SIAM J. MATRIX ANAL. APPL.
Vol. 30, No. 2, pp. 844–881
© 2008 Society for Industrial and Applied Mathematics

RELATIVE-ERROR *CUR* MATRIX DECOMPOSITIONS*

PETROS DRINEAS†, MICHAEL W. MAHONEY‡, AND S. MUTHUKRISHNAN§

Abstract. Many data analysis applications deal with large matrices and involve approximating the matrix using a small number of "components." Typically, these components are linear combinations of the rows and columns of the matrix, and are thus difficult to interpret in terms of the original features of the input data. In this paper, we propose and study matrix approximations that are explicitly expressed in terms of a small number of columns and/or rows of the data matrix, and thereby more amenable to interpretation in terms of the original data. Our main algorithmic results are two randomized algorithms which take as input an $m \times n$ matrix A and a rank parameter k. In our first algorithm, C is chosen, and we let $A' = CC^+A$, where C^+ is the Moore–Penrose generalized inverse of C. In our second algorithm C, U, R are chosen, and we let $A' = CUR$. (C and R are matrices that consist of actual columns and rows, respectively, of A, and U is a generalized inverse of their intersection.) For each algorithm, we show that with probability at least $1 - \delta$, $\|A - A'\|_F \leq (1 + \epsilon)\|A - A_k\|_F$, where A_k is the "best" rank-k approximation provided by truncating the SVD of A, and where $\|X\|_F$ is the Frobenius norm of the matrix X. The number of columns of C and rows of R is a low-degree polynomial in k, $1/\epsilon$, and $\log(1/\delta)$. Both the Numerical Linear Algebra community and the Theoretical Computer Science community have studied variants of these matrix decompositions over the last ten years. However, our two algorithms are the first polynomial time algorithms for such low-rank matrix approximations that come with relative-error guarantees; previously, in some cases, it was not even known whether such matrix decompositions exist. Both of our algorithms are simple and they take time of the order needed to approximately compute the top k singular vectors of A. The technical crux of our analysis is a novel, intuitive sampling method we introduce in this paper called "subspace sampling." In subspace sampling, the sampling probabilities depend on the Euclidean norms of the rows of the top singular vectors. This allows us to obtain provable relative-error guarantees by deconvoluting "subspace" information and "size-of-A" information in the input matrix. This technique is likely to be useful for other matrix approximation and data analysis problems.

Key words. *CUR* matrix decomposition, random sampling algorithms, data analysis, approximate least squares

AMS subject classification. 68W20

DOI. 10.1137/07070471X

1. Introduction.

Large $m \times n$ matrices are common in applications since the data often consist of m objects, each of which is described by n features. Examples of object–feature pairs include: documents and words contained in those documents; genomes and environmental conditions under which gene responses are measured; stocks and their associated temporal resolution; hyperspectral images and frequency resolution; and web groups and individual users. In each of these application areas, practitioners spend vast amounts of time analyzing the data in order to understand, interpret, and ultimately use this data for some application-specific task.

*Received by the editors October 8, 2007; accepted for publication (in revised form) by J. G. Nagy April 14, 2008; published electronically September 17, 2008. A preliminary version of this paper appeared in manuscript and technical report format as "Polynomial Time Algorithm for Column-Row-Based Relative-Error Low-Rank Matrix Approximation" [27, 28]. Preliminary versions of parts of this paper have also appeared as conference proceedings [29, 30, 31].

http://www.siam.org/journals/simax/30-2/70471.html

†Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180 (drinep@cs.rpi.edu).

‡Yahoo! Research, Sunnyvale, CA 94089 (mahoney@yahoo-inc.com). Part of this work was performed while at the Department of Mathematics, Yale University, New Haven, CT 06520.

§Google, Inc., New York, NY 10011 (muthu@google.com). Part of this work was performed while at the Department of Computer Science, Rutgers University, New Brunswick, NJ 08854.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Say that A is the $m \times n$ data matrix. In many cases, an important step in data analysis is to construct a compressed representation of A that may be easier to analyze and interpret. The most common such representation is obtained by truncating the SVD at some number $k \ll \min\{m, n\}$ terms, in large part because this provides the "best" rank-k approximation to A when measured with respect to any unitarily invariant matrix norm. Unfortunately, the basis vectors (the so-called eigencolumns and eigenrows) provided by this approximation (and with respect to which every column and row of the original data matrix is expressed) are notoriously difficult to interpret in terms of the underlying data and processes generating that data. For example, the vector $[(1/2)\ \text{age} - (1/\sqrt{2})\ \text{height} + (1/2)\ \text{income}]$, being one of the significant uncorrelated "factors" from a dataset of people's features, is not particularly informative. It would be highly preferable to have a low-rank approximation that is nearly as good as that provided by the SVD but that is expressed in terms of a small number of *actual columns* and/or *actual rows* of a matrix, rather than linear combinations of those columns and rows.

The main contribution of this paper is to provide such decompositions. In particular, we provide what we call a relative-error *CUR matrix decomposition*: given an $m \times n$ matrix A, we decompose it as a product of three matrices, C, U, and R, where C consists of a small number of actual columns of A, R consists of a small number of actual rows of A, and U is a small carefully constructed matrix that guarantees that the product CUR is "close" to A. In fact, CUR will be nearly as good as the best low-rank approximation to A that is traditionally used and that is obtained by truncating the SVD. Hence, the columns of A that are included in C, as well as the rows of A that are included in R, can be used in place of the eigencolumns and eigenrows, with the added benefit of improved interpretability in terms of the original data.

Before describing applications of our main results in the next subsection, we would like to emphasize that two research communities, the Numerical Linear Algebra (NLA) community and the Theoretical Computer Science (TCS) community, have provided significant practical and theoretical motivation for studying variants of these matrix decompositions over the last ten years. In section 3, we provide a detailed treatment of relevant prior work in both the NLA and the TCS literature. The two algorithms presented in this paper are the first polynomial time algorithms for such low-rank matrix approximations that come with relative-error guarantees; previously, in some cases, it was not even known whether such matrix decompositions exist.

1.1. Applications. As an example of this preference for having the data matrix expressed in terms of a small number of actual columns and rows of the original matrix, as opposed to a small number of eigencolumns and eigenrows, consider recent data analysis work in DNA microarray and DNA Single Nucleotide Polymorphism (SNP) analysis [44, 47, 52]. DNA SNP data are often modeled as an $m \times n$ matrix A, where m is the number of individuals in the study, n is the number of SNPs being analyzed, and A_{ij} is an encoding of the jth SNP value for the ith individual. Similarly, for DNA microarray data, m is the number of genes under consideration, n is the number of arrays or environmental conditions, and A_{ij} is the absolute or relative expression level of the ith gene in the jth environmental condition. Biologists typically have an understanding of a single gene that they fail to have about a linear combination of 6000 genes (and also similarly for SNPs, individuals, and arrays); thus, recent work in genetics on DNA microarray and DNA SNP data has focused on heuristics to extract *actual* genes, environmental conditions, individuals, and SNPs from the eigengenes, eigenconditions, eigenpeople, and eigenSNPs computed from the original

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

data matrices [44, 47].[1] Our *CUR* matrix decomposition is a direct formulation of this problem: determine a small number of actual SNPs that serve as a basis with which to express the remaining SNPs, and a small number of individuals to serve as a basis with which to express the remaining individuals. In fact, motivated in part by this, we have successfully applied a variant of the *CUR* matrix decomposition presented in this paper to intra- and interpopulation genotype reconstruction from tagging SNPs in DNA SNP data from a geographically diverse set of populations [52]. In addition, we have applied a different variant of our *CUR* matrix decomposition to hyperspectrally resolved medical imaging data [48]. In this application, a column corresponds to an image at a single physical frequency and a row corresponds to a single spectrally resolved pixel, and we have shown that data reconstruction and classification tasks can be performed with little loss in quality even after substantial data compression [48].

A quite different motivation for low-rank matrix approximations expressed in terms of a small number of columns and/or rows of the original matrix is to decompose efficiently large low-rank matrices that possess additional structure such as sparsity or nonnegativity. This often arises in the analysis of, e.g., large term-document matrices [58, 59, 8]. Another motivation comes from statistical learning theory, where the data need not even be elements in a vector space, and thus expressing the Gram matrix in terms of a small number of actual data points is of interest [64, 63, 24, 25]. This procedure has been shown empirically to perform well for approximate Gaussian process classification and regression [64], to approximate the solution of spectral partitioning for image and video segmentation [32], and to extend the eigenfunctions of a data-dependent kernel to new data points [7, 45]. Yet another motivation is provided by integral equation applications [40, 39, 38], where large coefficient matrices arise that have blocks corresponding to regions where the kernel is smooth and that are thus well-approximated by low-rank matrices. In these applications, partial SVD algorithms can be expensive, and a description in terms of actual columns and/or rows is of interest [39, 38]. A final motivation for studying matrix decompositions of this form is to obtain low-rank matrix approximations to extremely large matrices where a computation of the SVD is too expensive [33, 34, 21, 22, 23].

1.2. Our main results. Our main algorithmic results have to do with efficiently computing low-rank matrix approximations that are explicitly expressed in terms of a small number of columns and/or rows of the input matrix. We start with the following definition.

DEFINITION 1. *Let A be an $m \times n$ matrix. For any given C, an $m \times c$ matrix whose columns consist of c columns of the matrix A, the $m \times n$ matrix $A' = CX$ is a* column-based matrix approximation *to A, or* CX matrix decomposition*, for any $c \times n$ matrix X.*

Several things should be noted about this definition. First, we will be interested in $c \ll n$ in our applications. For example, depending on the application, c could be constant, independent of n, logarithmic in the size of n, or simply a large constant factor less than n. Second, a CX matrix decomposition expresses each of the columns

[1] For example, in their review article "Vector algebra in the analysis of genome-wide expression data" [44], which appeared in *Genome Biology*, Kuruvilla, Park, and Schreiber describe many uses of the vectors provided by the SVD and PCA in DNA microarray analysis. The three biologists then conclude by stating that: "While very efficient basis vectors, the vectors themselves are completely artificial and do not correspond to actual (DNA expression) profiles. ... Thus, it would be interesting to try to find basis vectors for all experiment vectors, using actual experiment vectors and not artificial bases that offer little insight." That is, they explicitly state that they would like decompositions of the form we provide in this paper!

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

of A in terms of a linear combination of "dictionary elements" or "basis columns," each of which is an actual column of A. Thus, a CX matrix decomposition provides a low-rank approximation to the original matrix, although one with structural properties that are quite different than those provided by the SVD. Third, given a set of columns C, the approximation $A' = P_C A = CC^+A$ (where $P_C A$ is the projection of A onto the subspace spanned by the columns of C and C^+ is the Moore–Penrose generalized inverse of C, as defined in section 2) clearly satisfies the requirements of Definition 1. Indeed, this is the "best" such approximation to A, in the sense that $\|A - C(C^+A)\|_F = \min_{X \in \mathbb{R}^{c \times n}} \|A - CX\|_F$.

Our first main result is the following.

THEOREM 1. *Given a matrix $A \in \mathbb{R}^{m \times n}$ and an integer $k \ll \min\{m, n\}$, there exist randomized algorithms such that either exactly $c = O(k^2 \log(1/\delta)/\epsilon^2)$ columns of A are chosen to construct C, or $c = O(k \log k \log(1/\delta)/\epsilon^2)$ columns are chosen in expectation to construct C, such that with probability at least $1 - \delta$,*

$$\min_{X \in \mathbb{R}^{c \times n}} \|A - CX\|_F = \left\|A - CC^+A\right\|_F \leq (1 + \epsilon) \|A - A_k\|_F . \tag{1}$$

Here, C is a matrix consisting of the chosen columns of A, CC^+A is the projection of A on the subspace spanned by the chosen columns, and A_k is the best rank-k approximation to A. Both algorithms run in time $O(SVD(A, k))$, which is the time required to compute the best rank-k approximation to the matrix A [37].

Note that we use $c > k$ and have an ϵ error, which allows us to take advantage of linear algebraic structure in order to obtain an efficient algorithm. In general, this would not be the case if, given an $m \times n$ matrix A, we had specified a parameter k and asked for the "best" subset of k columns, where "best" is measured, e.g., by maximizing the Frobenius norm captured by projecting onto those columns or by maximizing the volume of the parallelepiped defined by those columns. Also, it is not clear a priori that C with properties above even exists; see the discussion in sections 3.2 and 3.3. Finally, our result does not include any reference to regularization or conditioning, as is common in certain application domains; a discussion of similar work on related problems in numerical linear algebra may be found in section 3.1.

Our second main result extends the previous result to *CUR* matrix decompositions.

DEFINITION 2. *Let A be an $m \times n$ matrix. For any given C, an $m \times c$ matrix whose columns consist of c columns of the matrix A, and R, an $r \times n$ matrix whose rows consist of r rows of the matrix A, the $m \times n$ matrix $A' = CUR$ is a* column-row-based matrix approximation *to A, or* CUR matrix decomposition*, for any $c \times r$ matrix U.*

Several things should be noted about this definition. First, a *CUR* matrix decomposition is a CX matrix decomposition, but one with a very special structure; i.e., every column of A can be expressed in terms of the basis provided by C using only the information contained in a small number of rows of A and a low-dimensional encoding matrix. Second, in terms of its singular value structure, U must clearly contain "inverse-of-A" information. For the *CUR* decomposition described in this paper, U will be a generalized inverse of the intersection between C and R. More precisely, if $C = AS_C D_C$ and $R = D_R S_R^T A$, then $U = (D_R S_R^T A S_C D_C)^+$. (See section 2 for a review of linear algebra and notation, such as that for S_C, D_C, S_R, and D_R.) Third, the combined size of C, U, and R is $O(mc + rn + cr)$, which is an improvement over A's size of $O(mn)$ when $c, r \ll n, m$. Finally, note the structural simplicity of a *CUR*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

matrix decomposition:

$$\underbrace{\begin{pmatrix} \\ A \\ \\ \end{pmatrix}}_{m\times n} \approx \underbrace{\begin{pmatrix} \\ C \\ \\ \end{pmatrix}}_{m\times c} \underbrace{\begin{pmatrix} \\ U \\ \\ \end{pmatrix}}_{c\times r} \underbrace{\begin{pmatrix} \\ R \\ \\ \end{pmatrix}}_{r\times n}. \tag{2}$$

Our main result for *CUR* matrix decomposition is the following.

THEOREM 2. *Given a matrix $A \in \mathbb{R}^{m\times n}$ and an integer $k \ll \min\{m,n\}$, there exist randomized algorithms such that exactly $c = O(k^2 \log(1/\delta)/\epsilon^2)$ columns of A are chosen to construct C, and then exactly $r = O(c^2 \log(1/\delta)/\epsilon^2)$ rows of A are chosen to construct R, or $c = O(k \log k \log(1/\delta)/\epsilon^2)$ columns of A in expectation are chosen to construct C, and then $r = O(c \log c \log(1/\delta)/\epsilon^2)$ rows of A in expectation are chosen to construct R, such that with probability at least $1-\delta$,*

$$\|A - CUR\|_F \leq (1+\epsilon)\|A - A_k\|_F. \tag{3}$$

Here, the matrix U is a weighted Moore–Penrose inverse of the intersection between C and R, and A_k is the best rank-k approximation to A. Both algorithms run in time $O(SVD(A,k))$, which is the time required to compute the best rank-k approximation to the matrix A [37].

1.3. Summary of main technical result. The key technical insight that leads to the relative-error guarantees is that the columns are selected by a novel sampling procedure that we call "subspace sampling." Rather than sample columns from A with a probability distribution that depends on the Euclidean norms of the columns of A (which gives provable additive-error bounds [21, 22, 23]), in "subspace sampling" we randomly sample columns of A with a probability distribution that depends on the Euclidean norms of the rows of the top k right singular vectors of A. This allows us to capture entirely a certain subspace of interest. Let $V_{A,k}$ be the $n \times k$ matrix whose columns consist of the top k right singular vectors of A. The "subspace sampling" probabilities $p_i, i \in [n]$ will satisfy

$$p_i \geq \frac{\beta \left|(V_{A,k})_{(i)}\right|_2^2}{k} \qquad \forall i \in [n], \tag{4}$$

for some $\beta \in (0,1]$, where $(V_{A,k})_{(i)}$ is the ith *row* of $V_{A,k}$. That is, we will sample based on the norms of the rows (not the columns) of the truncated matrix of singular vectors. Note that $\sum_{j=1}^{n} |(V_{A,k})_{(j)}|_2^2 = k$ and that $\sum_{i\in[n]} p_i = 1$. To construct sampling probabilities satisfying Condition (4), it is sufficient to spend $O(SVD(A,k))$ time to compute (exactly or approximately, in which case $\beta = 1$ or $\beta < 1$, respectively) the top k right singular vectors of A. Sampling probabilities of this form will allow us to deconvolute subspace information and "size-of-A" information in the input matrix A, which in turn will allow us to obtain the relative-error guarantees we desire. Note that we have used this method previously [29], but in that case the sampling probabilities contained other terms that complicated their interpretation.

We will use these "subspace sampling" probabilities in our main technical result, which is a random sampling algorithm for approximating the following generalized

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

version of the standard ℓ_2 regression problem. Our main column/row-based approximation algorithmic results will follow from this result. Given as input a matrix $A \in \mathbb{R}^{m \times n}$ that has rank no more than k and a matrix of target vectors $B \in \mathbb{R}^{m \times p}$ compute

$$\mathcal{Z} = \min_{X \in \mathbb{R}^{n \times p}} \|B - AX\|_F . \tag{5}$$

That is, fit every column of the matrix B to the basis provided by the columns of the rank-k matrix A. Also of interest is the computation of

$$X_{opt} = A^+ B. \tag{6}$$

The main technical result of this paper is a simple sampling algorithm that represents the matrices A and B by a small number of rows so that this generalized ℓ_2 regression problem can be solved to accuracy $1 \pm \epsilon$ for any $\epsilon > 0$.

More precisely, we present and analyze an algorithm (Algorithm 3 of section 6) that constructs and solves an induced subproblem of the generalized ℓ_2 regression problem of (5) and (6). Let $DS^T A$ be the $r \times n$ matrix consisting of the sampled and appropriately rescaled rows of the original matrix A, and let $DS^T B$ be the $r \times p$ matrix consisting of the sampled and appropriately rescaled rows of B. Then consider the problem

$$\tilde{\mathcal{Z}} = \min_{X \in \mathbb{R}^{n \times p}} \left\| DS^T B - DS^T AX \right\|_F . \tag{7}$$

The "smallest" matrix $\tilde{X}_{opt} \in \mathbb{R}^{n \times p}$ among those that achieve the minimum value $\tilde{\mathcal{Z}}$ in this sampled ℓ_2 regression problem is

$$\tilde{X}_{opt} = \left(DS^T A \right)^+ DS^T B. \tag{8}$$

Since we will sample a number of rows $r \ll m$ of the original problem, we will compute (8), and thus (7), exactly. Our main theorem, Theorem 5, states that under appropriate assumptions on the original problem and on the sampling probabilities, the computed quantities $\tilde{\mathcal{Z}}$ and $\tilde{X}_{opt}$ will provide very accurate relative-error approximations to the exact solution $\mathcal{Z}$ and the optimal vector X_{opt}. Rows will be sampled with one of two random sampling procedures. In one case, exactly $r = O(k^2/\epsilon^2)$ rows are chosen, and in the other case, $r = O(k \log k/\epsilon^2)$ rows in expectation are chosen. In either case, the most expensive part of the computation involves the computation of the Euclidean norms of the rows of the right singular vectors of A which are used in the sampling probabilities.

1.4. Outline of the remainder of the paper. In the next two sections, we provide a review of relevant linear algebra, and we discuss related work. Then, in sections 4 and 5, we present in detail our main algorithmic results. In section 4, we describe our main column-based matrix approximation algorithm, and in section 5, we describe our main column-row-based matrix approximation algorithm. Then, in section 6, we present an approximation algorithm for generalized ℓ_2 regression. This is our main technical result, and from it our two main algorithmic results will follow. Finally, in section 7 we present an empirical evaluation of our algorithms, and in section 8 we present a brief conclusion. We devote Appendix A to two prior algorithms for approximate matrix multiplication. These two algorithms select columns and rows in a complementary manner, and they are essential in the proof of our main results.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

2. Review of linear algebra. In this section, we provide a review of linear algebra that will be useful throughout the paper; for more details, see [50, 43, 60, 37, 9, 6]. We also review a sampling matrix formalism that will be convenient in our discussion [21].

Let $[n]$ denote the set $\{1, 2, \ldots, n\}$. For any matrix $A \in \mathbb{R}^{m\times n}$, let $A_{(i)}, i \in [m]$ denote the ith row of A as a row vector, and let $A^{(j)}, j \in [n]$ denote the jth column of A as a column vector. In addition, let $\|A\|_F^2 = \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}^2$ denote the square of its Frobenius norm, and let $\|A\|_2 = \sup_{x\in\mathbb{R}^n,\ x\neq 0} |Ax|_2 / |x|_2$ denote its spectral norm. These norms satisfy $\|A\|_2 \leq \|A\|_F \leq \sqrt{\min\{m,n\}}\,\|A\|_2$ for any matrix A, and also $\|AB\|_F \leq \|A\|_F\,\|B\|_2$ for any matrices A and B.

If $A \in \mathbb{R}^{m\times n}$, then there exist orthogonal matrices $U = [u^1 u^2 \ldots u^m] \in \mathbb{R}^{m\times m}$ and $V = [v^1 v^2 \ldots v^n] \in \mathbb{R}^{n\times n}$ such that $U^T A V = \Sigma = \mathbf{diag}(\sigma_1, \ldots, \sigma_\xi)$, where $\Sigma \in \mathbb{R}^{m\times n}$, $\xi = \min\{m, n\}$ and $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_\xi \geq 0$. Equivalently, $A = U\Sigma V^T$. The three matrices U, V, and Σ constitute the SVD of A. The σ_i are the singular values of A, the vectors u^i and v^i are the ith left and the ith right singular vectors of A, respectively, and the condition number of A is $\kappa(A) = \sigma_{\max}(A)/\sigma_{\min}(A)$. If $k \leq r = \mathrm{rank}(A)$, then the SVD of A may be written as

(9)
$$A = U_A \Sigma_A V_A^T = \begin{bmatrix} U_k & U_k^\perp \end{bmatrix} \begin{bmatrix} \Sigma_k & \mathbf{0} \\ \mathbf{0} & \Sigma_{k,\perp} \end{bmatrix} \begin{bmatrix} V_k^T \\ V_k^{\perp T} \end{bmatrix} = U_k \Sigma_k V_k^T + U_k^\perp \Sigma_{k,\perp} V_k^{\perp T}.$$

Here, Σ_k is the $k \times k$ diagonal matrix containing the top k singular values of A, and $\Sigma_{k,\perp}$ is the $(r-k)\times(r-k)$ diagonal matrix containing the bottom $r-k$ nonzero singular values of A. Also, V_k^T is the $k \times n$ matrix whose rows are the top k right singular vectors of A, $V_k^{\perp T}$ is the $(r-k) \times n$ matrix whose rows are the bottom $r-k$ right singular vectors of A, and U_k and $U_k^\perp$ are defined similarly. If we define $A_k = U_k \Sigma_k V_k^T$, then the distance (as measured by both $\|\cdot\|_2$ and $\|\cdot\|_F$) between A and any rank k approximation to A is minimized by A_k. We will denote by $O(SVD(A,k))$ the time required to compute the best rank-k approximation to the matrix A [37]. Finally, for any orthogonal matrix $U \in \mathbb{R}^{m\times c}$, let $U^\perp \in \mathbb{R}^{m\times(m-c)}$ denote an orthogonal matrix whose columns are an orthonormal basis spanning the subspace of $\mathbb{R}^m$ that is orthogonal to the column space of U.

Given a matrix $A \in \mathbb{R}^{m\times n}$, the unweighted Moore–Penrose generalized inverse of A, denoted by A^+, is the unique $n \times m$ matrix that satisfies the four Moore–Penrose conditions [50, 6]. In terms of the SVD this generalized inverse may be written as $A^+ = V_A \Sigma_A^{-1} U_A^T$ (where the square diagonal $\mathrm{rank}(A)\times\mathrm{rank}(A)$ matrix Σ_A, as in (9), is invertible by construction). If, in addition, $D_1 \in \mathbb{R}^{m\times m}$ and $D_2 \in \mathbb{R}^{n\times n}$ are diagonal matrices with positive entries along the diagonal, then the $\{D_1, D_2\}$-Moore–Penrose generalized inverse of A, denoted by $A^+_{(D_1,D_2)}$, is a generalization of the Moore–Penrose inverse that can be expressed in terms of the unweighted generalized inverse of A as $A^+_{(D_1,D_2)} = D_2^{-1/2}(D_1^{1/2} A D_2^{-1/2})^+ D_1^{1/2}$. Also, in terms of the generalized inverse, the projection onto the column space of any matrix A may be written as $P_A = AA^+$.

Since our main algorithms will involve sampling columns and/or rows from input matrices (using one of two related random sampling procedures described in Appendix A), we conclude this subsection with a brief review of a sampling matrix formalism that was introduced in [21] and with respect to which our sampling matrix operations may be conveniently expressed. First, assume that c' (= c *exactly*) columns of A are chosen in c i.i.d. trials by randomly sampling according to

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

a probability distribution $\{p_i\}_{i=1}^n$ with the EXACTLY(c) algorithm (described in detail in Appendix A), and assume that the i_tth column of A is chosen in the tth (for $t = 1, \ldots, c$) independent random trial. Then, define the sampling matrix $S \in \mathbb{R}^{n \times c}$ to be the zero-one matrix where $S_{i_t t} = 1$ and $S_{ij} = 0$ otherwise, and define the diagonal rescaling matrix $D \in \mathbb{R}^{c \times c}$ to be the diagonal matrix with $D_{tt} = 1/\sqrt{cp_{i_t}}$, where p_{i_t} is the probability of choosing the i_tth column. Alternatively, assume that c' ($\leq c$ *in expectation*) columns of A are chosen with the EXPECTED(c) algorithm (also described in detail in Appendix A) by including the ith column of A in C with probability $\tilde{p}_i = \min\{1, cp_i\}$. Then, define the sampling matrix $S \in \mathbb{R}^{n \times n}$ to be the zero-one matrix where $S_{ii} = 1$ if the ith column is chosen and $S_{ij} = 0$ otherwise, and define the rescaling matrix $D \in \mathbb{R}^{n \times c'}$ to be the matrix with $D_{ij} = 1/\sqrt{c\tilde{p}_j}$ if $i - 1$ of the previous columns have been chosen and $D_{ij} = 0$ otherwise. Clearly, in both of these cases, $C = ASD$ is an $m \times c'$ matrix consisting of sampled and rescaled copies of the columns of A, and $R = (SD)^T A = DS^T A$ is a $c' \times n$ matrix consisting of sampled and rescaled copies of the rows of A. In certain cases, we will subscript S and D with C or R (e.g., $C = AS_C D_C$ and $R = D_R S_R^T A$) to make explicit that the corresponding sampling and rescaling matrices are operating on the columns or rows, respectively, of A.

3. Relationship with previous related work. In this section, we discuss the relationship between our results and related work in numerical linear algebra and theoretical computer science.

3.1. Related work in numerical linear algebra. Within the numerical linear algebra community, several groups have studied matrix decompositions with similar structural, if not algorithmic, properties to the CX and *CUR* matrix decompositions we have defined. Much of this work is related to the QR decomposition, originally used extensively in pivoted form by Golub [36, 11].

Stewart and collaborators were interested in computing sparse low-rank approximations to large sparse term-document matrices [58, 59, 8]. He developed the quasi-Gram–Schmidt method. This method is a variant of the QR decomposition which, when given as input an $m \times n$ matrix A and a rank parameter k, returns an $m \times k$ matrix C consisting of k columns of A whose span approximates the column space of A and also a nonsingular upper-triangular $k \times k$ matrix T_C that orthogonalizes these columns (but it does not explicitly compute the nonsparse orthogonal matrix $Q_C = CT_C^{-1}$). This provides a matrix decomposition of the form $A \approx CX$. By applying this method to A to obtain C and to A^T to obtain an $k \times n$ matrix R consisting of k rows of A, one can show that $A \approx CUR$, where the matrix U is computed to minimize $\|A - CUR\|_F^2$. Although provable approximation guarantees of the form we present were not provided, backward error analysis was performed, and the method was shown to perform well empirically [58, 59, 8].

Goreinov, Tyrtyshnikov, and Zamarashkin [39, 38, 61] were interested in applications such as scattering, in which large coefficient matrices have blocks that can be easily approximated by low-rank matrices. They show that if the matrix A is approximated by a rank-k matrix to within an accuracy ϵ, then *there exists* a choice of k columns and k rows, i.e., C and R, and a low-dimensional $k \times k$ matrix U constructed from the elements of C and R, such that $A \approx CUR$ in the sense that $\|A - CUR\|_2 \leq \epsilon f(m, n, k)$, where $f(m, n, k) = 1 + 2\sqrt{km} + 2\sqrt{kn}$. In [39], the choice for these matrices is related to the problem of determining the minimum singular value σ_k of $k \times k$ submatrices of $n \times k$ orthogonal matrices. In addition, in [38]

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

the choice for C and R is interpreted in terms of the maximum volume concept from interpolation theory, in the sense that columns and rows should be chosen such that their intersection W defines a parallelepiped of maximum volume among all $k \times k$ submatrices of A; in [61] an empirically effective deterministic algorithm is presented which ensures that U is well-conditioned.

Gu and Eisenstat, in their seminal paper [40], describe a strong rank-revealing QR factorization that deterministically selects exactly k columns from an $m \times n$ matrix A. The algorithms of [40] are efficient, in that their running time is $O(mn^2)$ (assuming that $m \geq n$), which is essentially the time required to compute the SVD of A. In addition, Gu and Eisenstat prove that if the $m \times k$ matrix C contains the k selected columns (without any rescaling), then $\sigma_{\min}(C) \geq \sigma_k(A)/f(k,n)$, where $f(k,n) = O(\sqrt{k(n-k)})$. Thus, the columns of C span a parallelepiped whose volume (equivalently, the product of the singular values of C) is "large." Currently, we do not know how to convert this property into a statement similar to that of Theorem 1, although perhaps this can be accomplished by relaxing the number of columns selected by the algorithms of [40] to $O(poly(k, 1/\epsilon))$. For related work prior to Gu and Eisenstat, see Chan and Hansen [12, 13].

Finally, very recently, Martinsson, Rokhlin, and Tygert [49] proposed another related method to efficiently compute an approximation to the best rank-k approximation of an $m \times n$ matrix A. The heart of their algorithm is a random projection method, which projects A to a small number, say ℓ, of random vectors; the entries of these random vectors are i.i.d. Gaussians of zero mean and unit variance. The general form of their bounds is quite complicated, but by setting, e.g., $\ell = k + 20$, they construct a rank-k approximation A' to A such that

$$\|A - A'\|_2 \leq 10\sqrt{(k+20)\,m}\,\|A - A_k\|_2 \tag{10}$$

holds with probability at least $1 - 10^{-17}$. In addition, the authors extend their algorithm to compute the so-called interpolative decomposition of a matrix A. This decomposition is explicitly expressed in terms of a small number of columns of A, and is a more restrictive version of our CX matrix decomposition. More specifically, it additionally requires that every entry of X is bounded in absolute value by a small constant (e.g., two). Thus, their algorithm computes an interpolative approximation $A' = CX$ to A, where C has only $\ell = k + 20$ columns—as opposed to the $O(k \log k)$ columns that are necessary in our work—and satisfies the bound of (10). Notice that their work provides bounds for the spectral norm, whereas our work focuses only on the Frobenius norm. However, their bounds are much weaker than our relative error bounds, since $\sqrt{m(k+20)}\,\|A - A_k\|_2$ might in general be larger even than $\|A\|_F$.

3.2. Related work in theoretical computer science. Within the theory of algorithms community, much research has followed the seminal work of Frieze, Kannan, and Vempala [33, 34]. Their work may be viewed, in our parlance, as sampling columns from a matrix A to form a matrix C such that $\|A - CX\|_F \leq \|A - A_k\|_F + \epsilon\,\|A\|_F$. The matrix C has $poly(k, 1/\epsilon, 1/\delta)$ columns and is constructed after making only two passes over A using $O(m+n)$ work space. Under similar resource constraints, a series of papers have followed [33, 34] in the past seven years [19, 22, 55], improving the dependency of c on $k, 1/\epsilon$, and $1/\delta$, and analyzing the spectral as well as the Frobenius norm, yielding bounds of the form

$$\|A - CX\|_\xi \leq \|A - A_k\|_\xi + \epsilon\,\|A\|_F \tag{11}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

for $\xi = 2, F$, and thus providing additive-error guarantees for column-based low-rank matrix approximations.

Additive-error approximation algorithms for *CUR* matrix decompositions have also been analyzed by Drineas, Kannan, and Mahoney [20, 21, 22, 23, 24, 25]. In particular, in [23], they compute an approximation to an $m \times n$ matrix A by sampling c columns and r rows from A to form $m \times c$ and $r \times n$ matrices C and R, respectively. From C and R, a $c \times r$ matrix U is constructed such that under appropriate assumptions

$$\|A - CUR\|_\xi \leq \|A - A_k\|_\xi + \epsilon \|A\|_F , \tag{12}$$

with high probability, for both the spectral and Frobenius norms, $\xi = 2, F$. In [24, 25], it is further shown that if A is a symmetric positive semidefinite (SPSD) matrix, then one can choose $R = C^T$ and $U = W^+$, where W is the $c \times c$ intersection between C and $R = C^T$, thus obtaining an approximation $A \approx A' = CW^+C^T$. This approximation is SPSD and has provable bounds of the form (12), except that the scale of the additional additive error is somewhat larger [24, 25].

Most relevant for our relative-error CX and *CUR* matrix decomposition algorithms is the recent work of Rademacher, Vempala, and Wang [53] and Deshpande, Rademacher, Vempala, and Wang [17]. Using two different methods (in one case iterative sampling in a backwards manner and an induction on k argument [53], and in the other case an argument that relies on estimating the volume of the simplex formed by each of the k-sized subsets of the columns [17]), they reported the *existence* of a set of $O(k^2/\epsilon^2)$ columns that provide relative-error CX matrix decomposition. No algorithmic result was presented, except for an exhaustive algorithm that ran in $\Omega(n^k)$ time. Note that their results did not apply to columns and rows *simultaneously*. Thus, ours is the first *CUR* matrix decomposition algorithm with relative error, and it was previously not even known whether such a relative-error *CUR* representation existed; i.e., it was not previously known whether columns and rows satisfying the conditions of Theorem 2 existed.

Other related work includes that of Rudelson and Vershynin [54, 62, 56], who provide an algorithm for CX matrix decomposition which has an improved additive error spectral norm bound of the form

$$\|A - CX\|_2 \leq \|A - A_k\|_2 + \epsilon \sqrt{\|A\|_2 \, \|A\|_F} .$$

Their proof uses an elegant result on random vectors in the isotropic position [54], and since we use a variant of their result, it is described in more detail in Appendix A. Achlioptas and McSherry have computed low-rank matrix approximations using sampling techniques that involve zeroing-out and/or quantizing individual elements [2, 1]. The primary focus of their work was in introducing methods to accelerate orthogonal iteration and Lanczos iteration methods, and their analysis relied heavily on ideas from random matrix theory [2, 1]. Agarwal, Har-Peled, and Varadarajan have analyzed so-called "core sets" as a tool for efficiently approximating various extent measures of a point set [3, 4]. The choice of columns and/or rows we present are a "core set" for approximate matrix computations; in fact, our algorithmic solution to Theorem 1 solves an open question in their survey [4]. The choice of columns and rows we present may also be viewed as a set of variables and features chosen from a data matrix [10, 14, 41]. "Feature selection" is a broad area that addresses the choice of columns explicitly for dimension reduction, but the metrics there are typically optimization based [14] or machine-learning based [10]. These formulations tend to have set cover-like solutions

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

and are incomparable with the linear-algebraic structure such as the low-rank criteria we consider here that is common among data analysts.

3.3. Very recent work on relative-error approximation algorithms. To the best of our knowledge, the first nontrivial *algorithmic result* for relative-error low-rank matrix approximation was provided by a preliminary version of this paper [27, 28]. In particular, an earlier version of Theorem 1 provided the first known relative-error, column-based, low-rank approximation in polynomial time [27, 28]. The major difference between our Theorem 1 and our result in [27, 28] is that the sampling probabilities in [27, 28] are more complicated. (See section 6.2 for details on this.) The algorithm from [27, 28] runs in $O(SVD(A,k))$ time (although it was originally reported to run in only $O(SVD(A))$ time), and it has a sampling complexity of $O(k^2 \log(1/\delta)/\epsilon^2)$ columns.

Subsequent to the completion of the preliminary version of this paper [27, 28], several developments have been made on relative-error low-rank matrix approximation algorithms. First, Har-Peled reported an algorithm that takes as input an $m \times n$ matrix A, and in roughly $O(mnk^2 \log k)$ time returns as output a rank-k matrix A' with a relative-error approximation guarantee [42]. His algorithm uses geometric ideas and involves sampling and merging approximately optimal k-flats; it is not clear if this approximation can be expressed in terms of a small number of columns of A. Then, Deshpande and Vempala [18] reported an algorithm that takes as input an $m \times n$ matrix A that also returns a relative-error approximation guarantee. Their algorithm extends ideas from [53, 17], and it leads to a CX matrix decomposition consisting of $O(k \log k)$ columns of A. The complexity of their algorithm is $O(Mk^2 \log k)$, where M is the number of nonzero elements of A, and their algorithm can be implemented in a data-streaming framework with $O(k \log k)$ passes over the data. In light of these developments, we simplified and generalized our preliminary results [27, 28], and we performed a more refined analysis to improve our sampling complexity to $O(k \log k)$. Most recently, we learned of work by Sarlos [57], who used ideas from the recently developed fast Johnson–Lindenstrauss transform of Ailon and Chazelle [5] to yield further improvements to a CX matrix decomposition.

4. Our main column-based matrix approximation algorithm. In this section, we describe an algorithm and a theorem, from which our first main result, Theorem 1, will follow.

4.1. Description of the algorithm. Algorithm 1 takes as input an $m \times n$ matrix A, a rank parameter k, and an error parameter ϵ. It returns as output an $m \times c$ matrix C consisting of a small number of columns of A. The algorithm is very simple: sample a small number of columns according to a carefully constructed nonuniform probability distribution. Algorithm 1 uses the sampling probabilities

$$p_i = \frac{1}{k} \left| \left(V_{A,k}^T\right)^{(i)} \right|_2^2, \qquad \forall i \in [n], \tag{13}$$

but it will be clear from the analysis of section 6 that any sampling probabilities such that $p_i \geq \beta |(V_{A,k}^T)^{(i)}|_2^2/k$, for some $\beta \in (0,1]$, will also work with a small β-dependent loss in accuracy. Note that Algorithm 1 actually consists of two related algorithms, depending on how exactly the columns are chosen. The EXACTLY(c) algorithm picks exactly c columns of A to be included in C in c i.i.d. trials, where in each trial the ith column of A is picked with probability p_i. The EXPECTED(c) algorithm picks in expectation at most c columns of A to create C, by including the ith column of A

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

in C with probability $\min\{1, cp_i\}$. See Algorithms 4 and 5 in Appendix A for more details about these two column-sampling procedures.

Data : $A \in \mathbb{R}^{m\times n}$, a rank parameter k, and an error parameter ϵ.
Result : $C \in \mathbb{R}^{m\times c}$

- Compute sampling probabilities p_i for all $i \in [n]$ given by (13);
- (Implicitly) construct a sampling matrix S_C and a diagonal rescaling matrix D_C with the EXACTLY(c) algorithm or with the EXPECTED(c) algorithm;
- Construct and return the matrix $C = AS_CD_C$ consisting of a small number of rescaled columns of A.

Algorithm 1. A randomized algorithm for CX matrix decomposition.

The running time of Algorithm 1 is dominated by the computation of the sampling probabilities (13), for which $O(SVD(A,k))$ time suffices. The top k right singular vectors of A can be efficiently (approximately) computed using standard algorithms [37, 51]. The building block of these algorithms is a series of matrix-vector multiplications, where the input matrix A is iteratively multiplied with a changing set of k orthogonal vectors. In each iteration (which can be implemented by making passes over the input matrix A), the accuracy of the approximation improves. Even though the number of iterations required to bound the error depends on quantities such as the gap between the singular values of A, these algorithms work extremely well in practice. As such, they are often treated as "black boxes" for SVD computation in the TCS literature; see, e.g., [2, 1].

4.2. Statement of the theorem. Theorem 3 is our main quality-of-approximation result for Algorithm 1.

THEOREM 3. *Let $A \in \mathbb{R}^{m\times n}$, let k be a rank parameter, and let $\epsilon \in (0,1]$. If we set $c = 3200k^2/\epsilon^2$ and run Algorithm* 1 *by choosing exactly c columns from A with the* EXACTLY(c) *algorithm, then with probability at least* 0.7

$$\left\|A - CC^+A\right\|_F \le (1+\epsilon)\left\|A - A_k\right\|_F. \tag{14}$$

Similarly, if we set $c = O(k\log k/\epsilon^2)$ and run Algorithm 1 *by choosing no more than c columns in expectation from A with the* EXPECTED(c) *algorithm, then* (14) *holds with probability at least* 0.7.

Proof. Since for every set of columns $C = AS_CD_C$, $X_{opt} = C^+A$ is the matrix that minimizes $\|A - CX\|_F$, it follows that

$$\begin{aligned}\left\|A - CC^+A\right\|_F &= \left\|A - (AS_CD_C)(AS_CD_C)^+A\right\|_F \\ &\le \left\|A - (AS_CD_C)(P_{A,k}AS_CD_C)^+P_{A,k}A\right\|_F,\end{aligned} \tag{15}$$

where $P_{A,k} = U_{A,k}U_{A,k}^T$ is a projection onto the top k left singular vectors of A. To bound (15), consider the problem of approximating the solution to $\min_{X\in\mathbb{R}^{m\times m}} \|XA_k - A\|_F$ by randomly sampling columns of A_k and of A. It follows as a corollary of (21) of Theorem 5 of section 6 that

$$\left\|A - (AS_CD_C)(A_kS_CD_C)^+A_k\right\|_F \le (1+\epsilon)\left\|A - AA_k^+A_k\right\|_F = (1+\epsilon)\left\|A - A_k\right\|_F, \tag{16}$$

which, when combined with (15), establishes the theorem. ∎

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Remark. For simplicity of presentation, we have presented Algorithm 1 and Theorem 3 such that (14) holds with only constant probability, but this can be boosted to hold with probability at least $1-\delta$ using standard methods. In particular, consider the following: run Algorithm 1 (using either the EXACTLY(c) algorithm or the EXPECTED(c) algorithm, but with the appropriate value of c) independently $\ln(1/\delta)$ times, and return the C such that $\|A - CC^+A\|_F$ is smallest. Then, since in each trial the claim of Theorem 3 fails with probability less than $0.3 < 1/e$, the claim of Theorem 3 will fail for every trial with probability less than $(1/e)^{\ln(1/\delta)} = \delta$. This establishes Theorem 1.

Remark. For simplicity of presentation, we have also stated Theorem 3 in such a way that the rank of the approximating matrix $A' = CC^+A$ may be greater than k. This possibility may be undesirable in certain applications, and it can be easily removed. Let $A'' = C(P_{A,k}C)^+P_{A,k}A$. Then, it follows from (16) that A'' is a CX matrix approximation that is within relative error ϵ of the best rank-k approximation to A and that has rank no more than k.

4.3. Discussion of the analysis. Given a matrix A, Theorem 1 asks us to find a set of columns $C = AS_CD_C$ such that CC^+A "captures" almost as much of A as does $A_k = U_{A,k}U_{A,k}^TA$. Given that set (or any other set) of columns C, it is well known that the matrix $X_{opt} = C^+A$ is the "smallest" matrix among those that solve the optimization problem (19). For a given A and C, let us approximate X_{opt} as

$$X_{opt} = C^+A \approx \left(P_{A,k}C\right)^+ P_{A,k}A.$$

This approximation is suboptimal with respect to solving the optimization problem (19), i.e.,

$$\left\|A - CC^+A\right\|_F \leq \left\|A - C\left(P_{A,k}C\right)^+ P_{A,k}A\right\|_F,$$

but it can be shown that by choosing C properly, i.e., by choosing S_C and D_C (the column sampling and rescaling matrices) properly, we have that

$$\left\|A - C\left(P_{A,k}C\right)^+ P_{A,k}A\right\|_F \leq (1+\epsilon)\left\|A - A_k\right\|_F.$$

The main technical challenge is to sample in a manner such that the column-sampled version of the matrix consisting of the top k right singular vectors of A is full rank; i.e., $\mathrm{rank}(V_{A,k}^TS_CD_C) = \mathrm{rank}(V_{A,k}^T) = k$. To accomplish this, we sample with respect to probabilities of the form (13). To understand these sampling probabilities, recall that we seek to pick columns that span almost the same subspace as the top k left singular vectors of A (i.e., U_k), and recall that the ith column of A is equal to

$$A^{(i)} = U_k\Sigma_k\left(V_k^T\right)^{(i)} + U_{\rho-k}\Sigma_{\rho-k}\left(V_{\rho-k}^T\right)^{(i)}.$$

Since postmultiplying U_k by Σ_k does not change the span of the columns of U_k, $|(V_k^T)^{(i)}|_2^2$ measures "how much" of the ith column of A lies in the span of $U_{A,k}$, independent of the magnitude of the singular values associated with those directions.

5. Our main column-row-based matrix approximation algorithm. In this section, we describe an algorithm and a theorem that, when combined with the results of section 4, will establish our second main result, Theorem 2.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

5.1. Description of the algorithm. Algorithm 2 takes as input an $m \times n$ matrix A, an $m \times c$ matrix C consisting of a small number of columns of A, and an error parameter ϵ. It returns as output an $r \times n$ matrix R consisting of a small number of rows of A and an $r \times c$ matrix W consisting of the corresponding rows of C. The algorithm is very simple: sample a small number of rows according to a carefully constructed nonuniform probability distribution. Algorithm 2 uses the sampling probabilities

$$p_i = \frac{1}{c}\left|\left(U_C^T\right)^{(i)}\right|_2^2, \qquad \forall i \in [m], \tag{17}$$

but it will be clear from the analysis of section 6 that any sampling probabilities $p_i, i \in [m]$, such that $p_i \geq \beta|(U_C^T)^{(i)}|_2^2/c$, for some $\beta \in (0,1]$, will also work with a small β-dependent loss in accuracy. Note that Algorithm 2 actually consists of two related algorithms, depending on how exactly the rows are chosen. The EXACTLY(c) algorithm picks exactly r rows of A to be included in R in r i.i.d. trials, where in each trial the ith row of A is picked with probability p_i. The EXPECTED(c) algorithm picks in expectation at most r rows of A to create R, by including the ith column of A in C with probability $\min\{1, rp_i\}$. See Algorithms 4 and 5 in Appendix A for more details about these two row-sampling procedures.

Data : $A \in \mathbb{R}^{m\times n}$, $C \in \mathbb{R}^{m\times c}$ consisting of c columns of A, a positive integer r, and an error parameter ϵ.

Result : $R \in \mathbb{R}^{r\times n}$ consisting of r rows of A and $W \in \mathbb{R}^{c\times r}$ consisting of the corresponding r rows of C, and $U \in \mathbb{R}^{r\times c}$.

- Compute sampling probabilities p_i for all $i \in [m]$ given by (17);
- (Implicitly) construct a sampling matrix S_R and a diagonal rescaling matrix D_R with the EXACTLY(c) algorithm or with the EXPECTED(c) algorithm;
- Construct and return the matrix $R = D_R S_R^T A$ consisting of a small number of rescaled rows of A;
- Construct and return the matrix $W = D_R S_R^T C$ consisting of the corresponding rescaled rows of C;
- Let $U = W^+$.

Algorithm 2. A randomized algorithm for *CUR* matrix decomposition.

Reading the input matrices to Algorithm 2 takes $O(mn)$ time; computing the full SVD of C requires $O(c^2m)$ time; constructing the matrix R requires $O(rn)$ time; constructing the matrix W requires $O(rc)$ time; and computing U requires $O(c^2r)$ time. Overall, the running time of the algorithm is $O(mn)$ since c, r are constants independent of m, n. This can be improved if the input matrices are sparse, but for simplicity we omit this discussion.

5.2. Statement of the theorem. Theorem 4 is our main quality-of-approximation result for Algorithm 2.

THEOREM 4. *Let $A \in \mathbb{R}^{m\times n}$, let $C \in \mathbb{R}^{m\times c}$ be a matrix consisting of any c columns of A, and let $\epsilon \in (0,1]$. If we set $r = 3200c^2/\epsilon^2$ and run Algorithm 2 by choosing r rows exactly from A and from C with the* EXACTLY(c) *algorithm, then*

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

with probability at least 0.7

$$\|A - CUR\|_F \leq (1+\epsilon) \left\|A - CC^+A\right\|_F. \tag{18}$$

Similarly, if we set $r = O(c \log c/\epsilon^2)$ *and run Algorithm* 2 *by choosing no more than* r *rows in expectation from* A *and from* C *with the* EXPECTED(c) *algorithm, then* (18) *holds with probability at least* 0.7.

Proof. Consider the problem of approximating the solution to $\min_{X \in \mathbb{R}^{c \times n}} \|CX - A\|_F$ by randomly sampling rows from C and A. It follows as a corollary of (21) of Theorem 5 of section 6 that

$$\left\|A - C(D_R S_R^T C)^+ D_R S_R^T A\right\|_F \leq (1+\epsilon) \left\|A - CC^+A\right\|_F,$$

where $R = D_R S_R^T A$ and $U = (D_R S_R^T C)^+$, which establishes the theorem. ∎

Remark. For simplicity of presentation, we have presented Algorithm 2 and Theorem 4 such that (18) holds with only constant probability, but this can be boosted to hold with probability at least $1 - \delta$ using standard methods. In addition, this can be combined with Algorithm 1 and Theorem 3 by doing the following: run Algorithm 1 $\ln(2/\delta)$ times, and return the best C; then, with that C run Algorithm 2 $\ln(2/\delta)$ times, and return the best U, R pair. Then

$$\|A - CUR\|_F \leq (1+\epsilon) \left\|A - CC^+A\right\|_F \leq (1+\epsilon)^2 \|A - A_k\|_F \leq (1+\epsilon') \|A - A_k\|_F,$$

where $\epsilon' = 3\epsilon$, and the combined failure probability is no more than $\delta/2 + \delta/2 = \delta$. This establishes Theorem 2.

5.3. Discussion of the analysis. Assume that we are given an $m \times c$ matrix C, consisting of any set of c columns of an $m \times n$ matrix A, and consider the following idea for approximating the matrix A. The columns of C are a set of "basis vectors" that are, in general, neither orthogonal nor normal. To express all the columns of A as linear combinations of the columns of C, we can solve

$$\min_{x_j \in \mathbb{R}^c} \left|A^{(j)} - Cx_j\right|_2,$$

for each column $A^{(j)}, j \in [n]$, in order to find a c-vector of coefficients x_j and get the optimal least-squares fit for $A^{(j)}$. Equivalently, we can solve an optimization problem of the form (19). Note that if m and n are large and $c = O(1)$, then this is an overconstrained least-squares fit problem. It is well known that $X_{opt} = C^+A$ is the "smallest" matrix solving this optimization problem, in which case we are using information from every row of A to compute the optimal coefficient matrix. Let us approximate X_{opt} as

$$X_{opt} = C^+A \approx \left(D_R S_R^T C\right)^+ D_R S_R^T A = \tilde{X}_{opt},$$

and note that $\tilde{X}_{opt} = W^+R$. This matrix $\tilde{X}_{opt}$ is clearly suboptimal with respect to solving the optimization problem (19), i.e.,

$$\left\|A - CC^+A\right\|_F \leq \left\|A - CW^+R\right\|_F,$$

but it can be shown that by choosing S_R and D_R (the row sampling and rescaling matrices) properly we have that

$$\left\|A - CW^+R\right\|_F \leq (1+\epsilon) \left\|A - CC^+A\right\|_F.$$

As in section 4.3, the main technical challenge is to sample in a manner such that the row-sampled version of the matrix consisting of the top c left singular vectors of C is full rank, i.e., $\operatorname{rank}(D_R S_R^T U_{C,c}) = \operatorname{rank}(U_{C,c}) = c$.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

6. An approximation algorithm for generalized ℓ_2 regression. The basic linear-algebraic problem of ℓ_2 regression is one of the most fundamental regression problems, and it has found many applications in mathematics and statistical data analysis. Recall the standard ℓ_2 regression (or least-squares fit) problem: given as input a matrix $A \in \mathbb{R}^{m \times n}$ and a target vector $b \in \mathbb{R}^m$, compute $\mathcal{Z} = \min_{x \in \mathbb{R}^n} |b - Ax|_2$. Also of interest is the computation of vectors that achieve the minimum $\mathcal{Z}$. If $m > n$ there are more constraints than variables and the problem is an overconstrained least-squares fit problem; in this case, there does not in general exist a vector x such that $Ax = b$. It is well known that the minimum-length vector among those minimizing $|b - Ax|_2$ is $x_{opt} = A^+ b$. We previously presented an elaborate sampling algorithm that represents the matrix A by a matrix by a small number of rows so that this ℓ_2 regression problem can be solved to accuracy $1 \pm \epsilon$ for any $\epsilon > 0$ [29].

This problem is of interest for CX and *CUR* matrix decomposition for the following reason. Given a matrix A and a set of its columns C, if we want to get the best fit for every column of A in terms of that basis, we want to solve $CX \approx A$ for the matrix X. More precisely, we would like to solve the optimization problem such as

$$\mathcal{Z} = \min_{X \in \mathbb{R}^{c \times n}} \|A - CX\|_F . \tag{19}$$

It is well known that the matrix $X = C^+ A$ is the "smallest" matrix among those that solve this problem. In this case, we are approximating the matrix A as $A' = CC^+ A = P_C A$, and by keeping only the columns C we are incurring an error of $\|A - CC^+ A\|_F$. Two questions arise:

- First, how do we choose the columns C such that $\|A - CC^+ A\|_F$ is within relative error ϵ of $\|A - A_k\|_F$?
- Second, how do we choose the rows R and a matrix U such that $\|A - CUR\|_F$ is within relative error ϵ of $\|A - CC^+ A\|_F$?

Motivated by these observations, we will consider the generalized version of the standard ℓ_2 regression problem, as defined in (5) and (6).

In this section, we first present Algorithm 3, which is our main random sampling algorithm for approximating the solution to the generalized ℓ_2 regression problem, and Theorem 5, which provides our main quality-of-approximation bound for Algorithm 3. Then, we discuss the novel nonuniform "subspace sampling" probabilities used by the algorithm. Finally, we present the proof of Theorem 5.

6.1. Description of the algorithm and theorem. Algorithm 3 takes as input an $m \times n$ matrix A with rank no greater than k, an $m \times p$ matrix B, a set of sampling probabilities $\{p_i\}_{i=1}^m$, and a positive integer $r \leq m$. It returns as output a number $\tilde{\mathcal{Z}}$ and a $n \times p$ matrix $\tilde{X}_{opt}$. Using the sampling matrix formalism described in section 2, the algorithm (implicitly) forms a sampling matrix S, the transpose of which samples a few rows of A and the corresponding rows of B, and a rescaling matrix D, which is a matrix scaling the sampled rows of A and B. Since r rows of A and the corresponding r rows of B are sampled, the algorithm randomly samples r of the m constraints in the original ℓ_2 regression problem. Thus, the algorithm approximates the solution of the regression problem $AX \approx B$, as formalized in (19) and (5), with the exact solution of the downsampled regression problem $DS^T AX \approx DS^T B$. Note that it is the space of constraints that is sampled and that the dimensions of the unknown matrix X are the same in both problems. Note also that although both m and n are permitted to be large, the problem is effectively overconstrained since $\text{rank}(A) \leq k$. As we will see below, $r = O(k \log k)$ or $r = O(k^2)$, depending on exactly how the random sample is constructed. Thus, we will compute the solution to the sampled problem exactly.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Data : $A \in \mathbb{R}^{m\times n}$ that has rank no greater than k, $B \in \mathbb{R}^{m\times p}$, sampling probabilities $\{p_i\}_{i=1}^m$, and $r \le m$.

Result : $\tilde{X}_{opt} \in \mathbb{R}^{n\times p}$, $\tilde{\mathcal{Z}} \in \mathbb{R}$.

- (Implicitly) construct a sampling matrix S and a diagonal rescaling matrix D with the EXACTLY(c) algorithm or with the EXPECTED(c) algorithm;
- Construct the matrix DS^TA consisting of a small number of rescaled rows of A;
- Construct the matrix DS^TB consisting of a small number of rescaled rows of B;
- $\tilde{X}_{opt} = \left(DS^TA\right)^+ DS^TB$;
- $\tilde{\mathcal{Z}} = \min_{X\in\mathbb{R}^{n\times p}} \left\|DS^TB - DS^TA\tilde{X}_{opt}\right\|_F$.

Algorithm 3. A Monte-Carlo algorithm for approximating ℓ_2 regression.

Theorem 5 is our main quality-of-approximation result for Algorithm 3. Its proof may be found in section 6.3. Recall that for our generalized ℓ_2 regression problem, the matrix A has rank no greater than k.

THEOREM 5. *Suppose $A \in \mathbb{R}^{m\times n}$ has rank no greater than k, $B \in \mathbb{R}^{m\times p}$, $\epsilon \in (0,1]$, and let $\mathcal{Z} = \min_{X\in\mathbb{R}^{n\times p}} \|B - AX\|_F = \|B - AX_{opt}\|_F$, where $X_{opt} = A^+B = A_k^+B$. Run Algorithm 3 with any sampling probabilities of the form*

$$p_i \ge \beta \frac{\left|(U_{A,k})_{(i)}\right|_2^2}{\sum_{j=1}^n \left|(U_{A,k})_{(j)}\right|_2^2} = \frac{\beta}{k}\left|(U_{A,k})_{(i)}\right|_2^2, \qquad \forall i \in [n], \tag{20}$$

for some $\beta \in (0,1]$, and assume that the output of the algorithm is a number $\tilde{\mathcal{Z}}$ and an $n\times p$ matrix $\tilde{X}_{opt}$. If exactly $r = 3200k^2/\beta\epsilon^2$ rows are chosen with the EXACTLY(c) *algorithm, then with probability at least* 0.7*:*

$$\left\|B - A\tilde{X}_{opt}\right\|_F \le (1+\epsilon)\,\mathcal{Z}, \tag{21}$$

$$\left\|X_{opt} - \tilde{X}_{opt}\right\|_F \le \frac{\epsilon}{\sigma_{\min}(A_k)}\mathcal{Z}. \tag{22}$$

If, in addition, we assume that $\|U_{A,k}U_{A,k}^TB\|_F \ge \gamma\|B\|_F$, for some fixed $\gamma \in (0,1]$, then with probability at least 0.7*:*

$$\left\|X_{opt} - \tilde{X}_{opt}\right\|_F \le \epsilon\left(\kappa(A_k)\sqrt{\gamma^{-2}-1}\right)\|X_{opt}\|_F. \tag{23}$$

Similarly, under the same assumptions, if $r = O(k\log k/\beta\epsilon^2)$ rows are chosen in expectation with the EXPECTED(c) *algorithm, then with probability at least* 0.7*,* (21)*,* (22)*, and* (23) *hold.*

Equation (21) states that if the matrix of minimum-length vectors achieving the minimum in the sampled problem is substituted back into the residual norm for the original problem, then a good approximation to the original ℓ_2 regression problem is obtained. Equation (22) provides a bound for $||X_{opt}-\tilde{X}_{opt}||_F$ in terms of $\sigma_{\min}(A_k)$ and

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

$\mathcal{Z}$. If most of the "weight" of B lies in the complement of the column space of $A = A_k$ then this will provide a very poor approximation in terms of $\|X_{opt}\|_F$. However, if we also assume that a constant fraction of the "weight" of B lies in the subspace spanned by the columns of A, then we obtain the relative-error approximation of (23). Thus, Theorem 5 returns a good bound for $||X_{opt} - \tilde{X}_{opt}||_F$ if A_k is well-conditioned and if B lies "reasonably well" in the column space of A. Note that if the matrix of target vectors B lies completely within the column space of A, then $\mathcal{Z} = 0$ and $\gamma = 1$. In this case, Theorem 5 shows that Algorithm 3 returns $\tilde{\mathcal{Z}}$ and $\tilde{x}_{opt}$ that are exact solutions of the original ℓ_2 regression problem, independent of $\kappa(A_k)$. Finally, note that in our analysis of CX and *CUR* matrix decompositions we use only the result (21) from Theorem 5, but (22) and (23) are included for completeness.

6.2. Discussion of the method of "Subspace Sampling". An important aspect of Algorithm 3 is the nonuniform sampling probabilities (20) used by the EXACTLY(c) algorithm and the EXPECTED(c) algorithm in the construction of the induced subproblem. We call sampling probabilities satisfying condition (20) *"subspace sampling"* probabilities. Condition (20) states that the sampling probabilities should be close to, or rather not much less than, the lengths, i.e., the Euclidean norms, of the rows of the left singular vectors of the matrix $A = A_k$. (Recall that in this section A is an $m \times n$ matrix with rank no more than k, and thus $U_{A,k}$ is an $m \times k$ matrix. Thus, the Euclidean norm of every *column* of $U_{A,k}$ equals 1, but the Euclidean norm of every *row* of $U_{A,k}$ is in general not equal and is only bounded above by 1.) Sampling probabilities of the form (20) should be contrasted with sampling probabilities that depend on the Euclidean norms of the columns or rows of A and that have received much attention recently [33, 34, 21, 22, 23, 26]. Since $A = U_A \Sigma_A V_A^T$, sampling probabilities with this latter form depend in a complicated manner on a mixture of subspace information (as found in U_A and V_A) and "size-of-A" information (as found in Σ_A). This convolution of information may account for their ability to capture coarse statistics such as approximating matrix multiplication or computing low-rank matrix approximations to additive error, but it also accounts for their difficulty in dealing with problems such as ℓ_2 regression or computing low-rank matrix approximations to relative error.

Since the solution of the ℓ_2 regression problem involves the computation of a pseudoinverse, the problem is not well-conditioned with respect to a perturbation (such as that introduced by sampling) that entails a change in dimensionality, even if (actually, especially if) that change in dimensionality corresponds to a small singular value. Since sampling probabilities satisfying (20) allow us to disentangle subspace information and "size-of-A" information, we will see that they will allow us to capture (with high probability) the *entire* subspace of interest by sampling. More precisely, as we will see in Lemma 1, by using sampling probabilities that satisfy condition (20) and by choosing r appropriately, it will follow that

$$\text{rank}(DS^T U_{A,k}) = \text{rank}(U_{A,k}) = k.$$

Thus, the lengths of the Euclidean norms of the rows of $U_{A,k}$ may be interpreted as capturing a notion of information dispersal by the matrix A since they indicate to which part of the m-dimensional vector space the singular value information of A is being dispersed. In this case, condition (20) ensures that the sampling probabilities provide a bias toward the part of the high-dimensional constraint space to which A disperses its singular value information. Then, having constructed the sample, we will go to the low-dimensional, i.e., the r-dimensional rather than the m-dimensional

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

space, and approximate the ℓ_2 regression problem by doing computations that involve "size-of-A" information on the random sample.

This method of "subspace sampling" was first used in a preliminary version of the ℓ_2 regression results of this section [29]. Note that an immediate generalization of the results of [29] to the generalized ℓ_2 regression problem considered in this section would involve sampling probabilities of the form

(24)
$$p_i = \frac{(1/3)\left|(U_{A,k})_{(i)}\right|_2^2}{\sum_{j=1}^n \left|(U_{A,k})_{(j)}\right|_2^2} + \frac{(1/3)\left|(U_{A,k})_{(i)}\right|_2 \left(U_{A,k}^{\perp} {U_{A,k}^{\perp}}^T B\right)_i}{\sum_{j=1}^n \left|(U_{A,k})_{(j)}\right|_2 \left(U_{A,k}^{\perp} {U_{A,k}^{\perp}}^T B\right)_j} + \frac{(1/3)\left(U_{A,k}^{\perp} {U_{A,k}^{\perp}}^T B\right)_i^2}{\sum_{j=1}^n \left(U_{A,k}^{\perp} {U_{A,k}^{\perp}}^T B\right)_j^2},$$

rather than of the form (20). Since the second and third terms in (24) provide a bias toward the part of the complement of the column space of $A = A_k$ where B has significant weight, we directly obtain variance reduction. Thus, by using probabilities of the form (24) we can sample $O(k^2 \log(1/\delta)/\epsilon^2)$ columns and directly obtain the claims of Theorem 5 with probability at least $1 - \delta$. Although sampling probabilities of the form (20) are substantially simpler, we obtain variance control indirectly. We first establish that each of the claims of Theorem 5 holds with constant probability, and we then can show that each of the claims holds with probability at least $1 - \delta$ by running $O(\log(1/\delta))$ trials and using standard boosting procedures.

6.3. Proof of Theorem 5. In this section we provide a proof of Theorem 5. We will first prove (21), (22), and (23) under the assumption that the rows of A and B are sampled with the EXACTLY(c) algorithm. Then, in section 6.3.5, we will outline modifications to the proof if the rows of A and B are sampled with the EXPECTED(c) algorithm. For simplicity of notation in this section, we will let $\mathcal{S} = DS^T$ denote the $r \times m$ rescaled row-sampling matrix. Let the rank of the $m \times n$ matrix A be $\rho \leq k$, and let its SVD be

$$A = U_A \Sigma_A V_A^T,$$

where $U_A \in \mathbb{R}^{n \times \rho}$, $\Sigma_A \in \mathbb{R}^{\rho \times \rho}$, and $V_A \in \mathbb{R}^{d \times \rho}$. In addition, let the rank of the $r \times \rho$ matrix $\mathcal{S}U_A = DS^T U_A$ be $\tilde{\rho}$, and let its SVD be

$$\mathcal{S}U_A = U_{\mathcal{S}U_A} \Sigma_{\mathcal{S}U_A} V_{\mathcal{S}U_A}^T,$$

where $U_{\mathcal{S}U_A} \in \mathbb{R}^{r \times \tilde{\rho}}$, $\Sigma_{\mathcal{S}U_A} \in \mathbb{R}^{\tilde{\rho} \times \tilde{\rho}}$, and $V_{\mathcal{S}U_A} \in \mathbb{R}^{\rho \times \tilde{\rho}}$. Recall that $\tilde{\rho} \leq \rho \leq k \leq r$.

In order to illustrate the essential difficulty in constructing a sampling algorithm to approximate the solution of the generalized ℓ_2 regression problem, consider inserting $\tilde{X}_{opt} = (\mathcal{S}A_k)^+ \mathcal{S}B$ into $B - A_k X$:

$$\begin{aligned}
B - A_k \tilde{X}_{opt} &= B - A_k \left(\mathcal{S}A_k\right)^+ \mathcal{S}B \\
&= B - U_{A,k}\Sigma_{A,k}V_{A,k}^T \left(\mathcal{S}U_{A,k}\Sigma_{A,k}V_{A,k}^T\right)^+ \mathcal{S}B \\
&= B - U_{A,k}\Sigma_{A,k} \left(\mathcal{S}U_{A,k}\Sigma_{A,k}\right)^+ \mathcal{S}B \\
&= B - U_{A,k}\Sigma_{A,k} \left(U_{\mathcal{S}U_{A,k}}\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\Sigma_{A,k}\right)^+ \mathcal{S}B \\
&= B - U_{A,k}\Sigma_{A,k} \left(\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\Sigma_{A,k}\right)^+ U_{\mathcal{S}U_{A,k}}^T \mathcal{S}B.
\end{aligned}$$

To proceed further, we must deal with the pseudoinverse, which is not well-behaved with respect to perturbations that involve a change in dimensionality. To deal with this, we will focus on probabilities that depend on the subspace that we are down-

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

sampling, i.e., that depend on $U_{A,k}$, in order to guarantee that we capture the full subspace of interest.

6.3.1. Several lemmas of general interest. In this subsection, we will present three lemmas of general interest. Then, in the next subsections, we will use these lemmas to prove each of the claims of Theorem 5.

Since the $m \times k$ matrix $U_{A,k}$ is a matrix with orthogonal columns, several properties hold for it. For example, $\text{rank}(U_{A,k}) = k$, $U^+ = U^T$, and $A_k^+ = V_{A,k}\Sigma_{A,k}^{-1}U_{A,k}^T$. Although the $r \times k$ matrix $\mathcal{S}U_{A,k}$ does not have orthogonal columns, the following lemma characterizes the manner in which each of these three properties holds, either exactly or approximately. For the first lemma, r depends quadratically on k.

LEMMA 1. *Let $\epsilon \in (0,1]$, and define $\Omega = (\mathcal{S}U_{A,k})^+ - (\mathcal{S}U_{A,k})^T$. If the sampling probabilities satisfy* (20) *and if $r \geq 400k^2/\beta\epsilon^2$, then with probability at least* 0.9*:*

$$\tilde{\rho} = \rho, \text{ i.e., } \mathit{rank}(\mathcal{S}U_{A,k}) = \mathit{rank}(U_{A,k}) = \mathit{rank}(A_k), \tag{25}$$

$$\|\Omega\|_2 = \left\|\Sigma_{\mathcal{S}U_{A,k}}^{-1} - \Sigma_{\mathcal{S}U_{A,k}}\right\|_2, \tag{26}$$

$$(\mathcal{S}A_k)^+ = V_{A,k}\Sigma_{A,k}^{-1}\left(\mathcal{S}U_{A,k}\right)^+, \tag{27}$$

$$\left\|\Sigma_{\mathcal{S}U_{A,k}} - \Sigma_{\mathcal{S}U_{A,k}}^{-1}\right\|_2 \leq \epsilon/\sqrt{2}. \tag{28}$$

Proof. To prove the first claim, note that for all $i \in [\rho]$

$$\begin{aligned} \left|1 - \sigma_i^2\left(\mathcal{S}U_{A,k}\right)\right| &= \left|\sigma_i\left(U_{A,k}^T U_{A,k}\right) - \sigma_i\left(U_{A,k}^T \mathcal{S}^T \mathcal{S}U_{A,k}\right)\right| \\ &\leq \left\|U_{A,k}^T U_{A,k} - U_{A,k}^T \mathcal{S}^T \mathcal{S}U_{A,k}\right\|_2 \qquad (29) \\ &\leq \left\|U_{A,k}^T U_{A,k} - U_{A,k}^T \mathcal{S}^T \mathcal{S}U_{A,k}\right\|_F. \qquad (30) \end{aligned}$$

Note that (29) follows from Corollary 8.1.6 of [37], and (30) follows since $\|\cdot\|_2 \leq \|\cdot\|_F$. To bound the error of approximating $U_{A,k}^T U_{A,k}$ by $U_{A,k}^T \mathcal{S}^T \mathcal{S}U_{A,k}$, we apply Theorem 6 of Appendix A. Since the sampling probabilities p_i satisfy (20), it follows from Theorem 6 and by applying Markov's inequality that with probability at least 0.9:

$$\begin{aligned} \left\|U_{A,k}^T U_{A,k} - U_{A,k}^T \mathcal{S}^T \mathcal{S}U_{A,k}\right\|_F &\leq 10\, \mathbf{E}\left[\left\|U_{A,k}^T U_{A,k} - U_{A,k}^T \mathcal{S}^T \mathcal{S}U_{A,k}\right\|_F\right] \\ &\leq \frac{10}{\sqrt{\beta r}}\left\|U_{A,k}\right\|_F^2, \qquad (31) \end{aligned}$$

where $\mathbf{E}[\cdot]$ denotes the expectation operator. By combining (30) and (31), recalling that $\|U_{A,k}\|_F^2 = \rho \leq k$, and using the assumed choice of r, it follows that

$$\left|1 - \sigma_i^2\left(\mathcal{S}U_{A,k}\right)\right| \leq \epsilon/2 \leq 1/2$$

since $\epsilon \leq 1$. This implies that all singular values of $\mathcal{S}U_{A,k}$ are strictly positive, and thus that $\text{rank}(\mathcal{S}U_{A,k}) = \text{rank}(U_{A,k}) = \text{rank}(A_k)$, which establishes the first claim.

To prove the second claim, we use the SVD of $\mathcal{S}U_{A,k}$ and note that

$$\begin{aligned} \|\Omega\|_2 &= \left\|\left(\mathcal{S}U_{A,k}\right)^+ - \left(\mathcal{S}U_{A,k}\right)^T\right\|_2 \\ &= \left\|\left(U_{\mathcal{S}U_{A,k}}\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\right)^+ - \left(U_{\mathcal{S}U_{A,k}}\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\right)^T\right\|_2 \\ &= \left\|V_{\mathcal{S}U_{A,k}}\left(\Sigma_{\mathcal{S}U_{A,k}}^{-1} - \Sigma_{\mathcal{S}U_{A,k}}\right)U_{\mathcal{S}U_{A,k}}^T\right\|_2. \end{aligned}$$

The claim follows since $V_{\mathcal{S}U_{A,k}}$ and $U_{\mathcal{S}U_{A,k}}$ are matrices with orthonormal columns.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

To prove the third claim, note that

$$\begin{aligned}(\mathcal{S}A_k)^+ &= \left(\mathcal{S}U_{A,k}\Sigma_{A,k}V_{A,k}^T\right)^+ \\ &= \left(U_{\mathcal{S}U_{A,k}}\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\Sigma_{A,k}V_{A,k}^T\right)^+ \\ &= V_{A,k}\left(\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\Sigma_{A,k}\right)^+ U_{\mathcal{S}U_{A,k}}^T. \qquad (32)\end{aligned}$$

To remove the pseudoinverse in the above derivations, notice that since $\rho = \tilde{\rho}$ with probability at least 0.9, all three matrices $\Sigma_{\mathcal{S}U_{A,k}}$, $V_{\mathcal{S}U_{A,k}}$, and $\Sigma_{A,k}$ are full rank square $\rho \times \rho$ matrices, and thus are invertible. In this case,

$$\begin{aligned}\left(\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\Sigma_{A,k}\right)^+ &= \left(\Sigma_{\mathcal{S}U_{A,k}}V_{\mathcal{S}U_{A,k}}^T\Sigma_{A,k}\right)^{-1} \\ &= \Sigma_{A,k}^{-1}V_{\mathcal{S}U_{A,k}}\Sigma_{\mathcal{S}U_{A,k}}^{-1}. \qquad (33)\end{aligned}$$

By combining (32) and (33) we have that

$$\begin{aligned}(\mathcal{S}A_k)^+ &= V_{A,k}\Sigma_{A,k}^{-1}V_{\mathcal{S}U_{A,k}}\Sigma_{\mathcal{S}U_{A,k}}^{-1}U_{\mathcal{S}U_{A,k}}^T \\ &= V_{A,k}\Sigma_{A,k}^{-1}\left(\mathcal{S}U_{A,k}\right)^+,\end{aligned}$$

which establishes the third claim.[2]

Finally, to prove the fourth claim, recall that under the assumptions of the lemma $\rho = \tilde{\rho}$ with probability at least 0.9, and thus $\sigma_i\left(\mathcal{S}U_{A,k}\right) > 0$ for all $i \in [\rho]$. Thus,

$$\begin{aligned}\left\|\Sigma_{\mathcal{S}U_{A,k}}^{-1} - \Sigma_{\mathcal{S}U_{A,k}}\right\|_2 &= \max_{i,j\in[\rho]}\left|\sigma_i\left(\mathcal{S}U_{A,k}\right) - \frac{1}{\sigma_j\left(\mathcal{S}U_{A,k}\right)}\right| \\ &= \max_{i,j\in[\rho]}\frac{\left|\sigma_i\left(\mathcal{S}U_{A,k}\right)\sigma_j\left(\mathcal{S}U_{A,k}\right) - 1\right|}{\left|\sigma_j\left(\mathcal{S}U_{A,k}\right)\right|} \\ &\leq \max_{j\in[\rho]}\frac{\left|\sigma_j^2\left(\mathcal{S}U_{A,k}\right) - 1\right|}{\left|\sigma_j\left(\mathcal{S}U_{A,k}\right)\right|}. \qquad (34)\end{aligned}$$

Using the fact that, by (29), for all $i \in [\rho]$,

$$\left|1 - \sigma_i^2\left(\mathcal{S}U_{A,k}\right)\right| \leq \left\|U_{A,k}^TU_{A,k} - U_{A,k}^T\mathcal{S}^T\mathcal{S}U_{A,k}\right\|_2,$$

it follows that for all $i \in [\rho]$

$$\frac{1}{\sigma_i\left(\mathcal{S}U_{A,k}\right)} \leq \frac{1}{\sqrt{1 - \left\|U_{A,k}^TU_{A,k} - U_{A,k}^T\mathcal{S}^T\mathcal{S}U_{A,k}\right\|_2}}.$$

When these are combined with (34) it follows that

$$\left\|\Sigma_{\mathcal{S}U_{A,k}} - \Sigma_{\mathcal{S}U_{A,k}}^{-1}\right\|_2 \leq \frac{\left\|U_{A,k}^TU_{A,k} - U_{A,k}^T\mathcal{S}^T\mathcal{S}U_{A,k}\right\|_2}{\sqrt{1 - \left\|U_{A,k}^TU_{A,k} - U_{A,k}^T\mathcal{S}^T\mathcal{S}U_{A,k}\right\|_2}}.$$

[2] One might be tempted to suggest that the proof of this third claim should be "simplified" by appealing to the result that the generalized inverse of the product of two matrices equals the product of the generalized inverse of those matrices. This result is, of course, false—see, e.g., section 3.1.1 of [60]—and so we need a more refined analysis such as the one presented here.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Combining this with the Frobenius norm bound of (31), and noticing that our choice for r guarantees that $1 - ||U_{A,k}^T U_{A,k} - U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}||_2 \geq 1/2$, concludes the proof of the fourth claim.

This concludes the proof of the lemma. ∎

The next lemma provides an approximate matrix multiplication bound that is useful in the proof of Theorem 5. For this lemma, r depends linearly on k.

LEMMA 2. *Let $\epsilon \in (0,1]$. If the sampling probabilities satisfy* (20) *and if $r \geq 400k/\beta\epsilon^2$, then with probability at least* 0.9*:*

$$\left\| U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F \leq \frac{\epsilon}{2} \left\| U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F .$$

Proof. First, note that since $U_{A,k}$ is an orthogonal matrix and since $U_{A,k}^T U_{A,k}^\perp = 0$, we have that

$$\begin{aligned} \left\| U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F &= \left\| U_{A,k} U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F \\ &= \left\| U_{A,k} U_{A,k}^T U_{A,k}^\perp {U_{A,k}^\perp}^T B - U_{A,k} U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F . \end{aligned} \tag{35}$$

Since $|(U_{A,k} U_{A,k}^T)_{(i)}|_2 = |(U_{A,k}^T)_{(i)}|_2$, the sampling probabilities (20) satisfy (45), where (45) will appear in Appendix A.2, and thus are appropriate for bounding the right-hand side of (35). Thus, it follows from Markov's inequality and Theorem 6 that with probability at least 0.9:

$$\begin{aligned} \left\| U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F &\leq 10\, \mathbf{E} \left[\left\| U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F \right] \\ &\leq \frac{10}{\sqrt{\beta r}} \left\| U_{A,k} U_{A,k}^T \right\|_F \left\| U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F . \end{aligned}$$

The lemma follows by the choice of r and since $||U_{A,k} U_{A,k}^T||_F = \sqrt{\rho} \leq \sqrt{k}$. ∎

The final lemma of this subsection relates the norm of the $m \times p$ matrix $U_{A,k}^\perp {U_{A,k}^\perp}^T B$ to the norm of the $r \times p$ matrix $\mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B$, i.e., the row sampled and rescaled version of the original $m \times p$ matrix. For this lemma, r is independent of k.

LEMMA 3. *With probability at least* 0.9*:*

$$\left\| \mathcal{S} U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F \leq 10 \left\| U_{A,k}^\perp {U_{A,k}^\perp}^T B \right\|_F .$$

Proof. Let $Q = U_{A,k}^\perp {U_{A,k}^\perp}^T B$, and let $j_1, j_2, \ldots, j_r$ be the r rows of Q that were included in $\mathcal{S}Q = DS^T Q$. Clearly,

$$\mathbf{E} \left[\left\| DS^T Q \right\|_F^2 \right] = \mathbf{E} \left[\sum_{t=1}^r \left| Q_{(j_t)} \right|_2^2 \right] = \sum_{t=1}^r \mathbf{E} \left[\left| Q_{(j_t)} \right|_2^2 \right] = \sum_{t=1}^r \sum_{j=1}^n p_j \frac{\left| Q_{(j)} \right|_2^2}{r p_j} = \left\| Q \right\|_F^2 , \tag{36}$$

where the penultimate equality follows by evaluating the expectation. The lemma follows by applying Markov's inequality and taking the square root of both sides of the resulting inequality. ∎

6.3.2. Proof of (21). In this subsection, we will bound $B - A_k \tilde{X}_{opt}$, thus proving (21). For the moment, let us assume that $r = 400k^2/\beta\epsilon^2$, in which case the assumption on r is satisfied for each of Lemma 1, Lemma 2, and Lemma 3. Thus, the claims of

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

all three lemmas hold simultaneously with probability at least $1 - 3(0.1) \geq 0.7$, and so let us condition on this event.

First, we have that

$$\begin{aligned} B - A_k\tilde{X}_{opt} &= B - A_k \left(\mathcal{S}A_k\right)^+ \mathcal{S}B \\ &= B - U_{A,k} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}B \qquad (37) \\ &= B - U_{A,k} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k}U_{A,k}^T B - U_{A,k} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B \qquad (38) \\ &= U_{A,k}^\perp {U_{A,k}^\perp}^T B - U_{A,k} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B. \qquad (39) \end{aligned}$$

Equation (37) follows from (27) of Lemma 1, (38) follows by inserting $U_{A,k}U_{A,k}^T + U_{A,k}^\perp {U_{A,k}^\perp}^T = I_n$, and (39) follows since $\left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k} = I_\rho$ by Lemma 1. We emphasize that $\left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k} = V_{\mathcal{S}U_{A,k}} V_{\mathcal{S}U_{A,k}}^T = I_\rho$ does not hold for general sampling methods, but it does hold in this case since $\tilde{\rho} = \rho$, which follows from Lemma 1.

By taking the Frobenius norm of both sides of (39), by using the triangle inequality, and recalling that $\Omega = \left(\mathcal{S}U_{A,k}\right)^+ - \left(\mathcal{S}U_{A,k}\right)^T$, we have that

$$\begin{aligned} \left\|B - A_k\tilde{X}_{opt}\right\|_F &\leq \left\|U_{A,k}^\perp {U_{A,k}^\perp}^T B\right\|_F + \left\|U_{A,k} \left(\mathcal{S}U_{A,k}\right)^T \mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B\right\|_F \\ &\quad + \left\|U_{A,k}\Omega\mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B\right\|_F \\ &\leq \left\|U_{A,k}^\perp {U_{A,k}^\perp}^T B\right\|_F + \left\|U_{A,k}^T \mathcal{S}^T \mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B\right\|_F \qquad (40) \\ &\quad + \|\Omega\|_2 \left\|\mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B\right\|_F, \end{aligned}$$

where (40) follows by submultiplicativity and since $U_{A,k}$ has orthogonal columns. By combining (40) with the bounds provided by Lemma 1 through Lemma 3, it follows that

$$\begin{aligned} \left\|B - A_k\tilde{X}_{opt}\right\|_F &\leq (1 + \epsilon/2 + 10\epsilon/\sqrt{2})\mathcal{Z} \\ &\leq (1 + 8\epsilon)\mathcal{Z}. \end{aligned}$$

Equation (21) follows by setting $\epsilon' = \epsilon/8$ and using the value of r assumed by the theorem.

6.3.3. Proof of (22). In this subsection, we will provide a bound for $||\tilde{X}_{opt} - X_{opt}||_F$ in terms of $\mathcal{Z}$, thus proving (22). For the moment, let us assume that $r = 400k^2/\beta\epsilon^2$, in which case the assumption on r is satisfied for each of Lemma 1, Lemma 2, and Lemma 3. Thus, the claims of all three lemmas hold simultaneously with probability at least $1 - 3(0.1) \geq 0.7$, and so let us condition on this event.

Since $U_{A,k}U_{A,k}^T + U_{A,k}^\perp {U_{A,k}^\perp}^T = I_n$ and $\left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k} = I_\rho$, we have that

$$\begin{aligned} X_{opt} - \tilde{X}_{opt} &= A_k^+ B - \left(\mathcal{S}A_k\right)^+ \mathcal{S}B \\ &= V_{A,k}\Sigma_{A,k}^{-1}U_{A,k}^T B - V_{A,k}\Sigma_{A,k}^{-1} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}B \\ &= V_{A,k}\Sigma_{A,k}^{-1}U_{A,k}^T B - V_{A,k}\Sigma_{A,k}^{-1} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k}U_{A,k}^T B \\ &\quad - V_{A,k}\Sigma_{A,k}^{-1} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B \\ &= -V_{A,k}\Sigma_{A,k}^{-1} \left(\mathcal{S}U_{A,k}\right)^+ \mathcal{S}U_{A,k}^\perp {U_{A,k}^\perp}^T B. \end{aligned}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

Thus, it follows that

$$\begin{aligned}
\left\|X_{opt} - \tilde{X}_{opt}\right\|_F &= \left\|V_{A,k}\Sigma_{A,k}^{-1}\left(\mathcal{S}U_{A,k}\right)^{+}\mathcal{S}U_{A,k}^{\perp}{U_{A,k}^{\perp}}^{T}B\right\|_F \\
&= \left\|\Sigma_{A,k}^{-1}\left(\left(\mathcal{S}U_{A,k}\right)^{T}+\Omega\right)\mathcal{S}U_{A,k}^{\perp}{U_{A,k}^{\perp}}^{T}B\right\|_F \\
&\leq \frac{1}{\sigma_{min}(A_k)}\left\|\left(\mathcal{S}U_{A,k}\right)^{T}\mathcal{S}U_{A,k}^{\perp}{U_{A,k}^{\perp}}^{T}B\right\|_F \\
&\quad + \frac{1}{\sigma_{min}(A_k)}\left\|\Omega\mathcal{S}U_{A,k}^{\perp}{U_{A,k}^{\perp}}^{T}B\right\|_F \\
&\leq \frac{1}{\sigma_{min}(A_k)}\left\|U_{A,k}^{T}\mathcal{S}^{T}\mathcal{S}U_{A,k}^{\perp}{U_{A,k}^{\perp}}^{T}B\right\|_F \qquad (41)\\
&\quad + \frac{1}{\sigma_{min}(A_k)}\left\|\Omega\right\|_2\left\|\mathcal{S}U_{A,k}^{\perp}{U_{A,k}^{\perp}}^{T}B\right\|_F.
\end{aligned}$$

By combining (41) with Lemmas 1, 2, and 3, it follows that

$$\begin{aligned}
\left\|\tilde{X}_{opt} - X_{opt}\right\|_F &\leq \sigma_{\min}^{-1}(A_k)\left(\epsilon/2 + 10\epsilon/\sqrt{2}\right)\mathcal{Z} \\
&\leq \frac{8\epsilon}{\sigma_{\min}(A_k)}\mathcal{Z}.
\end{aligned}$$

Equation (22) follows by setting $\epsilon' = \epsilon/8$ and using the value of r assumed by the theorem.

6.3.4. Proof of (23). The error bound provided by (22) could be quite weak, since $\min_{X\in\mathbb{R}^{n\times p}}\left\|B - A_kX\right\|_F$ could be quite close or even equal to $\left\|B\right\|_F$, if B has most or all of its "weight" outside of the column space of A_k. Under a slightly stronger assumption, we will provide a bound $||\tilde{X}_{opt} - X_{opt}||_F$ in terms of $\left\|X_{opt}\right\|_F$, thus proving (23).

If we make the additional assumption that a *constant fraction* of the "weight" of B lies in the subspace spanned by the columns of A_k, then it follows that

$$\begin{aligned}
\mathcal{Z}^2 &= \left(\min_{X\in\mathbb{R}^{n\times p}}\left\|B - A_kX\right\|_F\right)^2 \\
&= \left\|U_{A,k}^{\perp}{U_{A,k}^{\perp}}^{T}B\right\|_F^2 \\
&= \left\|B\right\|_F^2 - \left\|U_{A,k}U_{A,k}^{T}B\right\|_F^2 \\
&\leq \left(\gamma^{-2}-1\right)\left\|U_{A,k}U_{A,k}^{T}B\right\|_F^2. \qquad (42)
\end{aligned}$$

In order to relate $||U_{A,k}U_{A,k}^{T}B||_F$ and thus $\mathcal{Z}$ to $\left\|X_{opt}\right\|_F$ note that

$$\begin{aligned}
\left\|X_{opt}\right\|_F &= \left\|V_{A,k}\Sigma_{A,k}^{-1}U_{A,k}^{T}B\right\|_F \\
&= \left\|\Sigma_{A,k}^{-1}U_{A,k}^{T}B\right\|_F \\
&\geq \sigma_{\min}(\Sigma_{A,k}^{-1})\left\|U_{A,k}^{T}B\right\|_F \\
&= \frac{\left\|U_{A,k}U_{A,k}^{T}B\right\|_F}{\sigma_{\max}(A_k)}. \qquad (43)
\end{aligned}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

By combining (22) with (42) and (43), we get

$$\begin{aligned}\left\|\tilde{X}_{opt} - X_{opt}\right\|_F &\leq \frac{\epsilon}{\sigma_{\min}(A_k)}\mathcal{Z} \\ &\leq \frac{\epsilon}{\sigma_{\min}(A_k)}\sqrt{\gamma^{-2}-1}\left\|U_{A,k}U_{A,k}^T B\right\|_F \\ &\leq \epsilon\frac{\sigma_{\max}(A_k)}{\sigma_{\min}(A_k)}\sqrt{\gamma^{-2}-1}\left\|X_{opt}\right\|_F,\end{aligned}$$

which establishes (23).

6.3.5. Modifications to the proof with alternate row sampling procedure. If, in Algorithm 3, the rows are sampled with the EXPECTED(c) algorithm, then the proof of the claims of Theorem 5 is analogous to the proof described in the four previous subsections, with the following major exception. The claims of Lemma 1 hold if $r = O(k \log k/\beta\epsilon^2)$ rows are chosen with the EXPECTED(c) algorithm. To see this, recall that to bound the first claim of Lemma 1, we must bound the spectral norm $\|U_{A,k}^T U_{A,k} - U_{A,k}^T \mathcal{S}^T \mathcal{S} U_{A,k}\|_2$ in (29). If the sampling is performed with the EXACTLY(c) algorithm, then this is bounded in (30) by the corresponding Frobenius norm, which is then bounded with Theorem 6. On the other hand, if the sampling is performed with the EXPECTED(c) algorithm, then we can bound (29) directly with the spectral norm bound provided by Theorem 7.

Since the remaining claims of Lemma 1 follow from the first, they are also valid if $r = O(k \log k/\beta\epsilon^2)$ rows are chosen with the EXPECTED(c) algorithm. Lemma 2 still follows if $r = 400k/\beta\epsilon^2$, by using the Frobenius norm bound of Theorem 7, and Lemma 3 also follows immediately. The proofs of (21), (22), and (23) are identical, and thus Theorem 5, under the assumption that the the rows are chosen with the EXPECTED(c) algorithm, follows.

7. Empirical evaluation. Although this is a theoretical paper, it is motivated by applications, and thus one might wonder about the empirical applicability of our methods. For example, if we want to do as well as the best rank $k = 100$ (respectively, $k = 10$) approximation, with relative error bound $\epsilon = 0.1$, then our main theorem samples 3.2 billion (respectively, 32 million) columns of the matrix A using the EXACTLY(c) algorithm. Of course, our main theorem states that it in order to obtain our strong provable worst-case relative-error guarantees it *suffices* to choose that many columns. But, it would be a source of concern if anything like that number of columns is needed in "real" scientific and internet data applications.

In this section, we provide an empirical evaluation of the performance of our two main sampling procedures both for CX and *CUR* decompositions. In particular, we will evaluate how well the proposed column/row selection strategies perform at capturing the Frobenius norm for matrices derived from DNA SNP analysis, recommendation system analysis, and term-document analysis. By applying our algorithms to data sets drawn from these three diverse domains of modern data analysis, we will demonstrate that we can obtain very good Frobenius norm reconstruction by sampling a number of columns and/or rows that equals a small constant, e.g., 2 or 3 or 4 (as opposed to, e.g., a million or a billion), times the rank parameter k.

7.1. Details of our empirical evaluation. The empirical evaluation of our CX and *CUR* matrix decompositions has been performed using the following two types of column/row selection methods:

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

- "Subspace sampling" (with replacement) using the EXACTLY(c) algorithm; and
- "Subspace sampling" (without replacement) using the EXPECTED(c) algorithm.

In addition, the empirical evaluation has been performed on the following three data sets:

- Matrices derived from the DNA SNP HapMap data [15, 52]—see section 7.2;
- A matrix derived from the Jester recommendation system corpus [35, 48]—see section 7.3; and
- A matrix derived from the Reuters term-document corpus [46, 16]—see section 7.4.

We have chosen these three data sets on which to evaluate the empirical applicability of our algorithms for three reasons: first, these three application domains are representative of a wide range of areas of modern scientific and internet data analysis; second, these matrices are all approximately (to a greater or lesser extent) low-rank, and they are all data for which spectral methods such as low-rank approximations have been successfully applied; and third, we have already (with collaboraters from these application areas) applied our algorithms to these data sets [48, 52, 16]. In these data application papers [48, 52, 16], we have shown that our main CX and *CUR* decomposition algorithms (either the algorithms for which we have provable performance guarantees and/or greedy variants of these basic algorithms) perform well on tasks such as classification, denoising, reconstruction, prediction, and clustering—tasks that are of more immediate interest to data practitioners than simply capturing the norm of the data matrix.

In this section, however, we we will restrict ourselves to an empirical evaluation of our two main theorems. To do so, we will fix a rank parameter k, and we will present plots of the Frobenius norm error (normalized by $||A - A_k||_F$), as a function of the number of samples chosen. For example, we will consider $\Theta_1 \equiv ||A - CC^+A||_F/||A - A_k||_F$, where A_k is the best rank-k approximation to the matrix A, as a function of the number c of columns chosen. This ratio corresponds to the quantity that is bounded by $1 + \epsilon$ in Theorem 3. For $c = k$, this quantity will be no less than 1; of course, if we choose $c > k$ columns, then this ratio may be less than 1. Following the remark after Theorem 3, we will also consider $\Theta_2 \equiv ||A - CC^+A_k||_F/||A - A_k||_F$. This ensures that the approximation has a rank no greater than k (which is of interest in certain applications), and thus the plotted ratio will clearly be no less than 1, for every value of c. We will also consider $\Theta_3 \equiv ||A - CUR||_F/||A - A_k||_F$, which corresponds to the quantity that is bounded by $1 + \epsilon$ in Theorem 4.

Two technical points should be noted about these plots in the upcoming subsections. First, we ran our CX or *CUR* decomposition algorithm several—e.g., three or five, depending on the size of the data being plotted—times (corresponding, say, to multiple runs to boost the δ failure probability), and the minimum value over these repetitions was returned; this was repeated several times, and the average of those values is plotted. Second, for the plots of $||A - CUR||_F/||A - A_k||_F$, the number of rows selected is set to be twice the corresponding number of columns selected; optimizing over this would lead to marginally better performance than that presented.

7.2. DNA SNP HapMap data. Our first dataset comes from the field of human genetics. The HapMap project, a continuation of the Human Genome project, aims to map the loci in the human genome that differ among individuals [15]. The HapMap project focuses on the so-called SNPs, which are a very common type of

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

variation in the genome (nearly 10^7 such loci have been identified in the human genome). Significant motivation exists in the genetics community for minimizing the number of SNPs that must be assayed, and in [52], we demonstrated how *CUR*-type methods may be used to efficiently reconstruct unassayed SNPs from a small number of assayed SNPs.

Both in [52] and here, we consider two regions of the genome known as HOXB and 17q25. Three populations were studied for each region: Yoruban, a sub-Saharan African population; a European population; and a joint Japanese/Chinese population. Each population had 90 individuals, each corresponding to a row of the input matrix. Columns of each matrix correspond to SNPs within the HOXB or 17q25 regions. The genotypic data were encoded appropriately in order to be converted to numeric data in the form of matrices. (Careful preprocessing was done to remove fixed SNPs, as well as SNPs with too many missing entries, etc.) The HapMap project provided data on 370 SNPs in 17q25 and 571 SNP in HOXB [15]. Thus, for example, our matrix for the Yoruban population in HOXB is a 90×571 matrix, whose entries are in the set $\{-1, 0, +1\}$.[3] The other data matrices are of similar (not extremely large) size. See [52] and references therein for details.

Data not presented indicate that for all three populations and for both genomic regions, the data possess a great deal of linear structure. For example, in the 17q25 matrices, one needs 9, 9, and 7 singular vectors to capture 80% of the Frobenius norm for the Yoruban, European, and the Japanese/Chinese populations, respectively; and one needs 18, 16, and 13 singular vectors, respectively, to capture 90%. The matrices for the HOXB region of the genome are even more redundant; one needs only 7, 6, and 4 singular vectors, respectively, to capture 80% of the Frobenius norm.

In Figure 1, data are presented for the Yoruban HOXB data matrix. Each of the six subfigures presents a plot of the Frobenius norm error as a function of the number c of samples chosen. In particular, for two values of the rank parameter, i.e., $k = 5$ and $k = 10$, the ratio $\Theta_i = ||A - A'||_F / ||A - A_k||_F$ is plotted, where $A' = CC^+A$ for $i = 1$; $A' = CC^+A_k$ for $i = 2$; and $A' = CUR$ for $i = 3$. Clearly, in all these cases, only modest oversampling is needed to capture "nearly all" of the dominant part of the spectrum of the data matrix. For example, for $k = 5$: if $c = 5$, then $\Theta_1 = 1.12$; if $c \geq 6$, then $\Theta_1 < 1.1$; and if $c \geq 9$, then $\Theta_1 < 1.0$. Similarly, for $k = 10$: if $c = 10$, then $\Theta_1 = 1.22$; if $c \geq 15$, then $\Theta_1 < 1.1$; and if $c \geq 18$, then $\Theta_1 < 1.0$. Similar results hold if the projection onto the span of the columns is regularized through a rank-k space and also if rows are chosen after the columns. For example, for $k = 10$: if $c \geq 16$, then $\Theta_2 < 1.2$ and if $c \gtrsim 30$, then $\Theta_2 < 1.1$. Similarly, even though the computations for Θ_3 are slightly worse and somewhat noisier due to the second level of sampling (columns and then rows), the results still show that only modest oversampling (of c relative to k) is needed. For example, if $k = 10$, then $\Theta_3 < 1.1$ if $c \geq 20$ or $c \geq 28$, depending on precisely how the columns are chosen. Interestingly, in this last case, not only are the plots noisier, but the EXPECTED(c) algorithm and the EXACTLY(c) algorithm seem to lead to (slightly) different results as a function of c.

Qualitatively similar results are seen for the other populations and the other genomic regions. For example, in Figure 2, data are presented for the European

[3]The encoding should be interpreted as follows: each SNP consists of two alleles (nucleotide bases); these bases are the same for all humans. Say that these bases are A and G. Then a value of +1 corresponds to an individual whose genotype (pair of alleles) is AA, a value of 0 corresponds to an individual whose genotype is AG or GA, and a value of −1 corresponds to an individual whose genotype is GG.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

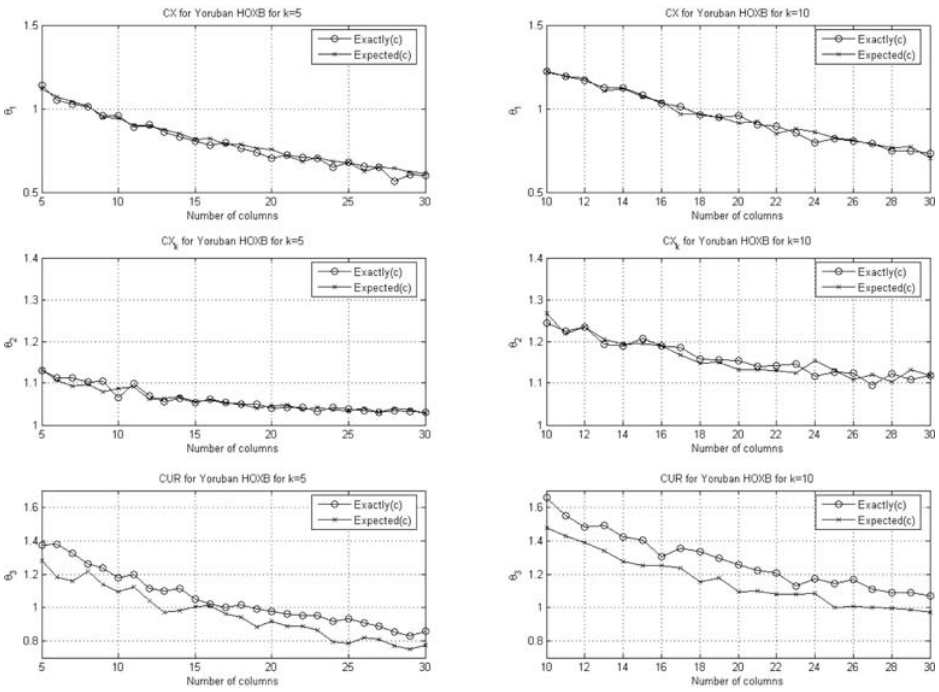

FIG. 1. *Reconstruction error for the Yoruban population in the HOXB region of the genome. Shown are* Θ_1, Θ_2, *and* Θ_3 *(as defined in the text) for two values of the rank parameter* k. *The X-axis corresponds to the number of columns sampled with the* EXACTLY(c) *algorithm or the* EXPECTED(c) *algorithm.*

population for both the HOXB and the 17q25 regions of the genome for the value of the rank parameter $k = 10$. For the HOXB region, $\Theta_1 = 1.36$ if $c = 10$ (this is higher than for the corresponding Yoruban data), $\Theta_1 < 1.0$ if $c \gtrsim 17$ (this is similar to the corresponding Yoruban data), and $\Theta_1 = 0.62$ if $c = 30$ (this is less than the Yoruban data). Similar trends are seen for Θ_2 and Θ_3 and also for the 17q25 region. In all cases, only very modest oversampling is needed for accurate Frobenius norm reconstruction. Data not presented indicate that the data for the joint Japanese/Chinese population is quite similar to or slightly better than those results presented.

7.3. Recommendation system jester data. Our second dataset comes from the field of recommendation system analysis, in which one is typically interested in making purchase recommendations to a user at an electronic commerce web site [35]. Collaborative methods (as opposed to content-based or hybrid) involve recommending to the user items that people with similar tastes or preferences liked in the past. Many collaborative filtering algorithms represent a user as an n dimensional vector, where n is the number of distinct products, and where the components of the vector are a measure of the rating provided by that user for that product. Thus, for a set of m users, the user-product ratings matrix is an $m \times n$ matrix A, where A_{ij} is the rating by user i for product j (or is null if the rating is not provided).

The so-called Jester joke dataset is a commonly used benchmark for recommendation system research and development [35]. In [48], we applied a *CUR* decomposition on this data to the problem of reconstructing missing entries and making accurate

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

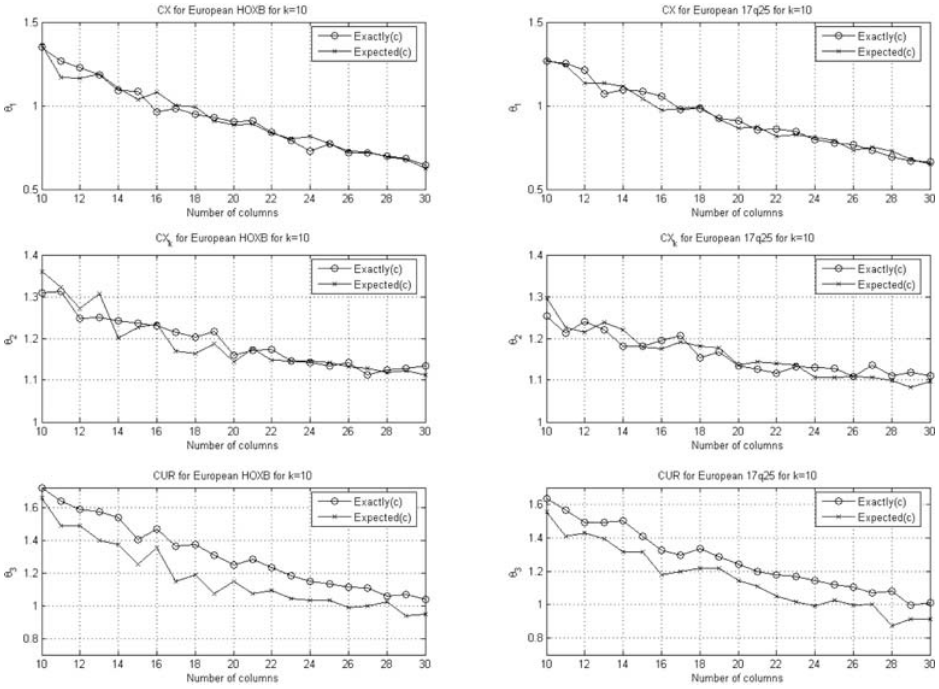

FIG. 2. *Reconstruction error for the European population in both the HOXB and 17q25 regions of the genome. Shown are* Θ_1, Θ_2, *and* Θ_3 *(as defined in the text) for* $k = 10$. *The X-axis corresponds to the number of columns sampled with the* EXACTLY(c) *algorithm or the* EXPECTED(c) *algorithm.*

recommendations. Here, we consider the $m = 14{,}116$ (out of ca. 73,000) users who rated all of the $n = 100$ products (i.e., jokes) in the Jester data. The entries in this $14{,}116 \times 100$ matrix A are real numbers between -10 and $+10$ that represent the user's rating of a product.

Figure 3 presents the empirical results for the Jester recommendation system data. The rank of the $14{,}116 \times 100$ matrix is 100, and although only seven singular vectors are needed to capture 50% of the Frobenius norm, 50 are needed to capture 80%, and 73 are needed to capture 90%. Thus, the spectrum and shape of this matrix (this matrix is very rectangular) are very different from that of the matrices of the previous subsection.

Figure 3 presents reconstruction error results for selecting columns (i.e., products or jokes), for selecting rows (i.e., users), and for selecting both columns and rows simultaneously. For example, when selecting columns from A, if $k = 15$, then $\Theta_1 = 1.14$ if $c = 15$, $\Theta_1 \leq 1$ if $c \gtrsim 29$, and $\Theta_1 = 0.99$ when $c = 30$. Although the matrix is very rectangular, quantitatively very similar results are obtained for the analogue of Θ_1 (called Θ_1^R in the figure) if rows are sampled (or, equivalently if columns are sampled from A^T). Thus, when our main CX decomposition is applied to either A or to A^T, a small number of columns (products) or rows (users), capture most of the Frobenius norm of A that is captured by the best rank k approximation to A. A similar result holds for the simultaneously choosing columns and rows of A (both users and products), and applying our CUR approximation algorithm. As with the data of the previous subsection, the data for Θ_3 are much noisier when both columns

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

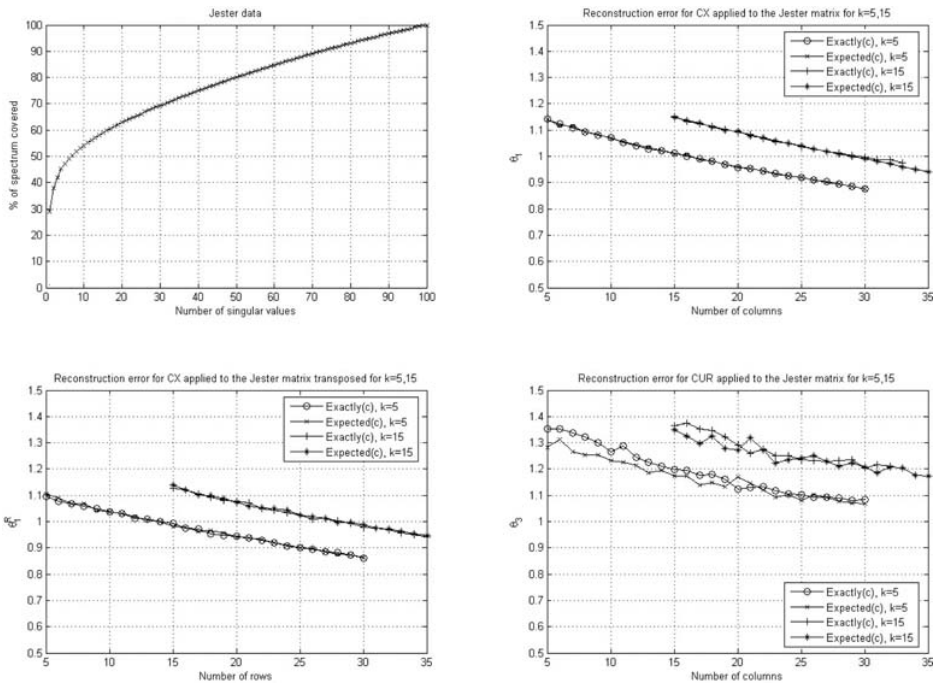

FIG. 3. *Empirical results for the Jester recommendation system data. Shown are: the percentage of the Frobenius norm captured as a function of the number of singular components;* Θ_1 *for sampling columns from* A *for* $k = 5$ *and* $k = 15$*; the analogue of* Θ_1 *for selecting rows from* A *(i.e.,* Θ_1 *for sampling columns from* A^T*); and* Θ_3 *for selecting columns and then rows for* $k = 5$ *and* $k = 15$*.*

and rows are chosen, but even in this case $\Theta_3 \leq 1.1$ for $k = 5$ if $c \gtrsim 25$ and $\Theta_3 \leq 1.2$ for $k = 15$ if $c \gtrsim 30$. In all these cases, data not presented indicate that qualitatively similar (but shifted) results are obtained for higher values of the rank parameter k.

7.4. Term-document Reuters data. Our third data set comes from the field of text categorization and information retrieval. In these applications, documents are often represented as a so-called "bag of words" and a vector space model is used. In 2000, Reuters Ltd made available a large collection of Reuters News stories for use in research and development of natural language processing, information retrieval, and machine learning systems. This corpus, known as "Reuters Corpus, Volume 1" or RCV1, is significantly larger (it contains over 800,000 news items from 1996-97) than the older, well-known Reuters-21578 collection, which has been heavily used in the text classification community [46]. In [16], we considered the problem of feature selection for improved classification, and we compared a CX-like column selection procedure to several traditional methods. The data come with class labels and possess a hierarchical class structure (which we used in [16]) which we ignored here. Here, we used the *ltc*-normalized term-document matrix and the training data from the one test-train split provided by Lewis, Yang, Rose, and Li [46]. Thus, the Reuters matrix we considered here is a (very sparse) 47,236 × 23,149 matrix whose elements are real numbers between 0 and 1 that represent a normalized frequency.

Figure 4 presents the empirical results for the Reuters term-document data. Note that this data is not only much larger than the data from the previous two subsec-

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

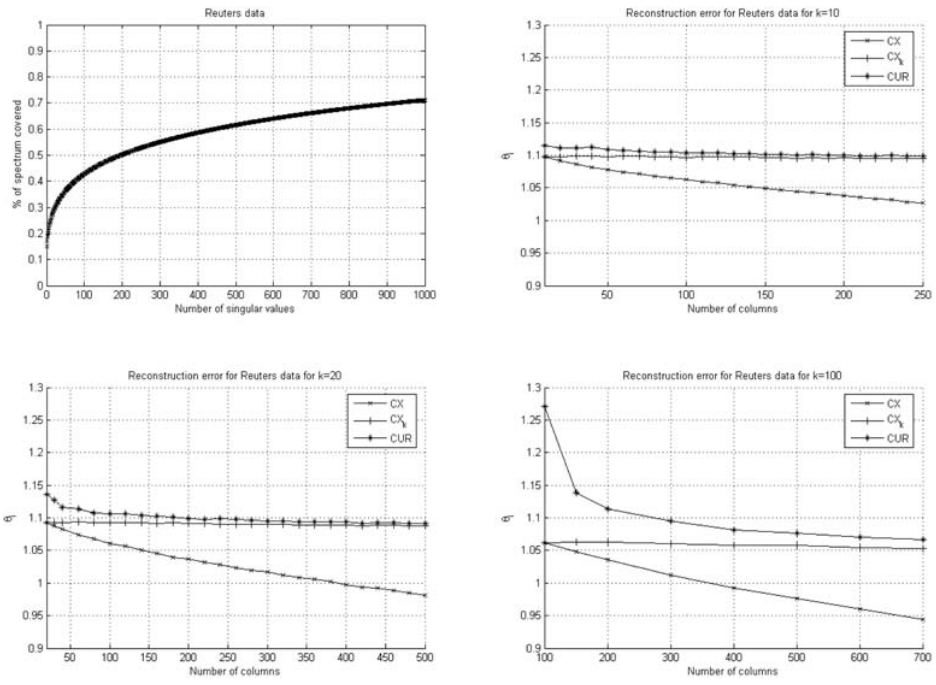

FIG. 4. *Empirical results for the Reuters term-document data. Shown are the percentage of the Frobenius norm captured as a function of the number of singular components;* Θ_1, Θ_2, *and* Θ_3 *as a function of the number of sampled columns and/or rows for three different values of the rank parameter* k.

tions, it is also less well-approximated by a low rank matrix. Less than 50% of the Frobenius norm is captured by the first $k = 100$ singular components, and less than 80% is captured by the first $k = 1500$ singular components. (Nevertheless, spectral methods have frequently been applied to this data.) The matrix is very sparse, and performing computations is expensive in terms of space and time (due to multiple randomized trials and since the dense matrices of singular vectors are large) if the rank parameter k is chosen to be more than a few hundred. Thus, to demonstrate the empirical applicability of our main algorithms, we considered several smaller values of k. Here, we report results for $k = 10$ and $c = 10$ to 250; for $k = 20$ and $c = 20$ to 500; and for $k = 100$ and $c = 100$ to 700. Note that we report results only for columns chosen with the EXPECTED(c) algorithm; initial unreported computations on several smaller systems indicate that very similar results will be obtained with the EXACTLY(c) algorithm.

In all of these cases, and for all values of Θ_1, Θ_2, and Θ_3, only modest oversampling leads to fairly small reconstruction error. The worst data point reported was for $\Theta_3 = 1.272$ for $k = 100$ and $c = 100$, and even in that case $\Theta_3 < 1.1$ for $c \geq 300$. Interestingly, all the curves tend to decrease somewhat more slowly (as a function of oversampling c, relative to k) than the corresponding curves in the previous subsections do. Note that Θ_1 does not decrease below 1.0 for $k = 10$ until after $c = 500$; for $k = 20$, it drops below 1.0 by $c = 400$, and for $k = 100$ (which obviously captures the largest fraction of the Frobenius norm) it drops below 1.0 at $c \approx 350$. Thus, this phenomenon is likely related to the degree to which the chosen value for

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

the rank parameter k captures a reasonable fraction of the norm of the original matrix. Nevertheless, in all cases, we can achieve $\Theta_i < 1.1$ with only a modest degree of oversampling c relative to k.

8. Conclusion. We have presented and analyzed randomized algorithms for computing low-rank matrix approximations that are explicitly expressed in terms of a small number of columns and/or rows of the input matrix. These algorithm achieve relative-error guarantees, whereas previous algorithms for these problems achieved only additive-error guarantees. These algorithms randomly sample in a novel manner that we call "subspace sampling," and their analysis amounts to approximating a generalized ℓ_2 regression problem by random sampling. As described in section 1.1 and in [52, 48], such low-rank matrix approximations have numerous applications for the improved analysis of data.

We conclude with several open problems:

- To what extent do the results of this paper generalize to other matrix norms?
- What hardness results can be established for the optimal choice of columns and/or rows?
- Does there exist a deterministic approximation algorithm for either of the problems we consider?
- Does there exist an efficient deterministic algorithm to choose columns and/or rows that exactly or approximately optimize the maximum volume of the induced parallelepiped? (As pointed out to us by an anonymous reviewer, [40, 61] provide such procedures; it would be interesting to see if bounds of the form we prove can be established for the algorithms of [40, 61]).
- Can we formulate a simple condition that we can check after we have sampled the columns and/or rows to determine whether we have achieved a $1 + \epsilon$ approximation with that sample?
- Can we obtain similar algorithms and comparable bounds for formulations of these problems that include regularization and/or conditioning?
- What heuristic variants of these algorithms are most appropriate in different application domains?
- Are the algorithms presented in this paper numerically stable?

Appendix A. Approximating matrix multiplication. In this section, we describe two complementary procedures for randomly sampling (and rescaling) columns and/or rows from an input matrix. Then, we describe an algorithm for approximating the product of two matrices by randomly sampling columns and rows from the input matrices using one of the two sampling procedures.

A.1. Sampling columns and rows from matrices. We describe two simple algorithms for randomly sampling a set of columns from an input matrix. Each algorithm takes as input an $m \times n$ matrix A and a probability distribution $\{p_i\}_{i=1}^n$, and each constructs a matrix C consisting of a rescaled copy of a small number of columns from A. Clearly, each algorithm can be modified to sample rows from a matrix. The first algorithm is the EXACTLY(c) algorithm, which is described in Algorithm 4 using the sampling matrix formalism described in section 2. In this algorithm, c columns *exactly* of A are chosen in c i.i.d. trials, where in each trial the ith column of A is picked with probability p_i. Note that because the sampling is performed with replacement, a single column of A may be included in C more than once. The second algorithm is the EXPECTED(c) algorithm, which is described in Algorithm 5, also using the sampling matrix formalism described in section 2. In this algorithm, at most c columns *in expectation* of A are chosen by including the

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

ith column of A in C with probability $\tilde{p}_i = \min\{1, cp_i\}$. Note that the exact value of the number of columns returned is not known before the execution of this second algorithm; we do not perform an analysis of this random variable.

Data : $A \in \mathbb{R}^{m\times n}$, $p_i \geq 0, i \in [n]$ s.t. $\sum_{i\in[n]} p_i = 1$, positive integer $c \leq n$.
Result : Sampling matrix S, rescaling matrix D, and sampled and rescaled columns C.
Initialize S and D to the all zeros matrices.
for $t = 1, \ldots, c$ **do**
  Pick $i_t \in [n]$, where $\mathbf{Pr}(i_t = i) = p_i$;
  $S_{i_t t} = 1$;
  $D_{tt} = 1/\sqrt{cp_{i_t}}$.
end
$C = ASD$.

Algorithm 4. The EXACTLY(c) algorithm to create S, D, and C.

Data : $A \in \mathbb{R}^{m\times n}$, $p_i \geq 0, i \in [n]$ s.t. $\sum_{i\in[n]} p_i = 1$, positive integer $c \leq n$.
Result : Sampling matrix S, rescaling matrix D, and sampled and rescaled columns C.
Initialize S and D to the all zeros matrices.
$t = 1$;
for $j = 1, \ldots, n$ **do**
  Pick j with probability $\min\{1, cp_j\}$;
  if j *is picked* **then**
    $S_{jt} = 1$;
    $D_{tt} = 1/\min\{1, \sqrt{cp_j}\}$;
    $t = t + 1$;
  end
end
$C = ASD$.

Algorithm 5. *The* EXPECTED(c) *algorithm to create S, D, and C.*

A.2. Approximate matrix multiplication algorithms. Algorithm 6 takes as input two matrices A and B, a number $c \leq n$, and a probability distribution $\{p_i\}_{i=1}^n$ over $[n]$. It returns as output two matrices C and R, where the columns of C are a small number of sampled and rescaled columns of A and where the rows of R are a small number of sampled and rescaled rows of B. The sampling and rescaling are performed by calling either the EXACTLY(c) algorithm or the EXPECTED(c) algorithm. When the EXACTLY(c) algorithm is used to choose column-row pairs in Algorithm 6, this is identical to the algorithm of [21]. In particular, note that *exactly* c column-row pairs are chosen, and a column-row pair could be included in the sample more than once. When the EXPECTED(c) algorithm is used to choose column-row pairs in Algorithm 6, this is a minor variation of the algorithm of [21]. In particular, the

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

main difference is that at most c column-row pairs *in expectation* are chosen, and no column-row pair is included in the sample more than once.

Data : $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{n\times p}$, $\{p_i\}_{i=1}^n$ such that $\sum_{i=1}^n p_i = 1$, $c \le n$.
Result : $C \in \mathbb{R}^{m\times c}$, $R \in \mathbb{R}^{c\times p}$.

- Form the matrix $C = ASD$ by sampling according to $\{p_i\}_{i=1}^n$ with the EXACTLY(c) algorithm or with the EXPECTED(c) algorithm;
- Form the matrix $R = DS^TB$ from the corresponding rows of B.

Algorithm 6. A fast Monte-Carlo algorithm for approximate matrix multiplication.

The next two theorems are our basic quality-of-approximation results for Algorithm 6. Each states that, under appropriate assumptions, $CR = ASDDS^TB \approx AB$. The most interesting of these assumptions is that the sampling probabilities used to randomly sample the columns of A and the corresponding rows of B are nonuniform and depend on the product of the Euclidean norms of the columns of A and/or the corresponding rows of B. For example, consider sampling probabilities $\{p_i\}_{i=1}^n$ such that

$$p_i \ge \beta \frac{\left|A^{(i)}\right|_2 \left|B_{(i)}\right|_2}{\sum_{j=1}^n \left|A^{(j)}\right|_2 \left|B_{(j)}\right|_2}, \tag{44}$$

for some $\beta \in (0,1]$. Sampling probabilities of the form (44) use information from the matrices A and B in a very particular manner. If $\beta = 1$, they are optimal for approximating AB by CR in a sense made precise in [21]. Alternatively, sampling probabilities $\{p_i\}_{i=1}^n$ such that

$$p_i \ge \beta \frac{\left|A^{(i)}\right|_2^2}{\|A\|_F^2}, \tag{45}$$

for some $\beta \in (0,1]$, are also of interest in approximating the product AB by CR if, e.g., only information about A is easily available.

The following theorem is our main quality-of-approximation result for approximating the product of two matrices with Algorithm 6, when column-row pairs are sampled using the EXACTLY(c) algorithm. Its proof (and the statement and proof of similar stronger results) may be found in [21].

THEOREM 6. *Suppose $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{n\times p}$, and $c \le n$. Construct C and R with Algorithm* 6*, using the* EXACTLY(c) *algorithm. If the sampling probabilities $\{p_i\}_{i=1}^n$ used by the algorithm are of the form* (44) *or* (45)*, then*

$$\mathbf{E}\left[\,\|AB - CR\|_F\right] \le \frac{1}{\sqrt{\beta c}} \|A\|_F \, \|B\|_F .$$

The following theorem is our main quality-of-approximation result for approximating the product of two matrices with Algorithm 6, when column-row pairs are sampled using the EXPECTED(c) algorithm. The Frobenius norm bound (46) is new, and the spectral norm bound (47) is due to Rudelson and Vershynin, who proved a similar result in a more general setting [54, 62, 56].

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

THEOREM 7. *Suppose $A \in \mathbb{R}^{m\times n}$, $B \in \mathbb{R}^{n\times p}$, and $c \le n$. Construct C and R with Algorithm 6, using the* EXPECTED(c) *algorithm. If the sampling probabilities $\{p_i\}_{i=1}^n$ used by the algorithm are of the form (44) or (45), then*

$$\mathbf{E}\left[\,\|AB - CR\|_F\right] \le \frac{1}{\sqrt{\beta c}}\,\|A\|_F\,\|B\|_F\,. \tag{46}$$

If, in addition, $B = A^T$, then

$$\mathbf{E}\left[\,\left\|AA^T - CC^T\right\|_2\right] \le O(1)\sqrt{\frac{\log c}{\beta c}}\,\|A\|_F\,\|A\|_2\,. \tag{47}$$

Proof. Equation (47) follows from the analysis of Rudelson and Vershynin, who considered spectral norm bounds on approximating the product of two matrices [54, 62, 56]. Note that they considered approximating the product AA^T by sampling with respect to probabilities of the form (45) with $\beta = 1$, but the analysis for general $\beta \in (0, 1]$ is analogous.

Next, we prove that for any set of probabilities $\{p_i\}_{i=1}^n$ the following holds:

$$\mathbf{E}\left[\,\|AB - CR\|_F^2\right] \le \frac{1}{c}\sum_{j=1}^{n}\frac{\left|A^{(j)}\right|_2^2\left|B_{(j)}\right|_2^2}{p_j}. \tag{48}$$

Equation (46) follows from (48) by using Jensen's inequality and using the form of the sampling probabilities (44) and (45).

To establish (48), recall that the sampling is performed with the EXPECTED(c) algorithm. Let I_j, $j \in [n]$ be the indicator variable that is set to 1 if the jth column of A and the jth row of B are sampled (with probability $\min\{1, cp_j\}$) and is set to 0 otherwise. Recall that if $I_j = 1$, we scale both the jth column of A and the jth row of B by $1/\sqrt{\min\{1, cp_j\}}$. Thus,

$$\begin{aligned}\|AB - CR\|_F^2 &= \left\|AB - ASDDS^TB\right\|_F^2 \\ &= \left\|\sum_{j=1}^{n}\left(1 - \frac{I_j}{\min\{1, cp_j\}}\right)A^{(j)}B_{(j)}\right\|_F^2.\end{aligned} \tag{49}$$

Clearly, if $\min\{1, cp_j\} = 1$, then $I_j = 1$ with probability 1, and $1 - I_j/\min\{1, cp_j\} = 0$. Thus, we can focus on the set of indices $\Lambda = \{j \in [n] : cp_j < 1\} \subseteq [n]$. By taking the expectation of both sides of (49), it follows that

$$\begin{aligned}\mathbf{E}\left[\,\|AB - CR\|_F^2\right] &= \mathbf{E}\left[\left\|\sum_{j\in\Lambda}\left(1 - \frac{I_j}{cp_j}\right)A^{(j)}B_{(j)}\right\|_F^2\right] \\ &= \mathbf{E}\left[\sum_{i_1=1}^{m}\sum_{i_2=1}^{p}\left(\sum_{j\in\Lambda}\left(1 - \frac{I_j}{cp_j}\right)A^{(j)}B_{(j)}\right)_{i_1i_2}^2\right] \\ &= \mathbf{E}\left[\sum_{i_1=1}^{m}\sum_{i_2=1}^{p}\left(\sum_{j\in\Lambda}\left(1 - \frac{I_j}{cp_j}\right)A_{i_1j}B_{ji_2}\right)^2\right].\end{aligned}$$

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

By multiplying out the right-hand side, it follows that

(50)

$$\mathbf{E}\left[\|AB-CR\|_F^2\right] = \mathbf{E}\left[\sum_{i_1=1}^{m}\sum_{i_2=1}^{p}\sum_{j_1\in\Lambda}\sum_{j_2\in\Lambda}\left(1-\frac{I_{j_1}}{cp_{j_1}}\right)\left(1-\frac{I_{j_2}}{cp_{j_2}}\right)A_{i_1j_1}B_{j_1i_2}A_{i_1j_2}B_{j_2i_2}\right]$$
$$= \sum_{i_1=1}^{m}\sum_{i_2=1}^{p}\sum_{j_1\in\Lambda}\sum_{j_2\in\Lambda}\mathbf{E}\left[\left(1-\frac{I_{j_1}}{cp_{j_1}}\right)\left(1-\frac{I_{j_2}}{cp_{j_2}}\right)\right]A_{i_1j_1}B_{j_1i_2}A_{i_1j_2}B_{j_2i_2}.$$

Notice that for $j\in[\Lambda]$, $\mathbf{E}\left[1-I_j/cp_j\right]=0$ and $\mathbf{E}\left[\left(1-I_j/cp_j\right)^2\right]=(1/cp_j)-1\le 1/cp_j$. Hence,

$$\mathbf{E}\left[\|AB-CR\|_F^2\right] = \sum_{i_1=1}^{m}\sum_{i_2=1}^{p}\sum_{j\in\Lambda}\mathbf{E}\left[\left(1-I_j/cp_j\right)^2\right]A_{i_1j}^2B_{ji_2}^2$$
$$\le \sum_{j\in\Lambda}\frac{1}{cp_j}\sum_{i_1=1}^{m}\sum_{i_2=1}^{p}A_{i_1j}^2B_{ji_2}^2 = \frac{1}{c}\sum_{j\in\Lambda}\frac{\left|A^{(j)}\right|_2^2\left|B_{(j)}\right|_2^2}{p_j}.$$

This concludes the proof of (48) and thus of the theorem. ∎

Acknowledgments. We would like to thank Sariel Har-Peled for writing up his results amidst travel in India [42], Amit Deshpande and Santosh Vempala for graciously providing a copy of [18], and two anonymous reviewers for useful comments.

REFERENCES

[1] D. Achlioptas and F. McSherry, *Fast computation of low rank matrix approximations*, J. ACM, 54 (2007).
[2] D. Achlioptas and F. McSherry, *Fast computation of low rank matrix approximations*, in Proceedings of the 33rd Annual ACM Symposium on Theory of Computing, 2001, pp. 611–6181.
[3] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, *Approximating extent measures of points*, J. ACM, 51 (2004), pp. 606–635.
[4] P. K. Agarwal, S. Har-Peled, and K. R. Varadarajan, *Geometric approximation via coresets survey*, in Current Trends in Combinatorial and Computational Geometry, E. Welzl, ed., Cambridge University Press, Cambridge, 2006.
[5] N. Ailon and B. Chazelle, *Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform*, in Proceedings of the 38th Annual ACM Symposium on Theory of Computing, 2006, pp. 557–563.
[6] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications*, Springer-Verlag, New York, 2003.
[7] 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, 2004, pp. 177–184.
[8] M. W. Berry, S. A. Pulatova, and G. W. Stewart, *Computing sparse reduced-rank approximations to sparse matrices*, ACM Trans. Math. Software, 31 (2005), pp. 252–269.
[9] R. Bhatia, *Matrix Analysis*, Springer-Verlag, New York, 1997.
[10] A. L. Blum and P. Langley, *Selection of relevant features and examples in machine learning*, Artif. Intell., 97 (1997), pp. 245–271.
[11] P. Businger and G. H. Golub, *Linear least squares solutions by Householder transformations*, Numer. Math., 7 (1965), pp. 269–276.
[12] T. F. Chan and P. C. Hansen, *Computing truncated singular value decomposition least squares solutions by rank revealing QR-factorizations*, SIAM J. Sci. Stat. Comput., 11 (1990), pp. 519–530.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

[13] T. F. Chan and P. C. Hansen, *Some applications of the rank revealing QR factorization*, SIAM J. Sci. Stat. Comput., 13 (1992), pp. 727–741.

[14] M. Charikar, V. Guruswami, R. Kumar, S. Rajagopalan, and A. Sahai, *Combinatorial feature selection problems*, in Proceedings of the 41st Annual IEEE Symposium on Foundations of Computer Science, 2000, pp. 631–642.

[15] The International HapMap Consortium, *A haplotype map of the human genome*, Nature, 437 (2005), pp. 1299–1320.

[16] A. Dasgupta, P. Drineas, B. Harb, V. Josifovski, and M. W. Mahoney, *Feature selection methods for text classification*, in Proceedings of the 13th Annual ACM SIGKDD Conference, 2007, pp. 230–239.

[17] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang, *Matrix approximation and projective clustering via volume sampling*, in Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, 2006, pp. 1117–1126.

[18] A. Deshpande and S. Vempala, *Adaptive sampling and fast low-rank matrix approximation*, Technical report TR06-042, Electronic Colloquium on Computational Complexity, 2006.

[19] 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, 1999, pp. 291–299.

[20] P. Drineas and R. Kannan, *Pass efficient algorithms for approximating large matrices*, in Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms, 2003, pp. 223–232.

[21] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* I: *Approximating matrix multiplication*, SIAM J. Comput., 36 (2006), pp. 132–157.

[22] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* II: *Computing a low-rank approximation to a matrix*, SIAM J. Comput., 36 (2006), pp. 158–183.

[23] P. Drineas, R. Kannan, and M. W. Mahoney, *Fast Monte Carlo algorithms for matrices* III: *Computing a compressed approximate matrix decomposition*, SIAM J. Comput., 36 (2006), pp. 184–206.

[24] P. Drineas and M. W. Mahoney, *Approximating a Gram matrix for improved kernel-based learning*, in Proceedings of the 18th Annual Conference on Learning Theory, 2005, pp. 323–337.

[25] P. Drineas and M. W. Mahoney, *On the Nyström method for approximating a Gram matrix for improved kernel-based learning*, J. Mach. Learn. Res., 6 (2005), pp. 2153–2175.

[26] P. Drineas and M. W. Mahoney, *A randomized algorithm for a tensor-based generalization of the Singular Value Decomposition*, Linear Algebra Appl., 420 (2007), pp. 553–571.

[27] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Polynomial time algorithm for column-row based relative-error low-rank matrix approximation*, manuscript, 2005.

[28] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Polynomial time algorithm for column-row based relative-error low-rank matrix approximation*, Technical report 2006-04, DIMACS, 2006.

[29] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Sampling algorithms for ℓ_2 regression and applications*, in Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms, 2006, pp. 1127–1136.

[30] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Subspace sampling and relative-error matrix approximation: Column-based methods*, in Proceedings of the 10th International Workshop on Randomization and Computation, 2006, pp. 316–326.

[31] P. Drineas, M. W. Mahoney, and S. Muthukrishnan, *Subspace sampling and relative-error matrix approximation: Column-row-based methods*, in Proceedings of the 14th Annual European Symposium on Algorithms, 2006, pp. 304–314.

[32] C. Fowlkes, S. Belongie, F. Chung, and J. Malik, *Spectral grouping using the Nyström method*, IEEE Trans. Pattern Anal. Mach. Intell., 26 (2004), pp. 214–225.

[33] 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, 1998, pp. 370–378.

[34] A. Frieze, R. Kannan, and S. Vempala, *Fast Monte-Carlo algorithms for finding low-rank approximations*, J. ACM, 51 (2004), pp. 1025–1041.

[35] K. Goldberg, T. Roeder, D. Gupta, and C. Perkins, *Eigentaste: A constant time collaborative filtering algorithm*, Inform. Retrieval, 4 (2001), pp. 133–151.

[36] G. Golub, *Numerical methods for solving linear least squares problems*, Numer. Math., 7 (1965), pp. 206–216.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.

[37] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1996.
[38] S. A. Goreinov and E. E. Tyrtyshnikov, *The maximum-volume concept in approximation by low-rank matrices*, Contemp. Math., 280 (2001), pp. 47–51.
[39] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, *A theory of pseudoskeleton approximations*, Linear Algebra Appl., 261 (1997), pp. 1–21.
[40] M. Gu and S. C. Eisenstat, *Efficient algorithms for computing a strong rank-revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.
[41] I. Guyon and A. Elisseeff, *An introduction to variable and feature selection*, J. Mach. Learn. Res., 3 (2003), pp. 1157–1182.
[42] S. Har-Peled, *Low rank matrix approximation in linear time*, manuscript, 2006.
[43] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, New York, 1985.
[44] F. G. Kuruvilla, P. J. Park, and S. L. Schreiber, *Vector algebra in the analysis of genome-wide expression data*, Genome Biol., 3 (2002), research0011.1–0011.11.
[45] S. Lafon, *Diffusion Maps and Geometric Harmonics*, Ph.D. thesis, Yale University, 2004.
[46] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, *RCV1: A new benchmark collection for text categorization research*, J. Mach. Learn. Res., 5 (2004), pp. 361–397.
[47] Z. Lin and R. B. Altman, *Finding haplotype tagging SNPs by use of principal components analysis*, Amer. J. Human Genetics, 75 (2004), pp. 850–861.
[48] M. W. Mahoney, M. Maggioni, and P. Drineas, *Tensor-CUR decompositions for tensor-based data*, in Proceedings of the 12th Annual ACM SIGKDD Conference, 2006, pp. 327–336.
[49] P.-G. Martinsson, V. Rokhlin, and M. Tygert, *A randomized algorithm for the approximation of matrices*, Technical report YALEU/DCS/TR-1361, Yale University Department of Computer Science, New Haven, CT, 2006.
[50] M. Z. Nashed, ed., *Generalized Inverses and Applications*, Academic Press, New York, 1976.
[51] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Classics in Applied Mathematics, SIAM, Philadelphia, 1998.
[52] P. Paschou, M. W. Mahoney, A. Javed, J. R. Kidd, A. J. Pakstis, S. Gu, K. K. Kidd, and P. Drineas, *Intra- and interpopulation genotype reconstruction from tagging SNPs*, Genome Res., 17 (2007), pp. 96–107.
[53] L. Rademacher, S. Vempala, and G. Wang, *Matrix approximation and projective clustering via iterative sampling*, Technical report MIT-LCS-TR-983, Massachusetts Institute of Technology, Cambridge, MA, 2005.
[54] M. Rudelson, *Random vectors in the isotropic position*, J. Funct. Anal., 164 (1999), pp. 60–72.
[55] M. Rudelson and R. Vershynin, *Approximation of matrices*, manuscript.
[56] M. Rudelson and R. Vershynin, *Sampling from large matrices: An approach through geometric functional analysis*, J. ACM, 54 (2007).
[57] T. Sarlós, *Improved approximation algorithms for large matrices via random projections*, in Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, 2006, pp. 143–152.
[58] G. W. Stewart, *Four algorithms for the efficient computation of truncated QR approximations to a sparse matrix*, Numer. Math., 83 (1999), pp. 313–323.
[59] G. W. Stewart, *Error analysis of the quasi-Gram-Schmidt algorithm*, Technical report UMIACS TR-2004-17 CMSC TR-4572, University of Maryland, College Park, MD, 2004.
[60] G. W. Stewart and J. G. Sun, *Matrix Perturbation Theory*, Academic Press, New York, 1990.
[61] E. Tyrtyshnikov, *Incomplete cross approximation in the mosaic-skeleton method*, Comput., 64 (2000), pp. 367–380.
[62] R. Vershynin, *Coordinate restrictions of linear operators in l_2^n*, manuscript.
[63] 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.
[64] 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, 2002, pp. 682–688.

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.