A fast randomized algorithm for the approximation of matrices — preliminary report*

Yale Department of Computer Science Technical Report #1380
Franco Woolfe, Edo Liberty, Vladimir Rokhlin, and Mark Tygert

April 30, 2007

Abstract

Given an $m \times n$ matrix A and a positive integer k, we introduce a randomized procedure for the approximation of A with a matrix Z of rank k. The procedure relies on applying an $l \times m$ random matrix with special structure to each column of A, where l is an integer near to, but greater than k. The spectral norm $\|A - Z\|$ of the discrepancy between A and Z is of the same order as $\sqrt{lm}$ times the $(k+1)^{\mathrm{st}}$ greatest singular value σ_{k+1} of A, with small probability of large deviations. The special structure of the $l \times m$ random matrix allows us to apply it to an arbitrary $m \times 1$ vector at a cost proportional to $m \log(l)$. Utilizing this special structure, the algorithm constructs the rank-k approximation Z from the entries of A at a cost proportional to $mn \log(k) + l^2(m+n)$. In contrast, the classical pivoted "$Q\,R$" decomposition algorithms such as Gram-Schmidt cost at least kmn. If l is significantly less than m and n, then the randomized algorithm tends to cost less than the classical algorithms; moreover, the constant of proportionality in the cost of the randomized algorithm appears to be small enough so that the randomized algorithm is at least as efficient as the classical algorithms even when k is quite small. Thus, given a matrix A of limited numerical rank, the scheme provides an efficient means of computing an accurate approximation to the singular value decomposition of A. Furthermore, the algorithm presented here operates reliably independently of the structure of the matrix A. The results are illustrated via several numerical examples.

1 Introduction

In many practical applications it is important to be able to construct a low-rank approximation to a matrix A. Such an approximation of A often facilitates the analysis of A. Furthermore, such a low-rank approximation facilitates rapid calculations involving A.

There exist at least two classical forms of such matrix approximations. One is an approximation to a singular value decomposition (SVD), which is known in the statistical literature as a principal component analysis (PCA). The other is an approximation obtained

*Partially supported by the AFOSR and the NGA.

via subset selection; we will refer to the matrix representation obtained via subset selection as an interpolative decomposition. These two types of matrix approximations are defined as follows.

An approximation to an SVD of a real $m \times n$ matrix A consists of nonnegative real numbers $\sigma_1, \sigma_2, \ldots, \sigma_{k-1}, \sigma_k$ known as singular values, orthonormal real $m \times 1$ column vectors $u^1, u^2, \ldots, u^{k-1}, u^k$ known as left singular vectors, and orthonormal real $n \times 1$ column vectors $v^1, v^2, \ldots, v^{k-1}, v^k$ known as right singular vectors, such that

$$\left\| A - \sum_{j=1}^{k} u^j \, \sigma_j \, (v^j)^{\mathrm{T}} \right\| \leq \delta, \tag{1}$$

where k, m, and n are positive integers with $k < m$ and $k < n$, δ is a positive real number specifying the precision of the approximation, and, for any matrix B, $\|B\|$ denotes the spectral (l^2-operator) norm of B, that is, $\|B\|$ is the greatest singular value of B. An approximation to an SVD of A is often written in the equivalent form

$$\left\| A - U \, \Sigma \, V^{\mathrm{T}} \right\| \leq \delta, \tag{2}$$

where U is a real $m \times k$ matrix whose columns are orthonormal, V is a real $n \times k$ matrix whose columns are orthonormal, and Σ is a real diagonal $k \times k$ matrix whose entries are all nonnegative. See, for example, [11] for a discussion of SVDs.

An interpolative decomposition of a real $m \times n$ matrix A consists of a real $m \times k$ matrix B whose columns constitute a subset of the columns of A, and a real $k \times n$ matrix P, such that

1. some subset of the columns of P makes up the $k \times k$ identity matrix,
2. no entry of P has an absolute value greater than 2, and
3. $A = B \, P$.

See, for example, [9], [3], [10], [6], [14], or Sections 4 and 5 of [2] for a discussion of interpolative decompositions.

The present article introduces an algorithm for the computation of a low-rank approximation of either type to an arbitrary dense matrix that is generally at least as efficient as pivoted Gram-Schmidt and the other classical pivoted "$Q\,R$" algorithms, and often substantially more efficient. In order to construct a nearly optimal rank-k approximation to a dense real $n \times n$ matrix, pivoted Gram-Schmidt with reorthogonalization, pivoted "$Q\,R$" based on plane (Householder) reflections, pivoted "$Q\,R$" based on plane (Givens) rotations, and the algorithm of [7] all require at least

$$\mathcal{O}(kn^2) \tag{3}$$

floating-point operations and words of memory (see, for example, Chapter 5 in [5]). In contrast, the algorithm of Subsection 5.2 of the present paper requires

$$\mathcal{O}(n^2 \, \log(k) + l^2 \, n) \tag{4}$$

floating-point operations and words of memory in order to construct a similarly nearly optimal rank-k approximation to a dense real $n \times n$ matrix, where l is an integer near to, but greater than k. When k is significantly less than n, (4) is less than (3). Moreover, the constant multiplying $n^2 \log(k)$ in (4) that is hidden by the big-$\mathcal{O}$ is about as small as the constant multiplying $n \log(n)$ in the cost of an application of a fast Fourier transform to a real $n \times 1$ vector (see, for example, [11] for information concerning the fast Fourier transform). The scheme of the present paper also appears to parallelize naturally.

Unlike the classical algorithms, the algorithm of the present paper is a randomized one, and fails with a small probability. However, we can determine rapidly whether the algorithm has succeeded, and run the algorithm again with an independent realization of the random variables involved, in effect boosting the probability of success at a reasonable additional expected cost.

The algorithm described below would seem to be preferable to the classical algorithms for the computation of low-rank approximations to a dense matrix, and (more or less equivalently) for the computation of a few of the greatest singular values of a matrix and their corresponding singular vectors.

The algorithm of [10] is very similar to the algorithm of the present paper. The core steps of both algorithms involve the rapid computation of the product of a random matrix and the matrix to be approximated. The algorithm of [10] assumes that the matrix to be approximated (and its transpose) may be applied rapidly to arbitrary vectors, thus enabling the rapid computation of the product of the matrix to be approximated (or its transpose) and any matrix. In contrast, the algorithm of the present paper utilizes a random matrix which may be applied rapidly to arbitrary vectors, thus enabling the rapid computation of the product of the random matrix and any matrix.

The random matrix employed in the present paper consists of several uniformly randomly selected rows of the product of the discrete Fourier transform matrix (or its close relative, the discrete Walsh-Hadamard transform matrix) and a random diagonal matrix. The fast Fourier transform and similar algorithms allow the rapid application of this random matrix to arbitrary vectors (see, for example, [11] for a discussion of the fast Fourier transform algorithm and its applications). The idea of using a random matrix with such special structure has been introduced in [1]. The idea of using such a random matrix in numerical linear algebra (specifically, for the purpose of computing a solution in the least squares sense to an overdetermined system of linear algebraic equations) has been introduced in [12], utilizing both [1] and [4].

For simplicity, we discuss only real matrices; the analysis below extends easily to the complex case. The present paper has the following structure: Section 2 collects together various known facts which later sections utilize. Section 3 introduces the trimmed Walsh-Hadamard transform, a somewhat more efficient variant of the classical fast Fourier transform. Section 4 provides the principal lemma which Section 5 uses to construct algorithms. Section 5 describes the algorithm of the present paper, providing details about its accuracy and computational costs. Section 6 illustrates the algorithm via several numerical examples.

2 Preliminaries

In this section, we summarize various facts from linear algebra.

In the present section and throughout the rest of the paper, we employ the following notation. We will denote an identity matrix by $\mathbf{1}$, and a matrix whose entries are all zero by $\mathbf{0}$. For any matrix A, we define the norm $\|A\|$ of A to be the spectral (l^2-operator) norm of A, that is, $\|A\|$ is the greatest singular value of A. We define δ to be the Kronecker symbol, that is, the matrix with the entry

$$\delta_{nn} = 1 \tag{5}$$

for any positive integer n, and

$$\delta_{mn} = 0 \tag{6}$$

for any positive integers m and n with $m \neq n$.

The following lemma states that, for any $m \times n$ matrix A whose rank is k, where k, m, and n are positive integers, there exist an $m \times k$ matrix B whose columns constitute a subset of the columns of A, and a $k \times n$ matrix P, such that

1. some subset of the columns of P makes up the $k \times k$ identity matrix,
2. P is not too large, and
3. $B\,P = A$.

Moreover, the lemma provides an analogous approximation $B\,P$ to A when the exact rank of A is not k, but the $(k+1)^{\text{st}}$ singular value of A is nevertheless small. The lemma is a reformulation of Theorem 3.2 in [9] and Theorem 3 in [3].

Lemma 2.1 *Suppose that m and n are positive integers, and A is a real $m \times n$ matrix.*

Then, for any positive integer k with $k \leq m$ and $k \leq n$, there exist a real $k \times n$ matrix P, and a real $m \times k$ matrix B whose columns constitute a subset of the columns of A, such that

1. *some subset of the columns of P makes up the $k \times k$ identity matrix,*
2. *no entry of P has an absolute value greater than* 1,
3. $\|P\| \leq \sqrt{k\,(n-k)+1}$,
4. *the least (that is, the k^{th} greatest) singular value of P is at least* 1,
5. $B\,P = A$ *when* $k = m$ *or* $k = n$, *and*
6. $\|B\,P - A\| \leq \sqrt{k\,(n-k)+1}\;\sigma_{k+1}$ *when $k < m$ and $k < n$, where σ_{k+1} is the $(k+1)^{st}$ greatest singular value of A.*

Remark 2.2 Properties 1, 2, 3, and 4 in Lemma 2.1 ensure that the interpolative decomposition $B\,P$ of A is numerically stable. Also, Property 3 follows directly from Properties 1 and 2, and Property 4 follows directly from Property 1.

Observation 2.3 *There exists an algorithm which computes B and P in Lemma 2.1 from A, provided that we require only that*

1. *some subset of the columns of P makes up the $k \times k$ identity matrix,*
2. *no entry of P has an absolute value greater than 2,*
3. $\|P\| \le \sqrt{4k\,(n-k)+1}$,
4. *the least (that is, the k^{th} greatest) singular value of P is at least 1,*
5. $B\,P = A$ *when $k = m$ or $k = n$, and*
6. $\|B\,P - A\| \le \sqrt{4k\,(n-k)+1}\;\sigma_{k+1}$ *when $k < m$ and $k < n$, where σ_{k+1} is the $(k+1)^{\text{st}}$ greatest singular value of A.*

For any positive real number ε, the algorithm can identify the least k such that $\|B\,P - A\| \approx \varepsilon$. Furthermore, there exists a real number C such that the algorithm computes both B and P using at most $C kmn \log(n)$ floating-point operations and $C kmn$ floating-point words of memory. The algorithm is based upon the Cramer rule and the ability to obtain the minimal-norm (or at least roughly minimal-norm) solutions to linear algebraic systems of equations (see [9], [3], and [7]).

The following lemma provides an approximation $Q\,S$ to an $n \times l$ matrix T via an $n \times k$ matrix Q whose columns are orthonormal, and a $k \times l$ matrix S.

Lemma 2.4 *Suppose that k, l, and n are positive integers with $k < l$ and $l \le n$, and T is a real $n \times l$ matrix.*

Then, there exist a real $n \times k$ matrix Q whose columns are orthonormal, and a real $k \times l$ matrix S, such that

$$\|Q\,S - T\| \le \tau_{k+1}, \tag{7}$$

where τ_{k+1} is the $(k+1)^{st}$ greatest singular value of T.

Proof. We start by forming an SVD of T,

$$T = U\,\Sigma\,V^{\mathrm{T}}, \tag{8}$$

where U is a real $n \times l$ matrix whose columns are orthonormal, V is a real $l \times l$ matrix whose columns are orthonormal, and Σ is a real $l \times l$ matrix whose entries are nonnegative everywhere and zero off of the main diagonal, such that

$$\Sigma_{j,j} = \rho_j \tag{9}$$

for all $j = 1, 2, \ldots, l-1, l$, where $\Sigma_{j,j}$ is the entry in row j and column j of Σ, and τ_j is the j^{th} greatest singular value of T. We define Q to be the leftmost $n \times k$ block of U, and P to be the rightmost $n \times (l-k)$ block of U, so that

$$U = \left(\begin{array}{c:c} Q & P \end{array}\right). \tag{10}$$

We define S to be the uppermost $k \times l$ block of ΣV^{T}, and R to be the lowermost $(l-k) \times l$ block of ΣV^{T}, so that

$$\Sigma V^{\mathrm{T}} = \left(\begin{array}{c} S \\ \hdashline R \end{array} \right). \tag{11}$$

Combining (8), (9), (10), (11), and the fact that the columns of U are orthonormal, as are the columns of V, yields (7). $\square$

Observation 2.5 *In order to compute the matrices Q and S in (7) from the matrix T, we can construct (8), and then form Q and S according to (10) and (11). (See, for example, Chapter 8 in [5] for details concerning the computation of the SVD.)*

The following lemma provides an efficient means of computing an SVD of a real $m \times n$ matrix A from a real $m \times k$ matrix B and a real $k \times n$ matrix P such that $A = B\,P$ and k is much less than both m and n. If, in addition, $\|B\| \le \|A\|$ and $\|P\|$ is not too large, then the scheme described by the lemma is numerically stable. Please note that if B and P arise from an interpolative decomposition, then indeed $\|B\| \le \|A\|$ and $\|P\|$ is not too large, and so the scheme described by the lemma is numerically stable.

Lemma 2.6 *Suppose that k, m, and n are positive integers with $k \le n$. Suppose further that A is a real $m \times n$ matrix, B is a real $m \times k$ matrix, and P is a real $k \times n$ matrix, such that*

$$A = B\,P. \tag{12}$$

Suppose in addition that L is a real $k \times k$ matrix, and Q is a real $n \times k$ matrix whose columns are orthonormal, such that

$$P = L\,Q^{\mathrm{T}}. \tag{13}$$

Suppose finally that C is a real $m \times k$ matrix, U is a real $m \times k$ matrix whose columns are orthonormal, Σ is a real $k \times k$ matrix, and W is a real $k \times k$ matrix whose columns are orthonormal, such that

$$C = B\,L \tag{14}$$

and

$$C = U\,\Sigma\,W^{\mathrm{T}}. \tag{15}$$

Then,

$$A = U\,\Sigma\,V^{\mathrm{T}}, \tag{16}$$

where V is the real $n \times k$ matrix given by the formula

$$V = Q\,W. \tag{17}$$

Moreover, the columns of V are orthonormal (as are the columns of U), and

$$\|L\| = \|P\|. \tag{18}$$

Proof. Combining (12), (13), (14), (15), and (17) yields (16). Combining (17) and the facts that W is unitary and the columns of Q are orthonormal yields that the columns of V are orthonormal. Combining (13) and the fact that the columns of Q are orthonormal yields (18). □

Remark 2.7 The matrices L and Q in (13) can be computed from P as follows. Using the algorithms described, for example, in Chapter 5 of [5], we construct an upper triangular real $k \times k$ matrix R, and a real $n \times k$ matrix Q whose columns are orthonormal, such that

$$P^{\mathrm{T}} = Q\,R. \tag{19}$$

We thus obtain Q. We then define L to be the transpose of R, that is,

$$L = R^{\mathrm{T}}. \tag{20}$$

3 Fast transform algorithms

In this section, we describe two simple variants of the fast Fourier transform algorithm (see, for example, [11] for a discussion of the fast Fourier transform algorithm and its applications, or the subsection on Walsh wavelet packets in [8] for a description of the fast Walsh-Hadamard transform). In Subsection 3.1, we discuss the classical fast Walsh-Hadamard transform. In Subsection 3.2, we describe how to compute rapidly randomly selected entries of the output of the Walsh-Hadamard transform.

3.1 Fast Walsh-Hadamard transform

Given any integer n that is a positive integer power of 2, and a real $n \times 1$ vector x, we define the Walsh-Hadamard transform $H^{(n)}\,x$ of x inductively, as follows. We define y to be the real $n \times 1$ vector whose first $\frac{n}{2}$ entries are the sums of the successive pairs of the entries of x, and whose last $\frac{n}{2}$ entries are the differences of the successive pairs of the entries of x, that is,

$$y_k = x_{2k-1} + x_{2k} \tag{21}$$

for $k = 1, 2, \ldots, \frac{n}{2} - 1, \frac{n}{2}$, and

$$y_{k+n/2} = x_{2k-1} - x_{2k} \tag{22}$$

for $k = 1, 2, \ldots, \frac{n}{2} - 1, \frac{n}{2}$. We then define z to be the real $n \times 1$ vector given by the formula

$$z = \frac{1}{\sqrt{2}}\,y. \tag{23}$$

Then, the Walsh-Hadamard transform $w = H^{(n)}\,x$ of x is the real $n \times 1$ vector whose first $\frac{n}{2}$ entries consist of the Walsh-Hadamard transform $H^{(n/2)}$ of the first $\frac{n}{2}$ entries of z, and whose last $\frac{n}{2}$ entries consist of the Walsh-Hadamard transform $H^{(n/2)}$ of the last $\frac{n}{2}$ entries of z, that is,

$$H^{(n)}\,x = \left(\begin{array}{c|c} H^{(n/2)} & \mathbf{0} \\ \hline \mathbf{0} & H^{(n/2)} \end{array}\right) z. \tag{24}$$

Clearly, the Walsh-Hadamard transform $H^{(n)}$ is linear and unitary, and the absolute value of every entry of $H^{(n)}$ is $\frac{1}{\sqrt{n}}$.

If we compute the Walsh-Hadamard transform as we have defined it, then we must perform $\frac{n}{2}$ additions, $\frac{n}{2}$ subtractions, and n multiplications, in addition to two Walsh-Hadamard transforms of size $\frac{n}{2}$. Thus, we must perform at most proportional to $n \log_2(n)$ floating-point operations in order to compute the Walsh-Hadamard transform $H^{(n)}\, x$ of a real $n \times 1$ vector x.

Figure 1 provides a graphical depiction of the fast Walsh-Hadamard transform.

3.2 Trimmed fast Walsh-Hadamard transform

Suppose that we wish to compute only k randomly chosen entries of $H^{(n)}\, x$. Each of these k entries is formed as $\frac{1}{\sqrt{2}}$ times a sum of a pair of entries from the previous level in the transform, as well as $\frac{1}{\sqrt{2}}$ times a difference of a pair of entries (see Figure 2). Hence, at most $2k$ entries from the previous level in the transform contribute to the k entries of $H^{(n)}\, x$ that interest us. Similarly, at most $2^2 k$ entries from the third-to-last level contribute to these $2k$ entries from the second-to-last level, and at most $2^3 k$ entries contribute from the fourth-to-last level. This potential doubling of the number of contributing entries continues until all of the entries of a level must be filled, namely when at most $n = 2^{\log_2(n/k)} k$ entries contribute, which happens by the $\log_2\left(\frac{2n}{k}\right)^{\text{th}}$-to-last level. Thus, the number of entries within the Walsh-Hadamard transform tree from the last level to the $\log_2\left(\frac{2n}{k}\right)^{\text{th}}$-to-last level which we need to compute is at most

$$2^{\log_2(n/k)} k + 2^{\log_2(n/k)-1} k + \cdots + 2^2 k + 2k + k < 2\, 2^{\log_2(n/k)} k = 2n. \tag{25}$$

The Walsh-Hadamard transform tree includes at most $\log_2(2n)$ levels in total, however. Although the number of entries within the tree from the last level to the $\log_2\left(\frac{2n}{k}\right)^{\text{th}}$-to-last level which we need to compute is at most $2n$, there could be $\log_2(2n) - \log_2\left(\frac{2n}{k}\right) = \log_2(k)$ levels at the beginning of the tree in which we would need to compute all of the n entries per level. Therefore, in order to compute the k entries of $H^{(n)}\, x$ which interest us, we might need to compute $n \log_2(k)$ entries in the beginning levels of the tree, in addition to the at most $2n$ entries in the last levels of the tree. Overall, then, we can compute k randomly chosen entries of $H^{(n)}\, x$ using at most proportional to $n \log_2(k) + 2n$ floating-point operations.

4 Mathematical apparatus

In this section, we describe the principal mathematical tool used in Section 5.

With the choice $\alpha = 8$ and $\beta = 2$, the following lemma shows that, with probability at least $\frac{1}{2}$, the least singular value of the real $k \times l$ matrix $U\, D\, H$ is at least $\sqrt{\frac{l}{8n}}$, and the greatest singular value is at most $\sqrt{\frac{15l}{8n}}$, where D is a real diagonal $n \times n$ matrix, whose diagonal entries $d_1, d_2, \ldots, d_{n-1}, d_n$ are i.i.d., taking values in $\{-1, 1\}$ uniformly at random, H is a real $n \times l$ matrix whose j^{th} column is the s_j^{th} column of the Walsh-Hadamard transform $H^{(n)}$ defined in Subsection 3.1, with $s_1, s_2, \ldots, s_{l-1}, s_l$ being i.i.d., and taking values in

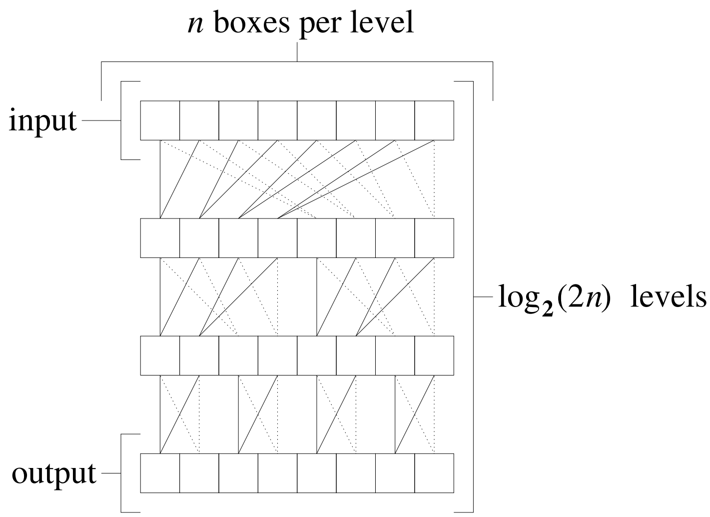

Figure 1 (depicting the Walsh-Hadamard transform):
The value in each box is $\frac{1}{\sqrt{2}}$ times the sum of the values in the boxes above it which connect to it via solid lines, and is $\frac{1}{\sqrt{2}}$ times the difference of the values in the boxes above it which connect to it via dotted lines. There are $\log_2(2n)$ levels in the whole tree (here $n = 8$).

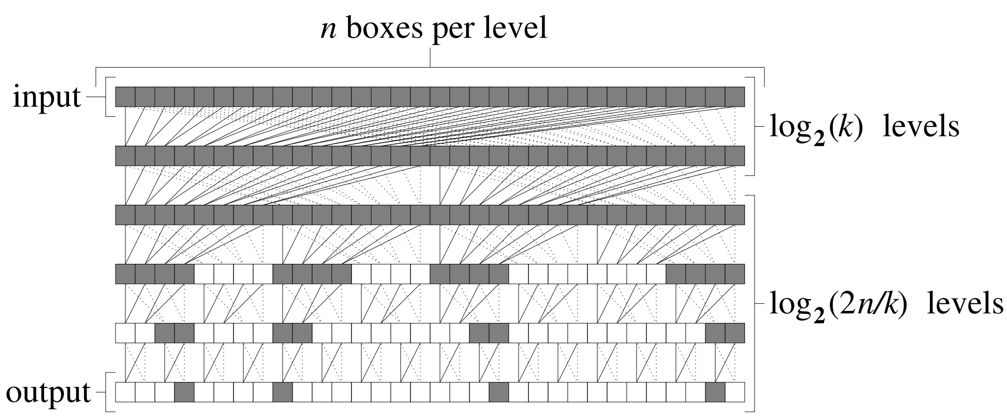

Figure 2 (depicting the trimmed Walsh-Hadamard transform):
The value in each box is $\frac{1}{\sqrt{2}}$ times the sum of the values in the boxes above it which connect to it via solid lines, and is $\frac{1}{\sqrt{2}}$ times the difference of the values in the boxes above it which connect to it via dotted lines. Only the $k = 4$ values in the darkened boxes at the last level in the diagram are desired, requiring the computation of the values in all of the darkened boxes in all levels in the diagram, but not in any of the empty boxes. There are $\log_2(2n)$ levels in the whole tree (here $n = 32$), with $\log_2(k)$ levels at the beginning of the tree consisting entirely of darkened boxes.

$\{1, 2, \ldots, n-1, n\}$ uniformly at random, U is a real $k \times n$ matrix whose rows are orthonormal, and $n > l \geq 3(k^2+k)$. This lemma is similar to the subspace Johnson-Lindenstrauss lemma (Corollary 11) of [13].

Lemma 4.1 *Suppose that α and β are real numbers greater than 1, and k, l, and n are positive integers, such that n is a positive integer power of 2, and*

$$n > l \geq \frac{\alpha^2 \beta}{(\alpha - 1)^2} (k^2 + k). \tag{26}$$

Suppose further that D is a real diagonal $n \times n$ matrix, whose diagonal entries d_1, d_2, $\ldots$, d_{n-1}, d_n are i.i.d., taking values in $\{-1, 1\}$ uniformly at random. Suppose in addition that H is a real $n \times l$ matrix whose j^{th} column is the s_j^{th} column of the Walsh-Hadamard transform $H^{(n)}$ defined in Subsection 3.1, with s_1, s_2, $\ldots$, s_{l-1}, s_l being i.i.d., and taking values in $\{1, 2, \ldots, n-1, n\}$ uniformly at random. Suppose finally that U is a real $k \times n$ matrix whose rows are orthonormal, and that C is the real $k \times k$ matrix defined via the formula

$$C = (U\,D\,H)\,(U\,D\,H)^{\mathrm{T}}. \tag{27}$$

Then, the least (that is, the k^{th} greatest) singular value σ_k of $U\,D\,H$ satisfies

$$\frac{1}{\sigma_k} = \sqrt{\|C^{-1}\|} \leq \sqrt{\frac{\alpha\, n}{l}} \tag{28}$$

and (simultaneously) the greatest singular value σ_1 of $U\,D\,H$ satisfies

$$\sigma_1 = \sqrt{\|C\|} \leq \sqrt{\frac{l}{n}\left(2 - \frac{1}{\alpha}\right)} \tag{29}$$

with probability at least $1 - \frac{1}{\beta}$.

Proof. For any integers a and p with $1 \leq a \leq k$ and $1 \leq p \leq k$,

$$C_{pa} = \sum_{r=1}^{l} \sum_{q=1}^{n} U_{pq}\, d_q\, H_{qs_r} \sum_{b=1}^{n} U_{ab}\, d_b\, H_{bs_r}. \tag{30}$$

Defining

$$G_{qbs} = H_{qs}\, H_{bs} \tag{31}$$

and

$$F_{qb} = \sum_{r=1}^{l} G_{qbs_r}, \tag{32}$$

we obtain from (30) that

$$C_{pa} = \sum_{q=1}^{n} U_{pq}\, U_{aq}\, (d_q)^2\, F_{qq} + \sum_{q=1}^{n} U_{pq}\, d_q \sum_{b \neq q} U_{ab}\, d_b\, F_{qb}. \tag{33}$$

Combining (31) and the fact that the rows of $H^{(n)}$ are orthonormal yields that

$$\sum_{s=1}^{n} G_{qqs} = 1, \tag{34}$$

and

$$\sum_{s=1}^{n} G_{qbs} = 0 \tag{35}$$

when $q \neq b$.

Combining (31) and the fact that the absolute value of any entry of $H^{(n)}$ is $\frac{1}{\sqrt{n}}$ yields that

$$G_{qqs} = \frac{1}{n}. \tag{36}$$

Combining (32) and (36) yields that

$$F_{qq} = \frac{l}{n}. \tag{37}$$

Combining (31), (35), the fact that the absolute value of any entry of $H^{(n)}$ is $\frac{1}{\sqrt{n}}$, and the fact that s_1, s_2, …, s_{l-1}, s_l are i.i.d., taking values uniformly at random in $\{1, 2, \ldots, n-1, n\}$, yields that G_{qbs_1}, G_{qbs_2}, …, $G_{qbs_{l-1}}$, G_{qbs_l} are i.i.d., taking values uniformly at random in $\left\{-\frac{1}{n}, \frac{1}{n}\right\}$, when $q \neq b$. Combining this last fact and (32) yields that

$$\mathbf{E}\, F_{qb} = 0 \tag{38}$$

and

$$\mathbf{E}\, (F_{qb})^2 = \frac{l}{n^2} \tag{39}$$

for $q, b = 1, 2, \ldots, n-1, n$ such that $q \neq b$.

Combining (33), (37), the fact that $(d_q)^2 = 1$, and the fact that the rows of U are orthonormal, yields that

$$C_{pp} = \frac{l}{n} + \sum_{q=1}^{n} U_{pq}\, d_q \sum_{b \neq q} U_{pb}\, d_b\, F_{qb}, \tag{40}$$

and

$$C_{pa} = \sum_{q=1}^{n} U_{pq}\, d_q \sum_{b \neq q} U_{ab}\, d_b\, F_{qb} \tag{41}$$

when $p \neq a$.

We define Δ to be the real diagonal $k \times k$ matrix with the diagonal entry

$$\Delta_{pp} = \frac{l}{n} \tag{42}$$

and E to be the real $k \times k$ matrix with the entry

$$E_{pa} = \sum_{q=1}^{n} U_{pq}\, d_q \sum_{b \neq q} U_{ab}\, d_b\, F_{qb} \tag{43}$$

for $p, a = 1, 2, \ldots, k-1, k$. Combining (40), (41), (42), and (43) yields that

$$C = \Delta + E = (\mathbf{1} + E\,\Delta^{-1})\,\Delta. \tag{44}$$

We now show that $\|E\,\Delta^{-1}\| \le 1 - \frac{1}{\alpha}$ with probability at least $1 - \frac{1}{\beta}$.

Now,

$$\mathbf{E}\left(\sum_{q=1}^{n} U_{pq}\,d_q \sum_{b\neq q} U_{ab}\,d_b\,F_{qb}\right)^2 = \mathbf{E}\left(\sum_{q=1}^{n} U_{pq}\,d_q \sum_{b\neq q} U_{ab}\,d_b\,F_{qb}\right)\left(\sum_{r=1}^{n} U_{pr}\,d_r \sum_{c\neq r} U_{ac}\,d_c\,F_{rc}\right). \tag{45}$$

Performing the summation over q and r separately for the cases when $q = r$ and when $q \neq r$, and using the fact that $(d_q)^2 = 1$, we obtain

$$\begin{aligned} &\mathbf{E}\sum_{q,r=1}^{n}\left(U_{pq}\,d_q \sum_{b\neq q} U_{ab}\,d_b\,F_{qb}\right)\left(U_{pr}\,d_r \sum_{c\neq r} U_{ac}\,d_c\,F_{rc}\right) \\ &\quad = \mathbf{E}\sum_{q=1}^{n}(U_{pq})^2\left(\sum_{b\neq q} U_{ab}\,d_b\,F_{qb}\right)^2 + \mathbf{E}\sum_{q\neq r} U_{pq}\,U_{pr}\,d_q\,d_r \sum_{b\neq q} U_{ab}\,d_b\,F_{qb} \sum_{c\neq r} U_{ac}\,d_c\,F_{rc}. \end{aligned} \tag{46}$$

To bound the first term in the right-hand side of (46), we observe that

$$\mathbf{E}\sum_{q=1}^{n}(U_{pq})^2\left(\sum_{b\neq q} U_{ab}\,d_b\,F_{qb}\right)^2 = \sum_{q=1}^{n}(U_{pq})^2\;\mathbf{E}\left(\sum_{b\neq q} U_{ab}\,d_b\,F_{qb}\right)^2. \tag{47}$$

But,

$$\mathbf{E}\left(\sum_{b\neq q} U_{ab}\,d_b\,F_{qb}\right)^2 = \mathbf{E}\sum_{b\neq q} U_{ab}\,d_b\,F_{qb} \sum_{c\neq q} U_{ac}\,d_c\,F_{qc}. \tag{48}$$

Moreover,

$$\mathbf{E}\sum_{b\neq q} U_{ab}\,d_b\,F_{qb} \sum_{c\neq q} U_{ac}\,d_c\,F_{qc} = \sum_{b,c\neq q} U_{ab}\,U_{ac}\,\mathbf{E}\,d_b\,d_c\,F_{qb}\,F_{qc}. \tag{49}$$

Performing the summation over b and c separately for the cases when $b = c$ and when $b \neq c$, and using the fact that $(d_b)^2 = 1$, we obtain

$$\sum_{b,c\neq q} U_{ab}\,U_{ac}\,\mathbf{E}\,d_b\,d_c\,F_{qb}\,F_{qc} = \sum_{b\neq q}(U_{ab})^2\,\mathbf{E}\,(F_{qb})^2 + \sum_{b,c\neq q \text{ and } b\neq c} U_{ab}\,U_{ac}\,\mathbf{E}\,d_b\,d_c\,F_{qb}\,F_{qc}. \tag{50}$$

It follows from (39) that

$$\sum_{b\neq q}(U_{ab})^2\,\mathbf{E}\,(F_{qb})^2 = \frac{l}{n^2}\sum_{b\neq q}(U_{ab})^2. \tag{51}$$

It follows from the fact that the rows of U are normalized that

$$\sum_{b\neq q}(U_{ab})^2 \le \sum_{b=1}^{n}(U_{ab})^2 = 1. \tag{52}$$

Combining (51) and (52) yields that

$$\sum_{b\neq q}(U_{ab})^2\,\mathbf{E}\,(F_{qb})^2 \leq \frac{l}{n^2}. \tag{53}$$

It follows from the independence of the random variables involved that

$$\sum_{b,c\neq q \text{ and } b\neq c} U_{ab}\,U_{ac}\,\mathbf{E}\,d_b\,d_c\,F_{qb}\,F_{qc} = \sum_{b,c\neq q \text{ and } b\neq c} U_{ab}\,U_{ac}\,(\mathbf{E}\,d_b)\,(\mathbf{E}\,d_c)\,\mathbf{E}\,(F_{qb}\,F_{qc}). \tag{54}$$

Combining (54) and the fact that $\mathbf{E}\,d_b = 0$ (or $\mathbf{E}\,d_c = 0$) yields that

$$\sum_{b,c\neq q \text{ and } b\neq c} U_{ab}\,U_{ac}\,\mathbf{E}\,d_b\,d_c\,F_{qb}\,F_{qc} = 0. \tag{55}$$

Combining (47), (48), (49), (50), (53), and (55) yields that

$$\mathbf{E}\sum_{q=1}^{n}(U_{pq})^2\left(\sum_{b\neq q}U_{ab}\,d_b\,F_{qb}\right)^2 \leq \frac{l}{n^2}\sum_{q=1}^{n}(U_{pq})^2 \tag{56}$$

Combining (56) and the fact that the rows of U are normalized yields that

$$\mathbf{E}\sum_{q=1}^{n}(U_{pq})^2\left(\sum_{b\neq q}U_{ab}\,d_b\,F_{qb}\right)^2 \leq \frac{l}{n^2}. \tag{57}$$

To bound the second term in the right-hand side of (46), we observe that

$$\begin{aligned}
&\mathbf{E}\sum_{q\neq r}U_{pq}\,U_{pr}\,d_q\,d_r\sum_{b\neq q}U_{ab}\,d_b\,F_{qb}\sum_{c\neq r}U_{ac}\,d_c\,F_{rc}\\
&\quad=\mathbf{E}\sum_{q\neq r}U_{pq}\,U_{pr}\,d_q\,d_r\left(U_{ar}\,d_r\,F_{qr}+\sum_{b\neq q,r}U_{ab}\,d_b\,F_{qb}\right)\left(U_{aq}\,d_q\,F_{rq}+\sum_{c\neq q,r}U_{ac}\,d_c\,F_{rc}\right).
\end{aligned} \tag{58}$$

Expanding the product, we obtain

$$\begin{aligned}
&\mathbf{E}\sum_{q\neq r}U_{pq}\,U_{pr}\,d_q\,d_r\left(U_{ar}\,d_r\,F_{qr}+\sum_{b\neq q,r}U_{ab}\,d_b\,F_{qb}\right)\left(U_{aq}\,d_q\,F_{rq}+\sum_{c\neq q,r}U_{ac}\,d_c\,F_{rc}\right)\\
&\qquad=\mathbf{E}\sum_{q\neq r}U_{pq}\,U_{pr}\,d_q\,d_r\,U_{ar}\,U_{aq}\,d_r\,d_q\,F_{qr}\,F_{rq}\\
&\qquad\quad+\mathbf{E}\sum_{q\neq r}U_{pq}\,U_{pr}\,d_q\,d_r\sum_{b\neq q,r}U_{ab}\,d_b\,F_{qb}\sum_{c\neq q,r}U_{ac}\,d_c\,F_{rc}\\
&\qquad\quad+\mathbf{E}\sum_{q\neq r}U_{pq}\,U_{pr}\,d_q\,d_r\,U_{ar}\,d_r\,F_{qr}\sum_{c\neq q,r}U_{ac}\,d_c\,F_{rc}\\
&\qquad\quad+\mathbf{E}\sum_{q\neq r}U_{pq}\,U_{pr}\,d_q\,d_r\,U_{aq}\,d_q\,F_{rq}\sum_{b\neq q,r}U_{ab}\,d_b\,F_{qb}.
\end{aligned} \tag{59}$$

Combining (32) and (31) yields that

$$F_{rq} = F_{qr}. \tag{60}$$

Combining (60) and the fact that $(d_q)^2 = 1$ and $(d_r)^2 = 1$ yields that

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r\, U_{ar}\, U_{aq}\, d_r\, d_q\, F_{qr}\, F_{rq} = \sum_{q \neq r} U_{pq}\, U_{pr}\, U_{ar}\, U_{aq}\, \mathbf{E}\, (F_{qr})^2. \tag{61}$$

It follows from (39) that

$$\sum_{q \neq r} U_{pq}\, U_{pr}\, U_{ar}\, U_{aq}\, \mathbf{E}\, (F_{qr})^2 = \frac{l}{n^2} \sum_{q \neq r} U_{pq}\, U_{pr}\, U_{ar}\, U_{aq}. \tag{62}$$

But,

$$\sum_{q \neq r} U_{pq}\, U_{pr}\, U_{ar}\, U_{aq} = \sum_{q,r=1}^{n} U_{pq}\, U_{pr}\, U_{ar}\, U_{aq} - \sum_{q=1}^{n} (U_{pq})^2\, (U_{aq})^2. \tag{63}$$

Furthermore,

$$\sum_{q,r=1}^{n} U_{pq}\, U_{pr}\, U_{ar}\, U_{aq} = \left(\sum_{q=1}^{n} U_{pq}\, U_{aq} \right)^2. \tag{64}$$

It follows from the definition of matrix multiplication that

$$\left(\sum_{q=1}^{n} U_{pq}\, U_{aq} \right)^2 = \left((U\, U^{\mathrm{T}})_{pa} \right)^2. \tag{65}$$

It follows from the fact that the rows of U are orthonormal that

$$\left((U\, U^{\mathrm{T}})_{pa} \right)^2 = \delta_{pa}, \tag{66}$$

where δ is the Kronecker symbol. Combining (61), (62), (63), (64), (65), and (66) yields that

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r\, U_{pr}\, U_{pq}\, d_r\, d_q\, F_{qr}\, F_{rq} \leq \frac{l}{n^2}\, \delta_{pa}. \tag{67}$$

It follows from the independence of the random variables involved that

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r \sum_{b \neq q,r} U_{ab}\, d_b\, F_{qb} \sum_{c \neq q,r} U_{ac}\, d_c\, F_{rc}$$
$$= \sum_{q \neq r} U_{pq}\, U_{pr}\, (\mathbf{E}\, d_q)\, (\mathbf{E}\, d_r) \left(\mathbf{E} \sum_{b \neq q,r} U_{ab}\, d_b\, F_{qb} \sum_{c \neq q,r} U_{ac}\, d_c\, F_{rc} \right). \tag{68}$$

Combining (68) and the fact that $\mathbf{E}\, d_q = 0$ (or $\mathbf{E}\, d_r = 0$) yields that

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r \sum_{b \neq q,r} U_{ab}\, d_b\, F_{qb} \sum_{c \neq q,r} U_{ac}\, d_c\, F_{rc} = 0. \tag{69}$$

It follows from the fact that $(d_r)^2 = 1$ that

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r\, U_{ar}\, d_r\, F_{qr} \sum_{c \neq q,r} U_{ac}\, d_c\, F_{rc} \;=\; \sum_{q \neq r} U_{pq}\, U_{pr}\, U_{ar}\, \mathbf{E}\, d_q\, F_{qr} \sum_{c \neq q,r} U_{ac}\, d_c\, F_{rc}. \tag{70}$$

It follows from the independence of the random variables involved that

$$\sum_{q \neq r} U_{pq}\, U_{pr}\, V_{1r}\, \mathbf{E}\, d_q\, F_{qr} \sum_{c \neq q,r} V_{1c}\, d_c\, F_{rc} = \sum_{q \neq r} U_{pq}\, U_{pr}\, V_{1r}\, (\mathbf{E}\, d_q) \left(\mathbf{E}\, F_{qr} \sum_{c \neq q,r} V_{1c}\, d_c\, F_{rc} \right). \tag{71}$$

It follows from the fact that $\mathbf{E}\, d_q = 0$ that

$$\sum_{q \neq r} U_{pq}\, U_{pr}\, U_{ar}\, (\mathbf{E}\, d_q) \left(\mathbf{E}\, F_{qr} \sum_{c \neq q,r} U_{ac}\, d_c\, F_{rc} \right) = 0. \tag{72}$$

Combining (70), (71), and (72) yields that

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r\, U_{ar}\, d_r\, F_{qr} \sum_{c \neq q,r} U_{ac}\, d_c\, F_{rc} = 0. \tag{73}$$

Similarly,

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r\, U_{aq}\, d_q\, F_{rq} \sum_{b \neq q,r} U_{ab}\, d_b\, F_{rb} = 0. \tag{74}$$

Combining (58), (59), (67), (69), (73), and (74) yields that

$$\mathbf{E} \sum_{q \neq r} U_{pq}\, U_{pr}\, d_q\, d_r \sum_{b \neq q} U_{ab}\, d_b\, F_{qb} \sum_{c \neq r} U_{ac}\, d_c\, F_{rc} \leq \frac{l}{n^2}\, \delta_{pa}. \tag{75}$$

Combining (45), (46), (57), and (75) yields that

$$\mathbf{E} \left(\sum_{q=1}^{n} U_{pq}\, d_q \sum_{b \neq q} U_{ab}\, d_b\, F_{qb} \right)^2 \leq \frac{l}{n^2} \left(1 + \delta_{pa}\right). \tag{76}$$

Combining (43) and (76) yields that

$$\mathbf{E} \sum_{p,a=1}^{k} \left(E_{pa}\right)^2 \leq \frac{l}{n^2} \sum_{p,a=1}^{k} \left(1 + \delta_{pa}\right). \tag{77}$$

It follows from (77) that

$$\mathbf{E} \sum_{p,a=1}^{k} \left(E_{pa}\right)^2 \leq \frac{l(k^2 + k)}{n^2}. \tag{78}$$

However,

$$\|E\|^2 \leq \sum_{p,a=1}^{k} \left(E_{pa}\right)^2. \tag{79}$$

Combining (79) and (78) yields that

$$\mathbf{E}\,\|E\|^2 \le \frac{l(k^2+k)}{n^2}. \tag{80}$$

Combining (80) and the Markov inequality yields that

$$\|E\| \le \frac{\sqrt{\beta\, l\,(k^2+k)}}{n} \tag{81}$$

with probability at least $1 - \frac{1}{\beta}$. It follows from (42) that

$$\|\Delta^{-1}\| = \frac{n}{l}. \tag{82}$$

Combining (81), (82), and (26) yields that

$$\|E\|\,\|\Delta^{-1}\| \le 1 - \frac{1}{\alpha} \tag{83}$$

with probability at least $1 - \frac{1}{\beta}$.

It follows from (44) that

$$\|C^{-1}\| \le \|\Delta^{-1}\|\,\|(\mathbf{1} + E\,\Delta^{-1})^{-1}\|. \tag{84}$$

However,

$$\|(\mathbf{1} + E\,\Delta^{-1})^{-1}\| \le \sum_{j=0}^{\infty} \|-E\,\Delta^{-1}\|^j. \tag{85}$$

Combining (85) and (83) yields that

$$\|(\mathbf{1} + E\,\Delta^{-1})^{-1}\| \le \alpha \tag{86}$$

with probability at least $1 - \frac{1}{\beta}$.

Finally, combining (84), (82), and (86) yields (28).

Combining (44), (42), (81), and (26) yields (29). □

5 Description of the algorithm

In this section, we describe the algorithm of the present paper. In Section 5.1, we discuss approximations to interpolative decompositions. In Section 5.2, we discuss approximations to SVDs. In Section 5.3, we discuss an alternative method for constructing approximations to SVDs. In Section 5.4, we tabulate the computational costs of various parts of the algorithm.

5.1 Interpolative decomposition

Suppose that k, m, and n are positive integers with $k < m$ and $k < n$, and A is a real $m \times n$ matrix. In this subsection, we will collect together k appropriately chosen columns of A into a real $m \times k$ matrix B, and construct a real $k \times n$ matrix P, such that

$$\|P\| \leq \sqrt{4k\,(n-k)+1} \tag{87}$$

and

$$\|B\,P - A\| \lesssim \sqrt{klmn}\;\sigma_{k+1}, \tag{88}$$

where σ_{k+1} is the $(k+1)^{\text{st}}$ greatest singular value of A, and l is a user-specified integer near to, but greater than k, such that $l < m$ and $l < n$ (for example, $l = 4k$). To do so, we make the following three steps:

1. Using the algorithm of Subsection 3.2, compute the $l \times n$ product matrix

   $$T = R\,A, \tag{89}$$

   where R is the real $l \times m$ matrix defined via the formula

   $$R = H\,D, \tag{90}$$

   with the rows of H consisting of l rows of the Walsh-Hadamard transform $H^{(m)}$ defined in Section 3 chosen uniformly at random, and D being a diagonal real $m \times m$ matrix whose diagonal entries are i.i.d. random variables distributed uniformly on $\{-1, 1\}$.

2. Using Observation 2.3, form a real $l \times k$ matrix S whose columns constitute a subset of the columns of T, and a real $k \times n$ matrix P satisfying (87), such that

   $$\|S\,P - T\| \leq \sqrt{4k\,(n-k)+1}\;\tau_{k+1}, \tag{91}$$

   where τ_{k+1} is the $(k+1)^{\text{st}}$ greatest singular value of T.

3. Using the fact that the columns of S constitute a subset of the columns of T, for any $j = 1, 2, \ldots, k-1, k$, let i_j denote an integer such that the j^{th} column of S is the ${i_j}^{\text{th}}$ column of T. Form the real $m \times k$ matrix B whose j^{th} column is the ${i_j}^{\text{th}}$ column of A for all $j = 1, 2, \ldots, k-1, k$.

An analysis similar to that in Section 4.1 of [10] (using Lemma 4.1 of the present paper in place of Lemma 14 of [10]) yields that the matrices B and P obtained via the preceding three steps satisfy (87) and (88) with high probability. Strictly speaking, our present analysis requires that l be proportional to k^2; however, our numerical experiments (described in Section 6) indicate that in fact l need be only proportional to k.

5.2 Singular value decomposition

Suppose that k, m, and n are positive integers with $k < m$ and $k < n$, and A is a real $m \times n$ matrix. In this subsection, we will construct an approximation to an SVD of A such that

$$\|U\,\Sigma\,V^{\mathrm{T}} - A\| \lesssim \sqrt{lm}\,\sigma_{k+1}, \tag{92}$$

where U is a real $m \times k$ matrix whose columns are orthonormal, V is a real $n \times k$ matrix whose columns are orthonormal, Σ is a diagonal real $k \times k$ matrix whose entries are all nonnegative, σ_{k+1} is the $(k+1)^{\mathrm{st}}$ greatest singular value of A, and l is a user-specified integer near to, but greater than k, such that $l < m$ and $l < n$ (for example, $l = 4k$). To do so, we make the following ten steps:

1. Using the algorithm of Subsection 3.2, compute the $l \times n$ product matrix

$$T = R\,A, \tag{93}$$

   where R is the real $l \times m$ matrix defined via the formula

$$R = H\,D, \tag{94}$$

   with the rows of H consisting of l rows of the Walsh-Hadamard transform $H^{(m)}$ defined in Section 3 chosen uniformly at random, and D being a diagonal real $m \times m$ matrix whose diagonal entries are i.i.d. random variables distributed uniformly on $\{-1, 1\}$.

2. Using the algorithm of Subsection 3.2, compute the $l \times m$ product matrix

$$Z = F\,A^{\mathrm{T}}, \tag{95}$$

   where F is the real $l \times n$ matrix defined via the formula

$$F = G\,C, \tag{96}$$

   with the rows of G consisting of l rows of the Walsh-Hadamard transform $H^{(n)}$ defined in Section 3 chosen uniformly at random, and C being a diagonal real $n \times n$ matrix whose diagonal entries are i.i.d. random variables distributed uniformly on $\{-1, 1\}$.

3. Using an SVD, form a real $n \times k$ matrix Q whose columns are orthonormal, such that there exists a real $k \times l$ matrix S for which

$$\|Q\,S - T^{\mathrm{T}}\| \leq \tau_{k+1}, \tag{97}$$

   where τ_{k+1} is the $(k+1)^{\mathrm{st}}$ greatest singular value of T. (See Observation 2.5 for details concerning the construction of such a matrix Q.)

4. Using an SVD, form a real $m \times k$ matrix P whose columns are orthonormal, such that there exists a real $k \times l$ matrix Y for which

$$\|P\,Y - Z^{\mathrm{T}}\| \leq \zeta_{k+1}, \tag{98}$$

   where ζ_{k+1} is the $(k+1)^{\mathrm{st}}$ greatest singular value of Z. (See Observation 2.5 for details concerning the construction of such a matrix P.)

5. Using the algorithm of Subsection 3.2, compute the $l \times k$ product matrix

$$W = F\,Q \tag{99}$$

where F is defined in (96), and Q is from (97).

6. Compute the $l \times k$ product matrix

$$B = Z\,P \tag{100}$$

where Z is defined in (95), and P is from (98).

7. Compute the real $k \times k$ matrix X which minimizes the quantity

$$\|W\,X - B\|, \tag{101}$$

where W is defined in (99), and B is defined in (100). (See, for example, Section 5.3 in [5] for details concerning the construction of such a minimizing X.)

8. Construct an SVD of X, that is,

$$X = U^X\,\Sigma\,(V^X)^{\mathrm{T}}, \tag{102}$$

where U^X is a real $k \times k$ matrix whose columns are orthonormal, V^X is a real $k \times k$ matrix whose columns are orthonormal, and Σ is a real $k \times k$ matrix whose entries are all nonnegative and zero off of the main diagonal. (See, for example, Chapter 8 in [5] for details concerning the construction of such an SVD.)

9. Compute the $m \times k$ product matrix

$$U = P\,V^X, \tag{103}$$

where P is from (98), and V^X is from (102).

10. Compute the $n \times k$ product matrix

$$V = Q\,U^X, \tag{104}$$

where Q is from (97), and U^X is from (102).

An analysis similar to that in Section 4.2 of [10] (using Lemma 4.1 of the present paper in place of Lemma 14 of [10]) yields that the matrices U, Σ, and V obtained via the preceding ten steps satisfy (92) with high probability. Strictly speaking, our present analysis requires that l be proportional to k^2; however, our numerical experiments (described in Section 6) indicate that in fact l need be only proportional to k.

Remark 5.1 Step 7 is motivated by an idea from [12], [1], [4] of using the random matrix defined in (96) for the purpose of computing a solution in the least squares sense to an overdetermined system of linear algebraic equations.

5.3 Singular value decomposition by means of the interpolative decomposition

In this subsection, we provide an alternative to the algorithm of Subsection 5.2 for computing an approximation to a singular value decomposition. This alternative is usually somewhat more efficient.

Suppose that k, m, and n are positive integers with $k < m$ and $k < n$, and A is a real $m \times n$ matrix. We will compute an approximation to an SVD of A such that

$$\|U\,\Sigma\,V^{\mathrm{T}} - A\| \lesssim \sqrt{klmn}\;\sigma_{k+1}, \tag{105}$$

where U is a real $m \times k$ matrix whose columns are orthonormal, V is a real $n \times k$ matrix whose columns are orthonormal, Σ is a diagonal real $k \times k$ matrix whose entries are all nonnegative, σ_{k+1} is the $(k+1)^{\mathrm{st}}$ greatest singular value of A, and l is a user-specified integer near to, but greater than k, such that $l < m$ and $l < n$ (for example, $l = 4k$). To do so, we first use the algorithm of Subsection 5.1 to construct the matrices B and P in (87) and (88). Then, we make the following four steps:

1. Construct a lower triangular real $k \times k$ matrix L, and a real $n \times k$ matrix Q whose columns are orthonormal, such that

   $$P = L\,Q^{\mathrm{T}}. \tag{106}$$

   (See Remark 2.7 for details concerning the construction of such matrices L and Q.)

2. Compute the $m \times k$ product matrix

   $$C = B\,L. \tag{107}$$

3. Construct an SVD of C, that is,

   $$C = U\,\Sigma\,W^{\mathrm{T}}, \tag{108}$$

   where U is a real $m \times k$ matrix whose columns are orthonormal, Σ is a diagonal $k \times k$ matrix whose entries are all nonnegative, and W is a real $k \times k$ matrix whose columns are orthonormal. (See, for example, Chapter 8 in [5] for details concerning the construction of such an SVD.)

4. Compute the $n \times k$ product matrix

   $$V = Q\,W. \tag{109}$$

An analysis similar to that in Section 4.1 of [10] (using Lemma 4.1 of the present paper in place of Lemma 14 of [10]) yields that the matrices U, Σ, and V obtained via the preceding four steps (after first using the algorithm of Subsection 5.1 to construct the matrices B and P) satisfy (105) with high probability. Strictly speaking, our present analysis requires that l be proportional to k^2; however, our numerical experiments (described in Section 6) indicate that in fact l need be only proportional to k.

Remark 5.2 Steps 2 and 4 in the procedure of the present subsection are somewhat subtle numerically. Both Steps 2 and 4 involve constructing products of matrices, and in general constructing the product $S\,T$ of matrices S and T can be numerically unstable. Indeed, in general some entries of S or T can have unmanageably large absolute values, while in exact arithmetic no entry of the product $S\,T$ has an unmanageably large absolute value; in such circumstances, constructing the product $S\,T$ can be unstable in finite-precision arithmetic. However, this problem does not arise in Steps 2 and 4 above, due to (87), (18), the fact that the columns of B constitute a subset of the columns of A (so that $\|B\| \le \|A\|$), and the fact that the columns of Q are orthonormal, as are the columns of W.

5.4 Costs

In this subsection, we tabulate the numbers of floating-point operations and words of memory required by the algorithms described in Subsections 5.1, 5.2, and 5.3, as applied once to a real $m \times n$ matrix A.

5.4.1 Interpolative decomposition

The algorithm of Subsection 5.1 incurs the following costs in order to compute an approximation to an interpolative decomposition of A:

1. Computing T in (89) costs $\mathcal{O}(mn\,\log(l))$.
2. Computing S and P in (91) costs $\mathcal{O}(lkn\,\log(n))$.
3. Forming B in Step 3 requires retrieving k columns of the $m \times n$ matrix A, which costs $\mathcal{O}(km)$.

Summing up the costs in Steps 1–3 above, we conclude that the algorithm of Subsection 5.1 costs

$$C_{\mathrm{ID}} = \mathcal{O}(mn\,\log(l) + lkn\,\log(n)). \tag{110}$$

Remark 5.3 When "$Q\,R$" decompositions are used as in [3] to compute the matrices S and P in (91), the cost of the algorithm of Subsection 5.1 is usually less than the cost of the algorithm of Subsection 5.2, typically

$$C'_{\mathrm{ID}} = \mathcal{O}(mn\,\log(l) + lkn + km). \tag{111}$$

5.4.2 Singular value decomposition

The algorithm of Subsection 5.2 incurs the following costs in order to compute an approximation to a singular value decomposition of A:

1. Computing T in (93) costs $\mathcal{O}(mn\,\log(l))$.
2. Computing Z in (95) costs $\mathcal{O}(mn\,\log(l))$.
3. Computing Q in (97) costs $\mathcal{O}(l^2\,n)$.

4. Computing P in (98) costs $\mathcal{O}(l^2\,m)$.

5. Computing W in (99) costs $\mathcal{O}(kn\,\log(l))$.

6. Computing B in (100) costs $\mathcal{O}(klm)$.

7. Computing X minimizing (101) costs $\mathcal{O}(k^2\,l)$.

8. Computing the SVD (102) of X costs $\mathcal{O}(k^3)$.

9. Computing U in (103) costs $\mathcal{O}(k^2\,m)$.

10. Computing V in (104) costs $\mathcal{O}(k^2\,n)$.

Summing up the costs in Steps 1–10 above, we conclude that the algorithm of Subsection 5.2 costs

$$C_{\text{SVD}} = \mathcal{O}(mn\,\log(l) + l^2(m+n)). \tag{112}$$

5.4.3 Singular value decomposition by means of the interpolative decomposition

The algorithm of Subsection 5.2 incurs the following costs in order to compute an approximation to a singular value decomposition of A, in addition to (110):

1. Constructing L and Q in (106) costs $\mathcal{O}(k^2\,n)$.

2. Computing C in (107) costs $\mathcal{O}(k^2\,m)$.

3. Computing the SVD of C in (108) costs $\mathcal{O}(k^2\,m)$.

4. Computing V in (109) costs $\mathcal{O}(k^2\,n)$.

Summing up the costs in Steps 1–4 above, plus (110), we conclude that the algorithm of Subsection 5.3 costs

$$C_{\text{SVD(ID)}} = \mathcal{O}(mn\,\log(l) + lkn\,\log(n) + k^2\,m). \tag{113}$$

Remark 5.4 As in Remark 5.3, when "$Q\,R$" decompositions are used as in [3] to compute the matrices S and P in (91), the cost of the algorithm of Subsection 5.3 is usually less than the cost of the algorithm of Subsection 5.2, typically

$$C'_{\text{SVD(ID)}} = \mathcal{O}(mn\,\log(l) + lkn + k^2\,m). \tag{114}$$

6 Numerical examples

In this section, we describe the results of several numerical tests of the algorithm of the present paper. Tables 1–7 summarize the numerical output of applying the algorithm to the matrix A defined below for each of the examples. Please note that we did not seriously optimize our implementation of the randomized algorithm.

In the first and second subtables of each of Tables 1–5, we set $l = 4k$ for the user-specified parameter l. In the third and fourth subtables of each of Tables 1–5, we set $l = 10k$. In both subtables of Table 6, we set $l = k + 10$. In Table 7, we set $l = 4k$ in the algorithm of the present paper.

The first and third subtables of each of Tables 1–4 display the results of applying the interpolative decomposition algorithm of Subsection 5.1 once to the matrix A defined below for each example. The second and fourth subtables of each of Tables 1–4 display the results of applying the singular value decomposition algorithm of Subsection 5.3. Both subtables of Table 5 display the results of applying the singular value decomposition algorithm of Subsection 5.2. The first subtable of Table 6 displays the results of applying the interpolative decomposition algorithm of Subsection 5.1, using a fast discrete cosine transform in place of the trimmed fast Walsh-Hadamard transform. The second subtable of Table 6 displays the results of applying the singular value decomposition algorithm of Subsection 5.3, using a fast discrete cosine transform in place of the trimmed fast Walsh-Hadamard transform. Table 7 displays the results of applying the interpolative decomposition algorithm of Subsection 5.1.

In all of the tables, σ_{k+1} is the $(k+1)^{\text{st}}$ greatest singular value of A; σ_{k+1} is also the spectral norm of the difference between the original matrix A and its best rank-k approximation.

In the first and third subtables of each of Tables 1–4, as well as in the first subtable of Table 6, δ_{direct} is the spectral norm of the difference between the original matrix A and the approximation $B\,P$ to an interpolative decomposition obtained via the pivoted "$Q\,R$" decomposition algorithm of [3] that is based upon plane (Householder) reflections. In the second and fourth subtables of each of Tables 1–4, as well as in both subtables of Table 5 and the second subtable of Table 6, δ_{direct} is the spectral norm of the difference between the original matrix A and the approximation $U\,\Sigma\,V^{\mathrm{T}}$ to an SVD obtained via following up a pivoted "$Q\,R$" decomposition algorithm based upon plane (Householder) reflections with a call to the LAPACK 3.1.1 divide-and-conquer SVD routine `dgesdd`.

In Table 7, δ_{prev} is the spectral norm of the difference between the original matrix A and the approximation $B\,P$ to an interpolative decomposition obtained via the algorithm of [10], using $k + 20$ random test vectors.

In the first and third subtables of each of Tables 1–4, as well as in the first subtable of Table 6 and in Table 7, δ_{fast} is the spectral norm of the difference between the original matrix A and the approximation $B\,P$ to an interpolative decomposition obtained via the randomized algorithm. In the second and fourth subtables of each of Tables 1–4, as well as in both subtables of Table 5 and the second subtable of Table 6, δ_{fast} is the spectral norm of the difference between the original matrix A and the approximation $U\,\Sigma\,V^{\mathrm{T}}$ to an SVD obtained via the randomized algorithm.

In the first and third subtables of each of Tables 1–4, as well as in the first subtable of Table 6, t_{direct} is the number of seconds of CPU time taken by the pivoted "$Q\,R$" decomposition algorithm of [3] that is based upon plane (Householder) reflections. In the second and fourth subtables of each of Tables 1–4, as well as in both subtables of Table 5 and the second subtable of Table 6, t_{direct} is the number of seconds of CPU time taken by the combination of a pivoted "$Q\,R$" decomposition algorithm based upon plane (Householder) reflections and the LAPACK 3.1.1 divide-and-conquer SVD routine `dgesdd`.

In Table 7, t_{prev} is the number of seconds of CPU time taken by the interpolative decomposition algorithm of [10], using $k+20$ random test vectors.

t_{fast} is the number of seconds of CPU time taken by the randomized algorithm.

The values of δ_{direct}, δ_{prev}, and δ_{fast} are those obtained via the power method for estimating the norm of a matrix. We terminated the power method after its estimates stabilized to four significant figures when the estimates stabilized in less than 100 iterations, and after 100 iterations when the estimates were near the machine precision and did not stabilize due to roundoff.

Table 1 reports the results of applying the algorithms of Subsections 5.1 and 5.3 to the 512×512 matrix A defined via the formula

$$A = \frac{T}{\|T\|}, \tag{115}$$

where T is the 512×512 matrix with the entry

$$T_{j,k} = \frac{1}{j^2 + k^2 + \frac{k^3}{1000}} \tag{116}$$

for $j, k = 1, 2, \ldots, 511, 512$. In Table 1, δ_{fast} is the maximum error encountered in 100 randomized trials, while t_{fast} is the average time over 50 randomized trials.

Table 2 reports the results of applying the algorithms of Subsections 5.1 and 5.3 to the $2{,}048 \times 2{,}048$ matrix defined via the formula

$$A = \sum_{k=1}^{65} u_k\, \sigma_k\, (v_k)^{\mathrm{T}}, \tag{117}$$

where

$$\sigma_1 = \sigma_2 = \cdots = \sigma_9 = \sigma_{10} = 1, \tag{118}$$

$$\sigma_{11} = \sigma_{12} = \cdots = \sigma_{19} = \sigma_{20} = 10^{-2}, \tag{119}$$

$$\sigma_{21} = \sigma_{22} = \cdots = \sigma_{29} = \sigma_{30} = 10^{-4}, \tag{120}$$

$$\sigma_{31} = \sigma_{32} = \cdots = \sigma_{39} = \sigma_{40} = 10^{-6}, \tag{121}$$

$$\sigma_{41} = \sigma_{42} = \cdots = \sigma_{49} = \sigma_{50} = 10^{-8}, \tag{122}$$

$$\sigma_{51} = \sigma_{52} = \cdots = \sigma_{59} = \sigma_{60} = 10^{-10}, \tag{123}$$

$$\sigma_{61} = \sigma_{62} = \sigma_{63} = \sigma_{64} = \sigma_{65} = 10^{-12}, \tag{124}$$

the v_k for $k = 1, 2, \ldots, 64, 65$ are the first 65 columns of the Walsh-Hadamard transform $H^{(2048)}$ defined in Section 3, and

$$\begin{aligned}
(u_1)^{\mathrm{T}} &= \frac{1}{\sqrt{n-1}} \begin{pmatrix} 1 & 1 & \ldots & 1 & 1 & 0 \end{pmatrix}, && (125)\\
(u_2)^{\mathrm{T}} &= \begin{pmatrix} 0 & 0 & \ldots & 0 & 0 & 1 \end{pmatrix}, && (126)\\
(u_3)^{\mathrm{T}} &= \frac{1}{\sqrt{n-2}} \begin{pmatrix} 1 & -1 & 1 & -1 & \ldots & 1 & -1 & 1 & -1 & 0 & 0 \end{pmatrix}, && (127)\\
(u_4)^{\mathrm{T}} &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 0 & -1 & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}, && (128)\\
(u_5)^{\mathrm{T}} &= \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}, && (129)\\
(u_6)^{\mathrm{T}} &= \frac{1}{\sqrt{2}} \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 & \ldots & 0 & 0 \end{pmatrix}, && (130)\\
& && (131)
\end{aligned}$$

and so on. More precisely, for $k = 4, 5, \ldots, 64, 65$, the $(4k-15)^{\text{th}}$ and $(4k-13)^{\text{th}}$ entries of u_k are $\frac{1}{\sqrt{2}}$ and $-\frac{1}{\sqrt{2}}$, and all other entries of u_k are zero. In Table 2, δ_{fast} is the maximum error encountered in 10 randomized trials, while t_{fast} is the average time over 2 randomized trials.

Table 3 reports the results of applying the algorithms of Subsections 5.1 and 5.3 to the 512×512 matrix A defined via the formula

$$A = \sum_{k=1}^{5} u_k \, \sigma_k \, (v_k)^{\mathrm{T}}, \tag{132}$$

where $\sigma_1 = 1$, $\sigma_2 = 10^{-9}$, $\sigma_3 = 10^{-10}$, $\sigma_4 = 10^{-11}$, $\sigma_5 = 10^{-12}$, and u_1, u_2, u_3, u_4, u_5 and v_1, v_2, v_3, v_4, v_5 are the left and right singular vectors corresponding to the 5 greatest singular values of a 512×512 matrix whose entries are drawn i.i.d. with a Gaussian distribution of zero mean and unit variance. In Table 3, δ_{fast} is the maximum error encountered in 100 randomized trials, while t_{fast} is the average time over 1,000 randomized trials. To be consistent, we used the same pivoted "$Q\,R$" decomposition algorithm based on plane (Householder) reflections as the direct algorithm for $k = 1$ as for $k > 1$, even though it is wasteful to maintain the data necessary for pivoting when $k = 1$.

Table 4 reports the results of applying the algorithms of Subsections 5.1 and 5.3 to the $1{,}024 \times 1{,}024$ matrix defined via the formula

$$A = \frac{S}{\|S\|}, \tag{133}$$

where S is a sparse $1{,}024 \times 1{,}024$ matrix having $k+1$ non-zero entries. The positions of the nonzero entries were chosen at random with a uniform distribution over all $1{,}024 \times 1{,}024$ possible locations. The values of k of the nonzero entries of S were chosen at random i.i.d. with a uniform distribution on $[0, 1]$, and the value of the remaining nonzero entry of S was chosen at random independently from the others with a uniform distribution on

$[0, 10^{-10}]$. In Table 4, δ_{fast} is the maximum error encountered in 10 randomized trials, while t_{fast} is the average time over 10 randomized trials. Please note that our implementations for sparse matrices were the same as for dense matrices. Clearly, neither the direct nor the fast algorithm takes full advantage of the sparsity of these matrices; the algorithm of [10] would seem to be more appropriate for such sparse matrices.

Table 5 reports the results of applying the algorithm of Subsection 5.2 to the 2,048 × 2,048 matrix defined in (117). In Table 5, δ_{fast} is the maximum error encountered in 10 randomized trials, while t_{fast} is the average time over 10 randomized trials.

Table 6 reports the results of applying the algorithms of Subsections 5.1 and 5.3 to the 2,048 × 2,048 matrix defined in (117), using the full fast discrete cosine transform as implemented in a double-precision version of P. N. Swarztrauber's FFTPACK routine `cost` (available at `http://www.netlib.org/fftpack/`) in place of the trimmed fast Walsh-Hadamard transform. In Table 6, δ_{fast} is the maximum error encountered in 10 randomized trials, while t_{fast} is the average time over 80 randomized trials.

Table 7 reports the results of applying the algorithm of Subsection 5.1 to the 32,768 × 32,768 matrix A defined via the formula

$$A = \frac{R}{\|R\|}, \tag{134}$$

where R is the 32,768 × 32,768 matrix with the entry

$$R_{j,k} = \frac{1}{\frac{(j-k)^2}{32768} + k + \frac{j-1}{2} - \frac{1700}{3}} \tag{135}$$

for $j = 1, 3, \ldots, 32{,}765, 32{,}767$ and $k = 1, 2, \ldots, 32{,}767, 32{,}768$, and

$$R_{j,k} = \frac{1}{\frac{(j-k)^2}{32768} + k + \frac{j}{2} - \frac{1700}{3}} \tag{136}$$

for $j = 2, 4, \ldots, 32{,}766, 32{,}768$ and $k = 1, 2, \ldots, 32{,}767, 32{,}768$. Since storing all of the individual entries of A would have required far more than our 2 GB of RAM, we computed the entries of A on the fly, as needed, via (134), (135), and (136) (after first computing that $\|R\| \approx .903\text{E+}03$).

We performed all computations using IEEE standard double-precision variables, whose mantissas have approximately one bit of precision less than 16 digits (so that the relative precision of the variables is approximately .2E-15). We ran all computations on a 2.8 GHz Pentium Xeon microprocessor with 1 MB of L2 cache and 2 GB of RAM. We compiled the Fortran 77 code using the fort77 script (which couples f2c to gcc) with the optimization flag `-O3` enabled.

Remark 6.1 Tables 1–4 indicate that the algorithms of Subsections 5.1 and 5.3 are generally more efficient than the classical pivoted "$Q\,R$" decomposition algorithm based on plane (Householder) reflections, followed by either the algorithm of [3] or the LAPACK 3.1.1 divide-and-conquer SVD routine. A comparison of Tables 1–4 and Table 5 indicates that the algorithm of Subsection 5.2 is not competitive with the algorithm of Subsection 5.3 in terms of either accuracy or efficiency, at least on the examples considered in the present paper.

Remark 6.2 Numerical experiments (including those reported in Table 6) indicate that the user-specified parameter l may be chosen to be smaller when a discrete cosine transform is used in place of the Walsh-Hadamard transform in the algorithm of the present paper. It may prove advantageous to replace the trimmed Walsh-Hadamard transform with the composition of a trimmed Walsh-Hadamard transform, followed by multiplication with a random diagonal matrix, followed by a trimmed discrete cosine or Fourier transform.

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
31	.406E-12	.143E-11	.365E-11	.828E-01	.220E-01	3.76
33	.557E-13	.342E-12	.427E-12	.876E-01	.230E-01	3.81
35	.745E-14	.191E-13	.588E-13	.930E-01	.247E-01	3.77
37	.969E-15	.344E-14	.797E-14	.977E-01	.263E-01	3.71
39	$\leq$ .2E-15	.356E-15	.118E-14	.103E-00	.279E-01	3.70

(1.1) ID of Subsection 5.1, $l = 4k$, 512×512 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
31	.406E-12	.143E-11	.365E-11	.778E-01	.289E-01	2.69
33	.557E-13	.342E-12	.427E-12	.828E-01	.308E-01	2.69
35	.745E-14	.191E-13	.588E-13	.889E-01	.336E-01	2.65
37	.969E-15	.344E-14	.974E-14	.951E-01	.363E-01	2.62
39	$\leq$ .2E-15	.181E-14	.108E-13	.993E-01	.393E-01	2.53

(1.2) SVD of Subsection 5.3, $l = 4k$, 512×512 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
31	.406E-12	.143E-11	.184E-11	.658E-01	.524E-01	1.26
33	.557E-13	.342E-12	.350E-12	.701E-01	.578E-01	1.21
35	.745E-14	.191E-13	.273E-13	.744E-01	.637E-01	1.17
37	.969E-15	.344E-14	.582E-14	.779E-01	.703E-01	1.11
39	$\leq$ .2E-15	.356E-15	.115E-14	.817E-01	.760E-01	1.08

(1.3) ID of Subsection 5.1, $l = 10k$, 512×512 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
31	.406E-12	.143E-11	.184E-11	.759E-01	.572E-01	1.33
33	.557E-13	.342E-12	.350E-12	.810E-01	.648E-01	1.25
35	.745E-14	.191E-13	.273E-13	.871E-01	.721E-01	1.21
37	.969E-15	.344E-14	.582E-14	.915E-01	.789E-01	1.16
39	$\leq$ .2E-15	.181E-14	.108E-13	.963E-01	.872E-01	1.10

(1.4) SVD of Subsection 5.3, $l = 10k$, 512×512 matrix

Table 1 (See Section 6.)

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
10	.100E-01	.358E-01	.788E-01	.400E+00	.160E+00	2.50
20	.100E-03	.283E-03	.283E-01	.750E+00	.185E+00	4.05
30	.100E-05	.403E-05	.622E-05	.110E+01	.225E+00	4.89
40	.100E-07	.245E-07	.348E-07	.145E+01	.285E+00	5.11
50	.100E-09	.490E-09	.618E-09	.180E+01	.345E+00	5.22
60	.100E-11	.416E-11	.582E-11	.213E+01	.425E+00	5.02

(2.1) ID of Subsection 5.1, $l = 4k$, 2,048×2,048 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
10	.100E-01	.358E-01	.788E-01	.420E+00	.165E+00	2.55
20	.100E-03	.283E-03	.283E-01	.780E+00	.200E+00	3.90
30	.100E-05	.403E-05	.622E-05	.116E+01	.260E+00	4.46
40	.100E-07	.245E-07	.348E-07	.155E+01	.345E+00	4.49
50	.100E-09	.490E-09	.618E-09	.193E+01	.445E+00	4.34
60	.100E-11	.416E-11	.582E-11	.233E+01	.560E+00	4.16

(2.2) SVD of Subsection 5.3, $l = 4k$, 2,048×2,048 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
10	.100E-01	.358E-01	.412E-01	.485E+00	.195E+00	2.49
20	.100E-03	.283E-03	.327E-03	.905E+00	.275E+00	3.29
30	.100E-05	.403E-05	.532E-05	.132E+01	.385E+00	3.43
40	.100E-07	.245E-07	.269E-07	.175E+01	.525E+00	3.33
50	.100E-09	.490E-09	.517E-09	.217E+01	.740E+00	2.93
60	.100E-11	.416E-11	.445E-11	.255E+01	.950E+00	2.68

(2.3) ID of Subsection 5.1, $l = 10k$, 2,048×2,048 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
10	.100E-01	.358E-01	.412E-01	.460E+00	.190E+00	2.42
20	.100E-03	.283E-03	.327E-03	.870E+00	.285E+00	3.05
30	.100E-05	.403E-05	.532E-05	.128E+01	.405E+00	3.17
40	.100E-07	.245E-07	.269E-07	.171E+01	.570E+00	2.99
50	.100E-09	.490E-09	.517E-09	.213E+01	.790E+00	2.70
60	.100E-11	.416E-11	.445E-11	.255E+01	.106E+01	2.41

(2.4) SVD of Subsection 5.3, $l = 10k$, 2,048×2,048 matrix

Table 2 (See Section 6.)

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
1	.100E-09	.105E-09	.283E-09	.618E-02	.597E-02	1.04
2	.100E-10	.104E-10	.416E-10	.835E-02	.748E-02	1.12
3	.100E-11	.103E-11	.223E-11	.105E-01	.844E-02	1.24
4	.100E-12	.113E-12	.180E-12	.127E-01	.864E-02	1.47
5	$\leq$ .2E-15	$\leq$ .2E-15	.344E-15	.156E-01	.930E-02	1.67

(3.1) ID of Subsection 5.1, $l = 4k$, 512×512 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
1	.100E-09	.105E-09	.283E-09	.684E-02	.605E-02	1.13
2	.100E-10	.104E-10	.416E-10	.944E-02	.746E-02	1.27
3	.100E-11	.103E-11	.223E-11	.119E-01	.849E-02	1.41
4	.100E-12	.113E-12	.180E-12	.145E-01	.903E-02	1.61
5	$\leq$ .2E-15	$\leq$ .2E-15	.452E-15	.178E-01	.955E-02	1.86

(3.2) SVD of Subsection 5.3, $l = 4k$, 512×512 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
1	.100E-09	.105E-09	.152E-09	.608E-02	.713E-02	.852
2	.100E-10	.104E-10	.125E-10	.832E-02	.880E-02	.945
3	.100E-11	.103E-11	.123E-11	.104E-01	.936E-02	1.11
4	.100E-12	.113E-12	.130E-12	.125E-01	.103E-01	1.21
5	$\leq$ .2E-15	$\leq$ .2E-15	.280E-15	.153E-01	.109E-01	1.40

(3.3) ID of Subsection 5.1, $l = 10k$, 512×512 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\text{direct}}/t_{\text{fast}}$
1	.100E-09	.105E-09	.152E-09	.742E-02	.796E-02	.932
2	.100E-10	.104E-10	.125E-10	.996E-02	.971E-02	1.03
3	.100E-11	.103E-11	.123E-11	.125E-01	.104E-01	1.21
4	.100E-12	.113E-12	.130E-12	.151E-01	.116E-01	1.30
5	$\leq$ .2E-15	$\leq$ .2E-15	.467E-15	.186E-01	.122E-01	1.52

(3.4) SVD of Subsection 5.3, $l = 10k$, 512×512 matrix

Table 3 (See Section 6.)

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
10	.339E-10	.339E-10	.396E-10	.103E+00	.400E-01	2.57
11	.350E-10	.350E-10	.449E-10	.113E+00	.400E-01	2.83
12	.121E-10	.121E-10	.152E-10	.123E+00	.390E-01	3.15
13	.478E-10	.478E-10	.568E-10	.130E+00	.410E-01	3.17
14	.263E-10	.263E-10	.321E-10	.139E+00	.410E-01	3.39
15	.175E-10	.175E-10	.214E-10	.147E+00	.430E-01	3.42

(4.1) ID of Subsection 5.1, $l = 4k$, 1,024×1,024 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
10	.339E-10	.339E-10	.396E-10	.116E+00	.440E-01	2.64
11	.350E-10	.350E-10	.449E-10	.126E+00	.450E-01	2.80
12	.121E-10	.121E-10	.152E-10	.136E+00	.450E-01	3.02
13	.478E-10	.478E-10	.568E-10	.145E+00	.470E-01	3.09
14	.263E-10	.263E-10	.321E-10	.158E+00	.480E-01	3.29
15	.175E-10	.175E-10	.214E-10	.168E+00	.490E-01	3.43

(4.2) SVD of Subsection 5.3, $l = 4k$, 1,024×1,024 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
10	.339E-10	.339E-10	.382E-10	.103E+00	.470E-01	2.19
11	.188E-10	.188E-10	.200E-10	.112E+00	.490E-01	2.29
12	.715E-10	.715E-10	.797E-10	.123E+00	.530E-01	2.32
13	.323E-10	.323E-10	.356E-10	.130E+00	.540E-01	2.41
14	.295E-10	.295E-10	.314E-10	.139E+00	.590E-01	2.36
15	.932E-10	.932E-10	.995E-10	.148E+00	.610E-01	2.43

(4.3) ID of Subsection 5.1, $l = 10k$, 1,024×1,024 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
10	.339E-10	.339E-10	.382E-10	.105E+00	.500E-01	2.10
11	.188E-10	.188E-10	.200E-10	.115E+00	.510E-01	2.25
12	.715E-10	.715E-10	.797E-10	.123E+00	.550E-01	2.24
13	.323E-10	.323E-10	.356E-10	.132E+00	.590E-01	2.24
14	.295E-10	.295E-10	.314E-10	.142E+00	.620E-01	2.29
15	.932E-10	.932E-10	.995E-10	.149E+00	.640E-01	2.33

(4.4) SVD of Subsection 5.3, $l = 10k$, 1,024×1,024 matrix

Table 4 (See Section 6.)

k	σ_{k+1}	δ_{direct}	$\delta_{\text{“fast”}}$	t_{direct}	$t_{\text{“fast”}}$	$t_{\text{direct}}/t_{\text{“fast”}}$
10	.100E-01	.358E-01	.793E-01	.440E+00	.708E+00	.621
20	.100E-03	.283E-03	.546E-03	.765E+00	.775E+00	.987
30	.100E-05	.403E-05	.638E-05	.114E+01	.961E+00	1.19
40	.100E-07	.245E-07	.650E-07	.152E+01	.118E+01	1.29
50	.100E-09	.490E-09	.207E-07	.192E+01	.141E+01	1.36
60	.100E-11	.416E-11	.387E-09	.231E+01	.168E+01	1.38

(5.1) SVD of Subsection 5.2, $l = 4k$, 2,048×2,048 matrix

k	σ_{k+1}	δ_{direct}	$\delta_{\text{“fast”}}$	t_{direct}	$t_{\text{“fast”}}$	$t_{\text{direct}}/t_{\text{“fast”}}$
10	.100E-01	.358E-01	.320E-01	.440E+00	.784E+00	.561
20	.100E-03	.283E-03	.331E-03	.765E+00	.979E+00	.781
30	.100E-05	.403E-05	.495E-05	.114E+01	.136E+01	.838
40	.100E-07	.245E-07	.400E-07	.152E+01	.187E+01	.813
50	.100E-09	.490E-09	.202E-07	.192E+01	.258E+01	.744
60	.100E-11	.416E-11	.296E-09	.231E+01	.332E+01	.696

(5.2) SVD of Subsection 5.2, $l = 10k$, 2,048×2,048 matrix

Table 5 (See Section 6.)

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
10	.100E-01	.358E-01	.954E-01	.405E+00	.714E+00	.567
20	.100E-03	.283E-03	.194E-02	.757E+00	.719E+00	1.05
30	.100E-05	.403E-05	.181E-04	.111E+01	.726E+00	1.52
40	.100E-07	.245E-07	.221E-06	.148E+01	.739E+00	2.00
50	.100E-09	.490E-09	.710E-08	.182E+01	.749E+00	2.43
60	.100E-11	.416E-11	.528E-10	.216E+01	.765E+00	2.83

(6.1) ID of Subsection 5.1 using a DCT, $l = k + 10$, 2,048×2,048 matrix

k	σ_{k+1}	δ_{direct}	δ_{fast}	t_{direct}	t_{fast}	$t_{\mathrm{direct}}/t_{\mathrm{fast}}$
10	.100E-01	.358E-01	.954E-01	.410E+00	.769E+00	.533
20	.100E-03	.283E-03	.194E-02	.774E+00	.788E+00	.982
30	.100E-05	.403E-05	.181E-04	.114E+01	.808E+00	1.41
40	.100E-07	.245E-07	.221E-06	.153E+01	.847E+00	1.80
50	.100E-09	.490E-09	.710E-08	.192E+01	.894E+00	2.15
60	.100E-11	.416E-11	.528E-10	.230E+01	.946E+00	2.44

(6.2) SVD of Subsection 5.3 using a DCT, $l = k+10$, 2,048×2,048 matrix

Table 6 (See Section 6.)

k	δ_{prev}	δ_{fast}	t_{prev}	t_{fast}	$t_{\mathrm{prev}}/t_{\mathrm{fast}}$
500	.501E-01	.728E-02	.241E+04	.394E+03	6.12
525	.463E-01	.463E-02	.243E+04	.428E+03	5.68
550	.275E-01	.395E-02	.250E+04	.451E+03	5.54
575	.195E-06	.551E-07	.264E+04	.494E+03	5.34
600	.430E-14	.105E-14	.275E+04	.528E+03	5.21

Table 7 (See Section 6): ID of Subsection 5.1, $l = 4k$, 32,768×32,768 matrix

References

[1] N. Ailon and B. Chazelle, *Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform*, SIAM J. Comput., (2007). To appear.

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

[3] H. Cheng, Z. Gimbutas, P.-G. Martinsson, and V. Rokhlin, *On the compression of low rank matrices*, SIAM J. Sci. Comput., 26 (2005), pp. 1389–1404.

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

[5] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, Maryland, third ed., 1996.

[6] S. A. Goreinov and E. E. Tyrtyshnikov, *The maximal-volume concept in approximation by low-rank matrices*, in Structured Matrices in Mathematics, Computer Science, and Engineering I: Proceedings of an AMS-IMS-SIAM Joint Summer Research Conference, University of Colorado, Boulder, June 27–July 1, 1999, V. Olshevsky, ed., vol. 280 of Contemporary Mathematics, Providence, RI, 2001, AMS Publications.

[7] M. Gu and S. C. Eisenstat, *Efficient algorithms for computing a strong rank-revealing QR factorization*, SIAM J. Sci. Comput., 17 (1996), pp. 848–869.

[8] S. Mallat, *A Wavelet Tour of Signal Processing*, Academic Press, San Diego, CA, second ed., 1999.

[9] P.-G. Martinsson, V. Rokhlin, and M. Tygert, *On interpolation and integration in finite-dimensional spaces of bounded functions*, Comm. Appl. Math. Comput. Sci., 1 (2006), pp. 133–142.

[10] ——, *A randomized algorithm for the approximation of matrices*, Tech. Rep. 1361, Yale University Department of Computer Science, June 2006.

[11] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery, *Numerical Recipes*, Cambridge University Press, Cambridge, UK, second ed., 1992.

[12] T. Sarlós, *Improved approximation algorithms for large matrices via random projections*, in Proceedings of FOCS 2006, the 47th Annual IEEE Symposium on Foundations of Computer Science, October 2006.

[13] ——, *Improved approximation algorithms for large matrices via random projections, revised, extended long form*. Manuscript in preparation for publication, currently available at http://www.ilab.sztaki.hu/∼stamas/publications/rp-long.pdf, 2006.

[14] E. E. Tyrtyshnikov, *Incomplete cross approximation in the mosaic-skeleton method*, Computing, 64 (2000), pp. 367–380.