SIMPLER IS BETTER: A COMPARATIVE STUDY OF RANDOMIZED PIVOTING ALGORITHMS FOR CUR AND INTERPOLATIVE DECOMPOSITIONS

YIJUN DONG AND PER-GUNNAR MARTINSSON

Abstract: Matrix skeletonizations like the interpolative and CUR decompositions provide a framework for low-rank approximation in which subsets of a given matrix's columns and/or rows are selected to form approximate spanning sets for its column and/or row space. Such decompositions that rely on "natural" bases have several advantages over traditional low-rank decompositions with orthonormal bases, including preserving properties like sparsity or non-negativity, maintaining semantic information in data, and reducing storage requirements. Matrix skeletonizations can be computed using classical deterministic algorithms such as column pivoted QR, which work well for small-scale problems in practice, but suffer from slow execution as the dimension increases and can be vulnerable to adversarial inputs. More recently, randomized pivoting schemes have attracted much attention, as they have proven capable of accelerating practical speed, scale well with dimensionality, and sometimes also lead to better theoretical guarantees. This manuscript provides a comparative study of various randomized pivoting based matrix skeletonization algorithms that leverage classical pivoting schemes as building blocks. We propose a general framework that encapsulates the common structure of these randomized pivoting based algorithms, and provide an a-posteriori-estimable error bound for the framework. Additionally, we propose a novel concretization of the general framework, and numerically demonstrate its superior empirical efficiency.

Keywords: Low rank approximation, interpolative decomposition, CUR decomposition, randomized numerical linear algebra, column pivoted QR factorization, rank revealing factorization.

1. INTRODUCTION

The problem of computing a low-rank approximation to a matrix is a classical one that has drawn increasing attention due to its importance in the analysis of large data sets. At the core of low-rank matrix approximation is the task of constructing bases that approximately span the column and/or row spaces of a given matrix. This manuscript investigates algorithms for low-rank matrix approximations with "natural bases" of the column and row spaces – bases formed by selecting subsets of the actual columns and rows of the matrix. To be precise, given an $m \times n$ matrix $\mathbf{A}$ and a target rank $k < \min(m,n)$, we seek to determine an $m \times k$ matrix $\mathbf{C}$ holding k of the columns of $\mathbf{A}$, and a $k \times n$ matrix $\mathbf{Z}$ such that

$$
\begin{array}{cccc}
\mathbf{A} & \approx & \mathbf{C} & \mathbf{Z}. \\
m \times n & & m \times k & k \times n
\end{array}
\tag{1}
$$

We let J_{s} denote the index vector of length k that identifies the k chosen columns, so that, in MATLAB notation,

$$
\mathbf{C} = \mathbf{A}(:, J_{\mathrm{s}}).
\tag{2}
$$

If we additionally identify an index vector I_s that marks a subset of the rows that forms an approximate basis for the row space of $\mathbf{A}$, we can then form the "CUR" decomposition

$$
\begin{array}{ccccc}
\mathbf{A} & \approx & \mathbf{C} & \mathbf{U} & \mathbf{R}, \\
m \times n & & m \times k & k \times k & k \times n
\end{array}
\tag{3}
$$

where $\mathbf{U}$ is a $k \times k$ matrix, and

$$
\mathbf{R} = \mathbf{A}(I_\mathrm{s}, :).
\tag{4}
$$

The decomposition (3) is also known as a "matrix skeleton" [25] approximation (hence the subscript "s" for "skeleton" in I_s and J_s). Matrix decompositions of the form (1) or (3) possess several compelling properties: (i) Identifying I_s and/or J_s is often helpful in data interpretation. (ii) The decompositions (1) and (3) preserve important properties of the matrix $\mathbf{A}$. For instance, if $\mathbf{A}$ is sparse/non-negative, then $\mathbf{C}$ and $\mathbf{R}$ are also sparse/non-negative. (iii) The decompositions (1) and (3) are often memory efficient. In particular, when the entries of $\mathbf{A}$ itself are available, or can inexpensively be computed or retrieved, then once J_s and I_s have been determined, there is no need to store $\mathbf{C}$ and $\mathbf{R}$ explicitly.

Deterministic techniques for identifying close-to-optimal index vectors I_s and J_s are well established. Greedy algorithms such as the classical column pivoted QR (CPQR) [24, Sec. 5.4.1], and variations of LU with complete pivoting [52, 34] often work well in practice. There also exist specialized pivoting schemes that come with strong theoretical performance guarantees [27].

While effective for smaller dense matrices, classical techniques based on pivoting become computationally inefficient as the matrix sizes grow. The difficulty is that a global update to the matrix is in general required before the next pivot element can be selected. The situation becomes particularly dire for sparse matrices, as each update tends to create substantial fill-in of zero entries.

To better handle large matrices, and huge sparse matrices in particular, a number of algorithms based on *randomized sketching* have been proposed in recent years. The idea is to extract a "sketch" $\mathbf{Y}$ of the matrix that is far smaller than the original matrix, yet contains enough information that the index vectors I_s and/or J_s can be determined by using the information in $\mathbf{Y}$ alone. Examples include:

(1) *Discrete empirical interpolation method (DEIM):* The sketching step consists of computing approximations to the dominant left and right singular vectors of $\mathbf{A}$, for instance using the randomized SVD (RSVD) [28, 36]. Then a greedy pivot based scheme is used to pick the index sets I_s and J_s [21, 51].

(2) *Leverage score sampling:* Again, the procedures start by computing approximations to the dominant left and right singular vectors of $\mathbf{A}$ through a randomized scheme. Then these approximations are used to compute probability distributions on the row and/or column indices, from which a random subset of columns and/or rows is sampled.

(3) *Pivoting on a random sketch:* With a random matrix $\boldsymbol{\Omega} \in \mathbb{R}^{k \times m}$ drawn from some appropriate distribution, a sketch of $\mathbf{A}$ is formed via $\mathbf{Y} = \boldsymbol{\Omega}\mathbf{A}$. Then, a classical pivoting strategy such as the CPQR is applied on $\mathbf{Y}$ to identify a spanning set of columns.

The existing literature [3, 11, 13, 15, 16, 19, 21, 27, 37, 51, 58] presents compelling evidence in support of each of these frameworks, in the form of mathematical theory and/or empirical numerical experiments.

The objective of the present manuscript is to organize different strategies, and to conduct a systematic comparison, with the focus on their empirical accuracy and efficiency. In particular, we compare different strategies for extracting a random sketch, such as techniques based on Gaussian random matrices [28, 32, 38, 60], random fast transforms [8, 28, 38, 48, 54, 61], and random sparse embeddings [12, 38, 39, 43, 55, 60]. We also compare different pivoting strategies such as pivoted QR [24, 58] versus pivoted LU [11, 51]. Finally, we compare how well sampling based schemes perform in relation to pivoting based schemes.

In addition to providing a comparison of existing methods, the manuscript proposes a general framework that encapsulates the common structure shared by some popular randomized pivoting based algorithms, and presents an a-posteriori-estimable error bound for the framework. Moreover, the manuscript introduces a novel concretization of the general framework that is faster in execution than the schemes of [51, 58], while

picking equally close-to-optimal skeletons in practice. In its most basic version, our simplified method for finding a subset of k columns of $\mathbf{A}$ works as follows:

> *Sketching step:* Draw $\mathbf{\Omega} \in \mathbb{R}^{k \times m}$ from a Gaussian distribution and form $\mathbf{Y} = \mathbf{\Omega A}$.

> *Pivoting step:* Perform a *partially pivoted* LU decomposition of $\mathbf{Y}^*$. Collect the chosen pivot indices in the index vector J_s.

What is particularly interesting about this process is that while the LU factorization with partial pivoting (LUPP) is *not* rank revealing for a general matrix $\mathbf{A}$, the randomized mixing done in the sketching step makes LUPP excel at picking spanning columns. Furthermore, the randomness introduced by sketching empirically serves as a remedy for the vulnerability of classical pivoting schemes like LUPP to adversarial inputs (*e.g.*, the Kahan matrix [33]). The scheme can be accelerated further by incorporating a structured random embedding $\mathbf{\Omega}$. Alternatively, its accuracy can be enhanced by incorporating one of two steps of power iteration when building the sample matrix $\mathbf{Y}$.

The manuscript is organized as following: Section 2 provides a brief overview for the interpolative and CUR decompositions (Section 2.2), along with some essential building blocks of the randomized pivoting algorithms, including randomized linear embeddings (Section 2.3), randomized low-rank SVD (Section 2.4), and matrix decompositions with pivoting (Section 2.5). Section 3 reviews existing algorithms for matrix skeletonizations (Section 3.1, Section 3.2), and introduces a general framework that encapsulates the structures of some randomized pivoting based algorithms. In Section 4, we propose a novel concretization of the general framework, and provide an a-posteriori-estimable bound for the associated low-rank approximation error. With the numerical results in Section 5, we first compare the efficiency of various choices for the two building blocks in the general framework: randomized linear embeddings (Section 5.1) and matrix decompositions with pivoting (Section 5.2). Then, we demonstrate empirical advantages of the proposed algorithm by investigating the accuracy and efficiency of assorted randomized skeleton selection algorithms for the CUR decomposition (Section 5.3).

2. BACKGROUND

We first introduce some closely related low-rank matrix decompositions that rely on "natural" bases, including the CUR decomposition, and the column, row, and two-sided interpolative decompositions (ID) in Section 2.2. Section 2.3 describes techniques for computing randomized sketches of matrices, based on which Section 2.4 discusses the randomized construction of low-rank SVD. Section 2.5 describes how these can be used to construct matrix decompositions. While introducing the background, we include proofs of some well-established facts that provide key ideas, but are hard to extract from the context of relevant references.

2.1. **Notation.** Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be an arbitrary given matrix of rank $r \leq \min\{m, n\}$, whose SVD is given by

$$\mathbf{A} = \underset{m \times r}{\mathbf{U}_A} \underset{r \times r}{\mathbf{\Sigma}_A} \underset{r \times n}{\mathbf{V}_A^\top} = [\mathbf{u}_{A,1}, \ldots, \mathbf{u}_{A,r}] \operatorname{diag}(\sigma_{A,1}, \ldots, \sigma_{A,r}) [\mathbf{v}_{A,1}, \ldots, \mathbf{v}_{A,r}]^\top$$

such that for any rank parameter $k \leq r$, we denote $\mathbf{U}_{A,k} \triangleq [\mathbf{u}_{A,1}, \ldots, \mathbf{u}_{A,k}]$ and $\mathbf{V}_{A,k} \triangleq [\mathbf{v}_{A,1}, \ldots, \mathbf{v}_{A,k}]$ as the orthonormal bases of the dimension-k leading left and right singular subspaces of $\mathbf{A}$, while $\mathbf{U}_{A,k}^\perp \triangleq [\mathbf{u}_{A,k+1}, \ldots, \mathbf{u}_{A,r}]$ and $\mathbf{V}_{A,k}^\perp \triangleq [\mathbf{v}_{A,k+1}, \ldots, \mathbf{v}_{A,r}]$ as the orthonormal bases of the respective orthogonal complements. The diagonal submatrices consisting of the spectrum, $\mathbf{\Sigma}_{A,k} \triangleq \operatorname{diag}(\sigma_{A,1}, \ldots, \sigma_{A,k})$ and $\mathbf{\Sigma}_{A,k}^\perp \triangleq \operatorname{diag}(\sigma_{A,k+1}, \ldots, \sigma_{A,r})$, follow analogously. We denote $\mathbf{A}_k \triangleq \mathbf{U}_{A,k} \mathbf{\Sigma}_{A,k} \mathbf{V}_{A,k}^\top$ as the rank-k truncated SVD that minimizes rank-k approximation error of $\mathbf{A}$ ([22]). Furthermore, we denote the spectrum of $\mathbf{A}$, $\sigma(\mathbf{A})$, as a $r \times r$ diagonal matrix, while for each $i = 1, \ldots, r$, let $\sigma_i(\mathbf{A})$ be the i-th singular value of $\mathbf{A}$.

For the QR factorization, given an arbitrary rectangular matrices $\mathbf{M} \in \mathbb{R}^{d \times l}$ with full column rank ($d \geq l$), let $\mathbf{M} = \begin{bmatrix} \mathbf{Q}_M, \mathbf{Q}_M^\perp \end{bmatrix} [\mathbf{R}_M; \mathbf{0}]$ be a full QR factorization of $\mathbf{M}$ such that $\mathbf{Q}_M \in \mathbb{R}^{d \times l}$ and $\mathbf{Q}_M^\perp \in \mathbb{R}^{d \times (d-l)}$ consist of orthonormal bases of the subspace spanned by the columns of $\mathbf{M}$ and its orthogonal complement. We denote $\mathrm{ortho} : \left\{ \mathbf{M} \in \mathbb{R}^{d \times l} \mid \mathrm{rank}\,(\mathbf{M}) = l \right\} \to \mathbb{R}^{d \times l}$ ($d \geq l$) as a map that identifies an orthonormal basis (not necessary unique) for $\mathbf{M}$, $\mathrm{ortho}(\mathbf{M}) = \mathbf{Q}_M$.

We adapt the MATLAB notation for matrices throughout this work. Unless specified otherwise (*e.g.*, with subscripts), we use $\|\cdot\|$ to represent either the spectral norm or the Frobenius norm (*i.e.*, holding simultaneously for both norms).

2.2. **Interpolative and CUR decompositions.**

We first recall the definitions of the interpolative and CUR decompositions of a given $m \times n$ real matrix $\mathbf{A}$. After providing the basic definitions, we discuss first how well it is theoretically possible to do low-rank approximation under the constraint that natural bases must be used. We then briefly describe the further suboptimality incurred by standard algorithms.

2.2.1. *Basic definitions.* We consider low-rank approximations for $\mathbf{A}$ with column and/or row subsets as bases. Given an arbitrary linearly independent column subset $\mathbf{C} = \mathbf{A}\,(:, J_s)$ ($J_s \subset [n]$), the rank-$|J_s|$ column ID of $\mathbf{A}$ with respect to column skeletons J_s can be formulated as,

$$(5) \qquad\qquad \widehat{\mathbf{A}}_{*,J_s} \triangleq \mathbf{C}\mathbf{C}^\dagger \mathbf{A},$$

where $\mathbf{C}\mathbf{C}^\dagger$ is the orthogonal projector onto the spanning subspace of column skeletons. Analogously, given any linearly independent row subset $\mathbf{R} = \mathbf{A}\,(I_s, :)$ ($I_s \subset [m]$), the rank-$|I_s|$ column ID of $\mathbf{A}$ with respect to row skeletons I_s takes the form

$$(6) \qquad\qquad \widehat{\mathbf{A}}_{I_s,*} \triangleq \mathbf{A}\mathbf{R}^\dagger \mathbf{R},$$

where $\mathbf{R}^\dagger \mathbf{R}$ is the orthogonal projector onto the span of row skeletons. While with both column and row skeletons, we can construct low-rank approximations for $\mathbf{A}$ in two forms – the two-sided ID and CUR decomposition: with $|I_s| = |J_s|$, let $\mathbf{S} \triangleq \mathbf{A}\,(I_s, J_s)$ be an invertible two-sided skeleton of $\mathbf{A}$ such that

$$(7) \qquad \text{Two-sided ID:} \qquad\qquad \widehat{\mathbf{A}}_{I_s,J_s} \triangleq \left(\mathbf{C}\mathbf{S}^{-1} \right) \mathbf{S} \left(\mathbf{C}^\dagger \mathbf{A} \right)$$

$$(8) \qquad \text{CUR decomposition:} \qquad\qquad \widetilde{\mathbf{A}}_{I_s,J_s} \triangleq \mathbf{C} \left(\mathbf{C}^\dagger \mathbf{A}\mathbf{R}^\dagger \right) \mathbf{R}$$

where in the exact arithmetic, since $\mathbf{S}^{-1}\mathbf{S} = \mathbf{I}$, the two sided ID is equivalent to the column ID characterized by $\mathbf{C}$, *i.e.*, $\widehat{\mathbf{A}}_{I_s,J_s} = \widehat{\mathbf{A}}_{*,J_s}$. Nevertheless, the two-sided ID $\widehat{\mathbf{A}}_{I_s,J_s}$ and CUR decomposition $\widetilde{\mathbf{A}}_{I_s,J_s}$ differ in both suboptimality and conditioning.

Remark 1 (Suboptimality of ID versus CUR). For any given column and row skeletons $\mathbf{C}$ and $\mathbf{R}$,

$$(9) \qquad \left\| \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A} \right\| \leq \left\| \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\mathbf{R}^\dagger \mathbf{R} \right\| \leq \left(\left\| \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A} \right\|^2 + \left\| \mathbf{A} - \mathbf{A}\mathbf{R}^\dagger \mathbf{R} \right\|^2 \right)^{\frac{1}{2}}.$$

This comes from a simple orthogonal decomposition

$$\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\mathbf{R}^\dagger \mathbf{R} = \left(\mathbf{I}_m - \mathbf{C}\mathbf{C}^\dagger \right) \mathbf{A} + \mathbf{C}\mathbf{C}^\dagger \left(\mathbf{A} - \mathbf{A}\mathbf{R}^\dagger \mathbf{R} \right)$$

where $\left(\mathbf{I}_m - \mathbf{C}\mathbf{C}^\dagger \right)$ and $\mathbf{C}\mathbf{C}^\dagger$ are orthogonal projectors. Therefore with the Frobenius norm,

$$\left\| \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\mathbf{R}^\dagger \mathbf{R} \right\|_F^2 = \left\| \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A} \right\|_F^2 + \left\| \mathbf{C}\mathbf{C}^\dagger \left(\mathbf{A} - \mathbf{A}\mathbf{R}^\dagger \mathbf{R} \right) \right\|_F^2$$

$$\leq \left\| \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A} \right\|_F^2 + \left\| \mathbf{A} - \mathbf{A}\mathbf{R}^\dagger \mathbf{R} \right\|_F^2,$$

while with the spectral norm

$$\left\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\mathbf{R}^\dagger \mathbf{R}\right\|_2^2 = \max_{\|\mathbf{v}\|_2 \leq 1} \left\|\left(\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\right)\mathbf{v}\right\|_2^2 + \left\|\mathbf{C}\mathbf{C}^\dagger \left(\mathbf{A} - \mathbf{A}\mathbf{R}^\dagger \mathbf{R}\right)\mathbf{v}\right\|_2^2$$

$$\begin{cases} \geq \max_{\|\mathbf{v}\|_2 \leq 1} \left\|\left(\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\right)\mathbf{v}\right\|_2^2 = \left\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\right\|_2^2 \\ \leq \max_{\|\mathbf{v}\|_2 \leq 1} \left\|\left(\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\right)\mathbf{v}\right\|_2^2 + \left\|\mathbf{C}\mathbf{C}^\dagger \left(\mathbf{A} - \mathbf{A}\mathbf{R}^\dagger \mathbf{R}\right)\mathbf{v}\right\|_2^2 \leq \left\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\right\|_2^2 + \left\|\mathbf{A} - \mathbf{A}\mathbf{R}^\dagger \mathbf{R}\right\|_2^2 \end{cases}.$$

Remark 2 (Conditioning of ID versus CUR). The construction of CUR decomposition tends to be more ill-conditioned than that of two-sided ID. Precisely, for properly selected column and row skeletons J_s and I_s, the corresponding skeletons $\mathbf{S}$, $\mathbf{C}$, and $\mathbf{R}$ shares similar spectrum decay as $\mathbf{A}$, which is usually ill-conditioned in the context. In the CUR decomposition, both the bases $\mathbf{C}$, $\mathbf{R}$ and the small matrix $\mathbf{C}^\dagger \mathbf{A}\mathbf{R}^\dagger$ in the middle tend to suffer from large condition numbers as that of $\mathbf{A}$. In contrast, the only potentially ill-conditioned component in the two-sided ID is $\mathbf{S}$ (*i.e.*, despite being expressed in $\mathbf{S}^{-1}$ and $\mathbf{C}^\dagger$, $\left(\mathbf{C}\mathbf{S}^{-1}\right)$ and $\left(\mathbf{C}^\dagger \mathbf{A}\right)$ in Equation (7) are actually well-conditioned, and can be evaluated without direct inversions).

Remark 3 (Stable CUR). Numerically, the stable construction of a CUR decomposition $\widetilde{\mathbf{A}}_{I_s, J_s}$ can be conducted via (unpivoted) QR factorization of $\mathbf{C}$ and $\mathbf{R}$ ([2], Algorithm 2): let $\mathbf{Q}_C \in \mathbb{R}^{m \times |J_s|}$ and $\mathbf{Q}_R \in \mathbb{R}^{n \times |I_s|}$ be matrices from the QR whose columns form orthonormal bases for $\mathbf{C}$ and $\mathbf{R}^\top$, respectively, then

$$(10) \qquad \widetilde{\mathbf{A}}_{I_s, J_s} = \mathbf{Q}_C \left(\mathbf{Q}_C^\top \mathbf{A}\mathbf{Q}_R\right) \mathbf{Q}_R^\top.$$

2.2.2. *Notion of suboptimality.* Both interpolative and CUR decompositions share the common goal of identifying proper column and/or row skeletons for $\mathbf{A}$ whose column and/or row spaces are well covered by the respective spans of these skeletons. Without loss of generality, we consider the column skeleton selection problem: for a given rank $k < r$, we aims to find a proper column subset, $\mathbf{C} = \mathbf{A}(:, J_s)$ ($J_s \subset [n]$, $|J_s| = k$), such that

$$(11) \qquad \left\|\mathbf{A} - \widehat{\mathbf{A}}_{*, J_s}\right\| \leq \phi(k, m, n) \left\|\mathbf{A} - \mathbf{A}_k\right\|$$

where common choices of the norm $\|\cdot\|$ include the spectral norm $\|\cdot\|_2$ and Frobenius norm $\|\cdot\|_F$; $\phi(k, m, n)$ is a function with $\phi(k, m, n) \geq 1$ for all k, m, n, and depends on the choice of $\|\cdot\|$; and we recall that $\mathbf{A}_k \triangleq \mathbf{U}_{A,k} \mathbf{\Sigma}_{A,k} \mathbf{V}_{A,k}^\top$ yields the optimal rank-k approximation error. Meanwhile, similar low-rank approximation error bounds are desired for the row ID $\widehat{\mathbf{A}}_{I_s, *}$, two-sided ID $\widehat{\mathbf{A}}_{I_s, J_s}$, and CUR decomposition $\widetilde{\mathbf{A}}_{I_s, J_s}$.

2.2.3. *Suboptimality of matrix skeletonization algorithms.* The suboptimality of column subset selection, as well as the corresponding ID and CUR decomposition, has been widely studied in a variety of literature. Specifically, [25] proved that with $\mathbf{S} = \mathbf{A}(I_s, J_s)$ ($|I_s| = |J_s| = k$) being the the maximal-volume submatrix in $\mathbf{A}$, the corresponding CUR decomposition (called pseudoskeleton component in the original paper) satisfies Equation (11) in $\|\cdot\|_2$ with $\phi = O\left(\sqrt{k}\left(\sqrt{m} + \sqrt{n}\right)\right)$. However, [25] pointed out that skeletons associated with the maximal-volume submatrix are not guaranteed to minimize the low-rank approximation error in Equation (11). Moreover, from the algorithmic perspective, identification of the maximal-volume submatrix is known to be NP-hard ([62]). As the derivation of analysis for the strong rank-revealing QR factorization, [27] demonstrated the existence of a rank-k column ID with $\phi = \sqrt{1 + k(n - k)}$, and proposed a polynomial algorithm for constructing a relaxation with $\phi = O\left(\sqrt{k(n - k)}\right)$. Leveraging sampling based strategies, [18] showed the existence of a rank-k column ID with $\phi = \sqrt{(k + 1)(n - k)}$ for $\|\cdot\|_2$ and $\phi = \sqrt{1 + k}$ for $\|\cdot\|_F$ by upper bounding the expectation of $\left\|\mathbf{A} - \widehat{\mathbf{A}}_{*, J_s}\right\|$ for volume sampling. Later on, [17, 14] proposed polynomial-time algorithms for selecting such column skeletons. The recent work [15] unveiled the multiple-descent trend of the suboptimality factor ϕ with respect to the approximation rank k, and illustrated that, depending on the spectrum decay, the suboptimality factor can be as tight as $\phi = O\left(k^{1/4}\right)$ for small ks, while for larger ks that fall in certain intervals, $\phi = \Omega(\sqrt{k})$.

Randomized linear embedding	Theoretical best dimension reduction	$T_s(l, \mathbf{A}_k)$
Gaussian embedding	$l = \Omega\left(k/\epsilon^2\right)$	$O(\mathrm{nnz}(\mathbf{A}_k)l)$
SRTT	$l = \Omega\left(k \log k/\epsilon^2\right)$	$O(mn \log l)$
Sparse sign matrix	$l = \Omega\left(k \log k/\epsilon^2\right), \zeta = \Omega\left(\log k/\epsilon\right)$	$O(\mathrm{nnz}(\mathbf{A}_k)\zeta)$

TABLE 1. Lower bounds of ls that provide theoretical guarantee for Equation (12), and asymptotic complexities of sketching, $T_s(l, \mathbf{A}_k)$, for some common randomized linear embeddings.

2.3. **Randomized linear embeddings.** For a given matrix $\mathbf{A}_k \in \mathbb{R}^{m \times n}$ of rank $k \leq \min(m, n)$ (typically we consider $k \ll \min(m, n)$), and a distortion parameter $\epsilon \in (0, 1)$, a linear map $\mathbf{\Gamma} : \mathbb{R}^m \to \mathbb{R}^l$ (*i.e.*, $\mathbf{\Gamma} \in \mathbb{R}^{l \times m}$, typically we consider $l \ll m$ for embeddings) is called an ℓ_2 *linear embedding* of $\mathbf{A}_k$ with distortion ϵ if

$$(12) \qquad (1-\epsilon)\|\mathbf{A}_k \mathbf{x}\|_2 \leq \|\mathbf{\Gamma} \mathbf{A}_k \mathbf{x}\|_2 \leq (1+\epsilon)\|\mathbf{A}_k \mathbf{x}\|_2 \quad \forall \mathbf{x} \in \mathbb{R}^n.$$

A distribution $\mathcal{S}$ over linear maps $\mathbb{R}^m \to \mathbb{R}^l$ (or equivalently, over $\mathbb{R}^{l \times m}$) generates *randomized oblivious ℓ_2 linear embeddings* (abbreviated as *randomized linear embeddings*) if over $\mathbf{\Gamma} \sim \mathcal{S}$, Equation (12) holds for all $\mathbf{A}_k$ with at least constant probability. Given $\mathbf{A}_k$ and a randomized linear embedding $\mathbf{\Gamma} \sim \mathcal{S}$, $\mathbf{\Gamma} \mathbf{A}_k$ provides a (row) sketch of $\mathbf{A}_k$, and the process of forming $\mathbf{\Gamma} \mathbf{A}_k$ is known as sketching ([60, 38]).

Randomized linear embeddings are closely related to various concepts like the Johnson-Lindenstrauss lemma and the restricted isometry property, and are studied in a broad scope of literature. [38] (Section 8, 9) provided an overview for and instantiated some popular choices of randomized linear embeddings, including

(1) Gaussian embedddings: $\mathbf{\Gamma} \in \mathbb{R}^{l \times m}$ consisting of *i.i.d.* Gaussian entries $S_{ij} \sim \mathcal{N}(\mu, 1/l)$ ([32, 28, 60, 38]);

(2) subsampled randomized trigonometric transforms (SRTT): $\mathbf{\Gamma} = \sqrt{\frac{m}{l}} \mathbf{\Pi}_{m \to l} \mathbf{T} \mathbf{\Phi} \mathbf{\Pi}_{m \to m}$ where $\mathbf{\Pi}_{m \to l} \in \mathbb{R}^{l \times m}$ is a uniformly random selection of l out of m rows; $\mathbf{T}$ is an $m \times m$ unitary trigonometric transform (*e.g.*, discrete Hartley transform for $\mathbb{R}$, and discrete Fourier transform for $\mathbb{C}$); $\mathbf{\Phi} \triangleq \mathrm{diag}(\varphi_1, \ldots, \varphi_m)$ with *i.i.d.* Rademacher random variables $\{\varphi_i\}_{i \in [m]}$ flips signs randomly; and $\mathbf{\Pi}_{m \to m}$ is a random permutation ([61, 28, 48, 54, 8, 38]); and

(3) sparse sign matrices: $\mathbf{\Gamma} = \sqrt{\frac{m}{\zeta}} [\mathbf{s}_1, \ldots, \mathbf{s}_m]$ for some $2 \leq \zeta \leq l$, with *i.i.d.* ζ-sparse columns $\{\mathbf{s}_j \in \mathbb{R}^l\}_{j \in [m]}$ constructed such that each $\mathbf{s}_j$ is filled with ζ independent Rademacher random variables at uniformly random coordinates ([39, 43, 60, 12, 55, 38]).

Table 1 summarizes lower bounds on ls that provide theoretical guarantee for Equation (12), along with asymptotic complexities of sketching, denoted as $T_s(l, \mathbf{A}_k)$, for these randomized linear embeddings. In spite of the weaker guarantees for structured randomized embeddings (*i.e.*, SRTTs and sparse sign matrices) in the theory by a logarithmic factor, from the empirical perpective, $l = \Omega\left(k/\epsilon^2\right)$ is usually sufficient for all the embeddings in Table 1 when considering tasks such as constructing randomized rangefinders (which we subsequently leverage for fast skeleton selection). For instance, [28, 38] suggested taking $l = k + \Omega(1)$ (*e.g.*, $l = k + 10$) for Gaussian embeddings, $l = \Omega(k)$ for SRTTs, and $l = \Omega(k), \zeta = \min(l, 8)$ ([56]) for sparse sign matrices in practice.

2.4. **Randomized rangefinder and low-rank SVD.** Given $\mathbf{A} \in \mathbb{R}^{m \times n}$, the randomized rangefinder problem aims to construct a matrix $\mathbf{X} \in \mathbb{R}^{l \times n}$ such that the row space of $\mathbf{X}$ aligns well with the leading right singular subspace of $\mathbf{A}$ ([38]): $\|\mathbf{A} - \mathbf{A}\mathbf{X}^\dagger \mathbf{X}\|$ is sufficiently small for some unitary invariant norm $\|\cdot\|$ (*e.g.*, $\|\cdot\|_2$ or $\|\cdot\|_F$). When $\mathbf{X}$ admits full row rank, we call $\mathbf{X}$ a rank-l row space approximator of $\mathbf{A}$. The well-known optimality result from [22] demonstrated that, for a fixed rank k, the optimal rank-k row space

approximator of $\mathbf{A}$ is given by its leading k right singular vectors: $\left\| \mathbf{A} - \mathbf{A} \mathbf{V}_{A,k} \mathbf{V}_{A,k}^\top \right\|_F^2 = \left\| \mathbf{A} - \mathbf{A}_k \right\|_F^2 = \sum_{i=k+1}^{\min\{m,n\}} \sigma_i \left(\mathbf{A} \right)^2$.

A row sketch $\mathbf{X} = \mathbf{\Gamma} \mathbf{A}$ generated by some proper randomized linear embedding $\mathbf{\Gamma}$ is known to serve as a good solution for the randomized rangefinder problem with high probability. For instance, [28] demonstrated that, with the Gaussian embedding, a small constant oversampling $l - k \geq 4$ is sufficient for a good approximation:

$$(13) \qquad \mathbb{E}\left[\left\| \mathbf{A} - \mathbf{A} \mathbf{X}^\dagger \mathbf{X} \right\|_F^2 \right] \leq \frac{l-1}{l-k-1} \left\| \mathbf{A} - \mathbf{A}_k \right\|_F^2,$$

and moreover, $\left\| \mathbf{A} - \mathbf{A} \mathbf{X}^\dagger \mathbf{X} \right\|_F^2 \lesssim l(l-k) \log (l-k) \left\| \mathbf{A} - \mathbf{A}_k \right\|_F^2$ with high probability. Similar guarantees hold for spectral norm ([28], Section 10). The randomized rangefinder error depends on the spectral decay of $\mathbf{A}$, and can be aggravated by a flat spectrum. In this scenario, power iterations (with proper orthogonalization, [28]), as well as Krylov and block Krylov subspace iterations ([42]), may be incorporated after the initial sketching as a remedy. For example, with a randomized linear embedding $\mathbf{\Omega}$ of size $l \times n$, a row space approximator with q power iterations ($q \geq 1$) is given by

$$(14) \qquad \mathbf{X} = \mathbf{\Omega} \left(\mathbf{A}^\top \mathbf{A} \right)^q,$$

and takes $\mathbf{X}$ takes $O\left(T_s(l, \mathbf{A}) + (2q-1)\,\mathrm{nnz}(\mathbf{A})l \right)$ operations to construct. However, such plain power iteration in Equation (14) is numerically unstable, and can lead to large errors when $\mathbf{A}$ is ill-conditioned and $q > 1$. For a stable construction, orthogonalization can be applied at each iteration:

$$\mathbf{Y}^{(1)} = \mathbf{A} \mathbf{\Omega}^\top$$
$$\mathbf{Y}^{(i)} = \mathrm{ortho}\left(\mathbf{A}\,\mathrm{ortho}\left(\mathbf{A}^\top \mathbf{Y}^{(i-1)} \right) \right) \ \forall\, i = 2, \ldots, q \ (\text{if } q > 1)$$
$$(15) \qquad \mathbf{X} = \mathrm{ortho}\left(\mathbf{Y}^{(q)} \right)^\top \mathbf{A}$$

with an additional cost of $O\left(q(m+n)l^2 \right)$ overall.

In addition, with a proper l that does not exceed the exact rank of $\mathbf{A}$, the row sketch $\mathbf{X} \in \mathbb{R}^{l \times n}$ has full row rank almost surely. Precisely, recall $r = \mathrm{rank}(\mathbf{A}) \leq \min \{m, n\}$ from Section 2.1:

Lemma 1. *Let $\mathbf{\Gamma} \in \mathbb{R}^{l \times m}$ be a randomized linear embedding with* i.i.d. *entries drawn from some absolutely continuous distribution (with respect to the Lebesgue measure). Then with $l \leq r$, the row sketch $\mathbf{X} = \mathbf{\Gamma} \mathbf{A}$ has full row rank with probability* 1.

Proof of Lemma 1. We first claim that all $l \times l$ submatrices in $\mathbf{\Gamma}$ are invertible with probability 1. It is sufficient to show that all $l \times l$ submatrices have nonzero determinants almost surely. The determinant of an $l \times l$ matrix can be expressed as a polynomial in l^2 entries of $\mathbf{\Gamma}$, where zero set of the polynomial has Lebesgue measure 0 in $\mathbb{R}^{l^2}$. Then since the distribution of entries in $\mathbf{\Gamma}$ is absolutely continuous, the determinant is nonzero with probability 1. Second, for $\mathbf{X} = \mathbf{\Gamma} \mathbf{A}$ admitting full row rank, we observe that rows in $\mathbf{\Gamma}$ can be interpreted as independent random vectors in $\mathbb{R}^m$ with *i.i.d.* entries from some absolutely continuous distribution. We aim to show that projection of the dimension-l row space of $\mathbf{\Gamma}$ onto the range of $\mathbf{A}$ remains dimension l almost surely. But since any proper subspaces of $\mathbb{R}^m$ have Lebesgue measure 0 in $\mathbb{R}^m$, projection of the row space of $\mathbf{\Gamma}$ onto the range of $\mathbf{A}$ is dimension l, and therefore $\mathbf{X}$ admits full row rank, almost surely. ∎

A low-rank row space approximator $\mathbf{X}$ can be subsequently leveraged to construct a randomized rank-l SVD. Assuming l is properly chosen such that $\mathbf{X}$ has full row rank, let $\mathbf{Q}_X \in \mathbb{R}^{n \times l}$ be an orthonormal basis

for the row space of $\mathbf{X}$. [28] pointed out that the exact SVD of the smaller matrix $\mathbf{AQ}_X \in \mathbb{R}^{m \times l}$:

$$(16) \qquad \underset{m \times l}{\left[\widehat{\mathbf{U}}_A, \underset{l \times l}{\widehat{\mathbf{\Sigma}}_A}, \underset{l \times l}{\widetilde{\mathbf{V}}_A}\right]} = \mathrm{svd}\left(\underbrace{\mathbf{AQ}_X}_{m \times l}, \text{`econ'}\right), \quad \underset{n \times l}{\widehat{\mathbf{V}}_A} = \mathbf{Q}_X \widetilde{\mathbf{V}}_A,$$

can be evaluated efficiently in $O\left(ml^2\right)$ operations (and $O\left(nl^2\right)$ additional operations for constructing $\widehat{\mathbf{V}}_A$) such that $\mathbf{A} \approx \mathbf{A}\mathbf{X}^\dagger \mathbf{X} = \widehat{\mathbf{U}}_A \widehat{\mathbf{\Sigma}}_A \widehat{\mathbf{V}}_A^\top$.

2.5. **Matrix decompositions with pivoting.** We next briefly survey how pivoted QR and LU decompositions can be leveraged to resolve the matrix skeleton selection problem. In this section, $\mathbf{X} \in \mathbb{R}^{l \times n}$ denotes a matrix of full row rank (that will typically arise as a "row space approximator"). Let $\mathbf{X}^{(t)} \in \mathbb{R}^{l \times n}$ be the resulted matrix after the t-th step of pivoting and matrix updating, so that $\mathbf{X}^{(0)} = \mathbf{X}$.

2.5.1. *Column pivoted QR (CPQR).* Applying the CPQR to $\mathbf{X}$ gives:

$$(17) \qquad \underset{n \times n}{\mathbf{X}\,\mathbf{\Pi}_n} = \mathbf{X}\,[\underset{n \times l}{\mathbf{\Pi}_{n,1}},\ \underset{n \times (n-l)}{\mathbf{\Pi}_{n,2}}] = \underset{l \times l}{\mathbf{Q}_l}\,\underset{l \times n}{\mathbf{R}^{QR}} = \mathbf{Q}_l\left[\underset{l \times l}{\mathbf{R}_1^{QR}},\ \underset{l \times (n-l)}{\mathbf{R}_2^{QR}}\right],$$

where $\mathbf{Q}_l$ is an orthogonal matrix; $\mathbf{R}_1^{QR}$ is upper triangular; and $\mathbf{\Pi}_n \in \mathbb{R}^{n \times n}$ is a column permutation. QR decompositions rank-1 update the active submatrix at each step for orthogonalization (*e.g.*, [31], [52], Algorithm 10.1). For each $t = 0, \ldots, l-2$, at the $(t+1)$-th step, the CPQR searches the entire active submatrix $\mathbf{X}^{(t)}\,(t+1:l, t+1:n)$ for the $(t+1)$-th column pivot with the maximal ℓ_2-norm:

$$j_{t+1} = \underset{t+1 \le j \le n}{\mathrm{argmax}}\left\|\mathbf{X}^{(t)}\,(t+1:l, j)\right\|_2.$$

[27] illustrated that CPQR satisfies $\max_{i,j}\left|\left(\left(\mathbf{R}_1^{QR}\right)^{-1}\mathbf{R}_2^{QR}\right)_{ij}\right| \le 2^{l-i}$, and instantiated the classical Kahan matrix ([33]) as a pathological case that admits exponential growth. Nevertheless, these adversarial inputs are scarce and sensitive to perturbations. The empirical success of CPQR also suggests that exponential growth with respect to l almost never occurs in practice ([53]). Meanwhile, there exist more sophisticated variations of CPQR, like the rank-revealing ([10, 30]) and strong rank-revealing QR ([27]), guaranteeing that $\max_{i,j}\left|\left(\left(\mathbf{R}_1^{QR}\right)^{-1}\mathbf{R}_2^{QR}\right)_{ij}\right|$ is upper bounded by some low-degree polynomial in l, but coming with higher complexities as trade-off.

2.5.2. *LU with partial pivoting (LUPP).* Applying the LUPP columnwise to $\mathbf{X}$ yields:

$$(18) \qquad \underset{n \times n}{\mathbf{X}\,\mathbf{\Pi}_n} = \mathbf{X}\,[\underset{n \times l}{\mathbf{\Pi}_{n,1}},\ \underset{n \times (n-l)}{\mathbf{\Pi}_{n,2}}] = \underset{l \times l}{\mathbf{L}_l}\,\underset{l \times n}{\mathbf{R}^{LU}} = \mathbf{L}_l\left[\underset{l \times l}{\mathbf{R}_1^{LU}},\ \underset{l \times (n-l)}{\mathbf{R}_2^{LU}}\right],$$

where $\mathbf{L}_l$ is lower triangular; $\mathbf{R}_1^{LU}$ is upper triangular; $\mathbf{R}_1^{LU}\,(i,i) = 1$ and $\left|\mathbf{R}^{LU}\,(i,j)\right| \le 1$ for all $i \in [l]$, $i \le j \le n$; and $\mathbf{\Pi}_n \in \mathbb{R}^{n \times n}$ is a column permutation. LU decompositions update active submatrices via Shur complements (*e.g.*, [52], Algorithm 21.1): for $t = 0, \ldots, l-2$,

$$\mathbf{X}^{(t+1)}\,(t+2:l, t+2:n) = \mathbf{X}^{(t)}\,(t+2:l, t+2:n) - \mathbf{X}^{(t)}\,(t+2:l, t)\,\mathbf{X}^{(t)}\,(t, t+2:n)\,/\mathbf{X}^{(t)}\,(t,t).$$

At the $(t+1)$-th step, the (column-wise) LUPP searches only the $(t+1)$-th row in the active submatrix, $\mathbf{X}^{(t)}\,(t+1:l, t+1:n)$, and pivots

$$j_{t+1} = \underset{t+1 \le j \le n}{\mathrm{argmax}}\left|\mathbf{X}^{(t)}\,(t+1, j)\right|,$$

such that $\mathbf{R}^{LU}\,(i,j) = \mathbf{X}^{(i-1)}\,(i,j)\,/\mathbf{X}^{(i)}\,(i,i)$ for all $i \in [l]$, $i+1 \le j \le n$ (except for $\mathbf{R}^{LU}\,(i, j_i) = \mathbf{X}^{(i-1)}\,(i,i)\,/\mathbf{X}^{(i)}\,(i,i)$), and therefore $\left|\mathbf{R}^{LU}\,(i,j)\right| \le 1$.

Analogous to CPQR, the pivoting strategy of LUPP leads to a loose, exponential upper bound:

Lemma 2. *The column-wise LUPP in Equation* (18) *satisfies that*

$$\max_{i,j} \left| \left(\left(\mathbf{R}_1^{LU} \right)^{-1} \mathbf{R}_2^{LU} \right)_{ij} \right| \leq 2^{l-i},$$

where the upper bound is tight, for instance, when $\mathbf{R}_1^{LU}(i,j) = -1$ *for all* $i \in [l-1]$, $i+1 \leq j \leq l$ *and* $\mathbf{R}_2^{LU}(i,j) = 1$ *for all* $i \in [l]$, $j \in [n-l]$ *(i.e., a Kahan-type matrix ([33, 47])).*

Proof of Lemma 2. We start by observing the following recursive relations: for all $j = 1, \ldots, n-l$,

$$\left(\left(\mathbf{R}_1^{LU} \right)^{-1} \mathbf{R}_2^{LU} \right)_{lj} = \mathbf{R}_2^{LU}(l,j)$$

$$\left(\left(\mathbf{R}_1^{LU} \right)^{-1} \mathbf{R}_2^{LU} \right)_{ij} = \mathbf{R}_2^{LU}(i,j) - \sum_{\iota=i+1}^{l} \mathbf{R}_1^{LU}(i,\iota) \left(\left(\mathbf{R}_1^{LU} \right)^{-1} \mathbf{R}_2^{LU} \right)_{\iota,j} \quad \forall\, i = l-1, \ldots, 1,$$

given $\mathbf{R}_1^{LU}(i,i) = 1$. Then both the upper bound and the Kahan example in Lemma 2 follow from the fact that $\left| \mathbf{R}^{LU}(i,j) \right| \leq 1$ for all $i \in [l]$, $i \leq j \leq n$. ∎

In addition to the exponential worse-case scenario in Lemma 2, LUPP is also vulnerable to rank deficiency since it only views one row for each pivoting step (in contrast to CPQR which searches the entire active submatrix). The advantage of the LUPP type pivoting scheme is its superior empirical efficiency and parallelizability ([23, 35, 26, 50]). Fortunately, as with CPQR, adversarial inputs for LUPP are sensitive to perturbations (*e.g.*, flip the signs of random off-diagonal entries in $\mathbf{R}_1^{LU}$), and rarely encounted in practice.

LUPP can be further stabilized with randomization ([45, 46, 53]). In terms of the worse-case exponential entry-wise bound in Lemma 2, [53] investigated average-case growth factors of LUPP on random matrices drawn from a variety of distributions (*e.g.*, the Gaussian distribution, uniform distributions, Rademacher distribution, symmetry / Toeplitz matrices with Gaussian entries, and orthogonal matrices following Haar measure), and conjectured that the growth factor increases sublinearly with respect to the problem size in average cases.

Remark 4 (Conjectured in [53]). With randomized preprocessing like sketching, LUPP is robust to adversarial inputs in practice, with $\max_{i,j} \left| \left(\left(\mathbf{R}_1^{LU} \right)^{-1} \mathbf{R}_2^{LU} \right)_{ij} \right| = O(l)$ in average cases.

Some common alternatives to the partial pivoting for LU decompositions include (adaptive) cross approximations ([5, 57, 34]), and complete pivoting. Specifically, the complete pivoting is a more robust (*e.g.*, to rank deficiency) alternative to partial pivoting that searches the entire active submatrix, and permutes rows and column simultaneously. In spite of lacking theoretical guarantees for the plain complete pivoting, like for QR decompositions, there exists modified complete pivoting strategies for LU that come with better rank-revealing guarantees ([44, 41, 3, 11]), but higher computational cost as trade-off.

3. SUMMARY OF EXISTING ALGORITHMS

A vast assortment of algorithms for interpolative and CUR decompositions have been proposed and analyzed (*e.g.*, [18, 17, 20, 37, 6, 59, 13, 9, 7, 51, 2, 1, 58, 3, 49, 14, 11, 15]) in the past decades. From the skeleton selection perspective, these algorithms broadly fall in two categories:

(1) sampling based methods that draw matrix skeletons (directly, adaptively, or iteratively) from some proper distributions, and
(2) pivoting based methods that pick matrix skeletons greedily by constructing low-rank matrix decompositions with pivoting.

In this section, we discuss existing algorithms for matrix skeletonizations, with a focus on algorithms based on randomized linear embeddings and matrix decompositions with pivoting.

3.1. **Sampling based skeleton selection.** The idea of skeleton selection via sampling is closely related to various topics including graph sparsification ([4]) and volume sampling ([18]). Concerning volume sampling, [17, 2, 14, 15] discussed adaptive sampling strategies that lead to matrix skeletons with close-to-optimal error guarantees, as discussed in Section 2.2.2. [20, 37, 6, 19] pointed out the effect of matrix coherence, an inherited property of the matrix, on skeleton sampling, and proposed the idea of leverage score sampling for constructing CUR decompositions, as well as efficient estimations for the leverage scores. Beginning with uniform sampling, [13] provided extensive analysis on sampling based matrix approximations, and proposed an iterative sampling scheme for skeleton selection. Some more sophisticated variations of sampling based skeleton selection algorithms were proposed in [9, 7, 59] that combined varieties of sampling methods.

3.2. **Skeleton selection via deterministic pivoting.** Greedy algorithms based on column and row pivoting can also be used for matrix skeletonizations. For instance, with proper rank-revealing pivoting like the strong rank-revealing QR proposed in [27], a rank-k ($k < r$) column ID can be constructed with the first k column pivots

$$\mathbf{A}[\underset{n \times l}{\mathbf{\Pi}_{n,1}}, \underset{n \times (n-l)}{\mathbf{\Pi}_{n,2}}] = [\underset{m \times k}{\mathbf{Q}_{A,1}}, \underset{m \times (r-k)}{\mathbf{Q}_{A,2}}] \begin{bmatrix} \mathbf{R}_{A,11} & \mathbf{R}_{A,12} \\ \mathbf{0} & \mathbf{R}_{A,22} \end{bmatrix} \approx (\mathbf{A}\mathbf{\Pi}_{n,1}) \begin{bmatrix} \mathbf{I}_k, & \mathbf{R}_{A,11}^{-1}\mathbf{R}_{A,12} \end{bmatrix}$$

where $\mathbf{\Pi}_n = [\mathbf{\Pi}_{n,1}, \mathbf{\Pi}_{n,2}]$ is a permutation of columns; and $\mathbf{R}_{A,11}$ and $\mathbf{R}_{A,22}$ are non-singular and upper triangular. $\mathbf{C} = (\mathbf{A}\mathbf{\Pi}_{n,1})$ are the selected column skeletons that satisfies $\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\| = \|\mathbf{R}_{A,22}\| \lesssim \sqrt{k(n-k)}\|\mathbf{A} - \mathbf{A}_k\|$. As a more affordable alternative to the rank-revealing pivoting, the CPQR discussed in Section 2.5 also works well for skeleton selection in practice ([58]), despite the weaker theoretical guarantee (*i.e.*, $\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\| \lesssim 2^k \|\mathbf{A} - \mathbf{A}_k\|$) due to the known existence of adversarial inputs. In addition to the QR based pivoting schemes, [44, 41, 1, 3, 49, 11] proposed various (randomized) LU based pivoting algorithms with rank-revealing guarantees that can be leveraged for greedy matrix skeleton selection (as discussed in section 2.5). Meanwhile, relaxing the rank-revealing requirements on pivoting schemes by pre-processing $\mathbf{A}$, [51, 21] proposed the DEIM skeleton selection algorithm that apply LUPP on the leading singular vectors of $\mathbf{A}$.

3.3. **Randomized pivoting based skeleton selection.** In comparison to the sampling based skeleton selection, the deterministic pivoting based skeleton selection methods suffer from two major drawbacks. First, pivoting is usually unaffordable for large-scale problems in common modern applications. Second, classical pivoting schemes like the CPQR and LUPP are vulnerable to antagonistic inputs. Fortunately, randomized pre-processing with sketching provides remedies both problems:
(1) Faster execution speed is attained by executing classical pivoting schemes on a sketch $\mathbf{X} = \mathbf{\Gamma}\mathbf{A} \in \mathbb{R}^{l \times n}$, for some randomized embedding $\mathbf{\Gamma}$, instead on $\mathbf{A}$ directly.
(2) With randomization, classical pivoting schemes like the CPQR and LUPP are robust to adversarial inputs in practice (Remark 4, [53]).
Algorithm 1 describes a general framework for randomized pivoting based skeleton selection. Grounding this framework down with different combinations of row space approximators and pivoting schemes, [58] took $\mathbf{X} = \mathbf{\Gamma}\mathbf{A}$ as a row sketch, and applied CPQR to $\mathbf{X}$ for column skeleton selection. Alternatively, the DEIM skeleton selection algorithm proposed in [51] can be accelerated by taking $\mathbf{X}$ as an approximation of the leading-l right singular vectors of $\mathbf{A}$ (Equation (16)), where LUPP is applied for skeleton selection.

First, we recall from Lemma 1 that when taking $\mathbf{X}$ as a row sketch, with Gaussian embeddings, $\mathbf{\Gamma}$ and $\mathbf{X} = \mathbf{\Gamma}\mathbf{A}$ are both full row rank with probability 1. Moreover, when taking $\mathbf{X}$ as an approximation of right singular vectors constructed with a row space approximator consisting of l linearly independent rows, $\mathbf{X}$ also admits full row rank.

Second, when both column and row skeletons are inquired, Algorithm 1 selects the column skeletons first with randomized pivoting, and subsequently identifies the row skeletons by pivoting on the selected columns. With $\mathbf{X}$ being full row rank (almost surely when $\mathbf{\Gamma}$ is a Gaussian embedding), the column skeletons $\mathbf{C}$ are

---

Algorithm 1 Randomized pivoting based skeleton selection: a general framework

---

Input: $\mathbf{A} \in \mathbb{R}^{m \times n}$ of rank r, rank $l \leq r$ (typically $l \ll \min(m, n)$).
Output: Column and/or row skeleton indices, $J_s \subset [n]$ and/or $I_s \subset [m]$, $|J_s| = |I_s| = l$.
  1: Draw an oblivious ℓ_2-embedding $\mathbf{\Gamma} \in \mathbb{R}^{l \times m}$.
  2: Construct a row space approximator $\mathbf{X} \in R^{l \times n}$ via sketching with $\mathbf{\Gamma}$.
     e.g., $\mathbf{X}$ can be 1) a row sketch or 2) approximations of right singular vectors.
  3: Perform column-wise pivoting on $\mathbf{X}$. Let J_s index the l column pivots.
  4: Perform row-wise pivoting on $\mathbf{C} = \mathbf{A}(:, J_s)$. Let I_s index the l row pivots.

---

linearly independent. Therefore, the row-wise skeletonization of $\mathbf{C}$ is exact, without introducing additional error. That is, the two-sided ID constructed by Algorithm 1 is equal to the associated column ID in exact arithmetic, $\widehat{\mathbf{A}}_{I_s, J_s} = \widehat{\mathbf{A}}_{*, J_s}$.

4. A SIMPLE BUT EFFECTIVE MODIFICATION: LUPP ON SKETCHES

Inspired by the idea of pivoting on sketches ([58]) and the remarkably competitive performance of LUPP when applied to leading singular vectors ([51]), we propose a simple but effective modification – applying LUPP directly to a sketch of $\mathbf{A}$. In terms of the general framework in Algorithm 1, this corresponds to taking $\mathbf{X}$ as a row sketch of $\mathbf{A}$, and then selecting skeletons via LUPP on $\mathbf{X}$ and $\mathbf{C}$, as summarized in Algorithm 2.

Algorithm 2 Randomized LUPP skeleton selection

Input: $\mathbf{A} \in \mathbb{R}^{m \times n}$ of rank r, rank $l \leq r$ (typically $l \ll \min(m, n)$).
Output: Column and/or row skeleton indices, $J_s \subset [n]$ and/or $I_s \subset [m]$, $|J_s| = |I_s| = l$.
  1: Draw an oblivious ℓ_2-embedding $\mathbf{\Gamma} \in \mathbb{R}^{l \times m}$.
  2: Construct a row sketch $\mathbf{X} = \mathbf{\Gamma} \mathbf{A}$.
  3: Perform column-wise LUPP on $\mathbf{X}$. Let J_s index the l column pivots.
  4: Perform row-wise LUPP on $\mathbf{C} = \mathbf{A}(:, J_s)$. Let I_s index the l row pivots.

Comparing to pivoting with CPQR ([58]), Algorithm 2 with LUPP is empirically faster, as discussed in Section 2.5, and illustrated in Figure 2. Meanwhile, assuming that the true SVD of $\mathbf{A}$ is unavailable, in comparison to pivoting on the approximated leading singular vectors ([51]) from Equation (16), Algorithm 2 saves the effort of constructing randomized SVD which takes $O\left(\operatorname{nnz}(\mathbf{A})l + (m + n)l^2\right)$ additional operations. Additionally, with randomization, the stability of LUPP conjectured in [53] (Remark 4) applies, and Algorithm 2 effectively circumvents the potential vulnerability of LUPP to adversarial inputs in practice. A formal error analysis of Algorithm 1 in general reflects these points:

Theorem 1 (Column skeleton selection by pivoting on a row space approximator). *Given a row space approximator* $\mathbf{X} \in \mathbb{R}^{l \times n}$ *($l \leq r$) of* $\mathbf{A}$ *that adimits full row rank, let* $\mathbf{\Pi}_n \in \mathbb{R}^{n \times n}$ *be the resulted permutation after applying some proper column pivoting scheme on* $\mathbf{X}$ *that identifies* l *linearly independent column pivots: for the* $(l, n - l)$ *column-wise partition* $\mathbf{X}\mathbf{\Pi}_n = \mathbf{X}[\mathbf{\Pi}_{n,1}, \mathbf{\Pi}_{n,2}] = [\mathbf{X}_1, \mathbf{X}_2]$, *the first* l *column pivots* $\mathbf{X}_1 = \mathbf{X}\mathbf{\Pi}_{n,1} \in \mathbb{R}^{l \times l}$ *admits full column rank. Moreover, the rank-l column ID* $\widehat{\mathbf{A}}_{*, J_s} = \mathbf{C}\mathbf{C}^\dagger \mathbf{A}$, *with linearly independent column skeletons* $\mathbf{C} = \mathbf{A}\mathbf{\Pi}_{n,1}$, *satisfies that*

$$(19) \qquad \left\| \mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A} \right\| \leq \eta \left\| \mathbf{A} - \mathbf{A}\mathbf{X}^\dagger \mathbf{X} \right\|,$$

where $\eta \leq \sqrt{1 + \left\| \mathbf{X}_1^\dagger \mathbf{X}_2 \right\|_2^2}$, *and* $\|\cdot\|$ *represents the spectral or Frobenius norm.*

Theorem 1 states that when selecting column skeletons by pivoting on a row space approximator, the low-rank approximation error of the resulting column ID is upper bounded by that of the associated row space approximator up to a factor $\eta > 1$ that can be computed a posteriori effciently in $O\left(l^2(n - l)\right)$ operations.

Equation (19) essentially decouples the error from the row space approximation with $\mathbf{X}$ ($\left\|\mathbf{A} - \mathbf{A}\mathbf{X}^\dagger\mathbf{X}\right\|$ corresponding to Line 1 and 2 of Algorithm 1) and that from the skeleton selection by pivoting on $\mathbf{X}$ (η corresponding to Line 3 and 4 of Algorithm 1). Now we ground Theorem 1 with different choices of row space approximation and pivoting strategies:

(1) With Algorithm 2, $\left\|\mathbf{A} - \mathbf{A}\mathbf{X}^\dagger\mathbf{X}\right\|$ is the randomized rangefinder error (Equation (13), [28] Section 10), and $\eta \leq \sqrt{1 + \left\|\left(\mathbf{R}_1^{LU}\right)^{-1}\mathbf{R}_2^{LU}\right\|_2^2}$ (recall Equation (18)). Although in the worse case scenario (where the entry-wise upper bound in Lemma 2 is tight), $\eta = \Theta\left(2^l\sqrt{n-l}\right)$, with a randomized row sketch $\mathbf{X}$, assuming the stability of LUPP conjectured in [53] holds (Remark 4), $\eta = O\left(l^{3/2}\sqrt{n-l}\right)$.

(2) Skeleton selection with CPQR on row sketches (*i.e.*, randomized CPQR proposed by [58]) shares the same error bound as Algorithm 2 (*i.e.*, analogous arguments hold for $\left\|\left(\mathbf{R}_1^{QR}\right)^{-1}\mathbf{R}_2^{QR}\right\|_2^2$).

(3) When applying LUPP on the true leading singular vectors (*i.e.*, DEIM proposed in [51], assuming that the true SVD is available), $\left\|\mathbf{A} - \mathbf{A}\mathbf{X}^\dagger\mathbf{X}\right\| = \left\|\mathbf{A} - \mathbf{A}_l\right\|$, but without randomization, LUPP is vulnerable to adversarial inputs which can lead to $\eta = \Theta\left(2^l\sqrt{n-l}\right)$ in the worse case.

(4) When applying LUPP on approximations of leading singular vectors (constructed via Equation (16), *i.e.*, randomized DEIM suggested by [51]), $\left\|\mathbf{A} - \mathbf{A}\mathbf{X}^\dagger\mathbf{X}\right\|$ corresponds to the randomized rangefinder error with power itertions ([28] Corollary 10.10), while η follows the analogous analysis as for Algorithm 2.

Proof of Theorem 1. We start by defining two oblique projectors

$$\underset{n\times n}{\mathbf{P}_X} \triangleq \mathbf{\Pi}_{n,1}\left(\mathbf{X}\mathbf{\Pi}_{n,1}\right)^\dagger\mathbf{X}, \quad \underset{n\times n}{\mathbf{P}_C} \triangleq \mathbf{\Pi}_{n,1}\left(\mathbf{C}^\top\mathbf{C}\right)^\dagger\mathbf{C}^\top\mathbf{A},$$

and observe that, since $\mathbf{C}$ consists of linearly independent columns, $\left(\mathbf{C}^\top\mathbf{C}\right)^\dagger\mathbf{C}^\top\mathbf{A}\mathbf{\Pi}_{n,1} = \mathbf{I}_l$, and

$$\mathbf{P}_C\mathbf{P}_X = \mathbf{\Pi}_{n,1}\left(\mathbf{C}^\top\mathbf{C}\right)^\dagger\mathbf{C}^\top\mathbf{A}\mathbf{\Pi}_{n,1}\left(\mathbf{X}\mathbf{\Pi}_{n,1}\right)^\dagger\mathbf{X} = \mathbf{P}_X.$$

With $\mathbf{P}_C$, we can express the column ID as

$$\widehat{\mathbf{A}}_{*,J_s} = \mathbf{C}\mathbf{C}^\dagger\mathbf{A} = \mathbf{A}\left(\mathbf{\Pi}_{n,1}\left(\mathbf{C}^\top\mathbf{C}\right)^\dagger\mathbf{C}^\top\mathbf{A}\right) = \mathbf{A}\mathbf{P}_C,$$

Therefore, the low-rank approximation error of $\widehat{\mathbf{A}}_{*,J_s}$ satisfies

$$\begin{aligned}
\left\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\right\| &= \left\|\mathbf{A}\left(\mathbf{I} - \mathbf{P}_C\right)\right\| \\
&= \left\|\mathbf{A}\left(\mathbf{I}_n - \mathbf{P}_C\right)\left(\mathbf{I}_n - \mathbf{P}_X\right)\right\| \\
&= \left\|\left(\mathbf{I}_m - \mathbf{C}\mathbf{C}^\dagger\right)\mathbf{A}\left(\mathbf{I}_n - \mathbf{P}_X\right)\right\| \\
&\leq \left\|\mathbf{I}_m - \mathbf{C}\mathbf{C}^\dagger\right\|_2\left\|\mathbf{A}\left(\mathbf{I}_n - \mathbf{P}_X\right)\right\|,
\end{aligned}$$

where $\left\|\mathbf{I}_m - \mathbf{C}\mathbf{C}^\dagger\right\|_2 = 1$, and since $\mathbf{X}\mathbf{P}_X = \mathbf{X}_1\mathbf{X}_1^\dagger\mathbf{X} = \mathbf{X}$ with $\mathbf{X}_1$ being full-rank,

$$\left\|\mathbf{A}\left(\mathbf{I}_n - \mathbf{P}_X\right)\right\| = \left\|\mathbf{A}\left(\mathbf{I}_n - \mathbf{X}^\dagger\mathbf{X}\right)\left(\mathbf{I}_n - \mathbf{P}_X\right)\right\| = \left\|\mathbf{I}_n - \mathbf{P}_X\right\|_2\left\|\mathbf{A}\left(\mathbf{I}_n - \mathbf{X}^\dagger\mathbf{X}\right)\right\|.$$

As a result, we have

$$\left\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\right\| \leq \left\|\mathbf{I}_n - \mathbf{P}_X\right\|_2\left\|\mathbf{A}\left(\mathbf{I}_n - \mathbf{X}^\dagger\mathbf{X}\right)\right\|,$$

and it is sufficient to show that $\eta \triangleq \|\mathbf{I}_n - \mathbf{P}_X\|_2 \leq \sqrt{1 + \left\|\mathbf{X}_1^\dagger \mathbf{X}_2\right\|_2^2}$. Indeed,

$$\mathbf{I}_n - \mathbf{P}_X = \mathbf{\Pi}_n^\top \mathbf{\Pi}_n - \mathbf{\Pi}_n^\top \mathbf{\Pi}_{n,1} \left(\mathbf{X}\mathbf{\Pi}_{n,1}\right)^\dagger \mathbf{X}\mathbf{\Pi}_n = \mathbf{I}_n - \begin{bmatrix} \mathbf{I}_l \\ \mathbf{0} \end{bmatrix} \mathbf{X}_1^\dagger \begin{bmatrix} \mathbf{X}_1 \mathbf{X}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{0} & -\mathbf{X}_1^\dagger \mathbf{X}_2 \\ \mathbf{0} & \mathbf{I}_{n-l} \end{bmatrix}$$

such that $\eta = \left\|\left[-\mathbf{X}_1^\dagger \mathbf{X}_2; \mathbf{I}_{n-l}\right]\right\|_2 \leq \sqrt{1 + \left\|\mathbf{X}_1^\dagger \mathbf{X}_2\right\|_2^2}$. ∎

Here, the proof of Theorem 1 is reminiscent of [51], while it generalizes the result for fixed right leading singular vectors to any proper row space approximators of $\mathbf{A}$ (*e.g.*, $\widehat{\mathbf{V}}_A$ in Equation (16), or simply a row sketch). The generalization of Theorem 1 leads to a factor η that is efficiently computable a posteriori, which can serve as an empirical replacement of the exponential upper bound induced by the scarce adversarial inputs.

In addition to empirical efficiency and robustness discussed above, Algorithm 2 has another potential advantage: the skeleton selection algorithm can be easily adapted to the streaming setting. The streaming setting considers $\mathbf{A}$ as a data stream that can only be accessed as a sequence of snapshots. Each snapshot of $\mathbf{A}$ can be viewed only once, and the storage of the entire matrix $\mathbf{A}$ is infeasible ([55, 56, 38]).

Algorithm 3 Streaming LUPP/CPQR skeleton selection

Input: $\mathbf{A} \in \mathbb{R}^{m \times n}$ of rank r, rank $l \leq r$ (typically $l \ll \min(m, n)$).
Output: Column and/or row skeleton indices, $J_s \subset [n]$ and/or $I_s \subset [m]$, $|J_s| = |I_s| = l$.
  1: Draw independent oblivious ℓ_2-embeddings $\mathbf{\Gamma} \in \mathbb{R}^{l \times m}$ and $\mathbf{\Omega} \in \mathbb{R}^{l \times n}$.
  2: Construct row and column sketches, $\mathbf{X} = \mathbf{\Gamma}\mathbf{A}$ and $\mathbf{Y} = \mathbf{A}\mathbf{\Omega}^\top$, in a single pass through $\mathbf{A}$.
  3: Perform column-wise pivoting (LUPP/CPQR) on $\mathbf{X}$. Let J_s index the l column pivots.
  4: Perform row-wise pivoting (LUPP/CPQR) on $\mathbf{Y}$. Let I_s index the l row pivots.

Remark 5. When only the column and/or row skeleton *indices* are required (and not the explicit construction of the corresponding interpolative or CUR decomposition), Algorithm 2 can be adapted to the streaming setting (as shown in Algorithm 3) by sketching both sides of $\mathbf{A}$ independently in a single pass, and pivoting on the resulting column and row sketches. Moreover, with the column and row skeletons J_s and I_s from Algorithm 3, Theorem 1 and its row-wise analog together, along with Equation (9), imply that

$$\left\|\mathbf{A} - \widetilde{\mathbf{A}}_{I_s, J_s}\right\| \leq \eta_X \left\|\mathbf{A} - \mathbf{A}\mathbf{X}^\dagger\mathbf{X}\right\| + \eta_Y \left\|\mathbf{A} - \mathbf{Y}\mathbf{Y}^\dagger\mathbf{A}\right\|,$$

where η_X and η_Y are small in practice with pivoting on randomized sketches, and have efficiently a posteriori computable upper bounds given $\mathbf{X}$ and $\mathbf{Y}$, as discussed previously. $\left\|\mathbf{A} - \mathbf{A}\mathbf{X}^\dagger\mathbf{X}\right\|_F^2$ and $\left\|\mathbf{A} - \mathbf{Y}\mathbf{Y}^\dagger\mathbf{A}\right\|$ are the randomized rangefinder errors with well-established upper bounds ([28] Section 10).

We point out that, although the column and row skeleton selection can be conducted in streaming fashion, the explicit stable construction of ID or CUR requires two additional passes through $\mathbf{A}$: one pass for retrieving the skeletons $\mathbf{C}$ and/or $\mathbf{R}$, and the other pass to construct $\mathbf{C}^\dagger \mathbf{A} \mathbf{R}^\dagger$ for CUR, or $\mathbf{C}^\dagger \mathbf{A}$, $\mathbf{A} \mathbf{R}^\dagger$ for IDs. In practice, for efficient estimations of the ID or CUR when revisiting $\mathbf{A}$ is expensive, it is possible to circumvent the second pass through $\mathbf{A}$ with compromise on accuracy and stability, albeit the inevitability of the first pass for skeleton retrieval.

Precisely for the ID, $\mathbf{C}^\dagger \mathbf{A}$ (in Equation (5)) or $\mathbf{A} \mathbf{R}^\dagger$ (in Equation (6)) can be estimated without revisiting $\mathbf{A}$ leveraging the associated row and column sketches:

$$\mathbf{C}^\dagger \mathbf{A} \approx \mathbf{X}_1^\dagger \mathbf{X}, \quad \mathbf{A} \mathbf{R}^\dagger \approx \mathbf{Y} \mathbf{Y}_1^\dagger,$$

where $\mathbf{X}_1 = \mathbf{X}(:, J_s)$ and $\mathbf{Y}_1 = \mathbf{Y}(I_s, :)$ are the l column and row pivots in $\mathbf{X}$ and $\mathbf{Y}$, respectively. Meanwhile for the CUR, by retrieving the skeletons $\mathbf{S} = \mathbf{A}(I_s, J_s)$, $\mathbf{C} = \mathbf{A}(:, J_s)$ and $\mathbf{R} = \mathbf{A}(I_s, :)$, we can construct a CUR decomposition $\mathbf{C}\mathbf{S}^{-1}\mathbf{R}$, despite the compromise on both accuracy and stability.

5. NUMERICAL EXPERIMENTS

In this section, we study the empirical performance of various randomized skeleton selection algorithms. Starting with the randomized pivoting based algorithms, we investigate the efficiency of two major components of Algorithm 1: (1) the sketching step for row space approximator construction, and (2) the pivoting step for greedy skeleton selection. Then we explore the suboptimality (in terms of low-rank approximation errors of the resulting CUR decompositions $\left\| \mathbf{A} - \widetilde{\mathbf{A}}_{I_s,J_s} \right\|$), as well as the efficiency (in terms of empirical run time), of different randomized skeleton selection algorithms.

We conduct all the experiments, except for those in Figure 1 on the efficiency of sketching, in MATLAB R2020a. In the implementation, the computationally dominant processes, including the sketching, LUPP, CPQR, and SVD, are performed by the MATLAB built-in functions. The experiments in Figure 1 are conducted in Julia Version 1.5.3 with the JuliaMatrices/LowRankApprox.jl package ([29]).

5.1. **Computational speeds of different embeddings.**

Here, we compare the empirical efficiency of constructing sketches with some common randomized embeddings listed in Table 1. We consider applying an embedding $\mathbf{\Gamma}$ of size $l \times m$ to a matrix $\mathbf{A}$ of size $m \times n$, which can be interpreted as embedding n vectors in an ambient space $\mathbb{R}^m$ to a lower dimensional space $\mathbb{R}^l$. We scale the experiments with respect to the ambient dimension m, at several different embedding dimension l, with a fixed number of repetitions $n = 1000$. Figure 1 suggests that, with proper implementation, the sparse sign matrices are more efficient than the Gaussian embeddings and the SRTTs, especially for large scale problems. The SRTTs outperform Gaussian embeddings in terms of efficiency, and such advantage can be amplified as l increases. These observations align with the asymptotic complexity in Table 1. While we also observe that, with MATLAB default implementation, the Gaussian embeddings usually enjoy matching efficiency as sparse sign matrices for moderate size problems, and are more efficient than SRTTs.
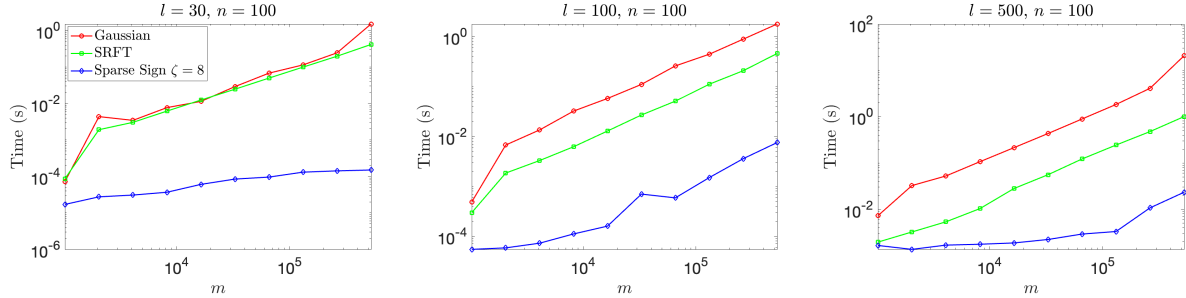

FIGURE 1. Run time of applying different randomized embeddings $\mathbf{\Gamma} \in \mathbb{R}^{l \times m}$ to some dense matrices of size $m \times n$, scaled with respect to the ambience dimension m, with different embedding dimension l, and a fixed number of embeddings $n = 100$.

5.2. **Computational speeds of different pivoting schemes.**

Given a sketch of $\mathbf{A}$, we isolate different pivoting schemes in Algorithm 1, and compare their run time as the problem size n increases. Specifically, the LUPP and CPQR pivot directly on the given row sketch $\mathbf{X} = \mathbf{\Gamma} \mathbf{A} \in \mathbb{R}^{l \times n}$, while the DEIM involves one additional power iteration with orthogonalization (Equation (15)) before applying the LUPP (*i.e.*, with a given column sketch $\mathbf{Y} = \mathbf{A}\mathbf{\Omega} \in \mathbb{R}^{m \times l}$, for DEIM, we first construct an orthonormal basis $\mathbf{Q}_Y \in \mathbb{R}^{m \times l}$ for columns of the sketch, and then we compute the reduced SVD for $\mathbf{Q}_Y^\top \mathbf{A} \in l \times n$, and finally we column-wisely pivot on the resulting right singular vectors of size $l \times n$). In Figure 2, we observe a considerable run time advantage of the LUPP over the CPQR and DEIM, especially when l is large. (Additionally, we see that DEIM slightly outperforms CPQR, which is perhaps surprising, given the substantially larger number of flops required by DEIM.)
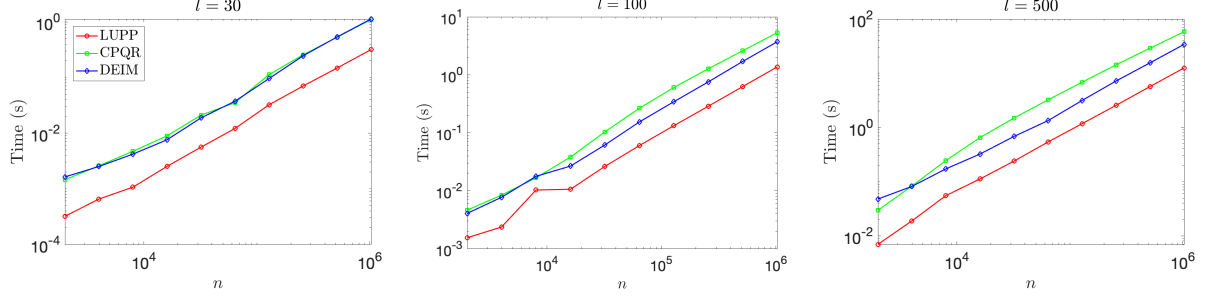
FIGURE 2. Run time of different pivoting schemes, scaled with respect to the problem size n, with different embedding dimension l.

5.3. **Randomized skeleton selection algorithms: accuracy and efficiency.** As we move from measuring speed to measuring the precision of revealing the numerical rank of a matrix, the choice of test matrix becomes important. We consider four different classes of test matrices, including some synthetic random matrices with different spectral patterns, as well as some empirical datasets, as summarized below:

(1) `large`: a full-rank $4,282 \times 8,617$ sparse matrix with $20,635$ nonzero entries from the SuiteSparse matrix collection, generated by a linear programming problem sequence [40].

(2) `YaleFace64x64`: a full-rank 165×4096 dense matrix, consisting of 165 face images each of size 64×64. The flatten image vectors are centered and normalized such that the average image vector is zero, and the entries are bounded within $[-1, 1]$.

(3) `MNIST` training set consists of $60,000$ images of hand-written digits from 0 to 9. Each image is of size 28×28. The images are flatten and normalized to form a full-rank matrix of size $N \times d$ where N is the number of images and $d = 784$ is the size of the flatten images, with entries bounded in $[0, 1]$. The nonzero entries take approximately 20% of the matrix for both the training and the testing sets.

(4) Random *sparse non-negative (SNN)* matrices are synthetic random sparse matrices used in [58, 51] for testing skeleton selection algorithms. Given $s_1 \geq \cdots \geq s_r > 0$, a random SNN matrix $\mathbf{A}$ of size $m \times n$ takes the form,

$$(20) \qquad \mathbf{A} = \mathrm{SNN}\Big(\{s_i\}_{i=1}^r;\ m, n\Big) := \sum_{i=1}^{r} s_i \mathbf{x}_i \mathbf{y}_i^T$$

where $\mathbf{x}_i \in \mathbb{R}^m$, $\mathbf{y}_i \in \mathbb{R}^n$, $i \in [r]$ are random sparse vectors with non-negative entries. In the experiments, we use two random SNN matrices of distinct sizes:

(i) `SNN1e3` is a 1000×1000 SNN matrix with $r = 1000$, $s_i = \frac{2}{i}$ for $i = 1, \dots, 100$, and $s_i = \frac{1}{i}$ for $i = 101, \dots, 1000$;

(ii) `SNN1e6` is a $10^6 \times 10^6$ SNN matrix with $r = 400$, $s_i = \frac{2}{i}$ for $i = 1, \dots, 100$, and $s_i = \frac{1}{i}$ for $i = 101, \dots, 400$.

Scaled with respect to the approximation ranks k, we compare the accuracy and efficiency of the following randomized CUR algorithms:

(1) Rand-LUPP (and Rand-LUPP-1piter): Algorithm 1 with $\mathbf{X} = \mathbf{\Gamma A}$ being a row sketch (or with one plain power iteration as in Equation (14)), and pivoting with LUPP;

(2) Rand-CPQR (and Rand-CPQR-1piter): Algorithm 1 with $\mathbf{X} = \mathbf{\Gamma A}$ being a row sketch (or with one power iteration as in Equation (14)), and pivoting with CPQR ([58]);

(3) RSVD-DEIM: Algorithm 1 with $\mathbf{X}$ being an approximation of leading-k right singular vectors (Equation (16)), and pivoting with LUPP ([51]);

(4) RSVD-LS: Skeleton sampling based on approximated leverage scores ([37]) from a rank-k SVD approximation (Equation (16));

(5) SRCUR: Spectrum-revealing CUR decomposition proposed in [11].

The asymptotic complexities of the first three randomized pivoting based skeleton selection algorithms based on Algorithm 1 are summarized in Table 2.

Algorithm	Row space approximator construction (Line 1,2)	Pivoting (Line 3)
Rand-LUPP	$O(T_s(l, \mathbf{A}))$	$O(nl^2)$
Rand-LUPP-1piter	$O(T_s(l, \mathbf{A}) + \text{nnz}(\mathbf{A})l)$	$O(nl^2)$
Rand-CPQR	$O(T_s(l, \mathbf{A}))$	$O(nl^2)$
Rand-CPQR-1piter	$O(T_s(l, \mathbf{A}) + \text{nnz}(\mathbf{A})l)$	$O(nl^2)$
RSVD-DEIM	$O\left(T_s(l, \mathbf{A}) + (m+n)l^2 + \text{nnz}(\mathbf{A})l\right)$	$O(nl^2)$

TABLE 2. Asymptotic complexities of various randomized pivoting based skeleton selection algorithms based on Algorithm 1.

For consistency, we use Gaussian embeddings for sketching throughout the experiments. With the selected column and row skeletons, we leverage the stable construction in Equation (10) to form the corresponding CUR decompositions $\widetilde{\mathbf{A}}_{I_s, J_s}$. Although oversampling (*i.e.*, $l > k$) is necessary for multiplicative error bounds with respect to the optimal rank-k approximation error (Equation (13), Theorem 1), since oversampling can be interpreted as a shift of curves along the axis of the approximation rank, for the comparison purpose, we simply treat $l = k$, and compare the rank-k approximation errors of the CUR decompositions against the optimal rank-k approximation error $\|\mathbf{A} - \mathbf{A}_k\|$.
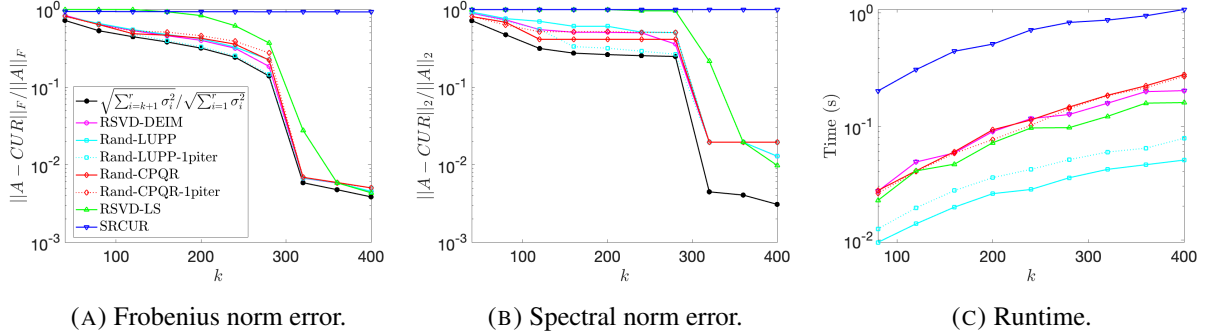

(A) Frobenius norm error.   (B) Spectral norm error.   (C) Runtime.

FIGURE 3. Relative error and run time of randomized skeleton selection on the `large` data set.

(A) Frobenius norm error.   (B) Spectral norm error.   (C) Runtime.
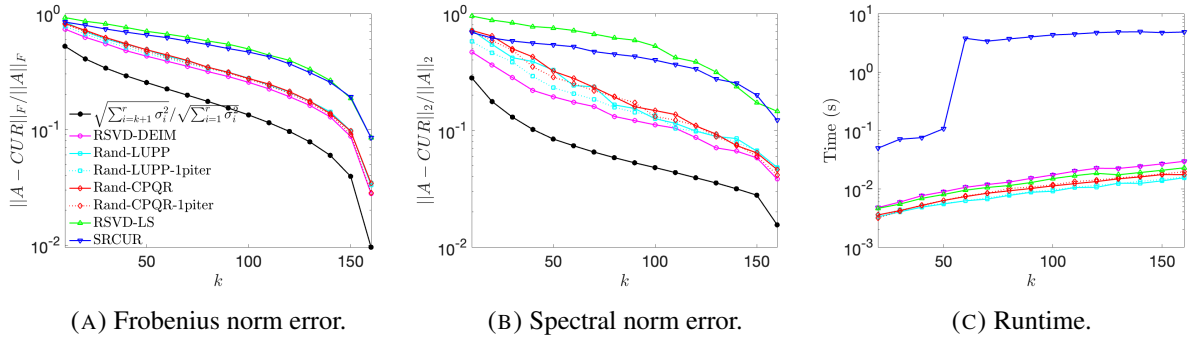
FIGURE 4. Relative error and run time of randomized skeleton selection on the `YaleFace64x64` data set.

(A) Frobenius norm error.   (B) Spectral norm error.   (C) Runtime.
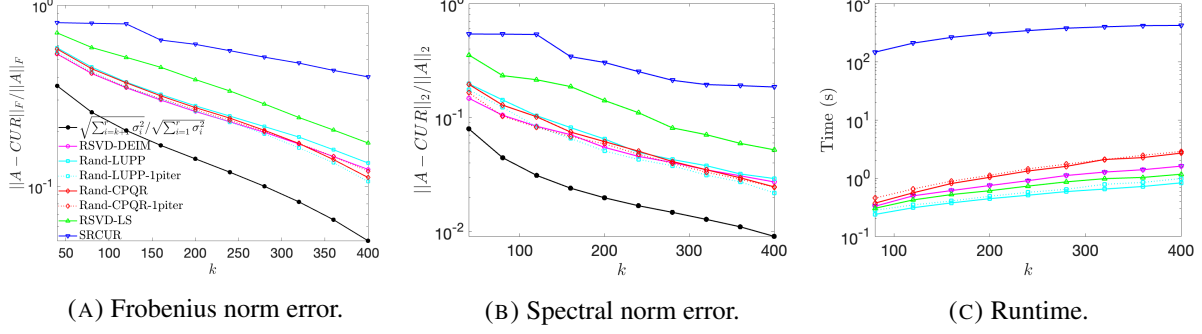
FIGURE 5. Relative error and run time of randomized skeleton selection on the training set of MNIST.

(A) Frobenius norm error.   (B) Spectral norm error.   (C) Runtime.
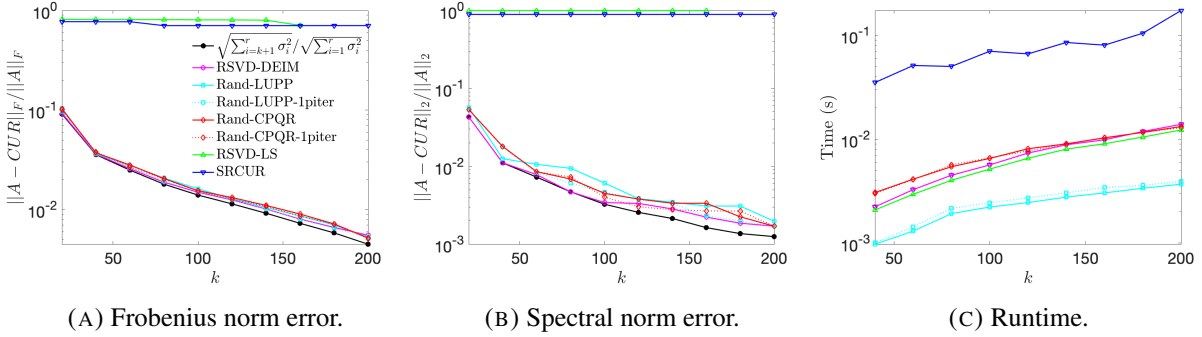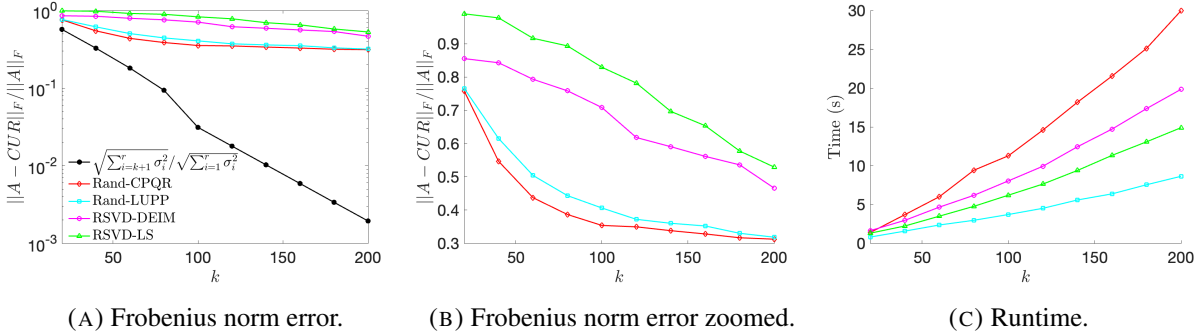
FIGURE 6. Relative error and run time of randomized skeleton selection on a 1000×1000 sparse non-negative random matrix, `SNN1e3`.

(A) Frobenius norm error.   (B) Frobenius norm error zoomed.   (C) Runtime.

FIGURE 7. Relative error and run time of randomized skeleton selection on a $10^6 \times 10^6$ sparse non-negative random matrix, `SNN1e6`.

From Figure 3-6, we observe that the randomized pivoting based skeleton selection algorithms that fall in Algorithm 1 (*i.e.*, Rand-LUPP, Rand-CPQR, and RSVD-DEIM) share the similar approximation accuracy, which is considerably lower than the RSVD-LS and SRCUR. From the efficiency perspective, Rand-LUPP provides the most competitive run time among all the algorithms, especially when **A** is sparse. Meanwhile, we observe that, for both Rand-CPQR and Rand-LUPP, constructing the sketches with one plain power iteration (*i.e.*, with Equation (14)) can observably improve the accuracy, without sacrificing the efficiency significantly (*e.g.*, in comparison to the randomized DEIM which involves one power iteration with orthogonalization as in Equation (15)). In Figure 7, the similar performance is also observed on a synthetic

large-scale problem, `SNN1e6`, where the matrix is only accessible as a fast matrix-vector multiplication (matvec) oracle such that each matvec takes $o(mn)$ (i.e., $O((m+n)r)$ in our construction) operations.

REFERENCES

[1] AIZENBUD, Y., SHABAT, G., AND AVERBUCH, A. Randomized lu decomposition using sparse projections. *Computers & Mathematics with Applications 72*, 9 (2016), 2525–2534.

[2] ANDERSON, D., DU, S., MAHONEY, M., MELGAARD, C., WU, K., AND GU, M. Spectral Gap Error Bounds for Improving CUR Matrix Decomposition and the Nyström Method. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics* (San Diego, California, USA, 09–12 May 2015), G. Lebanon and S. V. N. Vishwanathan, Eds., vol. 38 of *Proceedings of Machine Learning Research*, PMLR, pp. 19–27.

[3] ANDERSON, D., AND GU, M. An efficient, sparsity-preserving, online algorithm for low-rank approximation. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (2017), ICML'17, JMLR.org, p. 156–165.

[4] BATSON, J. D., SPIELMAN, D. A., AND SRIVASTAVA, N. Twice-ramanujan sparsifiers. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 2009), STOC '09, Association for Computing Machinery, p. 255–262.

[5] BEBENDORF, M. Approximation of boundary element matrices. *Numerische Mathematik 86* (2000), 565–589.

[6] BIEN, J., XU, Y., AND MAHONEY, M. Cur from a sparse optimization viewpoint. *Annual Advances in Neural Information Processing Systems 24: Proceedings of the 2010 Conference* (11 2010).

[7] BOUTSIDIS, C., DRINEAS, P., AND MAGDON-ISMAIL, M. Near optimal column-based matrix reconstruction. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science* (2011), pp. 305–314.

[8] BOUTSIDIS, C., AND GITTENS, A. Improved matrix algorithms via the subsampled randomized hadamard transform. *SIAM Journal on Matrix Analysis and Applications 34*, 3 (2013), 1301–1340.

[9] BOUTSIDIS, C., AND WOODRUFF, D. P. Optimal cur matrix decompositions. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 2014), STOC '14, Association for Computing Machinery, p. 353–362.

[10] CHAN, T. F. Rank revealing qr factorizations. *Linear Algebra and its Applications 88-89* (1987), 67–82.

[11] CHEN, C., GU, M., ZHANG, Z., ZHANG, W., AND YU, Y. Efficient spectrum-revealing cur matrix decomposition. In *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics* (26–28 Aug 2020), S. Chiappa and R. Calandra, Eds., vol. 108 of *Proceedings of Machine Learning Research*, PMLR, pp. 766–775.

[12] CLARKSON, K. L., AND WOODRUFF, D. P. Low-rank approximation and regression in input sparsity time. *J. ACM 63*, 6 (Jan. 2017).

[13] COHEN, M. B., LEE, Y. T., MUSCO, C., MUSCO, C., PENG, R., AND SIDFORD, A. Uniform sampling for matrix approximation. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science* (New York, NY, USA, 2015), ITCS '15, Association for Computing Machinery, p. 181–190.

[14] CORTINOVIS, A., AND KRESSNER, D. Low-rank approximation in the frobenius norm by column and row subset selection, 2019.

[15] DEREZINSKI, M., KHANNA, R., AND MAHONEY, M. W. Improved guarantees and a multiple-descent curve for the column subset selection problem and the nyström method. *ArXiv abs/2002.09073* (2020).

[16] DEREZIŃSKI, M., AND MAHONEY, M. Determinantal point processes in randomized numerical linear algebra. *Notices of the American Mathematical Society 68* (01 2021), 1.

[17] DESHPANDE, A., AND RADEMACHER, L. Efficient volume sampling for row/column subset selection. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science* (2010), pp. 329–338.

[18] DESHPANDE, A., RADEMACHER, L., VEMPALA, S., AND WANG, G. Matrix approximation and projective clustering via volume sampling. *Theory of Computing 2*, 12 (2006), 225–247.

[19] DRINEAS, P., MAGDON-ISMAIL, M., MAHONEY, M. W., AND WOODRUFF, D. P. Fast approximation of matrix coherence and statistical leverage. *J. Mach. Learn. Res. 13*, 1 (Dec. 2012), 3475–3506.

[20] DRINEAS, P., MAHONEY, M. W., AND MUTHUKRISHNAN, S. Relative-error cur matrix decompositions. *SIAM Journal on Matrix Analysis and Applications 30*, 2 (2008), 844–881.

[21] DRMAČ, Z., AND GUGERCIN, S. A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions. *SIAM Journal on Scientific Computing 38*, 2 (2016), A631–A648.

[22] ECKART, C., AND YOUNG, G. The approximation of one matrix by another of lower rank. *Psychometrika 1*, 3 (Sep 1936), 211–218.

[23] GEIST, G. A., AND ROMINE, C. H. lu factorization algorithms on distributed-memory multiprocessor architectures. *SIAM Journal on Scientific and Statistical Computing 9*, 4 (1988), 639–649.

[24] GOLUB, G. H., AND VAN LOAN, C. F. *Matrix Computations*, third ed. The Johns Hopkins University Press, 1996.

[25] GOREINOV, S., TYRTYSHNIKOV, E., AND ZAMARASHKIN, N. A theory of pseudoskeleton approximations. *Linear Algebra and its Applications 261*, 1 (1997), 1–21.

[26] GRIGORI, L., DEMMEL, J. W., AND XIANG, H. Calu: A communication optimal lu factorization algorithm. *SIAM Journal on Matrix Analysis and Applications 32*, 4 (2011), 1317–1350.

[27] GU, M., AND EISENSTAT, S. C. Efficient algorithms for computing a strong rank-revealing qr factorization. *SIAM Journal on Scientific Computing 17*, 4 (1996), 848–869.

[28] HALKO, N., MARTINSSON, P. G., AND TROPP, J. A. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review 53*, 2 (2011), 217–288.

[29] HO, K., OLVER, S., KELMAN, T., JARLEBRING, E., TAGBOT, J., AND SLEVINSKY, M. Lowrankapprox,jl: v0.4.3. https://github.com/JuliaMatrices/LowRankApprox.jl, 2020.

[30] HONG, Y. P., AND PAN, C.-T. Rank-revealing qr factorizations and the singular value decomposition. *Mathematics of Computation 58*, 197 (1992), 213–232.

[31] HOUSEHOLDER, A. S. Unitary triangularization of a nonsymmetric matrix. *J. ACM 5*, 4 (Oct. 1958), 339–342.

[32] INDYK, P., AND MOTWANI, R. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 1998), STOC '98, Association for Computing Machinery, p. 604–613.

[33] KAHAN, W. Numerical linear algebra. *Canadian Mathematical Bulletin 9*, 5 (1966), 757–801.

[34] KEZHONG ZHAO, VOUVAKIS, M. N., AND JIN-FA LEE. The adaptive cross approximation algorithm for accelerated method of moments computations of emc problems. *IEEE Transactions on Electromagnetic Compatibility 47*, 4 (2005), 763–773.

[35] KURZAK, J., LUSZCZEK, P., FAVERGE, M., AND DONGARRA, J. Lu factorization with partial pivoting for a multicore system with accelerators. *IEEE Transactions on Parallel and Distributed Systems 24*, 8 (2013), 1613–1621.

[36] LIBERTY, E., WOOLFE, F., MARTINSSON, P.-G., ROKHLIN, V., AND TYGERT, M. Randomized algorithms for the low-rank approximation of matrices. *Proceedings of the National Academy of Sciences 104*, 51 (2007), 20167–20172.

[37] MAHONEY, M. W., AND DRINEAS, P. Cur matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences 106*, 3 (2009), 697–702.

[38] MARTINSSON, P.-G., AND TROPP, J. A. Randomized numerical linear algebra: Foundations and algorithms. *Acta Numerica 29* (2020), 403–572.

[39] MENG, X., AND MAHONEY, M. W. Low-distortion subspace embeddings in input-sparsity time and applications to robust linear regression. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing* (New York, NY, USA, 2013), STOC '13, Association for Computing Machinery, p. 91–100.

[40] MESZAROS, C. Meszaros/large, lp sequence: large000 to large036. http://old.sztaki.hu/~meszaros/public_ftp/lptestset/, 2004.

[41] MIRANIAN, L., AND GU, M. Strong rank revealing lu factorizations. *Linear Algebra and its Applications 367* (07 2003), 1–16.

[42] MUSCO, C., AND MUSCO, C. Randomized block krylov methods for stronger and faster approximate singular value decomposition. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1* (Cambridge, MA, USA, 2015), NIPS'15, MIT Press, p. 1396–1404.

[43] NELSON, J., AND NGUYÊN, H. L. Osnap: Faster numerical linear algebra algorithms via sparser subspace embeddings. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science* (2013), pp. 117–126.

[44] PAN, C.-T. On the existence and computation of rank-revealing lu factorizations. *Linear Algebra and its Applications 316*, 1 (2000), 199–222. Special Issue: Conference celebrating the 60th birthday of Robert J. Plemmons.

[45] PAN, V. Y., QIAN, G., AND YAN, X. Random multipliers numerically stabilize gaussian and block gaussian elimination: Proofs and an extension to low-rank approximation. *Linear Algebra and its Applications 481* (2015), 202–234.

[46] PAN, V. Y., AND ZHAO, L. Numerically safe gaussian elimination with no pivoting. *Linear Algebra and its Applications 527* (2017), 349–383.

[47] PETERS, G., AND WILKINSON, J. H. On the stability of gauss-jordan elimination with pivoting. *Commun. ACM 18*, 1 (Jan. 1975), 20–24.

[48] ROKHLIN, V., AND TYGERT, M. A fast randomized algorithm for overdetermined linear least-squares regression. *Proceedings of the National Academy of Sciences 105*, 36 (2008), 13212–13217.

[49] SHABAT, G., SHMUELI, Y., AIZENBUD, Y., AND AVERBUCH, A. Randomized lu decomposition. *Applied and Computational Harmonic Analysis 44*, 2 (2018), 246–272.

[50] SOLOMONIK, E., AND DEMMEL, J. Communication-optimal parallel 2.5d matrix multiplication and lu factorization algorithms. In *Euro-Par 2011 Parallel Processing* (Berlin, Heidelberg, 2011), E. Jeannot, R. Namyst, and J. Roman, Eds., Springer Berlin Heidelberg, pp. 90–109.

[51] SORENSEN, D. C., AND EMBREE, M. A deim induced cur factorization. *SIAM Journal on Scientific Computing 38*, 3 (2016), A1454–A1482.

[52] TREFETHEN, L. N., AND BAU, D. *Numerical Linear Algebra.* SIAM, 1997.

[53] TREFETHEN, L. N., AND SCHREIBER, R. S. Average-case stability of gaussian elimination. *SIAM Journal on Matrix Analysis and Applications 11*, 3 (1990), 335–360.

[54] TROPP, J. A. Improved analysis of the subsampled randomized hadamard transform. *Advances in Adaptive Data Analysis 03*, 01n02 (2011), 115–126.

[55] TROPP, J. A., YURTSEVER, A., UDELL, M., AND CEVHER, V. Fixed-rank approximation of a positive-semidefinite matrix from streaming data. In *Advances in Neural Information Processing Systems* (2017), I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30, Curran Associates, Inc.

[56] TROPP, J. A., YURTSEVER, A., UDELL, M., AND CEVHER, V. Streaming low-rank matrix approximation with an application to scientific simulation. *SIAM Journal on Scientific Computing 41*, 4 (2019), A2430–A2463.

[57] TYRTYSHNIKOV, E. Incomplete cross approximation in the mosaic-skeleton method. *Computing 64*, 4 (Jun 2000), 367–380.

[58] VORONIN, S., AND MARTINSSON, P.-G. Efficient algorithms for cur and interpolative matrix decompositions. *Advances in Computational Mathematics 43*, 3 (Jun 2017), 495–516.

[59] WANG, S., AND ZHANG, Z. A scalable cur matrix decomposition algorithm: Lower time complexity and tighter bound. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1* (Red Hook, NY, USA, 2012), NIPS'12, Curran Associates Inc., p. 647–655.

[60] WOODRUFF, D. P. Sketching as a tool for numerical linear algebra. *Found. Trends Theor. Comput. Sci. 10*, 1–2 (Oct. 2014), 1–157.

[61] WOOLFE, F., LIBERTY, E., ROKHLIN, V., AND TYGERT, M. A fast randomized algorithm for the approximation of matrices. *Applied and Computational Harmonic Analysis 25*, 3 (2008), 335–366.

[62] ÇIVRIL, A., AND MAGDON-ISMAIL, M. On selecting a maximum volume sub-matrix of a matrix and related problems. *Theoretical Computer Science 410*, 47 (2009), 4801–4811.

ODEN INSTITUTE FOR COMPUTATIONAL ENGINEERING & SCIENCES, UNIVERSITY OF TEXAS AT AUSTIN