A Randomized Algorithm for a Tensor-Based Generalization of the Singular Value Decomposition

Petros Drineas [*] Michael W. Mahoney [†]

Technical Report, YALEU/DCS/TR-1327, June 2005.

Abstract

An algorithm is presented and analyzed that, when given as input a d-mode tensor $\mathcal{A}$, computes an approximation $\tilde{\mathcal{A}}$. The approximation $\tilde{\mathcal{A}}$ is computed by performing the following for each of the d modes: first, form (implicitly) a matrix by "unfolding" the tensor along that mode; then, choose columns from the matrices thus generated; and finally, project the tensor along that mode onto the span of those columns. An important issue affecting the quality of the approximation is the choice of the columns from the matrices formed by "unfolding" the tensor along each of its modes. In order to address this issue, two algorithms of independent interest are presented that, given an input matrix A and a target rank k, select columns that span a space close to the best rank k subspace of the matrix. For example, in one of the algorithms, a number c (that depends on k, an error parameter ϵ, and a failure probability δ) of columns are chosen in c independent random trials according to a nonuniform probability distribution depending on the Euclidean lengths of the columns. When this algorithm for choosing columns is used in the tensor approximation, then under appropriate assumptions bounds of the form

$$\left\|\mathcal{A}-\tilde{\mathcal{A}}\right\|_F \leq \sum_{i=1}^{d} \left\|A_{[i]}-(A_{[i]})_{k_i}\right\|_F + d\epsilon \left\|\mathcal{A}\right\|_F$$

are obtained, where $A_{[i]}$ is the matrix formed by "unfolding" the tensor along the i-th mode and where $(A_{[i]})_{k_i}$ is the best rank k_i approximation to the matrix $A_{[i]}$. Each $\left\|A_{[i]}-(A_{[i]})_{k_i}\right\|_F$ term is a measure of the extent to which the matrix $A_{[i]}$ is not well-approximated by a rank-k_i matrix, and the $\epsilon \left\|\mathcal{A}\right\|_F$ term is a measure of the loss in approximation quality due to the choice of columns (rather than, e.g., the top k_i singular vectors along each mode). Bounds of a similar form are obtained with the other column selection algorithm. When restricted to matrices, the main tensor decomposition provides a low-rank matrix decomposition that is expressed in terms of the columns and rows of the original matrix, rather than in terms of its singular vectors. Connections with several similar recently-developed matrix decompositions are discussed.

1 Introduction

In this introductory section, we first, in Section 1.1, provide a review of relevant tensor algebra, and then, in Section 1.2, we describe two results relating to the quality of approximation obtained by projecting a matrix onto a subset of its columns. (These results are of independent interest, and they will be used in an essential way in our main tensor approximation.) Then, in Section 1.3,

[*] Department of Computer Science, Rensselaer Polytechnic Institute, Troy, New York 12180, drinep@cs.rpi.edu

[†] Department of Mathematics, Yale University, New Haven, Connecticut, USA 06520, mahoney@cs.yale.edu

we describe our main result, a randomized algorithm to compute an approximation to a tensor and a theorem providing an associated quality-of-approximation bound. Finally, in Section 1.4 we provide an outline of the remainder of the paper.

1.1 Review of Tensor Algebra

Before stating our main result, we provide a brief review of relevant tensor algebra and a definition of notation that we will use. We shall use calligraphic letters to denote higher-order or multi-mode tensors with $d > 2$ modes. For example, let $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ be a d-mode tensor of size $n_1 \times n_2 \times \cdots \times n_d$. Consider the following definitions:

- Given a tensor $\mathcal{A}$ and a particular mode $\alpha \in \{1, \ldots, d\}$, define the matrix $A_{[\alpha]} \in \mathbb{R}^{n_\alpha \times N_\alpha}$, where $N_\alpha = \prod_{i \neq \alpha} n_i$ and where the columns of the matrix consist of varying the α^{th} coordinate of $\mathcal{A}$ while leaving the rest fixed. We refer to the (usually implicit) construction of $A_{[\alpha]}$ as *matricizing* [25] or *unfolding* [32] $\mathcal{A}$ along mode α and define the *α-rank* of the tensor $\mathcal{A}$ to be the rank of the matrix $A_{[\alpha]}$.

- Given a tensor $\mathcal{A}$, a particular mode α, and any $n_\alpha \times c_\alpha$ matrix B, define the *α-mode tensor-matrix product* to be the d-mode tensor of size $n_1 \times \cdots \times n_{\alpha-1} \times c_\alpha \times n_{\alpha+1} \times \cdots \times n_d$ whose $i_1 \cdots i_d$ element is

$$(\mathcal{A} \times_\alpha B)_{i_1 \cdots i_d} = \sum_{i=1}^{n_\alpha} \mathcal{A}_{i_1 \cdots i_{\alpha-1} i i_{\alpha+1} \cdots i_d} B_{i i_\alpha}. \qquad (1)$$

  (To understand this notation better, consider how it is applied to a $m \times n$ matrix A, i.e., to a 2-mode tensor. In this case, e.g., the SVD of A may be written $A = U \Sigma V^T = \Sigma \times_1 U \times_2 V$. Note that the α-mode tensor-matrix product satisfies $(\mathcal{A} \times_\alpha B) \times_{\alpha'} C = (\mathcal{A} \times_{\alpha'} C) \times_\alpha B = \mathcal{A} \times_\alpha B \times_{\alpha'} C$, assuming that the various individual products are defined.)

- Given a tensor $\mathcal{A}$, if we denote the SVD of $A_{[\alpha]}$ by $A_{[\alpha]} = U_{A_{[\alpha]}} \Sigma_{A_{[\alpha]}} V_{A_{[\alpha]}}^T = U_{[\alpha]} \Sigma_{[\alpha]} V_{[\alpha]}^T$ (the latter being an abuse of notation in favor of simplicity), where, e.g., $U_{[\alpha]}$ is an $n_\alpha \times \text{rank}(A_{[\alpha]})$ matrix, then the *higher-order SVD* of $\mathcal{A}$ is the decomposition of $\mathcal{A}$ of the form

$$\mathcal{A} = \mathcal{S} \times_1 U_{[1]} \times_2 \cdots \times_d U_{[d]}, \qquad (2)$$

  where the $\text{rank}(A_{[1]}) \times \cdots \times \text{rank}(A_{[d]})$ tensor $\mathcal{S}$ is the so-called core tensor.

- If $U_{[\alpha],k_\alpha}$ is a $n_\alpha \times k_\alpha$ matrix consisting of the left singular vectors corresponding to the top k_α singular values of $A_{[\alpha]}$, then

$$\tilde{\mathcal{A}} = \mathcal{S} \times_1 U_{[1],k_1} \times_2 \cdots \times_d U_{[d],k_d}, \qquad (3)$$

  is the *best rank-(k_1,k_2,...,k_d) approximation* to the tensor $\mathcal{A}$. (See [32, 33] for further discussion of this decomposition and approximation.)

- Given a d-mode tensor $\mathcal{A}$, define the (square of its) *Frobenius norm* to be

$$\|\mathcal{A}\|_F^2 = \sum_{i_1=1}^{n_1} \cdots \sum_{i_d=1}^{n_d} \mathcal{A}_{i_1 \cdots i_d}^2 .$$

Remark: See [32, 25] and references therein for a more detailed description of these tensor-related definitions.

Remark: Tensors are a natural generalization of matrices (see, e.g., [21] for more details) and have been studied in several fields. For example, tensors have been studied in mathematics and computer science for their algebraic properties, their ability to efficiently represent multidimensional functions, and the relationship between their properties and problems in complexity theory [21, 18, 23, 37, 4]. In addition, tensors provide a natural way to represent many large and complex data sets [32, 31, 22, 25, 34, 41, 5, 42, 43].

Remark: It is worth emphasizing that computing the rank of a general tensor $\mathcal{A}$ (defined as the minimum number of rank one tensors into which $\mathcal{A}$ can be decomposed) is an NP-hard problem, that only weak bounds are known relating the α-rank and the tensor rank, and that there does not exist definitions of tensor rank and associated tensor SVD such that the optimality properties of the matrix rank and matrix SVD are preserved [29, 24, 30, 23, 33, 27, 35, 36, 47].

1.2 Randomized Algorithms for Matrix Problems

Before stating our main tensor approximation result, we also discuss two intermediate results related to the quality of approximation obtained by projecting a matrix onto a subset of its columns. (These two intermediate results are discussed separately since they are of independent interest.) Given an $m \times n$ matrix A, we wish to choose columns of A such that the projection of the matrix onto those columns "captures" as much of the matrix as possible, i.e., that is a basis for a space close to the space spanned by the top singular vectors of the matrix. Thus, in particular, if A is well approximated by a low-rank matrix, then we would like $A \approx P_{span(C)}A$, where C is a matrix consisting of the chosen columns and where $P_{span(C)}$ is a projection onto the column space of C.

In Section 2 the following two algorithmic results are presented in more detail. Each algorithm takes as input a matrix A and a number c of columns to choose (the second algorithm also takes as input a number t of iterations to perform), and each algorithm returns as output an $m \times c$ matrix C whose columns consist of the chosen columns. Let ϵ be an error parameter and δ be a failure probability (due to the randomization in the algorithm). In addition, let A_k be the best rank-k approximation to A and let $\|\cdot\|_F$ denote the Frobenius norm of a matrix.

- In the SelectColumnsSinglePass algorithm (described more precisely in Algorithm 2 in Section 2.1), the columns of C are chosen from the columns of A in c i.i.d. trials by sampling randomly according to the probability distribution proportional to the Euclidean lengths squared of the column. If $c \geq 4(1+\sqrt{8\log(1/\delta)})^2 k/\epsilon^2$ columns are chosen in this manner then under appropriate assumptions (which are stated more precisely in Theorem 2 in Section 2.1) we have that

$$\left\|A - CC^{+}A\right\|_F^2 \leq \|A - A_k\|_F^2 + \epsilon\,\|A\|_F^2\,, \tag{4}$$

  with probability at least $1-\delta$.

- In the SelectColumnsMultiPass algorithm (described more precisely in Algorithm 3 in Section 2.2), the columns of C are chosen from the columns of A in t iterative rounds. In each round, c columns are chosen from A in c i.i.d. trials by sampling randomly according to the probability distribution proportional to the Euclidean lengths squared of the columns of residual of the matrix A after subtracting the projection of A on the subspace spanned by the columns sampled in all the previous rounds. If $c \geq 4(1+\sqrt{8\log(1/\delta)})^2 k/\epsilon^2$ columns are chosen in this manner in each of the t rounds, then under appropriate assumptions (which are stated more precisely in Theorem 3 in Section 2.2) we have that

$$\left\|A - CC^{+}A\right\|_F^2 \leq \frac{1}{1-\epsilon} \left\|A - A_k\right\|_F^2 + \epsilon^t \left\|A\right\|_F^2 , \tag{5}$$

with probability at least $1 - \delta$.

Remark: In (4) and (5), the $\|A - A_k\|_F^2$ term is a measure of the extent to which the matrix A is not well-approximated by a rank-k matrix, and the $\epsilon \|A\|_F^2$ or $\epsilon^t \|A\|_F^2$ term is a measure of the loss in approximation quality due to the choice of columns (rather than, e.g., the top k singular vectors). This latter measure is of the form of an arbitrary (but fixed) precision, scaled by a measure of the size of the matrix A.

Remark: Recent work in the randomized algorithms literature has focused on algorithms for extremely large linear algebra problems [16, 17, 44, 7, 2, 1, 8, 9, 10, 11, 12, 14, 15, 38, 6]. We should note that algorithms from this literature are quite different from traditional numerical analysis approaches and generally may be viewed as fitting within the following framework. They will be allowed to read the data from external storage a small number of times and keep a small randomly-chosen and rapidly-computable "sketch" of the data in RAM. Computations will then be performed on the "sketch" and the results of these computations will be returned as approximations to the solution of the original problem. In the interests of simplicity, we will not discuss in detail the resource requirements for the algorithms we present, except to note that they are efficient within the Pass Efficient Model of data streaming computation [9, 10]. In this model, the data are assumed to be stored externally (e.g., on a tape or disk and not in RAM) and the scarce computational resources are the number of sequential-access passes over the data and the additional scratch space and computational time required; see [10, 11] for a detailed discussion of these issues.

Remark: As discussed in Section 2, the second column selection algorithm is presented in the recent work by Rademacher, Vempala, and Wang [38].

1.3 Summary of the Main Result

In this subsection, we present the main result of the paper. Our main result is a randomized algorithm to compute an approximation to a tensor and a theorem providing an associated quality-of-approximation bound. We first describe the algorithm and then we present the theorem.

The ApproxTensorSVD algorithm (described in Algorithm 1) takes as input a d-mode tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ and numbers $c_i, i = 1, \ldots, d$. The algorithm returns as output matrices $C_{[i]}$ of size $n_i \times c_i$ for all $i = 1, \ldots, d$. Let $A_{[i]}$ be the matrix formed by "unfolding" the tensor $\mathcal{A}$ along the i-th mode. The algorithm works by choosing (for every $i \in \{1, \ldots, d\}$) c_i columns from the (not explicitly constructed) matrix $A_{[i]}$ to construct (explicitly) an $n_i \times c_i$ matrix, which we will denote (with an abuse of notation in the interests of simplicity) by $C_{[i]}$. The columns may be chosen from $A_{[i]}$ using either the SelectColumnsSinglePass algorithm or with the SelectColumnsMultiPass algorithm (in the latter case a number t of iterations to perform is also input to the ApproxTensorSVD algorithm). In both cases, the matrix bounds from (4) and (5) (see also Section 2.1 and Section 2.2) will translate into bounds for tensor approximation. Regardless of how the matrices $C_{[i]}$ are determined, we note that $C_{[i]}C_{[i]}^{+}$ is a projection matrix onto the space spanned by the columns of $C_{[i]}$. Thus,

$$\tilde{\mathcal{A}} = \mathcal{A} \times_1 C_{[1]}C_{[1]}^{+} \times_2 \cdots \times_d C_{[d]}C_{[d]}^{+} \tag{6}$$

is an approximation to the original tensor $\mathcal{A}$.

```
ApproxTensorSVD Algorithm
Data   : tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, $c_i, 1 \leq c_i \leq \prod_{j=1, j \neq i}^{d} n_j, i = 1, \ldots, d$.
Result : matrices $C_{[i]} \in \mathbb{R}^{n_i \times c_i}$ for all $i = 1, \ldots, d$ such that

         $\mathcal{A} \approx \mathcal{A} \times_1 C_{[1]} C_{[1]}^{+} \times_2 C_{[2]} C_{[2]}^{+} \times_3 \cdots \times_d C_{[d]} C_{[d]}^{+}$.

for $i = 1, \ldots, d$ do
    Form $C_{[i]}$ by calling
        • the SelectColumnsSinglePass algorithm with input $A_{[i]}$ and $c_i$, or
        • the SelectColumnsMultiPass algorithm with input $A_{[i]}$, $c_i$, and $t$ ;
end
```

Algorithm 1: The ApproxTensorSVD Algorithm.

The following theorem provides our main quality-of-approximation bound for the ApproxTensorSVD algorithm. In this theorem, we bound the Frobenius norm of the error tensor $\tilde{\mathcal{E}} = \mathcal{A} - \tilde{\mathcal{A}}$. The proof may be found in Section 3.2.

Theorem 1 *Let $\mathcal{A} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ be a d-mode tensor, and let $\eta = 1 + \sqrt{(8/\beta)\log(1/\delta)}$, for any $0 < \delta < 1$. Let matrices $C_{[i]}, i \in \{1, \ldots, d\}$, be computed by the* ApproxTensorSVD *algorithm (Algorithm 1).*

- *If the columns are chosen with the* SelectColumnsSinglePass *algorithm (see Algorithm 2 in Section 2.1) then, with probability at least $1 - d\delta$,*

$$\left\| \mathcal{A} - \mathcal{A} \times_1 C_{[1]} C_{[1]}^{+} \times_2 \ldots \times_d C_{[d]} C_{[d]}^{+} \right\|_F \leq \sum_{i=1}^{d} \left\| A_{[i]} - \left(A_{[i]}\right)_{k_i} \right\|_F + d\epsilon \left\| \mathcal{A} \right\|_F , \qquad (7)$$

  if $c_i \geq 4\eta^2 k_i / (\beta\epsilon^2)$ for all $i = 1, \ldots, d$.

- *If the columns are chosen with the* SelectColumnsMultiPass *algorithm (see Algorithm 3 in Section 2.2) then, with probability at least $1 - td\delta$,*

$$\left\| \mathcal{A} - \mathcal{A} \times_1 C_{[1]} C_{[1]}^{+} \times_2 \ldots \times_d C_{[d]} C_{[d]}^{+} \right\|_F \leq \frac{1}{1-\epsilon} \sum_{i=1}^{d} \left\| A_{[i]} - \left(A_{[i]}\right)_{k_i} \right\|_F + d\epsilon^t \left\| \mathcal{A} \right\|_F , \qquad (8)$$

  if $c_i \geq 4\eta^2 k_i / (\beta\epsilon^2)$ for every one of the t rounds and for all $i = 1, \ldots, d$.

Remark: In (7), each $\left\| A_{[i]} - (A_{[i]})_{k_i} \right\|_F$ term is a measure of the extent to which the matrix $A_{[i]}$ is not well-approximated by a rank-k_i matrix, and the $d\epsilon \left\| \mathcal{A} \right\|_F$ term is a measure of the loss in approximation quality due to the choice of columns (rather than, e.g., the top k_i singular vectors along each mode). This latter measure is of the form of an arbitrary (but fixed) precision, scaled by the (fixed) number of modes times a measure of the size of the tensor $\mathcal{A}$. A similar interpretation holds for the bound (8).

Remark: Recent work in applied linear algebra has focused on such areas as:

- computing approximate decompositions of tensors [32, 33, 31, 25, 26, 27, 35, 36],
- randomized algorithms for extremely large matrix problems [16, 7, 2, 10, 11, 12, 17, 38], and
- computing low-rank matrix approximations that are expressed in terms of the rows and columns of the matrix rather than in terms of its singular vectors [39, 40, 3, 20, 19, 12, 15].

Our main approximate tensor decomposition combines ideas from the first two of these areas. As discussed in greater detail in Section 4, when our main approximate tensor decomposition is applied to two-mode tensors, i.e., to matrices, it sheds light on the third area.

1.4 Outline of the Paper

In the next section, Section 2, we discuss several results related to approximating a matrix by projecting the matrix onto subsets of its columns. These results are of independent interest, and they will be used in our main tensor approximation. In Section 3, we prove Theorem 1 and provide a discussion of our main result. In Section 4, we consider the case when our main approximate tensor decomposition is applied to a two-mode tensor, i.e., to a matrix. Particular attention is paid to how our our main approximate tensor decomposition relates to several recently-developed low-rank matrix decompositions that are expressed in terms of the columns and rows of the original matrix, rather than in terms of its singular vectors. Finally, in Section 5, a brief conclusion is presented.

2 Projecting Matrices on Subsets of their Columns

Given an $m \times n$ matrix A, we wish to choose columns of A such that the projection of the matrix onto those columns "captures" as much of the matrix as possible, i.e., that is a basis for a space close to the space spanned by the top singular vectors of the matrix. Thus, in particular, if A is well approximated by a low-rank matrix, then we would like $A \approx P_{span(C)}A$, where C is a matrix consisting of the chosen columns and where $P_{span(C)}$ is a projection onto the column space of C.

In Section 2.1 and Section 2.2, two column selection results are presented. In each subsection, an algorithm is first described and then a theorem is presented that states the conditions under which an approximation of the given form is obtained. Thus, the two theorems in this section quantify the sense in which (for the two different choices of C) $A \approx P_{span(C)}A$.

2.1 Choosing the Columns in a Single Pass

The SelectColumnsSinglePass algorithm (described in Algorithm 2) takes as input a matrix A and a number c of columns to choose. It returns as output a matrix C such that the columns of C are chosen from the columns of A in c i.i.d. trials by sampling randomly according to the probability distribution (9). More formally, for an $m \times n$ matrix A and a multiset $S \subseteq \{1, \ldots, n\}$, let $C = A_S$ denote the $m \times |S|$ matrix whose columns are the columns of A with indices in S. The SelectColumnsSinglePass constructs the multiset S by randomly sampling according to (9) and returns the matrix $C = A_S$.

Remark: The SelectColumnsSinglePass algorithm is so-named since, given probabilities of the form (9), the matrix C can be constructed in one pass over the (externally-stored) data matrix A. If the probabilities $\{p_i\}_{i=1}^{n}$ that are used by the algorithm are not given based on a knowledge of the application area then one pass over the data and $O(m+n)$ additional space is sufficient to calculate them; see [10] for more details on these resource requirements.

```
SelectColumnsSinglePass Algorithm
Data   : $A \in \mathbb{R}^{m \times n}$, $c \in \mathbb{Z}^{+}$ s.t. $1 \le c \le n$.
Result : $C \in \mathbb{R}^{m \times c}$, s.t. $CC^{+}A \approx A$.
Compute (for some positive $\beta \le 1$) probabilities $\{p_i\}_{i=1}^{n}$ s.t.
    $p_i \ge \beta \left|A^{(i)}\right|^2 / \|A\|_F^2 ,$        (9)
where $A^{(i)}$ is the $i$-th column of $A$ as a column vector.
$S = \{\}$;
for $t = 1, \ldots, c$ do
    Pick $i_t \in \{1, \ldots, n\}$ with $\mathbf{Pr}\left[i_t = \alpha\right] = p_\alpha$;
    $S = S \cup \{i_t\}$;
end
$C = A_S$;
```

Algorithm 2: The SelectColumnsSinglePass Algorithm

Remark: Although any choice of sampling probabilities could be used in the SelectColumnsSinglePass algorithm, a judicious choice of these probabilities allows us to prove interesting error bounds for the quality of the approximation; see [10, 11] for a detailed discussion. Sampling probabilities of the form (9) (with $\beta = 1$ and where $A^{(i)}$ is the i-th column of the matrix A and $\left|A^{(i)}\right|$ is its Euclidean length) are optimal in a sense made precise in [10].

The following theorem is our main quality-of-approximation result for the SelectColumnsSinglePass algorithm.

Theorem 2 *Suppose $A \in \mathbb{R}^{m \times n}$, and let C be the $m \times c$ matrix constructed by sampling c columns of A with the* SelectColumnsSinglePass *algorithm. If $\eta = 1 + \sqrt{(8/\beta)\log(1/\delta)}$ for any $0 < \delta < 1$, then, with probability at least $1 - \delta$,*

$$\left\|A - CC^{+}A\right\|_F^2 \le \|A - A_k\|_F^2 + \epsilon \|A\|_F^2 , \tag{10}$$

if $c \ge 4\eta^2 k/(\beta\epsilon^2)$.

Proof: Let the $m \times c$ matrix $\hat{C}$ be that matrix whose columns consist of appropriately rescaled copies of the columns of C, as discussed in conjunction with the LinearTimeSVD algorithm of [11]. First, note that since $CC^{+} = P_C = P_{\hat{C}} = \hat{C}\hat{C}^{+}$ is a projection onto the full column space of C, it follows that

$$\left\|A - CC^{+}A\right\|_F^2 \le \left\|A - P_{\hat{C},k}A\right\|_F^2 . \tag{11}$$

The theorem follows by combining this with the results of [11].

◇

Remark: The crucial role played in the algorithm by the nonuniform sampling probabilities should be emphasized. By providing a bias toward columns that are more informative (in the sense of having greater Euclidean length), sampling probabilities of the form (9) allow the algorithm to obtain substantial variance reduction relative to less informative (e.g., uniform) probabilities. This allows a martingale argument to be successfully applied so that bounds of the form (10) hold not only in expectation but also with exponentially high probability.

Remark: In many applications the data may be generated in such a way that all of the columns have approximately the same Euclidean length. In these cases, uniform probabilities are approximately optimal and we can obtain meaningful bounds for the approximation error. More precisely, if there exists some positive constant $\beta \leq 1$ such that $\forall i : \left|A^{(i)}\right|^2 \leq \frac{1}{\beta n} \|A\|_F^2$ then the uniform probabilities are nearly optimal and we can sample uniformly with only a small β-dependent loss in accuracy; see [10, 11] for a detailed discussion.

Remark: The relationship of this algorithm with the LinearTimeSVD algorithm of [11] (and thus with the algorithms of [16, 7, 10, 12, 17, 38]) should also be emphasized. In the LinearTimeSVD algorithm of [11], the columns of A that are sampled by the algorithm are *scaled* prior to being included in C, by dividing each sampled column by a quantity proportional to the square root of the probability of picking it. This scaling allows one to prove that the top k singular values of the matrix $\hat{C}$, i.e., the scaled version of C, and the top k singular values of A are close. Additionally, it allows one to prove that under appropriate assumptions

$$\left\|A - P_{\hat{C},k}A\right\|_\xi^2 \leq \|A - A_k\|_\xi^2 + \epsilon \|A\|_F^2 , \tag{12}$$

in both expectation and with high probability, for both the spectral and Frobenius norms, $\xi = 2, F$. Algorithms of this flavor were first analyzed in [16], and have also been studied in [44, 7, 2, 1, 8, 9, 10, 11, 12, 14, 15, 17, 38]. In this paper in the projection matrix to the full space spanned by the columns of C, namely $P_C = CC^+ = \hat{C}\hat{C}^+ = P_{\hat{C}}$ rather than $P_{\hat{C},k}$. Clearly, any scaling of the columns of C does not affect this full projection matrix.

2.2 Choosing the Columns in Multiple Passes

The SelectColumnsMultiPass algorithm (described in Algorithm 3) was originally presented in [38]. The algorithm takes as input a matrix A, a number t of rounds to perform, and a number c of columns to choose per round. It returns as output a matrix C such that the columns of C are chosen from the columns of A in the following manner. There are t rounds, and each round consists of 2 passes over the data. In the first round, let $\ell = 1$. Sampling probabilities of the form (9) are computed in the first pass of the first round, and in the second pass a multiset S_1 of columns of A is picked in c i.i.d. trials by sampling according to the probabilities (9). For each subsequent round $\ell = 2, \ldots, t$, sampling probabilities of the form (14) are constructed that depend on the lengths of the columns of the the $m \times n$ matrix E_ℓ that is the residual of the matrix A after subtracting the projection of A on the subspace spanned by the columns sampled in the first $\ell - 1$ rounds.

More formally, let the indices of the columns that have been chosen in the first $\ell - 1$ rounds form the multiset $\{S_1, S_2, \ldots, S_{\ell-1}\}$ (where the multiset of columns S_i were chosen in the i-th round) and let $C_{\ell-1} = A_{\{S_1,S_2,\ldots,S_{\ell-1}\}}$ denote the $m \times |S_1||S_2| \cdots |S_{\ell-1}|$ matrix whose columns are the columns of A with indices in $\{S_1, S_2, \ldots, S_{\ell-1}\}$. Then,

$$E_\ell = A - A_{\{S_1,\ldots,S_{\ell-1}\}} A^+_{\{S_1,\ldots,S_{\ell-1}\}} A = A - C_{\ell-1} C^+_{\ell-1} A. \tag{13}$$

Sampling probabilities of the form (14) are then constructed in the first pass of each round $\ell = 2, \ldots, t$, and c columns are chosen from A by sampling in c i.i.d. trials according to the probabilities (14) in the second pass of each round $\ell = 2, \ldots, t$. (Note that if, by definition, $E_1 = A$, then for $\ell = 1$ the sampling probabilities (14) are the same as those of (9).)

Remark: The SelectColumnsMultiPass algorithm is so-named since, given probabilities of the form (14), c columns can be extracted in one pass over the (externally-stored) data matrix A. Of course, in each round the probabilities $\{p_i\}_{i=1}^n$ that are used by the algorithm may be

```
SelectColumnsMultiPass Algorithm
Data   : $A \in \mathbb{R}^{m \times n}$, $c \in \mathbb{Z}^+$ s.t. $1 \leq c \leq n$, $t \in \mathbb{Z}^+$.
Result : $C \in \mathbb{R}^{m \times c}$, s.t. $CC^+A \approx A$.
$S = \{\}$;
for $\ell = 1, \ldots, t$ do
    if $\ell == 1$ then
        $E_1 = A$;
    else
        $E_\ell = A - A_S A_S^+ A$;
    end
    Compute (for some positive $\beta \leq 1$) probabilities $\{p_i\}_{i=1}^n$ s.t.

        $p_i \geq \beta \left| E_\ell^{(i)} \right|^2 / \|E_\ell\|_F^2 ,$        (14)

    where $E_\ell^{(i)}$ is the $i$-th column of $E_\ell$ as a column vector.
    for $t = 1, \ldots, c$ do
        Pick $i_t \in \{1, \ldots, n\}$ with $\mathbf{Pr}[i_t = \alpha] = p_\alpha$;
        $S = S \cup \{i_t\}$;
    end
end
$C = A_S$;
```

Algorithm 3: The SelectColumnsMultiPass Algorithm

computed with one pass over the data and $O(1)$ additional space. The algorithm is thus efficient in the Pass Efficient Model; see [10] for more details on these resource requirements.

Remark: We should emphasize that Rademacher, Vempala and Wang, motivated to improve the magnitude of the additional additive error in the bound (12), explored alternative constructions for the matrix C [38]. They developed the column selection procedure we are presenting as the SelectColumnsMultiPass algorithm.

The following theorem is our main quality-of-approximation result for the SelectColumnsMultiPass algorithm.

Theorem 3 *Suppose $A \in \mathbb{R}^{m \times n}$ and let C be the $m \times tc$ matrix constructed by sampling c columns of A in each of t rounds with the* SelectColumnsMultiPass *algorithm. If $\eta = 1 + \sqrt{(8/\beta)\log(1/\delta)}$ for any $0 < \delta < 1$, then, with probability at least $1 - t\delta$,*

$$\left\|A - CC^+A\right\|_F^2 \leq \frac{1}{1-\epsilon}\left\|A - A_k\right\|_F^2 + \epsilon^t \left\|A\right\|_F^2, \tag{15}$$

if $c \geq 4\eta^2 k/(\beta\epsilon^2)$ columns are picked in each of the t rounds.

Proof: The proof will be by induction on the number of rounds t. Let S_1 denote the set of columns picked at the first round, and let $C^1 = A_{S_1}$. Thus, C^1 is an $m \times c$ matrix, where $c \geq 4\eta^2 k/(\beta\epsilon^2)$. By Theorem 2 and since $1 < 1/(1-\epsilon)$ for $\epsilon > 0$, we have that

$$\left\|A - C^1\left(C^1\right)^+ A\right\|_F^2 \leq \frac{1}{1-\epsilon}\left\|A - A_k\right\|_F^2 + \epsilon\left\|A\right\|_F^2 \tag{16}$$

holds with probability at least $1 - \delta$, thus establishing the base case of the induction.

Next, let $(S_1, \ldots, S_{t-1})$ denote the set of columns picked in the first $t-1$ rounds and let $C^{t-1} = A_{(S_1,\ldots,S_{t-1})}$. Assume that the proposition holds after $t-1$ rounds, i.e., assume that by choosing $c \geq 4\eta^2 k/(\beta\epsilon^2)$ columns in each of the first $t-1$ rounds, we have that

$$\left\|A - C^{t-1}\left(C^{t-1}\right)^+ A\right\|_F^2 \leq \frac{1}{1-\epsilon}\left\|A - A_k\right\|_F^2 + \epsilon^{t-1}\left\|A\right\|_F^2 \tag{17}$$

holds with probability at least $1 - (t-1)\delta$.

We will prove that it also holds after t rounds. Let $E_t = A - C^{t-1}\left(C^{t-1}\right)^+ A$ be the residual of the matrix A after subtracting the projection of A on the subspace spanned by the columns sampled in the first $t-1$ rounds. (Note that it is $\left\|E_t\right\|_F^2$ that is bounded by (17)). Consider sampling columns of E_t at round t with probabilities proportional to the square of their Euclidean lengths, i.e., according to (14), and let Z be the matrix of the columns of E_t that are included in the sample. (Note that these columns of E_t have the same span and thus projection as the corresponding columns of A when the latter are restricted to the residual space.) Then, by choosing at least $c \geq 4\eta^2 k/(\beta\epsilon^2)$ columns of E_t in the t-th round we can apply Theorem 2 to E_t and get that

$$\left\|E_t - ZZ^+E_t\right\|_F^2 \leq \left\|E_t - (E_t)_k\right\|_F^2 + \epsilon\left\|E_t\right\|_F^2 \tag{18}$$

holds with probability at least $1-\delta$. By combining (17) and (18) we see that if at least $4\eta^2 k/(\beta\epsilon^2)$ columns are picked in each of the t rounds then

$$\left\|E_t - ZZ^+E_t\right\|_F^2 \leq \left\|E_t - (E_t)_k\right\|_F^2 + \frac{\epsilon}{1-\epsilon}\left\|A - A_k\right\|_F^2 + \epsilon^t\left\|A\right\|_F^2 \tag{19}$$

holds with probability at least $1 - t\delta$. The theorem thus follows from (19) if we can establish that

$$\begin{aligned} E_t - ZZ^+E_t &= A - C^t\left(C^t\right)^+ A & (20)\\ \left\|E_t - (E_t)_k\right\|_F^2 &\leq \left\|A - A_k\right\|_F^2. & (21) \end{aligned}$$

But (20) follows from the definition of E_t, since $C^t\left(C^t\right)^+ = C^{t-1}\left(C^{t-1}\right)^+ + ZZ^+$ by the construction of Z, and since $ZZ^+C^{t-1}\left(C^{t-1}\right)^+ = \mathbf{0}$. To establish (21), and thus the theorem, notice that

$$\begin{aligned}
\left\|E_t - (E_t)_k\right\|_F^2 &= \left\|\left(I - C^{t-1}\left(C^{t-1}\right)^+\right)A - \left(\left(I - C^{t-1}\left(C^{t-1}\right)^+\right)A\right)_k\right\|_F^2 && (22)\\
&\le \left\|\left(I - C^{t-1}\left(C^{t-1}\right)^+\right)A - \left(I - C^{t-1}\left(C^{t-1}\right)^+\right)A_k\right\|_F^2 && (23)\\
&\le \left\|\left(I - C^{t-1}\left(C^{t-1}\right)^+\right)(A - A_k)\right\|_F^2 && (24)\\
&\le \left\|A - A_k\right\|_F^2 . && (25)
\end{aligned}$$

(22) follows by definition of E_t, (23) follows since $\left(I - C^{t-1}\left(C^{t-1}\right)^+\right)A_k$ is a rank k matrix, but not necessarily the optimal one, (24) follows immediately, and (25) follows since $I - C^{t-1}\left(C^{t-1}\right)^+$ is a projection.

$\diamond$

Remark: As contrasted with the algorithm and theorem of the previous subsection, this algorithm and theorem demonstrate that by sampling in t rounds and by judiciously computing sampling probabilities for picking columns of A in each of the t rounds, the overall error drops *exponentially* with t. This is a substantial improvement over the results of Theorem 2; in that case, if $c \ge 4\eta^2 kt/(\beta\epsilon^2)$ then the additional additive error is $(\epsilon/\sqrt{t})\left\|A\right\|_F^2$.

Remark: Rademacher, Vempala and Wang provided the first proof of a theorem in which the additional error drops exponentially with the number of passes [38]. In particular, they proved that there exists a rank k matrix in the subspace spanned by C that satisfies (in expectation) a bound of the form (15). Thus, by Markov's inequality, they obtain a bound of the form (15) that holds with probability at least $1 - \bar{\delta}$ if $c = O(t^2/\bar{\delta})$. Our proof is simpler, and we obtain (15) with probability at least $1 - \bar{\delta}$ if $c = O(t\log(t/\bar{\delta}))$.

3 Analysis and Discussion of the Main Tensor Approximation

In this section, our main result, which was summarized in Section 1.3, is analyzed in more detail. In Section 3.1, the ApproxTensorSVD algorithm (described in Algorithm 1 in Section 1.3) is elaborated upon. Then, in Section 3.2, the proof of Theorem 1, the main quality-of-approximation theorem for the ApproxTensorSVD algorithm (and which was also presented in Section 1.3), is presented. Finally, in Section 3.3, several remarks are made.

3.1 Elaboration on the Main Result

The ApproxTensorSVD algorithm (described in Algorithm 1 in Section 1.3) takes as input a d-mode tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$ and numbers $c_i, i = 1, \ldots, d$. The algorithm returns as output matrices $C_{[i]}$ of size $n_i \times c_i$ for all $i = 1, \ldots, d$. The algorithm works by choosing (for every $i \in \{1, \ldots, d\}$) c_i columns from the (not explicitly constructed) matrix $A_{[i]}$ to construct (explicitly) the $n_i \times c_i$ matrix $C_{[i]}$. The columns may be chosen from $A_{[i]}$ using either the SelectColumnsSinglePass algorithm (described in Section 2.1) or with the SelectColumnsMultiPass algorithm (described in Section 2.2). In the latter case a number t of iterations to perform is also input to the ApproxTensorSVD algorithm. In both cases, the matrix bound (10) from Theorem 2 (of Section 2.1) or (15) from Theorem 3 (of Section 2.2) will translate into a bound for the tensor approximation. Regardless of how the matrices $C_{[i]}$ are determined,

$$\tilde{\mathcal{A}} = \mathcal{A} \times_1 C_{[1]}C_{[1]}^+ \times_2 \cdots \times_d C_{[d]}C_{[d]}^+ \qquad (26)$$

is an approximation to the original tensor $\mathcal{A}$, and Theorem 1 provides a bound for a measure of the size of $\mathcal{A} - \tilde{\mathcal{A}}$.

3.2 Proof of Theorem 1

Let $\tilde{\mathcal{E}}_d = \tilde{\mathcal{E}} = \mathcal{A} - \tilde{\mathcal{A}}$, where $\tilde{\mathcal{A}}$ is the approximation (6), let $\tilde{\Pi}_i = C_{[i]}C_{[i]}^+$, for all $i = 1, \ldots, d$, and recall that $\tilde{\Pi}_i$ is an exact projection onto the column space of the matrix $C_{[i]}$. Then,

$$\begin{aligned}
\left\|\tilde{\mathcal{E}}_d\right\|_F &= \left\|\mathcal{A} - \mathcal{A} \times_d \tilde{\Pi}_d + \mathcal{A} \times_d \tilde{\Pi}_d - \mathcal{A} \times_1 \tilde{\Pi}_1 \times_2 \cdots \times_d \tilde{\Pi}_d\right\|_F & (27)\\
&\leq \left\|\mathcal{A} - \mathcal{A} \times_d \tilde{\Pi}_d\right\|_F + \left\|\left(\mathcal{A} - \mathcal{A} \times_1 \tilde{\Pi}_1 \times_2 \cdots \times_{d-1} \tilde{\Pi}_{d-1}\right) \times_d \tilde{\Pi}_d\right\|_F & (28)\\
&\leq \left\|\mathcal{A} - \mathcal{A} \times_d \tilde{\Pi}_d\right\|_F + \left\|\mathcal{A} - \mathcal{A} \times_1 \tilde{\Pi}_1 \times_2 \cdots \times_{d-1} \tilde{\Pi}_{d-1}\right\|_F, & (29)
\end{aligned}$$

where (28) follows by subadditivity and where (29) follows since the $\tilde{\Pi}_i$ are projection matrices, for all $i = 1, \ldots, d$. If we next define $\tilde{\mathcal{E}}_{d-1} = \mathcal{A} - \mathcal{A} \times_1 \tilde{\Pi}_1 \times_2 \cdots \times_{d-1} \tilde{\Pi}_{d-1}$, then we similarly see that

$$\left\|\tilde{\mathcal{E}}_{d-1}\right\|_F \leq \left\|\mathcal{A} - \mathcal{A} \times_{d-1} \tilde{\Pi}_{d-1}\right\|_F + \left\|\mathcal{A} - \mathcal{A} \times_1 \tilde{\Pi}_1 \times_2 \cdots \times_{d-2} \tilde{\Pi}_{d-2}\right\|_F. \quad (30)$$

Continuing in this manner, we can show that

$$\left\|\tilde{\mathcal{E}}\right\|_F \leq \sum_{i=1}^{d} \left\|\mathcal{A} - \mathcal{A} \times_i \tilde{\Pi}_i\right\|_F. \quad (31)$$

Since "unfolding" $\mathcal{A}$ along any mode does not change the value of its Frobenius norm (since it is simply a reordering of indices in a summation) it follows that, for every $i : 1 \leq i \leq d$,

$$\left\|\mathcal{A} - \mathcal{A} \times_i \tilde{\Pi}_i\right\|_F = \left\|A_{[i]} - \left(\mathcal{A} \times_i C_{[i]}C_{[i]}^+\right)_{[i]}\right\|_F. \quad (32)$$

Note that the Frobenius norm on the left hand side of (32) is a tensor norm and that the Frobenius norm on the right hand side of (32) is a matrix norm. It is this latter quantity that Theorems 2 and 3 bound. By applying the appropriate matrix bound from Theorem 2 or Theorem 3, the two statements of the theorem follow from (31) and (32).

3.3 Remarks on the Main Result

Remark: The error bounds (7) and (8) have a natural interpretation: projecting the tensor $\mathcal{A}$ onto an approximation to the best rank-k_i space across each of its modes incurs an error that depends on the sum of the residuals of the best low-rank approximations across each mode plus an additional error due to the degree to which the space spanned by the columns of $C_{[i]}$ is close to the space spanned by the top k singular vectors of $A_{[i]}$.

Remark: The ApproxTensorSVD algorithm may be viewed as computing an approximation to the best rank-(k_1,k_2,...,k_N) approximation to the tensor $\mathcal{A}$ [32, 33]. To see this, consider the TensorSVD algorithm (described in Algorithm 4). This algorithm takes as input a d-mode tensor $\mathcal{A}$ and numbers $k_i, i = 1, \ldots, d$ (which are the target ranks along each of the d modes), and it returns as output matrices $U_{[i],k_i}$ for $i = 1, \ldots, d$, where each $U_{[i],k_i}$ contains the top k_i singular vectors of $A_{[i]}$. In this case, if we define the error tensor

$$\mathcal{E} = \mathcal{A} - \mathcal{A} \times_1 U_{[1],k_1}U_{[1],k_1}^T \times_2 \cdots \times_d U_{[d],k_d}U_{[d],k_d}^T, \quad (33)$$

then by reasoning similar to that in the proof of Theorem 1 it can be shown that

$$\mathcal{E} \leq \sum_{i=1}^{d} \left\| A_{[i]} - \left(A_{[i]}\right)_{k_i} \right\|_F . \tag{34}$$

When applied to matrices, this bound is suboptimal by a factor of 2.

```
TensorSVD Algorithm
Data   : tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \cdots \times n_d}$, $k_i, 1 \leq k_i \leq n_i, i = 1, \ldots, d$.
Result : matrices $U_{[i],k_i} \in \mathbb{R}^{n_i \times k_i}$ for all $i = 1, \ldots, d$ such that

         $\mathcal{A} \approx \mathcal{A} \times_1 U_{[1],k_1} U_{[1],k_1}^T \times_2 U_{[2],k_2} U_{[2],k_2}^T \times_3 \cdots \times_d U_{[d],k_d} U_{[d],k_d}^T$.

for $i = 1, \ldots, d$ do
    Compute the top $k_i$ left singular vectors of $A_{[i]}$ and denote them by $U_{[i],k_i}$ ;
end
```

Algorithm 4: The TensorSVD Algorithm.

Remark: Assume, for simplicity, that the tensor $\mathcal{A}$ is stored externally and assume that $k_i = O(1)$ and that $n_i = n$ for every $i = 1, \ldots, d$. Then the matrices $C_{[i]}$ each occupy only $O(n)$ additional scratch space. In general, of course, $O(n^{d-1})$ additional scratch space will be needed to compute the probabilities of the form (9) and (14), and this will be comparable to the overall memory requirements. On the other hand, if the columns are chosen with the SelectColumnsSinglePass algorithm (Algorithm 2) and if the uniform probabilities are approximately optimal for each of the d nodes, then only $O(n)$ additional scratch space and computation time are needed by the ApproxTensorSVD algorithm (Algorithm 1), resulting in a substantial scratch memory and time savings. See [10] for additional discussion of resource issues within the framework of the Pass Efficient Model of data streaming computation.

Remark: Although sampling with respect to the proper probability distribution is critical for our provable results, one might expect that in many cases columns will all be approximately the same length due to the manner in which the data are generated, in which case uniform sampling may be successfully employed. This was seen to be the case for an application of our CUR algorithm (see [12] and Section 4.2) to kernel-based learning [14, 15, 45].

4 Restricting the Main Tensor Approximation to Matrices

In this section, we consider the special case of the ApproxTensorSVD algorithm when restricted to input tensors that are matrices, i.e., 2-mode tensors. In Section 4.1, we present the restriction of Theorem 1 to matrices. Then, in Section 4.2, we provide a discussion about how the approximate matrix decomposition generated by this restriction relates to several recently-developed low-rank matrix decompositions.

4.1 Approximating the Left and Right Singular Subspaces of a Matrix

When restricted to input tensor that are matrices, the ApproxTensorSVD algorithm (described in Algorithm 1 in Section 1.3) takes as input an $m \times n$ matrix A and two numbers, c and r. The algorithm returns as output two matrices, an $m \times c$ matrix C consisting of c columns of A and an $r \times n$ matrix R consisting of r rows of A. (For notational convenience, we have let $A = \mathcal{A}$ be the input matrix, and let $c = c_1$ and $r = c_2$.) The algorithm works by choosing c columns from A and r columns from A^T (or, equivalently, r rows from A). The columns and rows may be chosen from A using either the SelectColumnsSinglePass algorithm (described in Section 2.1) or with the SelectColumnsMultiPass algorithm (described in Section 2.2). In the latter case a number t of iterations to perform is also input to the ApproxTensorSVD algorithm.

When restricted to input tensors that are matrices, the approximation

$$\tilde{\mathcal{A}} = \mathcal{A} \times_1 C_{[1]} C_{[1]}^+ \times_2 C_{[2]} C_{[2]}^+ \tag{35}$$

may be written in usual matrix notation as

$$\tilde{A} = CC^+ AR^+ R. \tag{36}$$

As in the more general case of d-mode input tensors, quality-of-approximation bounds for the matrix A by the matrix $\tilde{A}$ will follow from the matrix bound (10) from Theorem 2 (of Section 2.1) or (15) from Theorem 3 (of Section 2.2). The following theorem bounds the error $A - \tilde{A}$ and is a corollary of Theorem 1.

Theorem 4 *Let $A \in \mathbb{R}^{m \times n}$ be a matrix, i.e., a 2-mode tensor, and let $\eta = 1 + \sqrt{(8/\beta)\log(1/\delta)}$, for any $0 < \delta < 1$. Let C and R be computed by the* ApproxTensorSVD *algorithm (Algorithm 1).*

- *If the columns and rows are chosen with the* SelectColumnsSinglePass *algorithm (Algorithm 2) then, with probability at least $1 - 2\delta$,*

$$\left\|A - CC^+ AR^+ R\right\|_F \leq 2\left\|A - A_k\right\|_F + 2\epsilon \left\|A\right\|_F, \tag{37}$$

  if $c, r \geq 4\eta^2 k/(\beta\epsilon^2)$.

- *If the columns and rows are chosen with the* SelectColumnsMultiPass *algorithm (Algorithm 3) then, with probability at least $1 - 2t\delta$,*

$$\left\|A - CC^+ AR^+ R\right\|_F \leq \frac{2}{1-\epsilon}\left\|A - A_k\right\|_F + 2\epsilon^t \left\|A\right\|_F, \tag{38}$$

  if $c, r \geq 4\eta^2 k/(\beta\epsilon^2)$ in each of the t passes.

4.2 Connections with Recently-Developed Low-Rank Matrix Decompositions

As described in Section 4.1, when our main algorithm is applied to an $m \times n$ matrix A rather than a general tensor $\mathcal{A}$, our main result consists of constructing an $m \times c$ matrix C and an $r \times n$ matrix R and approximating A by a matrix $\tilde{A}$ of the form

$$\tilde{A} = CC^+ AR^+ R = CUR, \tag{39}$$

where $U = C^+ AR^+$ is a $c \times r$ matrix. Thus, if c and r are constant (independent of m and n), then A is approximated by the product of three matrices, where the middle matrix is a constant-size matrix and the left and right matrices consist of a small number of columns and rows of A, respectively. More generally, a *CUR decomposition* is a low-rank matrix decomposition of the form $A \approx CUR$, where C is a matrix consisting of a small number of columns of A, R is a matrix consisting of a small number of rows of A, and U is an appropriately-defined low-dimensional matrix. Low-rank matrix decompositions with this structure, i.e., those expressed in terms of columns and rows of the original matrix, have been considered recently by Stewart [39, 40, 3], by Goreinov, Tyrtyshnikov, and Zamarashkin [20, 19], and by Drineas, Kannan, and Mahoney [9, 12, 14, 15].

Stewart was interested in computing sparse low-rank approximations to sparse matrices, and he developed the quasi-Gram-Schmidt method [39, 40, 3]. Given as input an $m \times n$ matrix A and a rank parameter k, this variant of the QR decomposition returns (in our notation) an $m \times k$ matrix C and a nonsingular upper-triangular $k \times k$ matrix S_C. The matrix C consists of k columns of A whose span approximates the column space of A, and the matrix S_C orthogonalizes these columns, i.e., is such that $C = Q_C S_C$, where the $m \times k$ matrix Q_C has orthogonal columns. A key feature of this algorithm is that the matrix Q_C is not explicitly computed, and it is not needed for certain computations (such as projections) since one can store S_C and C and use the relation $Q_C = CS_C^{-1}$ to recover the action of Q_C. The quasi-Gram-Schmidt method can be used to compute a sparse low-rank approximation to a sparse matrix A in the following manner. First, after applying it to A, apply it to A^T to get a $k \times n$ matrix R of rows of A and a nonsingular upper-triangular $k \times k$ matrix S_R such that $R^T = Q_R S_R$, where Q_R has orthogonal columns. Then, it can be shown that $A \approx CUR$, where the matrix U minimizing $\|A - CUR\|_F^2$ is of the form

$$\begin{aligned} U &= S_C^{-1} Q_C^T A Q_R S_R^{-T} && (40)\\ &= \left(S_C^T S_C\right)^{-1} C^T A R^T \left(S_R^T S_R\right)^{-1}. && (41) \end{aligned}$$

Several things should be noted about this result. First, the derivation of (41) assumes that the upper-triangular matrices S_C and S_R are invertible, which is guaranteed by the particular greedy strategy, based on the QR decomposition, that Stewart uses to select the columns and rows of A to be included in C and R, respectively. Second, with this U, the approximation takes the form

$$A \approx CUR = Q_C Q_C^T A Q_R Q_R^T. \qquad (42)$$

Thus, although Stewart forms his C (and R) by choosing columns (and rows) in a manner that guarantees their linear independence, whereas in this paper we randomly sample columns to form a C (and rows to form an R) that may be rank deficient, (39) and (42) indicate that both algorithms construct a low-rank approximation to a matrix A by projecting A onto the spaces spanned by C and R. Finally, an error bound provided by Stewart for the quality of approximation is

$$\|A - CUR\|_F^2 \leq \epsilon_C^2 + \epsilon_R^2, \qquad (43)$$

where ϵ_C and ϵ_R ($\sim \|A - A_k\|_F$) are errors associated with the column and row projections, respectively [39]. That this is likely to be an overestimate by a factor of approximately 2 [39, 3] is related to the suboptimality of (34) when applied to matrices and the multiplicative factor of 2 in in Theorem 4.

Goreinov, Tyrtyshnikov, and Zamarashkin [20, 19] were interested in applications such as scattering in which large coefficient matrices have blocks that can be easily approximated by low-rank matrices. They first note that if an $m \times n$ matrix A is exactly rank k then there exists a set of k columns of A, denoted by the $m \times k$ matrix C, and a set of k rows of A, denoted by the $k \times n$ matrix R, such that the $k \times k$ submatrix of A consisting of the intersection of those columns and those rows, denoted W, is nonsingular, and further that $A = CW^{-1}R$. They then show that if A is approximated by a rank-k matrix to within an accuracy ϵ, then for ϵ sufficiently small

$$\left\|A - CW^{-1}R\right\|_2 = O\left(\|A\|_2^2 \left\|W^{-1}\right\|_2^2 \epsilon\right)$$

provided that the matrix W is nonsingular. In both these cases, $U = W^{-1}$, but they note that this is clearly unsatisfactory since if W is ill-conditioned or singular then it may not be that $A \approx CW^{-1}R$ in any meaningful sense. Motivated by this, they define more general decompositions, which in our notation are of the form $A \approx CUR$ and which they call a pseudoskeleton component of A. In [20], they show that if the matrix A is approximated by a rank-k matrix to within an accuracy ϵ, i.e., if there exists a matrix E such that $\mathrm{rank}(A - E) \leq k$ and $\|E\|_2 \leq \epsilon$ (where ϵ does *not* need to be assumed to be sufficiently small) 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

$$\|A - CUR\|_2 \leq \epsilon \left(1 + 2\sqrt{km} + 2\sqrt{kn}\right) . \tag{44}$$

Although the primary goal of [20, 19] was not to analyze algorithms to construct the matrices C, R, and U, in [20] the choice for these matrices is related to the problem of determining the minimum singular value σ_k of $k \times k$ submatrices of $n \times k$ matrices with orthogonal columns. In addition, in [19] 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 maximim volume among all $k \times k$ submatrices of A.

In [12, 14, 15], Drineas, Kannan, and Mahoney describe the so-called CUR decomposition. In particular, the LinearTimeCUR and ConstantTimeCUR algorithms of [12] (so named due to their relationship with the correspondingly-named SVD algorithms of [11]) compute an approximation to an $m \times n$ matrix A by sampling c columns and r rows of the matrix A to form $m \times c$ and $r \times n$ matrices C and R, respectively. The matrices C and R are constructed by randomly sampling with carefully-chosen and data-dependent nonuniform probability distributions, and from C and R a $c \times r$ matrix U is constructed such that under appropriate assumptions:

$$\|A - CUR\|_\xi \leq \|A - A_k\|_\xi + \epsilon \|A\|_F , \tag{45}$$

with high probability, for both the spectral and Frobenius norms, $\xi = 2, F$. The form of the matrix U described in [12] permits bounds of the form (45) to be obtained, where, e.g., the suboptimal multiplicative factor of 2 is not present. In [14, 15] it is shown that if A is a symmetric positive semidefinite (SPSD) matrix, then one can choose $R = C^T$ and $U = W^+$, thus obtaining an approximation $A \approx \tilde{A} = CW^+C^T$. This approximation is SPSD and has provable bounds of the form (45), except that the scale of the additional additive error is somewhat larger [14, 15]. The CUR decompositions of [12, 14, 15] are quite similar in flavor to the algorithms discussed in this paper, and they have been applied to recommendation system analysis [13] and to kernel-based statistical learning [14, 15, 45]. See [14, 15] for a discussion of the CUR matrix decomposition, its relationship with recent work in learning theory, and its relationship to the Nyström method from integral equation theory.

5 Conclusion

We conclude by noting that tensor decompositions of the form we are considering have been studied in several data analysis areas. Depending on the processes generating a given data set, such a multilinear description may provide superior results relative to a simple linear description of the data. It has been argued, e.g., that data consisting of natural images consist of multiple factors, each of which has a linear property, and thus that the entire data set exhibits a multilinear property. For example, in [42, 43] the data consisted of images of p male subjects, each photographed in q poses, subject to r different illuminations, having s different expressions, and stored as a grayscale image with t pixils; this was represented as a $p \times q \times r \times s \times t$ tensor $\mathcal{A}$. In addition, higher-order tensors arise in higher-order statistics since for multivariate stochastic variables the basic higher-order statistics are symmetric higher-order tensors, in the same way as the covariance of a stochastic vector is a symmetric matrix. The use of these methods has been applied to independent component analysis by exploiting the statistical independence of the sources [32, 33, 31]. Finally, note that the model proposed by Tucker [41] as well as the related the "canonical decomposition" [5] or the "parallel factors" model [22] also provide decompositions for higher-order tensors and have a long history. They have received attention recently; see, e.g., [32, 28, 30, 34, 36] and references therein.

Acknowledgments We would like to thank Ravi Kannan for many fruitful discussions, Mark Tygert for constructive suggestions on the manuscript, and Efstratios Gallopoulos for pointing us to references [39, 40, 3, 46]. We would also like to thank the Institute for Pure and Applied Mathematics at UCLA and the American Institute of Mathematics for their generous hospitality.

References

[1] D. Achlioptas and F. McSherry. Fast computation of low rank matrix approximations. *Accepted for publication in the Journal of the A.C.M.*

[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*, pages 611–618, 2001.

[3] M.W. Berry, S.A. Pulatova, and G.W. Stewart. Computing sparse reduced-rank approximations to sparse matrices. Technical Report UMIACS TR-2004-32 CMSC TR-4589, University of Maryland, College Park, MD, 2004.

[4] G. Beylkin and M.J. Mohlenkamp. Numerical operator calculus in higher dimensions. *Proc. Natl. Acad. Sci. USA*, 99(16):10246–10251, 2002.

[5] J.D. Carroll and J.J. Chang. Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[6] W.F. de la Vega, R. Kannan, M. Karpinski, and S. Vempala. Tensor decomposition and approximation schemes for constraint satisfaction problems. In *Proceedings of the 37st Annual ACM Symposium on Theory of Computing*, pages 747–754, 2005.

[7] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. In *Proceedings of the 10th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 291–299, 1999.

[8] P. Drineas and R. Kannan. Fast Monte-Carlo algorithms for approximate matrix multiplication. In *Proceedings of the 42nd Annual IEEE Symposium on Foundations of Computer Science*, pages 452–459, 2001.

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

[10] P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo algorithms for matrices I: Approximating matrix multiplication. Technical Report YALEU/DCS/TR-1269, Yale University Department of Computer Science, New Haven, CT, February 2004. Accepted for publication in the SIAM Journal on Computing.

[11] P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. Technical Report YALEU/DCS/TR-1270, Yale University Department of Computer Science, New Haven, CT, February 2004. Accepted for publication in the SIAM Journal on Computing.

[12] P. Drineas, R. Kannan, and M.W. Mahoney. Fast Monte Carlo algorithms for matrices III: Computing a compressed approximate matrix decomposition. Technical Report YALEU/DCS/TR-1271, Yale University Department of Computer Science, New Haven, CT, February 2004. Accepted for publication in the SIAM Journal on Computing.

[13] P. Drineas, I. Kerenidis, and P. Raghavan. Competitive recommendation systems. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 82–90, 2002.

[14] 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*, pages 323–337, 2005.

[15] P. Drineas and M.W. Mahoney. On the Nyström method for approximating a Gram matrix for improved kernel-based learning. Technical Report YALEU/DCS/TR-1319, Yale University Department of Computer Science, New Haven, CT, April 2005.

[16] A. Frieze, R. Kannan, and S. Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, pages 370–378, 1998.

[17] A. Frieze, R. Kannan, and S. Vempala. Fast Monte Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51:1025–1041, 2004.

[18] T. Gonzalez and J. Ja'Ja'. On the complexity of computing bilinear forms with {0,1} constants. *Journal of Computer and System Sciences*, 20:77–95, 1980.

[19] S.A. Goreinov and E.E. Tyrtyshnikov. The maximum-volume concept in approximation by low-rank matrices. *Contemporary Mathematics*, 280:47–51, 2001.

[20] S.A. Goreinov, E.E. Tyrtyshnikov, and N.L. Zamarashkin. A theory of pseudoskeleton approximations. *Linear Algebra and Its Applications*, 261:1–21, 1997.

[21] W.H. Greub. *Multilinear Algebra.* Springer-Verlag, Berlin, 1967.

[22] R.A. Harshman and M.E. Lundy. The PARAFAC model for three-way factor analysis and multidimensional scaling. In H.G. Law, C.W. Snyder Jr., J. Hattie, and R.P. McDonald, editors, *Research Methods for Multimode Data Analysis*, pages 122–215. Praeger, 1984.

[23] J. Håstad. Tensor rank is NP-complete. *Journal of Algorithms*, 11(2):644–654, 1990.

[24] T.D. Howell. Global properties of tensor rank. *Linear Algebra and Its Applications*, 22:9–23, 1978.

[25] H.A.L. Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics*, 14:105–122, 2000.

[26] T.G. Kolda. Orthogonal tensor decompositions. *SIAM Journal on Matrix Analysis and Applications*, 23(1):243–255, 2001.

[27] T.G. Kolda. A counterexample to the possibility of an extension of the Eckart-Young low-rank approximation theorem for the orthogonal rank tensor decomposition. *SIAM Journal on Matrix Analysis and Applications*, 24(3):762–767, 2003.

[28] P.M. Kroonenberg and J. De Leeuw. Principal component analysis of three-mode data by means of alternating least squares algorithms. *Psychometrika*, 45(1):69–97, 1980.

[29] J.B. Kruskal. Three-way arrays: Rank and uniqueness of trilinear decompositions, with application to arithmetic complexity and statistics. *Linear Algebra and Its Applications*, 18:95–138, 1977.

[30] J.B. Kruskal. Rank, decomposition, and uniqueness for 3-way and N-way arrays. In R. Coppi and S. Bolasco, editors, *Multiway Data Analysis*, pages 7–18. Elesvier Science Publishers, 1989.

[31] L. De Lathauwer, B. De Moor, and J. Vandewalle. An introduction to independent component analysis. *Journal of Chemometrics*, 14:123–149, 2000.

[32] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear single value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000.

[33] L. De Lathauwer, B. De Moor, and J. Vandewalle. On the best rank-1 and rank-(R_1,R_2,...,R_N) approximation of higher-order tensors. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1324–1342, 2000.

[34] D. Leibovici and R. Sabatier. A singular value decomposition of a k-way array for a principal component analysis of multiway data, PTA-k. *Linear Algebra and Its Applications*, 269:307–329, 1998.

[35] L.-H. Lim. (unpublished results).

[36] L.-H. Lim and G.H. Golub. Tensors for numerical multilinear algebra: ranks and basic decompositions. Technical Report 05-02, Stanford University SCCM, Stanford, CA, April 2005.

[37] C.F. Van Loan. The ubiquitous Kronecker product. *Journal of Computational and Applied Mathematics*, 123:85–100, 2000.

[38] 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, March 2005.

[39] G.W. Stewart. Four algorithms for the efficient computation of truncated QR approximations to a sparse matrix. *Numerische Mathematik*, 83:313–323, 1999.

[40] 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.

[41] L.R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[42] M.A.O. Vasilescu and D. Terzopoulos. Multilinear analysis of image ensembles: TensorFaces. In *Proceedings of the 7th European Conference on Computer Vision*, pages 447–460, 2002.

[43] M.A.O. Vasilescu and D. Terzopoulos. Multilinear subspace analysis of image ensembles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 93–99, 2003.

[44] S. Vempala. Random projection: A new approach to VLSI layout. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science*, pages 389–395, 1998.

[45] C.K.I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Annual Advances in Neural Information Processing Systems 13: Proceedings of the 2000 Conference*, pages 682–688, 2001.

[46] D. Zeimpekis and E. Gallopoulos. CLSI: A flexible approximation scheme from clustered term-document matrices. Technical Report HPCLAB-SCG 2/10-04, University of Patras, Patras, Greece, October 2004.

[47] T. Zhang and G.H. Golub. Rank-one approximation to high order tensors. *SIAM Journal on Matrix Analysis and Applications*, 23(2):534–550, 2001.