

# Column Based Matrix Reconstruction via Greedy Approximation of SVD<sup>†</sup>

A. Civril\* and M. Magdon-Ismail

Rensselaer Polytechnic Institute, Computer Science Department, 110 8th Street Troy, NY 12180-3590 USA

## SUMMARY

Given a matrix  $A \in \mathbb{R}^{m \times n}$  of rank  $r$ , and an integer  $k < r$ , the top  $k$  singular vectors provide the best rank- $k$  approximation to  $A$ . When the columns of  $A$  have specific meaning, it might be desirable to find “good” approximations to  $A_k$  which use a small number of columns in  $A$ . We provide a simple greedy *deterministic* algorithm for this problem, with guarantees on the performance and the number of columns chosen. Our greedy algorithm chooses  $c$  columns from  $A$  with  $c = \tilde{O}\left(\frac{k^2 \log k}{\epsilon^2} \mu^2(A)\right)$  such that

$$\|A - CC^+A\|_F \leq (1 + \epsilon) \|A - A_k\|_F,$$

where  $C$  is the matrix composed of the  $c$  columns,  $C^+$  is the pseudo-inverse of  $C$ ,  $CC^+A$  is the best reconstruction of  $A$  from  $C$ , and  $\mu(A)$  is a measure related to the *coherence* in the normalized columns of  $A$ . The running time of the algorithm is  $O(SVD(A_k) + mnc)$  where  $SVD(A_k)$  is the time to compute the first  $k$  singular vectors of  $A$ . The algorithm is quite simple and intuitive and is obtained by combining a generalization of the well known *sparse approximation problem* from information theory with an *existence* result on the possibility of sparse approximation. We provide empirical results on various specially constructed matrices comparing our algorithm with the previous deterministic approaches based on QR factorizations and a recently proposed randomized algorithm. The results indicate that in practice, the performance of our algorithm can be significantly better than the bounds suggest. Copyright © 2009 John Wiley & Sons, Ltd.

## 1. Introduction

Most data can be represented as an  $m \times n$  matrix where the columns are objects and the rows are the features associated with them. Hence, given a matrix  $A \in \mathbb{R}^{m \times n}$ , one might be interested in obtaining the “important” spectral information of  $A$  by using some compressed representation. The usual approach to this problem is to take the best rank  $k$  ( $k \ll \min\{m, n\}$ ) approximation  $A_k$ , which minimizes the error with respect to any unitarily invariant norm.  $A_k$  can be constructed from the top  $k$  singular vectors in  $O(\min\{mn^2, m^2n\})$  time. The first

---

\*Correspondence to: Rensselaer Polytechnic Institute, Computer Science Department, 110 8th Street Troy, NY 12180-3590 USA

<sup>†</sup>A preliminary version of this paper has appeared in the 19th International Symposium on Algorithms and Computation (ISAAC 2008). This extended version contains the full proofs, the implementation details and the numerical experiments.

$k$  singular vectors required to construct  $A_k$  can be computed using Lanczos methods. In statistical data analysis, one problem with this general approach is that the singular vector representation might not be suitable to make inferences about the actual underlying data, because they are generally combinations of all the columns of the raw information in  $A$ . An example of this is the microarray data where the combinations of the column vectors have no sensible interpretation [1]. Hence, it is of practical importance to represent the approximation to  $A_k$  by a small number of columns of  $A$ .

The theoretical computer science community has investigated the problem of choosing as few columns as possible to get a reconstruction of  $A$  which is arbitrarily close to  $A_k$  in the spectral and Frobenius norm. The solutions developed thus far have focused on randomized algorithms, and the number of columns chosen by these algorithms are greater than  $k$ . The numerical linear algebra community on the other hand, implicitly provides deterministic solutions for reconstructing  $A$  in the context of rank revealing QR factorizations, which primarily aim to determine the numerical rank of  $A$ . The algorithms developed in this framework usually focus on spectral norm and they choose exactly  $k$  columns.

### 1.1. Our Contributions

We give a deterministic greedy algorithm for low rank matrix reconstruction which is based on the sparse approximation of the SVD of  $A$ . We first generalize the problem of sparse approximation of [2] to one of approximating a subspace, using the columns from  $A$ . We then propose and analyze a greedy algorithm for this problem and get our main result in the special case where the subspace to be approximated is the best rank- $k$  approximation to  $A$ .

In words, our algorithm first computes the top  $k$  left singular vectors of  $A$ , and then selects columns of  $A$  in a greedy fashion so as to “fit” the space spanned by the singular vectors, appropriately scaled according to the singular values. The performance characteristics of the algorithm depend on how well the greedy algorithm approximates the optimal choice of such columns from  $A$ , and on how good the optimal columns themselves are. We combine an existence result on the quality of the optimal columns with the analysis of the greedy algorithm to arrive at the following result:

**Theorem 1.1.** *The greedy algorithm chooses a column submatrix  $C \subseteq A$  with  $c = O\left(\frac{k^2 \log k}{\epsilon^2} \mu^2(A) \ln\left(\frac{\sqrt{k} \|A_k\|_F}{\epsilon \|A - A_k\|_F}\right)\right)$  columns such that*

$$\|A - CC^+A\|_F \leq (1 + \epsilon) \|A - A_k\|_F.$$

The term  $\frac{k^2 \log k}{\epsilon^2}$  arises from an upper bound on the number of columns the optimal solution would choose (the existence result), and the remaining terms are contributed by the analysis of the greedy algorithm. The coherence parameter  $\mu(A)$ , restricts the class of matrices for which the bound is useful. Note that, in order to achieve this approximation ratio, we choose more than  $k$  columns, which is reminiscent of the results provided by the theoretical computer science community. Under a reasonable assumption on the spectral norm of  $A$ , the natural logarithm in the expression of  $c$  vanishes for  $\epsilon = O(\mu(A)\sqrt{k \log k})$ , and we have a  $1 + O(\mu(A)\sqrt{k \log k})$  approximation ratio with  $k$  columns (see Corollary 3.3). We would like to note that, the parameter  $\mu(A)$ , which will be defined in the next section (see Definition 3), is related to the smallest singular value of a certain sub-matrix of  $A$ , and from this point of view, our approximation factor is similar to a set of greedy algorithms classified by Chandrasekaran and

Ipsen [3]. These are essentially QR factorization algorithms whose performance factor depends on  $n$ , the number of columns of  $A$  and the spectral norm of an inverse of a  $k \times k$  matrix revealed by the QR decomposition. While it is possible to bound this number in terms of an exponential expression depending on  $k$ , we cannot give any guarantee on  $\mu(A)$ . But our factor does not depend on  $n$ .

The term  $\mu(A)$  arises from the sparse approximation problem. We believe that a result without the parameter  $\mu(A)$  should be possible, however we have not been able to construct one. Improving either the upper bound on the optimal reconstruction of the singular vectors, or improving the analysis of the greedy algorithm would yield a tighter result.

### 1.2. Comparison to Related Work

With the advent of massive data sets, much work in theoretical computer science has been spent on finding algorithms for matrix reconstruction by considering a careful choice of a subset of the columns of the data matrix. The seminal paper by Frieze, Kannan and Vempala [4] gives a randomized algorithm that chooses a subset of columns  $C \in \mathbb{R}^{m \times c}$  of  $A$  such that  $\|A - \Pi_C A\|_F \leq \|A - A_k\|_F + \epsilon \|A\|_F$ , where  $\Pi_C$  is a projection matrix obtained by the SVD of  $C$  and  $c = \text{poly}(k, 1/\epsilon, 1/\delta)$ , where  $\delta$  is the failure probability of the algorithm. Subsequent work [5, 6, 7] introduced several improvements on the dependence of  $c$  on  $k, 1/\epsilon$  and  $1/\delta$  also extending the analysis to the spectral norm. Recently, the effort has been towards eliminating the additive term in the inequality thereby yielding a relative approximation in the form  $\|A - \Pi_C A\|_F \leq (1 + \epsilon)\|A - A_k\|_F$ . Along these lines, Deshpande et al. [8] first shows the existence of such approximations introducing a sampling technique related to the volume of the simplex defined by the column subsets of size  $k$ , without giving a polynomial time algorithm. Specifically, they show that there exists  $k$  columns with which one can get a  $\sqrt{k+1}$  relative error approximation in Frobenius norm, which is tight. Later, Deshpande and Vempala [9] and Drineas et al. [10] provided algorithms with different sampling schemes attaining the  $(1 + \epsilon)$  approximation where the number of columns is a function of  $k$  and  $\epsilon$ . Other recent approaches for the problem we consider includes random projections [11], and sampling which exploits geometric properties of high dimensional spaces [12]. [13] also considers the subspace approximation problem in general  $l_p$  norms. All of these algorithms exploit the power of randomization and they introduce a trade-off between the number of columns chosen, the error parameter and the failure probability of the algorithm. The proof techniques presented in these papers break when the random sampling approach is sacrificed and a deterministic column selection procedure is used.

When it comes to deterministic reconstruction, no  $(1 + \epsilon)$  approximation algorithms are known. The linear algebra community has developed deterministic algorithms in the framework of *rank revealing QR (RRQR) factorizations* [14] which yield some approximation guarantees in spectral norm. Given a matrix  $A \in \mathbb{R}^{n \times n}$ , consider the QR factorization of the form

$$A\Pi = Q \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix}, \tag{1}$$

where  $R_{11} \in \mathbb{R}^{k \times k}$  and  $\Pi \in \mathbb{R}^{n \times n}$  is a permutation matrix. By the interlacing property of singular values (see [15]),  $\sigma_k(R_{11}) \leq \sigma_k(A)$  and  $\sigma_1(R_{22}) \geq \sigma_{k+1}(A)$ . If the numerical rank of  $A$  is  $k$ , i.e.  $\sigma_k(A) \gg \sigma_{k+1}(A)$ , then one would like to find a permutation  $\Pi$  for which  $\sigma_k(R_{11})$  is sufficiently large and  $\sigma_1(R_{22})$  is sufficiently small. A QR factorization is said to be a rank

revealing QR (RRQR) factorization if  $\sigma_k(R_{11}) \geq \sigma_k(A)/p(k, n)$  and  $\sigma_1(R_{22}) \leq \sigma_{k+1}(A)p(k, n)$ , where  $p(k, n)$  is a low degree polynomial in  $k$  and  $n$ .

Much research on finding RRQR factorizations has yielded improved results for  $p(k, n)$  [3, 16, 14, 17, 18, 19, 20]. These algorithms make use of the *local maximum volume* concept and are generally complicated. Tight bounds for  $p(k, n)$  can be used to give deterministic low rank matrix reconstruction with respect to the spectral norm, via the following simple fact.

**Theorem 1.2.** *Let  $\Pi_k$  be the matrix of first  $k$  columns of  $\Pi$  in (1). Then,*

$$\|A - (A\Pi_k)(A\Pi_k)^+ A\|_2 \leq p(k, n)\|A - A_k\|_2.$$

The best  $p(k, n)$  was proposed by Gu and Eisenstat [18]. The authors show that there exists a permutation  $\Pi$  for which  $p(k, n) = \sqrt{1 + k(n - k)}$ . It is not known whether such a permutation can be computed in polynomial time. Instead, algorithms with  $p(k, n) = \sqrt{1 + f^2 k(n - k)}$  were given which run in  $O((m + n \log_f n)n^2)$  time for  $f > 1$  [18]. Hence, for constant  $f$ , the approximation ratio depends on  $n$  and the running time is  $O(mn^2 + n^3 \log n)$ . Note that, these algorithms consider choosing exactly  $k$  columns and the results are not directly comparable to ours as they provide bounds on the spectral norm. It is not clear whether these algorithmic results can be extended to give non-trivial bounds in Frobenius norm or to choose more than  $k$  columns so as to yield  $(1 + \epsilon)$  approximation.

Very recently, Boutsidis et al. [21] introduced a hybrid algorithm for the problem of choosing exactly  $k$  columns from a matrix  $A$  to approximate  $A_k$ , combining the random sampling schemes and the deterministic column pivoting strategies exploited by QR algorithms. Their algorithm provides a performance guarantee of  $p(k, n) = O\left(k^{\frac{3}{4}} \log^{\frac{1}{2}}(k)(n - k)^{\frac{1}{4}}\right)$  for spectral norm, and  $p(k, n) = O(k\sqrt{\log k})$ , with high probability.

### 1.3. Notation and Preliminaries

From now on  $A \in \mathbb{R}^{m \times n}$  is the matrix we wish to reconstruct.  $A_{(i)}$  denotes the  $i^{\text{th}}$  row of  $A$  for  $1 \leq i \leq m$ , and  $A^{(j)}$ , the  $j^{\text{th}}$  column of  $A$  for  $1 \leq j \leq n$ .  $A_{ij}$  is the element at  $i^{\text{th}}$  row and the  $j^{\text{th}}$  column. Typically, we use  $C$  to denote a subset of columns of  $A$ , written  $C \subset A$ , i.e.  $C$  is a column sub-matrix of  $A$ .  $\text{span}(C)$  denotes the subspace spanned by the column vectors in  $C$ . The Singular Value Decomposition of  $A \in \mathbb{R}^{m \times n}$  of rank  $r$  is denoted by  $A = U\Sigma V^T$  where  $U \in \mathbb{R}^{m \times m}$  is the matrix of left singular vectors,  $\Sigma \in \mathbb{R}^{m \times r}$  is the diagonal matrix containing the singular values of  $A$  in descending order, i.e.  $\Sigma = \text{diag}(\sigma_1, \dots, \sigma_r, 0, \dots, 0)$  where  $\sigma_1 \leq \sigma_2 \leq \dots \leq \sigma_r > 0$  are the singular values of  $A$ .  $V \in \mathbb{R}^{n \times n}$  is the matrix of right singular vectors. The “best” rank  $k$  approximation to  $A$  is  $A_k = U_k \Sigma_k V_k^T$  where  $U_k$  and  $V_k$  are the first  $k$  columns of the corresponding matrices in the full SVD of  $A$ , and  $\Sigma_k$  is the  $k \times k$  diagonal matrix of the first  $k$  singular values. The pseudo-inverse of  $A$  is denoted by  $A^+ = V\Sigma^+ U^T$ , where  $\Sigma^+ = \text{diag}\left(\frac{1}{\sigma_1}, \dots, \frac{1}{\sigma_r}, 0, \dots, 0\right)$ . The Frobenius norm of  $A$  is  $\|A\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n A_{ij}^2}$ , and the spectral norm of  $A$  is  $\|A\|_2 = \sigma_1(A)$ . We also define the maximum column norm of a matrix  $A$ ,  $\|A\|_{\text{col}} = \max_{i=1}^n \{\|A^{(i)}\|_2\}$ .  $S^\perp$  is the space orthogonal to the space spanned by the vectors in  $S$ .

## 2. Generalized Sparse Approximation

Instead of seeking sparse approximation to a single vector [2], we propose the following generalization: given matrices  $A \in \mathbb{R}^{m \times n}$ , a set of vectors  $B \in \mathbb{R}^{m \times k}$ , and  $\epsilon > 0$ , find a matrix  $X \in \mathbb{R}^{n \times k}$  satisfying

$$\|AX - B\|_F \leq \epsilon, \tag{2}$$

such that  $\sum_{i=1}^n \nu_i(X)$  is minimum over all possible choices of  $X$ , where  $\nu_i(X) = 1$  if the row  $X_{(i)}$  contains non-zero entries,  $\nu_i(X) = 0$  if  $X_{(i)} = \vec{0}$ . Intuitively, the problem asks for a minimum number of column vectors of  $A$  whose span is “close” to the span of  $B$ .

### 2.1. The Algorithm

A greedy strategy for solving this problem is to choose the column  $v$  from  $A$  at each iteration, for which  $\|B^T v\|_2$  is maximum, and project the column vectors of  $B$  and the other column vectors of  $A$  onto the space orthogonal to the chosen column. The algorithm proceeds greedily on these residual matrices until the norm of the residual  $B$  drops below the required threshold  $\epsilon$ . Naturally, if the error  $\epsilon$  cannot be attained, the algorithm will fail after selecting a maximal independent set of columns.

---

**Algorithm 1** A greedy algorithm for Generalized Sparse Approximation

---

```

1: procedure GREEDY( $A, B, \epsilon$ )
2:   normalize each column of  $A$  to have norm 1.
3:    $l \leftarrow 0, \Lambda \leftarrow \emptyset, A_0 \leftarrow A, B_0 \leftarrow B$ .
4:   while  $\|B_l\|_F > \epsilon$  do
5:     choose  $i \in \{1, \dots, n\} - \Lambda$  such that  $\|B_l^T A_l^{(i)}\|_2$  is maximum
6:      $B_{l+1}^{(j)} \leftarrow B_l^{(j)} - \left( B_l^{(j)T} A_l^{(i)} \right) A_l^{(i)}$  for  $i = 1, \dots, k$ , i.e. project  $B_l^{(j)}$ 's onto  $\{A_l^{(i)}\}^\perp$ .
7:      $\Lambda \leftarrow \Lambda \cup \{i\}$ .
8:      $A_{l+1}^{(j)} \leftarrow A_l^{(j)} - \left( A_l^{(j)T} A_l^{(i)} \right) A_l^{(i)}$  for  $j \in \{1, \dots, n\} - \Lambda$ , i.e. project  $A_l^{(j)}$ 's onto  $\{A_l^{(i)}\}^\perp$ .
9:     normalize  $A_{l+1}^{(j)}$  for  $j \in \{1, \dots, n\} - \Lambda$ .
10:     $l \leftarrow l + 1$ .
11:  end while
12:  return  $C = \Lambda(A)$ , the selected columns.
13: end procedure

```

---

### 2.2. Implementation Details and Running Time Analysis

Line 1,7 and 8 of Greedy takes  $O(mn)$  time. Line 5 takes  $O(mk)$  time since  $B \in \mathbb{R}^{m \times k}$ . The computationally intensive part of the algorithm in the while loop is the 4th step, which takes  $O(mnk)$  time with a naive implementation, since there are  $n$  matrix-vector multiplications of cost  $O(mk)$ . This makes a total of  $O(mnkc)$  running time complexity which amounts to

$O(mnk^2)$  in case  $k$  vectors are chosen. We make note of a simple observation which is akin to the pivoted QR algorithms and is called a norm update: Instead of performing matrix-vector multiplications at each iteration, we remember the dot products of the chosen column with the columns of  $B$  and the other columns in  $A$ . We also introduce a matrix  $D \in \mathbb{R}^{k \times n}$ , where  $(D_l)_{ij}$  denotes the dot product of the  $i^{\text{th}}$  column of  $B$  and the  $j^{\text{th}}$  column of  $A$  in the  $l^{\text{th}}$  iteration, i.e.  $(D_l)_{ij} = B_l^{(i)T} A_l^{(j)}$ . At the end of each iteration, we update  $D_l$  to get  $D_{l+1}$  where the update of each entry requires constant time. Hence, the 4th step takes  $O(nk)$  time complexity for each iteration. Overall, the running time of the algorithm is  $O((2mn + mk + nk)c) = O(mnc)$ .

The norm update is as follows: Suppose a column vector  $v$  is chosen at iteration  $l$ . Noting that  $v^T v = 1$ ,  $D_{l+1}$  satisfies

$$\begin{aligned} (D_{l+1})_{ij} &= B_{l+1}^{(i)T} A_{l+1}^{(j)} = \frac{\left( B_l^{(i)} - \left( B_l^{(i)T} v \right) v \right)^T \left( A_l^{(j)} - \left( A_l^{(j)T} v \right) v \right)}{\left\| A_l^{(j)} - \left( A_l^{(j)T} v \right) v \right\|_2} \\ &= \frac{B_l^{(i)T} A_l^{(j)} + \left( B_l^{(i)T} v \right) \left( A_l^{(j)T} v \right) (v^T v - 2)}{\left\| A_l^{(j)} - \left( A_l^{(j)T} v \right) v \right\|_2} \\ &= \frac{(D_l)_{ij} - \left( B_l^{(i)T} v \right) \left( A_l^{(j)T} v \right)}{\left\| A_l^{(j)} - \left( A_l^{(j)T} v \right) v \right\|_2}. \end{aligned}$$

The update per entry can be performed in constant time given the other values in the last expression, which are already computed.

### 2.3. Performance Analysis

Our analysis yields an approximation factor which includes a term related to the smallest singular value of a certain sub-matrix, which is relevant to the analysis. We begin with the following definition, which provides a general upper bound for the spectral norm of the pseudo-inverse of any sub-matrix of a matrix. We would like to note that similar definitions have appeared in [22] while analyzing algorithms for the sparse approximation problem.

**Definition 2.1.** [*Coherence*] Given a matrix  $A \in \mathbb{R}^{m \times n}$  of rank  $r$ , let  $\mathbf{A}$  be the matrix  $A$  with normalized columns. Then,  $\mu(A)$  is the maximum of the inverses of the least singular value of  $m \times r$  full-rank sub-matrices of  $\mathbf{A}$ . Namely,

$$\mu(A) = \max_{\substack{\mathbf{C} \subseteq \mathbf{A} \\ \mathbf{C} \in \mathbb{R}^{m \times r} \\ \text{rank}(\mathbf{C})=r}} \frac{1}{\sigma_r(\mathbf{C})}. \quad (3)$$

**Remark 2.2.**  $1 \leq \mu(A) < \infty$ . Small values of  $\mu(A)$  indicate a matrix with near orthonormal columns.

**Theorem 2.3.** *The number of columns chosen by Greedy is at most*

$$O\left(\text{Opt}(\epsilon/2)\mu^2(A)\ln\left(\frac{\|B\|_F}{\epsilon}\right)\right),$$

where  $\text{Opt}(\epsilon/2)$  is the optimal number of columns at error  $\epsilon/2$ .

We will establish Theorem 2.3 through a sequence of lemmas. The proof follows similar reasoning to the proof in [2]. Let  $t$  be the total number of iterations of Greedy. At the beginning of the  $l^{\text{th}}$  iteration of the algorithm, for  $0 \leq l < t$ , let  $U_l$  be an optimal solution to the generalized sparse approximation problem with error parameter  $\epsilon/2$ , i.e.  $U_l$  minimizes  $\sum_{i=1}^n \nu_i(X)$  over  $X \in \mathbb{R}^{n \times k}$  such that  $\|A_l U_l - B_l\|_F \leq \epsilon/2$ , where  $\nu_i(X) = 1$  if the row  $X_{(i)}$  contains non-zero entries,  $\nu_i(X) = 0$  if  $X_{(i)} = \vec{0}$ . Let  $N_l = \sum_{i=1}^n \nu_i(U_l)$  and  $Q_l = A_l U_l$ . Define

$$\lambda = 4 \max_{0 \leq l < t} \frac{N_l \|U_l\|_F^2}{\|B_l\|_F^2}. \quad (4)$$

Assuming that the Greedy has not terminated, the following lemma states that the next step makes significant progress.

**Lemma 2.4.** *For the  $l^{\text{th}}$  iteration of Greedy,  $\|B_l^T A_l\|_{\text{col}} \geq \frac{\|B_l\|_F^2}{2\sqrt{N_l}\|U_l\|_F}$ .*

*Proof.* Let  $E \in \mathbb{R}^{m \times k}$  be a generic error matrix such that  $\|E\|_F \leq \epsilon/2$ , and Let  $\|E^{(j)}\|_2 = \epsilon_j/2$  for  $i = 1, \dots, k$ . Hence,  $\sum_{i=1}^k \epsilon_j^2 \leq \epsilon^2$ . Now, we can write  $B_l^{(j)} = \left(\sum_{i=1}^n A_l^{(i)} U_{lij}\right) + E^{(j)}$  for  $j = 1, \dots, k$ . Then,

$$\|B_l\|_F^2 = \sum_{j=1}^k B_l^{(j)T} B_l^{(j)} = \sum_{j=1}^k \sum_{i=1}^n U_{lij} B_l^{(j)T} A_l^{(i)} + \sum_{j=1}^k B_l^{(j)T} E^{(j)}. \quad (5)$$

We will first bound the double summation in the above expression.

$$\begin{aligned} \sum_{j=1}^k \sum_{i=1}^n U_{lij} B_l^{(j)T} A_l^{(i)} &\leq \sum_{i=1}^n \left( \left( \sum_{j=1}^k U_{lij}^2 \right)^{1/2} \left( \sum_{j=1}^k \left( B_l^{(j)T} A_l^{(i)} \right)^2 \right)^{1/2} \right) \\ &\leq \max_{1 \leq i \leq n} \left\{ \left( \sum_{j=1}^k \left( B_l^{(j)T} A_l^{(i)} \right)^2 \right)^{1/2} \right\} \sum_{i=1}^n \left( \sum_{j=1}^k U_{lij}^2 \right)^{1/2} \\ &\leq \|B_l^T A_l\|_{\text{col}} \sqrt{N_l} \|U_l\|_F. \end{aligned}$$

The first line is due to Cauchy-Schwartz inequality. The last inequality bounds the double summation in the second line as follows. Define  $n$  dimensional vectors  $a$  and  $b$  such that  $a_i = \left(\sum_{j=1}^k U_{lij}^2\right)^{1/2}$  and  $b_i = 1$  if there exists a non-zero entry in the  $i^{\text{th}}$  row of  $U_l$ ,  $b_i = 0$  if all the elements in the  $i^{\text{th}}$  row of  $U_l$  are zero, for  $i = 1, \dots, n$ . Then, applying Cauchy-Schwartz inequality to  $a$  and  $b$ , we obtain  $\sum_{i=1}^n \left(\sum_{j=1}^k U_{lij}^2\right)^{1/2} = \sum_{i=1}^n a_i b_i \leq$

$(\sum_{i=1}^n a_i^2)^{1/2} (\sum_{i=1}^n b_i^2)^{1/2}$ . Since  $\sum_{i=1}^n a_i^2 = \sum_{i=1}^n \sum_{j=1}^k U_{lij}^2 = \|U_l\|_F^2$ , and  $\sum_{i=1}^n b_i^2 = N_l$ , we have that  $\sum_{i=1}^n \left(\sum_{j=1}^k U_{lij}^2\right)^{\frac{1}{2}} \leq \sqrt{N_l} \|U_l\|_F$ .

We will now bound the second term in (5).

$$\begin{aligned}
\sum_{j=1}^k B_l^{(j)T} E^{(j)} &\leq \sum_{j=1}^k \|B_l^{(j)T}\|_2 \|E^{(j)}\|_2 \quad (\text{Cauchy - Schwartz}) \\
&= \frac{1}{2} \sum_{j=1}^k \epsilon_j \|B_l^{(j)T}\|_2 \\
&\leq \frac{1}{2} \left(\sum_{j=1}^k \epsilon_j^2\right)^{1/2} \left(\sum_{j=1}^k \|B_l^{(j)T}\|_2^2\right)^{1/2} \quad (\text{Cauchy - Schwartz}) \\
&\leq \frac{1}{2} \epsilon \|B_l\|_F \\
&\leq \frac{1}{2} \|B_l\|_F^2,
\end{aligned}$$

where the last inequality is due to the fact that  $\|B_l\|_F > \epsilon$ , i.e. the algorithm is still running.

Combining these bounds in (5), we have  $\|B_l\|_F^2 \leq \|B_l^T A_l\|_{col} \sqrt{N_l} \|U_l\|_F + 1/2 \|B_l\|_F^2$ , which gives  $\|B_l\|_F^2 \leq 2 \|B_l^T A_l\|_{col} \sqrt{N_l} \|U_l\|_F$ . The lemma then immediately follows. ■

Thus, there exists a column in the residual  $A_l$  which will reduce the residual  $B_l$  significantly, because  $B_l$  has a large projection onto this column. Therefore, since every step of Greedy makes significant progress, there cannot be too many steps, which is the content of the next lemma.

**Lemma 2.5.**  $t \leq \left\lceil 2\lambda \ln \left(\frac{\|B\|_F}{\epsilon}\right) \right\rceil$ , where  $t$  is the number of Greedy iterations.

*Proof.* Let  $i$  be the index of the chosen column at step  $l$  and let  $j$  be a column index of  $B$ . Then, by the execution of the algorithm,  $B_{l+1}^{(j)} = B_l^{(j)} - \left(B_l^{(j)T} A_l^{(i)}\right) A_l^{(i)}$ . Since  $B_{l+1}^{(j)}$  is orthogonal to  $A_l^{(i)}$  and  $\|A_l^{(i)}\|_2 = 1$ , we can write  $\|B_{l+1}^{(j)}\|_2^2 = \|B_l^{(j)}\|_2^2 - |B_l^{(j)T} A_l^{(i)}|^2$ . Summing over all column indices of  $B_{l+1}$ , we obtain

$$\begin{aligned}
\|B_{l+1}\|_F^2 &= \sum_{j=1}^k \|B_{l+1}^{(j)}\|_2^2 = \sum_{j=1}^k \|B_l^{(j)}\|_2^2 - \sum_{j=1}^k |B_l^{(j)T} A_l^{(i)}|^2 \\
&= \|B_l\|_F^2 - \|B_l^T A_l^{(i)}\|_2^2 \\
&= \|B_l\|_F^2 - \|B_l^T A_l\|_{col}^2 \\
&\leq \|B_l\|_F^2 - \frac{\|B_l\|_F^4}{4N_l \|U_l\|_F^2} \quad (\text{Lemma 2.4}) \\
&= \|B_l\|_F^2 \left(1 - \frac{1}{\lambda}\right) \quad (\text{Equation (4)}),
\end{aligned}$$

where the third line follows since the algorithm chooses  $i$  to maximize  $\|B_l^T A_l^{(i)}\|_2$ . Hence,  $\|B_l\|_F^2 \leq (1 - 1/\lambda)\|B_0\|_F^2$ . Since the algorithm stops when  $\|B_t\|_F^2 \leq \epsilon^2$ , it suffices for  $t$  to satisfy  $(1 - 1/\lambda)^t \|B_0\|_F^2 \leq \epsilon^2$ . Rearranging, and taking logarithms we obtain  $t \ln(1 - 1/\lambda) \leq \ln(\epsilon^2/\|B_0\|_F^2)$ . Since  $\ln(1 - 1/\lambda) \leq -1/\lambda$ , we get that  $t \geq \lambda \ln(\|B\|_F^2/\epsilon^2) = 2\lambda \ln(\|B\|_F/\epsilon)$  iterations are enough for Greedy to terminate. ■

What remains is to bound  $\lambda$ . First, we will bound  $\|U_l\|_F$  in terms of  $\|B_l\|_F$  both of which appear in the expression for  $\lambda$ . Let  $\pi_l = \{i | U_{l(i)} \neq \vec{0}\}$  be the indices of rows of  $U_l$  which are not all zero. Recall that these indices denote which columns are chosen by the optimal solution for  $A_l$ . Let  $\tau_l = \{i_1, i_2, \dots, i_l\}$  be the indices of the first  $l$  columns picked by the algorithm. Given an index set  $\gamma$ , let the set of column vectors  $\{A^{(i)} | i \in \gamma\}$  be denoted by  $\gamma(A)$ .

**Lemma 2.6.**  $\pi_l(A) \cup \tau_l(A)$  is a linearly independent set for all  $l \geq 0$ .

*Proof.* Note that for  $l = 0$ , we only have  $\sigma_0(A)$  and by the definition of the optimality of  $U_0$ , this set should be linearly independent. For  $l \geq 1$ , we will argue by contradiction. Assume that the given set,  $\pi_l(A) \cup \tau_l(A)$  is not a linearly independent set. Hence, some linear combination of some vectors from the set sum to 0. Since, by the execution of the algorithm,  $\tau_l(A)$  is a linearly independent set, at least one of these vectors should be from  $\pi_l(A)$ , and this vector  $u$  can be written as a linear combination of some other vectors in  $\pi_l(A) \cup \tau_l(A)$ . To this end, recall that  $\pi_l$  denotes the indices of columns of  $A_l$  chosen by the optimal solution  $U_l$ , and  $\pi_l(A)$  is the set of columns of  $A$  with these indices. Consider a column vector  $v$  in  $\pi_l(A)$ . According to the algorithm, at the end of the  $l^{\text{th}}$  iteration, the residual vector  $v_l$  (which is in  $\pi_l(A_l)$ ) is precisely the projection of  $v$  onto the space orthogonal to the vectors chosen by the algorithm, namely  $\tau_l(A)$ . Since this is the case for all possible  $v$ 's, we have that  $\pi_l(A_l)$  is the projection of  $\pi_l(A)$  onto the space orthogonal to  $\tau_l(A)$ . Hence, according to our last assumption,  $u_l$  which is the projection of  $u$  onto the space orthogonal to  $\tau_l(A)$  can be expressed as a linear combination of some other vectors in  $\pi_l(A_l)$  since no vector from  $\tau_l(A)$  can contribute in the expansion of  $u_l$ . This contradicts the optimality of  $U_l$ , i.e. that the number of columns it “selects” from  $A_l$  is the fewest among all possible choices. ■

**Lemma 2.7.** For  $0 \leq l < t$ ,  $\|U_l\|_F \leq \frac{3}{2}\mu(A)\|B_l\|_F$ .

*Proof.* Consider the column indices  $\{i_1, i_2, \dots, i_l\}$  of the first  $l$  vectors chosen by the algorithm. Specifically, let  $\tau_l(A_l) = \{A_l^{(i_1)}, A_l^{(i_2)}, \dots, A_l^{(i_l)}\}$  be the columns in  $A_l$  chosen by the algorithm in the order selected. Note that these vectors are orthogonal due to the algorithm. At the end of the  $l^{\text{th}}$  iteration of the algorithm, for  $i \in \pi_l$ , we can write

$$A_l^{(i)} = \frac{A_{l-1}^{(i)} - v_l^{(i)}}{\sqrt{1 - \|v_l^{(i)}\|_2^2}}, \quad (6)$$

where  $v_l^{(i)}$  is in the span of  $A_l^{(i_1)}$ . Similarly, we can express  $A_{l-1}^{(i)}$  in terms of  $A_{l-2}^{(i)}$ , i.e.

$$A_{l-1}^{(i)} = \frac{A_{l-2}^{(i)} - v_{l-1}^{(i)}}{\sqrt{1 - \|v_{l-1}^{(i)}\|_2^2}},$$

where  $v_{l-1}^{(i)}$  is in the span of  $A_l^{(i_{l-1})}$ . Note that, since the vectors in  $\tau_l(A_l)$  are orthogonal, we have  $\|v_l^{(i)} + v_{l-1}^{(i)}\|_2^2 = \|v_l^{(i)}\|_2^2 + \|v_{l-1}^{(i)}\|_2^2$ . Using this, we can recursively express  $A_l^{(i)}$  in (6) as

$$A_l^{(i)} = \frac{A^{(i)} - v^{(i)}}{\sqrt{1 - \|v^{(i)}\|_2^2}}, \quad (7)$$

for some  $v^{(i)} \in \text{span}(\tau_l(A))$ . (Note that  $\text{span}(\tau_l(A_l)) = \text{span}(\tau_l(A_0)) = \text{span}(\tau_l(A))$  and the columns of  $A$  are normalized). Thus, noting that  $Q_l^{(j)} = \sum_{i \in \pi_l} A_l^{(i)} U_{lij}$ , and  $v^{(i)}$  can be expressed as a linear combination of the column vectors of  $\tau_l(A)$ , we have

$$Q_l^{(j)} = \sum_{i \in \pi_l} U_{lij} \frac{A^{(i)} - v^{(i)}}{\sqrt{1 - \|v^{(i)}\|_2^2}} = \sum_{i \in \pi_l} \frac{U_{lij}}{\sqrt{1 - \|v^{(i)}\|_2^2}} A^{(i)} + \sum_{i \in \tau_l} \delta_i A^{(i)}, \quad (8)$$

where  $\delta_i$ 's are appropriate coefficients in the expansion of  $v^{(i)}$ . Now, let  $S_l$  be the matrix of the columns from  $\pi_l(A) \cup \tau_l(A)$ . Note that,  $S_l$  is a column sub-matrix of  $A$  which has full rank by Lemma 2.6. Since  $S_l$  is a linearly independent set,  $Q_l$  has a unique expansion in the basis  $S_l$  given by  $W_l = S_l^+ Q_l$ . Specifically, for  $i \in \pi_l$ ,  $W_{lij} = U_{lij} / \sqrt{1 - \|v^{(i)}\|_2^2}$ , and for  $i \in \tau_l$ ,  $W_{lij} = \delta_i$ . Since  $\sqrt{1 - \|v^{(i)}\|_2^2} < 1$ ,  $|U_{lij}| \leq |W_{lij}|$  for  $i \in \pi_l$ . For  $i \in \tau_l$ , we have  $U_{lij} = 0$  and hence trivially  $|U_{lij}| \leq |W_{lij}|$ . Applying this inequality to the  $j^{\text{th}}$  column of  $U_l$ , we obtain  $\|U_l^{(j)}\|_2 \leq \|W_l^{(j)}\|_2 \leq \|S_l^+\|_2 \|Q_l^{(j)}\|_2$ . The last inequality is due to sub-multiplicativity of the spectral norm. Noting that  $Q_l^{(j)} = B_l^{(j)} + E^{(j)}$ , where  $E$  is a generic error matrix with  $\|E\|_F \leq \epsilon/2$ , we obtain

$$\begin{aligned} \|U_l\|_F^2 &= \sum_{j=1}^k \|U_l^{(j)}\|_2^2 \\ &\leq \|S_l^+\|_2^2 \sum_{j=1}^k \|Q_l^{(j)}\|_2^2 \\ &\leq \|S_l^+\|_2^2 \sum_{j=1}^k \left( \|B_l^{(j)} + E^{(j)}\|_2^2 \right) \\ &\leq \|S_l^+\|_2^2 \sum_{j=1}^k \left( \|B_l^{(j)}\|_2 + \|E^{(j)}\|_2 \right)^2, \end{aligned}$$

where the last step is due to the triangle inequality. We continue by expanding the last expression and note that  $\|E\|_F \leq \epsilon/2 = \sum_{j=1}^k \|E^{(j)}\|_2^2 \leq \epsilon^2/4$ :

$$\begin{aligned}
 \|U_l\|_F^2 &\leq \|S_l^+\|_2^2 \sum_{j=1}^k \left( \|B_l^{(j)}\|_2 + \|E^{(j)}\|_2 \right)^2 \\
 &= \|S_l^+\|_2^2 \left( \sum_{j=1}^k \|B_l^{(j)}\|_2^2 + \sum_{j=1}^k \|E^{(j)}\|_2^2 + 2 \sum_{j=1}^k \|B_l^{(j)}\|_2 \|E^{(j)}\|_2 \right) \\
 &\leq \|S_l^+\|_2^2 \left( \|B_l\|_F^2 + \frac{\epsilon^2}{4} + 2 \sum_{j=1}^k \|B_l^{(j)}\|_2 \|E^{(j)}\|_2 \right) \\
 &\leq \|S_l^+\|_2^2 \left( \frac{5}{4} \|B_l\|_F^2 + 2 \sum_{j=1}^k \|B_l^{(j)}\|_2 \|E^{(j)}\|_2 \right) \quad (\|B_l\|_F > \epsilon).
 \end{aligned}$$

Applying the Cauchy-Schwartz inequality to the second term in the parentheses, we obtain

$$\begin{aligned}
 \|U_l\|_F^2 &\leq \|S_l^+\|_2^2 \left( \frac{5}{4} \|B_l\|_F^2 + 2 \left( \sum_{j=1}^k \|B_l^{(j)}\|_2^2 \right)^{1/2} \left( \sum_{j=1}^k \|E^{(j)}\|_2^2 \right)^{1/2} \right) \\
 &= \|S_l^+\|_2^2 \left( \frac{5}{4} \|B_l\|_F^2 + 2 \|B_l\|_F \|E\|_F \right) \\
 &\leq \|S_l^+\|_2^2 \left( \frac{5}{4} \|B_l\|_F^2 + \epsilon \|B_l\|_F \right) \quad (\|E\|_F \leq \epsilon/2) \\
 &\leq \|S_l^+\|_2^2 \left( \frac{5}{4} \|B_l\|_F^2 + \|B_l\|_F^2 \right) \quad (\|B_l\|_F > \epsilon) \\
 &= \frac{9}{4} \|S_l^+\|_2^2 \|B_l\|_F^2.
 \end{aligned}$$

Hence, we have  $\|U_l\|_F \leq \frac{3}{2} \|S_l^+\|_2 \|B_l\|_F$ . Now, note that the rank of  $S_l$  is less than or equal to  $r$ , the rank of  $A$ .  $S_l$  can be obtained by deleting columns of a full-rank sub-matrix  $Z$  of  $A$ , which has exactly  $r$  columns.  $\|S_l^+\|_2$ , which is the inverse of the least singular value of  $S_l$  is smaller than that of such a matrix  $Z$  (see [15]). Then, by the definition of  $\mu(A)$ , we clearly have  $\|S_l^+\|_2 \leq \|Z^+\|_2 \leq \mu(A)$  and the lemma follows. ■

We can now prove the main theorem.

**Proof of Theorem 2.3:** First, we note that the number of non-zero rows in the optimal solution is non-increasing as the algorithm proceeds, that is  $N_l \geq N_{l+1}$  for  $l > 0$ , which follows from an argument identical to the proof of Lemma 3 in [2]. Since  $Opt(\epsilon/2) = N_0$ , we have

$$\lambda \leq 4 \max_{0 \leq l < t} \frac{N_0 \|U_l\|_F^2}{\|B_l\|_F^2} \leq 9 Opt(\epsilon/2) \mu^2(A),$$

where the last inequality is due to the result of Lemma 2.7. Combining this with Lemma 2.5, we have that the number of iterations of the algorithm is bounded by

$$t \leq \left\lceil 18Opt(\epsilon/2)\mu^2(A) \ln \left( \frac{\|B\|_F}{\epsilon} \right) \right\rceil.$$

■

### 3. Deterministic Low-Rank Matrix Reconstruction

In this section, we give a deterministic algorithm for low rank matrix reconstruction based on the greedy approach that we have introduced and analyzed for the generalized sparse approximation problem (see Figure 2).

---

**Algorithm 2** The low-rank reconstruction algorithm

---

- 1: **procedure** LOWRANKAPPROXIMATION( $A, k$ )
  - 2:   compute  $U_k$  and  $\Sigma_k$  of  $A$
  - 3:   return Greedy( $A, U_k \Sigma_k, \epsilon \|A - A_k\|_F$ )
  - 4: **end procedure**
- 

The algorithm first computes  $U_k$ , the top  $k$  left singular vectors of  $A$  and  $\Sigma_k$  the first  $k$  singular values of  $A$ , which can be performed by standard methods like Lanczos. The columns of  $A$  are then selected in a greedy fashion so as to “fit” them to the subspace spanned by the columns of  $U_k \Sigma_k$ . Intuitively, we select columns of  $A$  which are close to the columns of  $U_k \Sigma_k$  and the analysis shows that the sub-matrix  $C$  of  $A$  we obtain is provably close to the “best” rank- $k$  approximation to  $A$ . The error parameter which is given as an input to Greedy is  $\epsilon \|A - A_k\|_F$ . The following result provides an upper bound on the number of columns of the optimal solution at error  $\epsilon \|A - A_k\|_F/2$ .

**Lemma 3.1.** *There exists a column sub-matrix  $C$  of  $A$  with  $c = O(k \log k / \epsilon^2)$  columns such that  $\|U_k \Sigma_k - CC^+ U_k \Sigma_k\|_F \leq \epsilon \|A - A_k\|_F/2$ .*

*Proof.* We will make use of the following result which is proved in [10]. They give a randomized algorithm which constructs, with non-zero probability a set of columns with a particular approximation property which immediately translates to an existence result. For a set of columns  $C \in A$ , denote the sampling matrix which selects the columns by  $S$  so that  $C = AS$ . Let  $V_k$  be the matrix of the first  $k$  right singular vectors of  $A$ . Let  $V_{r-k}$  be the matrix containing the last  $r - k$  right singular vectors of  $A$ , and let  $\Sigma_k$  and  $\Sigma_{r-k}$  be the diagonal matrices containing the first  $k$  and the last  $r - k$  singular values of  $A$ .

**Theorem 3.2 ([10])** *There exists a set of  $c = O(k \log k / \epsilon^2)$  columns from  $A$  and corresponding sampling matrix  $S$ , with  $C = AS$  such that  $\text{rank}(V_k^T S) = \text{rank}(V_k)$ ,  $\|\Sigma_{r-k} V_{r-k}^T S (V_k^T S)^+\|_F \leq \epsilon \|A - A_k\|_F$  where  $\Sigma_{r-k}$  is the diagonal matrix containing the smallest  $r - k$  singular values of  $A$ , and  $V_{r-k}$  is the matrix containing the last  $r - k$  right singular vectors of  $A$ .*

To proceed with the proof of the lemma, let  $C = AS$  be the column sub-matrix whose existence is guaranteed by the theorem above. We have

$$\begin{aligned}\epsilon^2 \|A - A_k\|_F^2 &\geq \|\Sigma_{r-k} V_{r-k}^T S(V_k^T S)^+\|_F^2 \\ &= \|\Sigma_k - \Sigma_k V_k^T S(V_k^T S)^+\|_F^2 + \|\Sigma_{r-k} V_{r-k}^T S(V_k^T S)^+\|_F^2,\end{aligned}$$

where the first term in the last expression is just 0 as  $V_k^T S(V_k^T S)^+ = I_k$ . Combining the last two terms into one expression, we have

$$\begin{aligned}\epsilon^2 \|A - A_k\|_F^2 &\geq \left\| \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} - \begin{pmatrix} \Sigma_k V_k^T \\ \Sigma_{r-k} V_{r-k}^T \end{pmatrix} S(V_k^T S)^+ \right\|_F^2 \\ &= \left\| \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} - \begin{pmatrix} \Sigma_k & 0 \\ 0 & \Sigma_{r-k} \end{pmatrix} \begin{pmatrix} V_k^T \\ V_{r-k}^T \end{pmatrix} S(V_k^T S)^+ \right\|_F^2 \\ &= \left\| \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} - (\Sigma V^T S)(\Sigma_k V_k^T S)^+ \Sigma_k \right\|_F^2 \\ &= \left\| \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} - (\Sigma V^T S)Y \right\|_F^2,\end{aligned}$$

where  $Y = (\Sigma_k V_k^T S)^+ \Sigma_k$ . Let  $A, B$  be arbitrary matrices. Then,  $\min_X \|A - BX\|_F^2 = \|A - BB^+A\|_F^2$  (see [15]). We continue as follows:

$$\begin{aligned}\left\| \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} - (\Sigma V^T S)Y \right\|_F^2 &\geq \min_{X \in \mathbb{R}^{c \times k}} \left\| \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} - (\Sigma V^T S)X \right\|_F^2 \\ &= \left\| \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} - (\Sigma V^T S)(\Sigma V^T S)^+ \begin{pmatrix} \Sigma_k \\ 0 \end{pmatrix} \right\|_F^2 \\ &= \left\| \begin{pmatrix} I_k \\ 0 \end{pmatrix} \Sigma_k - (\Sigma V^T S)(\Sigma V^T S)^+ \begin{pmatrix} I_k \\ 0 \end{pmatrix} \Sigma_k \right\|_F^2 \\ &= \left\| U \begin{pmatrix} I_k \\ 0 \end{pmatrix} \Sigma_k - (U \Sigma V^T S)(\Sigma V^T S)^+ U^T U_k \Sigma_k \right\|_F^2 \\ &= \|U_k \Sigma_k - (U \Sigma V^T S)(U \Sigma V^T S)^+ U_k \Sigma_k\|_F^2 \\ &= \|U_k \Sigma_k - CC^+ U_k \Sigma_k\|_F^2,\end{aligned}$$

where we have used  $U \Sigma V^T = A$  and  $C = AS$ . Choosing an error parameter  $\epsilon' = \epsilon/2$  gives the desired result. ■

We now, give the proof of Theorem 1.1.

**Proof of Theorem 1.1:** By the algorithm, we have

$$U_k \Sigma_k = CC^+ U_k \Sigma_k + E,$$

for some generic error matrix  $E$  satisfying  $\|E\|_F \leq \epsilon \|A - A_k\|_F$ . Multiplying both sides by  $V_k^T$ , we get

$$U_k \Sigma_k V_k^T = CC^+ U_k \Sigma_k V_k^T + EV_k^T,$$

which is clearly

$$A_k = CC^+ A_k + EV_k^T.$$

Rearranging and adding  $A$  to the both sides of the equation, we obtain  $A - CC^+ A_k = A - A_k + EV_k^T$ . Taking norms of both sides, and noting that  $C^+ A$  is the minimizer of  $\|A - CX\|_F$  over  $X$ , we obtain

$$\begin{aligned} \|A - CC^+ A\|_F &\leq \|A - CC^+ A_k\|_F \\ &= \|A - A_k + EV_k^T\|_F, \\ &\leq \|A - A_k\|_F + \|E\|_F \|V_k^T\|_F \\ &\leq \|A - A_k\|_F + \epsilon \sqrt{k} \|A - A_k\|_F \\ &= (1 + \epsilon \sqrt{k}) \|A - A_k\|_F. \end{aligned}$$

Third line follows due to the triangle inequality and sub-multiplicativity of the Frobenius norm. The fourth line is due to the fact that  $\|E\|_F \leq \epsilon \|A - A_k\|_F$  and  $\|V_k^T\|_F \leq \sqrt{k}$ . Choosing an error parameter  $\epsilon' = \epsilon/\sqrt{k}$  and combining Theorem 2.3 and Lemma 3.1 gives the desired result. ■

Note that, the number of columns chosen by the algorithm depends on  $\mu(A)$ , i.e. the structure of  $A$ . To get an idea of what this result implies when the number of columns chosen is  $k$ , we give the following corollary, which immediately follows upon a careful choice of error parameter.

**Corollary 3.3.** *After  $k$  iterations of Greedy, the sub-matrix  $C$  chosen satisfies  $\|A - CC^+ A\|_F \leq O(\mu(A)\sqrt{k \log k}) \|A - A_k\|_F$ , provided  $\ln(\|A\|_2)$  is bounded above by a constant.*

*Proof.* For  $\epsilon = \mu(A)\sqrt{k \log k}$

$$c = O\left(\frac{k^2 \log k}{\epsilon^2} \mu^2(A) \ln\left(\frac{\sqrt{k} \|A_k\|_F}{\epsilon \|A - A_k\|_F}\right)\right) = O\left(k \ln\left(\frac{\|A_k\|_F}{\mu(A)\sqrt{\log k} \|A - A_k\|_F}\right)\right)$$

Since  $\mu(A) \geq \|A^+\|_2 = 1/\sigma_r(A)$  (see [15]),  $\|A - A_k\|_F \geq \sigma_r \sqrt{r - k + 1}$ , and  $\|A_k\|_F \leq \sqrt{k} \|A\|_2$ , we have

$$\ln\left(\frac{\|A_k\|_F}{\mu(A)\sqrt{\log k} \|A - A_k\|_F}\right) \leq \ln\left(\frac{\sqrt{k} \|A\|_2}{\sqrt{(r - k + 1) \log k}}\right)$$

Hence, the last expression is constant, given the condition in the corollary. It is clear that an appropriate constant can be chosen as a coefficient to  $\epsilon$  so as to make the number of columns  $c = k$ . ■

#### 4. Numerical Results

In this section, we present numerical experiments using our algorithm Greedy, comparing it to a few other significant algorithms providing bounds for the performance metric we have analyzed. We report the error ratios  $\|A - CC^+A\|_2/\|A - A_k\|_2$ ,  $\|A - CC^+A\|_F/\|A - A_k\|_F$  for various matrices and different values of  $k$  along with the running times on one of the matrices. We make use of 3 different types of  $n \times n$  matrices for  $n = 400$  and  $n = 1000$ , a total of 6 different matrices. Running times are only reported for  $n = 1000$ . Below are the matrices that are used in our experiments:

- *Log*: a random matrix  $A$  with singular values equally spaced between 1 and  $10^{-\log n}$ . More specifically,  $A = U\Sigma V^T$ , where  $\Sigma$  is the diagonal matrix with entries of the logarithmic distribution, and  $U$  and  $V$  are random orthogonal matrices.
- *Scaled Random*: a random matrix  $A$  created by assigning each entry a number between  $-1$  and  $1$  from uniform distribution, and then scaling the  $i^{th}$  row of that matrix by  $(20\epsilon)^{i/n}$  where  $\epsilon$  is the machine precision. In our case,  $\epsilon = 2.22 \cdot 10^{-16}$ . This matrix was utilized in [18].
- *Kahan*: a matrix

$$A = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & \zeta & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \zeta^{n-1} \end{pmatrix} \cdot \begin{pmatrix} 1 & -\phi & \dots & -\phi \\ 0 & 1 & \dots & -\phi \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

with  $\zeta, \phi > 0$ , and  $\phi^2 + \zeta^2 = 1$ . Kahan matrices are first mentioned in [23]. These matrices are low-rank and they provably yield bad results for the commonly used pivoted QR algorithm [24]. Along the same lines in [18], we set  $\phi = 0.285$  for our experiments.

The variation in the results were negligible with respect to the random choices in the construction of the first two classes of matrices, hence we report results of one randomly generated matrix in each class. We have implemented the following 3 algorithms in C++ along with Greedy and performed experiments on an Intel Core 2 Duo T4200 at 2.16 Ghz, 4 GB machine:

- *Pivoted-QR*: The algorithm of Golub and Businger [24]. [18] shows that it chooses a sub-matrix  $C$  satisfying  $\|A - CC^+A\|_2 \leq 2^k \sqrt{n-k} \|A - A_k\|_2$ . We report the running times of choosing exactly  $k$  columns, not of a complete decomposition.
- *Low-RRQR*: The algorithm introduced by Chan and Hansen in [16], which provides  $\|A - CC^+A\|_2 \leq 2^{k+1} \sqrt{(k+1)n} \|A - A_k\|_2$ . This algorithm also involves computation of singular vectors, and requires a full QR decomposition as a preliminary step. We report the running times including this preliminary step for which we used pivoted-QR, followed by  $k$  iterations of the algorithm.
- *Hybrid*: The algorithm by Boutsidis et al. [21], which combines random sampling techniques and deterministic approaches. It guarantees  $\|A - CC^+A\|_2 \leq O\left(k^{\frac{3}{4}} \log^{\frac{1}{2}}(k)(n-k)^{\frac{1}{4}}\right) \|A - A_k\|_2$ , and  $\|A - CC^+A\|_F = O(k\sqrt{\log k}) \|A - A_k\|_F$ , by using different sampling distributions for spectral and Frobenius norms. We report the

error ratios of the algorithm run using the specific sampling distribution tailored to the norm. This algorithm first chooses (on average)  $c$  columns randomly on a matrix related to the right singular vectors of  $A$ , and then makes use of a deterministic procedure to cut down the number of columns to  $k$ . We have chosen  $c = 6k$  and used Pivoted-QR algorithm as the deterministic step. The number  $c$  is theoretically of order  $O(k \log k)$ , but in practice the authors suggest to use a value between  $2k$  and  $10k$  [25]. We run the algorithm 40 times to boost the success probability and get the best error ratio, as suggested in [21].

For the computation of partial SVD (top  $k$  singular values and singular vectors), which are required for Hybrid and Greedy, we have used a C version of the SVDPACK library [26], which utilizes Lanczos methods.

We show the error ratios of the algorithms on matrices of size  $400 \times 400$  on Table I, II and III. The behavior of the algorithms on the matrices of size  $1000 \times 1000$  are quite similar, and for convenience we give the results for these matrices in Figures 1, 2 and 3. In Frobenius norm, Greedy consistently outperforms the other algorithms tested, especially when  $k$  is small. This is due to the rationale of the algorithm, that it is trying to choose column vectors whose span is as close as possible to  $A_k$ . It is intuitively reasonable to expect that the distance between any column vector to the subspace chosen by Greedy should be close to the distance between that vector to the optimal subspace, which is quantitatively expressed via the ratio of the Frobenius norm errors. The only exception is the matrix *Scaled Random* for large values of  $k$ . Note that, even the Pivoted-QR algorithm works very well for this type of matrix. Greedy also presents very good results in spectral norm except small values of  $k$  on *Kahan*. Low-RRQR gives the best results for small  $k$  on this matrix as it has very low-rank with rapidly decreasing singular values. Pivoted-QR performs poorly on *Kahan* as expected.

Table IV gives the running times of the algorithms on the  $1000 \times 1000$  *Scaled Random* matrix. Pivoted-QR is the fastest algorithm, and Greedy is faster than Low-RRQR. If the time-consuming preliminary decomposition in Low-RRQR is disregarded, these two algorithms have quite similar behavior in terms of running time. Hybrid is the slowest of all due to the large number of repetitions.

## 5. Conclusion

We have presented an algorithm that approximates the space spanned by the top  $k$  left singular vectors of a matrix by introducing a generalization of the sparse approximation problem. The analysis of the algorithm is based on the generalized case of approximating an arbitrary subspace. Hence, the term  $\mu(A)$  that appears in the analysis is a general bound for a term for which there is not an “easy” function to bound. We believe that a more refined analysis focusing on the specific problem of approximating  $A_k$  will yield much better theoretical guarantees. The experiments also suggest that one might get bounds in spectral norm.

## REFERENCES

1. Kuruvilla FG, Park PJ, Schreiber SL. Vector algebra in the analysis of genome-wide expression data. *Genome Biology* 2002; (3).

| $k$ | $\ A - CC^+A\ _2 / \ A - A_k\ _2$ |        |        |        | $\ A - CC^+A\ _F / \ A - A_k\ _F$ |        |        |        |
|-----|-----------------------------------|--------|--------|--------|-----------------------------------|--------|--------|--------|
|     | P-QR                              | L-RRQR | Hybrid | Greedy | P-QR                              | L-RRQR | Hybrid | Greedy |
| 1   | 1.035                             | 1.035  | 1.035  | 1.035  | 1.035                             | 1.035  | 1.035  | 1.035  |
| 2   | 1.042                             | 1.058  | 1.007  | 1.003  | 1.030                             | 1.029  | 1.039  | 1.020  |
| 3   | 1.069                             | 1.093  | 1.019  | 1.005  | 1.042                             | 1.045  | 1.049  | 1.034  |
| 4   | 1.105                             | 1.101  | 1.060  | 1.045  | 1.055                             | 1.062  | 1.068  | 1.042  |
| 5   | 1.137                             | 1.116  | 1.117  | 1.035  | 1.072                             | 1.074  | 1.072  | 1.051  |
| 6   | 1.130                             | 1.144  | 1.079  | 1.042  | 1.089                             | 1.092  | 1.089  | 1.064  |
| 7   | 1.153                             | 1.160  | 1.114  | 1.093  | 1.098                             | 1.104  | 1.109  | 1.075  |
| 8   | 1.192                             | 1.195  | 1.125  | 1.094  | 1.111                             | 1.120  | 1.114  | 1.083  |
| 9   | 1.233                             | 1.213  | 1.189  | 1.110  | 1.128                             | 1.136  | 1.145  | 1.097  |
| 10  | 1.275                             | 1.220  | 1.202  | 1.130  | 1.145                             | 1.147  | 1.151  | 1.107  |
| 20  | 1.500                             | 1.404  | 1.409  | 1.256  | 1.274                             | 1.266  | 1.296  | 1.222  |
| 30  | 1.508                             | 1.678  | 1.533  | 1.406  | 1.372                             | 1.395  | 1.404  | 1.327  |
| 40  | 1.813                             | 1.678  | 1.668  | 1.536  | 1.483                             | 1.509  | 1.522  | 1.432  |
| 50  | 1.935                             | 1.896  | 1.851  | 1.612  | 1.596                             | 1.621  | 1.582  | 1.539  |

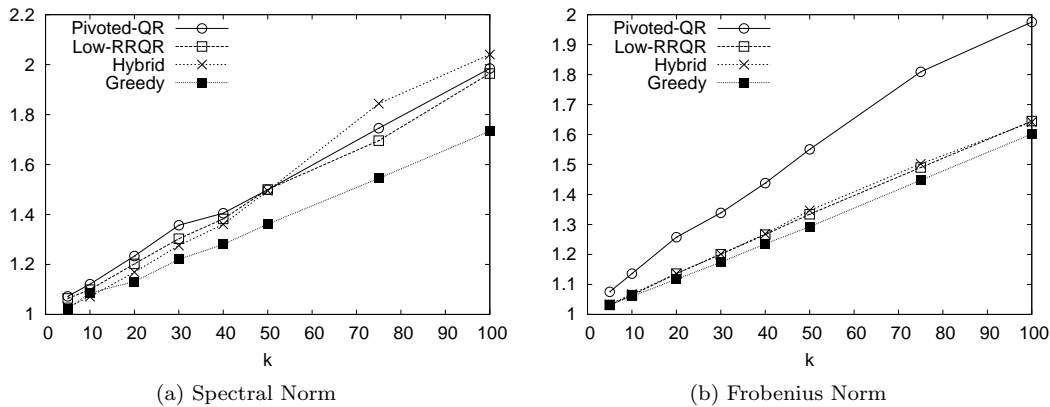
Table I: Error ratios for *Log*  $400 \times 400$

| $k$ | $\ A - CC^+A\ _2 / \ A - A_k\ _2$ |        |        |        | $\ A - CC^+A\ _F / \ A - A_k\ _F$ |        |        |        |
|-----|-----------------------------------|--------|--------|--------|-----------------------------------|--------|--------|--------|
|     | P-QR                              | L-RRQR | Hybrid | Greedy | P-QR                              | L-RRQR | Hybrid | Greedy |
| 1   | 1.015                             | 1.015  | 1.015  | 1.015  | 1.080                             | 1.080  | 1.080  | 1.080  |
| 2   | 1.119                             | 1.042  | 1.067  | 1.016  | 1.067                             | 1.048  | 1.073  | 1.040  |
| 3   | 1.118                             | 1.060  | 1.115  | 1.024  | 1.085                             | 1.080  | 1.097  | 1.069  |
| 4   | 1.231                             | 1.185  | 1.108  | 1.042  | 1.119                             | 1.121  | 1.129  | 1.095  |
| 5   | 1.101                             | 1.164  | 1.120  | 1.078  | 1.135                             | 1.136  | 1.160  | 1.111  |
| 6   | 1.183                             | 1.213  | 1.192  | 1.079  | 1.154                             | 1.158  | 1.218  | 1.142  |
| 7   | 1.225                             | 1.173  | 1.213  | 1.132  | 1.191                             | 1.167  | 1.223  | 1.168  |
| 8   | 1.276                             | 1.234  | 1.161  | 1.090  | 1.219                             | 1.192  | 1.215  | 1.190  |
| 9   | 1.339                             | 1.257  | 1.305  | 1.158  | 1.249                             | 1.210  | 1.258  | 1.231  |
| 10  | 1.317                             | 1.328  | 1.307  | 1.307  | 1.265                             | 1.233  | 1.282  | 1.241  |
| 20  | 1.577                             | 1.597  | 1.676  | 1.417  | 1.450                             | 1.435  | 1.451  | 1.456  |
| 30  | 1.673                             | 2.137  | 1.527  | 1.723  | 1.621                             | 1.705  | 1.695  | 1.708  |
| 40  | 2.067                             | 2.171  | 1.997  | 1.912  | 1.753                             | 1.833  | 1.845  | 1.905  |
| 50  | 2.222                             | 1.939  | 1.936  | 2.244  | 1.936                             | 1.935  | 1.929  | 2.085  |

Table II: Error ratios for *Scaled Random*  $400 \times 400$

2. Natarajan BK. Sparse approximate solutions to linear systems. *SIAM Journal on Computing* 1995; **24**(2):227–234.
3. Chandrasekaran S, Ipsen ICF. On rank-revealing factorizations. *SIAM Journal on Matrix Analysis and Applications* 1994; **15**:592–622.
4. Frieze A, Kannan R, Vempala S. Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the Association for Computing Machinery* 2004; **51**(6):1025–1041, doi: <http://doi.acm.org/10.1145/1039488.1039494>.

| $k$ | $\ A - CC^+A\ _2 / \ A - A_k\ _2$ |        |        |        | $\ A - CC^+A\ _F / \ A - A_k\ _F$ |        |        |        |
|-----|-----------------------------------|--------|--------|--------|-----------------------------------|--------|--------|--------|
|     | P-QR                              | L-RRQR | Hybrid | Greedy | P-QR                              | L-RRQR | Hybrid | Greedy |
| 1   | 10.343                            | 10.343 | 10.343 | 10.343 | 4.383                             | 4.383  | 4.383  | 4.383  |
| 2   | 8.539                             | 1.342  | 1.314  | 1.308  | 2.759                             | 1.064  | 1.103  | 1.063  |
| 3   | 9.401                             | 1.308  | 1.387  | 1.381  | 2.879                             | 1.084  | 1.069  | 1.068  |
| 4   | 9.806                             | 1.320  | 1.388  | 1.381  | 2.989                             | 1.083  | 1.068  | 1.068  |
| 5   | 10.218                            | 1.343  | 1.394  | 1.381  | 3.102                             | 1.083  | 1.068  | 1.068  |
| 6   | 10.638                            | 1.264  | 1.383  | 1.381  | 3.216                             | 1.103  | 1.069  | 1.068  |
| 7   | 11.063                            | 1.273  | 1.594  | 1.381  | 3.332                             | 1.101  | 1.123  | 1.068  |
| 8   | 11.496                            | 1.296  | 1.619  | 1.381  | 3.450                             | 1.099  | 1.121  | 1.068  |
| 9   | 11.988                            | 1.248  | 1.622  | 1.381  | 3.579                             | 1.122  | 1.141  | 1.068  |
| 10  | 12.449                            | 1.250  | 1.642  | 1.381  | 3.705                             | 1.119  | 1.113  | 1.068  |
| 20  | 17.340                            | 1.321  | 1.612  | 1.381  | 5.055                             | 1.136  | 1.114  | 1.068  |
| 30  | 18.477                            | 1.406  | 1.612  | 1.382  | 5.373                             | 1.183  | 1.117  | 1.068  |
| 40  | 18.215                            | 1.589  | 1.612  | 1.382  | 5.300                             | 1.181  | 1.133  | 1.068  |
| 50  | 15.633                            | 1.437  | 1.612  | 1.382  | 4.580                             | 1.141  | 1.114  | 1.068  |

Table III: Error ratios for *Kahan*  $400 \times 400$ Figure 1: Error ratios for *Log*  $1000 \times 1000$ 

5. Drineas P, Kannan R, Mahoney MW. Fast monte carlo algorithms for matrices II: Computing a low-rank approximation to a matrix. *SIAM Journal on Computing* 2006; **36**(1):158–183, doi: <http://dx.doi.org/10.1137/S0097539704442696>.
6. Drineas P, Frieze A, Kannan R, Vempala S, Vinay V. Clustering in large graphs and matrices. *SODA '99: Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete algorithms*, SIAM, 1999; 291–299.
7. Rudelson M, Vershynin R. Sampling from large matrices: An approach through geometric functional analysis. *Journal of the Association for Computing Machinery* 2007; **54**(4).
8. Deshpande A, Rademacher L, Vempala S, Wang G. Matrix approximation and projective clustering via volume sampling. *SODA '06: Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms*, ACM Press, 2006; 1117–1126.
9. Deshpande A, Vempala S. Adaptive sampling and fast low-rank matrix approximation. *RANDOM'06*:

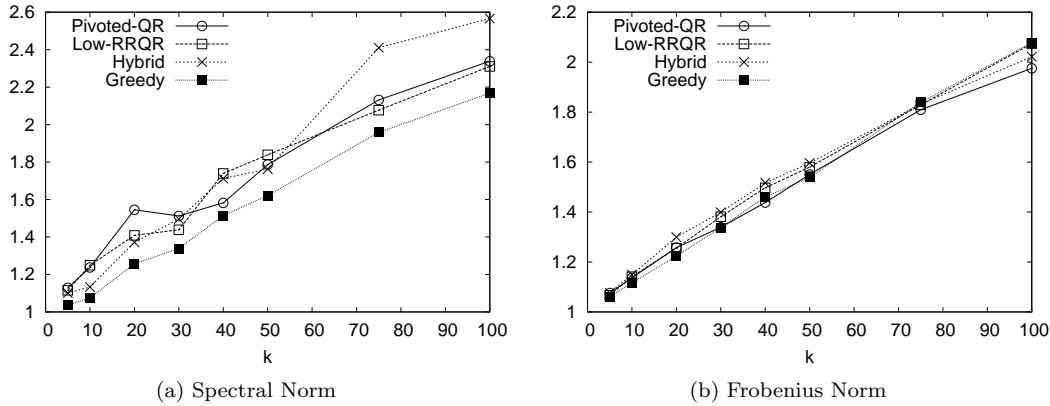


Figure 2: Error ratios for *Scaled Random*  $1000 \times 1000$

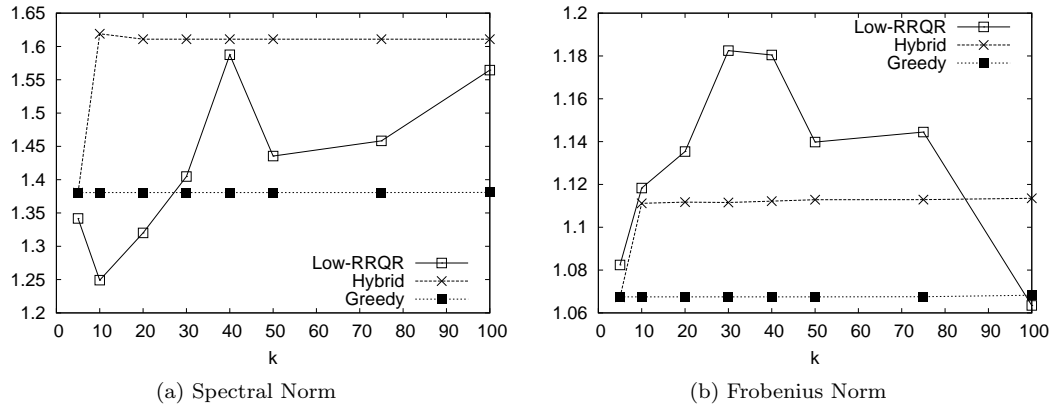


Figure 3: Error ratios for *Kahan*  $1000 \times 1000$

10th International Workshop on Randomization and Computation, Springer, 2006; 292–303.

10. Drineas P, Mahoney MW, Muthukrishnan S. Subspace sampling and relative-error matrix approximation: Column-based methods. *APPROX-RANDOM*, 2006; 316–326.
11. Sarlos T. Improved approximation algorithms for large matrices via random projections. *FOCS '06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, IEEE Computer Society: Washington, DC, USA, 2006; 143–152, doi:http://dx.doi.org/10.1109/FOCS.2006.37.
12. Shyamalkumar ND, Varadarajan K. Efficient subspace approximation algorithms. *SODA '07: Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete algorithms*, Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2007; 532–540.
13. Deshpande A, Varadarajan K. Sampling-based dimension reduction for subspace approximation. *STOC '07: Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of computing*, ACM: New York, NY, USA, 2007; 641–650, doi:http://doi.acm.org/10.1145/1250790.1250884.
14. Chan TF. Rank revealing QR factorizations. *Linear Algebra and its Applications* 1987; (88/89):67–82.
15. Golub GH, Loan CV. *Matrix Computations*. Johns Hopkins U. Press, 1996.
16. Chan TF, Hansen P. Low-rank revealing QR factorizations. *Numerical Linear Algebra with Applications*

| k   | P-QR  | L-RRQR | Hybrid | Greedy |
|-----|-------|--------|--------|--------|
| 5   | 0.047 | 2.359  | 2.235  | 0.375  |
| 10  | 0.078 | 2.625  | 3.235  | 0.501  |
| 20  | 0.140 | 3.047  | 4.875  | 0.891  |
| 30  | 0.203 | 3.468  | 7.001  | 1.079  |
| 40  | 0.265 | 4.234  | 8.985  | 1.468  |
| 50  | 0.359 | 4.313  | 12.359 | 1.922  |
| 75  | 0.453 | 5.109  | 19.078 | 2.798  |
| 100 | 0.578 | 6.063  | 25.546 | 3.687  |

Table IV: Running times for *Scaled Random*  $1000 \times 1000$ 

- 1994; (1):33–44.
17. de Hoog FR, Mattheijb RMM. Subset selection for matrices. *Linear Algebra and its Applications* 2007; (422):349–359.
  18. Gu M, Eisenstat SC. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM Journal on Scientific Computing* 1996; **17**(4):848–869, doi:http://dx.doi.org/10.1137/0917055.
  19. Hong YP, Pan CT. Rank-revealing QR factorizations and the singular value decomposition. *Mathematics of Computation* 1992; **58**:213–232.
  20. Pan CT, Tang PTP. Bounds on singular values revealed by QR factorizations. *BIT Numerical Mathematics* 1999; **39**:740–756.
  21. Boutsidis C, Mahoney MW, Drineas P. An improved approximation algorithm for the column subset selection problem. *SODA '09: Proceedings of the Nineteenth Annual ACM -SIAM Symposium on Discrete Algorithms*, Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 2009; 968–977.
  22. Tropp JA. Greed is good: algorithmic results for sparse approximation. *IEEE Transactions on Information Theory* 2004; **50**(10):2231–2242.
  23. Kahan W. Numerical linear algebra. *Canadian Mathematical Bulletin* 1966; (9):757–801.
  24. Businger PA, Golub GH. Linear least squares solutions by Householder transformations. *Numerische Mathematik* 1965; (7):269–276.
  25. Boutsidis C, Mahoney MW, Drineas P. Unsupervised feature selection for principal components analysis. *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM: New York, NY, USA, 2008; 61–69, doi:http://doi.acm.org/10.1145/1401890.1401903.
  26. Berry M, Do T, O'Brien G, Krishna V, Varadhan S. SVDPACKC (version 1.0) user's guide. *Technical Report*, Knoxville, TN, USA 1993. URL <http://portal.acm.org/citation.cfm?id=898726>.