Approximating a Gram Matrix for Improved Kernel-Based Learning

(Extended Abstract)

Petros Drineas[1] and Michael W. Mahoney[2]

[1] Department of Computer Science, Rensselaer Polytechnic Institute,
Troy, New York 12180
drinep@cs.rpi.edu
[2] Department of Mathematics, Yale University,
New Haven, CT 06520
mahoney@cs.yale.edu

Abstract. A problem for many kernel-based methods is that the amount of computation required to find the solution scales as $O(n^3)$, where n is the number of training examples. We develop and analyze an algorithm to compute an easily-interpretable low-rank approximation to an $n \times n$ Gram matrix G such that computations of interest may be performed more rapidly. The approximation is of the form $\tilde{G}_k = CW_k^+C^T$, where C is a matrix consisting of a small number c of columns of G and W_k is the best rank-k approximation to W, the matrix formed by the intersection between those c columns of G and the corresponding c rows of G. An important aspect of the algorithm is the probability distribution used to randomly sample the columns; we will use a judiciously-chosen and data-dependent nonuniform probability distribution. Let $\|\cdot\|_2$ and $\|\cdot\|_F$ denote the spectral norm and the Frobenius norm, respectively, of a matrix, and let G_k be the best rank-k approximation to G. We prove that by choosing $O(k/\epsilon^4)$ columns

$$\left\|G - CW_k^+C^T\right\|_\xi \leq \|G - G_k\|_\xi + \epsilon \sum_{i=1}^{n} G_{ii}^2,$$

both in expectation and with high probability, for both $\xi = 2, F$, and for all $k : 0 \leq k \leq \mathrm{rank}(W)$. This approximation can be computed using $O(n)$ additional space and time, after making two passes over the data from external storage.

1 Introduction

1.1 Background

Given a collection $\mathcal{X}$ of data points, which are often but not necessarily elements of $\mathbb{R}^m$, techniques such as linear Support Vector Machines (SVMs), Gaussian Processes (GPs), Principle Component Analysis (PCA), and the related Singular Value Decomposition (SVD), identify and extract structure from $\mathcal{X}$ by

P. Auer and R. Meir (Eds.): COLT 2005, LNAI 3559, pp. 323–337, 2005.
© Springer-Verlag Berlin Heidelberg 2005

computing linear functions, i.e., functions in the form of dot products, of the data. For example, in PCA the subspace spanned by the first k eigenvectors is used to give a k dimensional model of the data with minimal residual; thus, it provides a low-dimensional representation of the data. Such spectral analysis has a rich theoretical foundation and has numerous practical applications.

In many cases, however, there is nonlinear structure in the data (or the data, e.g. text, may not support the basic linear operations of addition and scalar multiplication). In these cases, kernel-based learning methods have proved to be quite useful [7, 27]. Kernel-based learning methods are a class of statistical learning algorithms, the best known examples of which are SVMs [7]. In this approach, data items are mapped into high-dimensional spaces, where information about their mutual positions (in the form of inner products) is used for constructing classification, regression, or clustering rules. Kernel-based algorithms exploit the information encoded in the inner product between all pairs of data items and are successful in part because there is often an efficient method to compute inner products between very complex or even infinite dimensional vectors. Thus, kernel-based algorithms provide a way to deal with nonlinear structure by reducing nonlinear algorithms to algorithms that are linear in some feature space $\mathcal{F}$ that is nonlinearly related to the original input space.

More precisely, assume that the data consists of vectors $X^{(1)}, \ldots, X^{(n)} \in \mathcal{X} \subset \mathbb{R}^m$ and let $X \in \mathbb{R}^{m \times n}$ be the matrix whose i-th column is $X^{(i)}$. In kernel-based methods, a set of features is chosen that define a space $\mathcal{F}$, where it is hoped relevant structure will be revealed, the data $\mathcal{X}$ are then mapped to the feature space $\mathcal{F}$ using a mapping $\Phi : \mathcal{X} \to \mathcal{F}$, and then classification, regression, or clustering is performed in $\mathcal{F}$ using traditional methods such as linear SVMs, GPs, or PCA. If $\mathcal{F}$ is chosen to be a dot product space and if one defines the kernel matrix, also known as the Gram matrix, $G \in \mathbb{R}^{n \times n}$ as $G_{ij} = k(x_i, x_j) = (\Phi(x_i), \Phi(x_j))$, then any algorithm whose operations can be expressed in the input space in terms of dot products can be generalized to an algorithm which operates in the feature space by substituting a kernel function for the inner product. In practice, this means presenting the Gram matrix G in place of the input covariance matrix $X^T X$. Relatedly, using the kernel k instead of a dot product in the input space corresponds to mapping the data set into a (usually) high-dimensional dot product space $\mathcal{F}$ by a (usually nonlinear) mapping $\Phi : \mathbb{R}^m \to \mathcal{F}$, and taking dot products there, i.e., $k(x_i, x_j) = (\Phi(x_i), \Phi(x_j))$. Note that for the commonly-used Mercer kernels, G is a symmetric positive semidefinite (SPSD) matrix.

The generality of this framework should be emphasized. For example, there has been much work recently on dimensionality reduction for nonlinear manifolds in high-dimensional spaces. See, e.g., Isomap, local linear embedding, and graph Laplacian eigenmap [29, 26, 4] as well as Hessian eigenmaps and semidefinite embedding [9, 30]. These methods first induce a local neighborhood structure on the data and then use this local structure to find a global embedding of the manifold in a lower dimensional space. The manner in which these different algorithms use the local information to construct the global embedding is quite

different, but in [22] they are interpreted as kernel PCA applied to specially constructed Gram matrices.

This "kernel trick" has been quite successful for extracting nonlinear structure in large data sets when the features are chosen such that the structure in the data is more manifest in the feature space than in the original space. Although in many cases the features are chosen such that the Gram matrix is sparse, in which case sparse matrix computation methods may be used, in other applications the Gram matrix is dense, but is well approximated by a low-rank matrix. In this case, calculations of interest (such as the matrix inversion needed in GP prediction, the quadratic programming problem for SVMs, and the computation of the eigendecomposition of the Gram matrix) will still generally take space which is $O(n^2)$ and time which is $O(n^3)$. This is prohibitive if n, the number of data points, is large. Recent work in the learning theory community has focused on taking advantage of this low-rank structure in order to perform learning tasks of interest more efficiently. For example, in [2], several randomized methods are used in order to speed up kernel PCA. These methods have provable guarantees on the quality of their approximation and may be viewed as replacing the kernel function k by a "randomized kernel" which behaves like k in expectation. Relatedly, in [33], uniform sampling without replacement is used to choose a small set of basis training points, from which an approximation to the Gram matrix is constructed. Although this algorithm does not come with provable performance guarantees, it may be viewed as a special case of our main algorithm, and it was shown empirically to perform well on two data sets for approximate GP classification and regression. It was also interpreted in terms of the Nyström method from integral equation theory; this method has also been applied recently in the learning theory community to approximate the solution of spectral partitioning for image and video segmentation [20] and to extend the eigenfunctions of a data-dependent kernel to new data points [5, 23]. Related work taking advantage of low-rank structure includes [28, 19, 32, 6, 24, 31, 3].

1.2 Summary of Main Result

In this paper, we develop and analyze an algorithm to compute an easily-interpretable low-rank approximation to an $n \times n$ Gram matrix G. Our main result, the Main Approximation algorithm of Section 3.2, is an algorithm that, when given as input a SPSD matrix $G \in \mathbb{R}^{n \times n}$, computes a low-rank approximation to G of the form $\tilde{G}_k = CW_k^+C^T$, where $C \in \mathbb{R}^{n \times c}$ is a matrix formed by randomly choosing a small number c of columns (and thus rows) of G and $W_k \in \mathbb{R}^{c \times c}$ is the best rank-k approximation to W, the matrix formed by the intersection between those c columns of G and the corresponding c rows of G. The columns are chosen in c independent random trials (and thus with replacement) according to a judiciously-chosen and data-dependent nonuniform probability distribution. The nonuniform probability distribution will be carefully chosen and will be important for the provable bounds we obtain. Let $\|\cdot\|_2$ and $\|\cdot\|_F$ denote the spectral norm and the Frobenius norm, respectively, and let G_k be the

best rank-k approximation to G. Our main result, presented in a more precise form in Theorem 1, is that under appropriate assumptions:

$$\left\|G - CW_k^+ C^T\right\|_\xi \leq \|G - G_k\|_\xi + \epsilon \sum_{i=1}^{n} G_{ii}^2, \tag{1}$$

in both expectation and with high probability, for both $\xi = 2, F$, for all $k : 0 \leq k \leq \text{rank}(W)$. This approximation can be computed in $O(n)$ space and time after two passes over the data from external storage.

1.3 Technical Report

In the interests of space, several sections have not been included in this extended abstract. For more details and discussion related to the results presented here, see the associated technical report [18]. In particular, [18] contains a discussion of the relationship between our work, recent work on Nyström-based kernel methods [33, 31, 20], and the low-rank approximation algorithm of Frieze, Kannan, and Vempala [21, 14].

2 Review of Relevant Linear Algebra

For the review of the linear algebra used in this paper, see the associated technical report [18]. Recent work in the theory of randomized algorithms has focused on matrix problems [21, 10, 1, 2, 11, 12, 13, 14, 15, 16, 17, 25]. In particular, our previous work has applied random sampling methods to the approximation of several common matrix computations such as matrix multiplication [13], the computation of low-rank approximations to a matrix [14], the computation of the CUR matrix decomposition [15], and approximating the feasibility of linear programs [16, 17]. For the review of two results from this random sampling methodology that will be used in this paper, see the associated technical report [18].

3 Approximating a Gram Matrix

Consider a set of n points in $\mathbb{R}^m$, denoted by $X^{(1)}, \ldots, X^{(n)}$, and let X be the $m \times n$ matrix whose i-th column is $X^{(i)}$. These points may be either the original data or the data after they have been mapped into the feature space. Then, define the $n \times n$ Gram matrix G as $G = X^T X$. Thus, G is a SPSD matrix and $G_{ij} = (X^{(i)}, X^{(j)})$ is the dot product between the data vectors $X^{(i)}$ and $X^{(j)}$. If G is dense but has good linear structure, i.e., is well-approximated by a low-rank matrix, then a computation of a easily-computable and easily-interpretable low-rank approximation to G, with provable error bounds, is of interest. In this section, two algorithms are presented that compute such an approximation to a Gram matrix G.

3.1 A Preliminary Nyström-Based Algorithm

In [33], a method to approximate G was proposed that, in our notation, chooses c columns from G uniformly at random and without replacement, and constructs an approximation of the form $\tilde{G} = CW^{-1}C^T$, where the $n \times c$ matrix C consists of the c chosen columns and W is a matrix consisting of the intersection of those c columns with the corresponding c rows. Analysis of this algorithm and issues such as the existence of the inverse were not addressed in [33], but computational experiments were performed and the procedure was shown to work well empirically on two data sets [33]. This method has been referred to as the Nyström method [33, 31, 20] since it has an interpretation in terms of the Nyström technique for solving linear integral equations [8]. See [18] for a full discussion.

In Algorithm 1, the PRELIMINARY APPROXIMATION algorithm is presented. It is an algorithm that takes as input an $n \times n$ Gram matrix G and returns as output an approximate decomposition of the form $\tilde{G} = CW^{+}C^T$, where C and W are as in [33], and where W^+ is the Moore-Penrose generalized inverse of W. The c columns are chosen uniformly at random and with replacement. Thus, the PRELIMINARY APPROXIMATION algorithm is quite similar to the algorithm of [33], except that we sample with replacement and that we do not assume the existence of W^{-1}. Rather than analyzing this algorithm (which could be done by combining the analysis of Section 3.3 with the uniform sampling bounds of [13]), we present and analyze a more general form of it, for which we can obtain improved bounds, in Section 3.2. Note, however, that if the uniform sampling probabilities are nearly optimal, in the sense that $1/n \geq \beta G_{ii}^2 / \sum_{i=1}^n G_{ii}^2$ for some positive $\beta \leq 1$ and for every $i = 1, \ldots, n$, then bounds similar to those in Theorem 1 will be obtained for this algorithm, with a small β-dependent loss in accuracy; see [13, 18].

Data : $n \times n$ Gram matrix G and $c \leq n$.

Result : $n \times n$ matrix $\tilde{G}$.

- Pick c columns of G in i.i.d. trials, uniformly at random with replacement; let $\mathcal{I}$ be the set of indices of the sampled columns.
- Let C be the $n \times c$ matrix containing the sampled columns.
- Let W be the $c \times c$ submatrix of G whose entries are $G_{ij}, i \in \mathcal{I}, j \in \mathcal{I}$.
- Return $\tilde{G} = CW^{+}C^T$.

Algorithm 1: The PRELIMINARY APPROXIMATION algorithm

3.2 The Main Algorithm and the Main Theorem

In [13, 14, 15, 16, 17], we showed the importance of sampling columns and/or rows of a matrix with carefully chosen nonuniform probability distributions in order to obtain provable error bounds for a variety of common matrix operations. In Algorithm 2, the MAIN APPROXIMATION algorithm is presented. It is a generalization of the PRELIMINARY APPROXIMATION algorithm that allows the column sample to be formed using arbitrary sampling probabilities. The MAIN

Approximation algorithm takes as input an $n \times n$ Gram matrix G, a probability distribution $\{p_i\}_{i=1}^n$, a number $c \leq n$ of columns to choose, and a rank parameter $k \leq c$. It returns as output an approximate decomposition of the form $\tilde{G}_k = CW_k^+C^T$, where C is an $n \times c$ matrix consisting of the chosen columns of G, each rescaled in an appropriate manner, and where W_k is a $c \times c$ matrix that is the best rank-k approximation to the matrix W, which is a matrix whose elements consist of those elements in G in the intersection of the chosen columns and the corresponding rows, each rescaled in an appropriate manner.

Data : $n \times n$ Gram matrix G, $\{p_i\}_{i=1}^n$ such that $\sum_{i=1}^n p_i = 1$, $c \leq n$, and $k \leq c$.

Result : $n \times n$ matrix $\tilde{G}$.

- Pick c columns of G in i.i.d. trials, with replacement and with respect to the probabilities $\{p_i\}_{i=1}^n$; let $\mathcal{I}$ be the set of indices of the sampled columns.
- Scale each sampled column (whose index is $i \in \mathcal{I}$) by dividing its elements by $\sqrt{cp_i}$; let C be the $n \times c$ matrix containing the sampled columns rescaled in this manner.
- Let W be the $c \times c$ submatrix of G whose entries are $G_{ij}/(c\sqrt{p_ip_j}), i \in \mathcal{I}, j \in \mathcal{I}$.
- Compute W_k, the best rank-k approximation to W.
- Return $\tilde{G}_k = CW_k^+C^T$.

Algorithm 2: The Main Approximation algorithm

To implement this algorithm, two passes over the Gram matrix G from external storage and $O(n)$, i.e. sublinear in $O(n^2)$, additional space and time are sufficient (assuming that the sampling probabilities of the form, e.g., $p_i = G_{ii}^2/\sum_{i=1}^n G_{ii}^2$ or $p_i = \left|G^{(i)}\right|^2 / \|G\|_F^2$ or $p_i = 1/n$ are used). Thus, this algorithm is efficient within the framework of the Pass-Efficient model; see [13] for more details. Note that if the sampling probabilities of the form $p_i = G_{ii}^2/\sum_{i=1}^n G_{ii}^2$ are used, as in Theorem 1 below, then one may store the $m \times n$ data matrix X in external storage, in which case only those elements of G that are used in the approximation need to be computed.

In the simplest application of this algorithm, one could choose $k = c$, in which case $W_k = W$, and the decomposition is of the form $\tilde{G} = CW^+C^T$, where W^+ is the exact Moore-Penrose generalized inverse of the matrix W. In certain cases, however, computing the generalized inverse may be problematic since, e.g., it may amplify noise present in the low singular values. Note that, as a function of increasing k, the Frobenius norm bound of Theorem 2 of [18] is not necessarily optimal for $k = \text{rank}(C)$. Also, although the bounds of Theorem 1 for the spectral norm for $k \leq \text{rank}(W)$ are in general worse than those for $k = \text{rank}(W)$, the former are of interest since our algorithms hold for any input Gram matrix and we make no assumptions about a model for the noise in the data.

The sampling matrix formalism of [13] is used in the proofs of Theorem 1 in Section 3.3, and thus we introduce it here. Let us define the sampling matrix $S \in \mathbb{R}^{n \times c}$ to be the zero-one matrix where $S_{ij} = 1$ if the i-th column of A is chosen

in the j-th independent random trial and $S_{ij} = 0$ otherwise. Similarly, define the rescaling matrix $D \in \mathbb{R}^{c \times c}$ to be the diagonal matrix with $D_{tt} = 1/\sqrt{cp_{i_t}}$. Then, the $n \times c$ matrix

$$C = GSD$$

consists of the chosen columns of G, each of which has been rescaled by $1/\sqrt{cp_{i_t}}$, where i_t is the label of the column chosen in the t-th independent trial. Similarly, the $c \times c$ matrix

$$W = (SD)^T GSD = DS^T GSD$$

consists of the intersection between the chosen columns and the corresponding rows, each element of which has been rescaled by with $1/c\sqrt{p_{i_t} p_{j_t}}$. (This can also be viewed as forming W by sampling a number c of rows of C and rescaling. Note, however, that in this case the columns of A and the rows of C are sampled using the same probabilities.) In Algorithm 3, the MAIN APPROXIMATION is restated using this sampling matrix formalism. It should be clear that Algorithm 3 and Algorithm 2 yield identical results.

Data : $n \times n$ Gram matrix G, $\{p_i\}_{i=1}^n$ such that $\sum_{i=1}^n p_i = 1$, $c \leq n$, and $k \leq c$.
Result : $n \times n$ matrix $\tilde{G}$.
- Define the $(n \times c)$ matrix $S = \mathbf{0}_{n \times c}$;
- Define the $(c \times c)$ matrix $D = \mathbf{0}_{c \times c}$;
- **for** $t = 1, \ldots, c$ **do**
  - Pick $i_t \in [n]$, where $\mathbf{Pr}(i_t = i) = p_i$;
  - $D_{tt} = (cp_{i_t})^{-1/2}$;
  - $S_{i_t t} = 1$;

end
- Let $C = GSD$ and $W = DS^T GSD$.
- Compute W_k, the best rank-k approximation to W.
- Return $\tilde{G}_k = CW_k^+ C^T$.

Algorithm 3: The MAIN APPROXIMATION algorithm, restated

Before stating our main theorem, we wish to emphasize the structural simplicity of our main result. If, e.g., we choose $k = c$, then our main algorithm provides a decomposition of the form $\tilde{G} = CW^+ C^T$:

$$\begin{pmatrix} \\ G \\ \\ \end{pmatrix} \approx \begin{pmatrix} \\ \tilde{G} \\ \\ \end{pmatrix} = \begin{pmatrix} \\ C \\ \\ \end{pmatrix} \left(W\right)^+ \begin{pmatrix} \; C^T \; \end{pmatrix}. \tag{2}$$

Up to rescaling, the MAIN APPROXIMATION algorithm returns an approximation $\tilde{G}$ which is created from two submatrices of G, namely C and W. In the uniform sampling case, $p_i = 1/n$, the diagonal elements of the rescaling matrix D are all n/c, and these all cancel out of the expression. In the nonuniform sampling case, C is a rescaled version of the columns of G and W is a rescaled version of the intersection of those columns with the corresponding rows. Alternatively, one

can view C as consisting of the actual columns of G, without rescaling, and W as consisting of the intersection of those columns with the corresponding rows, again without rescaling, in the following manner. Let $\hat{C} = GS$, let $\hat{W} = S^T GS$, and let

$$\hat{W}^+ = \hat{W}^+_{D^2,D^{-2}} = D\left(D\hat{W}D\right)^+ D \tag{3}$$

be the $\{D^2, D^{-2}\}$-weighted-$\{1,2\}$-generalized inverse of $\hat{W}$. Then, $G \approx \tilde{G} = \hat{C}\hat{W}^+\hat{C}^T$.

The following theorem states our main result regarding the Main Approximation algorithm. Its proof may be found in Section 3.3.

Theorem 1. *Suppose G is an $n \times n$ SPSD matrix, let $k \leq c$ be a rank parameter, and let $\tilde{G}_k = CW_k^+C^T$ be constructed from the* Main Approximation *algorithm of Algorithm 2 by sampling c columns of G with probabilities $\{p_i\}_{i=1}^n$ such that*

$$p_i = G_{ii}^2 / \sum_{i=1}^n G_{ii}^2. \tag{4}$$

Let $r = rank(W)$ and let G_k be the best rank-k approximation to G. In addition, let $\epsilon > 0$ and $\eta = 1 + \sqrt{8\log(1/\delta)}$. If $c \geq 64k/\epsilon^4$, then

$$\mathbf{E}\left[\left\|G - \tilde{G}_k\right\|_F\right] \leq \|G - G_k\|_F + \epsilon \sum_{i=1}^n G_{ii}^2 \tag{5}$$

and if $c \geq 64k\eta^2/\epsilon^4$ then with probability at least $1-\delta$

$$\left\|G - \tilde{G}_k\right\|_F \leq \|G - G_k\|_F + \epsilon \sum_{i=1}^n G_{ii}^2. \tag{6}$$

In addition, if $c \geq 4/\epsilon^2$ then

$$\mathbf{E}\left[\left\|G - \tilde{G}_k\right\|_2\right] \leq \|G - G_k\|_2 + \epsilon \sum_{i=1}^n G_{ii}^2 \tag{7}$$

and if $c \geq 4\eta^2/\epsilon^2$ then with probability at least $1-\delta$

$$\left\|G - \tilde{G}_k\right\|_2 \leq \|G - G_k\|_2 + \epsilon \sum_{i=1}^n G_{ii}^2. \tag{8}$$

Several things should be noted about this result. First, if $k \geq r = \mathrm{rank}(W)$ then $W_k = W$, and an application of Theorem 2 of [18] leads to bounds of the form $\left\|G - \tilde{G}_r\right\|_2 \leq \epsilon \sum_{i=1}^n G_{ii}^2$, in expectation and with high probability. Second, the sampling probabilities used in Thoerem 1 may be written as $p_i = \left|X^{(i)}\right|^2 / \|X\|_F^2$, which only depend on dot products from the data matrix X. This is useful if X consists of the data after it has been mapped to

the feature space $\mathcal{F}$. Finally, if the sampling probabilities were of the form $p_i = \left|G^{(i)}\right|^2 / \|G\|_F^2$ then they would preferentially choose data points that are more informative (in the sense of being longer) and/or more representative of the data (in the sense that they tend to be more well correlated with more data points). Instead the probabilities (4) ignore the correlations. As discussed in [18], this leads to somewhat worse error bounds. To the best of our knowledge, it is not known how to sample with respect to correlations while respecting the SPSD property and obtaining provably good bounds with improved error bounds. This is of interest since in many applications it is likely that the data are approximately normalized by the way the data are generated, and it is the correlations that are of interest. Intuitively, this difficulty arises since it is difficult to identify structure in a matrix to ensure the SPSD property, unless, e.g., the matrix is diagonally dominant or given in the form $X^T X$. As will be seen in Section 3.3, the proof of Theorem 1 depends crucially on the decomposition of G as $G = X^T X$.

3.3 Proof of Theorem 1

Since $G = X^T X$ it follows that both the left and the right singular vectors of G are equal to the right singular vectors of X and that the singular values of G are the squares of the singular values of X. More formally, let the SVD of X be $X = U\Sigma V^T$. Then,

$$G = V\Sigma^2 V^T = XUU^T X^T. \tag{9}$$

Now, let us consider $C_X = XSD \in \mathbb{R}^{m\times c}$, i.e., the column sampled and rescaled version of X, and let the SVD of C_X be $C_X = \hat{U}\hat{\Sigma}\hat{V}^T$. Thus, in particular, $\hat{U}$ contains the left singular vectors of C_X. We do not specify the dimensions of $\hat{U}$ (and in particular how many columns $\hat{U}$ has) since we do not know the rank of C_X. Let $\hat{U}_k$ be the $m \times k$ matrix whose columns consist of the singular vectors of C_X corresponding to the top k singular values. Instead of exactly computing the left singular vectors U of X, we can approximate them by $\hat{U}_k$, computed from a column sample of X, and use this to compute an approximation $\tilde{G}$ to G.

We first establish the following lemma, which provides a bound on $\left\|G - \tilde{G}_k\right\|_\xi$ for $\xi = 2, F$.

Lemma 1. *If* $\tilde{G}_k = CW_k^+ C^T$ *then*

$$\left\|G - \tilde{G}_k\right\|_F = \left\|X^T X - X^T \hat{U}_k \hat{U}_k X\right\|_F \tag{10}$$

$$\left\|G - \tilde{G}_k\right\|_2 = \left\|X - \hat{U}_k \hat{U}_k^T X\right\|_2^2. \tag{11}$$

Proof: Recall that $C = GSD$ and $W = (SD)^T GSD = C_X^T C_X$. Thus, $W = \hat{V}\hat{\Sigma}^2\hat{V}$ and $W_k = \hat{V}\hat{\Sigma}_k^2\hat{V}^T$, where $\hat{\Sigma}_k$ is the diagonal matrix with the top k singular values of C_X on the diagonal and the remainder set to 0. Then since

$C_X = XSD = \hat{U}\hat{\Sigma}\hat{V}^T$ and $W_k^+ = \hat{V}\hat{\Sigma}_k^{-2}\hat{V}^T$

$$\tilde{G}_k = GSD\left(W_k\right)^+ \left(GSD\right)^T \tag{12}$$

$$= X^T\hat{U}\hat{\Sigma}\hat{V}^T\left(\hat{V}\hat{\Sigma}_k^2\hat{V}^T\right)^+\hat{V}\hat{\Sigma}\hat{U}^TX \tag{13}$$

$$= X^T\hat{U}_k\hat{U}_k^TX, \tag{14}$$

where $\hat{U}_k\hat{U}_k^T$ is a projection onto the space spanned by the top k singular vectors of W. (10) then follows immediately, and (11) follows since

$$X^TX - X^T\hat{U}_k\hat{U}_k^TX = \left(X - \hat{U}_k\hat{U}_k^TX\right)^T\left(X - \hat{U}_k\hat{U}_k^TX\right)$$

and since $\|\Omega\|_2^2 = \left\|\Omega^T\Omega\right\|_2$ for any matrix Ω. ◇

By combining (11) with Theorem 2 of [18], we see that

$$\begin{aligned}\left\|G - \tilde{G}_k\right\|_2 &\leq \|X - X_k\|_2^2 + 2\left\|XX^T - C_XC_X^T\right\|_2 \\ &\leq \|G - G_k\|_2 + 2\left\|XX^T - C_XC_X^T\right\|_2.\end{aligned}$$

Since the sampling probabilities (4) are of the form $p_i = \left|X^{(i)}\right|^2/\|X\|_F^2$, this may be combined with Theorem 1 of [18], from which, by choosing c appropriately, the spectral norm bounds (7) and (8) of Theorem 1 follow.

To establish the Frobenius norm bounds, define $E = XX^TXX^T - C_XC_X^TC_XC_X^T$. Then, we have that:

$$\left\|G - \tilde{G}_k\right\|_F^2 = \left\|X^TX\right\|_F^2 - 2\left\|XX^T\hat{U}_k\right\|_F^2 + \left\|\hat{U}_k^TXX^T\hat{U}_k\right\|_F^2 \tag{15}$$

$$\leq \left\|X^TX\right\|_F^2 - 2\left(\sum_{t=1}^k \sigma_t^4(C_X) - \sqrt{k}\|E\|_F\right) + \sum_{t=1}^k \sigma_t^4(C_X) + \sqrt{k}\|E\|_F \tag{16}$$

$$= \left\|X^TX\right\|_F^2 - \sum_{t=1}^k \sigma_t^4(C_X) + 3\sqrt{k}\|E\|_F \tag{17}$$

$$\leq \left\|X^TX\right\|_F^2 - \sum_{t=1}^k \sigma_t^2(X^TX) + 4\sqrt{k}\|E\|_F, \tag{18}$$

where (15) follows by Lemmas 1 and 2, (16) follows by Lemmas 3 and 4, and (18) follows by Lemma 5. Since

$$\left\|X^TX\right\|_F^2 - \sum_{t=1}^k \sigma_t^2(X^TX) = \|G\|_F^2 - \sum_{t=1}^k \sigma_t^2(G) = \|G - G_k\|_F^2,$$

it follows that

$$\left\|G - \tilde{G}_k\right\|_F^2 \leq \|G - G_k\|_F^2 + 4\sqrt{k}\left\|XX^TXX^T - C_XC_X^TC_XC_X^T\right\|_F. \tag{19}$$

Since the sampling probabilities (4) are of the form $p_i = \left|X^{(i)}\right|^2 / \|X\|_F^2$, this may be combined with Lemma 6 and Theorem 1 of [18]. Since $(\alpha^2+\beta^2)^{1/2} \leq \alpha+\beta$ for $\alpha, \beta \geq 0$, by using Jensen's inequality, and by choosing c appropriately, the Frobenius norm bounds (5) and (6) of Theorem 1 follow.

The next four lemmas are used to bound the right hand side of (10).

Lemma 2. *For every $k : 0 \leq k \leq rank(W)$ we have that:*

$$\left\|X^T X - X^T \hat{U}_k \hat{U}_k^T X\right\|_F^2 = \left\|X^T X\right\|_F^2 - 2\left\|X X^T \hat{U}_k\right\|_F^2 + \left\|\hat{U}_k^T X X^T \hat{U}_k\right\|_F^2$$

Proof: Define $Y = X - \hat{U}_k \hat{U}_k^T X$. Then,

$$\begin{aligned}\left\|X^T X - X^T \hat{U}_k \hat{U}_k^T X\right\|_F^2 &= \left\|Y^T Y\right\|_F^2 \\ &= \mathbf{Tr}\left(Y^T Y Y^T Y\right) \\ &= \left\|X^T X\right\|_F^2 - 2\mathbf{Tr}\left(X X^T \hat{U}_k \hat{U}_k^T X X^T\right) \\ &\quad + \mathbf{Tr}\left(\hat{U}_k^T X X^T \hat{U}_k \hat{U}_k^T X X^T \hat{U}_k\right),\end{aligned}$$

where the last line follows by multiplying out terms and since the trace is symmetric under cyclic permutations. The lemma follows since $\|\Omega\|_F^2 = \mathbf{Tr}\left(\Omega\Omega^T\right)$ for any matrix Ω. ◇

Lemma 3. *For every $k : 0 \leq k \leq rank(W)$ we have that:*

$$\left|\left\|X X^T \hat{U}_k\right\|_F^2 - \sum_{t=1}^{k} \sigma_t^4(C_X)\right| \leq \sqrt{k}\left\|X X^T X X^T - C_X C_X^T C_X C_X^T\right\|_F$$

Proof: Since $\sigma_t(C_X C_X^T) = \sigma_t^2(C_X)$ and since $\hat{U}$ is a matrix consisting of the singular vectors of $C_X = XSD$, we have that

$$\begin{aligned}\left|\left\|X X^T \hat{U}_k\right\|_F^2 - \sum_{t=1}^{k}\sigma_t^4(C_X)\right| &= \left|\sum_{t=1}^{k}\left|X X^T \hat{U}^{(t)}\right|^2 - \sum_{t=1}^{k}\left|C_X C_X^T \hat{U}^{(t)}\right|^2\right| \\ &= \left|\sum_{t=1}^{k} \hat{U}^{(t)^T}\left(X X^T X X^T - C_X C_X^T C_X C_X^T\right)\hat{U}^{(t)}\right| \\ &\leq \sqrt{k}\left(\sum_{t=1}^{k}\left(\hat{U}^{(t)^T}\left(X X^T X X^T - C_X C_X^T C_X C_X^T\right)\hat{U}^{(t)}\right)^2\right)^{1/2},\end{aligned}$$

where the last line follows from the Cauchy-Schwartz inequality. The lemma then follows. ◇

Lemma 4. *For every $k : 0 \leq k \leq rank(W)$ we have that:*

$$\left\|\hat{U}_k^T X X^T \hat{U}_k\right\|_F^2 - \sum_{t=1}^{k}\sigma_t^4(C_X) \leq \sqrt{k}\left\|X X^T X X^T - C_X C_X^T C_X C_X^T\right\|_F$$

Proof: Recall that if a matrix U has orthonormal columns then $\left\|U^T\Omega\right\|_F \leq \|\Omega\|_F$ for any matrix Ω. Thus, we have that

$$\left\|\hat{U}_k^T XX^T\hat{U}_k\right\|_F^2 - \sum_{t=1}^k \sigma_t^4(C_X) \leq \left\|XX^T\hat{U}_k\right\|_F^2 - \sum_{t=1}^k \sigma_t^4(C_X) \leq \left|\left\|XX^T\hat{U}_k\right\|_F^2 - \sum_{t=1}^k \sigma_t^4(C_X)\right|$$

The remainder of the proof follows that of Lemma 3. ◇

Lemma 5. *For every $k : 0 \leq k \leq rank(W)$ we have that:*

$$\left|\sum_{t=1}^k \sigma_t^4(C_X) - \sigma_t^2(X^TX)\right| \leq \sqrt{k}\left\|XX^TXX^T - C_XC_X^TC_XC_X^T\right\|_F$$

Proof:

$$\begin{aligned}\left|\sum_{t=1}^k \sigma_t^4(C_X) - \sigma_t^2(X^TX)\right| &\leq \sqrt{k}\left(\sum_{t=1}^k \left(\sigma_t^4(C_X) - \sigma_t^2(X^TX)\right)^2\right)^{1/2} \\ &= \sqrt{k}\left(\sum_{t=1}^k \left(\sigma_t(C_XC_X^TC_XC_X^T) - \sigma_t(XX^TXX^T)\right)^2\right)^{1/2} \\ &\leq \sqrt{k}\left\|XX^TXX^T - C_XC_X^TC_XC_X^T\right\|_F,\end{aligned}$$

where the first inequality follows from the Cauchy-Schwartz inequality and the second inequality follows from matrix perturbation theory. ◇

The following is a result of the BasicMatrixMultiplication algorithm that is not found in [13], but that will be useful for bounding the additional error in (19). We state this result for a general $m \times n$ matrix A.

Lemma 6. *Suppose $A \in \mathbb{R}^{m\times n}$, $c \in \mathbb{Z}^+$ such that $1 \leq c \leq n$, and $\{p_i\}_{i=1}^n$ are such that $p_k = \left|A^{(k)}\right|^2 / \|A\|_F^2$. Construct C with the* BasicMatrixMultiplication *algorithm of [13]. Then,*

$$\mathbf{E}\left[\left\|AA^TAA^T - CC^TCC^T\right\|_F\right] \leq \frac{2}{\sqrt{c}}\|A\|_F^4. \tag{20}$$

Furthermore, let $\delta \in (0,1)$ and $\eta = 1 + \sqrt{8\log(1/\delta)}$. Then, with probability at least $1-\delta$,

$$\left\|AA^TAA^T - CC^TCC^T\right\|_F \leq \frac{2\eta}{\sqrt{c}}\|A\|_F^4. \tag{21}$$

Proof: First note that:

$$\begin{aligned}AA^TAA^T - CC^TCC^T &= AA^TAA^T - AA^TCC^T + AA^TCC^T - CC^TCC^T \\ &= AA^T\left(AA^T - CC^T\right) + \left(AA^T - CC^T\right)CC^T.\end{aligned}$$

Thus, by submultiplicitivity and subadditivity we have that for $\xi = 2, F$:

$$\left\|AA^TAA^T - CC^TCC^T\right\|_F \leq \|A\|_F^2 \left\|AA^T - CC^T\right\|_F + \left\|AA^T - CC^T\right\|_F \|C\|_F^2 .$$

The lemma follows since $\|C\|_F^2 = \|A\|_F^2$ when $p_k = \left|A^{(k)}\right|^2 / \|A\|_F^2$, and by applying Theorem 1 of [18]. ◇

4 Conclusion

We have presented and analyzed an algorithm that provides an approximate decomposition of an $n \times n$ Gram matrix G which is of the form $G \approx \tilde{G}_k = CW_k^+C^T$ and which has provable error bounds of the form (1). A crucial feature of this algorithm is the probability distribution used to randomly sample columns. We conclude with two open problems related to the choice of this distribution.

First, it would be desirable to choose the probabilities in Theorem 1 to be $p_i = \left|G^{(i)}\right|^2 / \|G\|_F^2$ and to establish bounds of the form (1) in which the scale of the additional error was $\|G\|_F = \left\|X^TX\right\|_F$ rather than $\sum_{i=1}^n G_{ii}^2 = \|X\|_F^2$. This would entail extracting linear structure while simultaneously respecting the SPSD property and obtaining improved scale of error. This would likely be a corollary of a CUR decomposition [15] for a general $m \times n$ matrix A with error bounds of the form found in [15] and in which $U = W_k^+$, where W is now the matrix consisting of the intersection of the chosen columns and (in general different) rows; see [18]. This would simplify considerably the form of U found in [15] and would lead to improved interpretability. Second, we should also note that if capturing coarse statistics over the data is not of interest, but instead one is interested in other properties of the data, e.g., identifying outliers, then probabilities that depend on the data in some other manner, e.g., inversely with respect to their lengths squared, may be appropriate. We do not have provable bounds in this case.

Acknowledgments. We would like to thank Ravi Kannan for many fruitful discussions and the Institute for Pure and Applied Mathematics at UCLA for its generous hospitality.

References

1. D. Achlioptas and F. McSherry. Fast computation of low rank matrix approximations. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 611–618, 2001.
2. D. Achlioptas, F. McSherry, and B. Schölkopf. Sampling techniques for kernel methods. In *Annual Advances in Neural Information Processing Systems 14: Proceedings of the 2001 Conference*, pages 335–342, 2002.

3. Y. Azar, A. Fiat, A.R. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. In *Proceedings of the 33rd Annual ACM Symposium on Theory of Computing*, pages 619–626, 2001.
4. M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.
5. Y. Bengio, J.F. Paiement, P. Vincent, O. Delalleau, N. Le Roux, and M. Ouimet. Out-of-sample extensions for LLE, Isomap, MDS, eigenmaps, and spectral clustering. In *Annual Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference*, pages 177–184, 2004.
6. C.J.C. Burges. Simplified support vector decision rules. In *Proceedings of the 13th International Conference on Machine Learning*, pages 71–77, 1996.
7. N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, Cambridge, 2000.
8. L.M. Delves and J.L. Mohamed. *Computational Methods for Integral Equations*. Cambridge University Press, Cambridge, 1985.
9. D.L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proc. Natl. Acad. Sci. USA*, 100(10):5591–5596, 2003.
10. P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 291–299, 1999.
11. P. Drineas and R. Kannan. Fast Monte-Carlo algorithms for approximate matrix multiplication. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 452–459, 2001.
12. P. Drineas and R. Kannan. Pass efficient algorithms for approximating large matrices. In *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 223–232, 2003.
13. P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. Technical Report YALEU/DCS/TR-1269, Yale University Department of Computer Science, New Haven, CT, February 2004.
14. P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. Technical Report YALEU/DCS/TR-1270, Yale University Department of Computer Science, New Haven, CT, February 2004.
15. P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition. Technical Report YALEU/DCS/TR-1271, Yale University Department of Computer Science, New Haven, CT, February 2004.
16. P. Drineas, R. Kannan, and M.W. Mahoney. Sampling sub-problems of heterogeneous Max-Cut problems and approximation algorithms. Technical Report YALEU/DCS/TR-1283, Yale University Department of Computer Science, New Haven, CT, April 2004.
17. P. Drineas, R. Kannan, and M.W. Mahoney. Sampling sub-problems of heterogeneous Max-Cut problems and approximation algorithms. In *Proceedings of the 22nd Annual International Symposium on Theoretical Aspects of Computer Science*, pages 57–68, 2005.
18. P. Drineas and M.W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. Technical Report 1319, Yale University Department of Computer Science, New Haven, CT, April 2005.

19. S. Fine and K. Scheinberg. Efficient SVM training using low-rank kernel representations. *Journal of Machine Learning Research*, 2:243–264, 2001.
20. C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):214–225, 2004.
21. A. Frieze, R. Kannan, and S. Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, pages 370–378, 1998.
22. J. Ham, D.D. Lee, S. Mika, and B. Schölkopf. A kernel view of the dimensionality reduction of manifolds. Technical Report TR-110, Max Planck Institute for Biological Cybernetics, July 2003.
23. S. Lafon. *Diffusion Maps and Geometric Harmonics*. PhD thesis, Yale University, 2004.
24. E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In *Proceedings of the 1997 IEEE Workshop on Neural Networks for Signal Processing VII*, pages 276–285, 1997.
25. L. Rademacher, S. Vempala, and G. Wang. Matrix approximation and projective clustering via iterative sampling. *manuscript*.
26. S.T. Roweis and L.K. Saul. Nonlinear dimensionality reduction by local linear embedding. *Science*, 290:2323–2326, 2000.
27. B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.
28. A.J. Smola and B. Schölkopf. Sparse greedy matrix approximation for machine learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 911–918, 2000.
29. J.B. Tenenbaum, V. de Silva, and J.C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.
30. K.Q. Weinberger, F. Sha, and L.K. Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the 21st International Conference on Machine Learning*, pages 839–846, 2004.
31. C.K.I. Williams, C.E. Rasmussen, A. Schwaighofer, and V. Tresp. Observations on the Nyström method for Gaussian process prediction. Technical report, University of Edinburgh, 2002.
32. C.K.I. Williams and M. Seeger. The effect of the input density distribution on kernel-based classifiers. In *Proceedings of the 17th International Conference on Machine Learning*, pages 1159–1166, 2000.
33. C.K.I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Annual Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, pages 682–688, 2001.