Numerical Linear Algebra in the Streaming Model

[Extended Abstract] *

Kenneth L. Clarkson
IBM Almaden Research Center
klclarks@us.ibm.com

David P. Woodruff
IBM Almaden Research Center
dpwoodru@us.ibm.com

ABSTRACT

We give near-optimal space bounds in the streaming model for linear algebra problems that include estimation of matrix products, linear regression, low-rank approximation, and approximation of matrix rank. In the streaming model, sketches of input matrices are maintained under updates of matrix entries; we prove results for turnstile updates, given in an arbitrary order. We give the first lower bounds known for the space needed by the sketches, for a given estimation error ϵ. We sharpen prior upper bounds, with respect to combinations of space, failure probability, and number of passes. The sketch we use for matrix A is simply $S^T A$, where S is a sign matrix.

Our results include the following upper and lower bounds on the bits of space needed for 1-pass algorithms. Here A is an $n \times d$ matrix, B is an $n \times d'$ matrix, and $c := d + d'$. These results are given for fixed failure probability; for failure probability $\delta > 0$, the upper bounds require a factor of $\log(1/\delta)$ more space. We assume the inputs have integer entries specified by $O(\log(nc))$ bits, or $O(\log(nd))$ bits.

1. (Matrix Product) Output matrix C with

$$\|A^T B - C\| \leq \epsilon \|A\| \|B\|.$$

We show that $\Theta(c\epsilon^{-2} \log(nc))$ space is needed.

2. (Linear Regression) For $d' = 1$, so that B is a vector b, find x so that

$$\|Ax - b\| \leq (1+\epsilon) \min_{x' \in \mathbb{R}^d} \|Ax' - b\|.$$

We show that $\Theta(d^2 \epsilon^{-1} \log(nd))$ space is needed.

3. (Rank-k Approximation) Find matrix $\tilde{A}_k$ of rank no more than k, so that

$$\|A - \tilde{A}_k\| \leq (1+\epsilon)\|A - A_k\|,$$

where A_k is the best rank-k approximation to A. Our lower bound is $\Omega(k\epsilon^{-1}(n+d)\log(nd))$ space, and we give a one-pass algorithm matching this when A is given row-wise or column-wise. For general updates, we give a one-pass algorithm needing

$$O(k\epsilon^{-2}(n + d/\epsilon^2)\log(nd))$$

space. We also give upper and lower bounds for algorithms using multiple passes, and a sketching analog of the CUR decomposition.

Categories and Subject Descriptors

F.2.0 [**Theory of Computation**]: Analysis of algorithms and problem complexity; G.1.3 [**Mathematics of Computing**]: Numerical linear algebra

General Terms

Algorithms Theory

1. INTRODUCTION

In recent years, starting with [16], many algorithms for numerical linear algebra have been proposed, with substantial gains in performance over older algorithms, at the cost of producing results satisfying Monte Carlo performance guarantees: with high probability, the error is small. These algorithms pick random samples of the rows, columns, or individual entries of the matrices, or else compute a small number of random linear combinations of the rows or columns of the matrices. These compressed versions of the matrices are then used for further computation. The two general approaches are *sampling* or *sketching*, respectively.

Algorithms of this kind are generally also *pass-efficient*, requiring only a constant number of passes over the matrix data for creating samples or sketches, and other work. Most such algorithms require at least two passes for their sharpest performance guarantees, with respect to error or failure probability. However, in general the question has remained of what was attainable in one pass. Such a one-pass algorithm is close to the *streaming* model of computation, where there is one pass over the data, and resource bounds are sublinear in the data size.

Muthukrishnan [21] posed the question of determining the complexity in the streaming model of computing or approximating various linear algebraic functions, such as the best rank-k approximation, matrix product, eigenvalues, determinants, and inverses. This problem was posed again by Sarlós [23], who asked what space and time lower bounds can be proven for any pass-efficient approximate matrix product, ℓ_2 regression, or SVD algorithm.

*A full version is available from the authors

STOC'09, May 31–June 2, 2009, Bethesda, Maryland, USA.

In this paper, we answer some of these questions. We also study a few related problems, such as rank computation. In many cases we give algorithms together with matching lower bounds. Our algorithms are generally sketching-based, building on and sometimes simplifying prior work on such problems. Our lower bounds are the first for these problems. Sarlós [23] also gives upper bounds for these problems, and our upper bounds are inspired by his and are similar, though a major difference here is our one-pass algorithm for low-rank approximation, improving on his algorithm needing two passes, and our space-optimal one-pass algorithms for matrix product and linear regression, that improve slightly on the space needed for his one-pass algorithms.

We generally consider algorithms and lower bounds in the most general *turnstile model* of computation [21]. In this model an algorithm receives arbitrary updates to entries of a matrix in the form "add x to entry (i, j)". An entry (i, j) may be updated multiple times and the updates to the different (i, j) can appear in an arbitrary order. Here x is an arbitrary real number of some bounded precision.

The relevant properties of algorithms in this setting are the space required for the sketches; the update time, for changing a sketch as updates are received; the number of passes; and the time needed to process the sketches to produce the final output. Our sketches are matrices, and the final processing is done with standard matrix algorithms. Although sometimes we give upper or lower bounds involving more than one pass, we reserve the descriptor "streaming" for algorithms that need only one pass.

1.1 Results and Related Work

The matrix norm used here will be the Frobenius norm $\|A\|$, where $\|A\| := \left[\sum_{i,j} a_{ij}^2\right]^{1/2}$, and the vector norm will be Euclidean. unless otherwise indicated. The spectral norm $\|A\|_2 := \sup_x \|Ax\|/\|x\|$.

We consider first the Matrix Product problem:

PROBLEM 1.1. Matrix Product. *Matrices A and B are given, with n rows and a total of c columns. The entries of A and B are specified by $O(\log nc)$-bit numbers. Output a matrix C so that*

$$\|A^T B - C\| \leq \epsilon \|A\| \|B\|.$$

Theorem 2.3 states that there is a streaming algorithm that solves an instance of this problem with correctness probability at least $1 - \delta$, for any $\delta > 0$, and using

$$O(c\epsilon^{-2} \log(nc) \log(1/\delta))$$

bits of space. The update time is $O(\epsilon^{-2} \log(1/\delta))$. This sharpens the previous bounds [23] with respect to the space and update time (for one prior algorithm) and update, final processing time, number of passes (which previously was two), and an $O(\log(1/\delta))$ factor in the space (for another prior algorithm) [23]. We note that it is also seems possible to obtain a one-pass $O(c\epsilon^{-2} \log(nc) \log(c/\delta))$-space algorithm via techniques in [2, 6, 10], but the space is suboptimal.

Moreover, Theorem 2.6 implies that this algorithm is optimal with respect to space, including for randomized algorithms. The theorem is shown using a careful reduction from an augmented version of the indexing problem, which has communication complexity restated in Theorem A.1.

The sketches in the given algorithms for matrix product, and for other algorithms in this paper, are generally of the form $S^T A$, where A is an input matrix and S is a *sign* matrix, also called a *Rademacher* matrix. Such a sketch satisfies the properties of the Johnson-Lindenstrauss Lemma for random projections, and the upper bounds given here follow readily using that lemma, except that the stronger conditions implied by the JL Lemma require resource bounds that are larger by a $\log n$ factor.

The algorithm mentioned above relies on a bound for the higher moments of the error of the product estimate, which is Lemma 2.2. The techniques used for that lemma also yield a more general bound for some other matrix norms, omitted in this abstract. The techniques of these bounds are not far from the *trace method* [24], which has been applied to analyzing the eigenvalues of a sign matrix. However, we analyze the use of sign matrices for matrix products, and in a setting of bounded independence, so that trace method analyses don't seem to immediately apply.

Second, we consider the following linear regression problem.

PROBLEM 1.2. Linear Regression. *Given an $n \times d$ matrix A and an $n \times 1$ column vector b, each with entries specified by $O(\log nd)$-bit numbers, output a vector x so that*

$$\|Ax - b\| \leq (1 + \varepsilon) \min_{x' \in \mathbb{R}^d} \|Ax' - b\|.$$

Theorem 3.4 gives a lower bound of $\Omega(d^2\epsilon^{-1} \log(nd))$ space for randomized algorithms for the regression problem (This is under a mild assumption regarding the number of bits per entry, and the relation of n to d.) Our upper bound algorithm requires a sketch with $O(d^2\epsilon^{-1} \log(1/\delta)))$ entries, with success probability $1 - \delta$, each entry of size $O(\log(nd))$, thus matching the lower bound, and improving on prior upper bounds by a factor of $\log d$ [23].

In Section 4, we give upper and lower bounds for low rank approximation:

PROBLEM 1.3. Rank-k Approximation. *Given integer k, value $\epsilon > 0$, and $n \times d$ matrix A, find a matrix $\tilde{A}_k$ of rank at most k so that*

$$\|A - \tilde{A}_k\| \leq (1 + \epsilon)\|A - A_k\|,$$

where A_k is the best rank-k approximation to A.

There have been several proposed algorithms for this problem, but all so far have needed more than 1 pass. A 1-pass algorithm was proposed by Achlioptas and McSherry [1], whose error estimate includes an additive term of $\|A\|$; that is, their results are not low relative error. Other work on this problem in the streaming model includes work by Desphande and Vempala [11], and by Har-Peled [17], but these algorithms require a logarithmic number of passes. Recent work on coresets [15] solves this problem for measures other than the Frobenius norm, but requires two passes. In particular, this algorithm solves Problem 28 of [21].

We give a one-pass algorithm needing

$$O(k\epsilon^{-2}(n + d/\epsilon^2) \log(nd) \log(1/\delta))$$

space. While this does not match the lower bound (given below), it is the first one-pass rank-k approximation with low relative error; only the trivial $O(nd \log(nd))$-space algorithm was known before in this setting, even for $k = 1$.

We also give a related construction, which may be useful in its own right: a low-relative-error sketching version of the

CUR decomposition; The CUR decomposition of Drineas *et al.* [13] is an approximation of a matrix A by the product of a matrix C whose columns are a subset of those of A, a matrix U, and a matrix R whose rows are a subset of those of A. That is, C and R are samples of A. Here we show that for appropriate sign matrices S and $\hat{S}$, the matrix $\tilde{A} := A\hat{S}(S^TA\hat{S})^-S^TA$ (where X^- denotes the pseudo-inverse of matrix X) satisfies $\|A - \tilde{A}\| \leq (1+\epsilon)\|A - A_k\|$, with probability $1-\delta$. The space needed by these three matrices is $O(k\epsilon^{-1}(n + d/\epsilon)\log(nd)\log(1/\delta))$. Such a decomposition of A may prove useful in itself. While samples of the columns and rows of A are more useful than the dense sketches $A\hat{S}$ and S^TA, the matrix $\tilde{A}$ is simple to compute, can be found in one pass, and has a better dependence on ϵ than the sampling CUR decomposition. It is also a bicriteria solution to the low-rank approximation problem, since its rank is at most $k\epsilon^{-1}\log(1/\delta)$.

When the entries of A are given a column or a row at a time, a streaming algorithm for low-rank approximation with the space bound

$$O(k\epsilon^{-1}(n+d)\log(nd)\log(1/\delta))$$

is achievable, as shown in Theorem 4.5. (It should be remarked that under such conditions, it may be possible to adapt earlier algorithms to use one pass.) Our lower bound Theorem 4.10 shows that at least $\Omega(k\epsilon^{-1}n)$ bits are needed, for row-wise updates, thus when $n \geq d$, this matches our upper bound up to a factor of $\log(nd)$ for constant δ.

Our lower bound Theorem 4.11, for general turnstile updates, is $\Omega(k\epsilon^{-1}(n+d)\log(nd))$, matching the row-wise upper bound. We give an algorithm for turnstile updates, also with space bounds matching this lower bound, but requiring two passes. (An assumption regarding the computation of intermediate matrices is needed for the multi-pass algorithms given here, as discussed in §1.4.)

Our lower bound Theorem 4.12 shows that even with multiple passes and randomization, $\Omega((n+d)k\log(nd))$ bits are needed for low-rank approximation, and we give an algorithm needing three passes, and $O(nk\log(nd))$ space, for n larger than a constant times $\max\{d/\epsilon, k/\epsilon^2\}\log(1/\delta)$.

In Section 5, we give bounds for the following.

PROBLEM 1.4. Rank Decision Problem. *Given an integer k, and a matrix A, output 1 iff the rank of A is at least k.*

The lower bound Theorem 5.2 states that $\Omega(k^2)$ bits of space are needed by a streaming algorithm to solve this problem with constant probability; the upper bound Theorem 5.1 states that $O(k^2\log(n/\delta))$ bits are needed for failure probability at most δ by a streaming algorithm. The lower bound is extended to the problem of checking the invertibility of A, and to eigenvalue or determinant estimation with small relative error, by reduction from Rank Decision.

Lower bounds for related problems have been studied in the two-party communication model [7, 8], but the results there only yield bounds for deterministic algorithms in the streaming model. Bar-Yossef [5] gives lower bounds for the *sampling complexity* of low rank matrix approximation and matrix reconstruction. We note that it is much more difficult to lower bound the space complexity. Indeed, for estimating the Euclidean norm of a length-n data stream, the sampling complexity is $\Omega(\sqrt{n})$ [4], while there is a sketching algorithm achieving $O((\log n)/\varepsilon^2)$ bits of space [3].

Space	Model	Theorem
Product		
$\Theta(c\epsilon^{-2}\log(nc))$	turnstile	2.3, 2.6
$O(c\epsilon^{-2})(\lg\lg(nc+\lg(1/\epsilon))$	col-wise	2.4
$\Omega(c\epsilon^{-2})$	A before B	2.7
Regression		
$\Theta(d^2\epsilon^{-1}\log(nd))$	turnstile	3.2, 3.4
$\Omega(d^2(\epsilon^{-1}+\log(nd)))$	insert-once	3.11
Rank-k Appr.		
$O(k\epsilon^{-2}(n+d\epsilon^{-2})\log(nd))$	turnstile	4.9
$\Omega(k\epsilon^{-1}(n+d)\log(nd))$	turnstile	4.11
$O(k\epsilon^{-1}(n+d)\log(nd))$	row-wise	4.5
$\Omega(k\epsilon^{-1}n)$	row-wise	4.10
$O(k\epsilon^{-1}(n+d)\log(nd))$	2, turnstile	4.4
$O(k(n+d\epsilon^{-1}+k\epsilon^{-2})\log(nd))$	3, row-wise	4.6
$\Omega(k(n+d)\log(nd))$	$O(1)$, turnstile	4.12
Rank Dec.		
$O(k^2\log n)$	turnstile	5.1
$\Omega(k^2)$	turnstile	5.2

Figure 1: Algorithmic upper and lower bounds given here; results are for one pass, unless indicated otherwise under "Model.". All space upper bounds are multiplied by $\log(1/\delta)$ for failure probability δ.

1.2 Techniques for the Lower Bounds

Our lower bounds come from reductions from the two-party communication complexity of augmented indexing. Alice is given $x \in \{0,1\}^n$, and Bob is given $i \in [n]$ together with $x_{i+1}, \ldots, x_n$. Alice sends a single message to Bob, who must output x_i with probability at least 2/3. Alice and Bob create matrices M_x and M_y, respectively, and use a streaming algorithm to solve augmented indexing.

For regression even obtaining an $\Omega(d^2\log(nd))$ bound is non-trivial. It is tempting for Alice to interpret x as a $d \times d$ matrix M_x with entries drawn randomly from $[nd]$. She sets $A = M_x^{-1}$, which she gives the streaming algorithm. Bob sets b to a standard unit vector, so that the solution is a column of $A^{-1} = M_x$, which can solve augmented indexing.

This argument is flawed because the entries of A may be exponentially small, so A is not a valid input. We instead design b in conjunction with A. We reduce from augmented indexing, rather than indexing (as is often done in streaming), since Bob must use his knowledge of certain entries of A to guarantee that A and b are valid inputs.

To achieve an extra factor of $1/\varepsilon$, we copy this construction $1/\varepsilon$ times. Bob can set b to force a large error on $1/\varepsilon - 1$ of the copies, forcing the regression coefficients to "approximately solve" the remaining copy. This approach loses a $\log(nd)$ factor, and to gain it back we let Bob delete entries that Alice places in A. The $\log(nd)$ factor comes from creating $\log(nd)$ groups, each group containing the $1/\varepsilon$ copies described above. The entries across the $\log(nd)$ groups grow geometrically in size. This idea is inspired by a lower bound for L_p-estimation in [22], though there the authors studied the Gap-Hamming Problem. Of the groups that are not deleted, only one contributes to the error, since the entries in other groups are too small.

1.3 Notation and Terminology

For integer n, let $[n]$ denote $\{1, 2, \ldots, n\}$.

A *Rademacher variable* is a random variable that is $+1$ or -1 with probability 1/2. A *sign* (or *Rademacher*) matrix has entries that are independent Rademacher variables. A *p-wise independent sign matrix* has entries that are Rademacher

variables, every subset of p or more entries being independent.

For a matrix A, let $a_{:j}$ denote the jth column of A, and a_{ij} denote the entry at row i and column j. More generally, use an upper case letter for a matrix, and the corresponding lower case for its columns and entries. We may write $a_{:j}^2$ for $\|a_{:j}\|^2$.

We say that matrices C and D are *conforming for multiplication*, or just *conforming*, if the number of columns of C equals the number of rows of D. If the appropriate number of rows and columns of a matrix can be inferred from context, we may omit it.

For a matrix A, let A^- denote the Moore-Penrose pseudo-inverse of A, so that $A^- = V\Sigma^- U^T$, where $A = U\Sigma V^T$ is the singular value decomposition of A.

The following is a simple generalization of the Pythagorean Theorem, and we will cite it that way.

THEOREM 1.5. **(Pythagorean Theorem)** *If C and D matrices with the same number of rows and columns, then $C^T D = 0$ implies $\|C + D\|^2 = \|C\|^2 + \|D\|^2$.*

PROOF. By the vector version, we have $\|C + D\|^2 = \sum_i \|c_{:i} + d_{:i}\|^2 = \sum_i \|c_{:i}\|^2 + \|d_{:i}\|^2 = \|C\|^2 + \|D\|^2$. □

For background on communication complexity, see Section A.

1.4 Bit Complexity

We will assume that the entries of an $n \times d$ input matrix are $O(\log(nd))$-bit integers. The sign matrices used for sketches can be assumed to have dimensions bounded by the maximum of n and d. Thus all matrices we maintain have entries of bit size bounded by $O(\log(nd))$. For low-rank approximation, some of the matrices we use in our two and three pass algorithms are rounded versions of matrices that are assumed to be computed exactly in between passes (though not while processing the stream). This "exact intermediate" assumption is not unreasonable in practice, and still allows our upper bounds to imply that the lower bounds cannot be improved.

2. MATRIX PRODUCTS

2.1 Upper Bounds

Given matrices A and B with the same number of rows, suppose S is a sign matrix also with the same number of rows, and with m columns. It is known that

$$\mathbf{E}[A^T SS^T B]/m = A^T \mathbf{E}[SS^T]B/m = A^T[mI]B/m = A^T B$$

and

$$\mathbf{E}[\|A^T SS^T B/m - A^T B\|^2] \leq 2\|A\|^2\|B\|^2/m. \qquad (1)$$

Indeed, the variance bound (1) holds even when the entries of S are not fully independent, but only 4-wise independent [23]. Such limited independence implies that the storage needed for S is only a constant number of random entries (a logarithmic number of bits), not the nm bits needed for explicit representation of the entries. The variance bound implies, via the Chebyshev inequality, that for given $\epsilon > 0$, there is an $m = O(1/\epsilon^2)$ such that $\|A^T SS^T B/m - A^T B\| \leq \epsilon\|A\|\|B\|$, with probability at least 3/4. We show that, also, if m is boosted up by a factor of $O(\log(1/\delta))$, for given $\delta > 0$, that the failure probability can be bounded by δ.

THEOREM 2.1. *For A and B matrices with n rows, and given $\delta, \epsilon > 0$, there is $m = \Theta(\log(1/\delta)/\epsilon^2)$, as $\epsilon \to 0$, so that for an $n \times m$ sign matrix S,*

$$\mathbf{P}\{\|A^T SS^T B/m - A^T B\| < \epsilon\|A\|\|B\|\} \geq 1 - \delta.$$

This bound holds also when S is a $4\lceil\log(\sqrt{2}/\delta)\rceil$-wise independent sign matrix.

Theorem 2.1 is proven using Markov's inequality and the following lemma, which generalizes (1), up to a constant. Here for a random variable X, $\mathbf{E}_p[X]$ denotes $[\mathbf{E}[|X|^p]]^{1/p}$.

LEMMA 2.2. *Given matrices A and B, suppose S is a sign matrix with $m > 15$ columns, and A, B, and S have the same number of rows. Then there is an absolute constant C so that for integer $p > 1$ with $m > Cp$,*

$$\mathbf{E}_p\left[\|A^T SS^T B/m - A^T B\|^2\right] \leq 4((2p-1)!!)^{1/p}\|A\|^2\|B\|^2/m.$$

This bound holds also when S is $4p$-wise independent.

The proof is omitted in this abstract.

For integer $p \geq 1$, the double factorial $(2p-1)!!$ denotes $(2p-1)(2p-3)\cdots 5\cdot 3\cdot 1$, or $(2p)!/2^p p!$. This is the number of ways to partition $[2p]$ into blocks all of size two. From Stirling's approximation, $(2p-1)!! \leq \sqrt{2}(2p/e)^p$.

Thus, the bound of Lemma 2.2 is $O(p)$ as $p \to \infty$, implying that

$$\mathbf{E}_p\left[\|A^T SS^T B/m - A^T B\|\right] = O(\sqrt{p})$$

as $p \to \infty$. It is well known that a random variable X with $\mathbf{E}_p[X] = O(\sqrt{p})$ is subgaussian, that is, its tail probabilities are dominated by those of a normal distribution.

The proof of Theorem 2.1 is omitted from this abstract.

The following algorithmic result is an immediate consequence of Theorem 2.1, maintaining sketches $S^T A$ and $S^T B$, and (roughly) standard methods to generate the entries of S with the independence specified by that theorem.

THEOREM 2.3. *Given $\delta, \epsilon > 0$, suppose A and B are matrices with n rows and a total of c columns. The matrices A and B are presented as turnstile updates, using at most $O(\log nc)$ bits per entry. There is a data structure that requires $m = O(\log(1/\delta)/\epsilon^2)$ time per update, and $O(cm\log(nc))$ bits of space, so that at a given time, $A^T B$ may be estimated such that with probability at least $1 - \delta$, the Frobenius norm of the error is at most $\epsilon\|A\|\|B\|$.*

2.2 Column-wise Updates

When the entries to A and B are received in column-wise order, a procedure using less space is possible. The sketches are not $S^T A$ and $S^T B$, but instead rounded versions of those matrices. If we receive entries of A (or B) one-by-one in a given column, we can maintain the inner product with each of the rows of S^T exactly using $m\log(cn)$ space. After all the entries of a column of A are known, the corresponding column of $S^T A$ is known, and its m entries can be rounded to the nearest power of $1 + \epsilon$. After all updates have been received, we have $\hat{A}$ and $\hat{B}$, where $\hat{A}$ is $S^T A$ where each entry has been rounded, and similarly for $\hat{B}$. We return $\hat{A}^T\hat{B}$ as our output.

The following theorem is an analysis of this algorithm. By Theorem 2.7 below, the space bound given here is within a factor of $\lg\lg(nc) + \lg(1/\epsilon)$ of best possible.

THEOREM 2.4. *Given $\delta, \epsilon > 0$, suppose A and B are matrices with n rows and a total of c columns. Suppose A and B are presented in column-wise updates, with integer entries having $O(\log(nc))$ bits. There is a data structure so that, at a given time, $A^T B$ may be estimated, so that with probability at least $1-\delta$ the Frobenius norm of the error at most $\epsilon\|A\|\|B\|$. There is $m = O(1/\epsilon^2)$ so that for c large enough, the data structure needs $O(cm \log 1/\delta)(\lg\lg(nc) + \lg(1/\epsilon))$ bits of space.*

The proof of Theorem 2.4 is omitted from this abstract.

The proof depends on the following, where the $\log n$ penalty of JL is avoided, since only a weaker condition is needed.

LEMMA 2.5. *For matrix A with n rows, and given $\delta, \epsilon > 0$, there is $m = \Theta(\epsilon^{-2}\log(1/\delta))$, as $\epsilon \to 0$, such that for an $n \times m$ sign matrix S,*

$$\mathbf{P}\{|\|S^T A\|/\sqrt{m} - \|A\|| \leq \epsilon\|A\|\} \geq 1-\delta.$$

The bound holds when the entries of S are p-wise independent, for large enough p in $O(\log(1/\delta))$.

The proof is omitted in this abstract.

2.3 Faster Products of Sketches

Taking the above rounding approach even further, we note that under some conditions it is possible to estimate the sketch product $A^T SS^T B$ more quickly than $O(dd'm)$, even as fast as $O(dd')$, where the constant in the $O(\cdot)$ notation is an absolute constant. As $dd' \to \infty$, if δ and ϵ are fixed (as so m is fixed), the necessary computation is to estimate the dot products of a large number of fixed-dimensional vectors (the columns of $S^T A$ with those of $S^T B$).

Suppose we build an ϵ-cover E for the unit sphere in $\mathbb{R}^m$, and map each column $\hat{a}_{:i}$ of $S^T A$ to $x \in E$ nearest to $\hat{a}_{:i}/\|\hat{a}_{:i}\|$, and similarly map each $\hat{b}_{:j}$ to some $y \in E$. Then the error in estimating $\hat{a}_{:i}^T \hat{b}_{:j}$ by $x^T y \|\hat{a}_{:i}\|\|\hat{b}_{:j}\|$ is at most $3\epsilon\|\hat{a}_{:i}\|\|\hat{b}_{:j}\|$, for ϵ small enough, and the sum of squares of all such errors is at most $9\epsilon^2\|S^T A\|^2\|S^T B\|^2$. By Lemma 2.5, this results in an overall additive error that is within a constant factor of $\epsilon\|A\|\|B\|$, and so is acceptable.

Moreover, if the word size is large enough that a table of dot products $x^T y$ for $x, y \in E$ can be accessed in constant time, then the time needed to estimate $A^T SS^T B$ is dominated by the time needed for at most dd' table lookups, yielding $O(dd')$ work overall.

Thus, under these word-size conditions, our algorithm is optimal with respect to number of passes, space, and the computation of the output from the sketches, perhaps leaving only the update time for possible improvement.

Even if δ and ε are not fixed, we can use fast rectangular matrix multiplication to compute $A^T SS^T B$. It is known [9, 18] that for a constant $\gamma > 0$, multiplying an $r \times r^\gamma$ matrix by an $r^\gamma \times r$ matrix can be done in r^2 polylog r time. An explicit value of $\gamma = .294$ is given in [9]. Thus, if $m \leq \min(d, d')^{.294}$, then $A^T SS^T B$ can be computed in dd'polylog$(\min(d, d'))$ time using block multiplication. This reduces the time complexity if m is larger than polylog$(\min(d, d'))$.

2.4 Lower Bounds for Matrix Product

THEOREM 2.6. *Suppose $n \geq \beta(\log_{10} cn)/\epsilon^2$ for an absolute constant $\beta > 0$, and that the entries of A and B are represented by $O(\log(nc))$-bit numbers. Then any randomized 1-pass algorithm which solves Problem 1.1, Matrix Product, with probability at least $4/5$ uses $\Omega(c\epsilon^{-2}\log(nc))$ bits of space.*

The proof of Theorem 2.6 is omitted from this abstract. For a less demanding computational model, we have:

THEOREM 2.7. *Suppose $n \geq \beta/\epsilon^2$ for an absolute constant $\beta > 0$, and that the entries of A and B are represented by $O(\log(nc))$-bit numbers. Then even if each entry of A and B appears exactly once in the stream, for every ordering of the entries of A and B for which every entry of A appears before every entry of B, any randomized 1-pass algorithm which solves Problem 1.1, Matrix Product, with probability at least $4/5$ uses $\Omega(c\epsilon^{-2})$ bits of space.*

The proof of Theorem 2.7 is omitted from this abstract.

3. REGRESSION

3.1 Upper Bounds

Our algorithm for regression is a consequence of the following theorem. For convenience of application of this result to algorithms for low-rank approximation, it is stated with multiple right-hand sides: that is, the usual vector b is replaced by a matrix B. Moreover, while the theorem applies to a matrix A of rank at most k, we will apply it to regression with the assumption that A has $d \leq n$ columns implying an immediate upper bound of d on the rank. This also is for convenience of application to low-rank approximation.

THEOREM 3.1. *Given $\delta, \epsilon > 0$, suppose A and B are matrices with n rows, and A has rank at most k. There is an $m = O(k\log(1/\delta)/\epsilon)$ such that, if S is an $n \times m$ sign matrix, then with probability at least $1-\delta$, if $\tilde{X}$ is the solution to*

$$\min_X \|S^T(AX - B)\|^2, \tag{2}$$

and X^* is the solution to

$$\min_X \|AX - B\|^2, \tag{3}$$

then

$$\|A\tilde{X} - B\| \leq (1+\epsilon)\|AX^* - B\|.$$

The entries of S need be at most $\eta(k + \log(1/\delta))$-wise independent, for a constant η.

This theorem has the following immediate algorithmic implication.

THEOREM 3.2. *Given $\delta, \epsilon > 0$, and $n \times d$ matrix A, and n-vector b, sketches of A and b of total size*

$$O(d^2\epsilon^{-1}\log(1/\delta)\log(nd))$$

can be maintained under turnstile updates, so that a vector $\tilde{x}$ can be found using the sketches, so that with probability at least $1-\delta$,

$$\|A\tilde{x} - b\| \leq (1+\epsilon)\|Ax^* - b\|,$$

where x^* minimizes $\|Ax - b\|$. The update time is

$$O(d\epsilon^{-2}\log(1/\delta)).$$

The proof of Theorem 3.1 is not far that of from Theorem 12 of [23], or that in [14]. The following lemma is crucial.

LEMMA 3.3. *For A, B, X^*, and $\tilde{X}$ as in Theorem 3.1,*

$$\|A(\tilde{X} - X^*)\| \leq 2\sqrt{\epsilon}\|B - AX^*\|$$

The proof of the lemma is omitted in this abstract. The proof of Theorem 3.1 is omitted from this abstract.

3.2 Lower Bounds for Regression

THEOREM 3.4. *Suppose $n \geq d(\log_{10}(nd))/(36\epsilon)$ and d is sufficiently large. Then any randomized 1-pass algorithm which solves the Linear Regression problem with probability at least 7/9 needs $\Omega(d^2\epsilon^{-1}\log(nd))$ bits of space.*

PROOF. Throughout we shall assume that $\varepsilon < 1/72$ and that $u := 1/(36\varepsilon)$ is an integer. Put $L := nd$.

We reduce from the AIND problem on strings of length $d(d-1)(\log_{10} L)/(72\epsilon)$. Alice interprets her input string as a $d(\log_{10} L)u \times d$ matrix A, which is constructed as follows.

For each $z \in \{0, \ldots, \log_{10} L - 1\}$ and each $k \in [u]$, we define an upper-triangular $d \times d$ matrix $A^{z,k}$. We say that $A^{z,k}$ is in *level* z and *band* k. The matrix $A^{z,k}$ consists of random $\{-10^z, +10^z\}$ entries inserted above the diagonal from Alice's input string. The diagonal entries of the matrices will be set by Bob. A is then the $d(\log_{10} L)u \times d$ matrix obtained by stacking the matrices $A^{0,1}, A^{0,2}, \ldots, A^{0,u}, A^{1,1}, \ldots, A^{1,u}, \ldots, A^{\log_{10} L - 1, u}$ on top of each other.

Bob has an index in the AIND problem, which corresponds to an entry $A^{z^*,k^*}_{i^*,j^*}$ for a $z^* \in \{0, 1, \ldots, \log_{10} L - 1\}$, a $k^* \in [u]$ and an $i^* < j^*$. Put $Q := 100^{z^*}(j^* - 1)$. Bob's input index is random and independent of Alice's input, and therefore, conditioned on the value of j^*, the value of i^* is random subject to the constraint $i^* < j^*$. Notice, in particular, that $j^* > 1$.

By definition of the AIND problem, we can assume Bob is given the entries in $A^{z,k}$ for all $z > z^*$ and each $k \in [u]$.

Let P be a large positive integer to be determined. Bob sets the diagonal entries of A as follows. Only matrices in level z^* have non-zero diagonal entries. Matrix A^{z^*,k^*} has all of its diagonal entries equal to P. The remaining matrices $A^{z^*,k}$ in level z^* in bands $k \neq k^*$ have $A^{z^*,k}_{j,j} = P$ whenever $j \geq j^*$, and $A^{z^*,k}_{j,j} = 0$ whenever $j < j^*$.

Alice feeds her entries of A into an algorithm Alg which solves the linear regression problem with probability at least 7/9, and transmits the state to Bob. Bob then feeds his entries of A into Alg. Next, using the entries that Bob is given in the AIND problem, Bob sets all entries of matrices $A^{z,k}$ in levels $z > z^*$ to 0, for every band k.

Bob creates the $d(\log_{10} L)u \times 1$ column vector b as follows. We think of b as being composed of $(\log_{10} L)u$ vectors $b^{z,k}$, $z \in \{0, \ldots, \log_{10} L - 1\}$, $k \in [u]$, so that b is the vector obtained by stacking $b^{0,1}, b^{0,2}, \ldots, b^{0,u}, \ldots, b^{\log_{10} L - 1, u}$ on top of each other. We say $b^{z,k}$ is in level z and band k.

For any $x \in \mathbb{R}^d$, the squared error of the linear regression problem is $\|Ax - b\|^2 = \sum_{z=0}^{\log_{10} L - 1} \sum_{k=1}^{u} \|A^{z,k}x - b^{z,k}\|^2$. For all vectors $b^{z^*,k}$ in level z^*, Bob sets $b^{z^*,k}_{j^*} = P$. He sets all other entries of b to 0, and feeds the entries of b to Alg.

We will show in Lemma 3.5 below that there exists a vector $x \in \mathbb{R}^d$ for which $\|Ax - b\|^2 \leq Q\left(u - \frac{97}{99}\right)$. It will follow by Lemma 3.9 that the vector x^* output by Alg satisfies various properties useful for recovering individual entries of A^{z^*,k^*}. By Lemma 3.10, it will follow that for most (j, j^*) pairs that Bob could have, the entry A^{z^*,k^*}_{j,j^*} can be recovered from x^*_j, and so this is also likely to hold of the actual input pair (i^*, j^*). Hence, Alice and Bob can solve the AIND problem with reasonable probability, thereby giving the space lower bound.

Consider the vector $x \in \mathbb{R}^d$ defined as follows. Let $x_j = 0$ for all $j > j^*$. Let $x_{j^*} = 1$. Finally, for all $j < j^*$, let $x_j = -A^{z^*,k^*}_{j,j^*}/P$.

LEMMA 3.5. $\|Ax - b\|^2 \leq Q\left(u - \frac{97}{99}\right)$.

PROOF. We start with three claims.

CLAIM 3.6. $(A^{z,k}x - b^{z,k})_j = 0$ *whenever* $z > z^*$.

PROOF. For $z > z^*$ and any k, $A^{z,k}$ is the zero matrix and $b^{z,k}$ is the zero vector. □

CLAIM 3.7. *For all $j \geq j^*$, $(A^{z,k}x - b^{z,k})_j = 0$.*

The proof of Claim 3.7 is omitted from this abstract.
Set $P = d^2L^4$. The number of bits needed to describe P is $O(\log L)$.

CLAIM 3.8. *For all $j < j^*$,*

- *For $(z,k) = (z^*, k^*)$, $(A^{z^*,k^*}x - b^{z^*,k^*})^2_j \leq \frac{1}{d^2L^4}$.*
- *For $(z,k) \neq (z^*, k^*)$, $(A^{z,k}x - b^{z,k})^2_j \leq 100^z + \frac{3}{dL}$.*

The proof of Claim 3.8 is omitted from this abstract.
From Claim 3.6, Claim 3.7, and Claim 3.8, we deduce:

- For any $z > z^*$ and any k, $\|A^{z,k}x - b^{z,k}\|^2 = 0$.
- $\|A^{z^*,k^*}x - b^{z^*,k^*}\|^2 = \sum_j (A^{z^*,k^*}x - b^{z^*,k^*})^2_j \leq \frac{1}{dL^4}$.
- For any $z \leq z^*$ and any $k \neq k^*$, $\|A^{z,k}x - b^{z,k}\|^2 = \sum_{j<j^*}(A^{z,k}x - b^{z,k})^2_j \leq 100^z(j^* - 1) + \frac{3}{L}$, where the inequality follows from the fact that we sum over at most d indices.

For $z = z^*$, using that $u - 1 = 1/(36\varepsilon) - 1 \geq 2$,

$$\begin{aligned}\sum_{k=1}^{u}\|A^{z^*,k}x - b^{z^*,k}\|^2 &\leq (u-1)\left[Q + \frac{3}{L}\right] + \frac{1}{dL^4} \\ &\leq (u-1)\left[Q + \frac{4}{L}\right],\end{aligned}$$

for sufficiently large d. Moreover,

$$\begin{aligned}\sum_{z<z^*}\sum_{k=1}^{u}\|A^{z,k}x - b^{z,k}\|^2 &\leq \sum_{z<z^*} u\left[100^z(j^* - 1) + \frac{4}{L}\right] \\ &\leq \left[\frac{Qu}{99}\right] + \frac{\log_{10} L}{9\varepsilon L}.\end{aligned}$$

We can bound the total error of x by adding these quantities,

$$\sum_{z=0}^{\log_{10} L - 1}\sum_{k=1}^{u}\|A^{z,k}x - b^{z,k}\|^2 \leq Q\left(u - \frac{98}{99}\right) + Err$$

where $Err = \frac{1}{9\varepsilon L} + \frac{\log_{10} L}{9\varepsilon L}$. Now, using the bound in the theorem statement, $\frac{1}{9\varepsilon} \leq \frac{4n}{d\log_{10} L}$, where the latter is upper-bounded by $\frac{n}{2d}$ for sufficiently large d. Hence, $\frac{1}{9\varepsilon L} \leq \frac{1}{2d^2}$. Moreover, $\frac{\log_{10} L}{9\varepsilon L}$ is at most $\frac{4n\log_{10} L}{nd^2\log_{10} L} \leq \frac{4}{d^2}$. It follows that $Err < \frac{5}{d^2}$. For sufficiently large d, $\frac{5}{d^2} \leq \frac{Q}{99}$, and so

$$\sum_{z=0}^{\log_{10} L - 1}\sum_{k=1}^{u}\|A^{z,k}x - b^{z,k}\|^2 \leq Q\left(u - \frac{97}{99}\right),$$

and the lemma follows.

Let x^* be the output of Alg. Then, using Lemma 3.5, and the fact that $u = 1/(36\varepsilon)$, with probability at least 7/9

$$\|Ax^* - b\|^2 \le (1+\varepsilon)^2\|Ax-b\|^2 \le (1+3\varepsilon)\left(u - \frac{97}{99}\right)Q$$
$$\le \left(u - \frac{97}{99} + \frac{1}{12}\right)Q \le \left(u - \frac{43}{48}\right)Q. \quad (4)$$

Call this event $\mathcal{E}$. We condition on $\mathcal{E}$ occurring in the remainder of the proof.

LEMMA 3.9. *The following conditions hold simultaneously:*

1. *For all $j > j^*$, $x_j^* \in [-L^2/P, L^2/P]$.*
2. *For $j = j^*$, $x_j^* \in [1 - L^2/P, 1 + L^2/P]$.*
3. *For $j < j^*$, $x_j^* \in [-L^2/P, L^2/P]$.*

The proof of Lemma 3.9 is omitted from this abstract.

LEMMA 3.10. *With probability at least $1 - 49/d$, for at least a $41/46$ fraction of the indices $j < j^*$, we have*

$$\operatorname{sign}(x_j^*) = -\operatorname{sign}(A_{j,j^*}^{z^*,k^*}).$$

Notice that $A_{j,j^*}^{z^*,k^*} \in \{-10^{z^*}, 10^{z^*}\}$, so its sign is well-defined.

The proof of Lemma 3.10 is omitted from this abstract.
Bob lets x^* be the output of Alg and outputs $-\operatorname{sign}(x_{i^*}^*)$. Since i^* is random subject to $i^* < j^*$, the previous lemma ensures that for sufficiently large d, Bob's correctness probability is at least $\frac{41}{46} - \frac{49}{d} \ge \frac{8}{9}$, given $\mathcal{E}$. By a union bound, Alice and Bob solve the AIND problem with probability at least $\frac{8}{9} - \frac{2}{9} \ge \frac{2}{3}$, and so the space complexity of Alg must be $\Omega(d^2(\log(nd))/\varepsilon)$.

THEOREM 3.11. *Suppose $n \ge d/(36\epsilon)$. Consider the Linear Regression problem in which the entries of A and b are inserted exactly once in the data stream. Then any randomized 1-pass algorithm which solves this problem with probability at least $7/9$ needs $\Omega(d^2(1/\epsilon + \log(nd)))$ bits of space.*

The proof of Theorem 3.11 is omitted from this abstract.

4. LOW-RANK APPROXIMATION

4.1 Upper Bounds

We give several algorithms, trading specificity and passes for space. As mentioned in §1.4, we will assume that we can compute some matrices exactly, and then round them for use. We will show that all matrices (up to those exact computations) can be used in rounded form during the streaming phase.

Throughout this section, A is an $n \times d$ input matrix of rank ρ, with entries of size $\gamma = O(\log(nd))$ as $nd \to \infty$. The value k is a given integer, A_k is the best rank-k approximation to A, $\Delta_k := \|A - A_k\|$ is the error of A_k, $\delta > 0$ is the probability of failure, and $\epsilon > 0$ is the given error parameter.

Bit Complexity.

To prove space and error bounds for these algorithms, we will show that the numerical error is small enough, using $O(\log(nd))$-bit entries, assuming the input matrix also has entries of $O(\log(nd))$ bits. We will assume that, between passes, we may do exact computations, and then round the results to use during and after the next pass. A key property here is that the singular values of an integer matrix with bounded entries cannot be too small or too large. We note that this lemma is also proven and used in [15], though there it is used for measures other than the Frobenius norm.

LEMMA 4.1. *If $n \times d$ matrix A has integer entries bounded in magnitude by γ, and has rank $\rho \ge 2k$, then the k'th singular value σ_k of A has $|\log \sigma_k| = O(\log(nd\gamma))$ as $nd \to \infty$. This implies that $\|A\|/\Delta_k \le (nd\gamma)^{O(1)}$ as $nd \to \infty$.*

The proof of Lemma 4.1 is omitted from this abstract.
The analysis of the low-rank approximation algorithms is based on the application of Theorem 3.1, as in the following theorem; again, the proof technique is similar to that of [23, 14].

THEOREM 4.2. *There is an $m = O(k\log(1/\delta)/\epsilon)$ such that, if S is an $n \times m$ sign matrix, then with probability at least $1 - \delta$, there is an $n \times m$ matrix Y of rank at most k, so that*

$$\|YS^TA - A\| \le (1+\epsilon)\Delta_k.$$

Similarly, for a $d \times m$ sign matrix R, with probability at least $1 - \delta$ there is an $m \times d$ matrix Z so that

$$\|ARZ - A\| \le (1+\epsilon)\Delta_k.$$

The entries of S and R need be at most $\eta(k + \log(1/\delta))$-wise independent, for a constant η.

The theorem says that the rowspace of S^TA contains a very good rank-k approximation to A, and similarly for the columnspace of AR.
The proof of Theorem 4.2 is omitted from this abstract.
The following lemma will be helpful.

LEMMA 4.3. *Given a matrix A and matrix U with orthonormal columns, both with the same number of rows, the best rank-k approximation to A in the columnspace of U is given by $U[U^TA]_k$, where $[U^TA]_k$ is the best rank-k approximation to U^TA. A similar claim applies for G a matrix with orthonormal rows, and the best rank-k approximation to A in the rowspace of G.*

The proof of Lemma 4.3 is omitted from this abstract.

4.1.1 Two passes

The most direct application of the above theorem and lemma yields a two pass algorithm, as follows. In the first pass, accumulate S^TA. Before the second pass, compute a matrix G whose rows are an orthonormal basis for the rowspace of S^TA. In the second pass, accumulate the coefficients AG^T of the projection $AG^TG = A(S^TA)^-S^TA$ of A onto the row space of S^TA. Finally, compute the best rank-k approximation $[AG^T]_k$ to AG^T, and return $\tilde{A}_k = [AG^T]_kG$. As proven below, this approximation is close to A.

Although this discussion assumes that G is computed exactly, we will show that an approximation $\hat{G}$ can be used: for an appropriate κ in $O(\log(nd))$, $\hat{G}$ is $2^{-\kappa}\lfloor 2^\kappa G \rfloor$, stored implicitly as a scaling factor and an integer matrix. (Here $\lfloor\ \rfloor$ denotes the floor function applied entrywise.)

THEOREM 4.4. *If the rank $\rho \ge 2(k+1)$, then there is an $m = O(k\log(1/\delta)/\epsilon)$ such that, if S is an $n \times m$ sign matrix,*

then with probability at least $1-\delta$, the rank-k matrix $\tilde{A}_k$, as returned by the above two-pass algorithm, satisfies

$$\|A - \tilde{A}_k\| \leq (1+\epsilon)\Delta_k.$$

An approximation $\hat{G}$ to G with $O(\log(nd))$-bit entries may be used, with the same asymptotic bounds. The space used is

$$O(m(n+d))\log(nd) = O(k\epsilon^{-1}(n+d))\log(1/\delta)\log(nd).$$

By Theorem 4.12, the space used is optimal for fixed ϵ (and δ).

The proof of Theorem 4.4 is omitted from this abstract.

4.1.2 One pass for Column-wise Updates

If A is given a column at a time, or a row at a time, then an efficient streaming algorithm is possible. By Theorem 4.10, for n within a constant factor of d, the space used by this algorithm is within a factor of $\log(nd)$ of optimal.

THEOREM 4.5. *Suppose input A is given as a sequence of columns or rows. There is an $m = O(k\log(1/\delta)/\epsilon)$, such that with probability at least $1-\delta$, a matrix $\tilde{A}_k$ can be obtained that satisfies*

$$\|\tilde{A}_k - A\| \leq (1+\epsilon)\Delta_k.$$

The space needed is

$$O((n+d)m) = O(k\epsilon^{-1}(n+d)\log(1/\delta)\log(nd)).$$

The update time is amortized $O(m)$ per entry.

The proof of Theorem 4.5 is omitted from this abstract.

4.1.3 Three passes for Row-wise Updates, With Small Space

We show the following.

THEOREM 4.6. *Suppose A is given row-wise. There is $m = O(k\log(1/\delta)/\epsilon)$ such that, a matrix $\tilde{A}_k$ can be found in three passes so that with probability at least $1-\delta$,*

$$\|\tilde{A}_k - A\| \leq (1+\epsilon)\Delta_k.$$

The algorithm uses space

$$O(k(n + d\log(1/\delta)/\epsilon + k\log(1/\delta)^2/\epsilon^2)\log(nd)).$$

A comparable approach, without sketching, would use $\Theta((nk+d^2)\log(nd))$ space over two passes, so this result becomes interesting when $k < \epsilon d$. As mentioned in the introduction, for n larger than a constant times $\max\{d/\epsilon, k/\epsilon^2\}\log(1/\delta)$, the space bound is $O(nk\log(nd))$, which is comparable to our lower bound Theorem 4.12, showing that $\Omega((n+d)k\log(nd))$ bits are needed even with multiple passes and randomization.

The proof of Theorem 4.6 is omitted from this abstract.

4.1.4 One pass, and a CUR decomposition

To obtain a low-rank approximation even for turnstile updates, we will need more space. First, we can apply Theorem 3.1 twice to obtain a sketching analog of the CUR decomposition [13, 12].

THEOREM 4.7. *There is an $m = O(k\log(1/\delta)/\epsilon)$ such that, if S is an $n \times (m/\epsilon)$ sign matrix, and R is a $d \times m$ sign matrix, then with probability at least $1-\delta$,*

$$\|A - \tilde{A}\| \leq (1+\epsilon)\Delta_k,$$

where $\tilde{A} := AR(S^TAR)^-S^TA$. The entries of S need be at most $\eta(k/\epsilon + \log(1/\delta))$-wise independent, for a constant η.

PROOF. We apply Theorem 3.1 with k, A, B, and m of the theorem mapping to k/ϵ, AR, A, and m/ϵ, respectively. The result is that for $\tilde{X}$ the solution to

$$\min_X \|S^TARX - S^TA\|,$$

we have

$$\|AR\tilde{X} - A\| \leq (1+\epsilon)\|ARX^* - A\| = (1+\epsilon)\min_X \|ARX - A\|,$$

and applying Theorem 3.1 again, with k, A, B, and m of the theorem mapping to m, A_k, A, and m, we have, with probability at least $1-\delta$,

$$\|ARX^* - A\| \leq (1+\epsilon)\|A - A_k\| = (1+\epsilon)\Delta_k. \quad (5)$$

Since $\tilde{X} = (S^TAR)^-S^TA$, we have

$$\begin{aligned}\|AR(S^TAR)^-S^TA - A\| &= \|AR\tilde{X} - A\| \\ &\leq (1+\epsilon)\|ARX^* - A\| \\ &\leq (1+\epsilon)^2\Delta_k,\end{aligned}$$

and the theorem follows, after adjusting δ and ϵ by constant factors. □

Note that by computing the SVD $\tilde{U}\tilde{\Sigma}\tilde{V}^T$ of $(S^TAR)^-$, we obtain a low-rank approximation to A of the form

$$AR\tilde{U}\tilde{\Sigma}\tilde{V}^TS^TA,$$

which is of the same form as an SVD. While this decomposition has rank $O(k\epsilon^{-1}\log(1/\delta))$, and is guaranteed to approximate A only nearly as well as the best rank-k approximation, it would be much quicker to compute, and potentially could be substituted for the SVD in many applications.

A rank-k approximation is similarly obtainable, as follows.

THEOREM 4.8. *Under the conditions of the previous theorem, let U be an orthonormal basis for the columnspace of AR. Then the best rank-k approximation $U[U^T\tilde{A}]_k$ to $\tilde{A}$ in the columnspace of U satisfies*

$$\|A - U[U^T\tilde{A}]_k\| \leq (1+\sqrt{\epsilon})\Delta_k.$$

For convenience of reference, we state a result giving a quality bound of the usual form, simply using a different ϵ.

THEOREM 4.9. *There is an $m = O(k\log(1/\delta)/\epsilon^2)$ such that, if S is an $n \times (m/\epsilon^2)$ sign matrix, and R is a $d \times m$ sign matrix, the following succeeds with probability $1-\delta$. Let U be an orthonormal basis for the columnspace of AR. Then the best rank-k approximation $U[U^T\tilde{A}]_k$ to $\tilde{A} := AR(S^TAR)^-S^TA$ in the columnspace of U satisfies*

$$\|A - U[U^T\tilde{A}]_k\| \leq (1+\epsilon)\Delta_k.$$

The entries of S need be at most $\eta(k/\epsilon + \log(1/\delta))$-wise independent, for a constant η.

PROOF. (of Theorem 4.8.) For any such U, there is a matrix Y so that $UY = AR$; in particular, we will take U to be the matrix of left singular vectors of AR, so that the corresponding Y is ΣV^T.

Consider the projections UU^TA and UU^TA_k of A and A_k to the columnspace of AR, as well as $\tilde{A}$, which is already in the columnspace of AR. We first obtain distance bounds

involving these projections, by applying Lemma 3.3 used in the proof of Theorem 3.1; this bound is used twice, first in the setting of the first application of Theorem 3.1, and then in the setting of the second application.

The projection UU^TA can also be expressed as ARX^*, with X^* as in the first application of Theorem 3.1, and $\tilde{A}$ then equal to the corresponding $AR\tilde{X}$. From Lemma 3.3 and (5),

$$\begin{aligned}\|UU^TA - \tilde{A}\| &= \|AR(X^* - \tilde{X})\| \\ &\le 2\sqrt{\epsilon}\|A - ARX^*\| \\ &\le 2\sqrt{\epsilon}(1+\epsilon)\Delta_k. \end{aligned} \quad (6)$$

Since the projection UU_TA_k is the closest matrix in the columnspace of AR to A_k, and again from Lemma 3.3, as in the second application above, we have

$$\|UU^TA_k - A_k\| \le \|AR(A_kR)^-A_k - A_k\| \le 2\sqrt{\epsilon}\Delta_k. \quad (7)$$

Also, since $[U^T\tilde{A}]_k$ is the closest rank-k matrix to $U^T\tilde{A} = Y(S^TAR)^-S^TA$, U^TA_k must be no closer to $U^T\tilde{A}$, and so

$$\begin{aligned}&\|\tilde{A} - U[U^T\tilde{A}]_k\| \le \|\tilde{A} - UU^TA_k\| \\ \text{(triangle ineq.)}\quad &\le \|\tilde{A} - UU^TA\| + \|UU^TA - UU^TA_k\| \\ \text{(By (6))}\quad &\le 2\sqrt{\epsilon}(1+\epsilon)\Delta_k + \|UU^T(A - A_k)\|. \end{aligned} \quad (8)$$

Since $\|UU^TZ\| \le \|Z\|$ for any Z, we have

$$\|UU^TA - UU^TA_k\| \le \Delta_k. \quad (9)$$

Since $(A - UU^TA)^TU = 0$, we have

$$\begin{aligned}&\|A - U[U^T\tilde{A}]_k\|^2 - \|A - UU^TA\|^2 \\ \text{(Pyth. Thm.)}\quad &= \|UU^TA - U[U^T\tilde{A}]_k\|^2 \\ \text{(triangle ineq.)}\quad &\le (\|UU^TA - \tilde{A}\| + \|\tilde{A} - U[U^T\tilde{A}]_k\|)^2 \\ \text{(By (6),(8))}\quad &\le (2\sqrt{\epsilon}(1+\epsilon)\Delta_k + [2\sqrt{\epsilon}(1+\epsilon)\Delta_k \\ &\quad + \|UU^T(A - A_k)\|])^2 \\ &= (4\sqrt{\epsilon}(1+\epsilon)\Delta_k + \|UU^T(A - A_k)\|])^2 \\ &= \|UU^TA - UU^TA_k\|^2 \\ &\quad + 8\sqrt{\epsilon}(1+\epsilon)\Delta_k\|UU^TA - UU^TA_k\| \\ &\quad + 16\epsilon(1+\epsilon)^2\Delta_k^2 \\ \text{(By (9))}\quad &\le \|UU^TA - UU^TA_k\|^2 \\ &\quad + 8\sqrt{\epsilon}(1+\epsilon)\Delta_k + 16\epsilon(1+\epsilon)^2\Delta_k^2 \\ &= \|UU^TA - UU^TA_k\|^2 \\ &\quad + 8\Delta_k^2(1+\epsilon)(\sqrt{\epsilon} + 2\epsilon(1+\epsilon)). \end{aligned}$$

Rearranging this bound,

$$\begin{aligned}&\|A - U[U^T\tilde{A}]_k\|^2 - 8\Delta_k^2(1+\epsilon)(\sqrt{\epsilon} + 2\epsilon(1+\epsilon)) \\ &\qquad \le \|A - UU^TA\|^2 + \|UU^TA - UU^TA_k\|^2 \\ \text{(Pyth. Thm.)}\quad &= \|A - UU^TA_k\|^2 \\ \text{(triangle ineq.)}\quad &\le (\|A - A_k\| + \|A_k - UU^TA_k\|)^2 \\ \text{(By (7))}\quad &\le (\Delta_k + 2\sqrt{\epsilon}\Delta_k)^2, \end{aligned}$$

which implies

$$\begin{aligned}\|A - U[U^T\tilde{A}]_k\|^2 &\le 8\Delta_k^2(1+\epsilon)(\sqrt{\epsilon} + 2\epsilon(1+\epsilon)) \\ &\quad + (\Delta_k + 2\sqrt{\epsilon}\Delta_k)^2 \\ &\le \Delta_k^2(1 + 12\sqrt{\epsilon} + O(\epsilon)), \end{aligned}$$

and the theorem follows upon taking square roots, and adjusting ϵ by a constant factor. □

4.2 Lower Bounds for Low-Rank Approximation

The next theorem shows that our 1-pass algorithm receiving entries in row or column order uses close to the best possible space of any streaming algorithm.

THEOREM 4.10. *Let $\epsilon > 0$ and $k \ge 1$ be arbitrary.*

- *Suppose $d > \beta k/\epsilon$ for an absolute constant $\beta > 0$. Then any randomized 1-pass algorithm which solves the Rank-k Approximation Problem with probability at least 5/6, and which receives the entries of A in row-order, must use $\Omega(nk/\epsilon)$ bits of space.*
- *Suppose $n > \beta k/\epsilon$ for an absolute constant $\beta > 0$. Then any randomized 1-pass algorithm which solves the Rank-k Approximation Problem with probability at least 5/6, and which receives the entries of A in column-order must use $\Omega(dk/\epsilon)$ bits of space.*

The proof of Theorem 4.10 is omitted from this abstract.

We can improve the bound of Theorem 4.10 if we assume the algorithm must work in the general turnstile model.

THEOREM 4.11. *Let $\varepsilon > 0$ and $k \ge 1$ be arbitrary. Suppose $\min(n,d) > \beta k \log_{10}(nd)/\varepsilon$ for an absolute constant $\beta > 0$. Then any randomized 1-pass algorithm which solves the Rank-k Approximation Problem with probability at least 5/6 in the general turnstile model uses $\Omega((n+d)k\log(dn))/\epsilon$ bits of space.*

The proof of Theorem 4.11 is omitted from this abstract.

Our $O(1)$-pass upper bounds match the following trivial lower bound, which is immediate from Corollary A.2.

THEOREM 4.12. *For any $1 \le k \le \min(n,d)$ and any $\epsilon > 0$, any multi-pass algorithm for the Rank-k Approximation Problem with probability of error at most 1/3 must use $\Omega((n+d)k\log(nd))$ bits of space. Moreover, this holds for any ordering of the entries of A.*

5. RANK DECISION

THEOREM 5.1. *Suppose A is an $n \times n$ matrix. The Rank Decision Problem can be solved in 1-pass with $O(k^2 \log n/\delta)$ bits of space with error probability at most δ.*

The proof of Theorem 5.1 is omitted from this abstract.

Via a few binary searches, one can design an algorithm using $O(\log \mathrm{rank}(A))$ passes and $O(\mathrm{rank}^2(A)\log(n/\delta))$ space to actually compute the rank of A based on the above decision problem. We omit the details.

THEOREM 5.2. *Any randomized 1-pass algorithm which solves the Rank Decision Problem with probability of error at most 1/3 must use $\Omega(k^2)$ bits of space.*

The proof of Theorem 5.2 is omitted from this abstract.

Any algorithm which computes the rank of A also solves the Rank Decision Problem for $k = \mathrm{rank}(A)$, so it has complexity $\Omega(\mathrm{rank}(A)^2)$. As the instance in the Rank Decision Problem concerns distinguishing a rank-k from a rank-$k-1$ square $k \times k$ matrix, it also gives an $\Omega(n^2)$ lower bound for

testing if an $n \times n$ matrix is invertible and for approximating the determinant to within any relative error. By adjusting the diagonal values in the upper left quadrant of the matrix A in the proof, one easily obtains an $\Omega(n^2)$ space bound for approximating the i-th largest eigenvalue for any value of i.

Acknowledgments: We thank the anonymous referees for helpful comments. The second author would also like to thank Dan Feldman, Morteza Monemizadeh, and Christian Sohler for helpful discussions.

6. REFERENCES

[1] D. Achlioptas and F. Mcsherry. Fast computation of low-rank matrix approximations. *J. ACM*, 54(2):9, 2007.

[2] N. Alon, P. B. Gibbons, Y. Matias, and M. Szegedy. Tracking join and self-join sizes in limited storage. *J. Comput. Syst. Sci.*, 64(3):719–747, 2002.

[3] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999.

[4] Z. Bar-Yossef. The complexity of massive data set computations, 2002.

[5] Z. Bar-Yossef. Sampling lower bounds via information theory. In *STOC*, pages 335–344, 2003.

[6] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *ICALP*, pages 693–703, 2002.

[7] J. I. Chu and G. Schnitger. The communication complexity of several problems in matrix computation. *J. Complexity*, 7(4):395–407, 1991.

[8] J. I. Chu and G. Schnitger. Communication complexity of matrix computation over finite fields. *Mathematical Systems Theory*, 28(3):215–228, 1995.

[9] D. Coppersmith. Rectangular matrix multiplication revisited. *J. Complexity*, 13(1):42–49, 1997.

[10] G. Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75, 2005.

[11] A. Deshpande and S. Vempala. Adaptive sampling and fast low-rank matrix approximation. In *APPROX-RANDOM*, pages 292–303, 2006.

[12] P. Drineas, R. Kannan, and M. W. Mahoney. Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition. *SIAM Journal on Computing*, 36(1):184–206, 2006.

[13] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error *cur* matrix decompositions. *SIAM Journal on Matrix Analysis and Applications*, 30(2):844–881, 2008.

[14] P. Drineas, M. W. Mahoney, S. Muthukrishnan, and T. Sarlós. Faster least squares approximation. Technical report, 2007. arXiv:0710.1435.

[15] D. Feldman, M. Monemizadeh, C. Sohler, and D. Woodruff. Coresets and sketches for subspace approximation problems, 2008.

[16] A. Frieze, R. Kannan, and S. Vempala. Fast monte-carlo algorithms for finding low-rank approximations. *J. ACM*, 51(6):1025–1041, 2004.

[17] S. Har-Peled. Low-rank approximation in linear time, 2006.

[18] X. Huang and V. Y. Pan. Fast rectangular matrix multiplication and applications. *J. Complexity*, 14(2):257–299, 1998.

[19] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[20] P. B. Miltersen, N. Nisan, S. Safra, and A. Wigderson. On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998.

[21] S. Muthukrishnan. *Data streams: algorithms and applications*. Foundations and Trends in Theoretical Computer Science, 2005.

[22] J. Nelson and D. Woodruff. Revisiting norm estimating in data streams, 2008.

[23] T. Sarlós. Improved approximation algorithms for large matrices via random projections. In *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 143–152, Washington, DC, USA, 2006. IEEE Computer Society.

[24] V. H. Vu. Spectral norm of random matrices. *Combinatorica*, 27(6):721–736, 2007.

APPENDIX

A. COMMUNICATION COMPLEXITY

For lower bounds, we will use a variety of definitions and basic results from two-party communication complexity, as discussed in [19]. We will call the two parties Alice and Bob.

For a function $f : \mathcal{X} \times \mathcal{Y} \to \{0,1\}$, we use $R_\delta^{1-way}(f)$ to denote the randomized communication complexity with two-sided error at most δ in which only a single message is sent from Alice to Bob. We also use $R_{\mu,\delta}^{1-way}(f)$ to denote the minimum communication of a protocol, in which a single message from Alice to Bob is sent, for solving f with probability at least $1-\delta$, where the probability is taken over both the coin tosses of the protocol and an input distribution μ.

In the augmented indexing problem, which we call $AIND$, Alice is given $x \in \{0,1\}^n$, while Bob is given both an $i \in [n]$ together with $x_{i+1}, x_{i+2}, \ldots, x_n$. Bob should output x_i.

THEOREM A.1. *([20]) $R_{1/3}^{1-way}(AIND) = \Omega(n)$ and also $R_{\mu,1/3}^{1-way}(AIND) = \Omega(n)$, where μ is uniform on $\{0,1\}^n \times [n]$.*

COROLLARY A.2. *Let A be a randomized algorithm, which given a random $x \in \{0,1\}^n$, outputs a string $A(x)$ of length m. Let B be an algorithm which given a random $i \in [n]$, outputs a bit $B(A(x))$ so that with probability at least $2/3$, over the choice of x, i, and the coin tosses of A and B, we have $B(A(x))_i = x_i$. Then $m = \Omega(n)$.*

PROOF. Given an instance of $AIND$ under distribution μ, Alice sends $A(x)$ to Bob, who computes $B(A(x))_i$, which equals x_i with probability at least $2/3$. Hence, $m = \Omega(n)$. □

Suppose x, y are Alice and Bob's input, respectively. We derive lower bounds for computing $f(x,y)$ on data stream $x \circ y$ as follows. Any r-pass streaming algorithm A yields a $(2r-1)$-round communication protocol for f in the following way. Alice computes $A(x)$ and sends the state of A to Bob, who computes $A(x \circ y)$. Bob sends the state of A back to Alice, who continues the execution of A (the second pass) on $x \circ y$. In the last pass, Bob outputs the answer. The communication is $2r-1$ times the space of the algorithm.