RC 22010 (22010) 4 April 2001 **Mathematics**

IBM Research Report

Some Approximation Bounds for Space Gaussian Processes

Tong Zhang
IBM Research Division
IBM Thomas J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598

Research Division
Almaden - Austin - Beijing - Haifa - T. J. Watson - Tokyo - Zurich

Some Approximation Bounds for Sparse Gaussian Processes

Tong Zhang TZHANG@WATSON.IBM.COM
IBM T.J. Watson Research Center, Yorktown Heights, NY 10598 USA

Abstract

Gaussian processes have been widely applied to regression problems with good performance. However, they can be computationally expensive. Recently, there have been studies on using sparse approximations in Gaussian processes in order to reduce the computational cost. In this paper, we investigate properties of certain sparse algorithms that approximately solve a Gaussian process. We obtain approximation bounds, and compare our results with related methods.

1. Introduction

Gaussian processes (Cristianini & Shawe-Taylor, 2000) have recently attracted much attention since they are relatively simple and achieve good performance for regression problems. From the convex analysis point of view, they can be regarded as dual forms of ridge regressions (Hoerl & Kennard, 1970). By using the idea of kernels, Gaussian processes can effectively solve ridge regression problems in infinite dimensional Hilbert spaces. Therefore the kernel representation in Gaussian processes allows the computation of linear regression in certain infinite dimensional spaces tractable.

One disadvantage associated with Gaussian processes is that there is a parameter for every data point. This implies that the computation can still be very expensive when we have a large number of training data. In order to overcome this problem, various methods have been proposed in the literature to accelerate the computation of Gaussian processes. In this paper, we are specially interested in certain sparse Gaussian process algorithms such as those in (Csató & Opper, 2000; Smola & Bartlett, 2000).

Sparse representation is naturally attractive. Not only it may considerably reduce computation, but also it complies with the philosophy of Occam's Razor which implies that the simplest solution (in our case, a solution with the fewest nonzero coefficients) is likely to be the best solution among all possible solutions. Therefore sparse approximation is also an important problem by itself.

Many methods have been proposed over the years to achieve sparseness. We shall only mention ideas that are directly related to this paper. For certain regression problems in Hilbert spaces, Jones observed in (Jones, 1992) that by using greedy approximation, one can achieve an error rate of $O(1/k)$ with a sparse representation of k nonzero coefficients. The same idea in an algorithmic form led to the *matching pursuit* method in (Mallat & Zhang, 1993). The idea has also been applied in (Barron, 1993) to analyze the approximation property of sigmoidal functions including neural networks. These methods are in the style of greedy approximation algorithms. It is also possible to directly impose the sparseness constraint into the problem formulation itself. However, a difficulty is that the resulting formulation is typically NP-hard (for example, see (Natarajan, 1995)). One way to remedy this problem is to include a 1-norm regularization condition in the ridge regression formulation, as in *basis pursuit* (Chen et al., 1999). In basis pursuit, we still have a convex programming problem, which can be solved efficiently. Furthermore, 1-norm regularization leads to a sparse weight vector.

The above mentioned methods directly deal with the primal form of regression. They compute sparse coefficients for the primal weight vector. As we have mentioned earlier, kernel methods such as Gaussian processes are expressed in the dual form of regression. The dual representation has the advantage that a finite dimensional kernel formulation corresponds to possibly infinite dimensional primal formulation. It is thus useful to consider an approximation of the solution by using a sparse dual representation (the primal weight vector can still be dense). One way to achieve dual sparseness is through Vapnik's ϵ-insensitive loss function in a support vector machine (Vapnik, 1998). Its relationship with the (primal) basis pursuit regression has been discussed in (Girosi, 1998). Note that although Girosi demonstrated certain "equivalence" be-

tween modifications of basis pursuit and some support vector machines, there are still some fundamental differences between the primal and the dual representations that cannot be captured by such an "equivalence". Some of these differences become more apparent when we compare our results with primal formulation results such as that of matching pursuit later in the paper.

In this paper, we are mainly interested in achieving sparsity through greedy approximation. This approach has a number of advantages over a direct sparse convex programming formulation such as basis pursuit or support vector learning. One advantage is that the sparsity is directly controlled in a greedy approximation algorithm. Another advantage is that greedy approximation does not change the objective optimization function, while a direct method such as basis pursuit (or SVM) modifies the objective function by including a 1-norm regularization (or ϵ-insensitive loss). The third advantage, which is more important for this paper, is that in general, provably good approximation bounds can be obtained for greedy methods, but not for direct convex programming methods. The reason of this becomes clear later in the paper.

Methods studied in this paper are closely related to some recent works. One method is a greedy Gaussian process method developed in (Smola & Bartlett, 2000), which corresponds to the dual of primal form matching pursuit. Another related method is a non-greedy algorithm described in (Csató & Opper, 2000).

However, the online update scheme in (Csató & Opper, 2000) only deals with algorithmic issues. It can essentially be regarded as a special form of rank-one matrix updates algorithms widely studied in numerical linear algebra (for example, see relevant sections in (Golub & Van Loan, 1996) and references therein). In particular, the work does not contain any theoretical result concerning the rate of approximation. Although some bounds were discussed in (Smola & Bartlett, 2000), those results rely on (Natarajan, 1995), which unlike bounds in (Barron, 1993; Jones, 1992; Mallat & Zhang, 1993), are not true approximation bounds. For example, the bound in (Smola & Bartlett, 2000) specifies the approximation quality of the proposed greedy procedure in term of the best possible sparse approximation. Note that for a true approximation bound, the latter quantity itself needs to be estimated. Also the bound relies on the smallest singular value of the column-normalized Gram matrix which in practice can become zero (or very small) and trivialize the bound.

In this paper, we analyze a number of sparse approximation algorithms for Gaussian processes using a technique related to the analysis of matching pursuit in (Barron, 1993; Jones, 1992; Mallat & Zhang, 1993). However, we study Gaussian processes in the dual representation, which has an objective function different from the primal form of matching pursuit. Therefore, in order to analyze the system, we also employ ideas from online learning. As we see later, this analysis leads to some interesting theoretical consequences and algorithmic implications.

This paper is organized as follows. In Section 2, we review the duality between ridge regression and Gaussian processes, and establish some notations and conventions used throughout the paper. Section 3 outlines some sparse Gaussian process algorithms. We then derive approximation bounds for these algorithms and compare them with related results. In Section 4, we use numerical examples to illustrate some aspects of the proposed algorithms. Section 5 summarizes the paper.

2. Preliminaries

We have mentioned that Gaussian processes are expressed in dual forms of regression problems. Since this duality is important in our discussion and in distinguishing primal greedy algorithms such as matching pursuit from their dual counterparts, we shall briefly review the basic ideas. This discussion also introduces notations which we use throughout the paper.

We consider the standard ridge regression model in the following setting. Assume we have a set of feature vectors $x_1, \ldots, x_n$, with corresponding real-valued output variables $y_1, \ldots, y_n$. Our goal is to find a linear weight vector w such that $y \approx w^T x$ for all future feature data x. In this paper, we assume that data x belong to a Hilbert space H that may be infinite dimensional. In ridge regression, we obtain a weight $\hat{w}$ by minimizing the following primal objective function:

$$\hat{w} = \arg\min_w \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(w^T x_i - y_i)^2 + \frac{\lambda}{2} w^2, \qquad (1)$$

where $w^T x$ is the inner product of the Hilbert space H, and w^2 denotes $w^T w$.

By differentiating (1) with respect to w at the optimal solution $\hat{w}$, we obtain a representation of $\hat{w}$ in the following form

$$\hat{w} = \sum_{i=1}^{n} \hat{\alpha}_i x_i, \qquad (2)$$

$$\hat{\alpha}_i = -\frac{1}{\lambda n}(\hat{w}^T x_i - y_i), \qquad i = 1, \ldots, n. \qquad (3)$$

Equation (1) is called the *primal formulation*, which involves the *primal variable* w. $\hat{\alpha}$ is the *dual variable*. To obtain the corresponding *dual formulation* in the dual variable α, we can eliminate $\hat{w}$ and obtain

$$\hat{\alpha}_i = -\frac{1}{\lambda n}(\sum_{k=1}^{n} \hat{\alpha}_k x_k^T x_i - y_i), \qquad i = 1, \ldots, n. \quad (4)$$

It is easy to check that $\hat{\alpha}$ is the solution of the following dual optimization problem:

$$\hat{\alpha} = \arg\max_{\alpha}[\sum_{i=1}^{n} \lambda(-\frac{\lambda n}{2}\alpha_i^2 + \alpha_i y_i) - \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \alpha_i \alpha_k x_i^T x_k]. \quad (5)$$

The prediction $\hat{w}^T x$ can be expressed in the dual variable as:

$$\hat{w}^T x = \sum_{k=1}^{n} \hat{\alpha}_k x_k^T x. \quad (6)$$

Since in the dual formulation (5) and in the dual linear prediction representation (6), we only need the data-space inner product of the form $x_k^T x$, we may replace it by any symmetric positive (semi)-definite Kernel function $K(x_k, x)$, which leads to the general form of Gaussian processes (Cristianini & Shawe-Taylor, 2000).

An equivalent formulation of Gaussian processes can be obtained by directly inserting (2) into the primal formulation (1) to obtain a system involving the dual variable α:

$$\hat{\alpha} = \arg\min_{\alpha} \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(\sum_{k=1}^{n} \alpha_k x_k^T x_i - y_i)^2 + \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \alpha_i \alpha_k x_i^T x_k. \quad (7)$$

Clearly, kernel forms of (7) can be obtained by replacing each inner product $x_k^T x$ with a symmetric positive (semi)-definite Kernel function $K(x_k, x)$. The solution of (5) is equivalent to the solution of (7).

For all primal vector w, we define

$$R(w) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(w^T x_i - y_i)^2 + \frac{\lambda}{2}w^2,$$

and for any dual vector α, we define

$$Q(\alpha) = [\sum_{i=1}^{n} \lambda(-\frac{\lambda n}{2}\alpha_i^2 + \alpha_i y_i) - \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \alpha_i \alpha_k x_i^T x_k].$$

We also define

$$\Delta R(w) = R(w) - \inf_{w} R(w)$$

and

$$\Delta Q(\alpha) = \sup_{\alpha} Q(\alpha) - Q(\alpha).$$

Lemma 2.1 *At the optimal solution $\hat{w}$ of (1) and $\hat{\alpha}$ of (5), $R(\hat{w}) = Q(\hat{\alpha})$.*

Proof. Since $\hat{w}$ and $\hat{\alpha}$ satisfies (2), therefore

$$\begin{aligned} Q(\hat{\alpha}) =& \sum_{i=1}^{n} \lambda(-\frac{\lambda n}{2}\hat{\alpha}_i^2 + \hat{\alpha}_i y_i) - \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \hat{\alpha}_i \hat{\alpha}_k x_i^T x_k \\ =& [\sum_{i=1}^{n} \lambda(-\frac{\lambda n}{2}\hat{\alpha}_i^2 + \hat{\alpha}_i(y_i - \hat{w}^T x_i))] + \\ & [\lambda \hat{w}^T \sum_{i=1}^{n} \hat{\alpha}_i x_i - \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \hat{\alpha}_i \hat{\alpha}_k x_i^T x_k] \\ =& [\sum_{i=1}^{n} \lambda(-\frac{\lambda n}{2}(\frac{1}{\lambda n}(\hat{w}^T x_i - y_i))^2 - \\ & \frac{1}{\lambda n}(\hat{w}^T x_i - y_i)(y_i - \hat{w}^T x_i))] \quad + [\lambda \hat{w}^T \hat{w} - \frac{\lambda}{2}\hat{w}^T \hat{w}] \\ =& R(\hat{w}) \qquad \square \end{aligned}$$

Lemma 2.2

$$\Delta R(w) \leq \frac{1}{2\lambda}(\partial R(w)/\partial w)^2 = \frac{1}{2\lambda}(\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w)^2.$$

Proof. Let

$$\alpha_i = -\frac{1}{\lambda n}(w^T x_i - y_i), \qquad i = 1, \ldots, n.$$

It follows from Lemma 2.1 that $\Delta R(w) \leq R(w) - Q(\alpha)$. We also have

$$\begin{aligned} Q(\alpha) =& \sum_{i=1}^{n} \lambda(-\frac{\lambda n}{2}\alpha_i^2 + \alpha_i y_i) - \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \alpha_i \alpha_k x_i^T x_k \\ =& [\sum_{i=1}^{n} \lambda(-\frac{\lambda n}{2}\alpha_i^2 + \alpha_i(y_i - w^T x_i))] + \\ & [\lambda w^T \sum_{i=1}^{n} \alpha_i x_i - \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \alpha_i \alpha_k x_i^T x_k] \\ =& \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(w^T x_i - y_i)^2 + \\ & [\lambda w^T \sum_{i=1}^{n} \alpha_i x_i - \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{n} \alpha_i \alpha_k x_i^T x_k] \\ =& \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}(w^T x_i - y_i)^2 + \frac{\lambda}{2}w^2 - \frac{\lambda}{2}(w - \sum_{i=1}^{n} \alpha_i x_i)^2 \\ =& R(w) - \frac{1}{2\lambda}(\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w)^2. \end{aligned}$$

This proves the Lemma. $\square$

Lemma 2.3 *Let $\hat{w}$ be the solution of (1), then*

$$\Delta Q(\alpha) \geq \frac{1}{2n}\sum_{i=1}^{n}(\hat{w}^T x_i - y_i + \lambda n \alpha_i)^2.$$

Proof. Let $\hat{\alpha}$ be the solution of (5). For any α, using (4), we have

$$\begin{aligned}
\Delta Q(\alpha) =& Q(\hat{\alpha}) - Q(\alpha) + \\
& \lambda\sum_{i=1}^{n}(\lambda n\hat{\alpha}_i - y_i + \sum_{k=1}^{n}\hat{\alpha}_k x_k^T x_i)(\hat{\alpha}_i - \alpha_i) \\
=& \lambda\sum_{i=1}^{n}[\frac{\lambda n}{2}(\hat{\alpha}_i - \alpha_i)^2 + \\
& \quad \frac{1}{2}\sum_{k=1}^{n}(\hat{\alpha}_i - \alpha_i)(\hat{\alpha}_k - \alpha_k)x_i^T x_k] \\
\geq& \lambda\sum_{i=1}^{n}\frac{\lambda n}{2}(\hat{\alpha}_i - \alpha_i)^2 \\
=& \frac{1}{2n}\sum_{i=1}^{n}(\hat{w}^T x_i - y_i + \lambda n\alpha_i)^2.
\end{aligned}$$

The inequality follows from the semi positive definiteness of the Gram matrix $[x_i^T x_k]$. □.

3. Sparse Approximation for Gaussian Processes

3.1 Sparse Algorithms

In order to obtain a sparse representation of the dual variable, it is necessary to work with system (7) instead of (5). The reason that system (5) is not appropriate for obtaining a sparse representation can be seen from Lemma 2.3. For a regression problem containing noise, there can be a significant portion of data such that $\hat{w}^T x_i - y_i$ is bounded away from zero. If α_i is sparse, then $\Delta Q(\alpha)$ has to be bounded away from zero. Note also that the ϵ-insensitive loss support vector formulation works with a perturbed form of (5). Therefore such a formulation is not very useful for noisy data either.

Equation (7) is simply equation (1) with the primal variable w substituted by (2). In order to obtain a sparse dual representation in (7), we only need to approximately minimize (1) with w expressed in the form $w = \sum_i \alpha_i x_i$. It is easy to see that under this assumption, the objective function $R(w) = R(\sum_i \alpha_i x_i)$ can be expressed in a kernel form. In this paper, we consider the following algorithms that approximately solve a Gaussian process with a sparse dual variable.

Algorithm 1 *(Greedy sparse Gaussian process)*

let $w^0 = 0$
for $k = 1, 2, \ldots$
  find $i_k \in \{1, \ldots, n\}, \alpha_k$ and β_k that minimize $R(\beta_k w^{k-1} + \alpha_k x_{i_k})$
  let $w^k = \beta_k w^{k-1} + \alpha_k x_{i_k}$
end

Algorithm 2 *(Random sparse Gaussian process)*

Given sparseness k
Randomly draw k samples $x_{i_1}, \ldots, x_{i_k}$ uniformly from $\{x_1, \ldots, x_n\}$
find $\alpha_1, \ldots, \alpha_k$ that minimize $R(\sum_{j=1}^{k}\alpha_j x_{i_j})$
let $w_k = \sum_{j=1}^{k}\alpha_j x_{i_j}$

Algorithm 3 *(κ-greedy sparse Gaussian process)*

Given integer κ: $1 \leq \kappa \leq n$
let $w^0 = 0$
for $k = 1, 2, \ldots$
  Randomly draw a subset S_k of size κ uniformly from $\{1, \ldots, n\}$
  find $i_k \in S_k, \alpha_k$ and β_k that minimize $R(\beta_k w^{k-1} + \alpha_k x_{i_k})$
  let $w^k = \beta_k w^{k-1} + \alpha_k x_{i_k}$
end

Algorithm 1 is a greedy sparse approximation algorithm. A similar algorithm has been proposed in (Smola & Bartlett, 2000). However, there are two major differences. One difference is that unlike the algorithm proposed in (Smola & Bartlett, 2000), we do not utilize the dual functional $Q(\alpha)$. The reason has been explained earlier: we show that in the most general situation, it is not possible to find a sparse solution α that approximately maximizes $Q(\alpha)$. The second difference is that at each step, in addition to α_k, we also seek a scaling factor β_k that shrinks the current weight w^{k-1}. As we shall see later, this scaling is important in our analysis.

Algorithm 2 is a random sparse approximation method. We show that this algorithm has similar worst case approximation property as the greedy approximation algorithm. However, in practice, the greedy algorithm is usually superior. The full greedy algorithm is computationally quite expensive, therefore a compromise is to consider searching through i_k in a subset of $\{1, \ldots, n\}$ at each step as in Algorithm 3. This idea of choosing a random subset, suggested in (Smola & Bartlett, 2000), is a compromise between the full greedy sparse algorithm and the random sparse algorithm. In the kernel formulation, we

make the assumption that the inner product $k(x_i, x_k)$ can be computed in $O(1)$ time. Then the computational costs of the three algorithms can be summarized in Table 1.

Table 1. Computational costs for k-term sparse approximation

algorithm	full greedy	random	κ-greedy
computation	$O(k^2n^2)$	$O(k^2n)$	$O(\kappa k^2 n)$

3.2 Analysis of Sparse Algorithms

Let w be an approximate solution to (1). We consider the following type of stochastic gradient update rule with data x_u:

$$w^u = w - \eta((w^T x_u - y_u)x_u + \lambda w). \tag{8}$$

Obviously, if w is of form (2), then w^u is also of form (2). We would like to find u such that it minimizes

$$R(w^u) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}(w^{uT}x_i - y_i)^2 + \frac{\lambda}{2}w^{u2}.$$

Note that

$$\begin{aligned}
&R(w^u) - R(w)\\
=&\frac{1}{2}[\frac{1}{n}\sum_{i=1}^{n}((w^u - w)^T x_i)^2 + \lambda(w^u - w)^2] +\\
&\quad (w^u - w)^T[\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w]\\
\le&\frac{1}{2}(w^u - w)^2[\frac{1}{n}\sum_{i=1}^{n}x_i^2 + \lambda] +\\
&\quad (w^u - w)^T[\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w].
\end{aligned}$$

Therefore

$$\begin{aligned}
&\frac{1}{n}\sum_{u=1}^{n}R(w^u) - R(w)\\
\le&\frac{1}{2n}\sum_{u=1}^{n}(w^u - w)^2[\frac{1}{n}\sum_{i=1}^{n}x_i^2 + \lambda] +\\
&\quad \frac{1}{n}\sum_{u=1}^{n}(w^u - w)^T[\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w]\\
=&\frac{\eta^2}{2n}\sum_{i=1}^{n}((w^T x_i - y_i)x_i + \lambda w)^2(\frac{1}{n}\sum_{i=1}^{n}x_i^2 + \lambda) -\\
&\quad \eta[\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w]^2.
\end{aligned}$$

If we pick η such that

$$\eta = \frac{[\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w]^2}{\frac{1}{n}\sum_{i=1}^{n}((w^T x_i - y_i)x_i + \lambda w)^2(\frac{1}{n}\sum_{i=1}^{n}x_i^2 + \lambda)},$$

then

$$\begin{aligned}
\frac{1}{n}\sum_{u=1}^{n}R(w^u) =& R(w) - \frac{\eta}{2}[\frac{1}{n}\sum_{i=1}^{n}(w^T x_i - y_i)x_i + \lambda w]^2\\
\le& R(w) - \eta\lambda\Delta R(w).
\end{aligned}$$

The above inequality follows from Lemma 2.2. Now, we assume $M = \max_i \|x_i\|_2$, then use Lemma 2.2 and after some simple algebra, we obtain

$$\eta \ge \frac{2\lambda\Delta R(w)}{((2R(w) - \lambda w^2)M^2 + 2\lambda\Delta R(w))(M^2 + \lambda)}.$$

This leads to the following result:

Lemma 3.1 *Let $M = \max_i \|x_i\|_2$. For all vector w,*

$$\frac{1}{n}\sum_{i=1}^{n}\inf_{\alpha,\beta}R(\alpha x_i + \beta w) \le R(w) - \frac{\lambda^2(\Delta R(w))^2}{R(w)(M^2 + \lambda)^2}.$$

Now, by using induction and applying Lemma 3.1 repeatedly, we obtain the following approximation bounds on sparse Gaussian process algorithms.

Theorem 3.1 *Let $M = \max_i \|x_i\|_2$. Then in Algorithm 1,*

$$\Delta R(w^k) \le \frac{R(0)(M^2 + \lambda)^2}{\lambda^2 k + (M^2 + \lambda)^2}.$$

Furthermore, in Algorithm 2 and Algorithm 3, the expected approximation error satisfies:

$$E\Delta R(w^k) \le \frac{R(0)(M^2 + \lambda)^2}{\lambda^2 k + (M^2 + \lambda)^2},$$

where the expectation E is taken over all possible randomized instances.

Although bounds described in Theorem 3.1 are the same for all three sparse Gaussian process algorithms considered in this paper, in reality, Algorithm 1 gives better sparse approximation than Algorithm 3, which gives better sparse approximation than Algorithm 2. An experiment is included in Section 4 to demonstrate this phenomenon. One may also draw this conclusion from our analysis. Note that Theorem 3.1 relies on Lemma 3.1, which is an average bound for stochastic gradient update (8). However in reality, a greedy algorithm can decrease the objective function much

more quickly than the average case. Unfortunately, this difference is difficult to quantify in our theoretical analysis.

We have only considered the worst case behavior in Theorem 3.1 without any attempt to optimize the bounds. These results show that with fixed λ, the approximation error decreases at a rate of $O(1/k)$ where k is the number of non-zero coefficients in the sparse approximation. An interesting observation is that bounds in Theorem 3.1 do not depend on the sample size n. There are a few advantages of our analysis compared with results in (Smola & Bartlett, 2000). For example, their bound is not a true approximation bound since it is only a ratio of the sparsity achieved by their greedy algorithm to the best possible sparse approximation. However, in a true approximation bound, the latter should also be estimated. In addition, the bound becomes trivial when the Gram matrix $G = [x_i^T x_j]$ after column-normalization becomes singular. Theorem 3.1 remedies these problems. Our bounds have styles similar to that of matching pursuit: both achieves an approximation rate of $O(1/k)$.

Bounds given in Theorem 3.1 are expressed in terms of the initial approximation $R(0)$, the maximum norm $M = \max_i \|x_i\|_2$, and the regularization parameter λ. The first two quantities, which also appear in the matching pursuit approximation bound, are easy to understand. In addition, we show that the dependency of regularization parameter λ cannot be avoided in the worst case scenario. The following example demonstrates that if we allow $\lambda \to 0$ and $n \to \infty$, then any k-term (with k fixed) sparse approximation error can be bounded below by a positive number that is independent of k. Thus if we let $\lambda \to 0$, the best possible k-term sparse approximation error does not decrease to zero as $k \to \infty$.

We consider an orthonormal basis $\{e_i : i = 1, \ldots, d\}$ in a d-dimensional real vector space. Let $x_i = e_i$ and $y_i = 1$ for $i = 1, \ldots, n$ where $n = d$. For any $\lambda > 0$, the optimal solution $\hat{w}$ of (1) is $\hat{w} = \sum_{i=1}^{n} \frac{1}{1+\lambda n} e_i$. For any k, the best k-term approximation is given by $w_k = \sum_{j=1}^{k} \frac{1}{1+\lambda n} e_{i_j}$ where $\{i_j : j = 1, \ldots, k\}$ is any k-member subset of $\{1, \ldots, n\}$. We have $\Delta R(w_k) = R(w_k) - R(\hat{w}) = \frac{1}{2(1+\lambda n)}(1 - k/n)$. Clearly, for all k, if we allow $n \to \infty$ and $\lambda = o(1/n)$, then $\Delta R(w_k) \to 0.5$.

3.3 Relationship with Other Methods

Since our analysis of sparse Gaussian process algorithms is based on the stochastic gradient descent update (8), it is closely related to online learning, which in general is also based on stochastic gradient descent. However, the usual mistake bound framework in online learning is interested in worst case upper bounds for stochastic gradient descent over a sequence of data. An example of such analysis for (non-regularized) least squares regression can be found in (Cesa-Bianchi et al., 1996). On the other hand, in Lemma 3.1, we consider upper bounds on the average improvement of stochastic gradient descent over the training data. This implies that we use different criteria to measure the performance of a stochastic gradient descent rule. Furthermore, the technique used in (Cesa-Bianchi et al., 1996) is not suitable for analyzing ridge regression which includes a regularization term. Consequently, the idea used in this paper is new and different from what was used in (Cesa-Bianchi et al., 1996), although the two ideas are related in a certain way.

Algorithms in this paper are also related to active learning. In active learning, we choose a subset of most informative data to train a predictor. Therefore this scheme applied to Gaussian processes can be regarded as a method to achieve sparse approximation. However, the goal of active learning is to reduce the effort of obtaining output data y for input data x. Therefore in active learning, the output y associated with an input x is unknown until the input is included in training. In our case, both x and y are known, and the purpose is to obtain a simpler representation or to reduce the computational cost. Therefore our algorithms that take advantage of output data y for all input data x can achieve better sparsity than active learning.

As we have mentioned earlier, our algorithms are also closely related to matching pursuit type greedy approximation. However, in matching pursuit, one works with the primal variable w and seeks a sparse representation of $\hat{w}$.[1] In our case, we seek a sparse representation of the dual variable $\hat{\alpha}$. Techniques used in the proofs of approximation rates are related. Both employ randomization which results in a rate of $O(1/k)$. However, in matching pursuit, the randomization is with respect to the measure $\hat{w}/\|\hat{w}\|_1$,[2] which in general is not available for learning problems. This means that the proof of matching pursuit is non-constructive, and one has to employ the full greedy approach to achieve the approximation rate of $O(1/k)$. Our proof given in this paper is constructive, and the randomization is with respect to a uniform distribution over the observed samples. For learning problems, this corresponds to the empirical distribution. As a result, for

[1] One may also apply the matching pursuit style analysis directly to $\hat{\alpha}$, which does not result in our bounds. Such a comparative analysis will be given in the full paper.

[2] We thus need to assume that $\|\hat{w}\|_1$ is bounded, and the final approximation rate depends on this quantity.

sparse Gaussian process algorithms, the approximation rate of $O(1/k)$ can be obtained by simple random sampling. This difference shows a significant computational advantage of dual sparse approximation.

Another related approach is to apply sparse approximation directly to the dual objective function $Q(\alpha)$. By Lemma 2.3, we know that this approach fails for noisy problems. As we have pointed out, SVMs with Vapnik's ϵ-insensitive loss slightly modifies the dual objective function $Q(\alpha)$ so that its solution may directly yield a sparse dual variable $\hat{\alpha}$. This is similar to basis pursuit which incorporates primal sparseness condition on $\hat{w}$ into the primal objective function. Our analysis implies that SVMs can only produce sparse coefficients for nearly noise-free problems. However, our sparse approximation algorithms have provable approximation bounds both for noise-free and for noisy problems.

4. Experiments

In this section, we use experiments to compare algorithms proposed in this paper. We use "greedy" to denote Algorithm 1, "κ-greedy" to denote Algorithm 3, and "random" to denote Algorithm 2.

We consider a regression problem in $d = 1000$ dimension, with $n = 1000$ samples. Each input datum x is a sparse vector, where each of its components is a random number independently generated as: 0 with probability 0.99, and a uniform random number in $(0, 1)$ with probability 0.01. We also generate a random vector $\bar{w}$ with each component a uniform random number in $(0, 1)$. We then set $y_i = \hat{w}^T x_i + n_i$ where n_i is a uniform noise in $(-1, 1)$.

We set $\lambda = 0.1$, and compare the three algorithms considered in this paper. In this example, $R(\hat{w}) \approx 3.03$ and $\Delta R(0) \approx 0.90$. In Figure 1, we compare the sparse approximation performance of the algorithms, where sparseness k is the number of nonzero terms in the approximation, and approximation error is ΔR. For Algorithm 2 and Algorithm 3 that are random, the curves are obtained by averaging over 5 different random runs. In Figure 2, we plot the average (the "avg" curves), the best (the "min" curves), and the worst (the "max" curves) values of ΔR over the 5 different random runs. This result suggests that the variance caused by randomization is very small. Figure 3 plots $k\Delta R$, which can be compared with the worst case approximation error bounds in Theorem 3.1 that are of the form $k\Delta R = O(1)$. Since these theoretical results are worst scenario upper bounds, we see that the predicted k-inverse law is not exactly observed in practice. However qualitatively, the experimental result is consistent with our theoretical bounds.

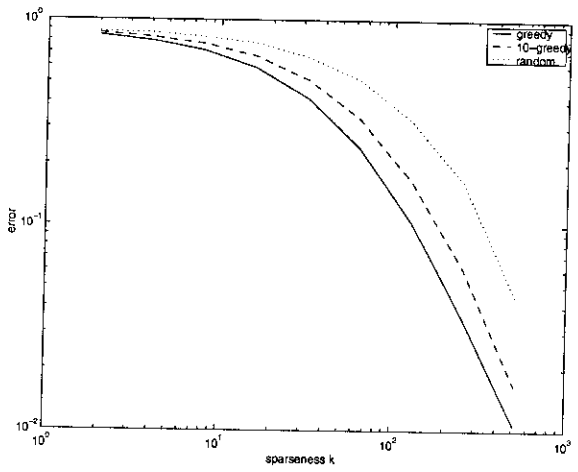

Figure 1. Approximation performance of different algorithms: ΔR as a function of sparseness k.

5. Summary

In this paper, we considered a few procedures for obtaining sparse approximations in a Gaussian process. We obtained upper bounds on approximation error that are of the form $O(1/k)$ where k is the number of non-zeros in the dual variable. This rate is in a similar form as that of matching pursuit approximation. However, unlike matching pursuit where a full greedy algorithm is necessary to guarantee this rate of approximation, randomization can be used in sparse Gaussian process algorithms to achieve this rate. This randomization can significantly reduce computation.

Some experimental results of sparse Gaussian processes have also been included. We demonstrated a trade-off between randomization and greedy search which is consistent with our theory.

Finally, there are still many open issues. For example, it may be possible to refine our analysis to quantitatively characterize the approximation gain of greedy search over randomization. The dependency of λ in our bounds can also be improved. Although we have shown that this λ dependency is necessary if we want to specify the approximation bounds in terms of $R(0)$, M and λ, it is possible to introduce other quantities that can better characterize the behavior of a Gaussian process than λ does. For example, ob-

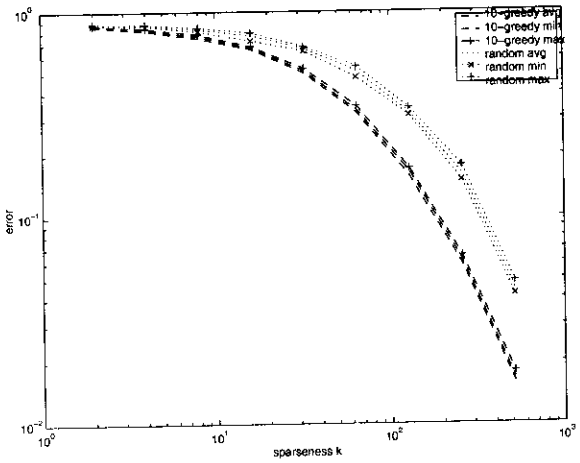

Figure 2. Variance in randomized algorithms: ΔR as a function of sparseness k.

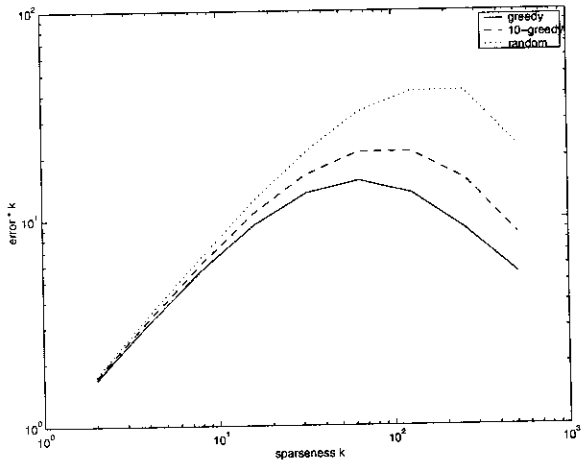

Figure 3. $k\Delta R$ as a function of sparseness k.

serve that this factor is introduced into our analysis through Lemma 2.2, therefore we can simply modify this lemma by replacing λ with another quantity (for example, $\|w\|_2$) that may remain bounded even when $\lambda \to 0$. We leave the details to another report.

References

Barron, A. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory, 39*, 930–945.

Cesa-Bianchi, N., Long, P., & Warmuth, M. K. (1996). Worst-case quadratic loss bounds for prediction using linear functions and gradient descent. *EEE Transactions on Neural Networks, 7*, 604–619.

Chen, S. S., Donoho, D. L., & Saunders, M. A. (1999). Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput., 20*, 33–61.

Cristianini, N., & Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods.* Cambridge University Press.

Csató, L., & Opper, M. (2000). Sparse representation for Gaussian process models. *NIPS 00.* to appear.

Girosi, F. (1998). An equivalence between sparse approximation and support vector machines. *Neural Computation, 10*, 1455–1480.

Golub, G., & Van Loan, C. (1996). *Matrix computations.* Baltimore, MD: Johns Hopkins University Press. Third edition.

Hoerl, A. E., & Kennard, R. W. (1970). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics, 12*, 55–67.

Jones, L. K. (1992). A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *Ann. Statist., 20*, 608–613.

Mallat, S., & Zhang, Z. (1993). Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing, 41*, 3397–3415.

Natarajan, B. K. (1995). Sparse approximate solutions to linear systems. *SIAM J. Comput., 24*, 227–234.

Smola, A. J., & Bartlett, P. (2000). Sparse greedy gaussian process regression. *NIPS 00.* to appear.

Vapnik, V. (1998). *Statistical learning theory.* New York: John Wiley & Sons.