Low-rank approximation in the Frobenius norm by column and row subset selection

Alice Cortinovis* Daniel Kressner†

August 18, 2019

Abstract

A CUR approximation of a matrix A is a particular type of low-rank approximation $A \approx CUR$, where C and R consist of columns and rows of A, respectively. One way to obtain such an approximation is to apply column subset selection to A and A^T. In this work, we describe a numerically robust and much faster variant of the column subset selection algorithm proposed by Deshpande and Rademacher, which guarantees an error close to the best approximation error in the Frobenius norm. For cross approximation, in which U is required to be the inverse of a submatrix of A described by the intersection of C and R, we obtain a new algorithm with an error bound that stays within a factor $k+1$ of the best rank-k approximation error in the Frobenius norm. To the best of our knowledge, this is the first deterministic polynomial-time algorithm for which this factor is bounded by a polynomial in k. Our derivation and analysis of the algorithm is based on derandomizing a recent existence result by Zamarashkin and Osinsky. To illustrate the versatility of our new column subset selection algorithm, an extension to low multilinear rank approximations of tensors is provided as well.

1 Introduction

Given an $m \times n$ matrix A and an integer k, typically much smaller than m and n, the *column subset selection problem* aims at determining an index set $I \subset \{1, \dots, n\}$ of cardinality k such that the corresponding k columns $A(:, I)$ represent a good approximation of the range of A. This problem has broad applications in a diversity of disciplines, including scientific computing, model reduction, and statistical data analysis. While column subset selection is a classical problem in numerical linear algebra, closely connected to rank-revealing QR factorizations [9, 10, 23], new significant theoretical and algorithmic developments have been achieved during the last two decades within the model reduction and theory of algorithms communities. In particular, this concerns the interplay between column subset selection and interpolation [3, 11, 18] as well

*Institute of Mathematics, EPF Lausanne, 1015 Lausanne, Switzerland. E-mail: alice.cortinovis@epfl.ch. The work of Alice Cortinovis has been supported by the SNSF research project *Fast algorithms from low-rank updates*, grant number: 200020_178806.

†Institute of Mathematics, EPF Lausanne, 1015 Lausanne, Switzerland. E-mail: daniel.kressner@epfl.ch.

as the development and analysis of randomized algorithms, see [6, 17, 19, 36] for a few references representing this research direction.

This paper is concerned with algorithmic improvements and extensions of the seminal work by Deshpande, Rademacher and co-authors [14, 15] on column subset selection. In [15] the existence of an index set I such that

$$\|A - A(:,I)A(:,I)^+A\|_F^2 \leq (k+1)\left(\sigma_{k+1}(A)^2 + \ldots + \sigma_{\min\{m,n\}}(A)^2\right) \qquad (1)$$

has been established. Here, $\|\cdot\|_F$ and $(\cdot)^+$ denote the Frobenius norm and the Moore–Penrose inverse of a matrix, respectively. We let $\sigma_1(A) \geq \sigma_2(A) \geq \cdots$ denote the singular values of A. Note that $A(:,I)A(:,I)^+$ is an orthogonal projector and the bound (1) measures how well all the columns of A are approximated by the subset of column contained in I. The bound (1) is remarkable because the singular value decomposition (SVD) of A implies that the best approximation error $\|A - QQ^+A\|_F^2$ attained by an *arbitrary* $m \times k$ matrix Q is given by $\sigma_{k+1}(A)^2 + \ldots + \sigma_{\min\{m,n\}}(A)^2$. The bound (1) is larger by a factor that is only linear in k. More generally, we will call any quasi-optimal bound with a factor that is at most polynomial in k (and independent of m, n or A) a *polynomial bound*. The proof of (1) proceeds by defining a suitable discrete probability distribution on index tuples such that the expected value of the error with respect to this distribution satisfies the bound. This then implies the existence of at least one index set satisfying the bound as well. We remark that the factor $k+1$ in (1) cannot be improved [15, Proposition 3.3]. In [14], a deterministic algorithm has been developed by derandomizing this approach using the method of conditional expectations. These conditional expectations are given in terms of coefficients of certain characteristic polynomials and the algorithm from [14] attains efficiency by cheaply updating these coefficients. However, it is well known that working with characteristic polynomials in finite precision arithmetic is prone to massive numerical cancellation [29] and, as we will see, the algorithm from [14] is also affected by numerical instability. Our first contribution, presented in Section 2, consists of deriving a formulation of the algorithm that updates singular values instead of coefficients of characteristic polynomials. While our new variant enjoys the same favorable complexity, numerical experiments with matrices of different singular value decay indicate that it is numerically robust, achieving (1) up to the level of roundoff error. Based on a minor extension of the theory from [14, 15], we will also present a modification of the column selection strategy that results in significant speed ups of the algorithm.

In Section 3, we extend the developments from [14] to the problem of determining a rank-k approximation of the form

$$A \approx CUR,$$

where $C = A(:,J)$ and $R = A(I,:)$ contain k selected columns and rows of A, respectively. There is a simple and well established strategy to derive such an approximation, see, e.g., [17, 31]: One first applies column subset selection to A and A^T in order to determine C and R, respectively. Given C and R, the choice

$$U = C^+AR^+ \qquad (2)$$

then minimizes the Frobenius norm error. We will show that this strategy combined with (1) results in an error that is at most a factor $\sqrt{2k+2}$ larger

than the best rank-k approximation error. While this is clearly a favorable bound, the choice (2) comes with a disadvantage. It involves the full matrix A, but sometimes only partial information on A is available and has been used to determine I, J. One example for such a situation is the Chebfun2 construction for approximating bivariate functions [34], which uses a coarse discretization to cheaply determine I, J and then evaluates the full matrix A only along the cross containing the rows and columns determined by I and J, respectively. The choice $U = A(I, J)^{-1}$ then leads to a rank-k approximation of the form

$$A \approx A(:, J)A(I, J)^{-1}A(I, :),$$

which is often called cross approximation. Choosing I, J via column subset selection is not advisable in this setting; it may lead to (nearly) singular $A(I, J)$ and result in an unfavorable approximation error. On the other hand, Goreinov and Tyrtyshnikov [21] have established a polynomial bound for cross approximation in the maximum norm when choosing I, J such that the volume of $A(I, J)$ is maximal. Recently, Zamarashkin and Osinsky [38] derived a polynomial bound in the Frobenius norm by extending the techniques from [15]. However, as far as we know, there is no polynomial time deterministic algorithm that guarantees a polynomial bound (in any norm); popular greedy algorithms lead to exponential bounds [12, 24] at best. One major contribution of this work is to derive such an algorithm via an extension of [14]; our algorithm guarantees a Frobenius norm error that is at most a factor $k + 1$ larger than the best approximation error.

Section 4 contains an extension to tensors. In particular, we derive a deterministic algorithm that obtains a multilinear low-rank approximation that is constructed from the fibers of the tensor and satisfies a polynomial bound. Although our approach is a relatively straightforward extension of (2) and related approaches have been proposed in the literature [16, 22, 30], we are not aware that such an algorithm has been explicitly spelled out and analyzed.

2 Column subset selection

We start by providing more details on the approach from [14, 15] for the column subset selection problem. In the following we consider a matrix $A \in \mathbb{R}^{m\times n}$ with $m \leq n$ and rank at least k. We let a_i denote the ith column of A and $\pi_{i_1,\dots,i_k}A$ the orthogonal projection of A on the subspace spanned by the columns $a_{i_1}, \dots, a_{i_k}$, that is,

$$\pi_{i_1,\dots,i_k}A := A(:, I) \cdot A(:, I)^{+} \cdot A = QQ^T A,$$

where $I = (i_1, \dots, i_k) \in \{1, \dots, n\}^k$ and Q denotes an orthonormal basis of $A(:, I)$. Let us emphasize that I is now a *tuple*. Although order is not important and we are ultimately interested in an index *set*, working with tuples simplifies the subsequent definition and manipulation of probability distributions. The *volume* of a rectangular matrix $B \in \mathbb{R}^{m\times k}$ with $k \leq m$ is defined as $\mathrm{Vol}(B) := \prod_{i=1}^{k} \sigma_i(B)^2$. Note that $\mathrm{Vol}(B)^2 = \det(B^T B)$.

We now define a discrete probability distribution on integer tuples of the form $I \in \{1, \dots, n\}^k$ corresponding to a selection of k columns from A. For this purpose, let $X = (X_1, \dots, X_k)$ be a k-tuple of random variables with values in

$\{1, \ldots, n\}$ such that

$$\mathbb{P}(X = I) := \frac{\mathrm{Vol}(A(:, I))^2}{\sum_{J \in \{1,\ldots,n\}^k} \mathrm{Vol}(A(:, J))^2}. \tag{3}$$

By convention, $\mathrm{Vol}\,(A(:, I)) = 0$ whenever $i_1, \ldots, i_k$ contain repeated indices. Then [15, Theorem 1.3] shows that

$$\mathbb{E}[\|A - \pi_{X_1,\ldots,X_k} A\|_F^2] \leq (k+1)\left(\sigma_{k+1}^2 + \ldots + \sigma_m^2\right). \tag{4}$$

In particular, this implies the existence of I satisfying this bound.

In view of (3) and the prominent role played by maximum volume submatrices in low-rank approximation [21], it is tempting to expect that the k columns of maximum volume satisfy (1). However, not only that it is NP hard to choose such columns [8], but they also fail to have this property. For instance, for $k = 1$ consider the $2 \times n$ matrix

$$A = \begin{bmatrix} a(1+\varepsilon) & b & b & \ldots & b \\ -b(1+\varepsilon) & a & a & \ldots & a \end{bmatrix}$$

with $a^2 + b^2 = 1$ and $\varepsilon > 0$. The column of maximum volume (that is, of maximum Euclidean norm) is the first one. The approximation error obtained by this choice is given by $\|A - \pi_1 A\|_F^2 = n - 1$, which is much larger than $2\sigma_2^2 = 2(1+\varepsilon)^2$ for ε sufficiently small. Note that choosing any of the other columns yields the best approximation error $(1+\varepsilon)^2 = \sigma_2^2$.

2.1 Algorithm by Deshpande and Rademacher

Deshpande and Rademacher [14] derived a deterministic algorithm for column subset selection by derandomizing (4) using the method of conditional expectations.

More specifically, the first step of the algorithm chooses an index i_1 such that

$$\mathbb{E}\left[\|A - \pi_{X_1,\ldots,X_k} A\|_F^2 \mid X_1 = i_1\right]$$

is minimized. By construction, this quantity still satisfies the bound (4). More generally, having $t-1$ indices $i_1, \ldots, i_{t-1}$ selected, step t chooses an index i_t such that

$$\mathbb{E}\left[\|A - \pi_{X_1,\ldots,X_k} A\|_F^2 \mid X_1 = i_1, \ldots, X_{t-1} = i_{t-1}, X_t = i_t\right] \tag{5}$$

is minimized. After k steps we arrive at an index set I of cardinality k such that the desired bound (1) holds.

For the algorithm to be practical, it is crucial to compute the conditional expectations (5) efficiently. Lemma 21 in [14] shows that

$$\mathbb{E}\left[\|A - \pi_{X_1,\ldots,X_k} A\|_F^2 \mid X_1 = i_1, \ldots, X_t = i_t\right] = (k-t+1)\frac{c_{m-k+t-1}(BB^T)}{c_{m-k+t}(BB^T)},$$

where the right-hand side involves the matrix $B = A - \pi_{i_1,\ldots,i_t} A$ and coefficients $c_j \equiv c_j(BB^T)$ of the characteristic polynomial

$$(-\lambda)^m + c_{m-1}(-\lambda)^{m-1} + \ldots + c_1(-\lambda) + c_0 := \det(BB^T - \lambda I). \tag{6}$$

It is therefore required to compute in every step for all values of i, the ratios

$$\frac{c_{m-k+t-1}(B_i B_i^T)}{c_{m-k+t}(B_i B_i^T)} \tag{7}$$

where $B_i = A - \pi_{i_1,\ldots,i_{t-1},i} A$.

In the following, we discuss the computation of (7) and show how the minimization problem (5) can be relaxed in order to accelerate the search for suitable indices.

2.2 Computation of characteristic polynomial coefficients

Assuming that the first $t-1$ indices have been selected, we set $B := A - \pi_{i_1,\ldots,i_{t-1}} A$. Then

$$B_i = B - \pi_i B = \left(I - \frac{b_i b_i^T}{\|b_i\|_2^2}\right) B$$

is a rank-1 modification of B. Deshpande and Rademacher [14] propose two methods to compute (7) for $i = 1, \ldots, n$. In the following, we summarize them briefly.

1. Algorithm 2 in [14] computes BB^T explicitly and then computes $B_i B_i^T$ as a rank-2 update of BB^T for every $i = 1, \ldots, n$. The characteristic polynomial of $B_i B_i^T$ is computed by establishing a similarity transformation to a matrix in Frobenius normal form [7, Section 16.6]. Fast matrix-matrix multiplication and inversion can be exploited so that the cost of this approach is $O(nm^\omega \log m)$, where $\omega \leq 2.373$ is the best exponent of matrix-matrix multiplication complexity.

2. Algorithm 3 in [14] computes the thin SVD of $B = U\Sigma V^T$, the characteristic polynomial of BB^T from the squared singular values of B, and the auxiliary polynomials $g_j(x) = \prod_{\ell \neq j} \left(x - \sigma_\ell(B)^2\right)$ for $j = 1, \ldots, m$. For $h = m - k + t$ and $h = m - k + t - 1$, the coefficient $c_h(B_i B_i^T)$ can then be computed as the coefficient of x^h in

$$\det(\lambda I - BB^T) + \frac{1}{\|b_i\|_2^2} \sum_{j=1}^{n} \sigma_j^2(B) v_{ij}^2 g_j(x). \tag{8}$$

   The cost of this second approach is $O(m^2 n)$.

The problem of computing the Frobenius normal form of a matrix is "numerically not viable" [28]. Also, updating directly the characteristic polynomial as in (8) is prone to numerical cancellation, leading to inaccurate results. For instance, consider the 2×2 matrix

$$A = \begin{bmatrix} 6.583644 \cdot 10^{-7} & 8.113362 \cdot 10^{-3} \\ 8.113362 \cdot 10^{-3} & 100 \end{bmatrix},$$

and the column selection problem for $k = 1$. Algorithm 4 in [14] using (8) selects the first column, giving an error $\|A - A(:,1)A(:,1)^+ A\|_F \approx 1.2 \cdot 10^{-6} \gg \sqrt{2}\sigma_2(A) = 1.4 \cdot 10^{-10}$.

Therefore, from now on we will avoid updating coefficients of characteristic polynomials and work with singular values instead. More specifically, we will compute the singular values of B_i by updating the SVD of B and then apply the Summation Algorithm [29, Algorithm 1] to compute the coefficients of the characteristic polynomial of $B_i B_i^T$ from its eigenvalues (that is, the squared singular values) with $O(m^2)$ operations in a numerically forward stable manner. To describe the updating procedure, consider the (thin) SVD

$$B = U\Sigma V^T, \qquad U \in \mathbb{R}^{m\times m}, \quad \Sigma \in \mathbb{R}^{m\times m}, \quad V \in \mathbb{R}^{n\times m}.$$

The (nonzero) singular values of B_i and

$$U^T B_i V = (I - U^T \pi_i U) U^T B V = \left(I - \frac{U^T b_i b_i^T U}{\|b_i\|_2^2} \right) \Sigma = \left(I - qq^T \right) \Sigma,$$

with $q = U^T b_i / \|b_i\|_2$, are identical. Using standard bulge chasing algorithms (see, e.g., [37, Algorithm 3.4] and [2]) it is possible to find orthogonal matrices $Q, W \in \mathbb{R}^{m\times m}$ such that $Q^T q = e_1$, where e_1 denotes the first unit vector, and $Q^T \Sigma W$ is upper bidiagonal. In turn, the singular values can be computed from the bidiagonal matrix

$$Q^T (I - qq^T) \Sigma W = (I - e_1 e_1^T)(Q^T \Sigma W).$$

The matrices Q and W are composed of $O(m^2)$ Givens rotations [20, Section 5.1] and the computation of $Q^T \Sigma W$ requires to apply each of these rotations to at most 3 vectors. In turn, the cost of computing this bidiagonal matrix is $O(m^2)$, which is identical with the cost of computing its singular values [20, Section 8.6].

2.3 Overall algorithm

The described variation of the column subset selection algorithm by Deshpande and Rademacher is summarized in Algorithm 1. One execution of line 3 is $O(nm^2)$, lines 6–9 are $O(m^2)$, and lines 14–15 are $O(knm)$. In summary, the overall complexity of Algorithm 1 is $O(knm^2)$. This is identical with the complexity of [14, Algorithm 4] combined with [14, Algorithm 3], and it is better than [14, Algorithm 4] combined with [14, Algorithm 2].

Note that instead of lines 14–15 we could have updated $B \leftarrow B - \pi_{i_t} B$. However, we noticed that recomputing B in lines 18–19 tends to improve accuracy and it does not change the overall complexity.

2.4 Early stopping of column search

For each column index, Algorithm 1 needs to traverse $O(n)$ columns in order to find the one that minimizes the coefficient ratio or, equivalently, the conditional expectation. This column search can be shortened. To describe the idea, suppose that $i_1, \ldots, i_{t-1}$ have already been selected such that

$$\mathbb{E}\left[\|A - \pi_{X_1,\ldots,X_k} A\|_F^2 \mid X_1 = i_1, \ldots, X_{t-1} = i_{t-1} \right] \leq (k+1)(\sigma_{k+1}^2 + \ldots + \sigma_m^2) \tag{9}$$

holds. Now, we can choose *any* i_t such that

$$\mathbb{E}\left[\|A - \pi_{X_1,\ldots,X_k} A\|_F^2 \mid X_1 = i_1, \ldots, X_t = i_t \right] \leq (k+1)(\sigma_{k+1}^2 + \ldots + \sigma_m^2) \tag{10}$$

Algorithm 1 Column Subset Selection

Input: $A \in \mathbb{R}^{m \times n}$, rank $1 \leq k < m$
Output: Column indices $S \in \{1, \ldots, n\}^k$

1: Initialize $S = \emptyset$ and $B = A$
2: **for** $t = 1, \ldots, k$ **do**
3: $[U, \Sigma, \sim] = \text{svd}(B)$
4: minRatio $= +\infty$
5: **for** $i = 1, \ldots, n$ **do**
6: $q = U^T b_i / \|b_i\|_2$
7: $D = Q^T \Sigma W$ bidiagonal matrix obtained by bulge chasing [37, Algorithm 3.4]
8: Compute singular values $\sigma_1, \ldots, \sigma_m$ of $(I - e_1 e_1^T)D$
9: Apply Summation Algorithm [29, Algorithm 1] to compute $c_{m-k+t-1}(B_i B_i^T)$ and $c_{m-k+t}(B_i B_i^T)$ from eigenvalues $\sigma_1^2, \ldots, \sigma_m^2$
10: Set ratio $= c_{m-k+t-1}(B_i B_i^T)/c_{m-k+t}(B_i B_i^T)$
11: **if** ratio $<$ minRatio **then** Set minRatio $=$ ratio and $i_t = i$ **end if**
12: **end for**
13: Append index $S \leftarrow (S, i_t)$
14: $[Q, \sim] = \text{qr}(A(:, S))$
15: $B = A - QQ^T A$
16: **end for**

holds. The existence of i_t is guaranteed by (9) but we do not need to find the one that minimizes the conditional expectation. It suffices to always choose in every step an index such that (10) is verified. By induction, the error bound (1) still holds.

The discussion above suggests to modify Algorithm 1 such that it computes

$$\text{bound} = (k+1) \cdot \left(\sigma_{k+1}^2 + \ldots + \sigma_m^2\right)$$

in the beginning and substitute line 11 with

11: **if** $(k - t + 1) \cdot \text{ratio} \leq$ bound **then** Set $i_t = i$ and break **end if**

To be able to stop the search early, it is important to test the columns in a suitable order. We found it beneficial to test the columns of B in descending norm. For each step t, computing the norms of all columns of B and sorting them has complexity $O(mn + n \log n)$. Although this choice is clearly heuristic, the following lemma provides some justification for it by showing that the column of largest norm is the right choice for $k = 1$ provided that all other columns are sufficiently small.

Lemma 1. *Let $A = \begin{bmatrix} a_1 & A_2 \end{bmatrix}$. If $\|A_2\|_F \leq \|a_1\|_2$ then choosing the first column solves the column selection problem for $k = 1$, that is,*

$$\|A - a_1 a_1^+ A\|_F^2 \leq 2(\sigma_2^2 + \ldots + \sigma_m^2).$$

Note that the condition of the lemma is satisfied if the column norms of A decay sufficiently fast, for instance if $\|a_i\|_2 \leq \frac{\|a_1\|_2}{i}$ for $i = 2, \ldots, n$.

Proof. Without loss of generality we may assume that $\|a_1\|_2 = 1$. By setting $B = A_2 - a_1 a_1^+ A_2 = A_2 - a_1 a_1^T A_2$ and $b = A_2^T a_1$, we have

$$A^T A = \begin{bmatrix} 1 & b^T \\ b & A_2^T A_2 \end{bmatrix} = \begin{bmatrix} 1 & b^T \\ b & B^T B + bb^T \end{bmatrix}$$

and obtain

$$\|A^TA\|_2 \leq \left\| \begin{bmatrix} 1 & \|b\|_2 \\ \|b\|_2 & \|B^TB + bb^T\|_2 \end{bmatrix} \right\|_2 \leq \left\| \begin{bmatrix} 1 & \|b\|_2 \\ \|b\|_2 & \|B\|_F^2 + \|b\|_2^2 \end{bmatrix} \right\|_2. \quad (11)$$

Here, the first inequality is a norm-compression inequality [5, Section 9.10] and the second inequality follows from the fact that the involved matrices are positive.

We aim at proving

$$\|A - a_1 a_1^+ A\|_F^2 = \|B\|_F^2 \leq 2(\|A\|_F^2 - \|A\|_2^2),$$

which is equivalent to

$$\|A\|_2^2 \leq 1 + \|b\|_2^2 + \frac{\|B\|_F^2}{2} =: \gamma.$$

Thus, it remains to show that the larger eigenvalue of the symmetric positive definite 2×2 matrix on the right-hand side of (11) is bounded by γ. For this purpose, we note that its characteristic polynomial is given by

$$p(\lambda) = (\lambda - 1)\left(\lambda - \|b\|_2^2 - \|B\|_F^2\right) - \|b\|_2^2.$$

Setting $\gamma = 1 + \|b\|_2^2 + \|B\|_F^2/2$, we obtain

$$p(\gamma) = \|B\|_F^2/2 \cdot \left(1 - \|b\|_2^2 - \|B\|_F^2/2\right) \geq 0,$$

where we used that $\|b\|_2^2 + \|B\|_F^2 = \|A_2\|_F^2 \leq \|a_1\|_2^2 = 1$. Because p is a parabola with vertex $(1 + \|b\|_2^2 + \|B\|_F^2)/2 \leq \gamma$, it follows that the larger root of p is bounded by γ, which completes the proof. □

It is important to not draw too many conclusions from Lemma 1. Consider, for example, the matrix

$$A = \begin{bmatrix} 1 & 0 & 10^{-b} \\ 0 & 1 & 10^{-b} \\ 0 & 0 & 10^{-2b} \end{bmatrix}$$

for some integer b, say $b = 16$. For $k = 1$, the optimal choice is the third column, which is the one of smallest norm. This matrix also nicely illustrates that the obvious greedy approach (in order to get k columns of A, one first chooses the best column, then the best column in the orthogonal complement, and so on) comes with no guarantees and may, in fact, utterly fail. For $k = 2$ the optimal choice consists of the first two columns. On the other hand, the greedy approach for $k = 2$ first selects the third column and then the first column, resulting in the arbitrarily bad error ratio $\frac{\text{error greedy}}{\text{error best}} \approx 10^b$.

2.5 Numerical experiments

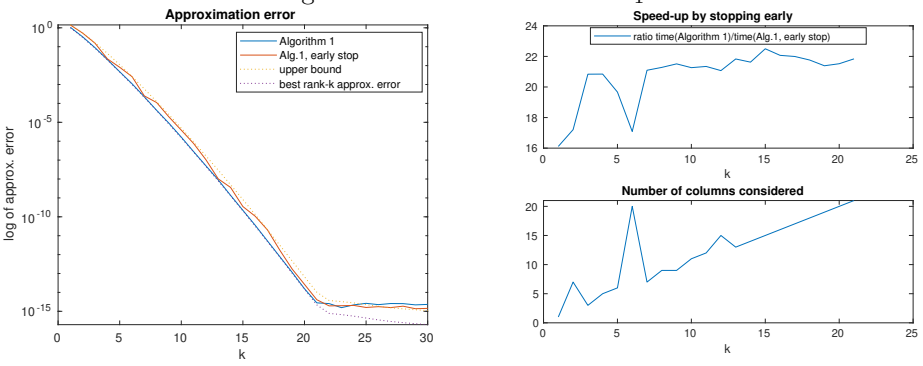
Both variants of Algorithm 1, without and with early stopping, have been implemented in Matlab version R2019a. As the bulge chasing algorithm in line 7 would perform poorly in Matlab, this part has been implemented in C++ and is called via a MEX interface. All numerical experiments in this work have been run on an eight-core Intel Core i7-8650U 1.90 GHz CPU, 256 KB of level 2 Cache and 16 GB of RAM. Multithreading has been turned off in order to not distort the findings.

We have applied the algorithm to the following three matrices:

1. the Hilbert matrix $A_1 \in \mathbb{R}^{200\times 200}$ given by $A_1(i,j) = \frac{1}{i+j-1}$;
2. $A_2 \in \mathbb{R}^{100\times 200}$ given by $A_2(i,j) = \exp(-0.3 \cdot |i-j|/200)$;
3. $A_3 \in \mathbb{R}^{100\times 200}$ given by $A_3(i,j) = \left(\left(\frac{i}{200}\right)^{20} + \left(\frac{j}{200}\right)^{20}\right)^{1/20}$.

The obtained results are shown in Figures 1, 2, and 3 respectively. Each left plot contains, for different values of k, the approximation error $\|A_i - A_i(:,S)A_i(:,S)^+\|_F$ returned by Algorithm 1, without and with early stopping. We compare with the best rank-k approximation error $\sqrt{\sigma_{k+1}^2 + \ldots + \sigma_m^2}$ and the upper bound (1), that is, $\sqrt{(k+1)(\sigma_{k+1}^2 + \ldots + \sigma_m^2)}$. It can be seen that both variants of our algorithm stay below the upper bound, until it reaches the level of roundoff error. Interestingly, for the matrix A_2, which features the slowest singular value decay, the observed approximation error is much closer to the best approximation error than to the upper bound. The right plots of the figures show, for different values of k, the ratio between the execution times of Algorithm 1 without early stopping and with early stopping. For the variant with early stopping, we also plot the number of columns that were examined. In the most optimistic scenario, only k columns need to be examined, which means that in every step of the algorithm already the first verifies the desired criterion. The plots reveal that our algorithm actually stays pretty close to this ideal situation, at least for the matrices considered. Note that for values of k larger than the numerical rank of the matrix, Algorithm 1 starts computing ratios (6) from singular values of the order of machine precision. In turn, the computations are severely affected by roundoff error and it may, in fact, happen that the early stopping criterion is never satisfied. This leads to meaningless results and we therefore truncate the plots before this happens. A proper implementation of Algorithm 1 needs to detect such a situation and reduce k accordingly.

Figure 1: Results for matrix A_1



3 Matrix approximation

In this section, we extend the developments from Section 2 on column subset selection to compute certain low-rank matrix approximations of a matrix $A \in \mathbb{R}^{m\times n}$ with $m \leq n$. As already discussed in the introduction, we will pursue two

Figure 2: Results for matrix A_2

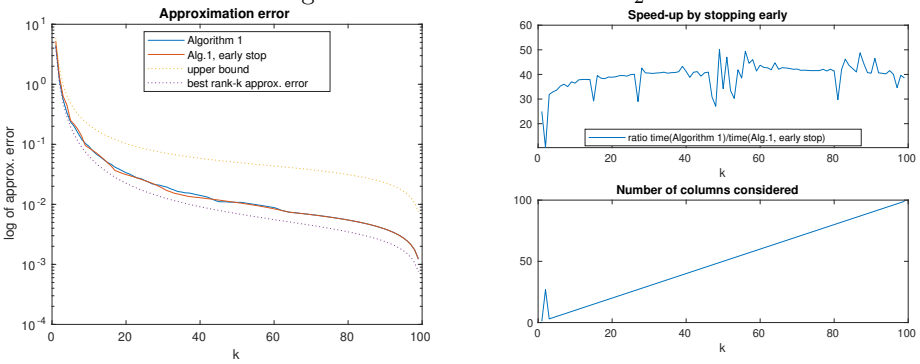
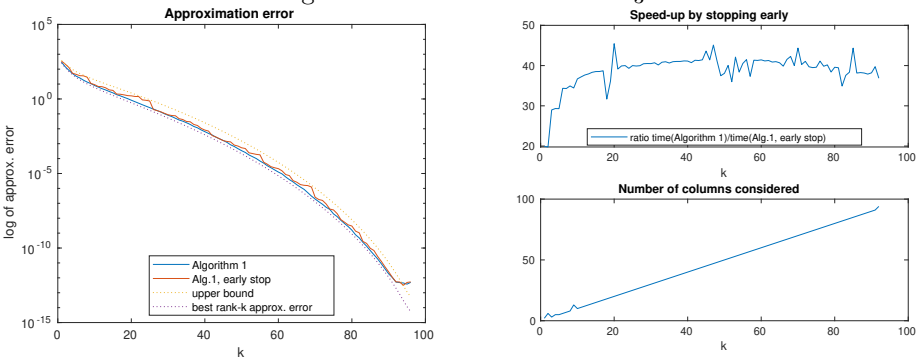


Figure 3: Results for matrix A_3



ways. First, in Section 3.1, we discuss a general CUR approximation obtained from applying column subset selection to the columns and rows of the matrix. Second, in Section 3.2, we present a novel approach to cross approximation, a specific type of CUR approximation, with guaranteed error bounds.

3.1 CUR approximation induced by column subset selection

Suppose that $C \in \mathbb{R}^{m\times k}$ and $R \in \mathbb{R}^{k\times n}$ have been chosen. Then the matrix $U \in \mathbb{R}^{k\times k}$ that minimizes $\|A-CUR\|_F$ is given by the projection $U = C^+AR^+$, see [33, p. 320]. The following corollary provides an error bound for the case when C and R are determined by the techniques from Section 2, leading to Algorithm 2. Closely related results can be found in the literature; see, for example, [17, Theorem 4], [30, Corollary 3.5], and [32, Theorem 4.1].

Algorithm 2 Matrix approximation by column subset selection

Input: $A \in \mathbb{R}^{m\times n}$, rank k

Output: Rank-k CUR approximation, with C, R containing columns and rows of A

1: Compute C by applying Algorithm 1 to select k columns of A
2: Compute R by applying Algorithm 1 to select k columns of A^T
3: Compute $U = C^+AR^+$

Corollary 2. *Let $A \in \mathbb{R}^{m\times n}$, with $1 \le k \le m \le n$. Then the CUR approximation returned by Algorithm 2 satisfies*

$$\|A - CUR\|_F \le \sqrt{2k+2}\sqrt{\sigma_{k+1}(A)^2 + \cdots + \sigma_m(A)^2}.$$

Proof. Using the inequality (1) twice and the fact that CC^+ is an orthogonal projection, we obtain

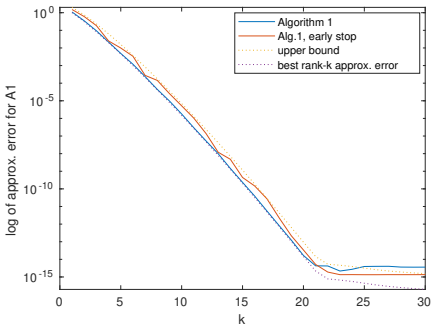
$$\begin{aligned}
\|A - CUR\|_F^2 &= \|A - CC^+AR^+R\|_F^2 \\
&= \|A - CC^+A\|_F^2 + \|CC^+(A - AR^+R)\|_F^2 \\
&\le \|(I - CC^+)A\|_F^2 + \|A(I - R^+R)\|_F^2 \\
&\le 2(k+1)\left(\sigma_{k+1}(A)^2 + \cdots + \sigma_m(A)^2\right).
\end{aligned}$$

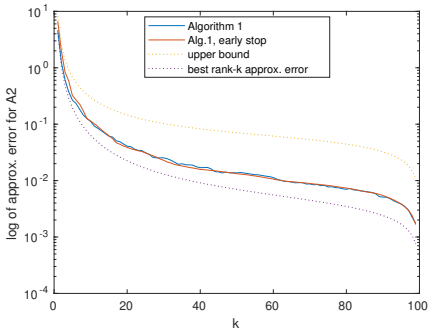
□

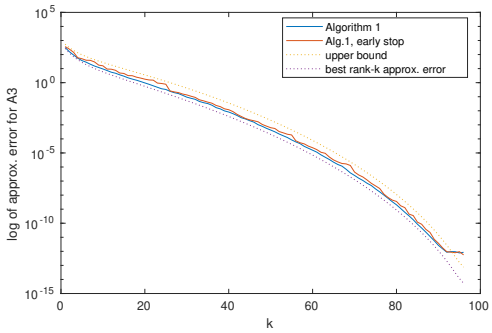
3.1.1 Numerical experiments

We have tested a Matlab implementation of Algorithm 2 in the setting and for the matrices A_1, A_2, A_3 described in Section 2.5. Figure 4 displays the obtained approximation errors $\|A_i - CUR\|_F$ for different values of k. Again, we have tested both variants of Algorithm 1, without and with early stopping, within Algorithm 2. The speedups obtained from early stopping are very similar to the ones reported Section 2.5 and, therefore, we refrain from providing details.

Figure 4: Approximation errors for matrices A_1 (top left), A_2 (top right), and A_3 (bottom).







We also consider, for $0 < \alpha < 1$, the $n \times n$ matrix

$$A = Q \cdot \mathrm{diag}(1, \alpha, \alpha^2, \ldots, \alpha^{n-1}) \cdot Q^T,$$

where $Q \in \mathbb{R}^{n \times n}$ is determined as the orthogonal factor from the QR decomposition of

$$\begin{bmatrix} 1 & & & & \\ -1 & 1 & & & \\ -1 & -1 & 1 & & \\ \vdots & \vdots & & \ddots & \\ -1 & -1 & -1 & \cdots & 1 \end{bmatrix}.$$

This is known to be a challenging example for the CUR approximation induced by DEIM (discrete interpolation method); see [32, Section 4.2], which determines the row and column indices by greedily choosing a maximum volume submatrix of U_k and V_k containing the first k left and right singular vectors of A, respectively. For the example above, the DEIM induced CUR approximation always chooses $1, \ldots, k$ for the column and row indices. For $\alpha = 0.1$, $n = 6$, $k = 5$, the error resulting from this choice is given by

$$\|A - A(:, 1:5)A(:, 1:5)^+ AA(1:5, :)^+ A(1:5, :)\|_F \approx 2.6 \cdot 10^{-9},$$

which is a magnitude larger than the upper bound $\sqrt{2(k+1)}\sigma_6 \approx 3.5 \cdot 10^{-10}$ guaranteed by Algorithm 2. Note that the latter algorithm selects the last 5 rows and columns for this example, leading to an error of $\approx 1.3 \cdot 10^{-10}$.

3.2 Cross approximation

We now consider *cross approximations*, which take the form

$$A \approx A(:, J)A(I, J)^{-1}A(I, :) \tag{12}$$

for row/column index tuples

$$(I, J) \in \Omega := \{1, \ldots, m\}^k \times \{1, \ldots, n\}^k.$$

The different choice of the middle matrix makes a fundamental difference. In particular, as the following example shows, choosing the indices I, J as in Algorithm 2 may lead to poor approximation error.

Example 3. *Consider* $A = \begin{bmatrix} 2\varepsilon & 1 \\ 1 & \varepsilon \end{bmatrix}$ *for* $\varepsilon > 0$ *and* $k = 1$. *Clearly, the first column and row satisfy the bound* (1) *for* $k = 1$ *with respect to* A *and* A^T, *respectively. However, the error of the corresponding cross approximation,* $\|A - A(:, 1)A(1, 1)^{-1}A(1, :)\|_F = \frac{1}{2\varepsilon} - \varepsilon$, *becomes arbitrarily large as* $\varepsilon \to 0$.

Zamarashkin and Osinsky [38] have shown the existence of a cross approximation that satisfies a polynomial error bound in the Frobenius norm. To summarize their result, let

$$(X, Y) = (X_1, \ldots, X_k, Y_1, \ldots, Y_k)$$

be a $(2k)$-tuple of random variables with values in Ω such that

$$\mathbb{P}\left(X = I, Y = J\right) := \frac{\operatorname{Vol}\left(A(I,J)\right)^2}{\sum_{(I',J')\in\Omega} \operatorname{Vol}(A(I',J'))^2}. \tag{13}$$

Note that $\operatorname{Vol}\left(A(I,J)\right) = 0$ whenever $i_1, \ldots, i_k$ or $j_1, \ldots, j_k$ contain repeated indices. Then [38, Theorem 1] shows that

$$\mathbb{E}[\|A - A(:,Y)A(X,Y)^{-1}A(X,:)\|_F^2] \leq (k+1)^2 \left(\sigma_{k+1}^2 + \ldots + \sigma_m^2\right). \tag{14}$$

In particular, this implies that there exists $(I,J) \in \Omega$ such that

$$\|A - A(:,J)A(I,J)^{-1}A(I,:)\|_F^2 \leq (k+1)^2 \left(\sigma_{k+1}^2 + \ldots + \sigma_m^2\right). \tag{15}$$

In analogy to Section 2.1 and [14], we will now derandomize this result producing a polynomial-time deterministic algorithm that returns a cross approximation satisfying (15). The key for doing so is to find an expression for the conditional expectations that is easy to work with.

3.2.1 Conditional expectations

Lemma 4. *Let $1 \leq t \leq k$ and $(i_1, \ldots, i_t, j_1, \ldots, j_t)$ be such that*

$$\mathbb{P}\left(\begin{smallmatrix} X_1 = i_1, \ldots, X_t = i_t \\ Y_1 = j_1, \ldots, Y_t = j_t \end{smallmatrix}\right) > 0$$

for a random $(2k)$-tuple (X,Y) with the probability distribution defined by (13)*. Consider*

$$B = A - A(:, [j_1 \quad \cdots \quad j_t])A([i_1 \quad \cdots \quad i_t], [j_1 \quad \cdots \quad j_t])^{-1}A([i_1 \quad \cdots \quad i_t], :),$$

the remainder of cross approximation after choosing row indices $i_1, \ldots, i_t$ and column indices $j_1, \ldots, j_t$. Then

$$\begin{aligned} &\mathbb{E}\left[\|A - A(:,Y)A(X,Y)^{-1}A(X,:)\|_F^2 \,\middle|\, \begin{smallmatrix} X_1 = i_1, \ldots, X_t = i_t \\ Y_1 = j_1, \ldots, Y_t = j_t \end{smallmatrix}\right] \\ &= (k-t+1)^2 \cdot \frac{c_{m-k+t-1}(BB^T)}{c_{m-k+t}(BB^T)}, \end{aligned}$$

with the coefficients $c_{m-k+t}, c_{m-k+t-1}$ defined as in (6)*.*

Proof. To simplify notation, we let $I_1 = (i_1, \ldots, i_t)$, $I_2 = (i_{t+1}, \ldots, i_k)$, $I = (I_1, I_2) = (i_1, \ldots, i_k)$ and define J_1, J_2, J analogously. In the following, we always use the convention that row and column summation indices range from 1 to m and from 1 to n, respectively. We have that

$$\begin{aligned} &\mathbb{E}\left[\|A - A(:,Y)A(X,Y)^{-1}A(X,:)\|_F^2 \,\middle|\, \begin{smallmatrix} X_1 = i_1, \ldots, X_t = i_t \\ Y_1 = j_1, \ldots, Y_t = j_t \end{smallmatrix}\right] \\ &= \sum_{\substack{i_{t+1}, \ldots, i_k \\ j_{t+1}, \ldots, j_k}} \|A - A(:,J)A(I,J)^{-1}A(I,:)\|_F^2 \cdot \mathbb{P}\left(X = I, Y = J \,\middle|\, \begin{smallmatrix} X_1 = i_1, \ldots, X_t = i_t \\ Y_1 = j_1, \ldots, Y_t = j_t \end{smallmatrix}\right) \\ &= \frac{1}{\gamma} \cdot \sum_{\substack{i_{t+1}, \ldots, i_k, i_{k+1} \\ j_{t+1}, \ldots, j_k, j_{k+1}}} \operatorname{Vol}\big(A((I, i_{k+1}), (J, j_{k+1}))\big)^2, \end{aligned} \tag{16}$$

with

$$\gamma = \sum_{\substack{i_{t+1},\dots,i_k \\ j_{t+1},\dots,j_k}} \mathrm{Vol}(A(I,J))^2.$$

For establishing the equality in (16) we used from [38, Lemma 1] that

$$\|A - A(:,J)A(I,J)^{-1}A(I,:)\|_F^2 = \frac{\sum_{i_{k+1},j_{k+1}} \mathrm{Vol}\big(A((I,i_{k+1}),(J,j_{k+1}))\big)^2}{\mathrm{Vol}(A(I,J))^2},$$

and, from (13), that

$$\mathbb{P}\left(X = I, Y = J \middle| \begin{smallmatrix} X_1=i_1,\dots,X_t=i_t \\ Y_1=j_1,\dots,Y_t=j_t \end{smallmatrix}\right) = \frac{\mathbb{P}(X=I,Y=J)}{\mathbb{P}\left(\begin{smallmatrix} X_1=i_1,\dots,X_t=i_t \\ Y_1=j_1,\dots,Y_t=j_t \end{smallmatrix}\right)} = \frac{1}{\gamma} \cdot \mathrm{Vol}(A(I,J))^2.$$

We now aim at simplifying the expression (16). For this purpose, we assume without loss of generality that $i_1 = 1, \dots, i_t = t$ and $j_1 = 1, \dots, j_t = t$. This allows us to partition

$$A(I,J) = \begin{bmatrix} A(I_1,J_1) & A(I_1,J_2) \\ A(I_2,J_1) & A(I_2,J_2) \end{bmatrix}, \quad B(I,J) = \begin{bmatrix} 0 & 0 \\ 0 & B(I_2,J_2) \end{bmatrix},$$

where $B(I_2,J_2) = A(I_2,J_2) - A(I_2,J_1)A(I_1,J_1)^{-1}A(I_1,J_2)$ by the definition of B. By the relation between determinants and Schur complements [25, Equation (0.8.5.1)], $\mathrm{Vol}(A(I,J)) = \mathrm{Vol}(A(I_1,J_1)) \cdot \mathrm{Vol}(B(I_2,J_2))$. Therefore,

$$\gamma = \sum_{\substack{i_{t+1},\dots,i_k \\ j_{t+1},\dots,j_k}} \mathrm{Vol}(A(I,J))^2 = \sum_{\substack{i_{t+1},\dots,i_k \\ j_{t+1},\dots,j_k}} \mathrm{Vol}(B(I_2,J_2))^2 \cdot \mathrm{Vol}(A(I_1,J_1))^2.$$

Analogously, one shows

$$\sum_{\substack{i_{t+1},\dots,i_{k+1} \\ j_{t+1},\dots,j_{k+1}}} \mathrm{Vol}\big(A((I,i_{k+1}),(J,j_{k+1}))\big)^2 = \sum_{\substack{i_{t+1},\dots,i_{k+1} \\ j_{t+1},\dots,j_{k+1}}} \mathrm{Vol}\big(B((I_2,i_{k+1}),(J_2,j_{k+1}))\big)^2 \mathrm{Vol}(A(I_1,J_1))^2.$$

Inserting these expressions into (16) yields

$$\frac{\sum_{\substack{i_{t+1},\dots,i_{k+1} \\ j_{t+1},\dots,j_{k+1}}} \mathrm{Vol}\big(B((I_2,i_{k+1}),(J_2,j_{k+1}))\big)^2}{\sum_{\substack{i_{t+1},\dots,i_k \\ j_{t+1},\dots,j_k}} \mathrm{Vol}(B(I_2,J_2))^2}.$$

By [26, Theorem 7] this ratio is equal to

$$\frac{c_{m-k+t-1}(BB^T) \cdot ((k-t+1)!)^2}{c_{m-k+t}(BB^T) \cdot ((k-t)!)^2} = (k-t+1)^2 \cdot \frac{c_{m-k+t-1}(BB^T)}{c_{m-k+t}(BB^T)}. \qquad \square$$

3.2.2 Derandomized cross approximation algorithm

With Lemma 4 at hand, we can proceed analogously to Section 2.1 and sequentially find k pairs of row/column indices such that (15) is satisfied. Suppose that $t-1$ index pairs $(i_1, j_1), \ldots, (i_{t-1}, j_{t-1})$ have been determined. Then the tth step of the algorithm proceeds by choosing (i_t, j_t) such that

$$\mathbb{E}\left[\|A - A(:,Y)A(X,Y)^{-1}A(X,:)\|_F^2 \,\middle|\, \begin{smallmatrix} X_1=i_1,\ldots,X_t=i_t \\ Y_1=j_1,\ldots,Y_t=j_t \end{smallmatrix}\right] \tag{17}$$

is minimized. We will show in Theorem 5 below that this choice of index pairs leads to a cross approximation satisfying the desired error bound (15). In view of Lemma 4, the minimization of (17) means that in each step of the algorithm we need to compute the ratios

$$\frac{c_{m-k+t-1}(C_{ij}C_{ij}^T)}{c_{m-k+t}(C_{ij}C_{ij}^T)}, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n, \tag{18}$$

where

$$C_{ij} = A - A(:, [j_1, \cdots, j_{t-1}, j])A([i_1, \cdots, i_{t-1}, i], [j_1, \cdots, j_{t-1}, j])^{-1}A([i_1, \cdots, i_{t-1}, i], :).$$

Parallelizing the developments in Section 2.2, we now show how the coefficients in (18) can be computed via updating the singular values of C_{ij}. Let us denote the remainder from the previous step by

$$B = A - A(:, [j_1, \cdots, j_{t-1}])A([i_1, \cdots, i_{t-1}], [j_1, \cdots, j_{t-1}])^{-1}A([i_1, \cdots, i_{t-1}], :).$$

Then it follows that

$$C_{ij} = B - \frac{1}{B(i,j)}B(:,j)B(i,:), \tag{19}$$

see, e.g., [4]. We compute a thin SVD $B = U\Sigma V^T$ such that $U \in \mathbb{R}^{m\times m}$, $V \in \mathbb{R}^{m\times n}$ have orthonormal columns and $\Sigma \in \mathbb{R}^{m\times m}$ is diagonal. Note that

$$B(:,j) = U\Sigma V(j,:)^T, \qquad B(i,:) = U(i,:)\Sigma V^T.$$

Inserted into (19), this shows that the nonzero singular values of C_{ij} match the singular values of

$$U^T C_{ij} V = \Sigma - \Sigma V(j,:)^T \cdot \frac{U(i,:)\Sigma}{B(i,j)} = \Sigma - xy^T,$$

where $x = \Sigma V(j,:)^T$ and $y = \frac{1}{B(i,j)}\Sigma U(i,:)^T$ are vectors of length m and can be computed with $O(m^2)$ operations.

Similarly as in Section 2.2, we transform $\Sigma - xy^T$ into bidiagonal form, after which its singular values can be computed with $O(m^2)$ operations. This transformation proceeds in three steps:

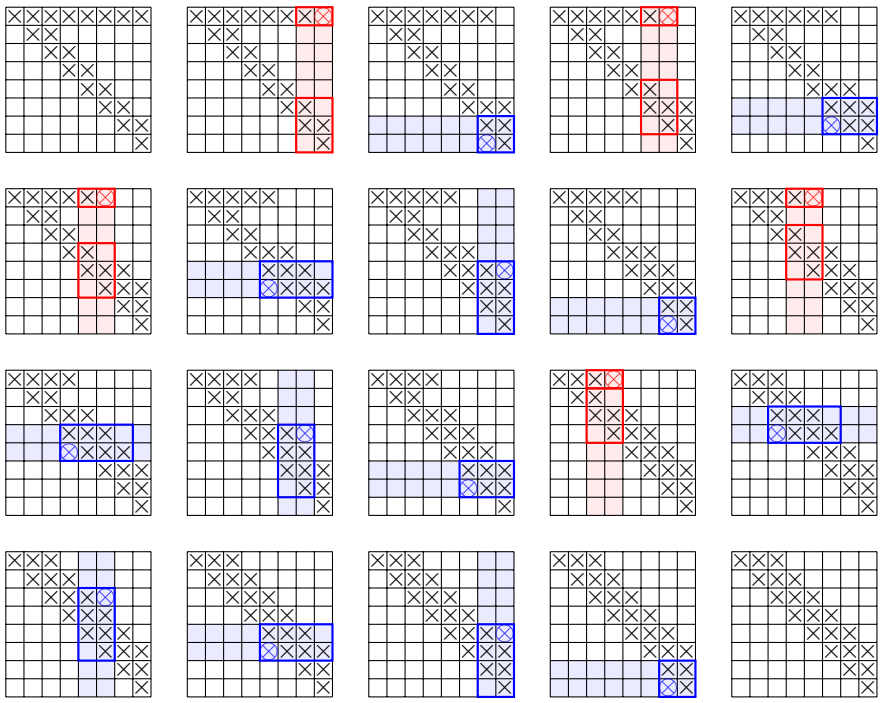
1. We compute orthogonal matrices Q and W such that $Q^T\Sigma W$ is upper bidiagonal and $Q^T x = \pm\|x\|_2 \cdot e_1$ using, for example, [37, Algorithm 3.4]. In turn, the matrix
$$D_1 := Q^T(\Sigma - xy^T)W \tag{20}$$
is bidiagonal with an additional nonzero first row; see the first plot in Figure 5 for an illustration.

2. By a bulge chasing algorithm, we transform D_1 to an upper banded matrix D_2 with two superdiagonals using $O(m^2)$ Givens rotations. We refrain from giving a detailed description of the algorithm and refer to Figure 5 for an illustration.

3. The banded matrix D_2 is reduced to a bidiagonal matrix D_3 using the LAPACK [1] routine `dgbbrd`.

The overall procedure described above can be implemended by means of $O(m^2)$ Givens rotations, each of which is applied to a small matrix of size independent of m, n. Hence, it has complexity $O(m^2)$.

Figure 5: Illustration of bulge chasing algorithm to transform a bidiagonal matrix with an additional nonzero first row to an upper banded matrix. In each plot, except for the first and last ones, a Givens rotation is applied to a pair of row or columns to zero out the entry denoted by $\otimes$.



Algorithm 3 summarizes our newly proposed method for cross approximation. The SVD needed at the beginning of each outer loop is of complexity $O(m^2 n)$ and each of the mn inner loops costs $O(m^2)$ operation; the total complexity of Algorithm 3 is therefore $O(km^3 n)$.

Theorem 5. *For a matrix A of rank at least k, Algorithm 3 returns index sets I and J such that* (15) *is satisfied.*

Algorithm 3 Derandomized cross approximation

Input: $A \in \mathbb{R}^{m\times n}$ with $m \le n$, integer $k \le m$
Output: Index sets I, J of cardinality k defining the cross approximation (12)

Initialize $I \leftarrow \emptyset, J \leftarrow \emptyset$, and $B \leftarrow A$
for $t = 1, \dots, k$ **do**
  $[U, \Sigma, V]$ = thin SVD of B
  minRatio $= +\infty$
  for $i = 1, \dots, m$ **do**
    for $j = 1, \dots, n$ **do**
      $x \leftarrow \Sigma V(j,:)^T$, $y \leftarrow \frac{1}{B(i,j)} \Sigma U(i,:)^T$
      Compute matrix D_1 defined in (20) using [37, Algorithm 3.4]
      Transform D_1 into upper banded form D_2 using bulge chasing algorithm
      Transform D_2 into bidiagonal matrix D_3 using LAPACK's `dgbbrd`
      Compute singular values $\sigma_1, \dots, \sigma_m$ of D_3
      Apply Summation Algorithm [29, Algorithm 1] to obtain $c_{m-k+t-1}(C_{ij}C_{ij}^T)$ and $c_{m-k+t}(C_{ij}C_{ij}^T)$ from eigenvalues $\sigma_1^2, \dots, \sigma_m^2$
      Set $r = \frac{c_{m-k+t-1}(C_{ij}C_{ij}^T)}{c_{m-k+t}(C_{ij}C_{ij}^T)}$
      if $r <$ minRatio **then** $i_t \leftarrow i$, $j_t \leftarrow j$, minRatio $= r$ **end if**
    end for
  end for
  $I \leftarrow I \cup \{i_t\}, J \leftarrow J \cup \{j_t\}$
  $B \leftarrow B - \frac{B(:,j_t)\cdot B(i_t,:)}{B(i_t,j_t)}$
end for

Proof. Let $B_{\{X,Y\}} = A - A(:,Y)A(X,Y)^{-1}A(X,:)$. For $t = 1, \dots, k$ we have that

$$\mathbb{E}\left[\|B_{\{X,Y\}}\|_F^2 \,\middle|\, \substack{X_1=i_1,\dots,X_{t-1}=i_{t-1} \\ Y_1=j_1,\dots,Y_{t-1}=j_{t-1}}\right]$$
$$= \sum_{i,j} \mathbb{E}\left[\|B_{\{X,Y\}}\|_F^2 \,\middle|\, \substack{X_1=i_1,\dots,X_{t-1}=i_{t-1},X_t=i \\ Y_1=j_1,\dots,Y_{t-1}=j_{t-1},Y_t=j}\right] \mathbb{P}\Big(X_t = i, Y_t = j_t \,\Big|\, \substack{X_1=i_1,\dots,X_{t-1}=i_{t-1} \\ Y_1=j_1,\dots,Y_{t-1}=j_{t-1}}\Big).$$

Therefore, as (14) holds, the choice (17) inductively ensures that

$$\begin{aligned}\mathbb{E}\left[\|B_{\{X,Y\}}\|_F^2 \,\middle|\, \substack{X_1=i_1,\dots,X_t=i_t \\ Y_1=j_1,\dots,Y_t=j_t}\right] &\le \mathbb{E}\left[\|B_{\{X,Y\}}\|_F^2 \,\middle|\, \substack{X_1=i_1,\dots,X_{t-1}=i_{t-1} \\ Y_1=j_1,\dots,Y_{t-1}=j_{t-1}}\right] \\ &\le (k+1)^2(\sigma_{k+1}^2 + \dots + \sigma_m^2).\end{aligned}$$

Therefore, the index sets I and J computed by Algorithm 3 satisfy the bound (15). □

In analogy to the discussion in Section 2.4, let us emphasize that it is not necessary to select the pair (i_t, j_t) that minimizes the ratio r. Any pair (i, j) for which the inequality

$$(k-t+1)^2 \frac{c_{m-k+t-1}(C_{ij}C_{ij}^T)}{c_{m-k+t}(C_{ij}C_{ij}^T)} \le (k+1)^2(\sigma_{k+1}(A)^2 + \dots + \sigma_m(A)^2) \qquad (21)$$

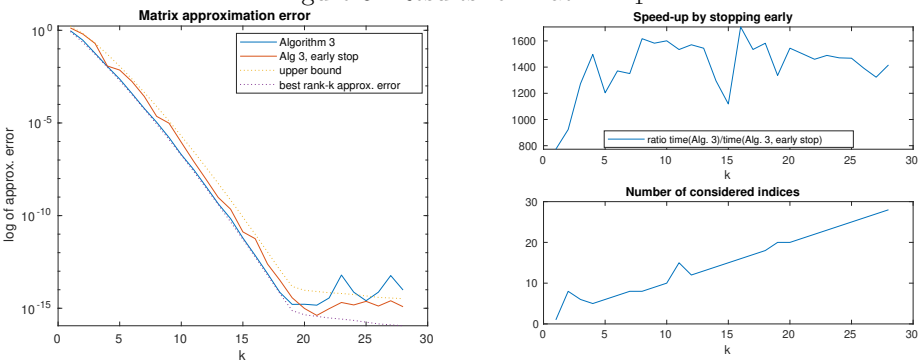
holds will lead to index sets I and J such that (15) is satisfied. Inspired by adaptive cross approximation with full pivoting [4], we traverse the entries of B from the largest to the smallest (in magnitude) and stop the search once we have found an index pair (i_t, j_t) satisfying (21).

3.2.3 Numerical experiments

We have implemented both variants of Algorithm 3, without and with early stopping, in Matlab. Again, the two inner loops have been implemented in a C++ function that is called via a MEX interface. The computational environment is the one described in Section 2.5 but the test matrices are smaller because Algorithm 3 without early stopping is significantly slower. We choose A_1 to be 100×100, A_2 to be 50×100, and the matrix $A_3 \in \mathbb{R}^{50\times 100}$ is given by $A_3(i,j) = \left(\left(\frac{i}{100}\right)^{10} + \left(\frac{j}{100}\right)^{10}\right)^{1/10}$.

The left plots of Figures 6, 7, 8 display the approximation error $\|A - A(:,J)A(I,J)^{-1}A(I,:)\|_F$ for the index sets returned by both variants of Algorithm 3. It can be seen that the approximation errors often stay close to the best rank-k approximation error $\sqrt{\sigma_{k+1}^2 + \ldots + \sigma_m^2}$ and do not exceed the upper bound (15), modulo roundoff error. However, for larger values of k, Algorithm 3 without early stopping appears to encounter stability issues; the approximation error is distorted well above the level of roundoff error. This appears to be due to the fact that $A(I,J)$ becomes almost singular. For instance, Algorithm 3 without early stopping applied to A_2 with $k = 48$ yields a matrix $A(I,J)$ with condition number $\approx 1.3 \cdot 10^{18}$. The variant with early stopping appears to lead to lower condition numbers and does not exhibit numerical instability for the matrices considered. The right plots of the figures display the ratios between the execution time of Algorithm 3 without and with early stopping, as well as the total number of index pairs that needed to be tested in Algorithm 3 with early stopping. It can be observed that early stopping dramatically accelerates the computation and is thus the preferred variant.

Figure 6: Results for matrix A_1



We also consider the $n \times n$ matrix $A = LDL^T$, where

$$L = \begin{bmatrix} 1 & & & & \\ -c & 1 & & & \\ -c & -c & 1 & & \\ \vdots & \vdots & & \ddots & \\ -c & -c & -c & \cdots & 1 \end{bmatrix}, \qquad D = \begin{bmatrix} 1 & & & & \\ & s^2 & & & \\ & & s^4 & & \\ & & & \ddots & \\ & & & & s^{2(n-1)} \end{bmatrix}$$

with $s = \sin(\theta), c = \cos(\theta)$ for some $0 < \theta < \pi$. This is known to be a challenging example for greedy cross approximation [24]: When $k = n - 1$ the greedy

Figure 7: Results for matrix A_2

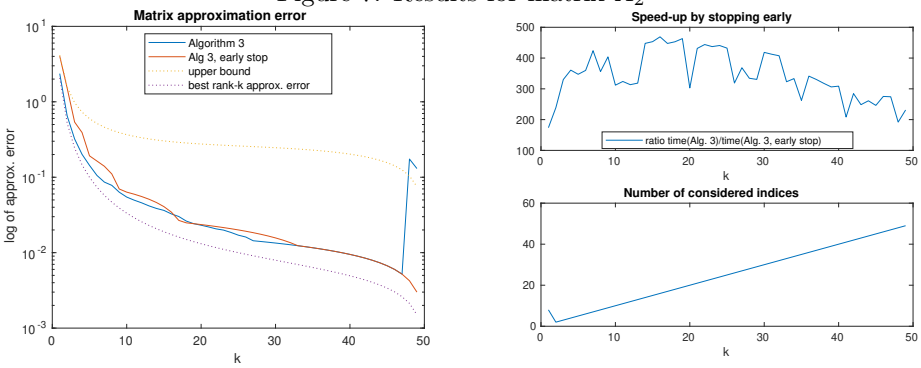
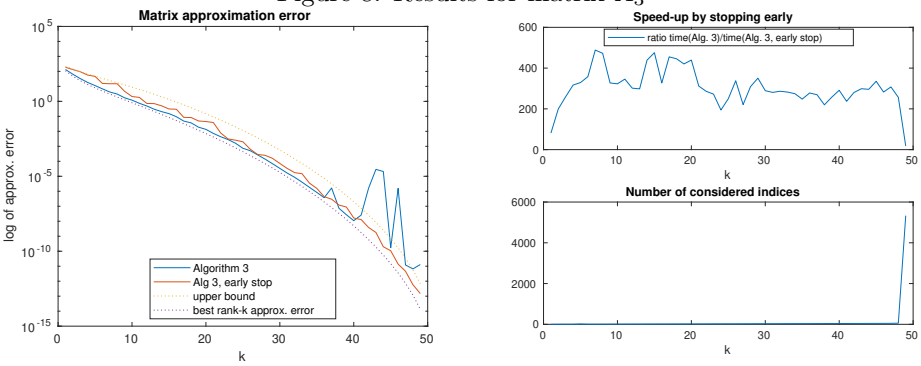


Figure 8: Results for matrix A_3



algorithm selects the leading $k \times k$ submatrix and returns an approximation error that is exponentially larger than the best approximation error. In contrast, Algorithm 3, with and without early stopping, makes the correct choice by selecting the last $n-1$ rows and columns. For instance, for $n = 6$ and $\theta = 0.1$, we obtain the error

$$\|A - A(:, 2:6)A(2:6, 2:6)^{-1}A(2:6, :)\|_F \approx 3.9 \cdot 10^{-13} < 1.8 \cdot 10^{-12} \approx \sqrt{6}\sigma_n.$$

Selecting the first 5 rows and columns results in an error of $9.8 \cdot 10^{-11}$.

Finally, we would like to point out an interesting observation concerning the preservation of structure. In joint work with Massei [12], we have shown that for a symmetric positive definite matrix A there is always a symmetric choice of indices, $J = I$, leading to a symmetric cross approximation such that the favorable error bound of Goreinov and Tyrtyshnikov [21] is attained. For cross approximation in the Frobenius norm, the situation appears to be more complicated; it is generally not true that a symmetric choice of indices achieves the error bound (15) even when A is symmetric positive definite. For instance, for $n = 3$ and $k = 1$ consider

$$A = \begin{bmatrix} 1.87 & -1.82 & -2.11 \\ -1.82 & 1.87 & 2.11 \\ -2.11 & 2.11 & 2.54 \end{bmatrix}.$$

The best symmetric choice is $I = J = \{3\}$ but this leads to an error $\approx 0.1911 > 2\sqrt{\sigma_2^2 + \sigma_3^2} \approx 0.1821$.

4 Tensor approximation

As shown, e.g., in [16, 22, 30], column subset selection can be used to approximate tensors as well. In the following, we demonstrate the use of the algorithm from Section 2 for to obtain approximations of low multilinear rank constructed from the fibers of a third-order tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$.

First, we briefly recall some basic definitions for tensors and refer to [27] for more details. Generalizing the notion of rows and columns of a matrix, the vectors obtained from $\mathcal{A}$ by fixing all indices but the μth one are called *μ-mode fibers*. The matrix $A^{(\mu)} \in \mathbb{R}^{n_\mu \times (n_1 n_2 n_3)/n_\mu}$ containing all μ-mode fibers as columns is called the *μ-mode matricization* of $\mathcal{A}$. The μ-mode product of a matrix $B \in \mathbb{R}^{m \times n_\mu}$ with $\mathcal{A}$ is denoted by $B \times_\mu \mathcal{A}$ and it is the tensor such that its μ-mode matricization is given by $B \cdot A^{(\mu)}$. We use the Frobenius norm of a tensor defined by

$$\|\mathcal{A}\|_F^2 := \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} \sum_{i_3=1}^{n_3} \mathcal{A}(i_1, i_2, i_3)^2$$

and recall that $\|\mathcal{A}\|_F = \|A^{(\mu)}\|_F$ for $\mu = 1, 2, 3$. The tuple (k_1, k_2, k_3) defined by $k_\mu = \operatorname{rank}(A^{(\mu)})$ is called the multilinear rank of $\mathcal{A}$ and we can decompose $\mathcal{A}$ as

$$\mathcal{A} = B_1 \times_1 B_2 \times_2 B_3 \times_3 \mathcal{C},$$

for coefficient matrices $B_\mu \in \mathbb{R}^{n_\mu \times k_\mu}$ for $\mu = 1, 2, 3$ and a so called *core tensor* $\mathcal{C} \in \mathbb{R}^{k_1 \times k_2 \times k_3}$. This so called *Tucker decomposition* is particularly beneficial when the multilinear rank is much smaller than the size of a tensor.

Algorithm 4 produces an approximate Tucker decomposition for a given tensor such each coefficient matrix B_μ is composed of μ-mode fibers. The following result shows that the obtained approximation error remains close to the best approximation error.

Algorithm 4 Approximation of tensors by column selection

Input: Tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, integers k_1, k_2, k_3

Output: Approximate Tucker decomposition of multilinear rank (k_1, k_2, k_3) in terms of coefficient matrices B_1, B_2, B_3 and core tensor $\mathcal{C}$

1: **for** $\mu = 1, 2, 3$ **do**
2: Compute $B_\mu = A^{(\mu)}(:, S_\mu)$ by applying Algorithm 1 to select k_μ columns from $A^{(\mu)}$
3: **end for**
4: Compute $\mathcal{C} = B_1^+ \times_1 B_2^+ \times_2 B_3^+ \times_3 \mathcal{A}$

Corollary 6. *Consider $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ and integers k_1, k_2, k_3 such that $1 \leq k_\mu \leq n_\mu$ for $\mu = 1, 2, 3$. Then the output of Algorithm 4 satisfies*

$$\|\mathcal{A} - B_1 \times_1 B_2 \times_2 B_3 \times_3 \mathcal{C}\|_F \leq \sqrt{k_1 + k_2 + k_3 + 3} \cdot \|\mathcal{A} - \mathcal{A}_{\text{best}}\|_F,$$

where $\mathcal{A}_{\text{best}}$ is the best Tucker approximation of $\mathcal{A}$ of multilinear rank at most (k_1, k_2, k_3).

Proof. The proof is similar to existing proofs on the quasi-optimality of the Higher-Order SVD [13] and related results in [16, 22, 30].

Using (1) and setting $\pi_\mu = B_\mu B_\mu^+$, the result of Algorithm 1 applied to $A^{(\mu)}$ satisfies

$$\begin{aligned}\|A^{(\mu)} - \pi_\mu(A^{(\mu)})\|_F^2 &\leq (k_\mu+1)\big(\sigma_{k_\mu+1}(A^{(\mu)})^2 + \cdots + \sigma_{n_\mu}(A^{(\mu)})^2\big)\\ &\leq (k_\mu+1)\|A^{(\mu)} - A^{(\mu)}_{\text{best}}\|_F^2 = (k_\mu+1)\|\mathcal{A} - \mathcal{A}_{\text{best}}\|_F^2,\end{aligned}$$

where the second inequality follows from the fact that $A^{(\mu)}_{\text{best}}$, the μ-mode matricization of $\mathcal{A}_{\text{best}}$, has rank at most k_μ. Using the orthogonality of the projections π_μ, we obtain

$$\begin{aligned}&\|\mathcal{A} - B_1 \times B_2 \times B_3 \times \mathcal{C}\|_F^2 = \|\mathcal{A} - \pi_1 \times \pi_2 \times \pi_3 \times \mathcal{A}\|_F^2\\ &\quad= \|\mathcal{A} - \pi_1 \times \mathcal{A}\|_F^2 + \|\pi_1 \times (\mathcal{A} - \pi_2 \times \mathcal{A})\|_F^2 + \|\pi_1 \times \pi_2 \times (\mathcal{A} - \pi_3 \times \mathcal{A})\|_F^2\\ &\quad\leq \sum_{\mu=1}^3 \|\mathcal{A} - \pi_\mu \times \mathcal{A}\|_F^2 = \sum_{\mu=1}^3 \|A^{(\mu)} - \pi_\mu(A^{(\mu)})\|_F^2 \leq \sum_{\mu=1}^3 (k_\mu+1)\|\mathcal{A} - \mathcal{A}_{\text{best}}\|_F^2\\ &\quad= (k_1+k_2+k_3+3)\|\mathcal{A} - \mathcal{A}_{\text{best}}\|_F^2,\end{aligned}$$

where the second equality follows from [35, Theorem 5.1]. □

Remark 7. *Algorithm 4 easily generalizes to tensors of arbitrary order. Given a tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times \ldots \times n_d}$ and integers $k_1, \ldots, k_d$, this generalization constructs subsets of fibers $B_1, \ldots, B_d$ and a core tensor $\mathcal{C}$ such that*

$$\|\mathcal{A} - B_1 \times_1 \ldots \times_{d-1} B_d \times_d \mathcal{C}\|_F \leq \sqrt{k_1 + \ldots + k_d + d} \cdot \|\mathcal{A} - \mathcal{A}_{\text{best}}\|_F.$$

4.1 Numerical experiments

We have implemented Algorithm 4 in Matlab and tested it on two $50 \times 50 \times 50$ tensors, given by $\mathcal{A}_1(i,j,h) = \frac{1}{i+j+h-1}$ and $\mathcal{A}_2(i,j,h) = \big(i^{10} + j^{10} + h^{10}\big)^{1/10}/50$. We choose $k_1 = k_2 = k_3 = k$ and report in Figure 9 the obtained approximation errors $\|\mathcal{A}_i - B_1 \times B_2 \times B_3 \times \mathcal{C}\|_F$ for different values of k, where B_1, B_2, B_3, $\mathcal{C}$ are returned by Algorithm 4, with and without early stopping in the column selection part. We compare with the quantity

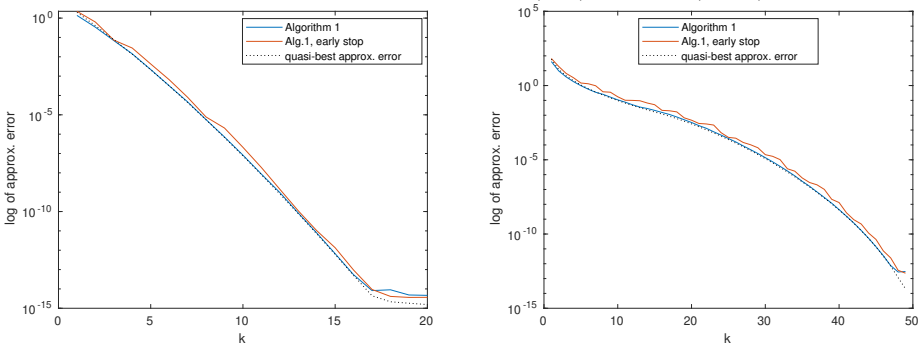
$$\Big(\sum_{\mu=1}^3 \sigma_{k_\mu+1}(A^{(\mu)})^2 + \cdots + \sigma_{n_\mu}(A^{(\mu)})^2\Big)^{1/2},$$

which provides a (tight) upper bound on the best approximation error. It can be seen that the errors obtained from Algorithm 4 remain close to this quasi-best approximation error.

5 Conclusions

In this work, we have proposed several improvements to the column selection algorithm by Deshpande and Rademacher [14]. The numerical experiments indicate that updating singular values (instead of characteristic polynomials) leads to numerical robustness, in the sense that the approximation error obtained in

Figure 9: Results for tensors $\mathcal{A}_1$ (left) and $\mathcal{A}_2$ (right).



finite precision arithmetic is not affected unduly by roundoff error. We have also developed an extension of [14] to produce cross approximations of matrices and, to the best of our knowledge, this extension constitutes the first deterministic polynomial time algorithm that yields a cross approximation with a guaranteed polynomial error bound. We have introduced a mechanism for stopping early the search for indices in column subset selection or cross approximation. Although relatively simple, this mechanism tremendously reduces the execution time for all examples tested.

A number of issues remain for future study, such as the numerical stability analysis of our algorithms. In particular, it would be desirable to study the numerical robustness of the cross approximation returned by Algorithm 3 with early stopping. Also, by combining early stopping with a more aggressive reuse of the SVD might lead to further complexity reduction, but a rigorous complexity analysis would require deeper understanding of early stopping, well beyond the limited scope of Lemma 1. Finally, we would like to stress that the algorithms presented in this work are intented for small to medium sized matrices and tensors. For large-scale data, the algorithms presented in this paper need to be combined with other, possibly heuristic dimensionality reduction techniques.

Acknowledgements. The authors thank Sergey Dolgov for helpful discussions on topics related to the work presented in this paper.

References

[1] E. Anderson, Z. Bai, C. H. Bischof, S. Blackford, J. W. Demmel, J. J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. C. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, PA, third edition, 1999.

[2] J. L. Aurentz, T. Mach, L. Robol, R. Vandebril, and D. S. Watkins. *Core-chasing algorithms for the eigenvalue problem*, volume 13 of *Fundamentals of Algorithms*. SIAM, Philadelphia, PA, 2018.

[3] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera. An 'empirical interpolation' method: application to efficient reduced-basis discretization

of partial differential equations. *C. R. Math. Acad. Sci. Paris*, 339(9):667–672, 2004.

[4] M. Bebendorf. Approximation of boundary element matrices. *Numer. Math.*, 86(4):565–589, 2000.

[5] D. S. Bernstein. *Matrix Mathematics*. Princeton University Press, Princeton, NJ, second edition, 2009.

[6] C. Boutsidis, M. W. Mahoney, and P. Drineas. An improved approximation algorithm for the column subset selection problem. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 968–977. SIAM, Philadelphia, PA, 2009.

[7] P. Bürgisser, M. Clausen, and M. A. Shokrollahi. *Algebraic complexity theory*, volume 315 of *Grundlehren der Mathematischen Wissenschaften*. Springer-Verlag, Berlin, 1997.

[8] A. Çivril and M. Magdon-Ismail. On selecting a maximum volume submatrix of a matrix and related problems. *Theoret. Comput. Sci.*, 410(47-49):4801–4811, 2009.

[9] T. F. Chan. Rank revealing QR factorizations. *Linear Algebra Appl.*, 88/89:67–82, 1987.

[10] S. Chandrasekaran and I. C. F. Ipsen. On rank-revealing factorisations. *SIAM J. Matrix Anal. Appl.*, 15(2):592–622, 1994.

[11] S. Chaturantabut and D. C. Sorensen. Nonlinear model reduction via discrete empirical interpolation. *SIAM J. Sci. Comput.*, 32(5):2737–2764, 2010.

[12] A. Cortinovis, D. Kressner, and S. Massei. On maximum volume submatrices and cross approximation for symmetric semidefinite and diagonally dominant matrices. *arXiv preprint arXiv:1902.02283*, 2019.

[13] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM J. Matrix Anal. Appl.*, 21(4):1253–1278, 2000.

[14] A. Deshpande and L. Rademacher. Efficient volume sampling for row/column subset selection. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science—FOCS 2010*, pages 329–338. IEEE Computer Soc., Los Alamitos, CA, 2010.

[15] A. Deshpande, L. Rademacher, S. Vempala, and G. Wang. Matrix approximation and projective clustering via volume sampling. *Theory Comput.*, 2:225–247, 2006.

[16] P. Drineas and M. W. Mahoney. A randomized algorithm for a tensor-based generalization of the singular value decomposition. *Linear Algebra Appl.*, 420(2-3):553–571, 2007.

[17] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error CUR matrix decompositions. *SIAM J. Matrix Anal. Appl.*, 30(2):844–881, 2008.

[18] Z. Drmač and S. Gugercin. A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions. *SIAM J. Sci. Comput.*, 38(2):A631–A648, 2016.

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

[20] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[21] 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 (Boulder, CO, 1999)*, volume 280 of *Contemp. Math.*, pages 47–51. Amer. Math. Soc., Providence, RI, 2001.

[22] S. A. Goreĭnov. Cross approximation of a multi-index array. *Dokl. Akad. Nauk*, 420(4):439–441, 2008.

[23] M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM J. Sci. Comput.*, 17(4):848–869, 1996.

[24] H. Harbrecht, M. Peters, and R. Schneider. On the low-rank approximation by the pivoted Cholesky decomposition. *Appl. Numer. Math.*, 62(4):428–440, 2012.

[25] R. A. Horn and C. R. Johnson. *Matrix analysis*. Cambridge University Press, Cambridge, second edition, 2013.

[26] O. Knill. Cauchy-Binet for pseudo-determinants. *Linear Algebra Appl.*, 459:522–547, 2014.

[27] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Rev.*, 51(3):455–500, 2009.

[28] R. Rehman and I. C. Ipsen. La budde's method for computing characteristic polynomials. *arXiv preprint arXiv:1104.3769*, 2011.

[29] R. Rehman and I. C. F. Ipsen. Computing characteristic polynomials from eigenvalues. *SIAM J. Matrix Anal. Appl.*, 32(1):90–114, 2011.

[30] A. K. Saibaba. HOID: higher order interpolatory decomposition for tensors based on Tucker representation. *SIAM J. Matrix Anal. Appl.*, 37(3):1223–1249, 2016.

[31] D. C. Sorensen and M. Embree. A DEIM induced CUR factorization. *SIAM J. Sci. Comput.*, 38(3):A1454–A1482, 2016.

[32] D. C. Sorensen and M. Embree. A DEIM induced CUR factorization. *SIAM J. Sci. Comput.*, 38(3):A1454–A1482, 2016.

[33] G. W. Stewart. Four algorithms for the efficient computation of truncated pivoted QR approximations to a sparse matrix. *Numer. Math.*, 83(2):313–323, 1999.

[34] A. Townsend and L. N. Trefethen. An extension of Chebfun to two dimensions. *SIAM J. Sci. Comput.*, 35(6):C495–C518, 2013.

[35] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen. A new truncation strategy for the higher-order singular value decomposition. *SIAM J. Sci. Comput.*, 34(2):A1027–A1052, 2012.

[36] D. P. Woodruff. Sketching as a tool for numerical linear algebra. *Found. Trends Theor. Comput. Sci.*, 10(1-2):iv+157, 2014.

[37] P. A.-C. Yoon. *Modifying two-sided orthogonal decompositions: Algorithms, implementation, and applications*. ProQuest LLC, Ann Arbor, MI, 1996. Thesis (Ph.D.)–The Pennsylvania State University.

[38] N. L. Zamarashkin and A. I. Osinsky. On the existence of a nearly optimal skeleton approximation of a matrix in the Frobenius norm. In *Doklady Mathematics*, volume 97, pages 164–166. Springer, 2018.