler (see Appendix A). For the other models, however, a MH sampler is required to update the s_i. This makes the corresponding MCMC algorithms take longer to mix. Thus, our models, which are based on the probit link, can be expected to be more efficient computationally than the BSVM and CSVM. However, to standardize the experimental comparison, we use the same setup for MCMC sweeps and burn-in for all algorithms.

Table 7 describes distributions of active vectors to appear after the burn-in (in the last $5,000$ sweeps). As we can see, the number n_γ of active vectors jumps between a small range for different data sets, due to the rapid mixing. The maximum frequency of active vectors corresponding to the number n_γ of active vectors is also given in Table 7.

Data Set	FBGKM			SGPC		
	Max	Min	Most	Max	Min	Most
BCI	100	85	99 (918)	100	87	99 (1145)
g241d	200	184	199 (1263)	200	153	196 (339)
Digit1	200	187	199 (1163)	200	184	188 (515)
COIL_2	151	138	146 (1493)	149	139	147 (2870)
USPS(0 vs. 1)	170	151	165 (1080)	151	132	140 (1288)
USPS(0 vs. 9)	200	172	195 (1258)	200	177	199 (1095)
Letters(A vs. B)	100	80	99 (796)	100	89	99 (1422)
Letters(A vs. C)	100	80	97 (551)	100	84	99 (1006)
NewsGroups	200	186	199 (918)	200	182	199 (720)

Table 7: Distributions of active vectors after the burn-in under FBGKM and SGPC. Max—the maximum number of active vectors to appear; Min—the minimum number of active vectors to appear; Most—the number of active vectors with the maximum frequency and the corresponding frequency shown in brackets.

5.3 Evaluation 2

We further evaluate the performance of our sparse Bayesian kernel methods under kernel parameter learning, and compare the FBGKM and SGPC with SGP+FIC and full GP (FGP) (Rasmussen and Williams, 2006). In particular, we use the Gaussian RBF kernel with multiple parameters, that is, $K(\mathbf{x}_i,\mathbf{x}_j) = \exp\left(-\sum_{l=1}^{p}(x_{il}-x_{jl})^2/\theta_l^2\right)$, and estimate those parameters $\theta = (\theta_1,\ldots,\theta_p)$ in all compared Bayesian kernel methods. In order to distinguish from the Bayesian methods with the fixed kernel parameters, we label the Bayesian methods with the learned parameters via "$\star$+KL." We conduct experimental analysis on the Adult, Mushrooms, Splice, and Astroparticle data sets.

Since for FGP+KL learning the kernel parameters results in a huge computational cost, we set the sizes of the training and test data as 1000 (i.e., $n = m = 1000$) in each data set. In this setting, there is no distinction between Adult_1 and Adult_2, so we just use Adult to denote the corresponding data set. Also, since it is infeasible to use MCMC inference for FGP+KL, we employ the

expectation propagation (EP) algorithm (Minka, 2001) for FGP+KL. However, to provide an apples-to-apples comparison with our sparse Bayesian kernel methods, we employ MCMC inference for SGP+FIC+KL. For the sparse methods compared here, we fix the size of active set to 100, that is, $k_{max} = 100$. Our implementations for SGP+FIC and FGP are based on the Matlab codes from `http://www.lce.hut.fi/research/mm/gpstuff/` and `http://www.gaussianprocess.org/gpml/`, respectively.

Tables 8 and 9 and Figure 3 report the performance of the SGP+FIC+KL, FGP+KL, FBGKM+KL, and SGPC+KL methods on the four data sets in terms of the average classification error rate (%), the standard deviation and the corresponding average computational time (s). Figure 3 depicts the logarithm scale of the corresponding average computational time (s) on the different data sets. It is clear that FBGKM+KL and SGPC+KL outperform other methods on the whole. Additionally, the computational times of all compared methods tend to increase when the number p of the kernel parameters increases. We note that the computational times of SGP+FIC+KL and FGP+KL would become huge if we directly applied them to the large data sets listed in Table 1—$\texttt{Adult}_1$, $\texttt{Adult}_2$, `Mushrooms`, `Splice`, and `Astroparticle`.

Data Set	SGP+FIC+KL		FGP+KL		FBGKM+KL		SGPC+KL	
	err	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$
`Splice`	18.05	($\pm$0.77)	9.19	($\pm$1.25)	7.49	($\pm$0.14)	11.54	($\pm$0.51)
`Astrop`	4.10	($\pm$0.36)	4.57	($\pm$0.41)	3.45	($\pm$0.31)	3.52	($\pm$0.31)
`Mushrooms`	1.70	($\pm$0.27)	0.20	($\pm$0.20)	0.24	($\pm$0.18)	0.45	($\pm$0.30)
`Adult`	18.50	($\pm$0.56)	17.85	($\pm$0.35)	15.94	($\pm$0.45)	15.56	($\pm$0.43)

Table 8: Experimental results for the four Bayesian kernel methods on the four data sets with learned kernel parameters θ, $k_{max} = 100$, $n = 1000$, and $m = 1000$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation.

Data Set	SGP+FIC+KL	FGP+KL	FBGKM+KL	SGPC+KL
`Splice`	2.542×10^5	1.228×10^5	1.132×10^4	1.121×10^4
`Astrop`	4.081×10^4	2.531×10^4	7.431×10^3	7.103×10^3
`Mushrooms`	4.639×10^5	1.583×10^5	1.551×10^4	1.534×10^4
`Adult`	4.713×10^5	1.605×10^5	1.606×10^4	1.557×10^4

Table 9: The computational times (s) of the four Bayesian kernel methods on the four data sets with learned kernel parameters θ, $k_{max} = 100$, $n = 1000$, and $m = 1000$.

In order to further evaluate the performance of our sparse Bayesian kernel methods on some larger data sets, we also conduct comparative experiments of FBGKM, SGPC, and SGP+FIC (Snelson and Ghahramani, 2006) on the $\texttt{Adult}_1$, $\texttt{Adult}_2$, `Mushrooms`, `Splice`, and `Astroparticle` data sets. Here, we consider both MCMC and EP inference methods for SGP+FIC to provide a fuller comparison, referring to them as SGP+FIC+MCMC and SGP+FIC+EP, respectively. For these sparse methods, we fix the size of active set k_{max} to 200.

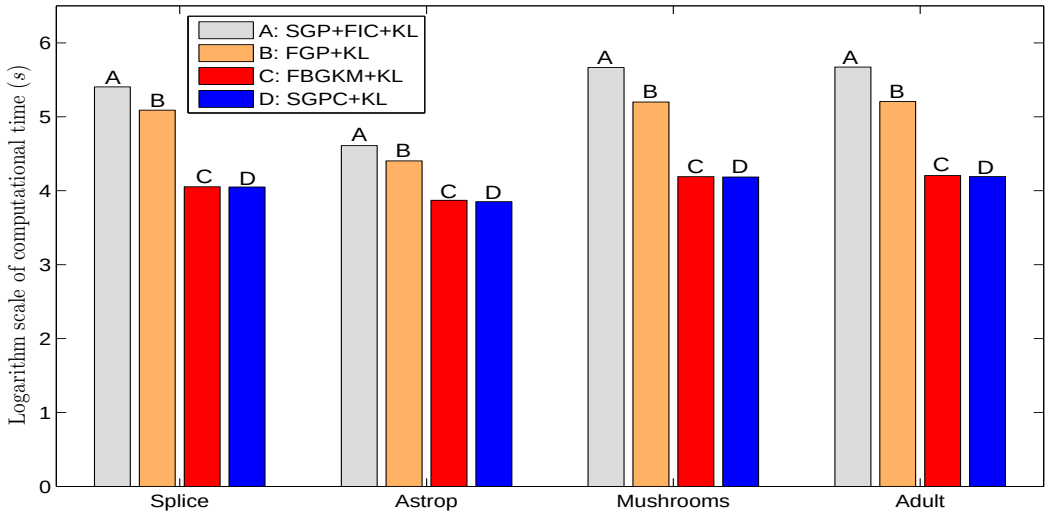


Figure 3: The computational times (s) for the four Bayesian kernel methods on the four data sets with learned kernel parameter θ, $k_{max} = 100$, $n = 1000$, and $m = 1000$.

Table 10 reports the classification performance of the SGP+FIC+MCMC, SGP+FIC+EP, FBGKM and SGPC methods on the five data sets. It should be pointed out here that we do not report the corresponding results of SGP+FIC+MCMC on the $\texttt{Adult}_2$ data set due to the huge computational times of performing it on this data set. From Table 10, we can see that our FBGKM and SGPC methods outperform other methods on the whole. Furthermore, it is still difficult for SGP+FIC+MCMC and SGP+FIC+EP to calculate the optimal solution for sparse approximation of full Gaussian process, due to the sensitivity of the performance to the initial active set.

Data Set	SGP+FIC+EP		SGP+FIC+MCMC		FBGKM		SGPC		SVM	
	err	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$
Splice	14.38	(±1.10)	16.32	(±0.15)	12.53	(±0.57)	12.07	(±0.45)	13.01	(±0.69)
Astrop	5.20	(±0.26)	3.38	(±0.11)	3.59	(±0.18)	3.34	(±0.16)	3.37	(±0.14)
Mushrooms	1.55	(±0.21)	1.38	(±0.13)	0.19	(±0.08)	0.21	(±0.06)	0.55	(±0.37)
$\texttt{Adult}_1$	15.89	(±0.38)	15.79	(±0.26)	15.24	(±0.21)	15.59	(±0.16)	16.64	(±0.33)
$\texttt{Adult}_2$	15.49	(±0.21)	–	–	15.01	(±0.17)	15.26	(±0.19)	16.27	(±0.28)

Table 10: Experimental results for the five methods on the Splice, Astroparticle, Mushrooms, $\texttt{Adult}_1$, and $\texttt{Adult}_2$ data sets with $k_{max} = 200$: $err-$ the test error rates (%); $std-$ the corresponding standard deviation.

Table 11 and Figure 4 report the average computational times of the compared sparse Bayesian kernel methods on the five data sets, with Figure 4 depicting the computational times on a logarithm scale. Table 11 and Figure 4 show that the SGP+FIC+MCMC has larger computational times than

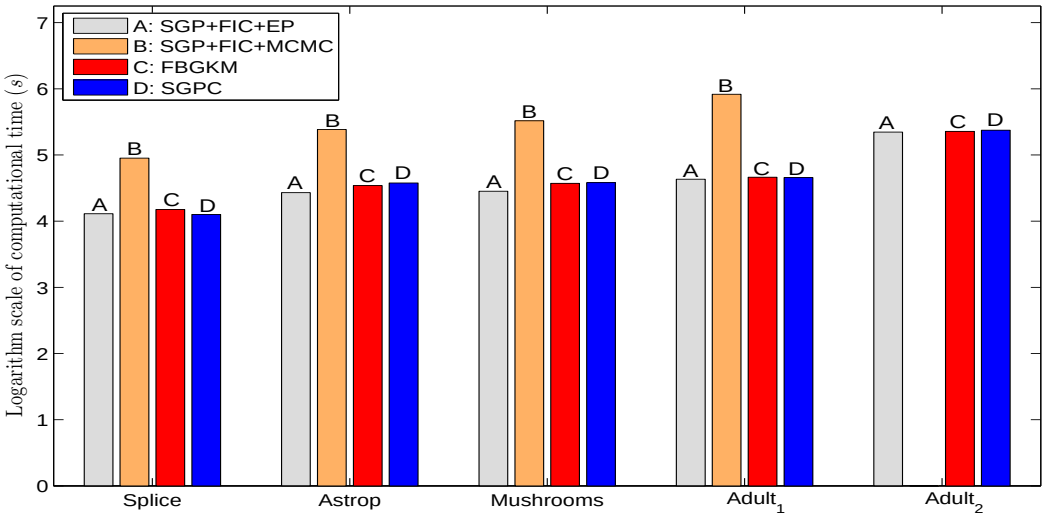


Figure 4: The computational times (s) of the four sparse Bayesian kernel methods on the `Splice`, `Astroparticle`, `Mushrooms`, $\texttt{Adult}_1$, and $\texttt{Adult}_2$ data sets with $k_{max} = 200$.

other methods on the five different data sets, and that the computational times of SGP+FIC+EP, FBGKM and SGPC are very close to each other.

Data Set	SGP+FIC+EP	SGP+FIC+MCMC	FBGKM	SGPC
`Splice`	1.293×10^4	8.952×10^4	1.497×10^4	1.262×10^4
`Astrop`	2.701×10^4	2.425×10^5	3.443×10^4	3.779×10^4
`Mushrooms`	2.830×10^4	3.290×10^5	3.726×10^4	3.814×10^4
$\texttt{Adult}_1$	4.298×10^4	8.283×10^5	4.620×10^4	4.548×10^4
$\texttt{Adult}_2$	2.221×10^5	–	2.276×10^5	2.369×10^5

Table 11: The computational times (s) of the four sparse Bayesian kernel methods on the `Splice`, `Astroparticle`, `Mushrooms`, $\texttt{Adult}_1$, and $\texttt{Adult}_2$ data sets with $k_{max} = 200$.

In addition, we conduct a quantitative assessment of convergence of the MCMC algorithms for the three sparse Bayesian kernel methods. We employ a method proposed by Brooks (1998). The method uses a cusum criterion with "hairiness" definition to monitor convergence. The length of chain for convergence is determined, once the sequence on the "hairiness" definition statistically lies within the 90% confidence intervals under the binomial distribution. Table 12 reports the convergence assessment results on the five data sets. From Table 12, we can see that all MCMC algorithms in the three sparse Bayesian kernel methods can achieve convergence on the five data sets after the first 5000 sweeps. Moreover, the convergence time is similar for each method, while the corresponding computational times of SGP+FIC+MCMC are obviously largest on the five data sets.

Data Set	SGP+FIC+MCMC		FBGKM		SGPC	
	time	*burn-in*	*time*	*burn-in*	*time*	*burn-in*
Splice	4.017×10^4	4566	1.019×10^3	906	2.195×10^3	1898
Astrop	9.265×10^4	3874	7.016×10^3	2125	1.123×10^4	3218
Mushrooms	9.857×10^4	3192	1.362×10^4	3924	9.677×10^3	2616
Adult_1	1.865×10^5	2451	8.605×10^3	1906	1.206×10^4	2873
Adult_2	–	–	5.217×10^4	2424	8.142×10^4	3605

Table 12: Monitoring convergence of MCMC algorithms for the three sparse Bayesian kernel methods on the Splice, Astroparticle, Mushrooms, Adult_1, and Adult_2 data sets with $k_{max} = 200$: *time*– the computational time (*s*) for convergence; *burn-in*– the length of chain for convergence.

5.4 Bayesian vs. Non-Bayesian

Since FBGKM and SGPC are Bayesian alternatives to IVM and SVM, it is useful to compare our FBGKM and SGPC with the conventional IVM and SVM. We compared these methods on the following data sets: Ringnorm, Thyroid, Twonorm and Waveform. These data sets were also used by Zhu and Hastie (2005) and a detailed presentation of results can be found in Rätsch et al. (2001). Each data set is randomly partitioned into two disjoint subsets as training and test data sets according to the training and test sizes *n* and *m* given in Table 1. In addition, the maximum number k_{max} of active vectors is set according to Table 1. The results in Table 13 are based on the average of twenty realizations and the results with the conventional IVM and SVM are cited from Zhu and Hastie (2005). We also conduct a comparison of FBGKM and SGPC with the conventional SVM on the Splice, Astroparticle, Mushrooms, Adult_1, and Adult_2 data sets. The classification results are given in Table 10. From Tables 10 and 13 we can see that the Bayesian approaches slightly outperform the non-Bayesian approaches.

Data Set	SVM		IVM		FBGKM		SGPC	
	err	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$	*err*	$(\pm std)$
Ringnorm	2.03	(± 0.19)	1.97	(± 0.29)	1.51	(± 0.10)	1.56	(± 0.12)
Thyroid	4.80	(± 2.98)	5.00	(± 3.02)	4.60	(± 2.65)	4.51	(± 2.32)
Twonorm	2.90	(± 0.25)	2.45	(± 0.15)	2.86	(± 0.21)	2.79	(± 0.23)
Waveform	9.98	(± 0.43)	10.13	(± 0.47)	9.80	(± 0.31)	9.73	(± 0.30)

Table 13: Experimental results for the four methods on the four data sets: *err*– the test error rates (%); *std*– the corresponding standard deviation.

6. Extensions

In this section we consider several extensions of the modeling framework that we have discussed thus far. One extension is immediate: We can obtain a fully Bayesian approach to model selection for the SVM by combining our work with the treatment of Mallick et al. (2005). That is, we form a

conditional likelihood from the hinge loss (also see Sollich, 2001) and assign a mixture of the point-mass distribution and the Silverman g-prior to the regression vector. In the following subsections we consider two additional extensions.

6.1 Multiple Kernel Learning

Kernel learning has emerged as an important theme in the machine learning community. We have provided a Bayesian foundation for kernel learning in Section 3. In particular, given a kernel function, we can estimate parameters of the kernel function. We now discuss how to extend this capability to the learning of combinations of kernels; the *multiple kernel learning problem* (Bach et al., 2004).

Assume that we are given q distinct kernel functions $K_l(\mathbf{x}_i, \mathbf{x}_j)$, for $l = 1, \ldots, q$. Correspondingly, we have q feature functions (say $\psi_l(\mathbf{x})$). In this case, the predictive function is expressed as

$$f(\mathbf{x}) = u + \sum_{l=1}^{q} \psi_l(\mathbf{x})' \mathbf{b}_l.$$

Letting $\mathbf{b}_l = g_l \Psi_l' \beta_l$ where $\Psi_l = [\psi_l(\mathbf{x}_1), \ldots, \psi_l(\mathbf{x}_n)]'$, $\beta_l = (\beta_{l1}, \ldots, \beta_{ln})'$ and $g_l \geq 0$, we have

$$f(\mathbf{x}) = u + \sum_{l=1}^{q} g_l \sum_{i=1}^{n} K_l(\mathbf{x}, \mathbf{x}_i) \beta_{li}.$$

Now we assign $\beta_l \sim N_n\big(\mathbf{0},\ \sigma^2 (\mathbf{K}^{(l)})^{-1}\big)$ and

$$g_l \sim \rho \delta_0(g_l) + (1-\rho) Ga(g_l | a_g/2,\ b_g/2),$$

where $\mathbf{K}^{(l)} = \Psi_l \Psi_l'$ ($n \times n$) is the lth kernel matrix, $\delta_0(\cdot)$ is a point-mass at zero and the user-specific parameter $\rho \in (0,1)$ controls the levels of the nonzero g_l. Thus, we only need to update the g_l instead of g in the Bayesian computation in Section 3. Note that kernel parameter learning and multiple kernel learning can be incorporated together.

6.2 Multi-class Learning

We consider the extension of our fully Bayesian modeling approach to a c-class ($c > 2$) classification problem where the class label $\mathbf{y}_i$ is a binary c-vector with values all zero except a one in position j if $\mathbf{x}_i$ belongs to the jth class. In this case, we define c regression vectors $\beta_j = (\beta_{1j}, \ldots, \beta nj)' \in \mathbf{R}^n$ and c auxiliary vectors $\mathbf{s}_j = (s_{1j}, \ldots, s_{nj})' \in \mathbf{R}^n$, $j = 1, \ldots, c$, for each class. We then have

$$\mathbf{s}_j = \mathbf{1}_n u_j + \mathbf{K} \beta_j + \mathbf{e}_j, \quad j = 1, \ldots, c,$$

where the $\mathbf{e}_j$ are i.i.d. from $N_n(\mathbf{0}, \mathbf{I}_n)$.

We now denote $\mathbf{u} = (u_1, \ldots, u_c)'$, $\mathbf{B} = [\beta_1, \ldots, \beta_c]$, $\mathbf{S} = [\mathbf{s}_1, \ldots, \mathbf{s}_c]$ and $\mathbf{E} = [\mathbf{e}_1, \ldots, \mathbf{e}_c]$. As in the binary problem, we also introduce a binary n-vector γ with either $\gamma_i = 1$ if $\mathbf{x}_i$ is an active vector or $\gamma_i = 0$ if $\mathbf{x}_i$ is not an active vector. Let $\mathbf{K}_\gamma'$ and $\mathbf{B}_\gamma$ be $\mathbf{K}'$ and $\mathbf{B}$ with the rows for which $\gamma_i = 0$ deleted. Thus, we can form the following sparse model:

$$\mathbf{S} = \mathbf{1}_n \mathbf{u}' + \mathbf{K}_\gamma \mathbf{B}_\gamma + \mathbf{E} = \widetilde{\mathbf{K}}_\gamma \widetilde{\mathbf{B}}_\gamma + \mathbf{E},$$

where $\widetilde{\mathbf{K}}_\gamma = [\mathbf{1}_n, \mathbf{K}_\gamma]$ and $\widetilde{\mathbf{B}}'_\gamma = [\mathbf{u}, \mathbf{B}'_\gamma]$. Now given γ, we treat the $\widetilde{\mathbf{B}}_\gamma$ as being independently from $N_{n_\gamma+1}(\mathbf{0}, \Sigma_\gamma^{-1})$.

To make the model identifiable, the constraint $\sum_{j=1}^c (u_j \mathbf{1}_n + \mathbf{K}\beta_j) = \mathbf{0}$ is typically required (see, e.g., Lee et al., 2004). Clearly, a sufficient condition for this constraint is that $\sum_{j=1}^c u_j = 0$ and $\sum_{j=1}^c \beta_j = \mathbf{0}$. To address this issue we impose the constraint $\sum_{j=1}^c \mathbf{s}_j = \mathbf{S}\mathbf{1}_c = \mathbf{0}$ and consider the following error model:

$$\mathbf{S} = \mathbf{1}_n \mathbf{u}'\mathbf{H} + \mathbf{K}_\gamma \mathbf{B}_\gamma \mathbf{H} + \mathbf{E}\mathbf{H} = \tilde{\mathbf{K}}_\gamma \tilde{\mathbf{B}}_\gamma \mathbf{H} + \mathbf{E}\mathbf{H}, \tag{13}$$

where $\mathbf{H} = \mathbf{I}_c - \frac{1}{c}\mathbf{1}_c\mathbf{1}'_c$ is the $c \times c$ centering matrix. Since $\tilde{\mathbf{B}}_\gamma \mathbf{H} \sim N_{n_\gamma+1,c}(\mathbf{0},\ \Sigma_{\gamma\gamma}^{-1} \otimes \mathbf{H})$ and $\mathbf{E}\mathbf{H} \sim N_{n,c}(\mathbf{0},\ \mathbf{I}_n \otimes \mathbf{H})$, we have $\mathbf{S} = \mathbf{S}\mathbf{H} \sim N_{n,c}(\mathbf{0},\ \mathbf{Q}_\gamma \otimes \mathbf{H})$. Here we use the formal of matrix-variate normal distributions; that is, $\mathbf{Z} \sim N_{m,p}(\mathbf{0}, \mathbf{M} \otimes \mathbf{N})$ if and only if $\mathrm{vec}(\mathbf{Z}') \sim N_{mp}(\mathbf{0}, \mathbf{M} \otimes \mathbf{N})$ where $\mathbf{Z} = [z_{ij}]$ is an $m \times p$ matrix, $\mathrm{vec}(\mathbf{Z}') = (z_{11}, z_{12}, \ldots, z_{mp})'$ is its arrangement in a stack, and $\mathbf{M} \otimes \mathbf{N}$ represents the Kronecker product of $\mathbf{M}$ and $\mathbf{N}$. Note that both $N_{n,c}(\mathbf{0},\ \mathbf{Q}_\gamma \otimes \mathbf{H})$ and $N_{n_\gamma+1,c}(\mathbf{0},\ \Sigma_{\gamma\gamma}^{-1} \otimes \mathbf{H})$ are singular matrix-variate distributions, because $\mathbf{H}$ is singular. Please refer to Gupta and Nagar (2000) for matrix-variate normal distributions and singular matrix-variate normal distributions.

We rewrite (13) in vector form as

$$\mathrm{vec}(\mathbf{S}) = (\mathbf{H} \otimes \widetilde{\mathbf{K}}_\gamma)\mathrm{vec}(\widetilde{\mathbf{B}}_\gamma) + (\mathbf{H} \otimes \mathbf{I}_n)\mathrm{vec}(\mathbf{E}).$$

We can apply the MCMC algorithm in Section 3.2 to the multi-class case. The main difference is in Step (a) for the update of $\mathbf{S}$. That is, in the multi-class probit setting, the relationship between the class labels and the auxiliary vectors becomes

$$y_{ij} = \begin{cases} 1 & \text{if } j = \mathrm{argmax}_{1 \le k \le c}\{s_{ik}\}, \\ 0 & \text{otherwise.} \end{cases}$$

Thus, the posterior distribution of each s_{ij} is truncated normal, $[s_{ij} | \gamma, u_j, \beta_j, y_{ij}] \sim N(u_j + \mathbf{k}'_i \beta_j, 1)$ subject to $s_{ij} > s_{il}$ for all $l \neq j$ if $y_{ij} = 1$.

Finally, it is straightforward to develop a sparse GP method for multi-class classification problems. We should note that Girolami and Rogers (2006) proposed a Bayesian multinomial probit regression model and derived a fully variational Bayesian method for multi-class Gaussian process classification. Specifically, $\mathbf{S}$ and $\mathbf{KB}$ respectively correspond to latent and manifest Gaussian random matrices in Bayesian multinomial probit regression. However, they did not consider the constraint $\sum_{j=1}^c (u_j \mathbf{1}_n + \mathbf{K}\beta_j) = \mathbf{0}$, which is theoretically necessary for making the multi-class classification problem identifiable.

7. Conclusion

In this paper we have discussed Bayesian generalized kernel mixed models, including Bayesian generalized kernel models and Gaussian processes for classification. In particular, we have proposed fully Bayesian kernel methods based on the Silverman g-prior and a Bayesian model averaging method. We have developed an MCMC algorithm for parameter estimation, model selection and posterior prediction. Because of the connection between kernel methods and Gaussian processes, the MCMC algorithm can be immediately applied to sparse Gaussian processes.

Sparsity is often treated using machinery that is not straightforward to emulate within the Bayesian paradigm (e.g., loss functions with discontinuous derivatives). In the current paper we

have provided a framework in which sparsity is treated explicitly using standard Bayesian tools. Our empirical results show that this framework can yield prediction performance that is comparable with the best non-Bayesian methods, while retaining the advantages (e.g., the natural treatment of hyperparameters and of uncertainty) of the Bayesian approach. The computational requirements of the framework are reasonable at the scale of the experiments we have performed; moreover, as emphasized in non-Bayesian treatments, the imposition of sparsity has computational advantages within our framework, advantages that we have only partially exploited in the work described here.

Acknowledgments

The authors would like to thank the Action Editor and three anonymous referees for their constructive comments and suggestions on the original version of this paper. Zhihua Zhang acknowledges support from the Natural Science Foundations of China (No. 61070239), the 973 Program of China (No. 2010CB327903), the Doctoral Program of Specialized Research Fund of Chinese Universities (No. 20090101120066), and the Fundamental Research Funds for the Central Universities.

Appendix A. Pseudo Matlab Code for the Updates of s and β_γ

Algorithm 1 Pseudo Matlab code for updates of **s** and β_γ

Input: $\tilde{\mathbf{K}}_\gamma$, Σ_γ, **s**;
Calculate and $\Upsilon_\gamma^{-1} = (\Sigma_\gamma + \tilde{\mathbf{K}}_\gamma' \tilde{\mathbf{K}}_\gamma)^{-1}$ and $\mathbf{Q}_\gamma^{-1} = \mathbf{I}_n - \tilde{\mathbf{K}}_\gamma \Upsilon_\gamma^{-1} \tilde{\mathbf{K}}_\gamma'$;
Calculate $\mathbf{A} = \Upsilon_\gamma^{-1} \tilde{\mathbf{K}}_\gamma'$, $\mathbf{b} = \mathbf{A}\mathbf{s}$ and $\mathbf{H} = \tilde{\mathbf{K}}_\gamma \mathbf{A}$;
for $i = 1$ **to** n **do**
  $a \leftarrow \mathbf{s}(i)$;
  $w \leftarrow \mathbf{H}(i,i)/(1-\mathbf{H}(i,i))$;
  $\rho \leftarrow 1 + w$;
  $\mu \leftarrow (1+w)\tilde{\mathbf{K}}_\gamma(i,:)\mathbf{b} - w\mathbf{s}(i)$;
  if $y(i) == 1$ **then**
    $s(i) \leftarrow ltnormrnd(\mu, \rho, 0)$; ▷ Generate from a left-truncated normal distribution
  else
    $s(i) \leftarrow rtnormrnd(\mu, \rho, 0)$; ▷ Generate a right-truncated normal distribution
    $\mathbf{b} \leftarrow \mathbf{b} + (\mathbf{s}(i) - a)\mathbf{A}(:,i)$;
  end if
end for
$\beta_\gamma \leftarrow mvnormrnd(\mathbf{b}, \Upsilon_\gamma^{-1}, 1)$. ▷ Generate from a multivariate normal distribution
Output: **s** and β_γ.

Appendix B. The Proof of Proposition 1

Proof. Recall that

$$K(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1}^{r} \psi_k(\mathbf{x}_i)\psi_k(\mathbf{x}_j).$$

The Karhunen-Loève expansion of $\zeta(\mathbf{x})$ is then given by

$$\zeta(\mathbf{x}) = \sum_{k=1}^{r} b_k \psi_k(\mathbf{x}),$$

where the b_k are random variables, which are given by $b_k = \frac{1}{l_k} \int \zeta(\mathbf{x}) \psi_k(\mathbf{x}) d\mathbf{x}$. It follows that the b_k are independent Gaussian variables with $E(b_k) = 0$ and $E(b_k^2) = g^{-1}$. We thus have the first part. To prove the second part of this proposition, we consider any n-dimensional vector $\zeta = (\zeta(\mathbf{x}_1), \ldots, \zeta(\mathbf{x}_n))'$. It is obvious that $E(\zeta(\mathbf{x}_i)) = 0$ and

$$\begin{aligned} E\big(\zeta(\mathbf{x}_i)\zeta(\mathbf{x}_j)\big) &= E\Big(\sum_{k=1}\sum_{k'} b_k \psi_k(\mathbf{x}_i) b_{k'} \psi_{k'}(\mathbf{x}_j)\Big) \\ &= g^{-1} \sum_{k=1}^{r} \psi_k(\mathbf{x}_i)\psi_k(\mathbf{x}_j) = g^{-1} K(\mathbf{x}_i, \mathbf{x}_j). \end{aligned}$$

This implies that ζ follows a multivariate normal distribution $N_n(\mathbf{0},\, g^{-1}\mathbf{K})$. Consequently, $\zeta(\mathbf{x})$ follows a Gaussian process.

References

J. H. Albert and S. Chib. Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669–679, 1993.

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the SMO algorithm. In *Proceedings of the Twenty-first International Conference on Machine Learning*, 2004.

J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. John Wiley and Sons, New York, 1994.

S. P. Brooks. Quantitative convergence diagnosis for MCMC via CUSUMS. *Statistics and Computing*, 8(3):267–274, 1998.

C. M. Carvalho, N. G. Polson, and J. G. Scott. The horseshoe estimator for sparse signals. *Biometrika*, 97(2):465–480, 2010.

S. Chakraborty, M. Ghosh, and B. K. Mallick. Bayesian nonlinear regression for large p small n problems. Technical report, Department of Statistics, University of Florida, 2005.

P. J. Diggle, J. A. Tawn, and R. A. Moyeed. Model-based geostatistics (with discussions). *Applied Statistics*, 47(3):299–350, 1998.

M. Figueiredo. Adaptive sparseness for supervised learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1150–1159, 2003.

E. I. George and R. E. McCulloch. Approaches for Bayesian variable selection. *Statistica Sinica*, 7:339–374, 1997.

M. A. Girolami and S. Rogers. Variational Bayesian multinomial probit regression with Gaussian process priors. *Neural Computation*, 18:1790–1817, 2006.

P. J. Green. Discussion of Dr. Silverman's paper. *Journal of the Royal Statistical Society, Series B*, 47(1):29, 1985.

P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82:711–732, 1995.

A.K. Gupta and D.K. Nagar. *Matrix Variate Distributions*. Chapman & Hall/CRC, 2000.

C. Hans. Bayesian lasso regression. *Biometrika*, 96(4):835–845, 2009.

D. A. Harville. Maximum likelihood approaches to variance component estimation and to related problems. *Journal of the American Statistical Association*, 72(358):320–338, 1977.

T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, New York, 2001.

C. C. Holmes and L. Held. Bayesian auxiliary variable models for binary and multinomial regression. *Bayesian Analysis*, 1(1):145–168, 2006.

R. Kohn, M. Smith, and D. Chan. Nonparametric regression using linear combinations of basis functions. *Statistics and Computing*, 11:313–322, 2001.

N. D. Lawrence, M. Seeger, and R. Herbrich. Fast sparse Gaussian process methods: the informative vector machine. In *Advances in Neural Information Processing Systems 15*, 2003.

Y. Lee, Y. Lin, and G. Wahba. Multicategory support vector machines: theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–81, 2004.

Q. Li and N. Lin. The Bayesian elastic net. *Bayesian Analysis*, 5(1):151–170, 2010.

F. Liang, R. Paulo, G. Molina, M. A. Clyde, and J. O. Berger. Mixtures of g-priors for Bayesian variable selection. *Journal of the American Statistical Association*, 103(481):410–423, 2008.

F. Liang, K. Mao, M. Liao, R. F. MacLehose, and D. B. Dunson. Nonparametric Bayesian kernel models. In *Discussion Paper 2005-09, Duke University ISDS*, 2009.

R. F. MacLehose and D. B. Dunson. Nonparametric Bayes kernel-based priors for functional data analysis. *Statistica Sinica*, 19:611–629, 2009.

B. K. Mallick, D. Ghosh, and M. Ghosh. Bayesian classification of tumours by using gene expression data. *Journal of the Royal Statistical Society Series B*, 67:219–234, 2005.

K. V. Mardia, J. T. Kent, and J. M. Bibby. *Multivariate Analysis*. Academic Press, New York, 1979.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman and Hall, New York, 1989.

T. P. Minka. Expectation propagation for approximate Bayesian inference. In *Proceedings of the 17th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 362–369, 2001.

R. M. Neal. Regression and classification using Gaussian process priors (with discussion). In J. M. Bernardo, J. O. Berger, A. P. Dawid, and A. F. M. Smith, editors, *Bayesian Statistics*, volume 6, pages 475–501. Oxford University Press, 1999.

D. J. Nott and P. J. Green. Bayesian variable selection and the Swendsen-Wang algorithm. *Journal of Computational and Graphical Statistics*, 13(1):1–17, 2004.

T. Park and G. Casella. The Bayesian lasso. *Journal of the American Statistical Association*, 103 (482):681–686, 2008.

N. S. Pillai, Q. Wu, F. Liang, S. Mukherjee, and R. L. Wolpert. Characterizing the function space for Bayesian kernel models. *Journal of Machine Learning Research*, 8:1769–1797, 2007.

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

A. E. Raftery, D. Madigan, and D. Hoeting. Bayesian model averaging for linear regression. *Journal of the American Statistical Association*, 92:179–191, 1997.

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

G. Rätsch, T. Onoda, and K. Müller. Soft margins for Adaboost. *Machine Learning*, 42:287–320, 2001.

N. Sha, M. Vannucci, M. G. Tadesse, P. J. Brown, I. Dragoni, N. Davies T. C. Roberts, A. Contestabile, M. Salmon, C. Buckley, and F. Falciani. Bayesian variable selection in multinomial probit models to identify molecular signatures of disease stage. *Biometrics*, 60:812–819, 2004.

B. W. Silverman. Some aspects of the spline smoothing approach to non-parametric regression curve fitting (with discussion). *Journal of the Royal Statistical Society, Series B*, 47(1):1–52, 1985.

M. Smith and R. Kohn. Nonparametric regression using Bayesian variable selection. *Journal of Econometrics*, 75:317–344, 1996.

A. J. Smola and P. Bartlett. Sparse greedy Gaussian process regression. In *Advances in Neural Information Processing Systems 13*, 2001.

E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Neural Information Processing Systems 18*, 2006.

E. L. Snelson. *Flexible and Efficient Gaussian Process Models for Machine Learning*. PhD thesis, University College London, 2007.

P. Sollich. Bayesian methods for support vector machines: evidence and predictive class probabilities. *Machine Learning*, 46:21–52, 2001.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1996.

M. E. Tipping. Sparse Bayesian learning and the relevance vector machine. *Journal of Machine Learning Research*, 1:211–244, 2001.

V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

G. Wahba. *Spline Models for Observational Data*. SIAM, Philadelphia, 1990.

M. West. Bayesian factor regression models in the "large p, small n" paradigm. In J. M. Bernardo, M. J. Bayarri, J. O. Berger, A. P. Dawid, D. Heckerman, A. F. M. Smith, and M. West, editors, *Bayesian Statistics 7*, pages 723–732. Oxford University Press, 2003.

C. K. I. Williams and D. Barber. Bayesian classification with Gaussian processes. *IEEE Trans. PAMI*, 20(12):1342–1351, 1998.

C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Advances in Neural Information Processing Systems 13*, 2001.

A. Zellner. On assessing prior distributions and Bayesian regression analysis with $g-$prior distributions. In P. K. Goel and A. Zellner, editors, *Bayesian Inference and Decision Techniques: Essays in Honor of Bruno de Finetti*, pages 233–243. North-Holland, Amsterdam, 1986.

Z. Zhang and M. I. Jordan. Bayesian multicategory support vector machines. In *the Twenty-Second Conference on Uncertainty in Artificial Intelligence (UAI)*, 2006.

Z. Zhang, G. Wu, and E. Y Chang. Semiparametric regression using Student t processes. *IEEE Transactions on Neural Networks*, 18(6):1572–1588, 2007.

Z. Zhang, M. I. Jordan, and D.-Y. Yeung. Posterior consistency of the Silverman g-prior in Bayesian model choice. In *Advances in Neural Information Processing Systems 22*, 2008.

J. Zhu and T. Hastie. Kernel logistic regression and the import vector machines. *Journal of Computational and Graphical Statistics*, 14(1):185–205, 2005.