Sublinear Randomized Algorithms for Skeleton Decompositions

Citation: Chiu, Jiawei, and Laurent Demanet. "Sublinear Randomized Algorithms for Skeleton Decompositions." SIAM Journal on Matrix Analysis and Applications 34, no. 3 (July 9, 2013): 1361-1383. © 2013, Society for Industrial and Applied Mathematics

As Published: http://dx.doi.org/10.1137/110852310

Publisher: Society for Industrial and Applied Mathematics

Persistent URL: http://hdl.handle.net/1721.1/83890

Version: Final published version: final published article, as it appeared in a journal, conference proceedings, or other formally published context

SIAM J. MATRIX ANAL. APPL.
Vol. 34, No. 3, pp. 1361–1383

SUBLINEAR RANDOMIZED ALGORITHMS FOR SKELETON DECOMPOSITIONS*

JIAWEI CHIU† AND LAURENT DEMANET‡

Abstract. A skeleton decomposition of a matrix A is any factorization of the form $A_{:C}ZA_{R:}$, where $A_{:C}$ comprises columns of A, and $A_{R:}$ comprises rows of A. In this paper, we investigate the conditions under which random sampling of C and R results in accurate skeleton decompositions. When the singular vectors (or more generally the generating vectors) are *incoherent*, we show that a simple algorithm returns an accurate skeleton in sublinear $O(\ell^3)$ time from $\ell \simeq k \log n$ rows and columns drawn uniformly at random, with an approximation error of the form $O(\frac{n}{\ell}\sigma_k)$ where σ_k is the kth singular value of A. We discuss the crucial role that *regularization* plays in forming the middle matrix U as a pseudoinverse of the restriction A_{RC} of A to rows in R and columns in C. The proof methods enable the analysis of two alternative sublinear-time algorithms, based on the rank-revealing QR decomposition, which allows us to tighten the number of rows and/or columns to k with error bound proportional to σ_k.

Key words. skeleton, column subset selection, low-rank approximations, CUR, interpolative decomposition, Nystrom method

AMS subject classifications. 68W20, 65F30

DOI. 10.1137/110852310

1. Introduction.

1.1. Skeleton decompositions.

This paper is concerned with the decomposition known as the matrix skeleton, pseudoskeleton [22], or CUR factorization [28, 18].

Throughout this paper we adopt the following MATLAB-friendly notation. Let R, C be index sets. Given $A \in \mathbb{C}^{m\times n}$, let $A_{:C}$ denote the restriction of A to columns indexed by C, and $A_{R:}$ denote the restriction of A to rows indexed by R. A skeleton decomposition of A is any factorization of the form

$$A_{:C}ZA_{R:} \text{ for some } Z \in \mathbb{C}^{k\times k}.$$

In general, storing a rank-k approximation of A takes $O((m+n)k)$ space. For skeletons however, only the middle factor Z and the two index sets C and R need to be stored, if we assume that A's entries can be sampled on-demand by an external function. Hence specifying the skeleton decomposition of A only requires $O(k^2)$ space. In addition, rows and columns from the original matrix may carry more physical significance than their linear combinations.

There are important examples where the full matrix itself is not low rank but can be partitioned into blocks each of which has low numerical rank. One example is the Green's function of elliptic operators with mild regularity conditions [5]. Another example is the amplitude factor in certain Fourier integral operators and wave propagators [12, 17]. Algorithms that compute good skeleton representations can be used to manipulate such matrices.

*Received by the editors October 19, 2011; accepted for publication (in revised form) by D. P. O'Leary June 20, 2013; published electronically September 17, 2013.
http://www.siam.org/journals/simax/34-3/85231.html
†Corresponding author. Department of Mathematics, MIT, Cambridge, MA 02139 (jiawei@mit.edu). This author was supported by the A*STAR fellowship from Singapore.
‡Department of Mathematics, MIT, Cambridge, MA 02139 (laurent@math.mit.edu). This author was supported in part by the National Science Foundation and the Alfred P. Sloan foundation.

ALGORITHM 1. $\widetilde{O}(k^3)$-time algorithm where $\widetilde{O}$ is the O notation with log factors dropped

Input: A matrix $A \in \mathbb{C}^{m\times n}$ that is approximately rank k, and user-defined parameters $\ell = \widetilde{O}(k)$ and δ.

Output: Column index set C of size ℓ, row index set R of size ℓ, center matrix Z of a matrix skeleton. Implicitly, we have the matrix skeleton $A_{:C}ZA_{R:}$.

Steps:

1. Let C be a random index set of size ℓ chosen uniformly from $\{1,\dots,n\}$. Implicitly, we have $A_{:C}$.
2. Let R be a random index set of size ℓ chosen uniformly from $\{1,\dots,m\}$. Implicitly, we have $A_{R:}$.
3. Form A_{RC}, the intersection of $A_{:C}$ and $A_{R:}$.
4. Compute the thin SVD of A_{RC} as $U_1\Sigma_1V_1^* + U_2\Sigma_2V_2^*$ where Σ_1, Σ_2 are diagonal, Σ_1 contains singular values $\geq \delta$, and Σ_2 contains singular values $< \delta$.
5. Compute $Z = V_1\Sigma_1^{-1}U_1^*$.

MATLAB code:

```
function [C,Z,R]=skeleton1(A,l,delta)
    C=randperm(n,l); R=randperm(m,l); Z=pinv(A(R,C),delta);
end
```

1.2. Overview. This paper mostly treats the case of skeleton decompositions with C and R drawn uniformly at random. Denote by A_{RC} the restriction of A to rows in R and columns in C, we compute the middle matrix Z as the pseudoinverse of A_{RC} with some amount of regularization. Algorithm 1 details the form of this regularization.

Throughout the paper we use the letter k to denote the baseline small dimension of the factorization: it is either exactly the rank of A, or, more generally, it is the index of the singular value σ_k that governs the approximation error of the skeleton decomposition. The small dimension of the skeleton decomposition may or may not be k; for instance, Algorithm 1 requires a small oversampling since $\ell = O(k \log \max(m,n))$. Later, we consider two algorithms where ℓ is exactly k.

The situation in which Algorithm 1 works is when A is a priori known to have a factorization of the form $\simeq X_1A_{11}Y_1^*$ where X_1, Y_1 have k orthonormal columns, and these columns are *incoherent*, or spread, in the sense that their uniform norm is about as small as their normalization allows. In this scenario, our main result in Theorem 1.1 states that the output of Algorithm 1 obeys

$$\|A - A_{:C}ZA_{R:}\| = O\left(\|A - X_1A_{11}Y_1^*\|\frac{(mn)^{1/2}}{\ell}\right)$$

with high probability, for some adequate choice of the regularization parameter δ.

The drawback of Algorithm 1 is that it requires setting an appropriate regularization parameter in advance. Unfortunately, there is no known way of estimating it fast, and this regularization step cannot be skipped. In section 4.1, we illustrate with a numerical example that without the regularization, the error in the operator norm can blow up in a way predicted by our main result, Theorem 1.1.

Finally, we use our proof framework to establish error estimates for two other algorithms. The goal of these algorithms is to further reduce the small dimension

of the skeleton decomposition to exactly k (instead of $\ell = O(k \log \max(m, n))$), with σ_k still providing control over the approximation error. The proposed methods still run in $o(mn)$ time (i.e., sublinear); they use well-known strong rank-revealing QR (RRQR) factorizations applied *after* some amount of pruning via uniform random sampling of rows and columns. This combination of existing ideas is an important part of the discussion of how skeleton factorizations can be computed reliably without visiting all the elements of the original matrix.

1.3. Related work. The idea of uniformly sampling rows and columns to build a matrix skeleton is not new. In particular, for the case where A is symmetric, this technique is known as the Nyström method.[1] The skeleton used is $A_{:C}A_{CC}^{+}A_{C:}$, which is symmetric, and the error in the operator norm was recently analyzed by Talwalkar [36] and Gittens [21]. Both papers make the assumption that X_1, Y_1 are incoherent. Gittens obtained relative error bounds that are similar to ours.

Nonetheless, our results are more general. They apply to nonsymmetric matrices that are low rank in a broader sense. Specifically, when we write $A \simeq X_1 A_{11} Y_1^*$, A_{11} is not necessarily diagonal and X_1, Y_1 are not necessarily the singular vectors of A. This relaxes the incoherence requirement on X_1, Y_1. Furthermore, in the physical sciences, it is not uncommon to work with linear operators that are known a priori to be almost (but not fully) diagonalized by the Fourier basis or related bases in harmonic analysis. These bases are often incoherent. One example is an integral operator with a smooth kernel. See section 4 for more details.

A factorization that is closely related to matrix skeletons is the interpolative decomposition [15], also called the column subset selection problem [20] or a RRQR [13, 23]. An interpolative decomposition of A is the factorization $A_{:C}D$ for some D. It is relevant to this paper because algorithms that compute interpolative decompositions can be used to compute matrix skeletons [27]. Algorithms 2 and 3 require the computation of interpolative decompositions.

One of the earliest theoretical results concerning matrix skeletons is due to Goreinov, Tyrtyshnikov, and Zamarashkin [22]. In that paper, it is shown that for any $A \in \mathbb{C}^{m\times n}$, there exists a skeleton $A_{:C}ZA_{R:}$ such that in the operator norm, $\|A - A_{:C}ZA_{R:}\| = O(\sqrt{k}(\sqrt{m} + \sqrt{n})\sigma_{k+1}(A))$. Although the proof is constructive, it requires computing the SVD of A, which takes much more time and space than the algorithms considered in this paper. A useful idea in [22] for selecting C and R is to maximize the volume or determinant of submatrices. This idea may date back to interpolating projections [14] and the proof of Auerbach's theorem [33].

A popular method of computing matrix skeletons is cross-approximation. The idea is to iteratively select good rows and columns based on the residual matrix. As processing the entire residual matrix is not practical, there are faster variants that operate on only a small part of the residual, e.g., adaptive cross approximation [3, 4] and incomplete cross approximation [38]. The algorithms considered in this paper are noniterative, arguably easier to implement and analyze, yet possibly less efficient for some applications.

In this paper, we compute a matrix skeleton by randomly sampling rows and columns of A. This idea dates back at least to the work of Frieze, Kannan, and Vempala [20]. One way of sampling rows and columns of A is called "subspace sampling" [28, 18] by Drineas, Mahoney, and Muthukrishnan. If we assume that the top k singu-

[1] In machine learning, the Nyström method can be used to approximate kernel matrices of support vector machines, or the Laplacian of affinity graphs, for instance.

lar vectors of A are incoherent, then a result due to Rudelson [31] and Rudelson and Vershynin [32] implies that *uniform sampling* of rows and columns, a special case of "subspace sampling," will produce a good skeleton representation $A_{:C}(A_{:C}^{+}AA_{R:}^{+})A_{R:}$. However, it is not clear how the middle matrix $A_{:C}^{+}AA_{R:}^{+}$ can be computed in sublinear time.

In the main algorithm analyzed in this paper, we *uniformly sample* rows and columns to produce a skeleton of the form $A_{:C}A_{RC}^{+}A_{R:}$, not $A_{:C}(A_{:C}^{+}AA_{R:}^{+})A_{R:}$. One major difference is that the skeleton $A_{:C}A_{RC}^{+}A_{R:}$ can be computed in $\widetilde{O}(k^3)$ time.[2] Note that the matrix skeleton output by our algorithms is represented by the index sets R, C and matrix Z, not $A_{R:}, A_{:C}$.

Finally, let us mention that the term "skeleton" may refer to other factorizations. Instead of $A \simeq A_{C:}ZA_{:R}$, we can have $A \simeq Z_1A_{RC}Z_2$, where Z_1, Z_2 are arbitrary $m \times k$ and $k \times n$ matrices [15]. As $O(mk + nk)$ space is needed to store Z_1, Z_2, this representation does not seem as appealing as the representation $A_{:C}ZA_{R:}$ in memory-critical situations. Nevertheless, it is numerically more stable and has found several applications [25].

Alternatively, when $A = MBN$, where M, B, N are $n \times n$ matrices, we can approximate M as $M_{:C}P$, N as $DN_{R:}$, where M_C has k columns of M and N_R has k rows of N. Thus, $A \simeq M_C(PBD)N_R$, effectively replacing B with the $k \times k$ matrix $\widetilde{B} := PBD$. Bremer calls $\widetilde{B}$ a skeleton and uses it to approximate scattering matrices [9].

1.4. Notation. The matrices we consider in this paper take the form

$$A = (X_1 \quad X_2)\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}\begin{pmatrix} Y_1^* \\ Y_2^* \end{pmatrix}, \tag{1}$$

where $X = (X_1 \quad X_2)$ and $Y = (Y_1 \quad Y_2)$ are unitary matrices, with columns being "*spread*," and the blocks A_{12}, A_{21}, and A_{22} are in some sense *small*. By "spread," we mean $\widetilde{O}(1)$-coherent.

Definition 1.1. *Let $X \in \mathbb{C}^{n\times k}$ be a matrix with k orthonormal columns. Denote $\|X\|_{\max} = \max_{ij}|X_{ij}|$. We say X is μ-coherent if $\|X\|_{\max} \le (\mu/n)^{1/2}$.*

This notion is well known in compressed sensing [11] and matrix completion [10, 29].

Formally, let

$$\Delta_k := \begin{pmatrix} 0 & A_{12} \\ A_{21} & A_{22} \end{pmatrix},$$

and assume that

$$\varepsilon_k := \|\Delta_k\| \text{ is small.} \tag{2}$$

That means A can be represented using only $O(k^2)$ data if we allow an ε_k error in the operator norm. Note that ε_k is equivalent to $\max(\|X_2^*A\|, \|AY_2\|)$ up to constants. To prevent clutter, we have suppressed the dependence on k from the definitions of X_1, Y_1, A_{11}, A_{12} etc.

If (1) is the SVD of A, then $\varepsilon_k = \sigma_{k+1}(A)$. It is good to keep this example in mind as it simplifies many formulas that we see later.

[2]Note that $\widetilde{O}$ is the O notation with log factors dropped.

An alternative to ε_k is

$$\varepsilon'_k := \sum_{i=1}^{m}\sum_{j=1}^{n} |(\Delta_k)_{ij}| \,. \tag{3}$$

In other words, ε'_k is the ℓ^1 norm of Δ_k reshaped into a vector. We know $\varepsilon_k \le \varepsilon'_k \le mn\varepsilon_k$. The reason for introducing ε'_k is that it is common for $(\Delta_k)_{ij}$ to decay rapidly such that $\varepsilon'_k \ll mn\varepsilon_k$. For such scenarios, the error guarantee of Algorithm 1 is much stronger in terms of ε'_k than in terms of ε_k, as we will see in the next section.

1.5. Main result. Random subsets of rows and columns are only representative of the subspaces of the matrix A under the incoherence assumption mentioned earlier; otherwise Algorithm 1 may fail. For example, if $A = X_1 A_{11} Y_1^*$ and $X_1 = \binom{I_{k\times k}}{0}$, then $A_{R:}$ is going to be zero most of the time, and so is $A_{:C} Z A_{R:}$. Hence, it makes sense that we want $X_{1,R:}$ to be "as nonsingular as possible" so that little information is lost. In particular, it is well known that if X_1, Y_1 are $\widetilde{O}(1)$-coherent, i.e., *spread*, then sampling $\ell = \widetilde{O}(k)$ rows will lead to $X_{1,R:}, Y_{1,C:}$ being well conditioned.[3]

Here is our main result. It is proved in section 2.

THEOREM 1.1. *Let A be given by* (1) *for some $k > 0$. Assume $m \ge n$ and X_1, Y_1 are μ-coherent where $\mu = \widetilde{O}(1)$*[4] *with respect to m, n. Recall the definitions of $\varepsilon_k, \varepsilon'_k$ in* (2) *and* (3). *Let $\ell \ge 10\mu k \log m$ and $\lambda = \frac{(mn)^{1/2}}{\ell}$. Then with probability at least $1 - 4km^{-2}$, Algorithm* 1 *returns a skeleton that satisfies*

$$\|A - A_{:C} Z A_{R:}\| = O(\lambda\delta + \lambda\varepsilon_k + \varepsilon_k^2 \lambda/\delta). \tag{4}$$

If, furthermore, the entire X and Y are μ-coherent, then with probability at least $1 - 4m^{-1}$,

$$\|A - A_{:C} Z A_{R:}\| = O(\lambda\delta + \varepsilon'_k + {\varepsilon'_k}^2/(\lambda\delta)). \tag{5}$$

The right-hand sides of (4) and (5) can be minimized with respect to δ. For (4), pick $\delta = \Theta(\varepsilon_k)$ so that

$$\|A - A_{:C} Z A_{R:}\| = O(\varepsilon_k \lambda) = O(\varepsilon_k (mn)^{1/2}/\ell). \tag{6}$$

For (5), pick $\delta = \Theta(\varepsilon'_k/\lambda)$ so that

$$\|A - A_{:C} Z A_{R:}\| = O(\varepsilon'_k). \tag{7}$$

Here are some possible scenarios where $\varepsilon'_k = o(\varepsilon_k \lambda)$ and (7) is strictly stronger than (6):

- The entries of Δ_k *decay exponentially* or there are only $O(1)$ nonzero entries as m, n increases. Then $\varepsilon'_k = \Theta(\varepsilon_k)$.
- Say $n = m$ and (1) is the SVD of A. Suppose the singular values decay as $m^{-1/2}$. Then $\varepsilon'_k = O(\varepsilon_k m^{1/2})$.

[3]Assume $\ell = \widetilde{O}(k)$. Then $\|Y_1\|_{\max} = \widetilde{O}(n^{-1/2})$ is a sufficient condition for $Y_{1,C:}$ to be well conditioned with high probability. This condition can be relaxed in at least two ways. First, all we need is that for each row i, $(\sum_j |(Y_1)_{ij}|^2)^{1/2} \le (\mu k/n)^{1/2}$. This would allow a few entries of each row of Y_1 to be bigger than $O(n^{-1/2})$. Second, we can allow a few rows of Y to violate the previous condition [2].

[4]More generally, the algorithm runs in $O(\ell^3) = O(k^3\mu^3 \log^3 m)$ time which is $o(mn)$ or sublinear if $k\mu = o((mn)^{1/3})$.

One important question remains: How can we guess ε_k, in order to then choose δ? Unfortunately, we are not aware of any $\widetilde{O}(k^3)$ algorithm that can accurately estimate ε_k. Here is one possible *heuristic* for choosing δ for the case where (1) is the SVD. Imagine $A_{RC} \simeq X_{1,R:} A_{11} Y_{1,C:}^*$. As we will see, the singular values of $X_{1,R:}, Y_{1,C:}$ are likely to be on the order of $(\ell/m)^{1/2}, (\ell/n)^{1/2}$. Therefore, it is not unreasonable to view $\varepsilon_k \simeq \sigma_{k+1}(A) \simeq \lambda\sigma_{k+1}(A_{RC})$.

Another approach is to begin with a big δ, run the $\widetilde{O}(k^3)$ algorithm, check $\|A - A_{:C} Z A_{R:}\|$, divide δ by two and repeat the whole process until the error does not improve. However, calculating $\|A - A_{:C} Z A_{R:}\|$ is expensive and other tricks are needed. This seems to be an open problem.

The $\widetilde{O}(k^3)$ algorithm is among the fastest algorithms for computing skeleton representations that one can expect to have. With more work, can the accuracy be improved? In section 3, we sketch two such algorithms. These two algorithms have, for the most part been analyzed in previous work: though their ability to perform, in sublinear-time complexity was not explicitly stated in those references, this fact should not come as a surprise. The first algorithm samples $\ell \simeq k \log m$ rows, columns, then reduces it to *exactly* k rows, columns using a RRQR decomposition, with an operator norm error of $O(\varepsilon_k (mk)^{1/2})$. This idea is also employed in Boutsidis, Mahoney, and Drineas [7] which pertains to the column subset selection problem. In the second algorithm, we uniformly sample $\ell \simeq k \log m$ rows to get $A_{R:}$, then run RRQR on $A_{R:}$ to select k columns of A. The overall error is $O(\varepsilon_k (mn)^{1/2})$. This is similar to the algorithm proposed in [27, 39] and more details can be found at the end of section 3.2.

Using the proof framework in section 2, we will derive error estimates for the above two algorithms. As mentioned earlier these error guarantees are not new, but (i) they concern provable sublinear-time complexity algorithms, (ii) they work for a more general model (1), and (iii) our proofs are also motivated differently. In section 3.3, we compare these three algorithms.

1.6. More on incoherence. If either X or Y is not $\widetilde{O}(1)$-coherent, we can use the idea of a randomized Fourier transform [1] to impose incoherence. The idea is to multiply them on the left by the unitary Fourier matrix with randomly rescaled columns. This has the effect of "blending up" the rows of X, Y—at the possible cost of requiring linear-time complexity. The following is a standard result that can be proved using Hoeffding's inequality.

PROPOSITION 1.2. *Let $X \in \mathbb{C}^{n\times k}$ with orthonormal columns. Let $D = \operatorname{diag}(d_1, \ldots, d_n)$, where $d_1, \ldots, d_n$ are independent random variables such that $\mathbb{E}d_i = 0$ and $|d_i| = 1$. Let $\mathcal{F}$ be the unitary Fourier matrix and let $\mu = \alpha \log n$ for some $\alpha > 0$. Define $U := \mathcal{F}DX$. Then $\|U\|_{\max} \le (\mu/n)^{1/2}$ with probability at least $1 - 2(nk)n^{-2\alpha}$*

In other words, no matter what X is, $U = \mathcal{F}DX$ would be $\widetilde{O}(1)$-coherent with high probability. Hence, we can write a wrapper around Algorithm 1. Call this Algorithm 1′. Let $\mathcal{F} \in \mathbb{C}^{n\times n}$ and $\mathcal{F}' \in \mathbb{C}^{m\times m}$ be unitary Fourier transforms.

1. Let $B := \mathcal{F}' D_2 A D_1 \mathcal{F}^*$, where D_1, D_2 are diagonal matrices with independent entries that are ± 1 with equal probability.
2. Feed B to the $\widetilde{O}(k^3)$ algorithm and obtain $B \simeq B_{:C} Z B_{R:}$.
3. It follows that $A \simeq (AD_1\mathcal{F}_{C:}^*) Z (\mathcal{F}_{R:} D_2 A)$.

The output $(AD_1\mathcal{F}_{C:}^*) Z (\mathcal{F}_{R:} D_2 A)$ is not a matrix skeleton, but the amount of space needed is $O(n) + \widetilde{O}(k^2)$ which is still better than $O(nk)$. Note that we are not storing $AD_1\mathcal{F}_{C:}^*$ just as we do not store $A_{:C}$ in Algorithm 1. Let T_A be the cost of

matrix-vector multiplication of A. Then Algorithm 1′ runs in $\widetilde{O}(T_A k + mk + k^3)$ time. The most expensive step is computing B_{RC} and it can be carried out as follows.

Compute $D_1(\mathcal{F}^* S_C)$ in $\widetilde{O}(nk)$ time. Multiply the result by A on the left in $\widetilde{O}(T_A k)$ time. Multiply the result by D_2 on the left in $\widetilde{O}(mk)$ time. Multiply the result by $\mathcal{F}'$ on the left in $\widetilde{O}(mk)$ time using FFT. Multiply the result by S_R^T on the left in $\widetilde{O}(k^2)$ time.

2. Error estimates for $\widetilde{O}(k^3)$ algorithm.

2.1. Notation. $S_C, S_R \in \mathbb{R}^{n\times k}$ are both *column* selector matrices. They are column subsets of permutation matrices. The subscripts "R :" and ": C" denote a row subset and a column subset respectively, e.g., $A_{R:} = S_R^T A$ and $A_{:C} = AS_C$, while A_{RC} is a row and column subset of A. Transposes and pseudoinverses are taken after the subscripts, e.g., $A_{R:}^*$ means $(A_{R:})^*$.

2.2. Two principles. Our proofs are built on two principles. The first principle is due to Rudelson [31] in 1999. Intuitively, it says the following:

> Let Y be a $n \times k$ matrix with orthonormal columns. Let $Y_{C:}$ be a *random* row subset of Y. Suppose Y is μ-coherent with $\mu = \widetilde{O}(1)$, and $|C| = \ell \gtrsim \mu k$. Then with high probability, $(\frac{n}{\ell})^{1/2} Y_{C:}$ is like an isometry.

To be precise, we quote [37, Lemma 3.4]. Note that their M is our μk.

THEOREM 2.1. *Let $Y \in \mathbb{C}^{n\times k}$ with orthonormal columns. Suppose Y is μ-coherent and $\ell \geq \alpha k \mu$ for some $\alpha > 0$. Let $Y_{C:}$ be a random ℓ-row subset of Y. Each row of $Y_{C:}$ is sampled independently, uniformly. Then*

$$\mathbb{P}\left(\left\|Y_{C:}^+\right\| \geq \sqrt{\frac{n}{(1-\delta)\ell}}\right) \leq k\left(\frac{e^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\alpha} \quad \textit{for any } \delta \in [0,1)$$

and

$$\mathbb{P}\left(\|Y_{C:}\| \geq \sqrt{\frac{(1+\delta')\ell}{n}}\right) \leq k\left(\frac{e^{\delta'}}{(1+\delta')^{1+\delta'}}\right)^{\alpha} \quad \textit{for any } \delta' \geq 0.$$

To be concrete, if $\delta = 0.57$ and $\delta' = 0.709$ and $\ell \geq 10k\mu \log n$, then

$$\mathbb{P}\left(\left\|Y_{C:}^+\right\| \leq 1.53(n/\ell)^{1/2} \text{ and } \|Y_{C:}\| \leq 1.31(\ell/n)^{1/2}\right) \geq 1 - 2kn^{-2}. \tag{8}$$

We will use (8) later. Let us proceed to the second principle, which says the following:

> Let C be an arbitrary index set. If $\|A_{:C}\|$ is small, then $\|A\|$ is also small, provided that we have control over $\|AY_2\|$ and $\|Y_{1,C:}^+\|$ for some unitary matrix $(Y_1 \quad Y_2)$.

The roadmap is as follows. If we ignore the regularization step, then what we want to show is that $A \simeq A_{:C} A_{RC}^+ A_{R:}$. But when we take row and column restrictions on both sides, we have trivially $A_{RC} = A_{RC} A_{RC}^+ A_{RC}$. Hence, we desire a mechanism to go backwards, that is to infer that "$E := A - A_{:C} A_{RC}^+ A_{R:}$ is small" from "E_{RC} is small." We begin by inferring that "E is small" from "$E_{:C}$ is small."

LEMMA 2.2. *Let $A \in \mathbb{C}^{m\times n}$ and let $Y = (Y_1 \quad Y_2) \in \mathbb{C}^{n\times n}$ be a unitary matrix such that Y_1 has k columns. Select $\ell \geq k$ rows of Y_1 to form $Y_{1,C:} = S_C^T Y_1 \in \mathbb{C}^{\ell\times k}$. Assume $Y_{1,C:}$ has full column rank. Then*

$$\|A\| \leq \left\|Y_{1,C:}^+\right\| \|A_{:C}\| + \left\|Y_{1,C:}^+\right\| \left\|AY_2 Y_{2,C:}^*\right\| + \|AY_2\|.$$

Proof. Note that $Y_{1,C:}^{*}Y_{1,C:}^{*+} = I_{k\times k}$. Now,

$$\begin{aligned}\|A\| &\le \|AY_1\| + \|AY_2\| \\ &= \left\|AY_1Y_{1,C:}^{*}Y_{1,C:}^{*+}\right\| + \|AY_2\| \\ &\le \|AY_1Y_1^{*}S_C\| \left\|Y_{1,C:}^{+}\right\| + \|AY_2\| \\ &\le \|(A - AY_2Y_2^{*})S_C\| \left\|Y_{1,C:}^{+}\right\| + \|AY_2\| \\ &\le \|A_{:C}\| \left\|Y_{1,C:}^{+}\right\| + \|AY_2Y_{2,C:}^{*}\| \left\|Y_{1,C:}^{+}\right\| + \|AY_2\|. \qquad \square\end{aligned}$$

Lemma 2.2 can be extended in two obvious ways. First, we can deduce that "A is small if $A_{R:}$ is small." Second, we can deduce that "A is small if A_{RC} is small." This is what the next lemma establishes. (Although its form looks unduly complicated, control over all the terms is *in fine* necessary.)

LEMMA 2.3. *Let $A \in \mathbb{C}^{m\times n}$, and let $X = (X_1 \quad X_2) \in \mathbb{C}^{m\times m}$, and let $Y = (Y_1 \quad Y_2) \in \mathbb{C}^{n\times n}$ be unitary matrices such that X_1, Y_1 each has k columns. Select $\ell \ge k$ rows and columns indexed by R, C, respectively. Assume $X_{1,R:}, Y_{1,C:}$ have full column rank. Then*

$$\|A\| \le \left\|X_{1,R:}^{+}\right\| \|A_{R:}\| + \left\|X_{1,R:}^{+}\right\| \|X_{2,R:}X_2^{*}A\| + \|X_2^{*}A\|$$

and

$$\begin{aligned}\|A\| \le{}& \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \|A_{RC}\| \\ &+ \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \|X_{2,R:}X_2^{*}AY_1Y_{1,C:}^{*}\| \\ &+ \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \|X_{1,R:}X_1^{*}AY_2Y_{2,C:}^{*}\| \\ &+ \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \|X_{2,R:}X_2^{*}AY_2Y_{2,C:}^{*}\| \\ &+ \left\|X_{1,R:}^{+}\right\| \|X_{1,R:}X_1^{*}AY_2\| \\ &+ \left\|Y_{1,C:}^{+}\right\| \|X_2^{*}AY_1Y_{1,C:}^{*}\| \\ &+ \|X_2^{*}AY_2\|.\end{aligned}$$

Proof. The top inequality is obtained by applying Lemma 2.2 to A^{*}.

The proof of the bottom inequality is similar to the proof of Lemma 2.2. For completeness,

$$\begin{aligned}\|A\| &\le \|X_1^{*}AY_1\| + \|X_1^{*}AY_2\| + \|X_2^{*}AY_1\| + \|X_2^{*}AY_2\| \\ &= \left\|X_{1,R:}^{+}X_{1,R:}X_1^{*}AY_1Y_{1,C:}^{*}Y_{1,C:}^{*+}\right\| \\ &\quad + \left\|X_{1,R:}^{+}X_{1,R:}X_1^{*}AY_2\right\| + \left\|X_2^{*}AY_1Y_{1,C:}^{*}Y_{1,C:}^{*+}\right\| + \|X_2^{*}AY_2\| \\ &\le \left\|X_{1,R:}^{+}\right\| \|S_R^{T}X_1X_1^{*}AY_1Y_1^{*}S_C\| \left\|Y_{1,C:}^{*+}\right\| \\ &\quad + \left\|X_{1,R:}^{+}\right\| \|X_{1,R:}X_1^{*}AY_2\| + \|X_2^{*}AY_1Y_{1,C:}^{*}\| \left\|Y_{1,C:}^{*+}\right\| + \|X_2^{*}AY_2\| \\ &= \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \|S_R^{T}(A - X_2X_2^{*}AY_1Y_1^{*} - X_1X_1^{*}AY_2Y_2^{*} - X_2X_2^{*}AY_2Y_2^{*})S_C\| \\ &\quad + \left\|X_{1,R:}^{+}\right\| \|X_{1,R:}X_1^{*}AY_2\| + \left\|Y_{1,C:}^{+}\right\| \|X_2^{*}AY_1Y_{1,C:}^{*}\| + \|X_2^{*}AY_2\|.\end{aligned}$$

Apply the triangle inequality to $\|S_R^T(A - X_2X_2^*AY_1Y_1^* - X_1X_1^*AY_2Y_2^* - X_2X_2^*AY_2 Y_2^*)S_C\|$ and we are done. ∎

We conclude this section with a useful corollary. It says that if $PA_{:C}$ is a good low rank approximation of $A_{:C}$ for some $P \in \mathbb{C}^{m\times m}$, then PA may also be a good low rank approximation of A.

COROLLARY 2.4. *Let $A \in \mathbb{C}^{m\times n}$ and $P \in \mathbb{C}^{m\times m}$. Let $Y = (Y_1 \quad Y_2)$ be a unitary matrix such that Y_1 has k columns. Let $Y_{1,C:} = S_C^T Y_1 \in \mathbb{C}^{\ell\times k}$, where $\ell \geq k$. Assume $Y_{1,C:}$ has full column rank. Let $I \in \mathbb{C}^{m\times m}$ be the identity. Then*

$$\|A - PA\| \leq \left\|Y_{1,C:}^+\right\| \|A_{:C} - PA_{:C}\| + \left\|Y_{1,C:}^+\right\| \|I - P\| \left\|AY_2Y_{2,C:}^*\right\| + \|I - P\| \|AY_2\|.$$

In particular, if P is the orthogonal projection $A_{:C}A_{:C}^+$, then

$$\left\|A - A_{:C}A_{:C}^+A\right\| \leq \left\|Y_{1,C:}^+\right\| \left\|AY_2Y_{2,C:}^*\right\| + \|AY_2\|. \tag{9}$$

Proof. To get the first inequality, apply Lemma 2.2 to $A - PA$. The second inequality is immediate from the first inequality since $\left\|A_{:C} - A_{:C}A_{:C}^+A_{:C}\right\| = 0$. ∎

For the special case where X, Y are singular vectors of A, (9) can be proved using the fact that $\left\|A - A_{:C}A_{:C}^+A\right\| = \min_D \|A - A_{:C}D\|$ and choosing an appropriate D. See Boutsidis, Mahoney, and Drineas [7].

Note that (9) can be strengthened to $\left\|A - A_{:C}A_{:C}^+A\right\|^2 \leq \left\|AY_2Y_{2,C:}^*Y_{1,C:}^+\right\|^2 + \|AY_2\|^2$ by modifying the first step of the proof of Lemma 2.2 from $\|A\| \leq \|AY_1\| + \|AY_2\|$ to $\|A\|^2 \leq \|AY_1\|^2 + \|AY_2\|^2$. A similar result for the case where X, Y are singular vectors can be found in Halko, Martinsson, and Tropp [24]. The originality of our results is that they hold for a more general model (1).

2.3. Proof of Theorem 1.1. The proof is split into two main parts. The first part is probabilistic. We will apply the first principle to control the largest and smallest singular values of $Y_{1,C:}, X_{1,R:}$ and other similar quantities. The second part, mainly linear algebra, uses these bounds on $Y_{1,C:}$ and $X_{1,R:}$ to help control the error $\left\|A - A_{:C}B_{RC}^+A_{R:}\right\|$.

2.3.1. Probabilistic part. Let $\lambda_X = (\frac{m}{\ell})^{1/2}$ and $\lambda_Y = (\frac{n}{\ell})^{1/2}$. To prove the first part of Theorem 1.1, i.e., (4), we apply Theorem 2.1. From (8), it is clear that the assumptions of Theorem 1.1 guarantee that $\|Y_{1,C:}\| = O(\lambda_Y^{-1})$, $\|Y_{1,C:}^+\| = O(\lambda_Y)$, $\|X_{1,R:}\| = O(\lambda_X^{-1})$, $\|X_{1,R:}^+\| = O(\lambda_X)$ hold simultaneously with probability at least $1 - 4km^{-2}$.

For the second part of Theorem 1.1, i.e., (5), we need to refine (1) as follows. Let $X = (\widetilde{X}_1, \ldots, \widetilde{X}_{\lceil m/k\rceil})$ and $Y = (\widetilde{Y}_1, \ldots, \widetilde{Y}_{\lceil n/k\rceil})$, where $\widetilde{X}_1, \ldots, \widetilde{X}_{\lceil m/k\rceil - 1}$ and $\widetilde{Y}_1, \ldots, \widetilde{Y}_{\lceil n/k\rceil - 1}$ has k columns, and $\widetilde{X}_{\lceil m/k\rceil}, \widetilde{Y}_{\lceil n/k\rceil}$ have $\leq k$ columns. Note that $\widetilde{X}_1 = X_1$, $\widetilde{Y}_1 = Y_1$, $\widetilde{A}_{11} = A_{11}$, where X_1, Y_1, A_{11} are defined in (1). Rewrite (1) as

$$A = (\widetilde{X}_1, \ldots, \widetilde{X}_{\lceil m/k\rceil}) \begin{pmatrix} \widetilde{A}_{11} & \ldots & \widetilde{A}_{1,\lceil n/k\rceil} \\ \vdots & \ddots & \vdots \\ \widetilde{A}_{\lceil m/k\rceil,1} & \ldots & \widetilde{A}_{\lceil m/k\rceil,\lceil n/k\rceil} \end{pmatrix} \begin{pmatrix} \widetilde{Y}_1^* \\ \vdots \\ \widetilde{Y}_{\lceil n/k\rceil}^* \end{pmatrix}. \tag{10}$$

By applying Theorem 2.1 to every $\widetilde{X}_i$, $\widetilde{Y}_j$ and doing a union bound, we see that with probability at least $1-4m^{-1}$, we will have $\|Y_{j,C:}\| = O(\lambda_Y^{-1})$, $\|Y_{j,C:}^+\| = O(\lambda_Y)$, $\|X_{i,R:}\| = O(\lambda_X^{-1})$, $\|X_{i,R:}^+\| = O(\lambda_X)$ for all i,j.

2.3.2. Deterministic part: Introducing B, an auxillary matrix. Recall that in Algorithm 1, we compute the SVD of A_{RC} as $U_1\Sigma_1V_1^* + U_2\Sigma_2V_2^*$ and invert only $U_1\Sigma_1V_1^*$ to get the center matrix Z.

Define $B \in \mathbb{C}^{m\times n}$ such that $B_{RC} = U_1\Sigma_1V_1^*$ and all other entries of B are the same as A's. In other words, define $E \in \mathbb{C}^{m\times n}$ such that $E_{RC} = -U_2\Sigma V_2^*$ and all other entries of E are zeros, and then let $B = A + E$.

The skeleton returned is $A_{C:}B_{RC}^+A_{R:}$. By construction,

$$\|A-B\| \le \delta; \quad \left\|B_{RC}^+\right\| \le \delta^{-1}.$$

Our objective is to bound $\left\|A - A_{:C}B_{RC}^+A_{R:}\right\|$, but it is $\left\|B - B_{:C}B_{RC}^+B_{R:}\right\|$ that we have control over by the second principle. Recall that $B_{RC} = B_{RC}B_{RC}^+B_{RC}$ is to be lifted to $B \simeq B_{:C}B_{RC}^+B_{R:}$ by Lemma 2.3. Thus, we shall first relate $\|A - A_{:C}B_{RC}^+A_{R:}\|$ to quantities involving *only* B by a perturbation argument:

$$\begin{aligned}
\left\|A - A_{:C}B_{RC}^+A_{R:}\right\| &\le \|A-B\| + \left\|B - B_{:C}B_{RC}^+B_{R:}\right\| \\
&\quad + \left\|B_{:C}B_{RC}^+B_{R:} - A_{:C}B_{RC}^+B_{R:}\right\| + \left\|A_{:C}B_{RC}^+B_{R:} - A_{:C}B_{RC}^+A_{R:}\right\| \\
&\le \delta + \left\|B - B_{:C}B_{RC}^+B_{R:}\right\| \\
&\quad + \|(B-A)S_C\| \left\|B_{RC}^+B_{R:}\right\| + \left\|A_{:C}B_{RC}^+\right\| \left\|S_R^T(B-A)\right\| \\
&\le \delta + \left\|B - B_{:C}B_{RC}^+B_{R:}\right\| \\
&\quad + \delta \left\|B_{RC}^+B_{R:}\right\| + (\left\|B_{:C}B_{RC}^+\right\| + \left\|A_{:C}B_{RC}^+ - B_{:C}B_{RC}^+\right\|)\delta \\
&\le \delta + \left\|B - B_{:C}B_{RC}^+B_{R:}\right\| \\
&\quad + \delta \left\|B_{RC}^+B_{R:}\right\| + \delta \left\|B_{:C}B_{RC}^+\right\| + \|(A-B)S_C\| \delta^{-1}\delta \\
&\le 2\delta + \left\|B - B_{:C}B_{RC}^+B_{R:}\right\| + \delta \left\|B_{RC}^+B_{R:}\right\| + \delta \left\|B_{:C}B_{RC}^+\right\|. \qquad (11)
\end{aligned}$$

2.3.3. Deterministic part: Bounds on $\|B_{RC}^+B_{R:}\|$ and $\|B_{:C}B_{RC}^+\|$. It remains to bound $\|B - B_{:C}B_{RC}^+B_{R:}\|$, $\|B_{RC}^+B_{R:}\|$, and $\|B_{:C}B_{RC}^+\|$. In this subsection, we obtain bounds for the last two quantities. By the second principle, we do not expect $\|B_{RC}^+B_{R:}\|$ to be much bigger than $\|B_{RC}^+B_{RC}\| \le 1$. Specifically, by Lemma 2.2, we have

$$\begin{aligned}
\left\|B_{RC}^+B_{R:}\right\| &\le \left\|Y_{1,C:}^+\right\| \left\|B_{RC}^+B_{RC}\right\| + \left\|Y_{1,C:}^+\right\| \left\|B_{RC}^+B_{R:}Y_2Y_{2,C:}^*\right\| + \left\|B_{RC}^+B_{R:}Y_2\right\| \\
&\le \left\|Y_{1,C:}^+\right\| + \left\|Y_{1,C:}^+\right\| \left\|B_{RC}^+\right\| (\left\|(B_{R:} - A_{R:})Y_2Y_{2,C:}^*\right\| + \left\|A_{R:}Y_2Y_{2,C:}^*\right\|) \\
&\quad + \left\|B_{RC}^+\right\| (\|(B_{R:} - A_{R:})Y_2\| + \|A_{R:}Y_2\|) \\
&\le \left\|Y_{1,C:}^+\right\| + \left\|Y_{1,C:}^+\right\| \delta^{-1}(\delta + \left\|A_{R:}Y_2Y_{2,C:}^*\right\|) + \delta^{-1}(\delta + \|A_{R:}Y_2\|) \\
&\le 1 + 2\left\|Y_{1,C:}^+\right\| + \left\|Y_{1,C:}^+\right\| \delta^{-1} \left\|A_{R:}Y_2Y_{2,C:}^*\right\| + \delta^{-1} \|A_{R:}Y_2\|.
\end{aligned}$$

By the first principle, the following holds with high probability:

$$\left\|B_{RC}^+B_{R:}\right\| = O(\lambda_Y + \lambda_Y\delta^{-1} \left\|A_{R:}Y_2Y_{2,C:}^*\right\| + \delta^{-1} \|A_{R:}Y_2\|). \qquad (12)$$

The same argument works for $\left\|B_{:C}B_{RC}^{+}\right\|$. With high probability,

$$\left\|B_{:C}B_{RC}^{+}\right\| = O(\lambda_X + \lambda_X \delta^{-1} \left\|X_{2,R:}X_2^* A_{:C}\right\| + \delta^{-1} \left\|X_2^* A_{:C}\right\|). \tag{13}$$

2.3.4. Deterministic part: Bounding $\|B - B_{:C}B_{RC}^{+}B_{R:}\|$. Bounding the third quantity requires more work, but the basic ideas are the same. Recall that the second principle suggests that $\|B - B_{:C}B_{RC}^{+}B_{R:}\|$ cannot be too much bigger than $\|B_{RC} - B_{RC}B_{RC}^{+}B_{RC}\| = 0$. Applying Lemma 2.3 with $B - B_{:C}B_{RC}^{+}B_{R:}$ in the role of A yields

$$\begin{aligned}
\left\|B - B_{:C}B_{RC}^{+}B_{R:}\right\| \leq{}& \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \left\|X_{2,R:}X_2^*(B - B_{:C}B_{RC}^{+}B_{R:})Y_1Y_{1,C:}^*\right\| \\
&+ \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \left\|X_{1,R:}X_1^*(B - B_{:C}B_{RC}^{+}B_{R:})Y_2Y_{2,:C}^*\right\| \\
&+ \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \left\|X_{2,R:}X_2^*(B - B_{:C}B_{RC}^{+}B_{R:})Y_2Y_{2,C:}^*\right\| \\
&+ \left\|X_{1,R:}^{+}\right\| \left\|X_{1,R:}X_1^*(B - B_{:C}B_{RC}^{+}B_{R:})Y_2\right\| \\
&+ \left\|Y_{1,C:}^{+}\right\| \left\|X_2^*(B - B_{:C}B_{RC}^{+}B_{R:})Y_1Y_{1,C:}^*\right\| \\
&+ \left\|X_2^*(B - B_{:C}B_{RC}^{+}B_{R:})Y_2\right\|,
\end{aligned}$$

which is, in turn, bounded by

$$\begin{aligned}
&\left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \left\|Y_{1,C:}\right\| \left(\left\|X_{2,R:}X_2^*B\right\| + \left\|X_{2,R:}X_2^*B_{:C}\right\| \left\|B_{RC}^{+}B_{R:}\right\|\right) \\
&+ \left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \left\|X_{1,R:}\right\| \left(\left\|BY_2Y_{2,C:}^*\right\| + \left\|B_RY_2Y_{2,C:}^*\right\| \left\|B_{:C}B_{RC}^{+}\right\|\right) \\
&\left\|X_{1,R:}^{+}\right\| \left\|Y_{1,C:}^{+}\right\| \left(\left\|X_{2,R:}X_2^*BY_2Y_{2,C:}^*\right\| + \left\|X_{2,R:}X_2^*B_{:C}\right\| \delta^{-1} \left\|B_{R:}Y_2Y_{2,C:}^*\right\|\right) \\
&+ \left\|X_{1,R:}^{+}\right\| \left\|X_{1,R:}\right\| \left(\left\|BY_2\right\| + \left\|B_{:C}B_{RC}^{+}\right\| \left\|B_{R:}Y_2\right\|\right) \\
&+ \left\|Y_{1,C:}^{+}\right\| \left\|Y_{1,C:}\right\| \left(\left\|X_2^*B\right\| + \left\|B_{RC}^{+}B_{R:}\right\| \left\|X_2^*B_{:C}\right\|\right) \\
&+ \left\|X_2^*BY_2\right\| + \left\|X_2^*B_{:C}\right\| \delta^{-1} \left\|B_{R:}Y_2\right\|.
\end{aligned}$$

In the expression above, we have paired $\|X_{1,R:}\|$ with $\|X_{1,R:}^{+}\|$, and $\|Y_{1,C:}\|$ with $\|Y_{1,C:}^{+}\|$ because the first principle implies that their products are $O(1)$ with high probability. This implies that $\|B - B_{:C}B_{RC}^{+}B_{R:}\|$ is less than a constant times,

$$\begin{aligned}
&\lambda_X(\left\|X_{2,R:}X_2^*B\right\| + \left\|X_{2,R:}X_2^*B_C\right\| \left\|B_{RC}^{+}B_{R:}\right\|) \\
&+ \lambda_Y(\left\|BY_2Y_{2,C:}^*\right\| + \left\|B_RY_2Y_{2,C:}^*\right\| \left\|B_{:C}B_{RC}^{+}\right\|) \\
&+ \lambda_X\lambda_Y(\left\|X_{2,R:}X_2^*BY_2Y_{2,C:}^*\right\| + \left\|X_{2,R:}X_2^*B_{:C}\right\| \delta^{-1} \left\|B_{R:}Y_2Y_{2,C:}^*\right\|) \\
&+ \left\|BY_2\right\| + \left\|B_{:C}B_{RC}^{+}\right\| \left\|B_{R:}Y_2\right\| \\
&+ \left\|X_2^*B\right\| + \left\|B_{RC}^{+}B_{R:}\right\| \left\|X_2^*B_{:C}\right\| \\
&+ \left\|X_2^*B_{:C}\right\| \delta^{-1} \left\|B_{R:}Y_2\right\|.
\end{aligned}$$

We have dropped $\|X_2^*BY_2\|$ because it is dominated by $\|X_2^*B\|$. Equations (13) and (12) can be used to control $\|B_{:C}B_{RC}^{+}\|$ and $\|B_{RC}^{+}B_{R:}\|$. Before doing so, we

want to replace B with A in all the other terms. This will introduce some extra δ's. For example, $\|X_{2,R:}X_2^*B\| \leq \|X_{2,R:}X_2^*B - X_{2,R:}X_2^*A\| + \|X_{2,R:}X_2^*A\| \leq \delta + \|X_{2,R:}X_2^*A\|$. Doing the same for other terms, we have that $\|B - B_{:C}B_{RC}^+B_{R:}\|$ is with high probability less than a constant times,

$$\begin{aligned}
&\lambda_X(\delta + \|X_{2,R:}X_2^*A\| + (\delta + \|X_{2,R:}X_2^*A_C\|)\left\|B_{RC}^+B_{R:}\right\|)\\
&+ \lambda_Y(\delta + \left\|AY_2Y_{2,C:}^*\right\| + (\delta + \left\|A_{R:}Y_2Y_{2,C:}^*\right\|)\left\|B_{:C}B_{RC}^+\right\|)\\
&+ \lambda_X\lambda_Y(\delta + \left\|X_{2,R:}X_2^*AY_2Y_{2,C:}^*\right\| + \|X_{2,R:}X_2^*A_{:C}\|\\
&+ \left\|A_{R:}Y_2Y_{2,C:}^*\right\| + \|X_{2,R:}X_2^*A_{:C}\|\,\delta^{-1}\left\|A_{R:}Y_2Y_{2,C:}^*\right\|)\\
&+ \delta + \|AY_2\| + (\delta + \|A_{R:}Y_2\|)\left\|B_{:C}B_{RC}^+\right\|\\
&+ \delta + \|X_2^*A\| + (\delta + \|X_2^*A_{:C}\|)\left\|B_{RC}^+B_{R:}\right\|\\
&+ \delta + \|X_2^*A_{:C}\| + \|A_{R:}Y_2\| + \|X_2^*A_{:C}\|\,\delta^{-1}\|A_{R:}Y_2\|\,.
\end{aligned}$$

Several terms can be simplified by noting that $\delta \leq \lambda_X\delta \leq \lambda_X\lambda_Y\delta$ and $\|X_2^*A_{:C}\| \leq \|X_2^*A\|$. We shall also use the estimates on $\|B_{:C}B_{RC}^+\|$ and $\|B_{RC}^+B_{R:}\|$, from (13) and (12). This leads to

$$\begin{aligned}
&\lambda_X(\|X_{2,R:}X_2^*A\| + (\delta + \|X_{2,R:}X_2^*A_{:C}\|)(\lambda_Y + \lambda_Y\delta^{-1}\left\|A_{R:}Y_2Y_{2,C:}^*\right\| + \delta^{-1}\|A_{R:}Y_2\|))\\
&+ \lambda_Y(\left\|AY_2Y_{2,C:}^*\right\| + (\delta + \left\|A_{R:}Y_2Y_{2,C:}^*\right\|)(\lambda_X + \lambda_X\delta^{-1}\|X_{2,R:}X_2^*A_{:C}\| + \delta^{-1}\|X_2^*A_{:C}\|))\\
&+ \lambda_X\lambda_Y(\delta + \left\|X_{2,R:}X_2^*AY_2Y_{2,C:}^*\right\| + \|X_{2,R:}X_2^*A_{:C}\|\\
&+ \left\|A_{R:}Y_2Y_{2,C:}^*\right\| + \|X_{2,R:}X_2^*A_{:C}\|\,\delta^{-1}\left\|A_{R:}Y_2Y_{2,C:}^*\right\|)\\
&+ \|AY_2\| + (\delta + \|A_{R:}Y_2\|)(\lambda_X + \lambda_X\delta^{-1}\|X_{2,R:}X_2^*A_{:C}\| + \delta^{-1}\|X_2^*A_{:C}\|)\\
&+ \|X_2^*A\| + (\delta + \|X_2^*A_{:C}\|)(\lambda_Y + \lambda_Y\delta^{-1}\left\|A_{R:}Y_2Y_{2,C:}^*\right\| + \delta^{-1}\|A_{R:}Y_2\|)\\
&+ \|X_2^*A_{:C}\|\,\delta^{-1}\|A_{R:}Y_2\|\,.
\end{aligned}$$

Collect the terms by their λ_X, λ_Y factors and drop the smaller terms to obtain that with high probability,

$$\begin{aligned}
\left\|B - B_{:C}B_{RC}^+B_{R:}\right\| &= O(\,\lambda_X(\|X_{2,R:}X_2^*A\| + \|A_{R:}Y_2\| + \delta^{-1}\|X_{2,R:}X_2^*A_{:C}\|\,\|A_{R:}Y_2\|)\\
&+ \lambda_Y(\left\|AY_2Y_{2,C:}^*\right\| + \|X_2^*A_{:C}\| + \delta^{-1}\left\|A_{R:}Y_2Y_{2,C:}^*\right\|\,\|X_2^*A_{:C}\|)\\
&+ \lambda_X\lambda_Y(\delta + \|X_{2,R:}X_2^*A_{:C}\| + \left\|A_{R:}Y_2Y_{2,C:}^*\right\|\\
&+ \delta^{-1}\|X_{2,R:}X_2^*A_{:C}\|\left\|A_{R:}Y_2Y_{2,C:}^*\right\| + \left\|X_{2,R:}X_2^*AY_2Y_{2,C:}^*\right\|)\\
&+ \|X_2^*A\| + \|AY_2\| + \delta^{-1}\|X_2^*A_{:C}\|\,\|A_{R:}Y_2\|\,).
\end{aligned} \tag{14}$$

2.3.5. Deterministic part: Conclusion of the proof. We now have control over all three terms $\left\|B - B_{:C}B_{RC}^+B_{R:}\right\|, \left\|B_{RC}^+B_{R:}\right\|, \left\|B_{:C}B_{RC}^+\right\|$. Substitute (12), (13), (14) into (11). As the right-hand side of (14) dominates δ multiplied by the right-hand side of (12), (13), we conclude that with high probability, $\left\|A - A_{:C}B_{RC}^+A_{R:}\right\|$ is also bounded by the right-hand side of (14).

To obtain the basic bound, (4), we note that all the "normed terms" on the right-hand side of (14), e.g., $\left\|A_{R:}Y_2Y_{2,C}^*\right\|$ and $\|X_2^*A\|$, are bounded by ε_k. It follows that with high probability, $\|A - A_{:C}B_{RC}^+A_{R:}\| = O(\lambda_X\lambda_Y(\delta + \varepsilon + \delta^{-1}\varepsilon^2))$.

To obtain the other bound, (5), we need to bound each "normed term" of (14) differently. Recall (10). Consider $\|X_{2,R:}X_2^*A_{:C}\|$. We have

$$X_{2,R:}X_2^*A_{:C} = (\widetilde{X}_{2,R:}, \ldots, \widetilde{X}_{\lceil m/k\rceil,R:}) \cdot \begin{pmatrix} \widetilde{A}_{21} & \ldots & \widetilde{A}_{2,\lceil n/k\rceil} \\ \vdots & \ddots & \vdots \\ \widetilde{A}_{\lceil m/k\rceil,1} & \ldots & \widetilde{A}_{\lceil m/k\rceil,\lceil n/k\rceil} \end{pmatrix} \begin{pmatrix} \widetilde{Y}_{1,C:}^* \\ \vdots \\ \widetilde{Y}_{\lceil n/k\rceil,C:}^* \end{pmatrix}.$$

In section 2.3.1, we show that with high probability, $\|\widetilde{X}_{i,R:}\| = O(\lambda_X^{-1})$ and $\|\widetilde{Y}_{j,C:}\| = O(\lambda_Y^{-1})$ for all i, j. Recall the definition of ε_k' in (3). It follows that with high probability,

$$\|X_{2,R:}X_2^*A_{:C}\| \le \sum_{i=2}^{\lceil m/k\rceil} \sum_{j=1}^{\lceil n/k\rceil} \left\|\widetilde{X}_{i,R:}\right\| \left\|\widetilde{A}_{ij}\right\| \left\|\widetilde{Y}_{j,C:}\right\| \le \lambda_X^{-1}\lambda_Y^{-1}\varepsilon_k'.$$

Apply the same argument to other terms on the right-hand side of (14), e.g., $\|X_{2,R:}X_2^* AY_2Y_{2,C:}^*\| = O(\lambda_X^{-1}\lambda_Y^{-1}\varepsilon_k')$ and $\|X_2^*A_{:C}\| = O(\lambda_Y^{-1}\varepsilon_k')$ with high probability. Mnemonically, a R in the subscript leads to a λ_X^{-1} and a C in the subscript leads to a λ_Y^{-1}.

Recall that $\|A_{:C}B_{RC}^+A_{R:}\|$ is bounded by the right-hand side of (14). Upon simplifying, we obtain that $\|A - A_{:C}B_{RC}^+A_{R:}\| = O(\lambda_X\lambda_Y\delta + \varepsilon_k' + \lambda_X\lambda_Y\delta^{-1}{\varepsilon_k'}^2)$, i.e., (5). The proof is complete. ∎

3. Alternative sublinear-time algorithms.

3.1. Second algorithm. In Algorithm 2, uniform random sampling first helps to trim down A to two factors, $A_{:C}$ and $A_{R:}^*$ with $|C| = |R| = \ell = \widetilde{O}(k)$, and then rank-revealing QR decompositions (RRQR) are used on $A_{:C}$ and $A_{R:}^*$ to further reduce the small dimension to exactly k.

For dense matrices, the most expensive step in Algorithm 2 is the multiplication of A by $A_{R':}^+$. However, for structured matrices, the most expensive steps of Algorithm 2 are likely to be the RRQR factorization of $A_{:C}$ and $A_{R:}^*$ and the inversion of $A_{:C'}, A_{R':}$, which all take $\widetilde{O}(mk^2)$ time. The overall running time is $O(T_Ak) + \widetilde{O}(mk^2)$, where T_A is the cost of a matrix-vector multiplication.

Note that in the MATLAB code, the call `rrqr(A,k)` is assumed to return an index set of size k specifying the selected columns. One can use Algorithm 782 [6] or its MATLAB port [34].

It can be easily shown [19] that once $A_{:C'}, A_{R':}$ are fixed, the choice of $Z = A_{:C'}^+AA_{R':}^+$ is optimal in the Frobenius norm (not operator norm), that is, $Z = \arg_{W\in\mathbb{C}^{\ell\times\ell}} \|A - A_{:C'}WA_{R':}\|_F$. Unsurprisingly, the error estimate is better than in Theorem 1.1.

THEOREM 3.1. *Let A be given by* (1) *for some $k > 0$. Assume $m \ge n$ and X_1, Y_1 are μ-coherent where $\mu = \widetilde{O}(1)$ with respect to m, n. Recall the definition of ε_k in* (2). *Let $\ell \ge 10\mu k \log m$. With probability at least $1 - 4km^{-2}$, Algorithm* 2 *returns a skeleton that satisfies*

$$\|A - A_{:C'}ZA_{R':}\| = O(\varepsilon_k(mk)^{1/2}).$$

ALGORITHM 2. $O(T_A k) + \widetilde{O}(mk^2)$-time algorithm
Input: A matrix $A \in \mathbb{C}^{m\times n}$ that is approximately rank k, and user-defined parameter $\ell = \widetilde{O}(k)$. Assume $m \geq n$.
Output: Column index set C' of size k, row index set R' of size k, center matrix Z of a matrix skeleton. Implicitly, we have the matrix skeleton $A_{:C'}ZA_{R':}$.
Steps:

1. Let C be a random index set of size ℓ chosen uniformly from $\{1,\dots,n\}$. Explicitly form $A_{:C}$.
2. Let R be a random index set of size ℓ chosen uniformly from $\{1,\dots,m\}$. Explicitly form $A_{R:}$.
3. Run RRQR on $A_{:C}$ to select k columns of $A_{:C}$. Denote the result as $A_{:C'}$ where $C' \subseteq C$ indexes the k selected columns of A. This takes $\widetilde{O}(mk^2)$ time and $\widetilde{O}(mk)$ space.
4. Run RRQR on $A^*_{R:}$ to select k rows of $A_{R:}$. Denote the result as $A_{R':}$ where $R' \subseteq R$ indexes the k selected rows of A. This takes $\widetilde{O}(nk^2)$ time and $\widetilde{O}(nk)$ space.
5. Compute $Z = A^+_{:C'}(AA^+_{R':})$. This takes $O(T_A k + mk^2)$ time and $O(mk)$ space, where T_A is the time needed to apply A to a vector.

MATLAB code:

```
function [Cp,Z,Rp]=skeleton2(A,l)
    C=randperm(n,l); ind=rrqr(A(:,C),k); Cp=C(ind);
    R=randperm(m,l); ind=rrqr(A(R,:)',k); Rp=R(ind);
    Z=pinv(A(:,Cp))*(A*pinv(A(Rp,:)));
end
```

Proof. Let $P = A_{:C'}A^+_{:C'} \in \mathbb{C}^{m\times m}$. RRQR [23] selects k columns from $A_{:C}$ such that

$$\|A_{:C} - PA_{:C}\| \leq f(k,\ell)\sigma_{k+1}(A_{:C}) \leq f(k,\ell)\sigma_{k+1}(A) \leq f(k,\ell)\varepsilon_k,$$

where $f(k,\ell) := \sqrt{1+2k(\ell-k)}$. We have used the fact that $\sigma_{k+1}(A) = \sigma_{k+1}\left(\begin{smallmatrix} A_{11} & A_{12} \\ A_{21} & A_{22}\end{smallmatrix}\right) \leq \sigma_1\left(\begin{smallmatrix} A_{12} \\ A_{22}\end{smallmatrix}\right) \leq \varepsilon_k$. See interlacing theorems [26, Corollary 3.1.3].

Recall from (8) that $\|Y^+_{1,C:}\| = O((n/\ell)^{1/2})$ with probability at least $1-2km^{-2}$. Apply Corollary 2.4 to obtain that with high probability,

$$\begin{aligned}\|A - PA\| &\leq O(\lambda_Y)\,\|A_{:C} - PA_{:C}\| + O(\lambda_Y)\varepsilon_k + \varepsilon_k \\ &= O(\varepsilon_k(n/\ell)^{1/2} f(k,\ell)) = O(\varepsilon_k(nk)^{1/2}).\end{aligned}$$

Let $P' = A^+_{R':}A_{R':}$. By the same argument, $\|A - AP'\| = O(\varepsilon_k(mk)^{1/2})$ with the same failure probability. Combine both estimates. With probability at least $1-4km^{-2}$,

$$\begin{aligned}\left\|A - A_{:C'}A^+_{:C'}AA^+_{R':}A_{R':}\right\| &= \|A - PAP'\| \\ &\leq \|A - PA\| + \|PA - PAP'\| \\ &\leq \|A - PA\| + \|A - AP'\| \\ &= O(\varepsilon_k(mk)^{1/2}). \qquad \square\end{aligned}$$

ALGORITHM 3. $\widetilde{O}(nk^2)$-time algorithm
Input: A matrix $A \in \mathbb{C}^{m\times n}$ that is approximately rank k, and user-defined parameter $\ell = \widetilde{O}(k)$.
Output: Column index set C' of size k, row index set R of size ℓ, center matrix Z of a matrix skeleton. Implicitly, we have the matrix skeleton $A_{:C'}ZA_{R:}$.
Steps:

1. Let R be a random index set of size ℓ chosen uniformly from $\{1,\dots,m\}$. Explicitly form $A_{R:}$.
2. Run RRQR on $A_{R:}$ and obtain a column index set C'. Note that $A_{R:} \simeq A_{RC'}(A^+_{RC'}A_{R:})$, where $A_{RC'}$ contains k columns of $A_{R:}$. This takes $\widetilde{O}(nk^2)$ time and $\widetilde{O}(nk)$ space.
3. Compute $Z = A^+_{RC'}$. This takes $\widetilde{O}(k^3)$ time and $\widetilde{O}(k^2)$ space.

MATLAB code:

```
function [Cp,Z,R]=skeleton3(A,l)
    R=randperm(m,l); Cp=rrqr(A(R,:),k); Z=pinv(A(R,Cp));
end
```

Many algorithms that use the skeleton $A_{:C}(A^+_{:C}AA^+_{R:})A_{R:}$, e.g., in [28], seek to select columns indexed by C such that $\|A - A_{:C}A^+_{:C}A\|$ is small. Here, we further select k out of $\ell = \widetilde{O}(k)$ columns, which is also suggested in [7]. Their estimate on the error in the operator norm is $O(k\log^{1/2}k)\varepsilon_k + O(k^{3/4}\log^{1/4}k)\|A - A_k\|_F$, where A_k is the optimal rank k approximation to A. In general, $\|A - A_k\|_F$ could be as large as $(n-k)^{1/2}\varepsilon_k$, which makes our bound better by a factor of $k^{1/4}$. Nevertheless, we make the extra assumption that X_1, Y_1 are incoherent.

3.2. Third algorithm. Consider the case where *only* X_1 is $\widetilde{O}(1)$-coherent. See Algorithm 3. It computes a skeleton with $\widetilde{O}(k)$ rows and k columns in $\widetilde{O}(nk^2 + k^3)$ time. Intuitively, the algorithm works as follows. We want to select k columns of A but running RRQR on A is too expensive. Instead, we randomly choose $\widetilde{O}(k)$ rows to form $A_{R:}$, and select our k columns using the much smaller matrix $A_{R:}$. This works because X_1 is assumed to be $\widetilde{O}(1)$-coherent and choosing almost any $\widetilde{O}(k)$ rows will give us a good sketch of A.

THEOREM 3.2. *Let A be given by* (1) *for some $k > 0$. Assume $m \geq n$ and X_1 is μ-coherent where $\mu = \widetilde{O}(1)$ with respect to m, n. (Y_1 does not need to be incoherent.) Recall the definition of ε_k in* (2). *Let $\ell \geq 10\mu k\log m$. Then, with probability at least $1 - 2km^{-2}$, Algorithm* 3 *returns a skeleton that satisfies*

$$\|A - A_{:C'}ZA_{R:}\| = O(\varepsilon_k(mn)^{1/2}).$$

Proof. We perform RRQR on $A_{R:}$ to obtain $A_{R:} \simeq A_{RC'}D$, where $D = A^+_{RC'}A_{R:}$ and C' indexes the selected k columns. We want to use the second principle to "undo the row restriction" and infer that $A \simeq A_{C'}D$, the output of Algorithm 3. The details are as follows.

Strong RRQR [23] guarantees that

$$\|A_{R:} - A_{RC'}D\| \leq \sigma_{k+1}(A_{R:})f(k,n) \leq \sigma_{k+1}(A)f(k,n) \leq \varepsilon_k f(k,n)$$

and

$$\|D\| \leq f(k,n),$$

TABLE 1

	No. of rows	No. of columns	Upper bound on error in the operator norm	Running time	Memory
Algorithm 1	$\ell = \widetilde{O}(k)$	$\ell = \widetilde{O}(k)$	$O(\varepsilon_k \frac{(mn)^{1/2}}{\ell})$	$\widetilde{O}(k^3)$	$\widetilde{O}(k^2)$
Algorithm 2	k	k	$O(\varepsilon_k (mk)^{1/2})$	$O(T_A k) + \widetilde{O}(mk^2)$	$\widetilde{O}(mk)$
Algorithm 3	$\widetilde{O}(k)$	k	$O(\varepsilon_k (mn)^{1/2})$	$\widetilde{O}(nk^2)$	$\widetilde{O}(nk)$

where $f(k,n) = \sqrt{1+2k(n-k)}$. Prepare to apply a transposed version of Corollary 2.4, i.e.,

$$\|A - AP\| \le \left\|X_{1,R:}^{+}\right\| \|A_{R:} - A_{R:}P\| + \left\|X_{1,R:}^{+}\right\| \|I-P\| \, \|X_{2,R:}X_2^* A\| + \|I-P\| \, \|X_2^* A\| . \tag{15}$$

Let $P = S_{C'}D$, so that $\|P\| \le \|D\| \le f(k,n)$. Note that $AP = A_{:C'}A_{RC'}^{+}A_{R:}$. By (8), with probability at least $1-2km^{-2}$, $\|X_{1,R:}^{+}\| = O((m/\ell)^{1/2})$. By (15),

$$\begin{aligned}\|A - AP\| &\le O(\lambda_X)\,\|A_{R:} - A_{RC'}D\| + O(\lambda_X)(1+\|P\|)\varepsilon_k + (1+\|P\|)\varepsilon_k \\ &= O(\varepsilon_k f(k,n)(m/\ell)^{1/2}) = O(\varepsilon_k (mn)^{1/2}). \qquad \square\end{aligned}$$

If X_1 is not incoherent and we fix it by multiplying on the left by a randomized Fourier matrix $\mathcal{F}D$ (cf. section 1.6), then we arrive at the algorithm in [27]. The linear algebraic part of their proof combined with the first principle will lead to similar bounds. What we have done here is split the proof into three simple parts: (1) show that $\widetilde{X}_1 := \mathcal{F}DX_1$ is incoherent, (2) use the first principle to show that $\widetilde{X}_{1,R:}$ is "sufficiently nonsingular," (3) apply the second principle.

3.3. Comparison of the three algorithms. Here is a summary of the three algorithms studied in this paper. Assume $m \ge n$. Recall that $A \simeq X_1 A_{11} Y_1^*$. For Algorithms 1 and 2, assume that X_1, Y_1 are both incoherent. For Algorithm 3, assume that X_1 is incoherent. See Table 1.

Recall that T_A is the cost of applying A to a vector. If $T_A = \widetilde{O}(nk)$ and $m = O(n)$, then the running time of Algorithms 2 and 3 are comparable and we would recommend using Algorithm 2 because it has a better error guarantee.

Otherwise, if T_A is on the order of mn, then Algorithm 2 is much slower than Algorithms 1 and 3, and is not recommended. Compared to Algorithm 3, and in that scenario, Algorithm 1 is much faster and has better error guarantees, so we view it as the better choice. The advantages of Algorithm 3 are that it selects exactly k columns and does not require Y_1 to be incoherent.

If we cannot afford using $\widetilde{O}(mk)$ memory or having a running time that scales with m, n, then Algorithm 1 is the only possible choice here. Although Theorem 1.1 suggests that the error for Algorithm 1 grows with $(mn)^{1/2}$, we believe that in practice, the error usually increases with $m^{1/2}$. See section 4 for some numerical results.

Finally, we remind the reader that these recommendations are made based on error guarantees which are not always tight.

4. Examples.

4.1. First toy example: Convolution. This first example shows that in Algorithm 1 it is crucial to regularize when inverting A_{RC} because, otherwise, the error

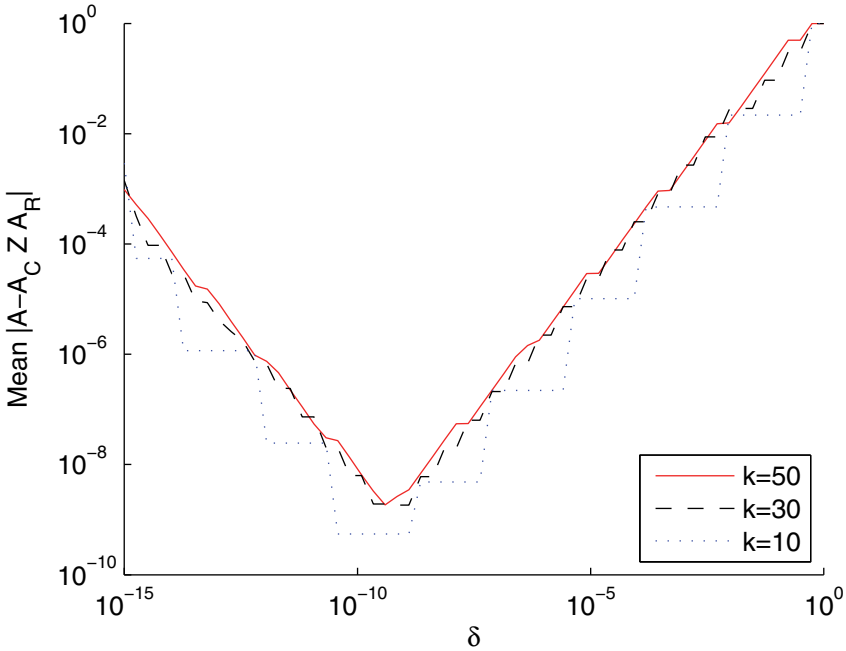

FIG. 1. *Log-log plot of the empirical mean of the error in operator norm by the $\widetilde{O}(k^3)$ algorithm versus δ, a regularization parameter. This relationship between the error and δ agrees with Theorem* 1.1. *See* (16). *More importantly, the error blows up for small δ, which implies that the regularization step should not be omitted.*

in the operator norm can blow up. In fact, even when A is positive definite and we pick $C = R$ as in the work of Gittens [21], we encounter the same need to regularize. The reason is that due to numerical errors, A_{RC} tends to be ill-conditioned when A_{RC} has more rows and columns than the rank of A. In other words, numerical errors introduce spurious small but nonzero singular values in A_{RC} and inverting the components corresponding to these small singular values leads to large errors.

The experiment is set up as follows. Let $A = X\Sigma X^* \in \mathbb{C}^{n\times n}$, where X is the unitary Fourier matrix and Σ is a diagonal matrix of singular values. Note that every entry of X is of magnitude $n^{-1/2}$ and X is 1-coherent. Fix $n = 301$, $\ell = 100$, and $k = 10, 30, 50$. Pick $\varepsilon = \varepsilon_k = \sigma_{k+1} = \cdots = \sigma_n = 10^{-15}$. Pick the largest k singular values to be *logarithmically spaced* between 1 and ε. Note that A is Hermitian and positive definite. In each random trial, we randomly shuffle the singular values, pick ℓ random rows and columns, and measure $\|A - A_{:C}ZA_{R:}\|$. The only parameters being varied are k and δ. Note that although $R \neq C$ in this experiment, similar results are obtained when $R = C$.

From (4) in Theorem 1.1, we expect that when variables such as n, m, ℓ, k are fixed,

$$\log \|A - A_{:C}ZA_{R:}\| \sim \log(\delta^{-1}(\varepsilon_k + \delta)^2) = -\log\delta + 2\log(\varepsilon_k + \delta). \tag{16}$$

Consider a plot of $\|A - A_{:C}ZA_{R:}\|$ versus δ on a log-log scale. According to the above equation, when $\delta \ll \varepsilon_k$, the first term dominates and we expect to see a line of slope -1, and when $\delta \gg \varepsilon_k$, $\log(\varepsilon_k + \delta) \simeq \log\delta$, and we expect to see a line of slope $+1$. Indeed, when we plot the experimental results in Figure 1, we see a right-angled

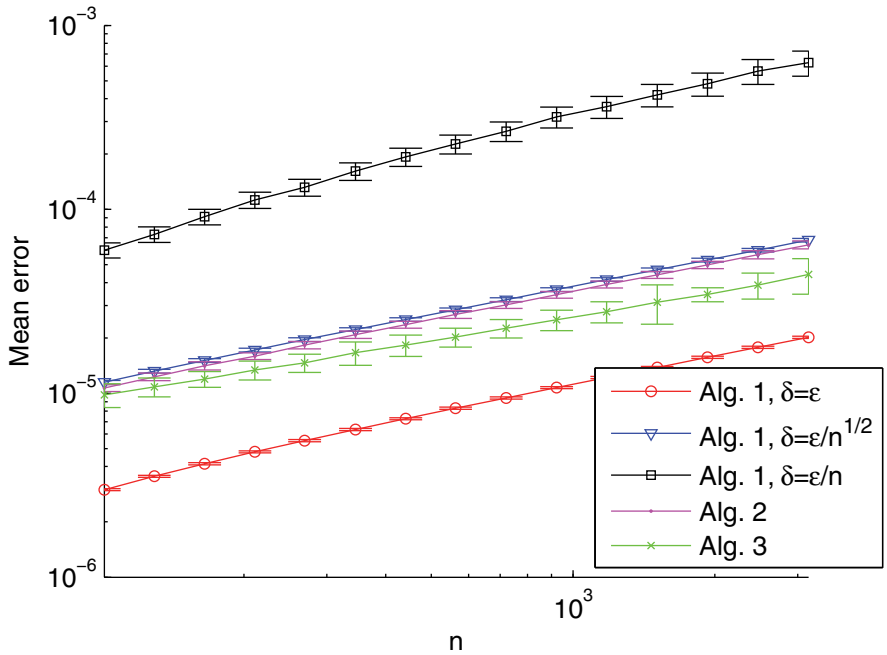

FIG. 2. *The above is a log-log plot of the empirical mean of the error in operator norm versus n, the size of square matrix A. Fix $k = 10$, $\ell = 40$, and $A = X\Sigma Y^*$, where X, Y are unitary Fourier matrices with randomly permuted columns and Σ is the diagonal matrix of singular values. The top k singular values are set to 1 and the others are set to $\varepsilon = 10^{-6}$. When we run Algorithm 1 with $\delta = \varepsilon, \varepsilon/\sqrt{n}, \varepsilon/n$, the expected errors seem to grow with $n^{0.55}, n^{0.51}, n^{0.69}$, respectively. For Algorithms 2 and 3, the expected errors seem to grow with $n^{0.52}, n^{0.43}$, respectively. The errorbars correspond to $\frac{1}{5}$ of the standard deviations obtained empirically. Observe that the error in Algorithm 3 fluctuates much more than Algorithm 1 with $\delta = \varepsilon, \varepsilon/\sqrt{n}$.*

V-curve.

The point here is that the error in the operator norm can blow up as $\delta \to 0$.

A curious feature of Figure 1 is that the error curves resemble staircases. As we decrease k, the number of distinct error levels seems to decrease proportionally. A possible explanation for this behavior is that the top singular vectors of $A_{:C}$ match those of A, and as δ increases from $\sigma_i(A)$ to $\sigma_{i-1}(A)$ for some small i, smaller components will not be inverted and the error is all on the order of $\sigma_i(A)$.

4.2. Second toy example. For the second experiment, we consider $A = X\Sigma Y^*$ where X, Y are unitary Fourier matrices with randomly permuted columns and Σ is the diagonal matrix of singular values. Fix $k = 10$, $\ell = 40$. The singular values are set such that the largest k singular values are all 1 and the other singular values are all $\varepsilon = 10^{-6}$. We consider all three algorithms. For Algorithm 1, we set δ in three different ways: $\delta = \varepsilon$, $\delta = \varepsilon/\sqrt{n}$, and $\delta = \varepsilon/n$.

We plot the error $\|A - A_{:C}ZA_{R:}\|$ versus n in Figure 2. The numerical results show that if we pick $\delta = \varepsilon/\sqrt{n}$ for Algorithm 1, then the estimated mean error is almost the same as that of Algorithm 2—they both scale with $n^{0.51}$, with k, ℓ fixed. On the other hand, if we pick $\delta = \varepsilon$ as suggested by (4) of Theorem 1.1, the expected error seems to grow with $n^{0.55}$, which is slightly worse than Algorithm 2 but much better than described in (6).

The expected error of Algorithm 3 seems to grow with $n^{0.43}$, which is the best in

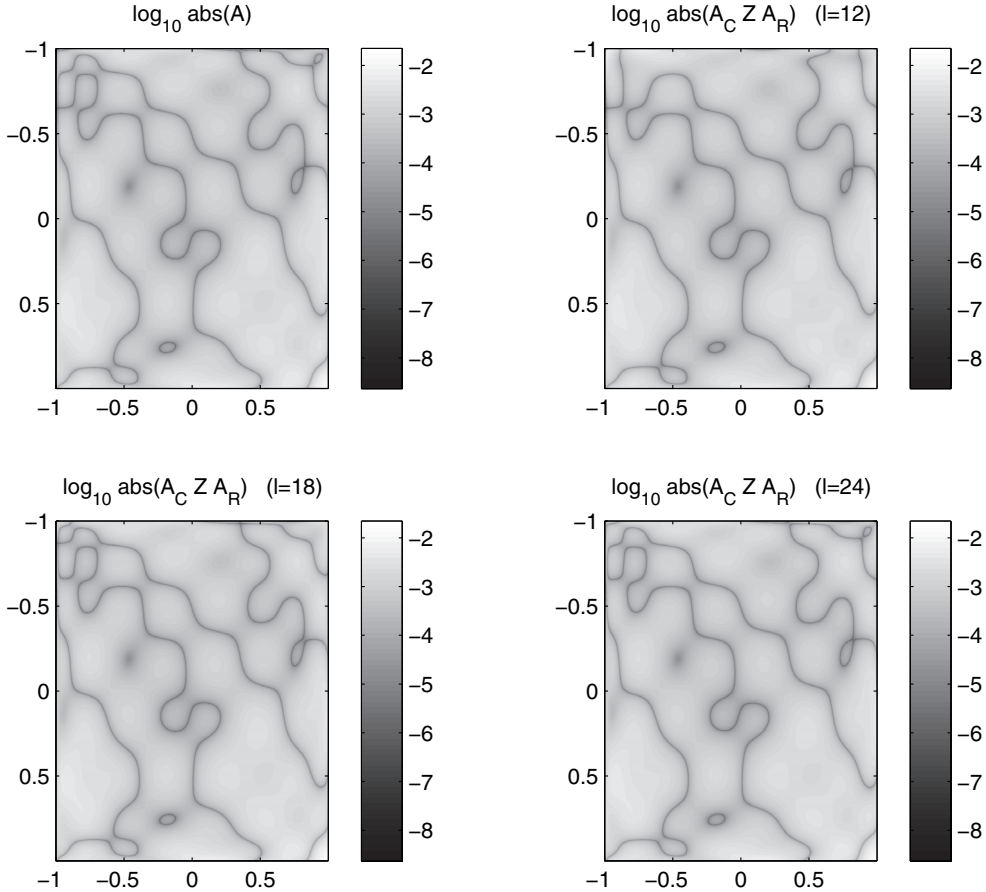

FIG. 3. *A is the smooth kernel $K(x,y)$ where K is the sum of 6^2 low degree Chebyshev polynomials evaluated on a $10^3 \times 10^3$ uniform grid. The top, left figure is A while the other figures show that the more intricate features of A start to appear as we increase ℓ from 12 to 18 to 24. Recall that we sample ℓ rows and ℓ columns in the $\widetilde{O}(k^3)$ algorithm.*

this experiment. However, compared with Algorithms 2 and 1 with $\delta = \varepsilon, \varepsilon/\sqrt{n}$, its error is not as concentrated around the mean.

4.3. Smooth kernel. Consider a one-dimensional integral operator with a kernel K that is analytic on $[-1,1]^2$. Define A as $(A)_{ij} = cK(x_i, y_j)$, where the nodes $x_1, \ldots, x_n$ and $y_1, \ldots, y_n$ are *uniformly spaced* in $[-1,1]$. First, suppose $K = \sum_{1 \le i,j \le 6} c_{ij} T_i(x) T_j(y) + 10^{-3} T_{10}(x) T_{10}(y) + 10^{-9} N$, where $T_i(x)$ is the ith Chebyshev polynomial and N is the random Gaussian matrix, i.e., noise. The coefficients c_{ij}'s are chosen such that $\|A\| \simeq 1$. Pick $n = m = 10^3$ and slowly increase ℓ, the number of rows and columns sampled by the $\widetilde{O}(k^3)$ algorithm. As shown in Figure 3, the skeleton representation $A_{:C} Z A_{R:}$ converges rapidly to A as we increase ℓ.

Next, consider $K(x,y) = c \exp(xy)$. Let $n = 900$ and pick c to normalize $\|A\| = 1$. We then plot the empirical mean of the error of the $\widetilde{O}(k^3)$ algorithm against ℓ on the left of Figure 4. Notice that the error decreases exponentially with ℓ.

To understand what is happening, imagine that the grid is infinitely fine. Let $\varphi_1, \varphi_2, \ldots$ be Legendre polynomials. Recall that these polynomials are orthogonal on

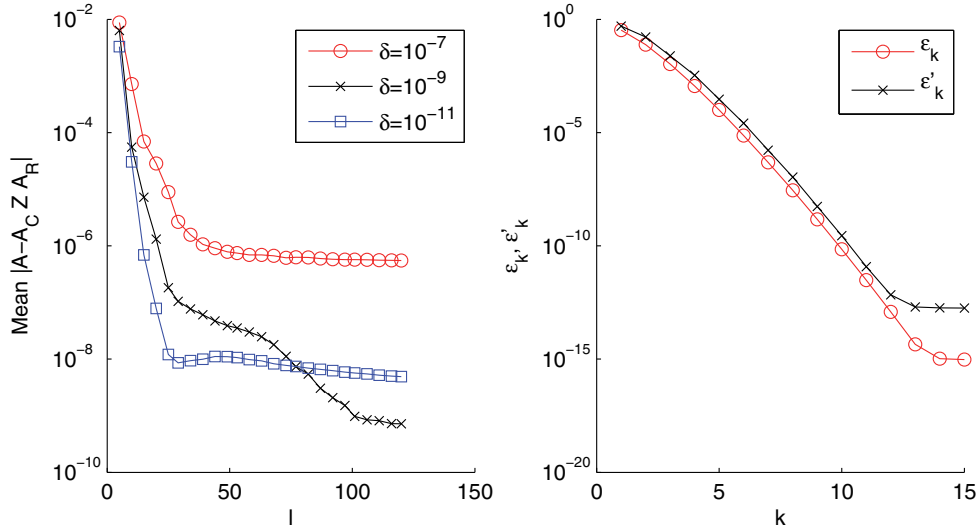

FIG. 4. *A is the smooth kernel $K(x,y) = \exp(-xy)$ sampled on a uniform grid. The graph on the left shows that the error of the $\widetilde{O}(k^3)$ algorithm decreases exponentially with ℓ, the number of sampled rows and columns. The figure on the right shows that if we expand A in terms of Legendre polynomials, the coefficients (and therefore $\varepsilon_k, \varepsilon'_k$) decay exponentially. See* (1), (2), *and* (3) *for the definitions of ε_k and ε'_k.*

$[-1,1]$. Define the matrices X, Y as $(X)_{ij} = \varphi_j(x_i)$ and $(Y)_{ij} = \varphi_j(y_i)$. Assume the φ_i's are scaled such that X, Y are unitary. It is well known that if we expand K in terms of Chebyshev polynomials or Legendre polynomials [8] or prolate spheroidal wave functions [40], the expansion coefficients will decay exponentially. This means that the entries of X^*AY should decay exponentially away from the topleft corner and $\varepsilon'_k = \Theta(\varepsilon_k)$ (cf. (2) and (3)). We confirm this by plotting $\varepsilon_k, \varepsilon'_k$ versus k on the right of Figure 4. The actual X, Y used to obtain this plot are obtained by evaluating the Legendre polynomials on the uniform grid and orthonormalizing. It can be verified that the entries of X, Y are of magnitude $O((k/n)^{1/2})$, which implies a coherence $\mu \simeq k$, independent of n. The implication is that the algorithm will continue to perform well as n increases.

As ℓ increases, we can apply Theorem 1.1 with a larger k. Since $\varepsilon_k, \varepsilon'_k$ decrease exponentially, the error should also decrease exponentially. However, as k increases beyond $\simeq 15$, ε_k stagnates and nothing can be gained from increasing ℓ. In general, as ε_k decreases, we should pick a smaller δ. But when $k \gtrsim 15$, choosing a smaller δ does not help and may lead to worse results due to the instability of pseudoinverses. This is evident from Figure 4.

A recent paper by Platte, Trefethen, and Kuijlaars [30] states that we cannot have an exponential decrease of the error without a condition number that grows exponentially. In our case, the random selection of columns and rows corresponds to selecting interpolation points randomly, and δ serves as a regularization parameter of the interpolation method. Due to the regularization, we can only expect an exponential decrease of the error up to a limit dependent on δ.

4.4. Fourier integral operators. In [12], Candes, Demanet, and Ying consider how to efficiently apply two-dimensional Fourier integral operators of the form

$$Lf(x) = \int_{\xi} a(x,\xi) e^{2\pi i \Phi(x,\xi)} \hat{f}(\xi) d\xi,$$

where $\hat{f}(\xi)$ is the Fourier transform of f, $a(x,\xi)$ is a smooth amplitude function, and Φ is a smooth phase function that is homogeneous, i.e., $\Phi(x,\lambda\xi) = \lambda\Phi(x,\xi)$ for any $\lambda > 0$. Say there are N^2 points in the space domain and also the frequency domain.

The main idea is to split the frequency domain into $\sqrt{N}$ wedges, perform a Taylor expansion of $\Phi(x,\cdot)$ about $|\xi|\,\hat{\xi}_j$ where j indexes a wedge, and observe that the residual phase $\Phi_j(x,\xi) := \Phi(x,\xi) - \Phi(x,|\xi|\,\hat{\xi}_j)\cdot\xi$ is nonoscillatory. Hence, the matrix $A^{(j)}_{st} := \exp(2\pi i \Phi_j(x_s,\xi_t))$ can be approximated by a low rank matrix, i.e., $\exp(2\pi i\Phi_j(x,\xi))$ can be written as $\sum_{q=1}^{r} f_q(x) g_q(\xi)$ where r, the separation rank, is independent of N. By switching the order of summations, the authors arrive at $\widetilde{O}(N^{2.5})$ algorithms for both the preprocessing and the evaluation steps. See [12] for further details.

What we are concerned with here is the approximate factorization of $A^{(j)}$. This is a N^2 by $N^{1.5}$ matrix since there are N^2 points in the space domain and $N^2/\sqrt{N}$ points in one wedge in the frequency domain. In [12], a slightly different algorithm is proposed:

1. Uniformly and randomly select ℓ rows and columns to form $A_{R:}$ and $A_{:C}$.
2. Perform SVD on $A_{:C}$. Say $A_{:C} = U_1\Lambda_1 V_1^* + U_2\Lambda_2 V_2^*$, where U, V are unitary and $\|\Lambda_2\| \le \delta$, a user specified parameter.
3. Return the low rank representation $U_1 U_{1,R:}^{+} A_{R:}$.

In the words of the authors, "this randomized approach works well in practice although we are not able to offer a rigorous proof of its accuracy, and expect one to be nontrivial" [12].

We are now in a position to explain why this randomized approach works well. Consider (1) and (2). Let B be a perturbation of A such that $B_C = U_1\Lambda_1 V_1^*$ and $\|A - B\| \le \delta$. Since Λ_1 is invertible, the output can be rewritten as

$$U_1 U_{1R}^{+} A_{R:} = B_{:C} B_{RC}^{+} A_{R:}.$$

By following the proof of Theorem 1.1, we see that

$$\left\|A - B_{:C} B_{RC}^{+} A_{R:}\right\| = O(\left\|B - B_{:C} B_{RC}^{+} B_{R:}\right\|)$$

and that all the estimates in Theorem 1.1 must continue to hold.

The analysis presented here, therefore, answers the questions posed in [12]. We believe that the assumption of incoherence of the generating vectors is precisely the right framework to express the error guarantees of the skeleton in such situations.

An important subclass of Fourier integral operators is pseudodifferential operators. These are linear operators with pseudodifferential symbols that obey certain smoothness conditions [35]. In [16], a similar randomized algorithm is used to derive low rank factorizations of such smooth symbols. It is likely that the method works well here in the same way as it works well for a smooth kernel as discussed in the previous section.

Acknowledgments. We would also like to thank the anonymous referees for making many valuable suggestions.

REFERENCES

[1] N. Ailon and B. Chazelle, *Approximate nearest neighbors and the fast Johnson-Lindenstrauss transform*, in Proceedings of the 38th Annual ACM Symposium on Theory of Computing, ACM, New York, 2006, pp. 557–563.

[2] H. Avron, P. Maymounkov, and S. Toledo, *Blendenpik: Supercharging LAPACK's least-squares solver*, SIAM J. Sci. Comput., 32 (2010), pp. 1217–1236.

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

[4] M. Bebendorf and R. Grzhibovskis, *Accelerating Galerkin BEM for linear elasticity using adaptive cross approximation*, Math. Methods Appl. Sci., 29 (2006), pp. 1721–1747.

[5] M. Bebendorf and W. Hackbusch, *Existence of H-matrix approximants to the inverse FE-matrix of elliptic operators with ℓ^∞-coefficients*, Numer. Math., 95 (2003), pp. 1–28.

[6] C. H. Bischof and G. Quintana-Ortí, *Algorithm 782: Codes for rank-revealing QR factorizations of dense matrices*, ACM Trans. Math. Software, 24 (1998), pp. 254–257.

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

[8] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*, Dover, Mineola, NY, 2001.

[9] J. Bremer, *A fast direct solver for the integral equations of scattering theory on planar curves with corners*, J. Comput. Phys., 231 (2012), pp. 1879–1899.

[10] E. J. Candès and B. Recht, *Exact matrix completion via convex optimization*, Found. Comput. Math., 9 (2009), pp. 717–772.

[11] E. Candès and J. Romberg, *Sparsity and incoherence in compressive sampling*, Inverse Problems, 23 (2007), pp. 969–985.

[12] E. J. Candès, L. Demanet, and L. Ying, *Fast computation of Fourier integral operators*, SIAM J. Sci. Comput., 29 (2007), pp. 2464–2493.

[13] T. F. Chan, *Rank revealing QR factorizations*, Linear Algebra Appl., 88 (1987), pp. 67–82.

[14] E. W. Cheney and K. H. Price, *Minimal projections*, in Approximation Theory, A. Talbot., ed., Academic Press, New York, 1970, pp. 261–289.

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

[16] L. Demanet and L. Ying, *Discrete symbol calculus*, SIAM Rev., 53 (2011), pp. 71–104.

[17] L. Demanet and L. Ying, *Fast wave computation via Fourier integral operators*, Math. Comp., 81 (2012), pp. 1455–1486.

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

[19] S. Friedland and A. Torokhti, *Generalized rank-constrained matrix approximations*, SIAM J. Matrix Anal. Appl., 29 (2007), pp. 656–659.

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

[21] A. Gittens, *The spectral norm error of the naive Nystrom extension*, arXiv:1110.5305, 2011.

[22] S. A. Goreinov, E. E. Tyrtyshnikov, and N. L. Zamarashkin, *A theory of pseudoskeleton approximations*, Linear Algebra Appl., 261 (1997), pp. 1–21.

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

[24] N. Halko, P. G. Martinsson, and J. A. Tropp, *Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions*, SIAM Rev., 53 (2011), pp. 217–288.

[25] K. L. Ho and L. Greengard, *A fast direct solver for structured linear systems by recursive skeletonization*, SIAM J. Sci. Comput., 34 (2012), pp. A2507–A2532.

[26] R. A. Horn and C. R. Johnson, *Topics in Matrix Analysis*, Cambridge University Press, Cambridge, UK, 1994.

[27] E. Liberty, F. Woolfe, P.-G. Martinsson, V. Rokhlin, and M. Tygert, *Randomized algorithms for the low-rank approximation of matrices*, Proc. Nat. Acad. Sci. USA, 104 (2007), pp. 20167–20172.

[28] M. W. Mahoney and P. Drineas, *CUR matrix decompositions for improved data analysis*, Proc. Nat. Acad. Sci. USA, 106 (2009), pp. 697–702.

[29] S. Negahban and M. J. Wainwright, *Restricted strong convexity and weighted matrix completion: Optimal bounds with noise*, Mach. Learn. Res., 13 (2012), pp. 1665–1697.

[30] R. B. Platte, L. N. Trefethen, and A. B. J. Kuijlaars, *Impossibility of fast stable approximation of analytic functions from equispaced samples*, SIAM Rev., 53 (2011), pp. 308–318.

[31] M. Rudelson, *Random vectors in the isotropic position*, J. Funct. Anal., 164 (1999), pp. 60–72.
[32] M. Rudelson and R. Vershynin, *Sampling from large matrices: An approach through geometrical functional analysis*, J. ACM, 54 (2007), 21.
[33] A. F. Ruston, *Auerbach's theorem and tensor products of Banach spaces*, in Proc. Cambridge Philos. Soc. 58, Cambridge University Press, Cambridge, UK, 1962, pp. 476–480.
[34] J. Saak and P. Benner, *Efficient solution of large scale Lyapunov and Riccati equations arising in model order reduction problems*, Proc. Appl. Math. Mech., 8 (2008), pp. 10085–10088.
[35] M. A. Shubin, *Pseudodifferential Operators and Spectral Theory*, Springer-Verlag, Berlin, 2001.
[36] A. Talwalkar and A. Rostamizadeh, *Matrix coherence and the Nystrom method*, arXiv:1004.2008, 2010.
[37] J. A. Tropp, *Improved analysis of the subsampled randomized Hadamard transform*, Adv. Adapt. Data Anal., 3 (2011), pp. 115–126.
[38] E. Tyrtyshnikov, *Incomplete cross approximation in the mosaic-skeleton method*, Computing, 64 (2000), pp. 367–380.
[39] F. Woolfe, E. Liberty, V. Rokhlin, and M. Tygert, *A fast randomized algorithm for the approximation of matrices*, Appl. Comput. Harmon. Anal., 25 (2008), pp. 335–366.
[40] H. Xiao, V. Rokhlin, and N. Yarvin, *Prolate spheroidal wavefunctions, quadrature and interpolation*, Inverse Problems, 17 (2001), pp. 805–838.