Superfast CUR Approximation of Low Rank Matrices

Victor Y. Pan\textsuperscript{[1,2,3],[a]}, Qi Luan\textsuperscript{[2],[b]}, John Svadlenka\textsuperscript{[3],[c]}, and Liang Zhao\textsuperscript{[1,3],[d]}

\textsuperscript{[1]} Department of Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA

\textsuperscript{[2]} Ph.D. Program in Mathematics
The Graduate Center of the City University of New York
New York, NY 10036 USA

\textsuperscript{[3]} Ph.D. Program in Computer Science
The Graduate Center of the City University of New York
New York, NY 10036 USA

\textsuperscript{[a]} victor.pan@lehman.cuny.edu
http://comet.lehman.cuny.edu/vpan/

\textsuperscript{[b]} qi-luan@yahoo.com

\textsuperscript{[c]} jsvadlenka@gradcenter.cuny.edu

\textsuperscript{[d]} lzhao1@gc.cuny.edu

Abstract

Low rank approximation of a matrix (hereafter referred to as LRA) is a fundamental subject of numerical linear algebra and data mining and analysis and is an enormously popular research subject of the last decade. The arithmetic time and memory space of the classical SVD-based optimal solution and of the faster algorithms of the 1990s have been dramatically decreased. We achieve new substantial progress by relying on our novel techniques but even more so on our new insight into the problem.

We study LRA in a transparent form of CUR where $C$ and $R$ are submatrices made up of $l$ input columns and $k$ input rows, respectively, the integers $k$ and $l$ are small relatively to the input dimensions, and an $l \times k$ matrix $U$ unifies the submatrices $C$ and $R$ in the product.

Cross-approximation algorithms routinely compute accurate CUR LRAs by using much fewer memory cells and arithmetic operations than the input matrix has entries. We call such algorithms superfast, assuming that they are also superefficient in using memory space. So far, however, no one proved that any superfast algorithm outputs accurate LRA of the worst case input.

We first show why such a proof has been elusive: for any matrix size $m \times n$ we specify a family of $2mn$ matrices such that for almost 50\% of them any superfast LRA algorithm fails to improve the poor trivial approximation by the matrix filled with zeros.

\begin{flushleft}
\textsuperscript{*}Some results of this paper have been presented at the Workshop on Fast Direct Solvers, November 12–13, 2016, Purdue University, West Lafayette, Indiana; the SIAM Conference on Computational Science and Engineering, February–March 2017, Atlanta, Georgia, USA, and the INdAM Meeting – Structured Matrices in Numerical Linear Algebra: Analysis, Algorithms and Applications, Cortona, Italy, September 4–8, 2017. Also see [PLSZ16].
\end{flushleft}
Then in contrast, however, we prove that the class of all such hard inputs is narrow – cross-approximation and even more primitive superfast algorithms compute reasonably accurate LRAs

(i) to any matrix allowing close LRA except for small norm perturbations of matrices of an algebraic variety of a smaller dimension,

(ii) to the average matrix allowing close LRA,

(iii) to the average sparse matrix allowing close LRA and

(iv) with a high probability to any matrix allowing close LRA if it is pre-processed fast with a Gaussian, SRHT or SRFT multiplier\(^1\).

Moreover empirically the output LRAs remain accurate when we perform the computations superfast by replacing such a multiplier with one of our sparse and structured multipliers.

Our study provides new insights into CUR LRA and reveals, demonstrates and enhances the power of simplified heuristics, the average case analysis, randomized pre-processing and cross-approximation. Our work should encourage further exploration of these techniques and their wider application.

A part of our progress is due to our synergistic combination of the techniques proposed in the communities of Numerical Linear Algebra and Computer Science. This should help bring together the research efforts in the two communities.

Our auxiliary results can be of independent interest. For example, we improve a decade-old estimate for the norm of the inverse of a Gaussian matrix, prove such an estimate also in the case of sparse Gaussian matrices, describe a very simple superfast algorithm for the transition from any LRA to CUR LRA and present some novel advanced pre-processing techniques for fast and superfast computation of LRA.

Fast LRA algorithms are applied to various highly important computational areas, for example, to tensor decomposition. Our progress should lead to more efficient applications. Moreover superfast LRA provides new opportunities, not available for fast LRA, as we demonstrate by means of dramatic acceleration of the Conjugate Gradient algorithms and the bottleneck stage of the construction of low rank generators for the Fast Multipole Method.

**Keywords:** Low rank approximation, Complexity, CUR approximation, Average matrices, Sketches, Cross-approximation, Maximal volume, Subspace sampling, Pre-processing, Gaussian random matrices, Fast Multipole Method, Conjugate Gradient algorithms.

# 1 Introduction

## 1.1 LRA: fast and superfast algorithms and hard inputs

Low rank approximation of a large matrix\(^2\) is a fundamental subject of numerical linear algebra and data mining and analysis with applications to image processing, noise reduction, seismic inversion, latent semantic indexing, principal component analysis, machine learning, regularization for ill-posed problems, web search models, tensor decomposition, system identification, signal processing, neuroscience, computer vision, social network analysis, antenna array processing, electronic design automation, telecommunications and mobile communication, chemometrics, psychometrics, biomedical engineering, and so on (see [CML15], [HMT11], [M11], [KS16], [KB09], [DMM08], [MMD08], [MD09], [OT10], [ZBD15]).

\(^1\)Here and hereafter we abbreviate “standard Gaussian random” by writing just “Gaussian”, while “SRHT” and “SRFT” stand for “subsampled randomized Hadamard and Fourier transforms”, respectively.

\(^2\)Hereafter we keep using the acronym LRA.
We call LRA algorithms fast if they use about as many as (or at least not much more) memory cells and flops than an input matrix has entries, but a small number of cross-approximation iterations, aka adaptive C-A iterations, routinely output close LRAs by involving much fewer memory cells and flops. We call such algorithms superfast, assuming that they are also superefficient in using small storage space. The empirical observation that superfast C-A algorithms routinely produce accurate LRAs, however, had no formal support so far, unlike the case of various fast LRA algorithms.

We explain why such a support has been elusive. First define the set of $\pm\delta$-matrices filled with zeros except for a single entry filled with $\pm 1$ (there are exactly $2^{mn}$ such $m \times n$ matrices). Then fix any superfast algorithm and observe that its computed approximations to almost 50% of these sparse inputs are no better than the poor trivial approximation by the matrix filled with zeros. Moreover define the rank-1 matrix filled with ones, add it to the $\pm\delta$-matrices and arrive at a family of $2^{mn}$ dense inputs, which are as hard for any superfast LRA algorithm as the $\pm\delta$-matrices.

1.2 CUR – a transparent LRA

We prove, however, that such hard inputs of superfast C-A LRA algorithms form a narrow subclass of the class of all matrices having small numerical rank, that is, matrices allowing close LRA. First recall a special transparent version of LRA called CUR approximation. Let the $r \times r$ leading principal block $W_{r,r}$ of an $m \times n$ matrix $W$ be nonsingular and well-conditioned. Call it a CUR generator, call its inverse $U = W_{r,r}^{-1}$ a nucleus, write $C$ and $R$ for the submatrices made up of the first $r$ columns and the first $r$ rows of the matrix $W$, respectively, and approximate that matrix by the rank-$r$ matrix CUR.

Given a pair of row and column sets defining the factors $C$ and $R$, we can compute a nucleus $U$ and arrive at a CUR approximation by using $r^2$ memory cells and $O(r^3)$ flops. The matrices $C$, $U$ and $R$ together occupy $(m + n + r)r$ memory cells; fewer cells and flops are involved if the matrix $W$ is sparse. These bounds are dramatically smaller than the number of the entries of the matrix $W$ if the ratio $r / \min\{m, n\}$ is small.

It is easy to prove that $CUR = W$ if rank($W$) = $r$ and unless a CUR generator degenerates into a singular matrix. Likewise $CUR \approx W$ if $W$ allows close rank-$r$ approximation and unless a CUR generator degenerates into an ill-conditioned, that is, a nearly singular, matrix. Such a degeneration only occurs for small norm perturbations of matrices that form an algebraic variety of a smaller dimension in the space of all matrices $W$ of rank $r$.

Intuitively a random matrix expected to have numerical rank $r$ is unlikely to lie in or near such a variety, and then we are likely to obtain a close CUR LRA to such a matrix superfast – simply by building it on a fixed $r \times r$ CUR generator, say, on the $r \times r$ leading principal block, or on a random $r \times r$ submatrix.

---

3 Here and hereafter “flop” stands for “floating point arithmetic operation”, and we estimate the usage of memory cells up to smaller order terms.

4 Hereafter we use the acronym C-A for cross-approximation; we describe C-A algorithms in Sections 1.4 and 7.3 and analyze them in Sections 7.4, 8, 9 and 10.

5 The pioneering papers [GZT95], [GTZ97], [GTZ97a], [GT01], [GOSTZ10], [M14], and [O16] deal with the nucleus $G$ and CGR approximation, where “G” can stand for “germ” or “generator”. We use the more customary acronym CUR, where “U” can stand, say, for “unification factor”, and notice the alternatives of CNR, CCR, or CSR with $N$, $C$, and $S$ standing for “nucleus”, “core”, and “seed”, respectively.

6 Theoretically we can decrease the exponent 3 below 2.38, by applying fast matrix multiplication, but see [P17] on limitations of its practical impact.
1.3 Superfast primitive algorithm and its success with CUR LRA of random, average and average sparse matrices

We formally support that intuition. Given an \( m \times n \) matrix \( W \) of numerical rank \( r \ll \min\{m, n\} \) (that is, allowing its rank-\( r \) approximation), fix its any \( r \times r \) submatrix, view it as a CUR generator and build on it a CUR approximation. Clearly this primitive algorithm is superfast, but we prove that with a high probability its output CUR LRA is reasonably accurate for a random and a random sparse input matrix \( W \) of numerical rank \( r \), and so it is reasonably accurate on the average input (see our definitions in Section 2.2). Moreover a posteriori error estimation and thus correctness verification for a CUR LRA can be performed superfast as well (see Section 3.4 and the end of Section 4.2).

1.4 Sketching approach, cynical and cross-approximation algorithms

By extending the primitive algorithm we can choose a CUR generator being a random \( k \times l \) submatrix of \( W \); the larger the size \( k \times l \) the less likely the algorithm fails to produce a close CUR LRA. It would even less likely to fail if we output a \( k \times l \) generator by applying an LRA algorithm to a random \( q \times s \) submatrix (sketch) of \( W \) (cf. [S06], [RV07], [W14]) for

\[
0 < r \leq k \leq q \leq m \quad \text{and} \quad r \leq l \leq s \leq n. \tag{1}
\]

For \( q = k \) and \( s = l \) this is just the primitive algorithm. Otherwise such algorithms are still quite primitive; we call them cynical,\(^7\) they are superfast if \( qs \min\{q, s\} \ll mn \).

We can enhance the power of cynical algorithms by applying them recursively and alternating their vertical and horizontal \( q \times s \) input sketches until we obtain a sufficiently close LRA. By following [T00] we call such loops cross-approximation or C-A.

Overall \( u \) C-A iteration loops involve \((mk + nl)u\) memory cells and \( O((m + n)(k + l)^2u)\) flops. So the computation is superfast unless \( u \), the number of loops, grows to or beyond the level of \( \min\{m, n\}/(k + l)^2 \), but already a single two-step C-A loop outputs an accurate LRA on the average input and with a high probability on a random input (see Section 10).

1.5 Three criteria for accurate CUR approximation

Each of the following three CUR criteria is sufficient for obtaining accurate CUR LRA, although they imply different output error bounds:

1. a CUR generator and an input matrix share numerical rank \( r \) (see Corollary 22),\(^8\)

2. a \( k \times l \) CUR generator has maximal volume among all \( k \times l \) submatrices of an input matrix (see Sections 8 – 10)\(^9\) and

3. the column set \( C \) of a CUR approximation and its row set \( R \) (defining a CUR generator and a nucleus \( U \)) are sampled according to properly pre-computed sampling probabilities, aka leverage scores (see Section 11 or [DMM08], [BW17], [SWZ17]).

\(^7\)We allude to the benefits of the austerity and simplicity of primitive life, advocated by Diogenes the Cynic, and not to shamelessness and distrust associated with modern cynicism.

\(^8\)This CUR criterion is also necessary in a small neighborhood of inputs of rank-\( r \) (see Theorem 11).

\(^9\)A \( k \times l \) matrix \( M \) has volume \((\max\{|\det(MM^*)|, |\det(M^*M)|\})^{1/2}\), which is, \( |\det(M)| \) for \( k = l \).
We prove that the CUR generators computed by the primitive, cynical and C-A superfast algorithms satisfy the first two CUR criteria for any input matrix allowing LRA except for small-norm perturbations of matrices forming an algebraic variety of a smaller dimension. We further prove that with a high probability these generators satisfy the two CUR criteria for random and random sparse inputs matrix allowing LRA and hence satisfy them on the average. The output CUR LRAs are accurate accordingly.

Based on the third CUR criterion we prove similar properties of the superfast variation of the randomized algorithm of [DMM08].

Surely the average input does not always represent real world computations, but the results of our tests for some benchmark real world inputs are in good accordance with the results of our formal study.

1.6 Fast and superfast pre-processing of LRA inputs

We further extend our results for a random input matrix to any input matrix having low numerical rank and multiplied fast by a Gaussian, SRHT or SRFT matrix. Namely, by extending the random sampling techniques of [HMT11], we prove that with a high probability C-A and some other superfast algorithms output accurate CUR LRA of such a matrix product. The algorithm is superfast except for its pre-processing stage of multiplication by Gaussian, SRHT or SRFT multiplier. Empirically we match this output accuracy when we apply the same algorithms and multiply the input matrix superfast with random sparse and structured matrices of certain classes. Recall that we cannot achieve more than that in view of our example of $\pm \delta$ matrices.

We also proved that having a crude but reasonably close LRA, typically devised by researchers in Numerical Linear Algebra (NLA), we can refine it superfast. Namely we can apply this LRA in order to bypass the initial fast stage of the otherwise superfast CUR LRA algorithm of [DMM08] proposed by researchers in Computer Science. This can be an example of synergistic combination of NLA and CS research efforts. Another, more minor example is our Algorithm 24a for transition from top SVD to CUR LRA, which completes superfast transition from any LRA to CUR LRA.

1.7 Brief summary of our progress and its extensions

We thoroughly study C-A and other superfast LRA algorithms. We begin with describing a family of hard inputs on which all superfast LRA algorithms fail, but then we supply the missing formal support for the output accuracy of C-A algorithms routinely observed in practice and provide some directions to enhancing it, simple recipes for its superfast correctness verification and new insights into the subject. Moreover we complement our formal analysis with numerical tests and propose various technical novelties of independent interest.

Our work reveals, demonstrates and enhances the power of various LRA techniques.

Our progress should motivate broader application of simplified heuristics, average case analysis, sketching, devising recursive C-A-like subspace sampling in alternate directions, randomized pre-processing and synergistic alliance of NLA and CS research communities.

Our techniques and auxiliary results can be of independent interest. For example, we improve a decade-old estimate for the norm of the inverse of a Gaussian matrix (see Remark 105 and Theorem 112 and compare a challenge stated in [SST06]), describe a very simple superfast algorithm for the
transition from any LRA to CUR LRA and present some novel advanced pre-processing

 techniques for fast and superfast computation of LRA. In Appendix D.4 we explore a novel
 numerical approach by Osinsky in [O16] to enhancing the output accuracy of numerical
 algorithms for LRA. In Section 11.5 we extend our techniques to superfast computation of
 the Lewis weights, involved in [SWZ17].

 In Section 16 we demonstrate new LRA applications based on using superfast LRA: we
 dramatically accelerate the Conjugate Gradient algorithms and the bottleneck stage of the
 construction of low rank generators for the Fast Multipole celebrated Method.

 Empirically we can extend our study to a larger area. Whenever our analysis covers
 small neighborhoods of rank-$r$ matrices, we can tentatively apply the algorithms to a larger
 neighborhood and then estimate a posteriori errors superfast. For another example, we
 simplify our analysis and exposition by assuming that numerical rank is given to us, but
 frequently we can compute it as by-product or by using binary search.

 In Section 15 we list other natural research directions and recall related works.

1.8 Organization of the paper

 In PART I of our paper, made up of the next six sections, we define CUR LRA and some
 small families of its hard inputs, estimate its output errors in terms of the norms of the
 factors $C$, $U$ and $R$, describe some superfast CUR LRA algorithms and prove, based on the
 first CUR criterion, that they are accurate for random inputs (with a high probability) and
 average inputs.

 In Section 2 we recall basic definitions for our study of general, random and average
 matrices.

 In Section 3 we state the LRA problem and its known complexity bounds, describe a
 class of hard inputs for LRA and propose superfast techniques for its a posteriori error
 estimation.

 In Section 4 we introduce CUR approximation, recall its benefits and the complexity of
 its known algorithms and deduce some basic estimates for its output errors.

 In Section 5 we define CUR generators and canonical CUR approximation, express its
 nucleus through its generator and estimate the output error norm of a canonical CUR
 approximation in terms of the norms of the three CUR factors. This verifies our first CUR
 criterion for supporting accurate CUR LRAs.

 In Section 6 we estimate the output error norms of CUR approximation of random and
 the average input matrices that allow close LRA.

 In Section 7 we briefly study primitive, cynical, and C-A algorithms.

 In PART II of our paper, made up of Sections 8 – 10, we study CUR LRA based on the
 concept of the matrix volume and our second CUR criterion for supporting accurate LRAs.

 In Section 8 we recall that concept and its upper bounds and estimate the volumes of a
 perturbed matrix and a matrix product.

 In Section 9 we study the impact of volume maximization on the accuracy of CUR
 approximation, which defines our second CUR criterion, and we again conclude that the
 superfast LRA algorithms of Section 7 are accurate on a random and the average input.

 In Section 10 we maximize the volumes of CUR generators and decrease the output
 error bounds by means of C-A.

 In PART III of our paper, made up of Sections 11 and 12, we study randomized tech-

 niques for CUR LRA, in particular based on our third CUR criterion for supporting accurate
LRAs and on randomized pre-processing.

In Section 11 we recall fast algorithm of [DMM08], which computes highly accurate LRA defined by sampling probabilities (thus fulfilling our third CUR criterion for supporting accurate LRAs). We show that the algorithm is superfast (i) for such an average input, (ii) with a high probability for a random input allowing LRA and (iii) for any input given with its reasonably good LRA. We also discuss some directions for the improvement of the efficiency of that algorithm.

In Section 12 we study randomized multiplicative pre-processing for LRA.

In PART IV of our paper, made up of Sections 13 – 16, we cover the results of our numerical tests, summarize our study, discuss its extensions and recall some related works and in particular the known estimates for the accuracy and complexity of LRA algorithms.

In Section 13 we present the results of our numerical experiments, which are in good accordance with our formal study.

In Section 14 we list the known estimates for the accuracy and complexity of LRA algorithms.

In Section 15 we summarize the results of our study, list our technical novelties and natural research directions and recall some related works.

In Section 16 we discuss some implications and extensions of our progress, in particular to the acceleration of the Fast Multipole Method and Conjugate Gradient algorithms.

In the APPENDIX we cover various auxiliary results, some of independent interest.

In Appendix A we recall and in one case improve the known estimate for the ranks of random matrices and the norms of Gaussian matrices and their pseudo inverses and extend the latter estimates to sparse Gaussian matrices.

In Appendix B we describe the well-known Hadamard and Fourier multipliers and our abridged variations.

In Appendix C we recall the sampling and rescaling algorithms of [DMM08] and some results showing their efficiency.

In Appendix D we recall some efficient CUR algorithms that can be used as subalgorithms of cynical algorithms, revisit the greedy cross-approximation iterations of [GOSTZ10] based on LUP factorization and devise their QRP counterparts.

PART I. CUR LRA: DEFINITIONS, HARD INPUTS, SUPERFAST ALGORITHMS AND ERROR BOUNDS

2 Basic Definitions and Properties

Hereafter the concepts “large”, “small”, “near”, “close”, “approximate”, “ill-conditioned” and “well-conditioned” are usually quantified in context. “≫” and “≪” mean “much greater than” and “much less than”, respectively.

2.1 General matrix computations

Recall some basic definitions for matrix computations (cf. [ABBB99], [GL13]).

\( \mathbb{C}^{m \times n} \) is the class of \( m \times n \) matrices with complex entries.

\( I_s \) denotes the \( s \times s \) identity matrix. \( O_{q,s} \) denotes the \( q \times s \) matrix filled with zeros.
\[
diag(B_1, \ldots, B_k) = \diag(B_j)_{j=1}^k \text{ denotes a } k \times k \text{ block diagonal matrix with diagonal blocks } B_1, \ldots, B_k.
\]

\((B_1 | \ldots | B_k)\) and \((B_1, \ldots, B_k)\) denote a \(1 \times k\) block matrix with blocks \(B_1, \ldots, B_k\).

\(W^T\) and \(W^*\) denote the transpose and the Hermitian transpose of an \(m \times n\) matrix \(W = (w_{ij})_{i,j=1}^{m,n}\), respectively. \(W^* = W^T\) if the matrix \(W\) is real.

\(\mathcal{N}_W\) denotes the number of nonzero entries of a matrix \(W\).

For two sets \(I \subseteq \{1, \ldots, m\}\) and \(J \subseteq \{1, \ldots, n\}\) define the submatrices

\[
W_{I,:} := (w_{i,j})_{i \in I, j = 1, \ldots, n}, \quad W_{:,J} := (w_{i,j})_{i = 1, \ldots, m; j \in J}, \quad \text{and } W_{I,J} := (w_{i,j})_{i \in I, j \in J}.
\]

\[
||W|| = ||W||_2, \quad ||W||_F, \quad \text{and } ||W||_C \text{ denote the spectral, Frobenius and Chebyshev norms of a matrix } W, \text{ respectively,}
\]

\[
||W||_F^2 := \sum_{i,j=1}^{m,n} |w_{ij}|^2 = \text{rank}(W) \sum_{j=1}^{\text{rank}(W)} \sigma_j^2(W), \quad ||W||_C := \max_{i,j=1}^{m,n} |w_{ij}|,
\]

\[
||W||_h := \sup_{|v| \neq 0} ||Wv||_h/||v||, \text{ for } h = 1, 2, \infty,
\]

such that (see [GL13, Section 2.3.2 and Corollary 2.3.2])

\[
||W||_C \leq ||W|| \leq ||W||_F \leq \sqrt{mn} ||W||_C, \quad ||W||_F^2 \leq \min\{m, n\} ||W||^2,
\]

\[
||W||_1 = ||W^T||_\infty = \max_{j=1,\ldots,n} \sum_{i=1}^{m} |w_{ij}|,
\]

\[
\frac{1}{\sqrt{m}} ||W||_1 \leq ||W|| \leq \sqrt{n} ||W||_1, \quad ||W||_2 \leq ||W||_1 ||W||_\infty.
\]

Represent a matrix \(W = (w_{i,j})_{i,j}^{m,n}\) as a vector \(w\) and define the \(l_1\)-norm

\[
||W||_{l_1} := ||w||_1 = \sum_{i,j} |w_{i,j}|
\]

(cf. [SWZ17]). Observe that \(||W||_F = ||w||\), and so

\[
||W||_F \leq ||W||_{l_1} \leq \sqrt{mn} ||W||_F.
\]

An \(m \times n\) matrix \(W\) is unitary (also orthogonal when real) if \(W^*W = I_n\) or \(WW^* = I_m\).

\(U = W(I)\) is a left inverse of \(W\) if \(UW = I_n\) (and then \(\text{rank}(W) = n\)) and its right inverse if \(WU = I_m\) (and then \(\text{rank}(W) = m\)). \(W(I) = W^{-1}\) if a matrix \(W\) is nonsingular.

\[
W = S_W \Sigma_W T_W^*.
\]

denotes its compact SVD, hereafter referred to just as SVD, such that

\[
S_W^*S_W = T_W^*T_W = I_{\text{rank}(W)}, \quad \Sigma_W := \diag(\sigma_j(W))_{j=1}^{\text{rank}(W)}.
\]

\(\sigma_j(W)\) denotes the \(j\)th largest singular value of \(W\) for \(j = 1, \ldots, \text{rank}(W)\),

\[
\sigma_j(W) = 0 \text{ for } j > \text{rank}(W), \quad ||W|| = \sigma_1(W) \text{ and } ||W||_F^2 = \sum_{j=1}^{\text{rank}(W)} \sigma_j^2(W).
\]
\(\tilde{\sigma}_{r+1} := \tilde{\sigma}_{r+1}(W) := \min_{\text{rank}(W') \leq r} |W - W'| \) for any \(r\)

(9)

and recall that

\(\tilde{\sigma}_{r+1} = \sigma_{r+1}(W)\) under the spectral norm

(10)

(this is the Eckart–Young theorem, see [GL13, page 79]) and

\[(\tilde{\sigma}_{r+1})^2 = \sum_{j=r+1}^{\text{rank}(W)} \sigma_j^2(W) \leq (\text{rank}(W) - r) \sigma_{r+1}^2(W) \text{ under the Frobenius norm.}\]

(11)

\(W^+ := T_W \Sigma_W^{-1} \Sigma_W^*\) is the Moore–Penrose pseudo inverse of an \(m \times n\) matrix \(W\), which is its left inverse if \(\text{rank}(W) = n\) and its right inverse if \(\text{rank}(W) = m\).

\(\kappa(W) := \sigma_1(W)/\sigma_\rho(W) = |W|/|W^+| \geq 1\) denotes its condition number.

A matrix \(W\) is unitary if and only if \(\kappa(W) = 1\); it is ill-conditioned if \(\kappa(W)\) is large in context, and it is well-conditioned if \(\kappa(W)\) is reasonably bounded. It has \(\epsilon\)-rank at most \(r > 0\) for a fixed tolerance \(\epsilon > 0\) if there is a matrix \(W'\) of rank \(r\) such that \(\|W' - W\|/\|W\| \leq \epsilon\).

A matrix \(W\) has numerical rank \(r\) if it has \(\epsilon\)-rank \(r\) for a small \(\epsilon\), and then we write \(\text{rank}(W) = r\). (A matrix is ill-conditioned if and only if it has a matrix of a smaller rank nearby. A matrix is well-conditioned if and only if its rank is equal to its numerical rank.)

A matrix \(W\) is said to be the rank-\(r\) truncation of the matrix \(\tilde{W}\) of a larger rank if it is obtained by setting to 0 all but the \(r\) largest singular values of the matrix \(\tilde{W}\).

If the matrix \(\tilde{W}\) is given with its SVD \(\tilde{W} = \tilde{S} \tilde{\Sigma} \tilde{T}^T\), then its top SVD of rank \(r\) is given by \(W = S \Sigma T^*\) where \(W\) is the rank-\(r\) truncation of \(\tilde{W}\) and \(S\), \(\Sigma\) and \(T\) are submatrices of the matrices \(\tilde{S}\), \(\tilde{\Sigma}\) and \(\tilde{T}\), respectively.

If the matrix \(\tilde{W}\) has numerical rank \(r\), then we define its SVD-truncation \(W\) and its top SVD assuming by default that \(\text{rank}(W) = r\).

2.2 Random and average matrices

Hereafter “i.i.d.” stands for “independent identically distributed”, \(\mathbb{E}(v)\) for the expected value of a random variable \(v\), and \(\text{Var}(v)\) for its variance.

Definition 1. Gaussian matrices.

(i) Call an \(m \times n\) matrix \(W\) Gaussian and write \(W \in \mathcal{G}^{m \times n}\) if its entries are i.i.d. Gaussian variables.

(ii) Call a matrix \(W\) \(\mathcal{N\mathcal{Z}}\)-Gaussian and write \(W \in \mathcal{G}_{\mathcal{N\mathcal{Z}}}^{m \times n}\) if it is filled with zeros except for at most \(\mathcal{N\mathcal{Z}}\) entries, filled with i.i.d. Gaussian variables.

Remark 2. An \(\mathcal{N\mathcal{Z}}\)-Gaussian matrix is either rank deficient identically in all its Gaussian entries or has full rank with probability 1 (see Theorem 100).

Lemma 3. (Orthogonal invariance of a Gaussian matrix,) Suppose \(k\), \(m\), and \(n\) are three positive integers, \(G\) is an \(m \times n\) Gaussian matrix, \(S\) and \(T\) are \(l \times m\) and \(n \times k\) orthogonal matrices, respectively, and \(k \leq \min\{m, n\}\). Then \(SG\) is a Gaussian matrices if \(m \leq n\) while \(GT\) is a Gaussian matrices if \(m \geq n\).
Definition 4. Factor-Gaussian and \( \mathcal{N} \mathcal{Z} \)-factor-Gaussian matrices. (i) Call a matrix \( W = G \Sigma H \) a diagonally scaled factor-Gaussian matrix of expected rank \( r \) and write \( W \in \mathcal{G}^{m \times n}_r(\Sigma) \) if \( G \in \mathcal{G}^{m \times r}_r, H \in \mathcal{G}^{r \times n}_r, \Sigma = \text{diag}(\sigma_i)_{i=1}^r, \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0 \) and unless the ratio \( \sigma_1/\sigma_r \) is large (see Figure 1). In the special case where \( \Sigma = \sigma I_r \) call the matrix \( W \) a scaled factor-Gaussian matrix of expected rank \( r \) and write \( \sigma := \sigma_1 = \sigma_r \) and \( W \in \mathcal{G}^{m \times n}_r. \)

(ii) Call an \( m \times n \) matrix \( W = GH \) a left factor-Gaussian matrix of expected rank \( r \) and write \( W \in \mathcal{G}^{m \times n}_{r,H} \) if \( G \in \mathcal{G}^{m \times r}_r, \) \( H \in \mathcal{G}^{r \times n}_r \) and the matrix \( H \) is well-conditioned.

(iii) Call an \( m \times n \) matrix \( W = GH \) a right factor-Gaussian matrix of expected rank \( r \) and write \( W \in \mathcal{G}^{m \times n}_{G,r} \) if \( G \in \mathcal{G}^{m \times r}_r, \) \( H \in \mathcal{G}^{r \times n}_r \) and the matrix \( G \) is well-conditioned.

(iv) Call the matrices \( W \) of parts (i)–(iii) diagonally scaled, scaled, left and right \( \mathcal{N} \mathcal{Z} \)-factor-Gaussian and write

\[
W \in \mathcal{G}^{m \times n}_{r,NZ_G,NZ_H}(\Sigma), \quad W \in \mathcal{G}^{m \times n}_{r,NZ_G,NZ_H}, \quad W \in \mathcal{G}^{m \times n}_{G,r,NZ_H} \text{ and } W \in \mathcal{G}^{m \times n}_{N \mathcal{Z}_G,r,H},
\]

respectively, if they are defined by \( \mathcal{N} \mathcal{Z} \)-Gaussian rather than Gaussian factors \( G \) and/or \( H \).

A submatrix of a well-conditioned matrix of full rank can be ill-conditioned or rank deficient, but clearly a submatrix of a Gaussian matrix is Gaussian, a submatrix of an \( \mathcal{N} \mathcal{Z} \)-Gaussian matrix is \( \mathcal{N} \mathcal{Z} \)-Gaussian, and we readily verify the following results.

Theorem 5. (i) A submatrix of a diagonally scaled (resp. scaled) factor-Gaussian matrix of expected rank \( r \) is a diagonally scaled (resp. scaled) factor-Gaussian matrix of expected rank \( r \) and (ii) a \( k \times n \) (resp. \( m \times 1 \)) submatrix of an \( m \times n \) left (resp. right) factor-Gaussian matrix of expected rank \( r \) is a left (resp. right) factor-Gaussian matrix of expected rank \( r \).

An \( m \times n \) matrix having a numerical rank at most \( r \) is a small norm perturbation of the product \( AB \) of two matrices \( A \in \mathbb{C}^{m \times r} \) and \( B \in \mathbb{C}^{r \times n} \). Together with Definition 4 this motivates the following definitions of the \( m \times n \) average matrices of rank \( r \leq \min\{m,n\} \).

Definition 6. The average matrices allowing LRA. Define the average \( m \times n \) matrices \( W \) of a rank \( r \) in four ways – as the diagonally scaled average, scaled average, left average and right average – by taking the average over the matrices of the classes \( \mathcal{G}^{m \times n}_r(\Sigma), \mathcal{G}^{m \times n}_r(\sigma I_r), \mathcal{G}^{m \times n}_{r,H} \) and \( \mathcal{G}^{m \times n}_{G,r} \) of Definition 4, respectively. Namely fix (up to scaling by a constant) non-Gaussian matrices \( \Sigma, \sigma I, G \) and \( H \) involved in the definition (see Remark 8 below) and take the average under the Gaussian probability distribution over the i.i.d. entries of the Gaussian factors \( G \) and/or \( H \). Similarly define the four classes of \( \mathcal{N} \mathcal{Z} \)-average matrices of rank \( r \) and the average and \( \mathcal{N} \mathcal{Z} \)-average matrices allowing rank-\( r \) approximation within a fixed norm bound \( \delta \).

Definition 7. Norms and expected values. Write \( \nu_{p,q} := \|G\|, \nu_{p,q,C} := \|G\|_C, \nu_{p,q}^+ := \|G^+\| \) and \( \nu_{p,q,C}^+ := \|G^+\|_C \) for a \( p \times q \) Gaussian matrix \( G \) and a pair of positive integers \( p \) and \( q \) and notice that \( \nu_{p,q} = \nu_{q,p} \) and \( \nu_{p,q}^+ = \nu_{q,p}^+ \). Write \( \nu_{p,q,NZ} := \|G\| \) and \( \nu_{p,q,NZ}^+ := \|G^+\| \) for a \( \mathcal{N} \mathcal{Z} \)-Gaussian \( p \times q \) matrix \( G \), for a fixed integer \( \mathcal{N} \mathcal{Z} \) and all pairs of \( p \) and \( q \).
Remark 8. Gaussian matrices tend to have small norms, but this is no obstacle for using them in Definitions 4 and 6 because we can scale the matrices $W = GH$ and $W = G\Sigma H$ at will by scaling the non-Gaussian factors $G$, $\Sigma$ or $H$ involved.

3 LRA Problem, Hard Inputs and Superfast Auxiliary Computations

3.1 LRA problem: definition

Rank-$r$ approximation $W'$ of an $m \times n$ matrix $W$ is given by a pair of matrices $A$ of size $m \times r$ and $B$ of size $r \times n$ such that

$$W = W' + E \quad \text{and} \quad |E| \leq \delta \quad \text{for} \quad W' := AB$$

and a fixed non-negative tolerance $\delta$ (see Figure 2).

![Figure 2: LRA of a matrix](image)

The transition $W \rightarrow AB$ dramatically decreases both storage space (from $mn$ to $(m+n)r$) and the number of flops involved in the computations, for example, from $(2n-1)m$ to $2(m+n)r - m - r$ for multiplication by a vector. The next challenge is the computation of accurate LRA within about the same time and space bounds.

We can only satisfy (12) for $\delta \geq \tilde{\sigma}_{r+1}$ (see (9)), and we assume that the ratios $\delta/\|W\|$, $\tilde{\sigma}_{r+1}(W)/m$ and $\tilde{\sigma}_r/\tilde{\sigma}_n$ are small, which is typically the case in applications of LRA to numerical integration, PDEs and integral equations.

How accurate and how costly is the computation of such LRA? We cover these issues in Section 14. The known estimates show no superfast algorithms for the worst case input, and no such algorithm can handle even a family of $2mn$ inputs of the next subsection.

3.2 A family of hard inputs for superfast LRA algorithms

Example 9. $\pm\delta$-matrices are the $2mn$ matrices $W$ of size $m \times n$ filled with zeros except for a single entry filled with the values $\pm 1$.

If an LRA algorithm applied to a matrix $W$ does not involve its $(i,j)$th entry $w_{ij}$, then $\|W' - W\|_C \geq |w'_{ij} - w_{ij}| \geq 1$ for some $\pm\delta$-matrix. This is no better than the trivial approximation by the matrix filled with zeros.

An LRA algorithm avoids such a failure only if it involves all $\mathcal{N}\mathcal{Z}_W$ input entries, and then it involves no less than $0.5\mathcal{N}\mathcal{Z}_W$ flops because a flop accesses at most two entries. Therefore every superfast LRA algorithm fails on the family of $\pm\delta$-matrices.
Moreover the algorithm can leave the failure undetected: let it be applied to a small
norm perturbation of a $\delta$-matrix $W$ with $w_{i,j} = \pm 1$ for a pair of integers $i$ and $j$. Unless the
algorithm involves the entry $w_{i,j}$, it would optimize LRA over the nearly vanishing input
text entries but would fail to approximate the entry $w_{i,j}$ and would never detect this failure.

Even if an algorithm could remove all nearly vanishing entries at no cost, the above
problems would persist for small norm perturbations of the sums of $\pm \delta$-matrices with the
rank-1 matrix filled with 1s.

In contrast to these disappointing examples, we describe some superfast auxiliary com-
putations for LRA in the rest of this section.

### 3.3 Superfast transition from an LRA to top SVD

Next, assuming that we are given an LRA of a matrix, we approximate superfast its top
SVD (see again Figure 1); moreover the algorithm works for a little more general LRA than
that of (12).

**Algorithm 10. Superfast transition from an LRA to top SVD.**

**INPUT:** Three matrices $A \in \mathbb{C}^{m \times l}$, $V \in \mathbb{C}^{l \times k}$, $B \in \mathbb{C}^{k \times n}$ and $W \in \mathbb{C}^{m \times n}$ such that

$$W = AVB + E, \quad ||E|| = O(\tilde{\sigma}_r + 1), \quad r \leq \min\{k, l\}, \quad k \ll m \text{ and } l \ll n \text{ for } \tilde{\sigma}_{r+1} \text{ of (9)}.$$

**OUTPUT:** Three matrices $S \in \mathbb{C}^{m \times r}$ (unitary), $\Sigma \in \mathbb{C}^{r \times r}$ (diagonal) and $T^* \in \mathbb{C}^{r \times n}$ (uni-
tary) such that $W = S\Sigma T^* + E'$ for $||E'|| = O(\tilde{\sigma}_r + 1)$.

**COMPUTATIONS:**

1. Compute QRP rank-revealing factorizations

$$A = (Q \mid E_{m,l-r})RP \quad \text{and} \quad B = P'R' \left( Q' \begin{pmatrix} Q' \\ E_{k-r,n} \end{pmatrix} \right)$$

where $Q \in \mathbb{C}^{m \times r}$, $Q' \in \mathbb{C}^{r \times n}$ and $||E_{m,l-r}|| + ||E_{k-r,n}|| = O(\tilde{\sigma}_r + 1)$. Substitute
the expressions for $A$ and $B$ into the matrix equation $W = AVB + E$ and obtain

$$W = QUQ' + E' \quad \text{where} \quad U = RPVP'R' \in \mathbb{C}^{r \times r} \text{ and } ||E'|| = O(\tilde{\sigma}_r + 1).$$

2. Compute SVD $U = S\Sigma T^*$. Output the $r \times r$ diagonal matrix $\Sigma$.

3. Compute and output the unitary matrices $S = Q\tilde{S}$ and $T^* = \tilde{T}^*Q'$.

This superfast algorithm uses $ml + lk + kn$ memory cells and $O(ml^2 + nk^2)$ flops.

### 3.4 Superfast a posteriori error estimation for LRA

Randomized algorithms of [HMT11, Sections 4.3 and 4.4] estimate a posteriori error super-
fast for every matrix allowing LRA. At the end of Section 5.2 we describe a simple alternative
deterministic superfast algorithm. In both cases the estimates involve the norm of an input
matrix or its submatrices and the smallest error norm of its rank-$r$ approximation.

In our distinct superfast randomized a posteriori error estimation below we do not
assume that this information is available. Instead we assume that the error matrix $E$ of an
LRA has enough entries, say, 100 or more, and that they are the observed i.i.d. values of a
single random variable. This is realistic, for example, where the deviation of the matrix $W$
from its rank-$r$ approximation is due to the errors of measurement or rounding.
In this case the Central Limit Theorem implies that the distribution of the variable is close to Gaussian (see \[EW07\]). Fix a pair of integers \( q \) and \( s \) such that \( qs \) is large enough (say, exceeds 100), but \( qs = O((m + n)k) \) and hence \( qs \ll mn \), and then apply our tests just to a random \( q \times s \) submatrix of the \( m \times n \) error matrix.

Under this policy we compute the error matrix at a dominated arithmetic cost in \( O((m + n)kl) \) but still verify correctness with high confidence, by applying the customary rules of hypothesis testing for the variance of a Gaussian variable.

Namely suppose that we have observed the values \( g_1, \ldots, g_K \) of a Gaussian random variable \( g \) with a mean value \( \mu \) and a variance \( \sigma^2 \) and that we have computed the observed average value and variance

\[
\mu_K = \frac{1}{K} \sum_{i=1}^{K} |g_i| \quad \text{and} \quad \sigma_K^2 = \frac{1}{K} \sum_{i=1}^{K} |g_i - \mu_K|^2,
\]

respectively. Then, for a fixed reasonably large \( K \), both

\[
\text{Probability}\{ |\mu_K - \mu| \geq t|\mu| \} \quad \text{and} \quad \text{Probability}\{ |\sigma_K^2 - \sigma^2| \geq t\sigma^2 \}
\]

converge to 0 exponentially fast as \( t \) grows to the infinity (see \[C46\]).

4 CUR Approximation, Its Definitions, Benefits, and Basic Error Estimates

4.1 Definitions and benefits of CUR approximation

For an \( m \times n \) matrix \( W \) of numerical rank \( r \) we fix two integers \( k \) and \( l \) such that

\[
0 < r \leq k \leq m \quad \text{and} \quad r \leq l \leq n \tag{13}
\]

and a small positive \( \delta \) and seek an LRA \( W' \) of \( W \) satisfying bound (12) and restricted to the following special form, called CUR (aka CGR and Pseudo-skeleton) approximation,

\[
W' := CUR, \quad W = W' + E, \quad |E| \leq \delta. \tag{14}
\]

Here \( C \) and \( R \) are \( m \times l \) and \( k \times n \) submatrices of the matrix \( W \), made up of its \( l \) columns and \( k \) rows, respectively, \( U \) is an \( l \times k \) matrix, said to be the nucleus of a CUR approximation \( W' \), and we call equation (14) a CUR decomposition if

\[
W = W' = CUR. \tag{15}
\]

We call the \( k \times l \) submatrix \( W_{k,l} \) shared by the matrices \( C \) and \( R \) a CUR generator.

**Theorem 11.** (i) Equation (15) implies that

\[
\text{rank}(C) = \text{rank}(R) = \text{rank}(W) = \text{rank}(W_{k,l}). \tag{16}
\]

(ii) For a sufficiently small positive \( \delta \) equation (14) implies that

\[
\text{nrank}(C) = \text{nrank}(R) = \text{nrank}(W) = \text{nrank}(W_{k,l}). \tag{17}
\]
Proof. Write \( r := \text{rank}(W) \) and \( r' := \text{rank}(W_{k,l}) \).

Deduce from (15) that \( \text{rank}(C) = \text{rank}(R) = r \geq r' \).

It remains to show that \( r' \geq r \) in order to deduce equation (16).

Apply Gauss-Jordan elimination to the matrix \( W_{k,l} \), transforming it into a diagonal matrix \( \tilde{W}_{k,l} \) with \( r' \) nonzero entries.

Extend this elimination to the matrix \( W \). The images of \( W' \), \( C' \) and \( R' \) of the matrices \( W \), \( C \) and \( R \) keep rank \( r \), and so the matrix \( C \) has at least \( r \) nonzero columns and the matrix \( R \) has at least \( r \) nonzero rows versus \( r' \) nonzero columns and rows of \( W_{k,l} \).

It follows that \( r = \text{rank}(W_{k,l}) \geq r' + 2(r - r') \), and so \( r' = r \), which proves claim (i).

Reduce claim (ii) to claim (i) for the SVD-truncation for the matrix \( W \).

Theorem implies that equation (17) is a necessary condition for a matrix \( W_{k,l} \) to be a CUR generator. In Section 5.2 we prove that this is also a sufficient condition for computing a close CUR approximation (even when we restrict it to its canonical version of then next section).

Given a CUR approximation (14) we can obtain an LRA of (12) by computing \( G := CU \) and writing \( H := R \), and we frequently refer to CUR approximations as CUR LRAs.

The matrix \( W' \) of (14) can be multiplied by a vector superfast – by using \( 2(ml + kl + kn) - m - k - l \) flops, that is, \( 2(m + n + r - 2)r - m \) for \( k = l = r \).

For fixed integers \( k \), \( l \), and \( r \) satisfying (13) we seek an \( l \times k \) nucleus \( U \) and a pair of submatrices \( C \) and \( R \) made up of \( l \) columns and \( k \) rows of an \( m \times n \) matrix \( W \), respectively, such that the matrix \( CUR \) defines an optimal LRA up to a reasonable factor \( f \).

The algorithms of [GE96], [P00], [BW17] and [SWZ17] achieve this goal but are not superfast (see their complexity estimates in Section 14).

4.2 Output errors of a CUR LRA: basic estimates

Our next task is the estimation of the spectral and Frobenius error norms \( |E| \) of a CUR LRA in terms of the norms of the factors \( C \), \( U \) and \( R \) and of the perturbation of the factor \( U \) caused by the transition to the SVD-truncation of an input matrix \( W \).

**Theorem 12.** Suppose that for a fixed spectral or Frobenius matrix norm \( | \cdot | \) we are given five integers \( k, l, m, n, \) and \( r \) satisfying (13), an \( m \times n \) matrix \( W \), its rank-\( r \) approximation \( \tilde{W} \) within a norm bound \( \tilde{\delta} \) not exceeded by the value \( \tilde{\sigma}_{r+1} \) of equations (9)–(11), and \( k \times l \) CUR generators \( W_{k,l} = \tilde{W}_{k,l} \) and \( \tilde{W}_{k,l} = \tilde{W}_{k,l} \) for \( W \) and \( \tilde{W} \), respectively, both defined by the same index sets \( I \) and \( J \), and nuclei \( U \) and \( \tilde{U} \). Write

\[
C := W_{U,J}, \quad R := W_{I,J}, \quad \tilde{C} := \tilde{W}_{U,J}, \quad \text{and} \quad \tilde{R} := \tilde{W}_{I,J}.
\]

Then

\[
|W - CUR| \leq |R| + |C| + \tilde{\delta} |\tilde{U}| \tilde{\delta} + |C| |R| |U - \tilde{U}| + \tilde{\delta}.
\]

**Proof.** Notice that

\[
\tilde{C}\tilde{U}\tilde{R} - CUR = (\tilde{C} - C)\tilde{U}\tilde{R} + C\tilde{U}(\tilde{R} - R) + C(\tilde{U} - U)R
\]

and that \( \tilde{W} = \tilde{C}\tilde{U}\tilde{R} \) by virtue of Theorem 13. Therefore

\[
|\tilde{W} - CUR| \leq |\tilde{C} - C| |\tilde{U}| |\tilde{R}| + |C| |\tilde{U}| |\tilde{R} - R| + |C| |\tilde{U} - U| |R|.
\]

Substitute the bound \( \max\{|C - \tilde{C}|, |R - \tilde{R}|\} \leq |W - \tilde{W}| = \tilde{\delta} \) and obtain (19). \( \square \)
5 Canonical CUR approximation

5.1 Computation of a nucleus from a CUR generator

Theorem 12 bounds the error norm $|W - CUR|$ in terms of the value $\tilde{\delta}$ and the norms $|C|$, $|R|$, $|\tilde{U}|$ and $|\tilde{U} - U|$ where $\max\{|C|, |R|\} \leq |W|$. Next we cover generation of the nucleus $U$ and in Section 5.2 estimate the norms $|\tilde{U} - U|$ and $|\tilde{U}|$ in terms of $|U|$ and $\tilde{\delta} = \tilde{\sigma}_{r+1}$.

First consider the case of CUR decomposition (15) for $k = l = r$ (see Figure 3).

Figure 3: CUR decomposition with a nonsingular generator

**Theorem 13.** Suppose that $k = l = r$ and the matrices $W$, $C = W_{r,r}$ and $R = W_{I,:}$ have rank $r$ (cf. (16)). Then the following two matrix equations imply one another:

$$W = CUR$$

and

$$U := W_{r,r}^{-1}.$$  (21)

Proof. Let (21) hold and without loss of generality let $W_{r,r} = W_{r,r}$ for $I = J = \{1, \ldots, r\}$ (see (2)). Then $CU = W_{r,r}U = (\mathbf{I}_r \mid X)^T$ for a $r \times (m - r)$ matrix $X$, and so

$$CUR = W_{r,r}UW_{I,:} = (\mathbf{I}_r \mid X)^TW_{I,:}.$$  

Hence the matrices $W$ and $CUR$, both of rank $r$, share their first $r$ rows. Likewise they share their first $r$ columns, and hence $W = CW_{r,r}^{-1}R = CUR$. This implies (20).

Conversely equation (20) implies that $CU = WR^{(I)}$. Since $\text{rank}(C) = r$, this matrix equation has unique solution $U$, which is $W_{r,r}^{-1}$, as we can readily verify. \hfill \square

Next we extend expression (21) to any triple of $k$, $l$ and $r$ satisfying (13).

Unless $k = l = r$ a nucleus of a CUR decomposition is not uniquely defined because we can add to it any $l \times k$ matrix orthogonal to one or both of the factors $C$ and $R$. Hereafter (except for Section 11) we narrow our study to canonical CUR decomposition (and similarly canonical CUR approximation) by generating its nucleus from a candidate CUR generator $W_{k,l}$ in two steps as follows.

**Algorithm 14.** Computation of a canonical nucleus.

Input: Three integers $k$, $l$ and $r$, $0 < r \leq \min\{k,l\}$, and a $k \times l$ CUR generator $W_{k,l}$.

Output: An $l \times k$ nucleus $U$.

Computations: 1. Compute the SVD-truncation $W_{k,l,r}$ of the generator $W_{k,l}$.
2. Compute and output the nucleus
\[ U := W_{k,l,r}^+ . \]  

**Theorem 15.** Algorithm 14 involves \( O((k + l)kl) \) flops and \( O(kl) \) memory cells.

**Theorem 16.** Suppose that a canonical CUR LRA is defined by (22) and that (16) holds. Then equation (15) holds.

**Proof.** Combine Corollary 21 for \( \tilde{\sigma}_{r+1} = 0 \) and equations (16) and (22).

**Remark 17.** \( W_{k,l,r} = W_{k,l} \) if \( r = \min\{k,l\} \); (22) turns into (21) if \( k = l = r \).

**Remark 18.** Instead of computing the SVD-truncation we can faster compute another rank-\( r \) approximation of the matrix \( W_{k,l,r} \) solely in terms of a tolerance \( \tilde{\sigma}_{r+1} \) to the error norm. Then we can replace \( \tilde{\sigma}_{r+1} \) by \( \tau \) in the error analysis. Based on computing nuclei by means of SVD-truncation and by applying a rank-revealing LUP factorization of the generators \( W_{k,l} \), which involves just \( O(mn \min\{m,n\}) \) flops (see Section 10.5), we consistently observed about the same accuracy of the resulting CUR LRA.

### 5.2 Output errors of a canonical CUR LRA

Next we extend Theorem 12 by estimating the norms \( |W - CUR| \) and \( |\tilde{U} - U| \) for a canonical CUR LRA. We begin with two lemmas.

**Lemma 19.** Under the assumptions of Theorem 12, define SVD-truncation \( W_{k,l,r} \) of the matrix \( W_{k,l} \) and let \( U := W_{k,l,r}^+ \) and \( \tilde{U} := \tilde{W}_{k,l} \) (cf. (22)). Then
\[ |\tilde{U} - U| \leq \mu|\tilde{U}| \ |U| \ \delta_{k,l,r}. \]

Here \( \delta_{k,l,r} = \|\tilde{W}_{k,l} - W_{k,l,r}\| \leq \eta\tilde{\sigma}_{r+1}, \eta = 1 \) if \( r = \min\{k,l\} \) and \( \eta = 2 \) if \( r < \min\{k,l\} \).

Furthermore \( \mu = 1 \) for \( \| \cdot \| = \| \cdot \|_F \),
\[ \mu = \frac{(1 + \sqrt{5})}{2} \] if \( \| \cdot \| = \| \cdot \| \) and if \( r < \min\{k,l\} \),
\[ \mu = \sqrt{2} \] if \( \| \cdot \| = \| \cdot \|_F \) and if \( r = \min\{k,l\} \).

**Proof.** Notice that \( \text{rank}(\tilde{W}_{k,l}) = \text{rank}(W_{k,l,r}) = r \) and apply [B15, Theorem 2.2.5].

**Lemma 20.** Under the assumptions and definitions of Lemma 19, write \( \theta := \|U\| \ \delta_{k,l,r} \) and let \( \theta < 1 \). Then \( |\tilde{U}| \leq \frac{1}{1 - \theta} \|U\| \).

**Proof.** This is [B15, Theorem 2.2.4].

By combining the two lemmas with Theorem 12 we estimate the output errors of a CUR LRA solely in terms of a tolerance \( \tilde{\delta} \) and the norms of the factors \( C, U \) and \( R \). Next we assume having a closest CUR LRA and then substitute \( \tilde{\delta} = \tilde{\sigma}_{r+1} \).

**Corollary 21.** (i) Under the assumptions and definitions of Lemma 19, it holds that
\[ \frac{1}{\tilde{\sigma}_{r+1}} |W - CUR| \leq (|R| + |C| + \tilde{\sigma}_{r+1} + \mu \eta |C| \ |R| \ |U|)|\tilde{U}| + 1. \]  

(ii) Under the assumptions and definitions of Lemma 20, let \( (1 - \theta)\xi = 1 \) for \( \| \cdot \| = \| \cdot \| \) and \( (1 - \theta)\xi = \sqrt{k}l \) for \( \| \cdot \| = \| \cdot \|_F \). Then
\[ \frac{1}{\tilde{\sigma}_{r+1}} |W - CUR| \leq (|R| + |C| + \tilde{\sigma}_{r+1} + \mu \eta |C| \ |R| \ |U|)\xi|U| + 1, \]  

and so
\[ |W - CUR| = O((\theta + 1)\theta \tilde{\sigma}_{r+1}) \text{ for } \theta = |W| \ |U|. \]
5.3 Superfast a posteriori error estimation for a CUR LRA and the first CUR criterion

The corollary enables superfast a posteriori error estimation for a CUR LRA. Given an \( m \times n \) matrix \( W \), its CUR factors \( U \in \mathbb{C}^{m \times k} \) of (22), \( C \in \mathbb{C}^{n \times l} \) and \( R \in \mathbb{C}^{k \times n} \), and an upper bound on the value \( \tilde{\sigma}_{r+1} \) of equations (9)–(11), we can readily estimate at first the norms of these factors and then the norm of the matrix \( W - \text{CUR} \) of Corollary 21. Even if we are not given the factor \( U \), we can compute it superfast (see Theorem 15).

The error norm bounds of the corollary are proportional to \( \tilde{\sigma}_{r+1} \) and are converging to 0 as \( (\theta + 1)\tilde{\sigma}_{r+1} \to 0 \). By combining these observations criterion with Theorem 11 we deduce that the first CUR criterion is necessary and sufficient for computing accurate CUR LRA.

Corollary 22. A submatrix \( W_{k,l} \) of a matrix \( W \) is a generator of a close CUR LRA of a matrix \( W \) if and only if \( \text{rank}(W_{k,l}) = \text{rank}(W) \).

Remark 23. A submatrix \( W_{k,l} \) satisfies the first CUR criterion unless this submatrix is a small norm perturbation of a matrix of rank less than \( \text{rank}(W) \), that is, of a matrix from an algebraic variety of a smaller dimension in the variety of matrices of rank \( \text{rank}(W) \).

5.4 Superfast transition from top SVD to CUR LRA

Next we complement superfast extension of an LRA to top SVD in Section 3.3 by superfast extension of top SVD of a matrix allowing LRA to its CUR LRA.

Algorithm 24. Superfast transition from top SVD to CUR LRA.

**Input:** Five integers \( k, l, m, n, \) and \( r \) satisfying (13) and matrices \( W \in \mathbb{C}^{m \times n}, \Sigma \in \mathbb{C}^{r \times r} \) (diagonal), \( S \in \mathbb{C}^{m \times r} \) and \( T \in \mathbb{C}^{n \times r} \) (both unitary) such that \( 1 \leq r \ll \min\{m, n\} \), \( W = S\Sigma T^* + E \) and \( ||E|| = O(\tilde{\sigma}_{r+1}) \) for \( \tilde{\sigma}_{r+1} \) of (9).

**Output:** Three matrices \( C \in \mathbb{C}^{m \times l}, U \in \mathbb{C}^{l \times k} \) and \( R \in \mathbb{C}^{k \times n} \) such that \( W = \text{CUR} + E' \) and \( ||E'|| = O(\tilde{\sigma}_{r+1}) \).

**Computations:** 1. By applying the algorithms of [GE96] or [P00] to the matrices \( S \) and \( T \) compute their \( k \times r \) and \( r \times l \) submatrices \( S_{I,:} \) and \( T_{:,J}^* \), respectively. Output the CUR factors \( C = W_{:,J} = S\Sigma T_{:,J}^* \) and \( R = W_{I,:} = S_{I,:} \Sigma T^* \).

2. Compute and output a nucleus \( U = W_{k,k}^+ = W_{k,l}^+ \) for the CUR generator \( W_{k,l} = W_{I,:J} = S_{I,:} \Sigma T_{:,J}^* \).

The algorithm uses \( kn + lm + kl \) memory cells and \( O(ml^2 + nk^2) \) flops.

Clearly \( ||W_{k,l}|| \leq ||W|| \) and deduce from Lemma 27 that

\[
||U|| = ||W_{k,l}|| \leq ||S_{I,:}^+|| ||\Sigma^+|| ||T_{:,J}^+|| \leq t_{m,l,h}t_{n,k,h}/\sigma_r(W) \tag{26}
\]

because \( ||S_{I,:}^+|| \leq t_{m,l,h} \) and \( ||T_{:,J}^+|| \leq t_{n,k,h} \) by virtue of Theorem 119 (we can choose \( h = 1.1 \), say \( h = 1.1 \)), while \( ||\Sigma^+|| = ||W^+|| = 1/\sigma_r(W) \). Now Corollary 22 bounds the output error norm of the algorithm and shows its correctness.

Error bound (26) is minimized for \( k = l = r \), but we can decrease this bound and the computational cost if we choose larger integers \( k \) and \( l \) instead of deterministic.
algorithms of [GE96] or [P00] apply randomized Algorithm 116 with sampling probabilities computed from equations (62).

Hereafter we refer to such a variant of Algorithm 24 as Algorithm 24a. It involves \(mk + nl + kl\) memory cells and \(O(mk + nl + (k + l)kl)\) flops. By virtue of Theorems 117 and 118 combined it implies that for any positive \(\delta \leq 0.4\) with a probability at least \(0.8 - 2\delta\) it holds that

\[
10/||S^+_k||^2 \geq 1 - \sqrt{4r \ln(2r/\delta)/k} \quad \text{and} \quad 10/||T^+_{i,j}||^2 \geq 1 - \sqrt{4r \ln(2r/\delta)/l}
\]

and consequently \(||S^+_k|| \quad ||T^+_{i,j}|| \leq 12.5\) for \(k = l = 100r \ln(2r/\delta)\), say.

6 Accuracy of Superfast CUR LRAs of Random and Average Matrices: Direct Error Estimation

Next we estimate the norms of the matrices \(C, R\) and \(U = W^+_k\) of (22) for a small perturbation of a diagonally scaled factor-Gaussian matrix \(W\) of Definition 4 and then substitute our probabilistic estimates into Corollary 21. Clearly the computation fails at the stage of computing the nucleus \(U = W^+_k\) of (22) unless a submatrix \(W_{k,l}\) has numerical rank \(r\); such a failure, however, occurs with a low probability (see Theorem 104). Moreover we arrive at a reasonably small probabilistic upper bounds on the error norm \(||W - CUR||\) for any choice of a \(k \times l\) CUR generator unless the integer \((k - r)(l - r)\) is small.

We estimate just the spectral norm \(||W - CUR||\); bounds (4) enable us to extend the estimates to bounding the Frobenius norm \(||W - CUR||_F\).

Our results can be readily extended to the average matrices allowing LRA (see Definition 6) and also to the left (resp. right) factor-Gaussian inputs provided that \(l = n\) (resp. \(k = m\)). We need this provision because the upper bounds \(g_+ \geq \sigma_r(G)\) and \(h_+ \geq \sigma_r(H)\) for non-Gaussian factors \(G\) and \(H\) in parts (ii) and (iii) of Definition 4 do not generally hold for the submatrices of these factors. We relax this assumption in Section 7.4.

We deduce our estimates for a rank-\(r\) input by relying on bounding the condition number of a CUR generator. If this number is nicely bounded we can extend the estimates to nearby input matrices, having numerical rank \(r\).

6.1 The norm bounds for CUR factors in a CUR decomposition of a factor-Gaussian matrix

Consider a CUR decomposition of a diagonally scaled factor-Gaussian matrix \(W = CUR\) having an expected rank \(r\), a generator \(W_{k,l}\) and nucleus \(U = W^+_k\) of (22). By virtue of Theorem 5 the submatrices \(C, R\), and \(W_{k,l}\) are also diagonally scaled factor-Gaussian matrices having an expected rank \(r\).

By virtue of Theorem 100 they have full rank \(r\) with probability 1, and we deduce our probabilistic estimates assuming that these matrices do have full rank \(r\).

Next observe that

\[
||C|| \leq \nu_{m,r} \nu_{r,r} \sigma \quad \text{and} \quad ||R|| \leq \nu_{r,r} \nu_{r,n} \sigma, \quad (27)
\]

for \(\sigma =: \sigma_1\) of Definition 4 and \(\nu_{p,q}\) denoting the random variables of Definition 7, whose probability distributions and expected values have been estimated in Theorem 102. In particular obtain the following upper bounds on the expected values,

\[
\frac{\mathbb{E}(||C||)}{\sigma} \leq \mathbb{E}(\nu_{m,r}) \mathbb{E}(\nu_{r,r}) \leq 2(\sqrt{mr} + r), \quad \frac{\mathbb{E}(||R||)}{\sigma} \leq \mathbb{E}(\nu_{r,r}) \mathbb{E}(\nu_{r,n}) \leq 2(\sqrt{nr} + r). \quad (28)
\]

18
Recall that $W_{k,l,r} = W_{k,l}$ for $W_{k,l} \in \mathbb{C}^{k \times l}$, and so $U = W_{k,l}^+$ (cf. Remark 18).

We complete our estimates with the following theorem and corollary, which we prove in Section 6.3. We write $e := 2.7182828\ldots$, and so $e^2 < 7.4$.

**Theorem 25.** (i) Let $W'$ be an $m \times n$ diagonally scaled factor-Gaussian matrix with an expected rank $r$ (a scaled factor-Gaussian matrix is a special case where $\sigma_1 = \sigma_r$). Let $W_{k,l}$ be its $k \times l$ submatrix for $\min\{k, l\} > r$ and define $U = W_{k,l,r}^+$ (cf. (22)). Then

$$
\mathbb{E}(\|U\|) \leq \frac{e^2 \sqrt{kl}}{(k-r)(l-r)\sigma_r} < \frac{7.4 \sqrt{kl}}{(k-r)(l-r)\sigma_r},
$$

(29)

for $\sigma_r$ of Definition 4. Furthermore, for any fixed $\zeta > 1$,

$$
\text{Probability}\left\{ \|U\| > \frac{e^2 \sqrt{kl}}{(k-r)(l-r)\sigma_r} \right\}
$$

converges to 0 exponentially fast as $\min\{k-r, l-r\}$ grows to the infinity.

(ii) Consider an $m \times n$ left factor-Gaussian matrix $W$ with an expected rank $r$ (see Definition 4). Let $W_{k} := W_{k,n}$ denote its $k \times n$ submatrix. Then

$$
\mathbb{E}(\|U\|) = \mathbb{E}(\|G_{k,r}^+ ||H^+\|) = \mathbb{E}(\|G_{k,r}^+ \|) \quad h_+ = \mathbb{E}(v_{k,r}^+)h_+ \leq \frac{e \sqrt{k} h_+}{k-r} < \frac{7.2 \sqrt{k} h_+}{k-r}.
$$

(31)

(iii) Consider an $m \times n$ right factor-Gaussian matrix $W'$ with an expected rank $r$ (see Definition 4). Let $W_{m,l} := W'_{m,l}$ denote its $m \times l$ submatrix. Then

$$
\mathbb{E}(\|U\|) = \mathbb{E}(\|G^+ ||H_{r}^+\|) = g_+ \mathbb{E}(\|H_{r}^+\|) = g_+ \mathbb{E}(v_{l,r}^+) \leq \frac{e \sqrt{l} g_+}{l-r} < \frac{7.2 \sqrt{l} g_+}{l-r}.
$$

(32)

**Corollary 26.** For the average matrices $W$ of Definition 6 upper bound (24) on the ratio $||W - CUR||/\tilde{\sigma}_{r+1}$ turns into

$$
((4\sqrt{mr} + r)(\sqrt{nr} + r)\sigma \mu \eta \mathbb{E}(\|U\|) + 4r + 2\sqrt{mr} + 2\sqrt{nr})\sigma + \tilde{\sigma}_{r+1})\xi \mathbb{E}(\|U\|) + 1
$$

for $\mu$, $\eta$, $\xi$ and $\sigma$ of Theorem 25 and for $\mathbb{E}(\|U\|)$ bounded according to (29), (31) and (32).

### 6.2 A bound on the norm of pseudo inverse of a matrix product

Recall that

$$
\|M^+\|\sigma_r(M) = 1
$$

(33)

for a matrix $M$ of full rank $r$ and then estimate the norm of a pseudo inverse of a product.

**Lemma 27.** Let $G \in \mathbb{C}^{k \times r}$, $\Sigma \in \mathbb{C}^{r \times r}$ and $H \in \mathbb{C}^{r \times l}$ and let the matrices $G$, $H$ and $\Sigma$ have full rank $r \leq \min\{k, l\}$. Then $\|G\Sigma H^+\| \leq ||G^+|| \|\Sigma^+\| \|H^+\|$.\n
**Proof.** Let $G = S_G\Sigma_G T_G$ and $H = S_H\Sigma_H T_H$ be SVDs where $S_G$, $T_G$, $D_H$, and $T_H$ are unitary matrices, $\Sigma_G$ and $\Sigma_H$ are the $r \times r$ nonsingular diagonal matrices of the singular values, and $T_G$ and $S_H$ are $r \times r$ matrices. Write

$$
M := \Sigma_G T_G \Sigma_S T_H.
$$
Then
\[ M^{-1} = \Sigma_H^{-1} S_H^* \Sigma^{-1} T_G^* \Sigma_G^{-1}, \]
and consequently
\[ ||M^{-1}|| \leq ||\Sigma_H^{-1}|| \ ||S_H^*|| \ ||\Sigma^{-1}|| \ ||T_G^*|| \ ||\Sigma_G^{-1}||. \]
Hence
\[ ||M^{-1}|| \leq ||\Sigma_H^{-1}|| \ ||\Sigma^{-1}|| \ ||\Sigma_G^{-1}|| \]
because \( S_H \) and \( T_G \) are unitary matrices. It follows from (33) that
\[ \sigma_r(M) \geq \sigma_r(G)\sigma_r(\Sigma)\sigma_r(H). \]

Now let \( M = S_M \Sigma_M T_M \) be SVD where \( S_M \) and \( T_M \) are \( r \times r \) unitary matrices.
Then \( S := S_G S_M \) and \( T := T_M T_H \) are unitary matrices, and so \( G \Sigma H = S \Sigma_M T \) is SVD.
Therefore \( \sigma_r(G \Sigma H) = \sigma_r(M) \geq \sigma_r(G)\sigma_r(\Sigma)\sigma_r(H) \).
Combine this bound with (33) for \( M \) standing for \( G, \Sigma, H, \) and \( G \Sigma H \).

\[ \square \]

6.3 The proof of the norm bounds for CUR factors

Since \( W' \) is a diagonally scaled factor-Gaussian matrix with an expected rank \( r \), write
\[ W' = G \Sigma H \]
where \( G \in \mathcal{G}^{m \times r}, \Sigma \in \mathbb{C}^{r \times r}, H \in \mathcal{G}^{r \times n} \), and \( ||\Sigma^+|| = 1/\sigma_r \) for \( \sigma_r \) of Definition 4.
It follows that \( W'_{L,J} = G_{L,:}\Sigma H_{:,J}^* \) for \( G_{L,:} \in \mathcal{G}^{k \times r} \) and \( H_{:,J} \in \mathcal{G}^{r \times l} \).
Hereafter
\[ \Gamma(x) := \int_0^\infty \exp(-t)t^{x-1}dt \]
denotes the Gamma function. Apply Theorem 104 and obtain
\[ \text{Probability}\{||G_{L,:}^+|| \geq x\} \leq \frac{(x/k)^{-0.5(k-r+1)}}{\Gamma(k-r+2)}, \]
\[ \text{Probability}\{||H_{:,J}^+|| \geq x\} \leq \frac{(x/k)^{-0.5(l-r+1)}}{\Gamma(l-r+2)}, \]
\[ \mathbb{E}(||G_{L,:}^+||) = \mathbb{E}(\nu_k^+) \leq \frac{e\sqrt{k}}{k-r}, \text{ and } \mathbb{E}(||H_{:,J}^+||) = \mathbb{E}(\nu_l^+) \leq \frac{e\sqrt{l}}{l-r}. \]
Recall that the matrix \( \Sigma \) is nonsingular and the matrices \( G \) and \( H \) have full rank with probability 1 (see Theorem 100), apply Lemma 27 to the matrix \( W = G \Sigma H \) and obtain that
\[ ||W_{L,J}^+|| = ||(G_{L,:}\Sigma H_{:,J})^+|| \leq ||G_{L,:}^+|| ||H_{:,J}^+|| ||\Sigma^+||. \]
Substitute equation \( ||\Sigma^+|| = 1/\sigma_r \) and bounds (35) and (36) and obtain bound (29).
Similarly complete the derivation of bounds (30)–(32).

7 Primitive, Cynical and Cross-approximation CUR Algorithms

In view of Corollary 26 we expect to compute a close LRA (unless the integer \((k-r)(l-r)\) is small) even when we apply the following simple algorithm to the average input for any choice of a \( k \times l \) CUR generator, and in particular for its random choice.
7.1 Randomized primitive CUR algorithm and the probability of its failure on $\pm\delta$-matrices

**Primitive CUR algorithm** randomly chooses a $k \times l$ CUR generator $W_{k,l}$ of an $m \times n$ input matrix $W$ having numerical rank $r$ and then computes a nucleus $U$ of (22) and defines a CUR LRA of the matrix $W$.

The algorithm involves $O((k+l)kl)$ flops (or heuristically $O(kl \min\{k,l\})$ flops, see Remark 18) and $O(kl)$ memory cells, that is, $O(r^3)$ flops and $O(r^2)$ cells if $k + l = O(r)$. The algorithm is superfast if $(k + l)kl \ll mn$.

Like any LRA algorithm it either fails on the family of $\pm\delta$-matrices of Example 9 or involves all $mn$ entries of the matrix $W$ and therefore uses at least $0.5mn$ flops.

Let $p_{q,s,m,n}$ denote the probability that a random $k \times l$ CUR generator contains a single fixed entry of $W$ encountered with the smallest probability. Clearly the value $p_{q,s,m,n}$ is maximized where the column set and row set defining a CUR generator are chosen under the random uniform probability distribution, and then we estimate that

$$p_{k,l,m,n} := \frac{k}{m} + \frac{l}{n} - \frac{kl}{mn}, \quad p_{k,l,m,n} \to 0 \quad \text{as} \quad \frac{k}{m} + \frac{l}{n} \to 0. \quad (37)$$

The $\pm\delta$-matrices, however, is a specially concocted family, and in our tests primitive algorithm has output quite accurate CUR LRAs to a large class of real world matrices (see Tables 1, 3 and 4). We succeeded even more consistently, however, when we applied a little more advanced algorithms that we describe and analyze next.

### 7.2 Sketching techniques and cynical CUR algorithms

The upper bound of Theorem 25 on the expected value of the norm $||U||$ of the nucleus decreases proportionally to $\sqrt{kl}$ as $k$ and $l$ grow large. Computation of canonical CUR LRA involves order of $(k+l)kl$ flops and remains superfast as long as $(k+l)kl \ll mn$. Trying to keep the benefit of decreasing the error bound by involving more rows and columns but to bound the cost of computing a nucleus we arrive at the following recipes.

First compute a $k \times l$ CUR generator by applying a CUR subalgorithm to a sketch of a larger size $q \times s$. This defines a $k \times l$ CUR generator for the sketch, but use it for the input matrix in order to produce its CUR LRA. Below we specify the class of the resulting algorithms. They are still very simple, and so we call them **cynical**.

---

**Figure 4**: A cynical CUR algorithm (the strips mark a sketch; the CUR generator is black)

**A cynical CUR Algorithm** (see Figure 4):
INPUT: Five integers $k, l, m, n$ and $r$ satisfying (13) and such that $kl \ll mn$; an $m \times n$ matrix $W$.

INITIALIZATION: (i) Fix two integers $q$ and $s$ satisfying bounds (1) and $qs \ll mn$.
(ii) Randomly choose a $q \times s$ sketch of the matrix $W$.
(iii) Select a CUR subalgorithm.

COMPUTATIONS: Apply the selected subalgorithm to compute a CUR approximation of the sketch and extend it to a CUR LRA of the matrix $W$.

For $q \times s$ sketch both primitive and cynical algorithms involve $qs$ memory cells but cynical algorithms use fewer flops: $O(qs \min\{q, s\})$ versus $O(qs \max\{q, s\})$, in both cases provided that we use the subalgorithms from [GE96] and [P00].

Moreover in our tests cynical algorithms with a subalgorithm from [P00] succeeded more consistently than the primitive algorithm (see Tables 1 and 4).

7.3 Cross-approximation (C-A) algorithms

If a primitive or cynical algorithm fails we can reapply it recursively with new random CUR generators. With every new application we have new very good chances for success. Empirically the following policy is particularly efficient; later we prove its various benefits.

Suppose that we have applied a cynical algorithm to a $q \times n$ sketch of an $m \times n$ input matrix $W$ but are not satisfied with the accuracy of the CUR LRA defined by the $k \times l$ output submatrix $W_{k,l}$. Then we can apply a cynical algorithm to a new $m \times s$ sketch containing the submatrix $W_{k,l}$ and can recursively repeat this process.

Next, for the sake of simplicity, we only study the case where $q = k$ and $s = l$, although allowing any pair of $q$ and $s$ satisfying (1) would give us more power.

Cross-approximation (C-A) algorithm for $q = k$ and $s = l$ (see Figure 5).

INPUT: five positive integers $k, l, m, n$ and $r$ satisfying (13) and an $m \times n$ matrix $M$ allowing close rank-$r$ approximation.

INITIALIZATION: Fix a positive integer $u$. Choose a $k \times n$ initial sketch $W_0$ of $W$. Fix a stopping criterion for CUR approximation.

COMPUTATIONS: By recursively applying cynical algorithms for $i = 0, 1, \ldots, u - 1$, compute $k \times l$ CUR generators $W'_{2i}$ and $W'_{2i+1}$ of $k \times n$ horizontal sketches $W_{2i}$ and $m \times l$ vertical sketches $W_{2i+1}$, made up of $k$ rows and $l$ columns of the matrix $W$, respectively. (Call the even and odd numbered C-A steps horizontal and vertical, respectively.)
For every $j$, $j = 0, 1, \ldots, 2u - 1$, define the sketch $W_{j+1}$ by its $k \times l$ submatrix $W_j^t$, computed at the previous recursive step as a CUR generator of the previous sketch $W_{j-1}$.

If at the $j$th C-A iteration for $j < 2u$ the current $k \times l$ matrix $W_j^t$ satisfies the fixed stopping criterion, then use this matrix as a CUR generator, compute a nucleus and output a rank-$r$ CUR approximation $W'$ for the matrix $W$.

Unless this occurs for some $j < 2u$, output FAILURE.

The algorithm performs at the cost of at most $2u$ applications of cynical algorithms.

We can apply cynical algorithms based on using the algorithms of [GE96] or [P00] but in this case we deal with tall-skinny input matrices or their transposes, so we are motivated to apply equation (62) and Algorithm 115 instead.

**Remark 28.** The study of horizontal and vertical C-A steps is symmetric and can be reduced to one another by means of transposition of the matrix $W$; in the Dual Version of C-A algorithms we begin iterations with a vertical initial sketch $W_1$ made up of $l$ columns of matrix $W$ rather than with horizontal sketch $W_0$, made up of its $k$ rows.

In both primal and dual versions practical convergence of C-A iterations substantially depends on the initial choice of the sketch. We can choose such a sketch at random, but here are two better options.

(a) Compute a $k \times l$ initial sketch by applying a cynical algorithm to a random $q \times s$ sketch for a fixed pair of integers $q$ and $s$ satisfying (1).

(b) Choose a random $k \times l$ matrix, then apply to it both C-A primal step and C-A dual step and let the initial sketch be one of the two outputs of these two steps, namely the one that is superior according to a fixed stopping criterion.

**Remark 29.** Extending C-A iterations to $d$-dimensional tensor for $d \geq 3$ we would search in all $d$ directions initially and in $d - 1$ directions at all subsequent iterations; among these options we can choose one supporting closer LRA. For a sample variation we can search among about $0.5d^2$ slices instead or in addition to the search among all $d$ fibers.

### 7.4 CUR LRA of left, right and sparse factor-Gaussian and average matrices

Let us revisit the estimation of the norm $\|W_{I,J}^+\|$ for some submatrices $W_{I,J}$ of left and right factor-Gaussian and $NZ$-factor-Gaussian matrices $W = GH$. In Section 6.2 we reduced this task to the estimation of the norms $\|G_{I,J}^+\|$ and $\|H_{I,J}^+\|$ of some submatrices $G_{I,J}$ and $H_{I,J}$ of the factors $G$ and $H$, respectively.

So far we have no good bounds on these norms, but next we deduce such bounds by extending the reasonable upper estimates for the norms $\|G^+\|$ and $\|H^+\|$, available in the case of these matrix classes due to Theorems 104 and 110.

Namely, we assume that the submatrices $G_{I,J}$ and $H_{I,J}$ have been output in two successive C-A steps that use the algorithms of [GE96] as the subalgorithms. In this case by virtue of Corollary 120 it holds that $\|G_{I,J}^+\| \leq t_{n,r,h} \|G^+\|$ and $\|H_{I,J}^+\| \leq t_{m,r,h} \|H^+\|$ for $t_{p,r,h}^2 = (p - r)rh^2 + 1$ of (57) and any fixed $h > 1$.

The bounds on the norms $\|G^+\|$ and $\|H^+\|$ mildly increase by the factors $t_{n,r,h}$ and $t_{m,r,h}$ in the transition to the bounds on the norms $\|G_{I,J}^+\|$ and $\|H_{I,J}^+\|$, but remain reasonable. (Recall that we can decrease the factors $t_{n,r,h}$ and $t_{m,r,h}$ to constants by applying the recipe of Remark 121.) Combine these bounds with those of (27) and (28) and with Corollary 21 and arrive at the following result.
Theorem 30. Perform two successive steps of a C-A algorithm with subalgorithms from the paper [GE96] and build a CUR LRA of an input matrix $W$ on the CUR generator output by the second step. Then this approximation is accurate with a high probability as long as the input matrix $W$ is a small norm perturbation of a left, right or $NZ$-factor-Gaussian matrix with a small expected rank $r$ and therefore also in the case where $W$ is the left, right or $NZ$-average matrix allowing close approximation by rank-$r$ matrices.

Large PART II. CUR LRA BASED ON VOLUME MAXIMIZATION

8 Volumes of Matrices and Matrix Products

In the next section we study CUR LRA based on the second CUR criterion, that is, on the maximization of the volume of a CUR generator. (For comparison the first CUR criterion implies that the condition number of a CUR generator should not be large.)

In the next subsection we define and estimate the basic concept of matrix volume.

In Section 8.2 we estimate the impact of a perturbation of a matrix on its volume.

In Section 8.3 we bound the volume of a matrix product via the volumes of the factors.

8.1 Volume of a matrix: definitions and the known upper estimates

For a triple of integers $k$, $l$, and $r$ such that $1 \leq r \leq \min\{k,l\}$, the volume $v_2(M)$ and the $r$-projective volume $v_{2,r}(M)$ of a $k \times l$ matrix $M$ are defined as follows:

$$v_2(M) := \prod_{j=1}^{\min\{k,l\}} \sigma_j(M), \quad v_{2,r}(M) := \prod_{j=1}^{r} \sigma_j(M),$$

(38)

$$v_{2,r}(M) = v_2(M) \text{ if } r = \min\{k,l\},$$

(39)

$$v_2^2(M) = \det(MM^*) \text{ if } k \geq l; \quad v_2^2(M) = \det(M^*M) \text{ if } k \leq l; \quad v_2^2(M) = |\det(M)| \text{ if } k = l.$$

Here are some upper bounds on the matrix volume.

For a $k \times l$ matrix $M = (m_{ij})_{i,j=k,l}$ write $m_j := (m_{ij})_{i=1}^k$ and $\bar{m}_i := ((m_{ij})_{j=1}^l)^*$ for all $i$ and $j$. For $k = l = r$ recall Hadamard’s bounds

$$v_2(M) = |\det(M)| \leq \min \left\{ \prod_{j=1}^{r} ||m_j||, \prod_{i=1}^{r} ||\bar{m}_i||, \ r^{r/2} \max_{i,j=1}^r |m_{ij}|^r \right\}.$$

(40)

For $k \leq l \leq q$, a $k \times q$ matrix $M$ and its any $k \times l$ submatrix $M_{k,l}$ observe that

$$v_2(M) = v_{2,k}(M) \geq v_2(M_{k,l}) = v_{2,k}(M_{k,l}).$$

(41)

8.2 The volume and $r$-projective volume of a perturbed matrix

Theorem 31. Suppose that $W'$ and $E$ are $k \times l$ matrices, $\text{rank}(W') = r \leq \min\{k,l\}$, $W = W' + E$, and $||E|| \leq \epsilon$. Then

$$\left(1 - \frac{\epsilon}{\sigma_r(W)}\right)^r \leq \prod_{j=1}^{r} \left(1 - \frac{\epsilon}{\sigma_j(W)}\right) \leq \frac{v_{2,r}(W)}{v_{2,r}(W')} \leq \prod_{j=1}^{r} \left(1 + \frac{\epsilon}{\sigma_j(W)}\right) \leq \left(1 + \frac{\epsilon}{\sigma_r(W)}\right)^r.$$

(42)

If $\min\{k,l\} = r$, then $v_2(W) = v_{2,r}(W)$, $v_2(W') = v_{2,r}(W')$, and
\[
\left(1 - \frac{\epsilon}{\sigma_r(W)}\right)^r \leq \frac{v_2(W)}{v_2(W')} = \frac{v_{2,r}(W)}{v_{2,r}(W')} \leq \left(1 + \frac{\epsilon}{\sigma_r(W)}\right)^r. \tag{43}
\]

**Proof.** Bounds (42) follow because a perturbation of a matrix within a norm bound \(\epsilon\) changes its singular values by at most \(\epsilon\) (see [GL13, Corollary 8.6.2]). Bounds (43) follow because \(v_2(N) = \prod_{j=1}^k \sigma_j(N)\) for any \(k \times l\) matrix \(N\) with \(\min\{k, l\} = r\), in particular for \(N = W\) and \(N = W + E\).

If the ratio \(\frac{\epsilon}{\sigma_r(W)}\) is small, then \(\left(1 - \frac{\epsilon}{\sigma_r(W)}\right)^r = 1 - O\left(\frac{\epsilon}{\sigma_r(W)}\right)\) and \(\left(1 + \frac{\epsilon}{\sigma_r(W)}\right)^r = 1 + O\left(\frac{\epsilon}{\sigma_r(W)}\right)\), which shows that the volume perturbation is amplified by at most a factor of \(r\) versus the perturbation of singular values.

**Remark 32.** Theorem 31 implies that a small norm perturbation of a matrix \(W\) of rank \(r\) little changes its volume unless the matrix is ill-conditioned.

### 8.3 The volume and \(r\)-projective volume of a matrix product

**Theorem 33.** (See Examples 34 and 35 and Remark 36 below.)

Suppose that \(W = GH\) for an \(m \times q\) matrix \(G\) and a \(q \times n\) matrix \(H\). Then

(i) \(v_2(W) = v_2(G)v_2(H)\) if \(q = \min\{m, n\}\); \(v_2(W) = 0 \leq v_2(G)v_2(H)\) if \(q < \min\{m, n\}\).

(ii) \(v_{2,r}(W) \leq v_{2,r}(G)v_{2,r}(H)\) for \(1 \leq r \leq q\),

(iii) \(v_2(W) \leq v_2(G)v_2(H)\) if \(m = n = q\).

The following examples show some limitations on the extension of the theorem.

**Example 34.** If \(G\) and \(H\) are unitary matrices and if \(GH = O\), then \(v_2(G) = v_2(H) = v_{2,r}(G) = v_{2,r}(H) = 1\) and \(v_2(GH) = v_{2,r}(GH) = 0\) for all \(r \leq q\).

**Example 35.** If \(G = (1 \mid 0)\) and \(H = \text{diag}(1, 0)\), then \(v_2(G) = v_2(GH) = 1\) and \(v_2(H) = 0\).

**Remark 36.** See distinct proofs of claims (i) and (iii) in [O16].

**Proof.** We first prove claim (i).

Let \(G = S_G\Sigma_G T_G^*\) and \(H = S_H\Sigma_H T_H^*\) be SVDs such that \(\Sigma_G, \ T_G, \ S_H, \ \Sigma_H, \) and \(U = T_G^*S_H\) are \(q \times q\) matrices and \(S_G, \ T_G, \ S_H, \ T_H, \) and \(U\) are unitary matrices.

Write \(V := \Sigma_G U\Sigma_H\) and notice that \(\det(V) = \det(\Sigma_G)\det(U)\det(\Sigma_H)\). Substitute \(1 = \det(U)\) (which holds because \(U\) is a square unitary matrix) and obtain \(v_2(V) = \det(\Sigma_G)\det(\Sigma_H) = v_2(G)v_2(H)\).

Now let \(V = S_V\Sigma_V T_V^*\) be SVD where \(S_V, \Sigma_V\) and \(T_V\) are \(q \times q\) matrices and \(S_V\) and \(T_V\) are unitary matrices.

Observe that \(W = S_GVT_H^* = S_G S_V \Sigma_V T_V^* T_H^* = S_W \Sigma_W T_W^*\) where \(S_W = S_G S_V\) and \(T_W^* = T_V^* T_H^*\) are unitary matrices. Consequently \(W = S_W \Sigma_W T_W^*\) is SVD, and so \(\Sigma_W = \Sigma_V\).

Therefore \(v_2(W) = v_2(V) = v_2(G)v_2(H)\) unless \(q < \min\{m, n\}\). This proves claim (i) because clearly \(v_2(W) = 0\) if \(q < \min\{m, n\}\).

Next we prove claim (ii).

First assume that \(q \leq \min\{m, n\}\) as in claim (i) and let \(W = S_W \Sigma_W T_W^*\) be SVD.

In this case we have proven that \(\Sigma_W = \Sigma_V\) for \(V = \Sigma_G U\Sigma_H\), \(q \times q\) diagonal matrices \(\Sigma_G\) and \(\Sigma_H\), and a \(q \times q\) unitary matrix \(U\). Consequently \(v_{2,r}(W) = v_{2,r}(\Sigma_V)\).
In order to prove claim (ii) in the case where \( q \leq \min\{m, n\} \), it remains to deduce that

\[
v_{2,r}(\Sigma_V) \leq v_{2,r}(G)v_{2,r}(H). \tag{44}
\]

Notice that \( \Sigma_V = S^*_V V T_V = S^*_V \Sigma_G U \Sigma_H T_V \) for \( q \times q \) unitary matrices \( S^*_V \) and \( H_V \).

Let \( \Sigma_{r,V} \) denote the \( r \times r \) leading submatrix of \( \Sigma_V \), and so \( \Sigma_{r,V} = G\hat{H} \) where \( \hat{G} := S^*_r V \Sigma_G U \) and \( \hat{H} := \Sigma_H T_{r,V} \) and where \( S_{r,V} \) and \( T_{r,V} \) denote the \( r \times q \) leftmost unitary submatrices of the matrices \( S_V \) and \( T_V \), respectively.

Observe that \( \sigma_j(\hat{G}) \leq \sigma_j(G) \) for all \( j \) because \( \hat{G} \) is a submatrix of the \( q \times q \) matrix \( S^*_V \Sigma_G U \), and similarly \( \sigma_j(\hat{H}) \leq \sigma_j(H) \) for all \( j \). Therefore \( v_{2,r}(\hat{G}) = v_2(\hat{G}) \leq v_{2,r}(G) \) and \( v_{2,r}(\hat{H}) = v_2(\hat{H}) \leq v_{2,r}(H) \). Also notice that \( v_{2,r}(\Sigma_{r,V}) = v_2(\Sigma_{r,V}) \).

Furthermore \( v_2(\Sigma_{r,V}) \leq v_2(\hat{G})v_2(\hat{H}) \) by virtue of claim (i) because \( \Sigma_{r,V} = \hat{G}\hat{H} \).

Combine the latter relationships and obtain (44), which implies claim (ii) in the case where \( q \leq \min\{m, n\} \).

Next we extend claim (ii) to the general case of any positive integer \( q \).

Embed a matrix \( H \) into a \( q \times q \) matrix \( H' := (H \mid O) \) banded by zeros if \( q > n \). Otherwise write \( H' := H \). Likewise embed a matrix \( G \) into a \( q \times q \) matrix \( G' := (G^T \mid O)^T \) banded by zeros if \( q > m \). Otherwise write \( G' := G \).

Apply claim (ii) to the \( m' \times q \) matrix \( G' \) and \( q \times n' \) matrix \( H' \) where \( q \leq \min\{m', n'\} \).

Obtain that \( v_{2,r}(G'H') \leq v_{2,r}(G'H) \).

Substitute equations \( v_{2,r}(G') = v_{2,r}(G) \), \( v_{2,r}(H') = v_{2,r}(H) \), and \( v_{2,r}(G'H') = v_{2,r}(GH) \), which hold because the embedding keeps invariant the singular values and therefore keeps invariant the volumes of the matrices \( G \), \( H \), and \( GH \). This completes the proof of claim (ii), which implies claim (iii) because \( v_2(V) = v_{2,n}(V) \) if \( V \) stands for \( G \), \( H \) or \( GH \) and if \( m = n \leq q \).

**Corollary 37.** Suppose that \( BW = (BU|BV) \) for a nonsingular matrix \( B \) and that the submatrix \( U \) is \( h \)-maximal in the matrix \( W = (U|V) \). Then the submatrix \( BU \) is \( h \)-maximal in the matrix \( BW \).

## 9 Criterion of Volume Maximization (the Second CUR Criterion) and the Second Proof of the Accuracy of Superfast CUR LRA of Random and Average Matrices

In this section we estimate the errors of CUR LRAs in terms of the maximization of the volume of a CUR generator, which leads to our second CUR criterion, and yield an alternative proof that primitive, cynical and C-A CUR superfast algorithms are accurate on almost any input allowing LRA and hence on the average input allowing LRA.

As in Section 6 we estimate the errors of CUR LRA for random and average inputs of rank \( r \); by virtue of Theorem 31 the estimates can be extended to a small neighborhood of such inputs. The estimates are strong for inputs lying near well-conditioned matrices of rank \( r \) (see Remark 32) provided that a CUR generator has maximal volume.

In the next subsection we recall the error bounds of [O16] for a CUR LRA in terms of the volume of CUR generator and conclude that the errors are small if the volume is close to maximal.

In Section 9.2 we observe that with a high probability the volume is nearly maximal for any \( k \times l \) submatrix of an \( m \times n \) diagonally scaled, left or right factor-Gaussian matrix with expected rank \( r \) for any \( k, l, m, n \) and \( r \) satisfying (13).
Together the two subsections imply that for any choice of a $k \times l$ submatrix for a CUR generator we obtain with a high probability an accurate CUR LRA of a matrix of the above classes as well as of its small norm perturbation and hence also of the average matrix allowing LRA. This yields an alternative proof to that of Section 6, and this time we allow any choice of integers $k$ and $l$ satisfying (13).

We estimate the errors in the Chebyshev norm. Transition to the Frobenius norm may increase the estimates by at most a factor of $\sqrt{mn}$ for an $m \times n$ input (see (4)).

9.1 Volume maximization as the second CUR criterion

The following result is [O16, Theorem 6]; it is also [GT01, Corollary 2.3] in the special case where $k = l = r$ and $m = n$.

**Theorem 38.** Suppose that $r := \min\{k, l\}$, $W_{\mathcal{I}, \mathcal{J}}$ is the $k \times l$ CUR generator, $U = W_{\mathcal{I}, \mathcal{J}}^+$ is the nucleus defining a canonical CUR LRA $W' = \text{CUR}$ of an $m \times n$ matrix $W$, $E = W - W'$ (see (12), (14) and (22)), $h \geq 1$, and

$$h \ v_2(W_{\mathcal{I}, \mathcal{J}}) = \max_B v_2(B)$$

where the maximum is over all $k \times l$ submatrices $B$ of the matrix $W$. Then

$$||E||_C \leq h \ f(k, l) \ \sigma_{r+1}(W) \ \text{for} \ f(k, l) := \sqrt{(k + 1)(l + 1)}/|l - k|.$$ 

**Theorem 39.** [O16, Theorem 7]. Suppose that $W_{k,l} = W_{\mathcal{I}, \mathcal{J}}$ is a $k \times l$ submatrix of an $m \times n$ matrix $W$, $U = W_{k,i}^+$ (cf. (22)) is the nucleus of a canonical CUR LRA of $W$, $E = W - W'$, $h \geq 1$, and

$$h \ v_{2, r}(W_{\mathcal{I}, \mathcal{J}}) = \max_B v_{2, r}(B)$$

where the maximum is over all $k \times l$ submatrices $B$ of the matrix $W$. Then

$$||E||_C \leq h \ f(k, l, r) \ \sigma_{r+1}(W) \ \text{for} \ f(k, l, r) := \sqrt{(k + 1)(l + 1)/(k - r + 1)(l - r + 1)}.$$ 

9.2 Any submatrix of the average matrix allowing LRA has a nearly maximal volume

Claim (i) of Theorem 33 implies the following result.

**Theorem 40.** Represent an $m \times n$ diagonally scaled factor-Gaussian matrix $W$ having an expected rank $r$ as the product $G \Sigma H$ where $G \in \mathcal{G}^{m \times r}$, $\Sigma \in \mathbb{C}^{r \times r}$, $H \in \mathcal{G}^{r \times n}$, and the matrix $\Sigma$ is well-conditioned, so that a $k \times l$ submatrix $W_{k,l}$ of $W$ is the product $G_k \Sigma H_l$ where the well-conditioned matrix $\Sigma$ is independent of the choice of a $k \times l$ submatrix. Then

$$v_2(W_{k,l}) = v_2(G_k \Sigma H_l) = v_2(G_k) v_2(\Sigma) v_2(H_l).$$

Now recall the following theorem.
Theorem 41. (See [MZ08].) Let $G \in \mathcal{G}^{r \times l}$ for $1 \leq r \leq l$ and let $\epsilon$ be a small positive number. Then

$$\text{Probability}\left\{ v_2(G)^{1/r} \geq (1 + 2\epsilon)\mathbb{E}(v_2(G)^{1/r}) \right\} = O(\exp(-l\epsilon^2))$$

and

$$\text{Probability}\left\{ v_2(G)^{1/r} \leq (1 - \epsilon)\mathbb{E}(v_2(G)^{1/r}) \right\} = O(\exp(-l\epsilon^2)).$$

The theorem implies that the volume of a Gaussian matrix is very strongly concentrated about its expected value. Therefore the volume of a $k \times l$ submatrix $W_{k,l}$ of $W$ is also very strongly concentrated about the expected value $\nu_{k,l,r,\Sigma} = \mathbb{E}(v_2(G_k \Sigma H_l))$ of the volume $v_2(G_k \Sigma H_l) = v_2(G_k)v_2(\Sigma)v_2(H_l)$ of the matrix $G_k \Sigma H_l$. This value only depends on the integers $k$, $l$, and $r$ and the matrix $\Sigma$, all of which are invariant in the choice of a specific $k \times l$ submatrix of $W = G \Sigma H$.

It follows that with a high probability the volume of any such a submatrix is close to the value $\nu_{k,l,r,\Sigma}$ and thus is within a factor close to 1 from the maximal volume; moreover this factor little depends on the choice of a triple of integers $k$, $l$ and $r$.

The scaled factor-Gaussian inputs with expected rank $r$ are a special case, and the argument and the results of our study are readily extended to the case of left or right factor-Gaussian matrix $W$ with expected rank $r$ and then to the average inputs $W$ obtained over the scaled, diagonally scaled, left and right factor-Gaussian inputs as well as small norm perturbations of the matrices of all these classes.

Theorem 31 implies extension of our results to inputs lying near factor-Gaussian and average matrices.

10 C-A Towards Volume Maximization

10.1 Section's outline

Corollary 22 and Theorem 38 combined imply that a submatrix $W_{I,J}$ can have maximum volume in a matrix $W$ only if $\text{rank}(W_{I,J}) = \text{rank}(W)$. Suppose that they do coincide for a vertical or horizontal C-A step, and apply two successive C-A steps initiated at this sketch of a matrix $W$.

In the next two subsections we prove that even for the worst case input matrix $W$ closely approximated by a matrix of a low rank $r$, two successive C-A steps output a matrix having $\bar{h}$-maximal volume in the input matrix $W$ for a certain bound $\bar{h}$, which is rather reasonable in case of small ranks $r$. Hence by virtue of Theorem 38 already the two C-A steps generate a CUR LRA of the matrix $W$ within a bounded error.

In Sections 10.4 – 10.6 we extend this result to the maximization of $r$-projective volume rather than just volume of a CUR generator. (Theorem 39 shows benefits of such maximization.)

In Section 10.7 we summarize our study in this section and comment on the estimated and empirical performance of C-A algorithms.

10.2 We only need to maximize $k \times l$ volume in the input sketches of a two-step C-A loop

We begin with some definitions and simple auxiliary results.
**Definition 42.** The volume of a $k \times l$ submatrix $W_{\mathcal{I},\mathcal{J}}$ of a matrix $W$ is $h$-maximal if over all its $k \times l$ submatrices this volume is maximal up to a factor of $h$. The volume $v_2(W_{\mathcal{I},\mathcal{J}})$ is column-wise (resp. row-wise) $h$-maximal if it is $h$-maximal in the submatrix $W_{\mathcal{I},\mathcal{J}}$ (resp. $W_{\mathcal{I},\mathcal{J}}$). Such a volume is column-wise (resp. row-wise) locally $h$-maximal if it is $h$-maximal over all submatrices of $W$ that differ from it by a single column (resp. single row). We call volume $(h_c, h_r)$-maximal if it is both column-wise $h_c$-maximal and row-wise $h_r$-maximal. Likewise we define locally $(h_c, h_r)$-maximal volume. We call $1$-maximal and $(1,1)$-maximal volumes maximal. All these definitions can be restated in terms of $r$-projective volumes.

By comparing SVDs of two matrices $W$ and $W^+$ we obtain the following lemma.

**Lemma 43.** $\sigma_j(W)\sigma_j(W^+) = 1$ for all matrices $W$ and all subscripts $j$, $j \leq \text{rank}(W)$.

**Corollary 44.** $v_2(W)v_2(W^+) = 1$ and $v_{2,r}(W)v_{2,r}(W^+) = 1$ for all matrices $W$ of full rank and all integers $r$ such that $1 \leq r \leq \text{rank}(W)$.

Now we are ready to prove that for some proper constants $g$ and $h$ nonzero volume of a $k \times l$ submatrix of the matrix $W$ is $g$-maximal globally, that is, over all its $k \times l$ submatrices, if it is $h$-maximal locally, just over the $k \times l$ submatrices of two sketches at two successive C-A steps.

**Theorem 45.** Suppose that a nonzero volume of a $k \times l$ submatrix $W_{\mathcal{I},\mathcal{J}}$ is $(h, h')$-maximal for $h \geq 1$ and $h' \geq 1$ in a matrix $W$. Then this volume is $hh'$-maximal over all its $k \times l$ submatrices of the matrix $W$.

**Proof.** The matrix $W_{\mathcal{I},\mathcal{J}}$ has full rank because its volume is nonzero.

Fix any $k \times l$ submatrix $W_{\mathcal{I}',\mathcal{J}'}$ of the matrix $W$, apply Theorem 16 and obtain that

$$W_{\mathcal{I}',\mathcal{J}'} = W_{\mathcal{I}',\mathcal{J}}W_{\mathcal{I},\mathcal{J}}^+W_{\mathcal{I},\mathcal{J}'}.$$

If $k \leq l$, then first apply claim (iii) of Theorem 33 for $G := W_{\mathcal{I}',\mathcal{J}}$ and $H := W_{\mathcal{I},\mathcal{J}}^+$; then apply claim (i) of that theorem for $G := W_{\mathcal{I}',\mathcal{J}}W_{\mathcal{I},\mathcal{J}}^+$ and $H := W_{\mathcal{I},\mathcal{J}'}$ and obtain that

$$v_2(W_{\mathcal{I}',\mathcal{J}}W_{\mathcal{I},\mathcal{J}}^+W_{\mathcal{I},\mathcal{J}'}) \leq v_2(W_{\mathcal{I}',\mathcal{J}})v_2(W_{\mathcal{I},\mathcal{J}}^+)v_2(W_{\mathcal{I},\mathcal{J}'}).$$

If $k > l$ deduce the same bound by applying the same argument to the matrix equation

$$W_{\mathcal{I}',\mathcal{J}}^T = W_{\mathcal{I},\mathcal{J}}^TW_{\mathcal{I},\mathcal{J}}^+W_{\mathcal{I},\mathcal{J}'}^T.$$

Combine this bound with Corollary 44 for $W$ replaced by $W_{\mathcal{I},\mathcal{J}}$ and deduce that

$$v_2(W_{\mathcal{I}',\mathcal{J}'}) = v_2(W_{\mathcal{I}',\mathcal{J}}W_{\mathcal{I},\mathcal{J}}^+W_{\mathcal{I},\mathcal{J}'}) \leq v_2(W_{\mathcal{I}',\mathcal{J}})v_2(W_{\mathcal{I},\mathcal{J}}^+)/v(W_{\mathcal{I},\mathcal{J}}).$$

Recall that the matrix $W_{\mathcal{I},\mathcal{J}}$ is $(h, h')$-maximal and conclude that

$$h v_2(W_{\mathcal{I},\mathcal{J}}) \geq v_2(W_{\mathcal{I},\mathcal{J}'}) \text{ and } h' v_2(W_{\mathcal{I},\mathcal{J}}) \geq v_2(W_{\mathcal{I},\mathcal{J}}).$$

Substitute these inequalities into the above bound on the volume $v_2(W_{\mathcal{I}',\mathcal{J}'})$ and obtain that $v_2(W_{\mathcal{I}',\mathcal{J}'}) \leq hh'v_2(W_{\mathcal{I},\mathcal{J}})$.

□
10.3 From locally to globally \(h\)-maximal volumes of full rank submatrices in sketches of the same full rank

Notice that a \(k \times l\) submatrix of maximal volume in a rank-\(r\) matrix has rank \(r\) if \(r = \min\{k, l\}\), suppose that a \(r \times l\) submatrix \(U\) of a \(r \times n\) matrix \(W\) has nonzero column-wise \(h\)-maximal volume and estimate a value \(\hat{h} > h\) for which this volume is globally \(\hat{h}\)-maximal in \(W\). In view of Remark 28 similar estimate holds for a \(k \times r\) submatrix \(U\) of an \(m \times r\) matrix \(W\).

**Theorem 46.** Suppose that \(r \times l\) submatrix \(U\) has a nonzero column-wise locally \(h\)-maximal volume in the matrix \(W = (U \mid V) \in \mathbb{C}^{r \times n}\) for \(h \geq 1\). Then this submatrix has \(\hat{h}\)-maximal volume in the matrix \(W\) for \(\hat{h} = t_{n,r,h}^r\) and \(t_{n,r,h}^r = (p - r)rh^2 + 1\) of (57).

**Proof.** By means of orthogonalization of the rows of the matrix \(W\) obtain its factorization \(W = RV\) where \(R\) is a \(r \times r\) nonsingular matrix and \(Q = (R^{-1}U \mid R^{-1}V)\) is a \(r \times n\) unitary matrix and deduce from Corollary 37 that the volume of the matrix \(R^{-1}U\) is column-wise locally \(\hat{h}\)-maximal in the matrix \(Q\).

Therefore \(\sigma_r(R^{-1}U) \geq \sigma_r(Q)/t_{n,r,h}\) by virtue of Theorem 119.

Combine this bound with the relationships \(\sigma_r(Q) = 1\) and \(v_2(R^{-1}V) \geq (\sigma_r(R^{-1}V))^r\) and deduce that \(hv_2(R^{-1}U) \geq 1\) for \(\hat{h} = t_{n,r,h}^r\).

Notice that \(v_2(Q_1) \leq v_2(Q) = 1\) for any \(r \times l\) submatrix \(Q_1\) of \(Q\).

Hence the volume \(v_2(R^{-1}U)\) is \(\hat{h}\)-maximal in \(Q\).

Now Theorem 46 follows from Corollary 37.

**Example 47.** The bound of Theorem 46 is quite tight for \(r = h = 1\). Indeed the unit row vector \(v = \frac{1}{\sqrt{n}}(1, \ldots, 1)^T\) of dimension \(n\) is a \(r \times n\) matrix for \(r = 1\). Its coordinates are \(r \times r\) submatrices, all having volume \(\frac{1}{\sqrt{n}}\). Now notice that \(\sqrt{n} \approx \hat{h} = ((n - 1) + 1)^{1/2} = t_{n,1,1}\).

**Remark 48.** The theorem is readily extended to the case of a \(k \times n\) matrix \(W\) of rank \(r\), \(0 < r \leq k \leq n\), where \(r\)-projective volume replaces volume. Indeed row orthogonalization reduces the extended claim precisely to Theorem 46.

By following [GOSTZ10] we decrease the upper bound \(\hat{h} = t_{n,l,h}^r\) of Theorem 46 in the case where \(l = r\). We begin with a lemma.

**Lemma 49.** (Cf. [GOSTZ10].) Let \(W = (I_r \mid V) \in \mathbb{C}^{r \times n}\) for \(r \leq n\) and let the submatrix \(I_r\) have column-wise locally \(h\)-maximal volume in \(W\) for \(h \geq 1\). Then \(\|W\|_C \leq h\).

**Proof.** Let \(|w_{ij}| > h\) for an entry \(w_{ij}\) of the matrix \(W\), where, say, \(i = 1\). Interchange its first and \(j\)th columns. Then the submatrix \(I_r\) turns into the matrix \(R = \begin{pmatrix} w_{1j} & u^T \\ 0 & I_{r-1} \end{pmatrix}\). Hence \(v_2(R) = \|\text{det}(R)\| = |w_{1j}| > h\), and so \(I_r\) is not a column-wise locally \(h\)-maximal submatrix of \(W\). The contradiction implies that \(\|W\|_C \leq h\).

**Theorem 50.** (Cf. [GOSTZ10].) Suppose that \(r \times r\) submatrix \(U\) has a nonzero column-wise locally \(h\)-maximal volume in a matrix \(W = (U \mid V') \in \mathbb{C}^{r \times n}\) for \(h \geq 1\). Then this submatrix has \(\hat{h}\)-maximal volume in \(W\) for \(\hat{h} = h^{r/r^2}\).

**Proof.** Apply Lemma 49 to the matrix \(U^{-1}W = (I_r \mid V')\) for \(V = U^{-1}V'\) and obtain that \(\|U^{-1}W\|_C \leq h\). Hadamard’s bound (40) for \(M = V\) implies that the volume 1 of the submatrix \(I\) is \(\hat{h}\)-maximal in the matrix \(U^{-1}W\) for the claimed value of \(\hat{h}\). Now deduce from Corollary 37 that the submatrix \(U\) has \(\hat{h}\)-maximal volume in \(W\).
Remark 51. Clearly the bound $\hat{h} = t_{n,r,h}^r$ of Theorem 46 is larger than the bound $\tilde{h} = r^{r/2}$ of Theorem 50, but how much larger? Substitute a slightly smaller expression $((k-r)rh^2 + 1)^{1/2}$ for $t_{k,r,h} = ((k-r)rh^2 + 1)^{1/2}$ into the equation $\hat{h} = t_{n,r,h}^r$ and observe that the resulting decreased value is still larger than $\tilde{h} = h^{r/r^2}$ by a factor of $(n-r)^{r/2}$.

10.4 Optimization of the sizes of CUR generators

Let us optimize the size $k \times l$ of a CUR generator towards minimization of the bounds of Theorems 38 and 39 on the error norm $|||E|||_C$.

The bound of Theorem 38 turns into

$$|||E|||_C \leq (r + 1) h \sigma_{r+1}(W)$$

for $k = l = r$ and into

$$|||E|||_C \leq \sqrt{(1 + 1/b)(r + 1)} h \sigma_{r+1}(W)$$

if $k = r = (b+1)l - 1$ or $l = r = (b+1)k - 1$ and if $b > 0$, that is, we decrease the output error bound by a factor of $\sqrt{(r+1)/(1+1/b)}$ in the latter case.

The bound of Theorem 39 turns into

$$|||E|||_C \leq (1 + 1/b)h \sigma_{r+1}(W)$$

and is minimized for $k = l = (b+1)r - 1$ and a positive $b$.

This upper estimate shows that the volume is maximal where $\min\{k,l\} = r < \max\{k,l\}$ and that the $r$-projective volume is maximal where $l = k > r$. Furthermore the upper estimate of Theorem 39 for the norm $|||E|||_C$ converges to $\sigma_{r+1}(W)$ as $h \to 1$ and $b \to \infty$.

10.5 Maximal $r$-projective volume from maximal volume of a full rank matrix

The output errors bound of Theorems 38 and 39 are strengthened when we maximize $r$-projective volume for $r < k = l$. Next we reduce such a task to the maximization of the volume of $k \times r$ or $r \times l$ CUR generators of full rank for $r = \min\{k,l\}$.

Corollary 37 implies the following lemma.

Lemma 52. Let $M$ and $N$ be a pair of $k \times l$ submatrices of a $k \times n$ matrix and let $Q$ be a $k \times k$ unitary matrix. Then $v_2(M)/v_2(N) = v_2(QM)/v_2(QN)$ and if $r \leq \min\{k,l\}$, then also $v_{2,r}(M)/v_{2,r}(N) = v_{2,r}(QM)/v_{2,r}(QN)$.

Algorithm 53. Maximization of $r$-projective volume via maximization of the volume of a full rank matrix.

Input: Four integers $k$, $l$, $n$ and $r$ such that $0 < r \leq k$ and $r \leq l \leq n$, a $k \times n$ matrix $W$ of rank $r$ and a black box algorithm that computes a $r \times l$ submatrix of maximal volume in a $r \times n$ matrix of full rank $r$.

Output: A column set $\mathcal{J}$ such that the $k \times l$ submatrix $W_{:,\mathcal{J}}$ has maximal $r$-projective volume in the matrix $W$. 

31
1. Compute a rank-revealing QRP factorization $W = QRP$, where $Q$ is a unitary matrix, $P$ is a permutation matrix, $R = \begin{pmatrix} R' \\ O \end{pmatrix}$ and $R'$ is a $r \times n$ matrix.\(^{10}\) (See [GL13, Sections 5.4.3 and 5.4.4] and [GE96].)

2. Compute a $r \times l$ submatrix $R'_{i',j'}$ of $R'$ having maximal volume $v_2(R')$ and output the matrix $W_{i',j'}$.

The submatrices $U'$ and $\begin{pmatrix} U' \\ O \end{pmatrix}$ have maximal volume and maximal $r$-projective volume in the matrix $R$, respectively, by virtue of Theorem 33 and because $v_2(\tilde{U}) = v_2, r(\tilde{U}) = v_2, r(U')$. Therefore the submatrix $W_{i',j'}$ has maximal $r$-projective volume in the matrix $W$ by virtue of Lemma 52.

**Remark 54.** By transposing a horizontal input matrix $W$ and interchanging the integers $m$ with $n$ and $k$ with $l$ we extend the algorithm to computing a $k \times l$ submatrix of maximal or nearly maximal $r$-projective volume in an $m \times l$ matrix of rank $r$.

### 10.6 From local to global $h$-maximization of the volume of a submatrix

By combining Theorems 46 and 50 deduce that the volume of a $r \times l$ submatrix is $\tilde{h}$-maximal in a $r \times n$ matrix of rank $r$ for $\tilde{h} = t^r_{n,r,h}$ if $l > r$ and for $\tilde{h} = h^r r / 2$ if $l = r$ provided that the volume of the submatrix is column-wise locally $h$-maximal. The theorems imply that Algorithm 53 computes a $k \times l$ submatrix having maximal $r$-projective volume in an $m \times n$ matrix of rank $r$ for any integers $k$, $l$, $m$, $n$, and $r$ satisfying (13). The following theorem summarizes these observations.

**Theorem 55.** Given five integers $k$, $l$, $m$, $n$ and $r$ satisfying (13), write $p := \max\{k,l\}$ and suppose that two successive C-A steps (say, based on the algorithms of [GE96] or [P00]) have been applied to an $m \times n$ matrix $W$ of rank $r$ and have output $k \times l$ submatrices $W'_1$ and $W'_2 = W_{I_2,J_2}$ having nonzero $r$-projective column-wise locally $h$-maximal and row-wise locally $h'$-maximal volumes, respectively. Then the submatrix $W'_2$ has $h$-maximal $r$-projective volume in the matrix $W$ where either $\tilde{h} := (t_{n,r,h} t_{m,r,h})^r$ for $t^r_{p,r,h} := (p - r) r h^2 + 1$ of (57) if $p > r$ or $\tilde{h} := (h'^r)^r$ if $p = r$ and for real values $h$ and $h'$ slightly exceeding 1.

**Proof.** By applying Algorithm 53 reduce the claim of the theorem to the case where $r$-projective volume is equal to the volume of a matrix of full rank $r$. Then combine Theorems 45, 46 and 50. \(\square\)

**Remark 56.** (Cf. Remark 51.) Substitute a slightly smaller expression $((k - r) r h^2)^{1/2}$ for $t_{k,r,h} = ((k - r) r h^2 + 1)^{1/2}$ into the product $(t_{n,r,h} t_{m,r,h})^r$. Then its value decreases but still exceeds the bound $(h'h'^r)^r$ by a factor of $((m - r)(n - r))^{r/2}$.

### 10.7 Complexity, accuracy and extension of a two-step C-A loop

In this section we arrived at a C-A algorithm that computes a CUR approximation of a rank-$r$ matrix $W$. The following corollary, based on combining Theorems 38, 39 and 55, summarizes our study.

---

\(^{10}\)We can apply other rank-revealing factorizations instead.
Corollary 57. Under the assumptions of Theorem 55 suppose that we have applied a two-step C-A loop based on the algorithms that output $k \times l$ submatrices having nonzero $r$-projective column-wise and row-wise locally $h$ maximal volumes. Suppose that a canonical CUR LRA of the $m \times n$ input matrix $W$ of rank $r$ has been built on a CUR generator $W'_2 = W_{k,l}$ of rank $r$ output by the second C-A step. Then this computation outputs a CUR LRA with an error matrix $E$ such that $||E||_C \leq g(k,l,r) \bar{h} \sigma_{r+1}(W)$ for $\bar{h}$ of Theorem 55 and $g(k,l,r)$ denoting the functions $f(k,l)$ of Theorem 38 or $f(k,l,r)$ of Theorem 39. In particular the computation can rely on application of the algorithms of [GE96] or [P00], and then it involves $(m+n)r$ memory cells and $O((m+n)r^2)$ flops.

Remark 58. Theorem 31 enables us to extend Algorithm 53, Theorem 55 and Corollary 57 to the case where an input matrix $W$ of numerical rank $r$ is closely approximated by a rank-$r$ matrix (see Remark 32).

The factor $\bar{h}$ of Theorem 55 is large already for moderately large integers $r$, but

(i) in many important applications the numerical rank $r$ is small,

(ii) our upper bounds $\bar{h}$ are overly pessimistic on the average input in view of our study in Section 9.2,

(iii) we can continue iterations of the C-A algorithm, counting on strict decrease of the volume of a CUR generator and consequently an upper bound on Chebyshev’s norm of the CUR output error matrix in every C-A step.

Such a progress is formally ensured in the algorithm of [GOSTZ10], which locally maximizes the volume in every C-A step. In the worst case the decrease is slow, but empirically the convergence is fast, and we can monitor the dynamic of the error norm by estimating a posteriori error for the computed CUR LRA superfast (see Section 3.4).

PART III. RANDOMIZED ALGORITHMS FOR CUR LRA

11 Computation of LRA with Random Subspace Sampling Directed by Sampling Probabilities

A CUR generator is expected to define a close CUR LRA if it has been defined by sufficiently many rows and columns of an input matrix sampled according to properly computed sampling probabilities. This is the third CUR criterion, based on statistical approach to generating CUR LRA. The CUR LRA algorithm of [DMM08] implements this approach and is superfast except that it computes sampling probabilities only fast for the worst case input. We, however, bypass that stage for the average input matrix allowing its close LRA and with a high probability for a small norm perturbation of a factor-Gaussian input matrix. Moreover we refine a crude but reasonably close LRA of any input superfast.

11.1 Fast accurate LRA with sampling probabilities defined by a singular space

Given top SVD of an $m \times n$ matrix $W$ of numerical rank $r \ll \min\{m,n\}$, we can apply randomized Algorithm 24a, which with a high probability computes a close CUR LRA of the matrix $W$. Moreover we can decrease the output error bound by sampling more columns...
and rows. By following [DMM08] and sampling sufficiently many columns and rows, one can ensure a nearly optimal output error bound such that
\[ \|W - CUR\|_F \leq (1 + \epsilon)\tilde{\sigma}_{r+1} \] (46)

for \( \tilde{\sigma}_{r+1} \) of (9) and any positive \( \epsilon \).

Let us supply some details. Let the columns of an \( n \times r \) unitary matrix \( V^{(r)T} = (v_{h,j}^{(r)})_{h,j=1}^{n,r} = (V_j^{(r)})_{j=1}^{r} \) span the right singular subspace associated with the \( r \) largest singular values of an input matrix \( W \) of numerical rank \( r \). Fix scalars \( \beta \) and \( p_1, \ldots, p_n \) such that
\[ 0 < \beta \leq 1, \quad p_j \geq (\beta/r)\|v_j^{(r)}\|_2 \text{ for } j = 1, \ldots, n, \quad \text{and } \sum_{j=1}^{n} p_j = 1, \] (47)

and then call the scalars \( p_1, \ldots, p_n \) the \textit{SVD-based sampling probabilities} for the matrix \( W \) (cf. (62)). They stay invariant if we post-multiply the matrix \( V^{(r)} \) by a unitary matrix. Furthermore notice that
\[ p_j = v_j^{(r)} \text{ for } j = 1, \ldots, n \text{ and } \beta = 1. \] (48)

For any \( m \times n \) matrix \( W \), [HMT11, Algorithm 5.1] computes the matrix \( V^{(r)} \) and sampling probabilities \( p_1, \ldots, p_n \) by using \( mn \) memory units and \( O(mnr) \) flops. Given an integer parameter \( l, 1 \leq l \leq n \), and sampling probabilities \( p_1, \ldots, p_n \), Algorithms 115 and 116, reproduced from [DMM08], compute sampling and rescaling matrices, \( S = S_{W,l} \) and \( D = D_{W,l} \), respectively. In particular Algorithms 115 and 116 sample and rescale either exactly \( l \) columns of an input matrix \( W \) or at most its \( l \) columns in expectation (the \( i \)th column with probability \( p_i \) or \( \min\{1, lp_i\} \), respectively.)

Then [DMM08, Algorithms 1 and 2] compute a CUR LRA of a matrix \( W \) as follows.

**Algorithm 59.** CUR LRA by using SVD-based sampling probabilities.

**INPUT:** A matrix \( W \in \mathbb{C}^{m \times n} \) with \( \text{nr}(W) = r > 0 \).

**INITIALIZATION:** Choose integers \( k \) and \( l \) satisfying (13) and \( \beta \) and \( \bar{\beta} \) in the range \((0,1]\).

**COMPUTATIONS:**
1. Compute the sampling probabilities \( p_1, \ldots, p_n \) of (47).
2. Compute sampling and rescaling matrices \( S \) and \( D \) by applying Algorithm 115 or 116. Compute and output a CUR factor \( C := WS \).
3. Compute sampling probabilities \( \bar{p}_1, \ldots, \bar{p}_m \) satisfying relationships (47) with the following replacements: \( \{p_1, \ldots, p_n\} \leftarrow \{\bar{p}_1, \ldots, \bar{p}_m\} \), \( W \leftarrow (CD)^T \) and \( \beta \leftarrow \bar{\beta} \).
4. By applying Algorithm 115 or 116 to these probabilities compute \( k \times l \) sampling matrix \( \bar{S} \) and \( k \times k \) rescaling matrix \( \bar{D} \).
5. Compute and output a CUR factor \( R := \bar{S}^T \bar{D} \).
6. Compute and output a CUR factor \( U := DM^+ \bar{D} \) for \( M := \bar{D} \bar{S}^T WSD \).

**Remark 60.** At stage 6 we can alternatively apply the equivalent but simpler expressions \( U := (S^T W S)^+ = (S^T C)^+ = (RS)^+ \) and then guard against numerical stability problems.

Bound (46) is expected to hold for the output of the algorithm if we bound the integers \( k \) and \( l \) by combining [DMM08, Theorems 4 and 5] as follows.
Remark 63. The bounds (as long as they stay in their ranges) and still ensure the randomized error bound (46).

\[ l_\ast = 3200r^2/(\epsilon^2 \beta) \leq n \quad \text{and} \quad k_\ast = 3200l^2/(\epsilon^2 \beta) \leq m \]  

or

\[ l_\ast = c \ r \ \log(r)/(\epsilon^2 \beta) \leq n \quad \text{and} \quad k_\ast = c \ l \ \log(l)/(\epsilon^2 \beta) \leq m \]  

for a sufficiently large constant \( c \) or. Then fix two integers \( k \) and \( l \) such that \( k_\ast \leq k \leq m \) and \( l_\ast \leq l \leq n \) and apply Algorithm 59 invoking at stages 2 and 4 either Algorithm 115 for the choice of \( l_\ast \) and \( k_\ast \) under (49) or Algorithm 116 for the choice of \( l_\ast \) and \( k_\ast \) under (50). Then in both cases bound (46) holds with a probability of at least 0.7.

Remark 62. We fix sampling probabilities \( p_i \) according to equations (47) and minimize the lower bounds \( k_\ast \) on \( k \) and \( l_\ast \) on \( l \) if we choose \( \beta = \bar{\beta} = 1 \). By decreasing the values of \( \beta \) and \( \bar{\beta} \) we increase the lower bounds \( l_\ast \) and \( k_\ast \) by factors of 1/\( \beta \) and 1/(\( \bar{\beta}^2 \beta \)), respectively, but at that price allow the probabilities \( p_i \) vary in certain ranges, which increase as the values of \( \beta \) and \( \bar{\beta} \) decrease. Then we can apply approximate values of the probabilities \( p_i \) (as long as they stay in their ranges) and still ensure the randomized error bound (46).

Remark 61. The bounds \( k_\ast \leq m \) and \( l_\ast \leq n \) imply that either \( \epsilon^6 \geq 3200^3 r^4/(m\beta^2 \bar{\beta}) \) and \( \epsilon^2 \geq 3200r/(n\beta) \) if Algorithm 115 is applied or \( \epsilon^4 \geq c^2 r \log(r) \log(cr \log(r)/(\epsilon^2 \beta))/\bar{\beta})/(m\beta^2 \bar{\beta}) \) and \( \epsilon^2 \geq cr \log(r)/(n\beta) \) if Algorithm 116 is applied.

Complexity estimates: Overall Algorithm 59 involves \( kn + ml + kl \) memory cells and \( O((m + k)l^2 + kn) \) flops in addition to \( mn \) cells and \( O(mnr) \) flops used for computing SVD-based sampling probabilities at stage 1. Except for that stage the algorithm is superfast if \( k + l^2 \ll \min\{m, n\} \).

11.2 Superfast LRA with sampling probabilities for random and average inputs

Next we are going to apply Algorithm 59 to diagonally scaled, left and right factor-Gaussian \( m \times n \) matrices with expected rank \( r \) and to the average of such matrices (see Definitions 4 and 6). We begin with definitions and Theorem 66 of independent interest.

Definition 64. \( \chi^2(s) := \sum_{i=1}^{s} g_i^2 \) is the \( \chi^2 \)-function for s i.i.d. Gaussian variables \( g_1, \ldots, g_s \).

Definition 65. Let \( n \geq r \) and the \( n \) scalars \( p_1, \ldots, p_n \) satisfy (47) for a scalar \( \beta \) in (0, 1] and a \( r \times n \) Gaussian matrix. Then call these \( n \) scalars \( \mathcal{G}_{n,r,\beta} \)-sampling probabilities.

Theorem 66. Let \( W = GH \) for \( G \in \mathbb{C}^{m \times r} \) and \( H \in \mathbb{C}^{r \times n} \) and let \( r = \text{rank}(G) = \text{rank}(H) \leq \min\{m, n\} \). Then the matrices \( W \) and \( W^T \) share their SVD-based sampling probabilities with the matrices \( G^T \) and \( H \), respectively.

Proof. Let \( G = SG \Sigma_GT_G^* \in \mathbb{C}^{m \times r} \) and \( H = SH \Sigma_HT_H^* \) be SVDs.

Write \( M := GT_G^*S_H^*S_H \) and let \( M = SM \Sigma_M^*T_M^* \) be SVD.

Notice that \( \Sigma_G, T_G, S_H \) and \( \Sigma_H \) are \( r \times r \) matrices.

Consequently so are the matrices \( S_M, \Sigma_M \) and \( T_M^* \).

Hence \( W = SG \Sigma_MT_H^* \) where \( SG = SG \Sigma_M \) and \( T_H^* = T_M^*T_H^* \) are unitary matrices.

Therefore \( W = SG \Sigma_MT_H^* \) is SVD.

35
Hence the columns of the unitary matrices $\bar{S}_G$ and $\bar{T}_H^T$ span the top right singular spaces of the matrices $W^T$ and $W$, respectively. But so do the columns of the matrices $S_G$ and $T_H^*$ as well because $\bar{S}_G = S_G S_M$ and $\bar{T}_H^T = T_M^* \bar{T}_H^*$ where $S_M$ and $T_M^*$ are $r \times r$ unitary matrices. This proves the theorem.

**Corollary 67.** Under the assumptions of Theorem 66 the SVD-based sampling probabilities for the matrix $W$ are $G_{n,r,\beta}$-sampling probabilities if $H$ is a Gaussian matrix, and the SVD-based sampling probabilities for the matrix $W^T$ are $G_{m,r,\beta}$-sampling probabilities if $G$ is a Gaussian matrix.

The following well-known result implies that the norms of the rows of the basis matrix $V^{(r)}$ for a right singular space of an $n \times r$ Gaussian matrix $W$ are uniformly distributed.

**Theorem 68.** Let $M = GH$ for $G \in \mathbb{C}^{m \times r}$ and $H \in \mathbb{C}^{r \times n}$ for $r \leq \min\{m,n\}$. Then

(i) the rows of the matrix $M$ and its left singular vectors are uniformly distributed on the unit sphere if $G$ is a Gaussian matrix and

(ii) the columns of the matrix $M$ and its right singular vectors are uniformly distributed on the unit sphere if $H$ is a Gaussian matrix.

**Proof.** We prove claim (ii); its application to the matrix $M^T$ yields claim (i).

Let $V$ be an $n \times n$ unitary matrix. By virtue of Lemma 3 the matrix $HV$ is Gaussian. Therefore the entries of the matrices $M$ and $MV$ as well as of the matrices $U_M$ and $U_MV$ of their right singular spaces have the same probability distribution pairwise. In particular the distribution of the entries of the unitary matrix $U_M$ is orthogonally invariant, and therefore these entries are uniformly distributed over the set of unitary matrices.

Next assume that $n \gg r$ and recall from [E89, Theorem 7.3] that $\kappa(W) \to 1$ as $r/n \to 0$ if $W \in \mathcal{G}^{r \times n}$. Therefore for $r \ll n$ a matrix $W \in \mathcal{G}^{r \times n}$ is close to a scaled unitary matrix, and hence within a constant factor is close to the unitary matrix of its right singular space.

Now observe that the norms of the column vectors of such a matrix $W$ are i.i.d. random variables $\chi^2(r)$ and therefore are quite strongly concentrated in a reasonable range about the expected value of such a variable. So we can obtain reasonably good approximations to SVD-based sampling probabilities for such a matrix by choosing the norms $\nu_j^{(r)}$ equal to each other for all $j$, and then we can satisfy expressions (47) and consequently the bound (46) by choosing a reasonably small value of $\beta$.

**Lemma 69.** (Cf. [LM00, Lemma 1]) Let $\chi^2_r = \sum_{i=1}^r X_i^2$, where $X_i \sim \mathcal{N}(0,1)$. Then for every $x > 0$, we have that

\[
\text{Probability}\{\chi^2_r - r \geq 2\sqrt{rx} + 2rx\} \leq \exp(-x)
\]

**Corollary 70.** Let $w_j \sim \chi^2(r)$ be independent Chi-square random variables with degrees of freedom $r$, for $j = 1, 2, \ldots, n$. Then for any $\beta \in (0, 1]$,

\[
\text{Probability}\left\{\frac{1}{n} > \beta \cdot \frac{w_j}{nr} \text{ for all } j = 1, 2, \ldots, n\right\} \geq 1 - \exp\left(-c \cdot \frac{\ln n}{r}\right),
\]

where $c = \frac{1 - \sqrt{2\beta - \beta^2}}{2\beta}$.  

36
Proof. By Lemma 69, for any $j \in \{1, 2, \ldots, n\}$,

\[
\text{Probability}\left\{ \frac{1}{n} > \beta \cdot \frac{w_j}{nr} \right\} = 1 - \text{Probability}\left\{ w_j - r \geq \left( \frac{1}{\beta} - 1 \right) r \right\} \\
\geq 1 - \exp\left( -c/r \right)
\]

where $c = \frac{1 - \sqrt{2 - \beta^2}}{2 \beta}$. The independence of $w_j$ implies that

\[
\text{Probability}\left\{ \frac{1}{n} > \beta \cdot \frac{w_j}{nr} \text{ for all } j = 1, 2, \ldots, n \right\} \geq (1 - \exp(-c/r))^n \\
\geq 1 - \exp\left( -c \cdot \frac{\ln n}{r} \right).
\]

In combination with Corollary 67 this enables us to bypass the bottleneck stage of the computation of sampling probabilities for a diagonally scaled $m \times n$ factor-Gaussian matrix and of an $m \times n$ right factor-Gaussian matrix, in both cases we assume expected rank $r \ll n$.

Now let such a matrix $W$ be in the class of either left or right factor-Gaussian matrices with expected rank $r \ll n$, but let us not know in which of the two classes. Then by assuming that both matrices $W$ and $W^T$ are right factor-Gaussian and applying Algorithm 59 to both of them, we can compute their sampling probabilities superfast. We would expect to obtain an accurate CUR LRA for at least one of them, and we would verify correctness of the output by estimating a posteriori error norm superfast.

Finally let us extend our analysis and superfast accurate CUR LRA algorithms to small norm perturbations of the matrices of these classes. The output error bounds of Algorithm 59 are stable in small perturbations of sampling probabilities (cf. Remark 62). The following result implies that the sampling probabilities are stable in perturbation of a matrix:

**Theorem 71.** (See [GL13, Theorem 8.6.5].) Suppose that

\[ g =: \sigma_r(M) - \sigma_{r+1}(M) > 0 \text{ and } ||E||_F \leq 0.2g. \]

Then there exist unitary matrix bases $B_{r,\text{left}}(M)$, $B_{r,\text{right}}(M)$ $B_{r,\text{left}}(M + E)$ and $B_{r,\text{right}}(M + E)$ for the singular spaces associated with the $r$ largest singular values of the matrices $M$ and $M + E$, respectively, such that

\[
\max\{||B_{r,\text{left}}(M + E) - B_{r,\text{left}}(M)||_F, ||B_{r,\text{right}}(M + E) - B_{r,\text{right}}(M)||_F\} \leq 4||E||_F/g.
\]

For example, if $\sigma_r(M) \gg \sigma_{r+1}(M)$, implying that $g \approx \sigma_r(M)$, then the upper bound on the right-hand side is approximately $4||E||_F/\sigma_r(M)$.

### 11.3 Refinement of an LRA by using sampling probabilities

Next suppose that $AB = W' \approx W$ is a reasonably close but still crude LRA for a matrix $W$ and let us refine it. First approximate top SVD of the matrix $W'$ by applying to it Algorithm 10, then fix a positive value $\beta' \leq 1$ and compute sampling probabilities $p'_1, \ldots, p'_n$ of $W'$ by applying (47) with $\beta'$ replacing $\beta$. Perform all these computations superfast.

By virtue of Theorem 71 the computed values $p'_1, \ldots, p'_n$ approximate the sampling probabilities $p_1, \ldots, p_n$ of the matrix $W$, and so we satisfy (47) for an input matrix $W$.  

37
and properly chosen parameters $\beta$ and $\bar{\beta}$. Then we arrive at a CUR LRA satisfying (46) if we sample $k$ rows and $l$ columns within the bounds of Theorem 61, which are inversely proportional to $\beta^2\bar{\beta}$ and $\beta$, respectively.

**Remark 72.** LRAs of a matrix $W$ computed by superfast algorithms of the previous sections have error norm bound $O(\sigma_{r+1}(W))$, and so by virtue of Theorem 71 the perturbation of sampling probabilities caused by the LRA errors is also in $O(\sigma_{r+1}(W))$ if the ratio $\sigma_r(W)/\sigma_{r+1}(W)$ is reasonably large.

11.4 A fast CUR LRA algorithm sampling fewer rows

By virtue of Theorem 61 Algorithm 59 outputs nearly optimal CUR LRA provided that the integers $l$ and particularly $k$ are fairly large, even for relatively large values of $\epsilon \leq 1$ (cf. Remark 63). These upper bounds, however, seem to be overly pessimistic in the case of various real world inputs. In the tests in [DMM08, Section 7], Algorithm 59 has output accurate CUR LRAs for such inputs while using moderate or only reasonably large integers $l$ and $k$.

The upper bounds of Theorem 61 on the parameters $k$ and $l$ have been decreased in the subsequent papers. Already the algorithm of the paper [MD09] extends the results of [DMM08] by sampling as many columns and much fewer rows. Namely the algorithm computes CUR factor $C$ and $R$ by applying stages 1 and 2 of Algorithm 59 to the matrices $W$ and $W^T$, respectively, and then computes a nucleus

$$U = C^+WR^+.\tag{51}$$

Theorem 61 is readily extended if under the same bounds on the integer $l$ we choose $k = l$ and increase by twice the value $\epsilon$ and the bound of the probability of failure.

The proof relies on the following simple estimates of independent interest:

$$|W - CUR| \leq |W - CC^+W| + |CC^+W - CC^+WR^+R| \leq |W - CC^+W| + |W - WR^+R|.$$  

The latter inequality follows because the multiplier $CC^+$ is a projection matrix and thus does not increase the spectral and Frobenius norms.

As by-product the inequality implies the following extension of Theorem 16.

**Theorem 73.** Suppose that (16) and (51) hold. Then equation (15) holds.

The nucleus $U$ of (51) is not canonical, and its computation is not superfast. One can ignore (51) and compute superfast the canonical nucleus $U$ of (22), use it unless its norm is large, and estimate the output error norm superfast.

This yields verifiable CUR LRA for a large class of inputs, although we cannot formally extend the worst case error estimate of [MD09] to this algorithm. Indeed for a fixed pair of CUR factors $C$ and $R$ the nuclei of (22) and (51) define the same CUR LRA for a matrix $W$ under (16), by virtue of Theorem 73, but the perturbation of this matrix can make different impacts onto these two candidate nuclei: the perturbation can stronger affect the pseudo inverse of a CUR generator $W_{k,l}$ than those of the more rectangular factors $C$ and $R$.

11.5 Can we succeed with computing accurate LRA superfast if we sample fewer row and columns?

Similar challenges arise for the extensions of Algorithm 59 in the subsequent papers [WZ13], [BW14] and [BW17]. They decrease the asymptotic upper bounds of [DMM08] on the
parameters \( k \) and \( l \) in terms of \( m, n \) and \( r \), but the overhead constants of these bounds are hidden in the "O" and "poly" notation. Similar comments apply to the paper [SWZ17], which dramatically advanced the state of the art for LRA in terms of \( l_1\)-norm.

We have already observed some difficulties with the extension of the superfast and accurate performance of Algorithm 59 to the algorithm of [MD09], but the task can become harder for the much more involved algorithms of [BW17] and [SWZ17].

Here are some relevant important research challenges regarding performance of the above algorithms and other algorithms that define CUR LRA based on sampling probabilities:

(i) Extend the analysis of the algorithms to estimating the overhead constants hidden in the "O" and "poly" notation for the complexity bounds.

(ii) Complement the formal analysis with test results on real world input data.

(iii) Analyze and test the output accuracy of the algorithms when they use uniform sampling probabilities being applied to perturbed factor-Gaussian inputs and the average inputs allowing LRA.

(iv) Analyze and test performance of the algorithms when their input matrices are given together with their crude but reasonably accurate LRAs.

Some ground for optimism can be given by the following observation that the computation of the Lewis weights, involved in [SWZ17], can be performed superfast in the cases of perturbed factor-Gaussian inputs, the average inputs allowing LRA and the inputs given with their crude but reasonably close LRAs.

Indeed, for a matrix \( V \), let \( v_i^T \) denote its \( i \)th row vector and define \( l_p \) Lewis weights \( x_{i,p} \) for \( 1 \leq p < 4 \) as follows: \( x_{i,2}^{2/p} := v_i^T (V X_1^{1-2/p})^{-1} v \) for all \( i \) (see [SWZ17, Section 3.5]).

In particular \( x_{i,2} := v_i^T (V^T V)^{-1} v \) for all \( i \), and our superfast algorithms for SVD-based sampling probabilities of random and average matrices and matrices given with their crude but close LRAs are readily extended.

Moreover, we can extend the computations from \( l_2\)-weights to \( l_1\)-weights. Indeed \( x_{i,1}^{2} := v_i^T (V X_1^{1} V)^{-1} v \) for all \( i \), and we can write \( x_{i,1,j+1} := v_i^T (V X_1^{1} V)^{-1} v \) for all \( i \) and \( j = 0, 1, \ldots \). Having computed Lewis weights \( x_{i,2} \), substitute the matrix \( X_{1,0} := X_2 \) on the right-hand side of these expressions for \( j = 0 \) and then recursively update the approximate weights \( x_{i,1,j} \), by recursively increasing \( j \) by 1, with the hope for fast convergence.

We can increase the chances for it by applying homotopic continuation, that is, making transition from \( l_2 \) to \( l_1 \) in \( q \) steps, each of the size \( 1/q \) for a fixed integer \( q > 1 \).

Even if such observations cannot be extended to all algorithms of the papers [BW17] and [SWZ17], their powerful techniques could help improve practical performance of Algorithm 59 and decrease the overhead constants of the estimates of Theorem 61.

12 CUR LRA with randomized pre-processing

In this section we compute CUR LRA by applying randomized multiplicative pre-processing to an input matrix allowing close LRA.

12.1 Pre-processing into factor-Gaussian matrices: a theorem

Next we prove that pre-processing with Gaussian multipliers turns any matrix of numerical rank \( r \) into a small norm perturbation of a diagonally scaled, left or right factor-Gaussian matrix, for which we can compute CUR generators satisfying any or all of the three CUR
criteria superfast and can apply the error estimates of Sections 6, 9.2 and 11.2 to the resulting CUR LRAs.

**Theorem 74.** For \( k, l, m, n \) and \( r \) satisfying (13), \( G \in \mathcal{G}^{k \times m}, \ H \in \mathcal{G}^{n \times l} \), an \( m \times n \) well-conditioned matrix \( W \) of rank \( r \) and \( \nu_{p,q} \) and \( \nu_{p,q}^+ \) of Definition 7 it holds that

(i) \( GW \) is a left factor-Gaussian matrix of expected rank \( r \), \( ||GW|| \leq ||W|| \nu_{k,r} \) and \( ||(GW)^+|| \leq ||W^+|| \nu_{k,r}^+ \) if \( m \geq n \),

(ii) \( WH \) is a right factor-Gaussian matrix of expected rank \( r \), \( ||WH|| \leq ||W|| \nu_{r,l} \) and \( ||(WH)^+|| \leq ||W^+|| \nu_{r,l}^+ \) if \( m \leq n \), and

(iii) \( GWH \) is a diagonally scaled factor-Gaussian matrix of expected rank \( r \), \( ||GWH|| \leq ||W|| \nu_{k,r} \nu_{r,l} \) and \( ||(GWH)^+|| \leq ||W^+|| \nu_{k,r}^+ \nu_{r,l}^+ \) if \( m = n \).

**Proof.** Let \( W = S_W \Sigma_W T_W^* \) be SVD where \( \Sigma_W \) is the diagonal matrix of the singular values of \( W \); it is well-conditioned since so is the matrix \( W \). Then

(i) \( GW = G \Sigma_W T^* \),

(ii) \( WH = S_W \Sigma_W H \), and

(iii) \( GWH = G \Sigma_W \bar{H} \)

where \( \bar{G} := GS_W \) and \( \bar{H} := T_W^* H \) are Gaussian matrices by virtue of orthogonal invariance of Gaussian matrices (see Lemma 3) because \( S_W \in \mathbb{C}^{m \times r} \) and \( T_W \in \mathbb{C}^{r \times n} \) are unitary matrices, \( r \leq mn\{m,n\} \), \( G \in \mathcal{G}^{m \times m} \) and \( H \in \mathcal{G}^{n \times n} \).

Combine these bounds with Lemma 27 and obtain the claimed norm estimates. \( \square \)

### 12.2 Computation of a CUR LRA with Gaussian multipliers and their pseudo inverses

The latter results motivate application of the following simple algorithm (apply it to the transposed matrix \( W^T \) if \( m \geq n \)).

**Algorithm 75.** *Computation of a CUR LRA with Gaussian pre-processing.*

**Input:** A matrix \( W \in \mathbb{C}^{m \times n} \) having numerical rank \( r \ll m \leq n \).

**Output:** A CUR LRA of the matrix \( W \).

**Initialization:** Generate an \( n \times u \) Gaussian matrix \( H \) for \( u \geq n \).

**Computations:**

1. Compute the \( m \times u \) matrix \( WH \).

2. Compute SVD-based sampling probabilities \( p_1, \ldots, p_u \) for this matrix satisfying (47) for \( n = u \), \( ||v_1^{(r)}|| = \cdots = ||v_n^{(r)}|| \) and a sufficiently small \( \beta \).

3. Compute a CUR LRA of the matrix \( WH = CUR_H \) by applying to it Algorithm 59 for these sampling probabilities.

4. Compute a CUR LRA of the matrix \( W = CUR \) where \( R = R_H H^+ \).

The computations by the algorithm are superfast except for its stages 1 and 4. We can dramatically accelerate them by replacing the Gaussian multiplier \( H \) by sparse and structured multipliers of Sections 12.5 – 12.7, although one can recall that for the worst case input and even for the input family of \( \pm \delta \) matrices neither algorithm cannot be both superfast and accurate.

40
**Remark 76.** Having performed Algorithm 75, one can estimate a posteriori errors for the computed CUR LRA of $W$ superfast. If the accuracy of the output is reasonably good, but is still insufficient, then one can refine this LRA superfast by applying the techniques of Section 11.3.

**Remark 77.** If $m > n$, then instead of pre-processing by means of post-multiplication one should apply pre-multiplication by Gaussian or random sparse and structured multipliers. For $m = n$ one can apply pre-processing in any or both of these ways. In the extension of our study to $d$-dimensional tensors one can consider pre-processing by using up to $d$ directions for application of random multipliers.

### 12.3 Computation of a CUR LRA with Gaussian sampling

For an alternative modification of Algorithm 75 we can use multipliers of smaller sizes $k \times l$ for $k, l, m, n$ and $r$ satisfying (13) if we only seek a $k \times l$ CUR generator. At this stage we can incorporate fast and highly accurate randomized algorithms for LRA surveyed and thoroughly analyzed in [HMT11, Sections 4, 10 and 11]. We can extend them to computation of a CUR LRA by applying superfast Algorithms 10 and 24 or 24a, but next we describe a simpler and faster extension, which empirically reaches superfast level.

The idea is to apply random $n \times l$ multipliers $H_l$ (say, those of [HMT11] or [PZ16]) not for random sampling of columns, but for pre-processing the input $W$. Then we superfast compute a row set $I$ for a CUR LRA of the tall-skinny matrix $WH_l$ by applying the algorithms of [GE96] or our Remark 121 and finally reuse this set as the row index set in a CUR LRA for the input matrix $W$.

At its first slower stage of pre-processing, the resulting algorithm proceeds fast (by incorporating a recipe from [HMT11]) but then computes a CUR LRA superfast.

**Algorithm 78.** Computation of a CUR LRA with Gaussian pre-processing.

**Input:** Five integers $k$, $l$, $m$, $n$ and $r$ such that $\max\{k, l\} \ll \min\{m, n\}$ and (13) holds and generate an $m \times n$ matrix $W$ of numerical rank $r$.

**Output:** A CUR generator $W_{\mathcal{I}, \mathcal{J}} \in \mathbb{C}^{k \times l}$.

**Initialization:** Fix a sufficiently large integer $\bar{l} \geq l$ and an $n \times \bar{l}$ Gaussian matrix $\bar{H}$.

**Computations:**

1. Compute $m \times \bar{l}$ matrix $\bar{W}H$ (cf. [HMT11]).

2. By applying to it the algorithms of [GE96] compute its $k \times \bar{l}$ submatrix $(\bar{W}H)_{\mathcal{I}}$.

3. Compute a $k \times l$ submatrix $W_{\mathcal{I}, \mathcal{J}}$ of the matrix $W_{\mathcal{I}}$ by applying to it the algorithms of [GE96]. Output this submatrix.

**Theorem 79.** For the output matrix $W_{\mathcal{I}, \mathcal{J}}$ of Algorithm 78, for $t_{p, r, h}$ of (57) and $\nu_{p, q}^+$ of Definition 7 where $\mathbb{E}(\nu_{p, q}^+)^2 < e\sqrt{p}/|q-p|$ (see Theorem 104) and where we can assume that $h \approx 1$ (see (57)), it holds that

$$||W_{\mathcal{I}, \mathcal{J}}^+|| \leq ||W^+|| \nu_{k, l}^+ \nu_{t, n, l, h}^+.$$

**Proof.** Theorem 74 implies that $||(W\bar{H})^+|| \leq ||W^+|| \nu_{k, \bar{l}}^+$. Furthermore $||(W\bar{H})_{\mathcal{I}}|| \leq ||(W\bar{H})^+|| t_{m, k, h}$ and $||W_{\mathcal{I}, \mathcal{J}}^+|| \leq ||W_{\mathcal{I}}^+|| t_{n, l, h}$ (see Corollary 120).

Now let $W_{\mathcal{I}} = S\Sigma T^*$ be SVD; then $(W\bar{H})_{\mathcal{I}} = W_{\mathcal{I}} \bar{H} = S\Sigma G_{r, l}$ because the matrix $T^*$ is orthogonal and $\bar{H}$ is Gaussian (see Lemma 3). Hence $||W_{\mathcal{I}, \mathcal{J}}^+|| \leq ||(W\bar{H})_{\mathcal{I}}^+|| \nu_{r, l}^+$. Combine the above bounds and obtain the theorem. □
The complexity estimates for the algorithm: We need $N Z_W + nl \leq (m + \bar{l})n$ memory cell and less than $2\bar{l} \cdot N Z_W$ or $(2n - 1)m\bar{l}$ flops at stage 1, $m\bar{l}$ cells and $O(m\bar{l}^2)$ flops at stage 2 and $kn$ cells and $O(nk^2)$ flops at stage 3. Thus stages 2 and 3 are superfast.

Remark 80. The input matrix at stage 2 and the transpose of the input matrix at stage 3 are tall-skinny, and so by applying randomized techniques of Remark 121 we can decrease the error bound of Theorem 79 by almost a factor of $t_{m,k,h} t_{n,l,h}$ and can decrease the flop bounds at stage 2 to $O(m\bar{l})$ flops and at stage 3 to $O(nk)$. An application of C-A iterations is a heuristic alternative where one optimizes the Chebyshev error norm.

Remark 81. It is straightforward to extend our study to application of Cynical algorithms to $q \times s$ sketches for $k \leq q \leq m$ and $l \leq s \leq n$.

12.4 Randomized Hadamard and Fourier multipliers

Towards acceleration of the bottleneck stages 1 and 4 of Algorithm 75 and stages 1 of Algorithm 78 we recall the matrices of randomized Hadamard and Fourier transforms in Appendix B and their SRHT and SRFT modifications and next recall the following result from [T11] and [HMT11, Theorem 11.1].

Theorem 82. Suppose that $T_r$ is a $r \times n$ unitary matrix, $H_l$ is an $n \times l$ matrix of SRHT or SRFT, and

$$4(\sqrt{r} + \sqrt{8 \log \left(\frac{rn}{l}\right)})^2 \log(r) \leq l \leq n.$$  \hspace{1cm} (52)

Then with a failure probability in $O(1/k)$ it holds that

$$0.40 \leq \sigma_r(T_rH_l) \quad \text{and} \quad \sigma_1(T_rH_l) \leq 1.48.$$  \hspace{1cm} (53)

Corollary 83. Under the assumptions of Theorem 82, let $W_r$ be a $r \times n$ matrix with singular values $\sigma_j = \sigma_j(W_r)$, $j = 1, \ldots, r$. Let $\Sigma_r = \text{diag}(\sigma_j^r)_{j=1}^r$, $\sigma_r > 0$ and let an integer $l$ satisfy (52). Then

$$\frac{0.40}{\sigma_r} \leq \sigma_r(W_rH_l) \quad \text{and} \quad \sigma_1(W_rH_l) \leq 1.48\sigma_1$$

with failure probability in $O(1/k)$.

Proof. Consider SVD $W_r = S_r\Sigma_rT_r$ with $\Sigma_r = \text{diag}(\sigma_j^r)_{j=1}^r$ and unitary matrices $S_r$ of size $r \times r$ and $T_r$ of size $r \times n$. Deduce the corollary from Theorem 82.

Pre-processing $W \rightarrow W H_l$ performed with SRHT or SRFT multiplier $H_l$ involves $2n + l$ random parameters, $mn$ memory cells and $O(mn \log(l) + kn \log(n))$ flops. This computation is not superfast but is noticeably faster than Gaussian pre-processing, at least in the case of dense matrices $W$.

Corollary 83 implies that SRHT and SRFT pre-processing are still expected to output a well-conditioned matrix $W H_l$, although a little less well-conditioned and with a higher failure probability than in the case of Gaussian pre-processing.

Having the well-conditioned matrix $W H_l$ computed, we can compute CUR LRA superfast in the same way as in the previous subsection.

Although Gaussian pre-processing is slower than SHRT and SRFT pre-processing, it enables us to satisfy any of the three CUR criteria and with a higher probability while SHRT and SRFT pre-processing enables us to satisfy only the first CUR criterion and with a lower probability.
12.5 Pre-processing with matrices of abridged Hadamard and Fourier transforms

With Gaussian, SRHT or SRFT multipliers Algorithms 75 and 78 have formal support but are not superfast. Next we consider heuristic variations of pre-processing. Empirically we compute CUR LRA superfast by applying various random sparse and structured multipliers of [PZ16], in particular the matrices of abridged randomized Hadamard and Fourier transforms (ARHT and ARFT), which we revisit in Appendix B.

Actually application of Primitive and Cynical algorithms is a LRA with rudimentary pre-processing, and we proved that already in this case we compute accurate CUR LRA for factor-Gaussian inputs (with a high probability) and for the average input.

2^h \times 2^h Hadamard and Fourier matrices are generated in \( h \) recursive steps. We successively apply pre-processing with such multipliers generated in \( j \) steps for \( j = 1, 2, 3, \ldots \) until we succeed with CUR LRA. In the worst case we would arrive at fast SRHT or SRFT pre-processing, but in our tests we have consistently computed CUR LRAs superfast - by using ARHT and ARFT multipliers computed within three recursive steps.

12.6 Pre-processing with quasi Gaussian matrices

Generation, multiplication and inversion of Gaussian matrices are quite expensive. Towards simplification of these operations we consider SRHT, SRFT, ARHT and ARFT multipliers but also (i) sparse Gaussian multipliers in \( G_{\mathcal{N}Z} \) and (ii) incompletely factored Gaussian multipliers \( G \). In order to strengthen the chances for success of these heuristic recipes we can apply multipliers on both sides of the input matrix (cf. Remark 76).

Case (i). We state this recipe as heuristic, but Theorem 110 indicates some potentials for its analytic support.

Case (ii). First decompose a Gaussian matrix \( G \) into a product of factors that can be readily inverted, then delete part of them in order to decrease the cost of computing the sets of \( k \) selected rows and \( l \) selected columns of the product \( F_1 \cdots F_h W \) of the remaining factors with an input matrix \( W \). There is a variety of options based on the known matrix factorizations (cf. [GL13], [S98]). Here is an example.

**Example 84.** First compute PLU factorization of an \( n \times n \) Gaussian matrix \( G \) by applying Gaussian elimination with row pivoting, where \( P^* G = L U \) is still a Gaussian matrix. Then decompose each triangular matrix \( L \) and \( U \) into the product of \( n - 1 \) bidiagonal matrices. Finally keep some selected bidiagonal factors and drop the rest of them.

This is a purely heuristic recipe, but our next nontrival formal support for its variation may be of independent interest: we prove that the probability distribution of the product of random and randomly permuted cyclic bidiagonal matrices converges to a Gaussian matrix as the number of factors grows to the infinity.

We proceed as follows: generate random multipliers of the form \( B_i P_i \) where \( B_i \) are bidiagonal matrices with ones on the diagonal and \( \pm 1 \) on the first subdiagonal for random choice of the signs \( \pm \), \( P_i \) are random permutation matrices, \( i = 1, 2, \ldots, q \), and \( q \) is the minimal integer for which our selected superfast algorithm outputs accurate LRA of the matrix \( W G_q := W \prod_{i=1}^{q} B_i P_i \).

In both cases factorized multipliers use small storage space. For SRHT, SRFT, ARHT and ARFT multipliers it is small anyway, but Gaussian factorization allows us to store only a single bidiagonal factor at any moment, instead of storing the entire Gaussian matrix.
Empirical convergence to Gaussian distribution is quite fast (see Section 13.7). A matrix that approximates a Gaussian matrix must be dense, and using it as a multiplier would be expensive, but we can do better by keeping it factored. We hope to satisfy one, two or all three CUR criteria already for inputs pre-processed with the products of a small number of bidiagonal factors while their generation and application is still superfast.

12.7 Approximation of a Gaussian matrix by the product of random bidiagonal matrices

Suppose that \( n \) is a positive integer, \( P \) is a random permutation matrix, and define \( n \times n \) matrix

\[
B := \begin{bmatrix}
1 & \pm1 \\
\pm1 & 1 \\
\pm1 & 1 \\
\vdots & \ddots & \ddots \\
\pm1 & 1 \\
\end{bmatrix} P
\]

where each \( \pm1 \) represents an independent Bernoulli random variable.

**Theorem 85.** Let \( B_0, \ldots, B_T \) be independent random matrices of the form (54). As \( T \to \infty \), the distributions of the matrices \( G_T := \prod_{t=1}^{T} B_t \) converge to the distribution of a Gaussian matrix.

For demonstration of the approach first consider its simplified version. Let

\[
A := \begin{bmatrix}
1/2 & 1/2 \\
1/2 & 1/2 \\
\vdots & \ddots & \ddots \\
1/2 & 1/2 \\
\end{bmatrix} P
\]

where \( P \) is a random permutation matrix. We are going to prove the following theorem.

**Theorem 86.** Let \( A_0, \ldots, A_T \) be independent random matrices defined by (55). As \( T \to \infty \), the distributions of the matrices \( \Pi_T := \prod_{t=1}^{T} A_t \) converge to the distribution of the matrix

\[
\begin{bmatrix}
1/n & \cdots & \cdots & 1/n \\
\vdots & \ddots & \ddots & \vdots \\
1/n & \cdots & \cdots & 1/n \\
\end{bmatrix}.
\]

**Proof.** First examine the effect of multiplying by such a random matrix \( A_i \): let

\[
M := (m_1 | m_2 | \ldots | m_n)
\]

and let the permutation matrix of \( A_i \) defines a column permutation \( \sigma : \{1, \ldots, n\} \to \{1, \ldots, n\} \). Then

\[
MA_i = \begin{pmatrix}
\frac{m_{\sigma(1)} + m_{\sigma(2)}}{2} & \frac{m_{\sigma(2)} + m_{\sigma(3)}}{2} & \ldots & \frac{m_{\sigma(n)} + m_{\sigma(1)}}{2}
\end{pmatrix}.
\]
Here each new column is written as a linear combination of \( \{m_1, \ldots, m_n\} \). More generally, if we consider \( MA_0A_1 \cdots A_t \), i.e., \( M \) multiplied with \( t \) matrices of the form (55), each new column has the following linear expression:

\[
MA_0A_1 \cdots A_t = \left( \sum_k \pi_{k1}^t m_k \right) \left| \sum_k \pi_{k2}^t m_k \right| \ldots \left| \sum_k \pi_{kn}^t m_k \right)
\]

Here \( \pi_{ij}^t \) is the coefficient of \( m_i \) in the linear expression of the \( j \)th column of the product matrix. Represent the column permutation defined by matrix of \( A_t \) by the map

\[
\sigma_t : \{1, \ldots, n\} \rightarrow \{1, \ldots, n\}
\]

and readily verify the following lemma.

**Lemma 87.** It holds that \( \sum_j \pi_{ij}^t = 1 \) for all \( i \) and

\[
\pi_{ij}^{t+1} = \frac{1}{2}(\pi_{i\sigma_t(j)}^t + \pi_{i\sigma_t(j+1)}^t)
\]

for all pairs of \( i \) and \( j \).

Next we prove the following result.

**Lemma 88.** For any triple of \( i, j \) and \( \epsilon > 0 \),

\[
\lim_{T \rightarrow \infty} \text{Probability}(|\pi_{ij}^T - 1/n| > \epsilon) = 0.
\]

**Proof.** Fix \( i \), define

\[
F^t := \sum_j (\pi_{ij}^t - 1/n)^2.
\]

Then

\[
F^{t+1} - F^t = \sum_j (\pi_{ij}^{t+1} - 1/n)^2 - \sum_j (\pi_{ij}^t - 1/n)^2
\]

\[
= \sum_j \left[ \left( \frac{\pi_{i\sigma_t(j)}^t + \pi_{i\sigma_t(j+1)}^t}{2} \right)^2 - \left( \frac{\pi_{i\sigma_t(j)}^t - 1/n}{2} \right)^2 - \left( \frac{\pi_{i\sigma_t(j+1)}^t - 1/n}{2} \right)^2 \right]
\]

\[
\leq \sum_j \left[ \left( \frac{\pi_{i\sigma_t(j)}^t}{2} \right)^2 - \frac{1}{2} \left( \pi_{i\sigma_t(j)}^t \right)^2 \right]
\]

\[
= -\frac{1}{4} \sum_j (\pi_{i\sigma_t(j)}^t - \pi_{i\sigma_t(j+1)}^t)^2
\]

\[
\leq -\frac{1}{4n} \left( \sum_j (\pi_{i\sigma_t(j)}^t - \pi_{i\sigma_t(j+1)}^t)^2 \right)
\]

\[
= -\frac{1}{4n} (\pi_{\text{max}}^t - \pi_{\text{min}}^t)^2.
\]

Here \( \pi_{\text{max}}^t := \max_i \{\pi_{ij}^t\} \) and \( \pi_{\text{min}}^t := \min_i \{\pi_{ij}^t\} \).
Furthermore, since $\pi_{t_{\max}} \geq \pi_{t_{ij}}$, $\forall \pi_{t_{ij}}$ and $\frac{1}{n} \geq \pi_{t_{\min}} \geq 0$, it follows that

$$F^t = \sum_j (\pi_{t_{ij}} - \frac{1}{n})^2 \leq n(\pi_{t_{\max}} - \pi_{t_{\min}})^2.$$ 

Therefore

$$F^{t+1} - F^t \leq -\frac{1}{4n}(\pi_{t_{\max}} - \pi_{t_{\min}})^2 \leq -\frac{1}{4n^2}F^t.$$ 

Now our monotone decreasing sequence has the only stationary value when all values $\pi_{t_{ij}}$ coincide with each other. Together with Lemma 88 this implies

$$\lim_{T \to \infty} \text{Probability}(\left|\pi_{t_{ij}} - \frac{1}{n}\right| > \epsilon) = 0.$$ 

Next we prove Theorem 87.

Proof. Let $S^t_i$ denote the values $\pm 1$ on each row. By definition $S^t_i$ and $S^{t'}_i$ are independent for $i \neq i'$. Moreover the following lemma can be readily verified.

**Lemma 89.** $\prod_{t=1}^T S^t_i$ and $\prod_{t=1}^T S^{t'}_i$ are independent as long as there is at least one index $t$ such that $i_t \neq i'_t$.

Write

$$MB_0B_1 \cdots B_t = \left( \sum_k \gamma_{k1}^t m_k \right) \left| \sum_k \gamma_{k2}^t m_k \right| \cdots \left| \sum_k \gamma_{kn}^t m_k \right|.$$ 

and notice that each $\gamma_{ij}^t$ can be written as a sum of random values $\pm 1$ whose signs are determined by $\prod_{t=1}^T S^t_i$. Since different signs are independent, we can represent $\gamma_{ij}^t$ as the difference of two positive integers $\alpha - \beta$ whose sum is $2^T \pi_{t_{ij}}$.

Theorem 86 implies that the sequence $\pi_{t_{ij}}$ converges to $\frac{1}{n}$ almost surely as $T \to \infty$.

Therefore $\gamma_{ij}^T / 2^T$ converges to Gaussian distribution as $T \to \infty$. Together with independence of the random values $\gamma_{ij}^T$ for all pairs $i$ and $j$, this implies that eventually the entire matrix converges to a Gaussian matrix (with i.i.d. entries). 

The speed of the convergence to Gaussian distribution is determined by the speed of the convergence (i) of the values $\pi_{t_{ij}}$ to $1$ as $T \to \infty$ and (ii) of the binomial distribution with the mean $\pi_{t_{ij}}$ to the Gaussian distribution. For (i), we have the following estimate:

$$|\pi_{t_{ij}} - 1| \leq F^t \leq (1 - \frac{1}{4n^2})^{t-1} F^0;$$

and for (ii) we have the following Berry–Esseen theorem (cf. [B41]).

**Theorem 90.** Let $X_1, X_2, \ldots$ be independent random variables with $E(X_i) = 0, E(X_i^2) = \sigma_i^2 > 0$ and $E(|X_i|^3) = \rho_i < \infty$ for all $i$. Furthermore let

$$S_n := \frac{X_1 + \cdots + X_n}{\sqrt{\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_n^2}}$$ 

be a normalized $n$-th partial sum. Let $F_n$ and $\Phi$ denote the cumulative distribution functions of $S_n$ a Gaussian variable, respectively. Then for some constant $c$ and for all $n$,
\[
\sup_{x \in \mathbb{R}} |F_n(x) - \Phi(x)| \leq c\psi \quad \text{where} \quad \psi := \left( \sum_{i=1}^{n} \sigma_i^2 \right)^{-3/2} \max_{1 \leq i \leq n} \rho_i.
\]

In our case, for any fixed \(i\) and \(j\), write \(N_t = 2^t \pi^t_{ij}\) and \(\gamma^t_{ij} = X_1 + \cdots + X_{N_t}\), where \(X_i\) are i.i.d. \(\pm 1\) variables. Then \(E(X_i) = 0, E(X_i^2) = 1/4, E(X_i^3) = 1/8\), and

\[
S_{N_t} = \frac{X_1 + \cdots + X_{N_t}}{\sqrt{\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_{N_t}^2}} = \frac{\gamma^t_{ij}}{\sqrt{\frac{N_t}{4}}} = \frac{\sqrt{\pi^t_{ij} \gamma^t_{ij}}}{2^{t/2-1}}.
\]

Furthermore

\[
\sup_{x \in \mathbb{R}} |F_N(x) - \Phi(x)| \leq c \cdot \left( \sum_{i=1}^{N_t} \sigma_i^2 \right)^{-3/2} \max_{1 \leq i \leq N_t} \rho_i
\]

\[
\leq \frac{1}{8} c \cdot \left( \frac{N_t}{4} \right)^{-3/2} \to 0 \quad \text{as} \quad t \to \infty.
\]

**PART IV. TESTS, SUMMARY, HISTORY, EXTENSIONS**

### 13 Numerical Experiments

#### 13.1 Test Overview

We tested CUR LRA algorithms on random input matrices and benchmark matrices of discretized Integral and Partial Differential Equations (PDEs). We have run the tests in the Graduate Center of the City University of New York by using MATLAB, and we applied its standard normal distribution function ”randn()” in order to generate Gaussian matrices.

We calculated numerical ranks of the input matrices by using the MATLAB’s function ”rank(-,1e-6)”, which only counts singular values greater than \(10^{-6}\).

Our tables display the mean spectral norm of the relative output error over 1000 runs for every class of inputs as well as the standard deviation (std).

In Section 13.8 we present test results for the approximation of SVD-based sampling probabilities for selected matrices by the sampling probabilities for their LRAs.

#### 13.2 Four algorithms used

In our tests we compared the following four algorithms for computing CUR LRAs to input matrices \(W\) having numerical rank \(r\):

- **Tests 1 (primitive algorithm):** Randomly choose two index sets \(I\) and \(J\), both of cardinality \(r\), then compute a nucleus \(U = W_{I,J}^{-1}\) and define a CUR LRA

\[
W' := CUR = W_{:,J} \cdot W_{I,J}^{-1} \cdot W_{I,:}.
\]

- **Tests 2 (Five loops of C-A):** Randomly choose an initial row index set \(I_0\) of cardinality \(r\), then perform five loops of C-A by applying Algorithm 1 of [P00] as a subalgorithm producing \(r \times r\) CUR generators. At the end compute a nucleus \(U\) and define a CUR LRA as in Tests 1.
• **Tests 3** (Cynical algorithm, for \( k = l = 4r \)): Randomly choose a row index set \( K \) and a column index set \( L \), both of cardinality \( 4r \), and then apply Algs. 1 and 2 from [P00] to compute a \( r \times r \) submatrix \( W_{I,J} \) of \( W_{K,L} \) having locally maximal volume. Compute a nucleus and obtain a CUR LRA by applying equation (56).

• **Tests 4** (Combination of a single C-A loop with Test 3): Randomly choose a column index set \( L \) of cardinality \( 4r \); then perform a single C-A loop (of horizontal and vertical steps): First by applying Alg. 1 from [P00] find an index set \( K' \) of cardinality \( 4r \) such that \( W_{K',L} \) has locally maximal volume in \( W_{I,J} \); then by applying this algorithm to matrix \( W_{K',L} \) find an index set \( L' \) of of cardinality \( 4r \) such that \( W_{K',L'} \) has locally maximal volume in \( W_{K',L} \). Then proceed as in Tests 3 – find an \( r \times r \) submatrix \( W_{I,J} \) having locally maximal volume in \( W_{K',L'} \), compute a nucleus and define a CUR LRA.

### 13.3 CUR LRAs of random input matrices

In the tests of this subsection we computed CUR LRAs with random row- and column-selection for a perturbed \( n \times n \) factor-Gaussian matrices with expected rank \( r \), i.e., matrices \( W \) in the form

\[
W = G_1 * G_2 + 10^{-10} G_3,
\]

for three Gaussian matrices \( G_1 \) of size \( n \times r \), \( G_2 \) of size \( r \times n \), and \( G_3 \) of size \( n \times n \). Table 1 shows the test results for all four test algorithms for \( n = 256, 512, 1024 \) and \( r = 8, 16, 32 \).

| \( n \)  | \( r \) | \( \text{mean} \) | \( \text{std} \) | \( \text{mean} \) | \( \text{std} \) | \( \text{mean} \) | \( \text{std} \) | \( \text{mean} \) | \( \text{std} \) |
|------|------|------|------|------|------|------|------|------|------|
| 256  | 8    | 1.51e-05 | 1.40e-04 | 5.39e-07 | 5.31e-06 | 8.15e-06 | 6.11e-05 | 8.58e-06 | 1.12e-04 |
| 256  | 16   | 5.22e-05 | 8.49e-04 | 5.06e-07 | 1.38e-06 | 1.52e-05 | 8.86e-05 | 1.38e-05 | 7.71e-05 |
| 256  | 32   | 2.86e-05 | 3.03e-04 | 1.29e-06 | 1.30e-05 | 4.39e-05 | 3.22e-04 | 1.22e-05 | 9.30e-04 |
| 512  | 8    | 1.47e-05 | 1.36e-04 | 3.64e-06 | 8.56e-05 | 2.04e-05 | 2.77e-04 | 1.54e-05 | 7.43e-05 |
| 512  | 16   | 3.44e-05 | 3.96e-04 | 8.51e-06 | 1.92e-04 | 2.46e-05 | 1.29e-04 | 1.92e-05 | 7.14e-05 |
| 512  | 32   | 8.83e-05 | 1.41e-03 | 2.27e-06 | 1.55e-05 | 9.06e-05 | 1.06e-03 | 2.14e-05 | 3.98e-05 |
| 1024 | 8    | 3.11e-05 | 2.00e-04 | 4.21e-06 | 5.79e-05 | 3.64e-05 | 2.06e-04 | 1.94e-04 | 1.34e-03 |
| 1024 | 16   | 1.60e-04 | 3.87e-03 | 4.57e-06 | 3.55e-05 | 1.72e-04 | 3.54e-03 | 4.34e-05 | 1.11e-04 |
| 1024 | 32   | 1.72e-04 | 1.89e-03 | 3.20e-06 | 1.09e-05 | 1.78e-04 | 1.68e-03 | 1.43e-04 | 6.51e-04 |

Table 1: CUR LRA of random input matrices

### 13.4 CUR LRA of matrices of discretized Integral Equations

Table 2 displays the results of Tests 2 applied to 1,000 \( \times \) 1,000 matrices from the Singular Matrix Database of the San Jose University. (Tests 1 produced much less accurate CUR LRAs for the same input sets, and we do not display their results.) We have tested dense matrices with smaller ratios of “numerical rank/min(\( m,n \))” from the built-in test problems in Regularization Tools.\(^{11}\) The matrices came from discretization (based on Galerkin or

----

\(^{11}\)See http://www.math.sjsu.edu/singular/matrices and http://www2.imm.dtu.dk/~pch/Regutools

For more details see Chapter 4 of the Regularization Tools Manual at http://www.imm.dtu.dk/~pcha/Regutools/RTv4manual.pdf

48
quadrature methods) of the Fredholm Integral Equations of the first kind.

We applied our tests to the following six input classes from the Database:

*baart*: Fredholm Integral Equation of the first kind,
*shaw*: one-dimensional image restoration model,
*gravity*: 1-D gravity surveying model problem,
*wing*: problem with a discontinuous solution,
*foxgood*: severely ill-posed problem,
*inverse Laplace*: inverse Laplace transformation.

| Inputs | Tests 2 |
|--------|---------|
|        | m | r | mean | std   |
| baart  | 1000 | 4 | 1.69e-04 | 2.63e-06 |
|        | 1000 | 6 | 1.94e-07 | 3.57e-09 |
|        | 1000 | 8 | 2.42e-09 | 9.03e-10 |
| shaw   | 1000 | 10 | 9.75e-06 | 3.12e-07 |
|        | 1000 | 12 | 3.02e-07 | 6.84e-09 |
|        | 1000 | 14 | 5.25e-09 | 3.02e-10 |
| gravity | 1000 | 23 | 1.32e-06 | 6.47e-07 |
|        | 1000 | 25 | 3.35e-07 | 1.97e-07 |
|        | 1000 | 27 | 9.08e-08 | 5.73e-08 |
| wing   | 1000 | 2 | 9.23e-03 | 1.46e-04 |
|        | 1000 | 4 | 1.92e-06 | 8.78e-09 |
|        | 1000 | 6 | 8.24e-10 | 9.79e-11 |
| foxgood | 1000 | 8 | 2.54e-05 | 7.33e-06 |
|        | 1000 | 10 | 7.25e-06 | 1.09e-06 |
|        | 1000 | 12 | 1.57e-06 | 4.59e-07 |
| inverse Laplace | 1000 | 23 | 1.04e-06 | 2.85e-07 |
|        | 1000 | 25 | 2.40e-07 | 6.88e-08 |
|        | 1000 | 27 | 5.53e-08 | 2.00e-08 |

Table 2: CUR LRA of benchmark input matrices of discretized Integral Equations from the San Jose University Singular Matrix Database

### 13.5 Testing benchmark input matrices with bidiagonal pre-processing

Tables 3 and 4 display the results of Tests 1, 3 and 4 applied to pre-processed matrices of two kinds, both replicated from [HMT11], namely, the matrices of

(i) the discretized single-layer Laplacian operator (see Table 3) and

(ii) the approximation of the inverse of a finite-difference operator (see Table 4).

Application of Tests 1, 3, and 4 to these matrices without pre-processing tended to produce results with large errors, and so we pre-processed every input matrix by multiplying it by 20 matrices, each obtained by means of random column permutations of a random bidiagonal matrix (see Section 12.7).

Input matrices (i). We considered the Laplacian operator from [HMT11, Section 7.1]:

\[
[S\sigma](x) := c \int_{\Gamma_1} \log |x - y|\sigma(y)dy, \text{ for } x \in \Gamma_2,
\]
for two circles $\Gamma_1 := C(0,1)$ and $\Gamma_2 := C(0,2)$ on the complex plane with the center at the origin and radii 1 and 2, respectively. Its discretization defines an $n \times n$ matrix $W = (w_{i,j})_{i,j=1}^n$ for $w_{i,j} := c \int_{\Gamma_{1,j}} \log |2 \omega - y| dy$, a constant $c$ such that $||W|| = 1$, and the arc $\Gamma_{1,j}$ of the contour $\Gamma_1$ defined by the angles in $\left[\frac{2j\pi}{n}, \frac{2(j+1)\pi}{n}\right]$.

|    | Tests 1 | Tests 4 |
|----|---------|---------|
| n  | r       | mean    | std     | mean    | std     |
| 256| 31      | 1.37e-04| 2.43e-04| 9.46e-05| 2.11e-04|
| 256| 35      | 5.45e-05| 7.11e-05| 1.03e-05| 1.08e-05|
| 256| 39      | 6.18e-06| 6.32e-06| 1.24e-06| 1.72e-06|
| 512| 31      | 7.80e-05| 6.00e-05| 2.04e-05| 1.52e-05|
| 512| 35      | 1.56e-04| 1.53e-04| 6.74e-05| 1.79e-04|
| 512| 39      | 5.91e-05| 1.10e-04| 4.27e-05| 1.20e-04|
| 1024| 31     | 9.91e-05| 6.69e-05| 2.79e-05| 3.13e-05|
| 1024| 35     | 4.87e-05| 4.35e-05| 1.66e-05| 1.50e-05|
| 1024| 39     | 6.11e-05| 1.33e-04| 3.83e-06| 5.77e-06|

Table 3: CUR LRA of Laplacian input matrices

**Inputs matrices (ii).** We similarly applied our Tests 1, 3, and 4 with pre-processing to the input matrices $W$ being the inverses of large sparse matrices representing a finite-difference operator from [HMT11, Section 7.2], and we observed similar results with all structured and Gaussian multipliers.

|    | Tests 1 | Tests 3 | Tests 4 |
|----|---------|---------|---------|
| n  | r       | mean    | std     | mean    | std     | mean    | std     |
| 800 | 78      | 4.85e-03| 4.25e-03| 3.30e-03| 8.95e-03| 3.71e-05| 3.27e-05|
| 800 | 82      | 2.67e-03| 3.08e-03| 4.62e-04| 6.12e-04| 2.23e-05| 2.24e-05|
| 800 | 86      | 2.14e-03| 1.29e-03| 4.13e-04| 8.45e-04| 6.73e-05| 9.37e-05|
| 1600| 111     | 1.66e-01| 4.71e-01| 1.11e-03| 1.96e-03| 1.21e-04| 1.17e-04|
| 1600| 115     | 3.75e-03| 3.18e-03| 1.96e-03| 3.93e-03| 4.03e-05| 2.79e-05|
| 1600| 119     | 3.54e-03| 2.27e-03| 5.56e-04| 7.65e-04| 5.38e-05| 8.49e-05|
| 3200| 152     | 1.87e-03| 1.37e-03| 3.23e-03| 3.12e-03| 1.68e-04| 2.30e-04|
| 3200| 156     | 1.92e-03| 8.61e-04| 1.66e-03| 1.65e-03| 1.86e-04| 1.17e-04|
| 3200| 160     | 2.43e-03| 2.00e-03| 1.98e-03| 3.32e-03| 1.35e-04| 1.57e-04|

Table 4: CUR LRA of finite difference matrices

### 13.6 Tests with abridged randomized Hadamard and Fourier pre-processing

Table 5 displays the results of our Tests 2 for CUR LRA with abridged randomized Hadamard and Fourier pre-processing (see Appendix B for definitions). We used the same input matrices as in previous two subsections. For these input matrices Tests 1 have no longer output stable accurate LRA.
Table 5: Tests 2 for CUR LRA with ARFT/ARHT pre-processors

| Multipliers | Hadamard | Fourier |
|-------------|----------|---------|
|             | m  | n  | r | mean | std | mean | std  |
| gravity     | 1000 | 1000 | 25 | 2.72e-07 | 3.95e-08 | 2.78e-07 | 4.06e-08 |
| wing        | 1000 | 1000 | 4  | 1.22e-06 | 1.89e-08 | 1.22e-06 | 2.15e-08 |
| foxgood     | 1000 | 1000 | 10 | 4.49e-06 | 6.04e-07 | 4.50e-06 | 5.17e-07 |
| shaw        | 1000 | 1000 | 12 | 3.92e-07 | 2.88e-08 | 3.91e-07 | 2.98e-08 |
| bart        | 1000 | 1000 | 6  | 1.49e-07 | 1.37e-08 | 1.49e-07 | 1.33e-08 |
| inverse Laplace | 1000 | 1000 | 25 | 3.62e-07 | 1.00e-07 | 3.45e-07 | 8.64e-08 |
| Laplacian   | 256  | 256  | 15 | 4.08e-03 | 1.14e-03 | 3.94e-03 | 5.21e-04 |
|             | 512  | 512  | 15 | 3.77e-03 | 1.34e-03 | 4.28e-03 | 6.07e-04 |
|             | 1024 | 1024 | 15 | 3.97e-03 | 1.22e-03 | 4.09e-03 | 4.47e-04 |
| finite difference | 408 | 800  | 41 | 4.50e-03 | 1.12e-03 | 3.76e-03 | 8.36e-04 |
|             | 808  | 1600 | 59 | 4.01e-03 | 1.10e-03 | 3.87e-03 | 1.70e-03 |
|             | 1608 | 3200 | 80 | 4.60e-03 | 1.53e-03 | 3.85e-03 | 1.27e-03 |

13.7 Randomized factorization of Gaussian matrices

We have tested multiplication of 20 inverse-bidiagonal matrices (with random column permutation). Figure 6 shows the distribution of a single randomly chosen entry and the scattered plot of two such entries. The output 1024-by-1024 matrices were very close to Gaussian distribution and have passed the Kolmogorov-Smirnov test for normality in all tests repeated 1000 times.

13.8 Testing perturbation of sampling probabilities

Table 6 shows the mean and standard deviation of the norm of the relative errors of approximation of the input matrix \( W \) and of its LRA \( AB \) and similar data for the maximum difference between the SVD-based sampling probabilities of the pairs of these matrices. We also include numerical ranks of the input matrices \( W \) defined up to a tolerance of \( 10^{-6} \).

In these tests we reused input matrices \( W \) and their approximations \( AB \) from our tests in Section 13.4 (using the Singular Matrix Database of San Jose University).

In addition, the last three lines of Table 6 show similar results for diagonally scaled factor-Gaussian matrices \( GH \) with expected rank \( r \) approximating input matrices \( W \) up to small norm perturbation.

14 Accuracy and Complexity of LRA Algorithms

Next we survey the known complexity and accuracy estimates for LRA and then display them in Table 7. See more in [BW17], in particular in [BW17, Table 2.1], and [SWZ17].

14.1 A brief survey of the accuracy and complexity estimates

Classical algorithms compute an optimal LRA of an \( m \times n \) matrix given by its SVD-truncation. This deterministic computation involves \( mn \) memory cells and order of
More recent algorithms compute LRA within a factor \( f \) from optimal, based on either QRP (orthogonal) or LUP (triangular) rank-revealing factorization of a matrix \( W \), for \( f^2 \leq t_{p,r,h} \) and \( f \leq t_{p,r,h} \), respectively, \( p = \max\{m,n\} \), a real \( h \) a little exceeding 1, say \( h = 1.1 \) or \( h = 1.01 \), and

\[
t_{p,r,h} := \sqrt{(p - r)r h^2 + 1}.
\]

For both QRP and LUP approaches the computations are deterministic and involve \( O(mn \min\{m,n\}) \) flops. See [GE96], [P00], [GL13, Section 5.4.6] and our Appendix D.1.

Randomized algorithms of the survey [HMT11, Sections 10] first fix oversampling parameter \( p = l - r \) such that \( 4 \leq p \leq n - r \), then generate \( ln \) i.i.d. Gaussian variables defining an \( n \times l \) Gaussian multiplier \( H_l \) and compute the matrix \( WH_l \), its orthogonalization \( A = Q \) and the matrix \( B = Q^*W \). With a probability close to 1 this defines an LRA \( AB \) of the matrix \( W \) up to a factor \( f \) in the Frobenius norm expected to be strongly concentrated about its expected value \( \mathbb{E}(f) \leq (1 + \frac{r}{p-1})^{1/2} \) (see detailed estimates in [HMT11, Theorem 10.7]). For \( p = 4r + 1 \), say, this upper bound already decreases below 1.12. The algorithms use \( NZ_W + nl \) memory cells and \( O(l \cdot NZ_W) \) flops.

The alternative randomized algorithms of [HMT11, Section 11] generate \( 2n + l \) random parameters defining an \( n \times l \) SRFT or SRHT multiplier \( H \) and then use \( NZ_W + n \) memory cells and \( O(\min\{l \cdot NZ_W, mn \log(n)\}) \) flops for computing the matrix \( WH \); the subsequent computation of the matrices \( A = Q(H) \) and \( B = A^*W \) still involves order of \( l \cdot NZ_W \) flops. With a probability of failure of order \( O(1/r) \) this matrix is an LRA for a factor \( f = (1 + 7n/(r \log(r)))^{1/2} \) in the Frobenius norm and similarly in the spectral norm.

\[12\text{We cite the estimate for the arithmetic cost of computing SVD from [GL13, page 493].}\]
### Table 6: Tests for Perturbation of Sampling Probabilities

| Input Matrix | r  | rank | LRA Rel Error | Leverage Score Error |
|--------------|----|------|---------------|----------------------|
|              |    |      | mean          | std                  |
|              |    |      | mean          | std                  |
| baart        |  4 |  6   | 6.57e-04      | 1.17e-03             |
|              |    |      | 5.17e-05      | 5.81e-05             |
| baart        |  6 |  6   | 7.25e-07      | 9.32e-07             |
|              |    |      | 5.10e-06      | 3.32e-05             |
| baart        |  8 |  6   | 7.74e-10      | 2.05e-09             |
|              |    |      | 1.15e-06      | 3.70e-06             |
| foxgood      |  8 | 10   | 5.48e-05      | 5.70e-05             |
|              |    |      | 7.89e-03      | 7.04e-03             |
| foxgood      | 10 | 10   | 9.09e-06      | 8.45e-06             |
|              |    |      | 1.06e-02      | 6.71e-03             |
| foxgood      | 12 | 10   | 1.85e-06      | 1.68e-06             |
|              |    |      | 5.60e-03      | 3.42e-03             |
| gravity      | 23 | 25   | 3.27e-06      | 1.82e-06             |
|              |    |      | 4.02e-04      | 3.30e-04             |
| gravity      | 25 | 25   | 8.69e-07      | 7.03e-07             |
|              |    |      | 4.49e-04      | 3.24e-04             |
| gravity      | 27 | 25   | 2.59e-07      | 2.88e-07             |
|              |    |      | 4.64e-04      | 3.61e-04             |
| laplace      | 23 | 25   | 2.45e-05      | 9.40e-05             |
|              |    |      | 4.85e-04      | 3.03e-04             |
| laplace      | 25 | 25   | 3.73e-06      | 1.30e-05             |
|              |    |      | 4.47e-04      | 2.78e-04             |
| laplace      | 27 | 25   | 1.30e-06      | 4.67e-06             |
|              |    |      | 3.57e-04      | 2.24e-04             |
| shaw         | 10 | 12   | 6.40e-05      | 1.16e-04             |
|              |    |      | 2.80e-04      | 5.17e-04             |
| shaw         | 12 | 12   | 1.61e-06      | 1.60e-06             |
|              |    |      | 2.10e-04      | 2.70e-04             |
| shaw         | 14 | 12   | 4.11e-08      | 1.00e-07             |
|              |    |      | 9.24e-05      | 2.01e-04             |
| wing         |  2 |  4   | 1.99e-02      | 3.25e-02             |
|              |    |      | 5.17e-05      | 2.07e-04             |
| wing         |  4 |  4   | 7.75e-06      | 1.59e-05             |
|              |    |      | 7.17e-06      | 2.30e-05             |
| wing         |  6 |  4   | 2.57e-09      | 1.15e-08             |
|              |    |      | 9.84e-06      | 5.52e-05             |
| Factor-Gaussian | 25 | 25 | 1.61e-05 | 3.19e-05 |
|              |    |      | 4.05e-08      | 8.34e-08             |
| Factor-Gaussian | 50 | 50 | 2.29e-05 | 7.56e-05 |
|              |    |      | 2.88e-08      | 6.82e-08             |
| Factor-Gaussian | 75 | 75 | 4.55e-05 | 1.90e-04 |
|              |    |      | 1.97e-08      | 2.67e-08             |

**Remark 91.** Our modification in Remark 80 simplifies the transition from the factor $WH$ to an LRA: we involve just order of $(kn + lm)$ memory cells and $O(kn + lm \log(l))$ flops at that stage and (more importantly) yield optimal error bound up to a constant factor $f$.

The Power Scheme variation of the above algorithms replaces the product $WH$ by $(W^*W)^qWH$ for a fixed positive integer $q$. The computation of this product involves by $2q + 1$ times more flops than the original computation of the matrix $WH$ but is expected to output an LRA with optimal spectral error norm $\sigma_{r+1}$ up to a factor

$$f \leq \left(1 + \sqrt{\frac{r}{p-1}} + \frac{e \sqrt{r+p}}{p} \sqrt{n-r}\right)^{1/(2q+1)}.$$  

As $q$ increases, this bound is expected to converge very fast to the optimal value 1 and to nearly reach this value for $q$ of order $\log(\min\{m, n\})$ (see [HMT11, Section 10.4]). Recall, however, that already with an $n \times (n + p)$ Gaussian multiplier for a reasonably large ratio $(p + 1)/r$ we decrease the expected value of the error factor $f$ to nearly 1.

The C-A algorithm of [GOSTZ10] computes CUR LRA of an $m \times n$ matrix of numerical rank $r$ within a factor of $\sqrt{mn}$ from optimal by using $(m + n)r u$ memory cells and $O(r^3) + (m + n - 2r)ru$ flops where the number $u$ of C-A steps cannot exceed $\binom{m}{r} \binom{n}{r}$ but empirically is a small constant and is at most 2 on the average input (see Section 6).

Computation with sampling probabilities in [FKW98], [DK03], [FKW04], [DKM06] and
[RV07] enabled CUR LRA within probabilistic error bounds of the form

$$||W - CUR||_F^2 \leq \tilde{\sigma}_{r+1}^2 + \epsilon||W||_F^2$$

for \(\min\{m, n\} \leq p(r/\epsilon)\),

involving \(k = k(r, 1/\epsilon, 1/\delta)\) rows and \(l = l(r, 1/\epsilon, 1/\delta)\) columns of an input matrix for some polynomials \(k(r, x, y)\) and \(l(r, x, y)\) of low degrees and for a failure probability \(\delta\). This restricted the tolerance \(\epsilon\) in (46) from below because \(k \leq m\) and \(l \leq n\).

Then much stronger bound (46) was proved in the paper [DMM08], which used \(mn\) memory cells and \(O(mnk + k^3)\) flops and required to involve \(l \geq cr\log(r)/\epsilon^2\) columns and \(k \geq cl\log(l)/\epsilon^2\) rows of an input matrix \(W\), respectively, for unspecified constant \(c\). This is inferior to the values \(k\) and \(l\) in [HMT11], but in its tests the paper [DMM08] reached LRAs for \(\epsilon = 0.1\) and moderate heuristic values of \(k\) and \(l\).

Subsequent randomized algorithms yielded bound (46) under smaller asymptotic estimates for these parameters. In particular the randomized algorithms of [BW14] and [BW17] compute matrices \(C\) and \(R\) made up of \(O(r/\epsilon)\) columns and \(O(r/\epsilon)\) rows of \(W\) by using \(O(NZ_W \log(n)) + m \text{ poly}(r, n, 1/\epsilon)\) flops for \(m \geq n\) and for \(\text{poly}(v)\) denoting a polynomial in \(v\). By using \(O(mn^3r/\epsilon)\) flops overall the authors make their CUR LRA algorithms deterministic. Deterministic algorithms of [GS12] and [BDM-I 14] use a little fewer flops for computing LRAs, and we can extend them to the computation of CUR LRAs by applying our simple superfast Algorithms 10 and either 24 or 24a.

Randomized algorithms of [SWZ17] involve \(NZ_W + (m+n)\text{poly}(r)\) flops, use matrices \(C\) with \(O(r \log(r))\) columns and \(R\) with \(O(r \log(r))\) rows and achieve optimal output accuracy up to a factor of \(f = \log(n)\text{poly}(r)\) in norm (7). In the case of a constant \(r\) the authors decrease the approximation error norm factor \(f\) to a constant by using \(\text{poly}(m + n)\) flops.

The cited algorithms (except for [GOSTZ10] and stages 2 and 3 of Alg. 78) use at least \(NZ_W\) memory cells and are not superfast for the worst case input. If, however, we apply them to a \(q \times s\) sketch and replace \(m\) by \(q \ll m\) and \(n\) by \(s \ll n\) in their estimates, then the algorithms become superfast.

We proved that they remain accurate in the cases of factor-Gaussian inputs (with a high probability) and the average inputs allowing close LRA.

Likewise randomized Algorithm 78 becomes superfast also at its stage 1 (not covered in Table 7) if we apply it by computing superfast quasi Gaussian, Abridged Hadamard or Abridged Fourier multipliers instead of Gaussian; then we lose our formal estimates for the output accuracy for the worst case input but they hold for the average input.

14.2 A display of the accuracy and complexity estimates

Table 7 displays the relevant data for the listed LRA algorithms applied to \(m \times n\) matrices of numerical rank \(r\) where we assume that \(m \geq n\); otherwise we could work with the transposed input matrix.

The column of the table marked by \(f\) shows the known upper bounds on the ratio of the Frobenius error norm of the output with the minimal error norm \(\tilde{\sigma}_{r+1}\), except that the ratio is expressed in \(l_1\)-norm \(||\cdot||_{l_1}\) in the line of the paper [SWZ17].

The column marked by \(k\) shows upper bound on the maximal numbers of rows or columns of the computed CUR generator.

The column marked “cells” displays the estimated number of memory cells used (up to smaller order terms).
The lines of the papers [GL13], [HMT11], [GS12] and [BDM-I 14] cover computation of LRAs that are not CUR LRAs. Then again, however, our Algorithms 10 and 24 or 24a enable simple superfast transition from any LRA to a CUR LRA.

The line for the paper [HMT11] displays data for the expected value of the norm $\|E\|_F$ (see probabilistic bounds on that norm in [HMT11, Section 10.3]).

The line for Alg. 78 displays the data for its stages 2 and 3.

The six papers of the last seven lines of the table came from the Computer Science Community. The algorithms and techniques of these papers are advanced and important, but the overhead constants hidden in the notation “O” and “poly” and an upper bound on the positive scalar $\epsilon$ involved in the algorithms are not explicitly specified so far (see Section 11.5), although the overhead was reasonable in the tests reported in [DMM08]. We also use “O” notation in the line of the table for our Alg. 78, but in that case the overhead constants are readily defined by the estimates at the end of Section 12.3.

Let us list some assumptions, auxiliary bounds and notations.

- $t_{p,g}^2 = (p-r)rg^2 + 1$ (cf. (57)).
- The factor $\|E\|_F/\|E\|_C$ in the line of the paper [GOSTZ10] is bounded according to (4) and (7) as follows:
  \[
  \|E\|_C \leq \|E\|_F \leq \sqrt{mn} \|E\|_C \text{ and } \|E\|_F \leq \|E\|_{l_1} \leq \|E\|_F \sqrt{mn}.
  \]
- $b$ in the line of the paper [GOSTZ10] is a positive scalar of our choice; by increasing it we can decrease the ratio $f$.
- ITER denotes the number of iterations in the algorithm of [GOSTZ10], empirically it is a constant and is equal to 2 on the average input.
- We can obtain the upper bound 1.12 on the factor $f$ in the line of the paper [HMT11] by choosing the oversampling parameter $p := 4r + 1$: indeed $f \leq (1 + \frac{r}{p-1})^{1/2} = \frac{\sqrt{5}}{2} < 1.12$. Notice that $f \geq 1$, $0 \leq p \leq n - r$ and $f \to 1$ as $p \to \infty$.
- $NZ_W$ denotes the number of nonzero entries of an input matrix $W$.
- $\lg$ stands for log.
- $\epsilon \leq 1$ is a positive parameter of our choice limited by the bound $k \leq m$.
- poly($\lg(n), \frac{1}{\epsilon}$), poly($m$) and poly($r$) denote some polynomials in the three variables $\lg(n)$, $r$ and $1/\epsilon$, in $m$ and in $r$, respectively.

15 Summary, Research Directions and Related Work

15.1 Superfast LRAs: narrow classes of hard inputs and superfast computation of LRAs of random and the average inputs

We first recalled the LRA problem and then specified a family of $2mn$ matrices of size $m \times n$ (we call them $\pm \delta$ matrices) for which no accurate LRA can be computed unless the algorithm accesses all $N$ entries of the input matrix. In that case, however, the algorithm involves at least $N$ memory cells and at least $N/2$ flops.
Table 7: Accuracy and complexity of some LRA algorithms (proven estimates for $m \geq n$)

| Source      | $f$   | $k$   | cells  | flops                                      |
|------------|-------|-------|--------|--------------------------------------------|
| [GL13]     | $1$   | $r$   | $mn$   | $O(m^2 n)$                                 |
| [GE96]     | $t_{m,r,h}$ | $r$   | $mn$   | $O(mn^2)$                                 |
| [P00]      | $t_{m,r,h}$ | $r$   | $mn$   | $O(mn^2)$                                 |
| [GOSTZ10]  | $(b+1) \frac{1}{\|E\|_F} \frac{1}{\|E\|_G}$ | $(b+1)r - 1$ | $rm \cdot \text{ITER}$ | $O(r^3) + (m - r)r \cdot \text{ITER}$ |
| [HMT11]    | $(1 + \frac{r}{k+r})^{1/2}$ | any $k > r + 1$ | $(m + k)n$ | $16k \cdot NZW + O(r^2m)$                  |
| Alg. 78    | $O(1)$ | $O(r)$ | $km$   | $O(mk)$                                   |
| [DMM08]    | $1 + \epsilon$ | $O(k \log^2(k)/\epsilon^4)$ | $mn$ | $O(mnk)$                                  |
| [MD09]     | $1 + \epsilon$ | $O(r \log(r)/\epsilon^2)$ | $mn$ | $O(mnk)$                                  |
| [BDM-I14]  | $2 + \epsilon$ | $O(r/\epsilon)$ | $mn$ | $O(mn^3)$                                 |
| [BW17]     | $1 + \epsilon$ | $O(r/\epsilon)$ | $mn$ | $O(NZW \log(n) + m \cdot \text{poly}(\log(n,r,1/\epsilon))$ |
| [BW17]     | $1 + \epsilon$ | $O(r/\epsilon)$ | $mn$ | $O(mn^2r/\epsilon)$                      |
| [SWZ17]    | $O(\log(n) \log(r))$ | $O(r \log(r))$ | $mn$ | $NWZ + m \cdot \text{poly}(r))$           |
| [SWZ17]    | $r \cdot \text{poly}(\log r)$ | $mn$ | $n^{O(r^2 \log^2 r)} \cdot \text{poly}(m)$ |

Computational time and memory space used by various fast LRA algorithms nearly match these lower bounds, but cross-approximation (C-A) superfast algorithms use much fewer flops and memory cells. Although their output LRAs cannot be accurate for all inputs and not even for all $\pm \delta$-matrices, these LRAs are routinely accurate in practice. We provided formal support, so far missing, for this empirical observation.

Namely we proved that both for the average input matrix allowing LRA and with a high probability for small norm perturbations of factor-Gaussian inputs this algorithm and even more primitive superfast LRA algorithms compute accurate LRAs. We proved this by applying two distinct techniques of analysis based on two distinct sufficient criteria for reaching close LRAs of matrices that allow LRAs.

The algorithms output transparent CUR LRAs defined by a CUR generator, which is a submatrix of an input matrix that has a much smaller size. Given an LRA, we presented superfast algorithms for the computation of a CUR generator and therefore a CUR LRA, as well as for a posteriori error estimation.

15.2 Superfast LRA computation based on random pre-processing

Besides providing the missing formal support for an important empirical observation, we proposed some new tools for devising efficient CUR LRA algorithms. We proved that C-A iterations and various other superfast algorithms (such as C-A iterations) compute accurate CUR LRA to the average input matrix allowing LRA and with a high probability to perturbed factor-Gaussian inputs.

This does not necessarily apply to real world inputs because they are not random. We, however, randomized input matrices by means of multiplicative pre-processing and proved that with a high probability C-A and other superfast algorithms output an accurate CUR LRA of any matrix allowing LRA and pre-processed with a Gaussian, SRFT or SRHT multiplier.

This pre-processing is fast and cannot be superfast for the worst case input, but empirically our superfast randomized pre-processing stage with sparse multipliers turned out
to be as efficient as with Gaussian. Moreover we can apply our superfast pre-processing recursively. In the worst case it turns into fast SRHT or SRFT pre-processing in $\log_2(n)$ steps, for which we have formal support, but in our tests a few superfast recursive steps were always sufficient in order to compute a close CUR LRA.

Likewise we recursively factored a Gaussian multiplier into the product of random bidiagonal and permutation matrices, which enabled memory efficient computations. We non-trivially proved that the factorization converges, and in our tests it converged very fast.

### 15.3 Superfast LRA based on using sampling probabilities

The algorithms devised in the research community of Numerical Linear Algebra (NLA) compute CUR generators satisfying one or both of two sufficient criteria for producing accurate CUR LRA: it is sufficient if a generator shares numerical rank with the input matrix or has maximal volume among all submatrices of the same size.

In Section 11 we studied the CUR LRA algorithm of [DMM08], devised in the research community of Computer Science (CS) and based on the third sufficient criterion for computing accurate (and even nearly optimal) CUR LRA: a CUR generator should be defined by sufficiently many rows and columns of an input matrix sampled according to properly pre-computed probabilities.

Unless we sample too many rows and columns, the algorithm is superfast except for its fast initial stage of computing sampling probabilities. We made it superfast by dropping this stage and choosing essentially uniform sampling probabilities. Then we proved that the output CUR LRA is still accurate for $m \times n$ inputs obtained by perturbation of factor-Gaussian matrices and for the average input matrix of numerical rank $r$ such that $r \ll \min\{m, n\}$.

Moreover if the input matrix is given with its crude but reasonably close LRA, say, supplied by a superfast algorithm from the NLA community, then we can compute sampling probabilities superfast, perform the algorithm of [DMM08] superfast by using these sampling probabilities, and again arrive at an accurate LRA, thus demonstrating benefits of synergistic alliance of the research in the NLA and CS communities.

### 15.4 Our technical novelties, some impacts, applications and extensions

Our study provided some new insights into LRA, demonstrated the benefits of synergistic alliance between researchers in the CS and NLA communities and should embolden wider application of our successful LRA techniques such as simplified heuristics, the average case analysis, sketching methods, randomized sparse and structured pre-processing and recursive algorithms of C-A type.

Our progress should be interesting for a wide range of computations involving LRA, but our superfast CUR LRA algorithms enable new surprising applications, in particular (see Section 16) to dramatic acceleration of the Conjugate Gradient (CG) algorithms and the construction of low rank generators for the Fast Multipole Method, which is frequently its bottleneck.

We achieved our progress in a hot area of intensive research by applying relatively simple means. This is partly due to common distrust to the average case complexity, which indeed can be quite distinct from the complexity of many realistic input classes. Nevertheless our study covers a large class of real world inputs, as our tests testify. Moreover our results enable superfast accurate computation of CUR LRA of any input allowing close LRA and
pre-processed with random multipliers. We also describe superfast refinement of any matrix allowing its LRA and given together with a crude but reasonably close initial LRA.

Besides formally supporting empirical accuracy of superfast algorithms, enhancing it with randomized pre-processing and providing new insights into the subject, we revealed and handled its delicate hidden technicalities and introduced some concepts and methods of independent interest. Here are some examples:

1. We specified hard inputs for superfast LRA algorithms based on $\pm\delta$-matrices; they can be useful in the studies of other matrix computations as well.

2. The concept of the average matrix having a fixed small numerical rank may not be intuitively obvious, but we proposed its natural definitions in Section 2.2 by introducing and exploring the auxiliary concepts of factor-Gaussian matrices and (in Section 11.1) $G_{n,r}$-sampling probabilities, both of independent interest. We also defined the average sparse inputs allowing LRA.

3. We formally supported the observed power of superfast LRA algorithms by studying them for random and average inputs – both in depth and from various angles – by using error analysis, volume maximization, and sampling probabilities.

4. We estimated the volume of a matrix in terms of its local volume, estimated the volume of matrix products in terms of the volumes of the factors and linked $r$-projective volume of a CUR generator to the volume of an associated full rank submatrix.

5. We computed an LRA with randomized pre-processing superfast except for the fast pre-processing stage.

6. Motivated by our proofs of superfast CUR LRA for factor-Gaussian inputs, we proposed random multiplicative pre-processing and proved that with a high probability C-A and some other superfast algorithms compute close CUR LRAs when they are applied to any matrix allowing its close LRA and pre-processed by using Gaussian, SRHT or SRFT multipliers.

7. We empirically accelerated fast pre-processing for LRA to superfast level by applying quasi Gaussian, Abridged Hadamard and Abridged Fourier random multipliers.

8. We proposed memory efficient randomized factorization of a Gaussian multiplier into the product of random bidiagonal and permutation matrices, nontrivially proved that the product converges to a Gaussian matrix as the number of multipliers grow large and empirically verified that the convergence is fast.

9. We proposed a superfast extension of any LRA to CUR LRA.

10. We improved a decade-old estimate for the norm of the inverse of a Gaussian matrix and extended it to the case of a sparse Gaussian matrix.

11. Our simple but novel link to LRA via FMM enables dramatic acceleration of the Conjugate Gradient algorithms and of the construction of low rank generators for the Fast Multipole Method (FMM), which is frequently the FMM’s bottleneck.
15.5 Superfast algorithms for CUR LRA and related subjects

The recipes for superfast heuristic computation of a CUR LRA in this subsection should be combined with repeated verification of the output accuracy by means of superfast a posteriori error estimation. Let us list some superfast algorithms that empirically output accurate CUR LRAs or are involved in their computations.

1. Primitive algorithm,
2. Cynical algorithm,
3. C-A iterations,
4. The algorithm of [DMM08] if its original fast computation of SVD-based sampling probabilities is dropped or accelerated to superfast level.
5. Pre-processing with sparse and structured multipliers such as quasi Gaussian, Abridged Hadamard and Abridged Fourier multipliers.
6. A posteriori estimation of the errors of LRA algorithms.
7. Transition from any close LRA to CUR LRA.
8. Calculation of close SVD-based sampling probabilities for the algorithm of [DMM08] provided that a crude but sufficiently close LRA of the input matrix is available.

If a reasonably close CUR LRA has been computed but must be refined, one can apply the following steps towards fulfilling the third CUR criterion.

**A superfast procedure for the refinement of an LRA.**

1. Fix two initial values $k_0$ and $l_0$ for the number of samples of rows and columns. (They can be much smaller than the bounds on $k$ and $l$ in Theorem 61.)
2. Approximate SVD-based sampling probabilities for the algorithms of [DMM08]; perform this stage superfast by using a crude initial CUR LRA.
3. Apply the algorithms of [DMM08] by sampling $k_0$ rows and $l_0$ columns according to pre-computed sampling probabilities.
4. If this does not work, recursively reapply the algorithm by increasing (say, doubling) the numbers of sampled rows and columns.

One can repeat this procedure if the output LRA is reasonably accurate but still must be improved.

**The second superfast procedure for CUR LRA.**

Alternatively one can modify the latter procedure by dropping its stage 2 and instead applying pre-processing with quasi-Gaussian or other random multipliers and then performing stages 3 and 4 with uniformly distributed sampling probabilities. This variant assumes no initial LRA and is superfast if its pre-processing stage is superfast.
15.6 Further research directions

How much can one extend our analysis where (i) numerical rank of the input matrix is not known or (ii) a gap between a pair of its consecutive singular values is not dramatic?

Another important challenge is further study of the algorithms of [DMM08], revisited in our Section 11. We have already adjusted them to superfast computation of accurate CUR LRA of the average matrix allowing close LRA or a matrix given with its crude LRA, but can we achieve this by sampling fewer rows and columns (see Section 11.5)?

Can we achieve such progress by means of incorporating some techniques of [BW17], [SWZ17] and [HMT11]? Can we achieve LRA in terms of $l_1$-norm superfast for perturbed factor-Gaussian input (with a high probability) or the average input allowing LRA in terms of $l_1$-norm? Can we refine such an LRA superfast?

Other natural research challenges include

1. design, analysis and experimental study of new multipliers for pre-processing LRA,
2. the study of the benefits of the size variation of candidate CUR generators, say, of their alternate expansion and compression,
3. further formal and experimental study of C-A techniques and their extension to tensor decomposition (cf. Remarks 29 and 77),
4. extension of the application area of fast LRA by using superfast LRA,
5. investigation of the links among three sufficient CUR criteria and
6. synergistic combinations of LRA algorithms originated in the communities of NLA and CS towards enhancing the power of both groups of algorithms.

15.7 Related Works

We refer the reader to [HMT11], [M11], [W14], [CBSW14], [O16], [KS16], [BW17], [SWZ17] and the references therein for the extensive bibliography for the study of LRA and CUR approximation (also see our Section 14). Next we very briefly comment on the steps of that study that are most relevant to our present work.

The study of CUR (aka CGR and pseudo-skeleton approximation) can be traced back to the skeleton decomposition in [G59] and QRP factorization in [G65] and [BG65], refined and redefined as rank-revealing factorization in [C87].

Further work in [CH90], [CH92], [HP92], [HLY92], [CI94], [GE96], and [P00] largely relied on the maximization of the volumes of CUR generators. This fundamental idea goes back to [K85] and has been developed in [GZT95], [T96], [GTZ97], [GTZ97a], [B00], [GT01], [BR03], [GOSTZ10], [GT11], [B11], [M14], and [O16] (recall some distinct nontrivial algorithmic applications of matrix volume in [B-I92].) The study in [GZT95], [T96], [GTZ97], [GTZ97a] towards volume maximization revealed a simple but crucial property that computation of a CUR generator requires no factorization of the input matrix but just proper selection of its row and column sets.

$^{13}$Here are some relevant dates for these papers: [CH90] was submitted on 15 December 1986, accepted on 01 February 1989; [CH92] was submitted on 14 May 1990, accepted on 10 April 1991; [HP92] was submitted on December 1, 1990, revised on February 8, 1991; [HLY92] was submitted on October 8, 1990, accepted on July 9, 1991, and [CI94] appeared as Research Report YALEU/DCS/RR-880, December 1991.
C-A iteration was a natural extension of this observation preceded by the Alternating Least Squares method of [CC70] and [H70] and leading to dramatic empirical decrease of quadratic and cubic memory space and arithmetic time of LRA algorithms, respectively. The concept of C-A was coined in [T00], and we credit [B00], [BR03], [MMD08], [MD09], [GOSTZ10], [OT10] and [B11], and [KV16] for devising some efficient C-A algorithms.

Important sketching techniques can be traced back to [S06] and were surveyed in details in [W14].

The random sampling approach to LRA surveyed in [HMT11] and [M11] can be traced back to the breakthrough of applying pre-computed sampling probabilities for random sampling of the rows and columns of an input matrix in [FKW98] and [DK03]. See also early works [FKW04], [S06], [DKM06] and [MRT06] and landmarks of further research progress in the papers [DMM08], [BW14], [BW17] and [SWZ17], which also survey the related work.

Our paper extends our study in the papers [PZ16], [PZ17], and [PZ17a], devoted to
(i) efficiency of heuristic sparse and structured multipliers for LRA (see [PZ16]),
(ii) approximation of trailing singular spaces associated with the \(\nu\) smallest singular values of a matrix having numerical nullity \(\nu\) (see [PZ17]), and
(iii) using random multipliers instead of pivoting in Gaussian elimination (see [PZ17a]).

Presently, however, we greatly supersede these papers in terms of the scale of the acceleration of the known formally supported algorithms.

16 Some Implications and Extensions

Our study can be extended to numerous important computational problems linked to LRA. We refer the reader to the long list of the known applications of fast LRA in [HMT11], [M11], [W14], [CBSW14], [BW17], [KS16], [SWZ17], and the bibliography therein, but more applications should be inspired by our study. For example, computation of Tucker’s and TT decompositions of a tensor can be reduced to the computation of LRA of some associated matrices [MMD08], [OT10], [VL16], but one can also directly extend our analysis of the average case behavior of superfast algorithms (see [LPSZZa]) and explore the variation of cross-approximation techniques pointed out in a footnote in Section 7.3.

Superfast LRA algorithms should have a large area of important applications not open for fast LRA. In this section we demonstrate such new opportunities by applying superfast LRA algorithms to achieve dramatic acceleration of the bottleneck stage of the construction of low rank generators for the Fast Multipole Method (FMM) with further surprising extension to the Conjugate Gradient algorithms,\(^1\) both FMM and CG being among the most celebrated achievements of the 20th century in Numerical Linear Algebra (see [C00] and [BY13] for FMM and [GL13, Section 11.3] for the CG algorithms).

16.1 Acceleration of the Fast Multipole Method (FMM)

In this subsection we extend our superfast LRA and CUR approximation to the acceleration of a basic construction stage of the FMM, that is, multiplication by a vector of an HSS matrix\(^2\) defined by its average case generators. Although FMM is justly celebrated for its fast and memory efficient performance, it is not always emphasized that its performance

---

\(^1\)Hereafter we use the acronyms FMM and CG.

\(^2\)Here and hereafter “HSS” stands for “hierarchically semiseparable”.

61
is frequently kept down at its bottleneck stage of the construction of low rank generators. Application of superfast LRA should fix this deficiency.

We first recall that HSS matrices naturally extend the class of banded matrices and their inverses, are closely linked to FMM, and are highly and increasingly popular (see [BGH03], [GH03], [MRT05], [CGS07], [VVGM05], [VVM07/08], [B10], [X12], [XXG12], [EGH13], [X13], [XXCB14], and the bibliography therein).

**Definition 92.** (See [MRT05].) With each diagonal block of a block matrix associate its complement in its block column, that is, the union of the pair of the maximal sub- and super-diagonal blocks in that block column, and call this complement a neutered block column.

**Definition 93.** (See [CGS07], [X12], [X13], [XXCB14].) A block matrix $M$ of size $m \times n$ is called a $r$-HSS matrix, for a positive integer $r$,

(i) if all diagonal blocks of this matrix consist of $O((m+n)r)$ entries overall and

(ii) if $r$ is the maximal rank of its neutered block columns.

**Remark 94.** Many authors work with $(l,u)$-HSS (rather than $r$-HSS) matrices $M$ for which $l$ and $u$ are the maximal ranks of the sub- and super-diagonal blocks, respectively. The $(l,u)$-HSS and $r$-HSS matrices are closely related. If a neutered block column $N$ is the union of a sub-diagonal block $B_-$ and a super-diagonal block $B_+$, then $\text{rank}(N) \leq \text{rank}(B_-) + \text{rank}(B_+)$, and so an $(l,u)$-HSS matrix is a $r$-HSS matrix, for $r \leq l + u$, while clearly a $r$-HSS matrix is an $(r,r)$-HSS matrix.

The FMM exploits the $r$-HSS structure of a matrix as follows:

(i) Cover all off-block-diagonal entries with a set of non-overlapping neutered block columns.

(ii) Express every neutered block column $N$ of this set as the product $FH$ of two generator matrices, $F$ of size $h \times r$ and $H$ of size $r \times k$. Call the pair $\{F,H\}$ a length $r$ generator of the neutered block column $N$.

(iii) Multiply readily the matrix $M$ by a vector by separately multiplying generators and diagonal blocks by subvectors, involving $O((m+n)r)$ flops overall, and

(iv) in a more advanced application of FMM solve a nonsingular $r$-HSS linear system of $n$ equations by using $O(nr \log^2(n))$ flops under some mild additional assumptions on the input.

This approach is readily extended to the same operations with $(r,\xi)$-HSS matrices, that is, matrices approximated by $r$-HSS matrices within a perturbation norm bound $\xi$ where a positive tolerance $\xi$ is small in context (for example, is the unit round-off). Likewise, one defines an $(r,\xi)$-HSS representation and $(r,\xi)$-generators.

$(r,\xi)$-HSS matrices (for $r$ small in context) appear routinely in matrix computations, and computations with such matrices are performed efficiently by using the above techniques.

In some applications of the FMM (see [BGP05], [VVVF10]) stage (ii) is omitted because short generators for all neutered block columns are readily available, but this is not the case in a variety of other important applications (see [XXG12], [XXCB14], [P15], and the next subsection). This stage of the computation of generators is precisely the LRA of the neutered block columns, which turns out to be the bottleneck stage of FMM in these applications, and our work provides a remedy.

Indeed apply a fast algorithm at this stage, e.g., the algorithm of [HMT11] with a Gaussian multiplier. Multiplication of a $q \times h$ matrix by an $h \times r$ Gaussian matrix requires
(2h − 1)qr flops, while standard HSS-representation of an \( n \times n \) HSS matrix includes \( q \times h \) neutered block columns for \( q \approx m/2 \) and \( h \approx n/2 \). In this case the cost of computing an \( r \)-HSS representation of the matrix \( M \) is at least of order \( mnr \). For large integers \( m \) and \( n \), this greatly exceeds the above estimate of \( O((m+n)r^2) \) flops at the other stages of the computations.

Alternative customary techniques for LRA rely on computing SVD or rank-revealing factorization of an input matrix and are at least as costly as the computations by means of random sampling.

Can we alleviate such a problem? Yes, for the average input we can compute LRAs to \((r, \xi)\)-generators much faster by applying superfast CUR LRA algorithms studied in this paper.

### 16.2 Acceleration of the Conjugate Gradient (CG) algorithms

We recall that a real \( n \times n \) matrix \( M \) and a linear system of \( n \) equations \( Mx = b \) are said to be symmetric positive definite\(^{16}\) if \( M = V^T V \) for a nonsingular matrix \( V \) [GL13], and now we introduce the following concept:

**Definition 95.** An \( n \times n \) matrix \( M \) is \((r, \xi)\)-concentrated if the set of its singular values is clustered (within a small positive tolerance \( \xi \)) about at most \( r + 1 \) values. Such a matrix is strongly \((r, \xi)\)-concentrated if this set contains a cluster of at least \( n - r \) singular values, each counted with its multiplicity.

The following two facts show efficiency of the CG algorithms:

1. Given an spd linear system of equations \( Mx = b \) whose matrix \( M \) is \((r, \xi)\)-concentrated, the CG algorithms converge to its solution within an error norm in \( O(\xi) \) in at most \( r \) iterations [A94], [BBC93], [G97], [B02], [S03].

2. Various highly important present day computations routinely involve matrices made strongly \((r, \xi)\)-concentrated and hence \((r, \xi)\)-concentrated, for reasonably small integers \( r \) and small positive \( \xi \), by means of various efficient preconditioning techniques.

The next critical issue is whether we can decrease the computational cost of a CG iteration, which is reduced essentially to computation or approximation of the product of the matrix \( M \) by a vector or by a few vectors.

Next we prove that strongly \((r, \xi)\)-concentrated matrices are also \((r, \xi)\)-HSS matrices, and so, by applying our accelerated variant of FMM, we can approximate the product of such an \( n \times n \) average matrix \( M \) by a vector significantly faster than by applying the known algorithms, which involve \((2n-1)n\) flops.

**Theorem 96.** If an \( n \times n \) spd matrix \( M \) is strongly \((r, \xi)\)-concentrated for a small positive tolerance \( \xi \), then the numerical rank of any of its off-diagonal submatrix is at most \( r \).

**Proof.** Since \( M \) is a strongly \((r, \xi)\)-concentrated matrix, at least \( n - r \) its singular values are clustered about a certain value \( s \). Consequently at least \( n - r \) singular values of the matrix \( \hat{M} = M - sI \) are clustered about 0, that is, \( \text{rrank}(\hat{M}) \leq r \). Hence \( r \) is an upper bound on the numerical rank of any submatrix and in particular of any neutered block column \( \hat{M}_{K,L} \) of the matrix \( \hat{M} \). Therefore it is also an upper bound on the numerical rank of any neutered block column \( M_{K,L} \) of the matrix \( M \) because all neutered block columns of the matrix \( sI \) vanish and thus have rank and numerical rank 0. □
Corollary 97. If an $n \times n$ spd matrix $M$ is strongly $r$-concentrated, then we can approximate the solution of a linear system of $n$ equations $Mx = b$ by using $O(r^2 n \log(n)) + \gamma(M)$ flops provided that one can compute generators of length at most $r$ for a $r$-HSS approximate representation of the matrix $M$ by using $\gamma(M)$ flops.

By virtue of Theorem 96 the neutered block columns in the $r$-HSS representation of the matrix $M$ have numerical ranks at most $r$. By virtue of the results of the previous subsection, $\gamma(M) = O(rn)$ in the case of the average matrix $M$, and so in this case we can accelerate the CG algorithms accordingly – by a factor of $n/r$.

Remark 98. In numerical implementation of this approach one should apply advanced numerically stable variants of CG algorithms (see [KL07], [BDK15]) in order to offset the output errors of the superfast LRA algorithms.

Remark 99. Extension to nonsymmetric inputs. We can extend our results to a nonsymmetric nonsingular linear system of equations $Ay = f$ by means of symmetrization of a matrix $A$ in any of the two ways, $A \rightarrow M = A^T A$ or $A \rightarrow M = AA^T$, and then apply CG algorithms to the matrix $M$ defined implicitly by the above products, and never computed explicitly. This leads to the CG normal equation residual method and the CG normal equation error method, respectively (see [GL13, Section 11.3.9]), to which we should extend our study of the CG method.

Efficient implementation of the accelerated CG algorithms is our next challenge: in order to avoid numerical problems one should incorporate our fast LRA algorithms into numerically stable versions of the CG algorithms such as [KL07], or [BDK15].

A Ranks and Norms of Random Matrices

A.1 Ranks of random matrices

Theorem 100. Suppose that $A$ is an $m \times n$ matrix of full rank $k = \min\{m, n\}$, $F$ and $H$ are $r \times m$ and $n \times r$ matrices, respectively, for $r \leq k$, and the entries of these two matrices are nonconstant linear combinations of finitely many i.i.d. random variables $v_1, \ldots, v_h$.

Then the matrices $F$, $FA$, $H$, and $AH$ have full rank $r$.

(i) with probability 1 if $v_1, \ldots, v_h$ are Gaussian variables and
(ii) with a probability at least $1 - r/\text{card}(S)$ if they are random variables sampled under the uniform probability distribution from a finite set $S$ having cardinality $\text{card}(S)$.

Proof. The determinant, $\det(B)$, of any $r \times r$ block $B$ of a matrix $F$, $FA$, $H$, or $AH$ is a polynomial of degree $r$ in the variables $v_1, \ldots, v_h$, and so the equation $\det(B) = 0$ defines an algebraic variety of a lower dimension in the linear space of these variables (see [BV88, Proposition 1]). Clearly such a variety has Lebesgue and Gaussian measures 0, both being absolutely continuous with respect to one another. This implies claim (i) of the theorem. Derivation of claim (ii) from a celebrated lemma of [DL78], also known from [Z79] and [S80], is a well-known pattern, specified in some detail in [PW08].

Remark 101. The theorem implies that the matrices $G$, $H$ and $W$ of Definition 4 have rank $r$ with probability 1.
Theorem 104. Let \((ii)\) and our Theorem 112 for claim \((iii)\).

See [CD05, Proof of Lemma 4.1] for claim \((i)\), [HMT11, Proposition 10.2] for claim

Proof. Let \(p \times q \) be positive integers and let \(t \geq 0\). Then

(i) Probability \( \{ \nu_{p,q} > t + \sqrt{p} + \sqrt{q} \} \leq \exp(-t^2/2) \) (see [DS01, Theorem II.7]),

(ii) \( \mathbb{E}(\nu_{p,q}) \leq \sqrt{p} + \sqrt{q} \) (see [HMT11, Proposition 10.1] for \(S = T = 1\)), and

(iii) \( \mathbb{E}(\nu_{p,q,C}) = \mathbb{E}(\nu_{1, pq,C}) \leq f + \frac{1}{f} \), for \( f := \sqrt{2 \ln(\max\{2, pq\})} \) (see [SST06, Lemma A.3]).

Remark 103. Given a \( p \times q \) Gaussian matrix \( W \), apply Theorem 102 and obtain that

\[ E(||W||) \leq \sqrt{p} + \sqrt{q}, \quad E(||W||_C) \leq f + \frac{1}{f} \] where \( f \approx \sqrt{2 \ln(pq)} \) for large integers \( pq \);

furthermore both norms \( ||W|| \) and \( ||W||_C \) deviate from their expected values by more than a factor \( \zeta > 1 \) with a probability that decays exponentially fast as \( \zeta \) grows to infinity. Notice that the theorem also implies estimates for \( ||W||_F = \nu_{mn,1} \) where \( W \in \mathcal{G}^{m\times n} \).

Theorem 104. Let \( x > 0 \) and let \( \Gamma(x) \) denote the Gamma function of (34). Then

(i) Probability \( \{ \nu_{p,q}^+ \geq px^2 \} \leq \frac{x^{p-1}}{\Gamma(p-p+2)} \) for \( p \geq q \geq 2 \),

(ii) \( \mathbb{E}(\nu_{p,q}^+) < e^{\sqrt{p}/|p-q|} \) provided that \( p \neq q > 1 \) and \( e = 2.71828 \ldots \) and

(iii) Probability \( \{ \nu_{q,q}^+ \geq 1/x \} \leq \sqrt{\frac{2}{\pi}} \) for \( q \geq 2 \).

Proof. See [CD05, Proof of Lemma 4.1] for claim (i), [HMT11, Proposition 10.2] for claim (ii) and our Theorem 112 for claim (iii).

Remark 105. Part (iii) improves the following estimate of [SST06, Theorem 3.3]: Probability \( \{ \nu_{q,q}^+ \geq x \} \leq 2.35 \sqrt{x} \) for \( q \geq 2 \).

Remark 106. The probabilistic upper bounds of Theorem 104 on \( \nu_{p,q}^+ \) are quite reasonable even where \( p = q \), but are strengthened very fast as the difference \( |p-q| \) grows from 1. Furthermore recall that \( 1/\nu_{p,q} \geq |\sqrt{p} - \sqrt{q}| \) with high probability (see [RV09], [TV10]).

In sum all these bounds combined imply that a \( p \times q \) Gaussian matrix is well-conditioned unless the integer \( p+q \) is large or the integer \( |p-q| \) is close to 0. With some grain of salt we can consider such a matrix well-conditioned even where \( p = q \). (Also notice that \( \kappa(W) = 1 \) for \( p = q = 1 \).)

A.3 Norms of sparse Gaussian matrices and their pseudo inverses

The estimates of Theorem 102 cover sparse matrices as well, and we immediately bound the norm of a sparse Gaussian matrix as follows.

Theorem 107. The squared Frobenius norm of a \( \mathcal{N} \mathcal{Z} \)-Gaussian matrix \( W \) is equal in distribution to the \( \chi^2 \)-function of \( \mathcal{N} \mathcal{Z} W \) i.i.d. random variables, \( ||W||_F^2 \overset{d}{=} \chi^2(\mathcal{N} \mathcal{Z} W) \)

Next we estimate the norm of the Moore-Penrose pseudo inverse of a sparse Gaussian matrix. We assume that the matrix has full rank with probability 1 (cf. Theorem 100), and in our probabilistic analysis assume that the matrix does have full rank.

Definition 108. Let \( \mathcal{N} \mathcal{Z}_{i,:}(W) \) denote the numbers of nonzero entries in the \( i \)th row of a \( p \times q \) matrix \( W = (w_{i,j})_{p,q} \), and let “\( \preceq \)” stand for “stochastically less than”.

65
Lemma 109. (See [SST06, Lemma A.2].) Consider the inner product \( p = v^T g \) where \( ||v|| = 1 \) and \( g \) is a Gaussian vector. Then for any real value \( t \) it holds that

\[
\text{Probability}\{ |p - t| \leq x \} \leq \sqrt{\frac{2}{\pi}} x.
\]

Theorem 110. Let \( W \) be a \( p \times q \mathcal{N}\mathcal{Z} \)-Gaussian matrix of rank \( q \). Then

\[
\text{Probability}\{ ||W^+|| \geq 1/x \} \leq \sqrt{\frac{2q}{\pi}} x.
\]

Proof. Deduce from [GL13, Theorem 8.1.2]) that \( 1/||W^+|| = \sigma_q(W) = \min_{||v||=1} ||Wv||. \)

Let \( v = (v_j)_{j=1}^q \) be a vector such that \( ||v|| = 1 \) and \( 1/||W^+|| = ||Wv||. \)

Notice that \( ||Wv||^2 = \sum_i \left( \sum_j v_j w_{i,j} \right)^2 \geq \left( \sum_j v_j w_{i,j} \right)^2 \), and so

\[
1/||W^+|| = ||Wv|| \geq \left| \sum_j v_j w_{i,j} \right| \quad \text{for all } i.
\]

For every \( i \) drop the terms of this sum where \( w_{i,j} = 0 \), that is, rewrite the sum as \( |\sum_{j \in \mathcal{N}\mathcal{Z}_i} v_j w_{i,j}| \). Then rewrite it as the inner product \( u^{(i)}^T g^{(i)} \) where \( g^{(i)} = (w_{i,j})_{j \in \mathcal{N}\mathcal{Z}_i} \) is the Gaussian vector made up of all nonzero components of the vector \((\pm w_{i,j})_{j=1}^q\) and \( u^{(i)} = (\pm v_j)_{j \in \mathcal{N}\mathcal{Z}_i} \) is a subvector of the vector \((v_j)_j\).

Notice that \( \max_j |v_j| \geq 1/\sqrt{q} \) and let \( |v_1| \geq 1/\sqrt{q} \), say. Then notice that \( \pm w_{i,1} \) is a Gaussian variable rather than 0 for some \( i \), say, for \( i = 1 \). (If \( w_{i,1} = 0 \) for all \( i \), then rank\((W) < q \), contrary to our assumption.)

Hence \( |\sum_j v_j w_{1,j}| \) is the inner product of a Gaussian vector and a vector \( u^{(1)} \) with \( ||u^{(1)}|| \geq 1/\sqrt{q} \). Apply Lemma 109 for \( t = 0 \) to the inner product \( u^{(1)}^T g^{(1)}/||u^{(1)}|| \) and deduce the theorem.

Remark 111. If rank\((W) = p \), then apply the theorem to the matrix \( W^T \) replacing \( W \), recall that \( ||W^+|| = ||W^T|| \), and obtain the following dual counterparts to the theorem:

\[
\text{Probability}\{ ||W^+|| \geq 1/x \} \leq \sqrt{\frac{2p}{\pi}} x.
\]

Theorem 112. Let \( W \) be a \( p \times q \) dense-Gaussian matrix, then

\[
\text{Probability}\{ ||W^+|| \geq 1/x \} \leq \sqrt{\frac{2}{\pi}} x.
\]

Proof. Proceeding as in the proof of Theorem 110 notice that now

\[
|\sum_j v_j w_{i,j}| = (\pm v_j)^T (\pm w_{i,j})_{j=1}^q
\]

where \((\pm w_{i,j})_{j=1}^q\) is a Gaussian vector of dimension \( q \) and \((\pm v_j)_j\) is a unit vector.
B Abridged Hadamard and Fourier Multipliers

Recall the following recursive definition of dense and orthogonal (up to scaling by constants) \( n \times n \) matrices \( H_n \) of Walsh-Hadamard transform for \( n = 2^k \) (see [M11, Section 3.1] and our Remark 113):

\[
H_2 := \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \text{ and } H_{2q} := \begin{pmatrix} H_q & H_q \\ H_q & -H_q \end{pmatrix} \quad (58)
\]

for \( q = 2^h, h = 0, 1, \ldots, k - 1 \). Recursive representation (58) enables fast multiplication of a matrix \( H_n \) by a vector, using \( nk \) additions and subtractions for \( n = 2^k \).

Now shorten the recursive process by fixing a recursion depth \( d, 1 \leq d < k \), and applying equation (58) where \( q = 2^h s, h = k - d, k - d + 1, \ldots, k - 1 \), and \( H_s = I_s \) for \( n = 2^d s \). For two positive integers \( d \) and \( s \), denote the resulting \( n \times n \) matrix \( H_{n,d} \) and call it the matrix of \( d \)-abridged Hadamard (AH) transform if \( 1 \leq d < k \). In particular

\[
H_{n,1} := \begin{pmatrix} I_s & I_s \\ I_s & -I_s \end{pmatrix}, \text{ for } n = 2s; \quad H_{n,2} := \begin{pmatrix} I_s & I_s & I_s & I_s \\ I_s & -I_s & I_s & -I_s \\ I_s & I_s & -I_s & -I_s \\ I_s & -I_s & -I_s & I_s \end{pmatrix}, \text{ for } n = 4s. \quad (59)
\]

For a fixed \( d \), the matrix \( H_{n,d} \) is still orthogonal up to scaling, has \( q = 2^d \) nonzero entries in every row and column, and hence is sparse unless \( k - d \) is a small integer. Then again, representation via recursive process (58) enables us to multiply such a matrix by a vector fast, by using just \( dn \) additions/subtractions and allowing efficient parallel implementation (see Remark 114).

Likewise recall a recursive process of the generation of the \( n \times n \) matrix \( \Omega_n \) of discrete Fourier transform (DFT) at \( n \) points, for \( n = 2^k \):

\[
\Omega_n := (\omega_{n}^{ij})_{i,j=0}^{n-1}, \text{ for } n = 2^k \text{ and a primitive } n \text{th root of unity } \omega_n := \exp \left( \frac{2\pi}{n} \sqrt{-1} \right). \quad (60)
\]

The matrix \( \Omega_n \) is unitary up to scaling by \( \frac{1}{\sqrt{n}} \). We can multiply it by a vector by using \( 1.5nk \) flops, and can efficiently parallelize this computation if, instead of its representation by entries, we apply its recursive representation, called decimation in frequency (DIF) radix-2 representation (see [P01, Section 2.3] and our Remark 113):

\[
\Omega_{2q} := \tilde{P}_{2q} \begin{pmatrix} \Omega_q & \Omega_q \\ \Omega_qD_q & -\Omega_qD_q \end{pmatrix}, \tilde{D}_q := \text{diag}(\omega_{n}^{ij})_{i=0}^{n-1}. \quad (61)
\]

Here \( \tilde{P}_{2q} \) is the matrix of odd/even permutations such that \( \tilde{P}_{2q}(\mathbf{u}) = \mathbf{v}, \mathbf{u} = (u_i)_{i=0}^{2^q-1}, \mathbf{v} = (v_i)_{i=0}^{2^q-1}, v_j = u_{2j}, v_{j+2^q-1} = u_{2j+1}, j = 0, 1, \ldots, 2^q - 1 \); \( q = 2^h, h = 0, 1, \ldots, k \), and \( \Omega_1 = (1) \) is the scalar 1.

Now shorten the recursive process by fixing a recursion depth \( d, 1 \leq d < k \), replacing \( \Omega_s \) for \( s = n/2^d \) by the identity matrix \( I_s \), and then applying equation (61) for \( q = 2^h, h = k - d, k - d + 1, \ldots, k - 1 \). For \( 1 \leq d < k \) and \( n = 2^d s \), we denote the resulting \( n \times n \) matrix \( \Omega_{n,d} \) and call it the matrix of \( d \)-abridged Fourier (AF) transform. It is also unitary (up to scaling), has \( q = 2^d \) nonzero entries in every row and column, and thus is sparse unless \( k - d \) is a small integer. We can represent such a matrix by its entries, but if we rely on recursive representation (61), then again its multiplication by a vector involves just \( 1.5dn \) flops and allows highly efficient parallel implementation.
By complementing the above definitions of the matrices with randomization we define the matrices $DH_n P$ and $DH_n P$ of randomized Hadamard and Fourier transforms or (by using acronyms) the matrices of $RHT$ and $RFT$. Here $D$ is a random $n \times n$ diagonal matrix whose entries are independent and uniformly distributed on the complex unit circle $\{ x : ||x|| = 1 \}$ and $P$ is a random $n \times n$ permutation matrix. Suppose that we sample $l$ columns from such a matrix uniformly at random and without replacement and scale the resulting matrix by a constant $\sqrt{\frac{n}{l}}$. Then such a scaled $n \times l$ matrix is called the matrix of a subsampled randomized Hadamard or Fourier transform ($SRHT$ or $SRFT$).

Likewise we define the matrices $DF_{n,d} P$ and $DH_{n,d} P$ of abridged randomized Fourier and Hadamard’s transforms.

**Remark 113.** The paper [PZ16] defines the same matrices of abridged randomized Hadamard and Fourier transforms but calls them matrices of abridged scaled and permutated Hadamard and Fourier transforms.

**Remark 114.** Other observations besides flop estimates can decide the efficiency of multipliers. For example, a special recursive structure the matrices $H_{2k,d}$ and $\Omega_{2k,d}$ abridged Hadamard and Fourier transforms allows highly efficient parallel implementation of their multiplication by a vector based on Application Specific Integrated Circuits (ASICs) and Field-Programmable Gate Arrays (FPGAs), incorporating Butterfly Circuits [DE].

### C Computation of sampling and rescaling matrices

We begin with the following simple computations. Given an $n$ vectors $v_1, \ldots, v_n$ of dimension $l$, write $V = (v_i)_{i=1}^n$ and compute $n$ sampling probabilities

$$p_i = v_i^T v_i / ||V||^2_F, i = 1, \ldots, n.$$  \hfill (62)

Notice that $p_i \geq 0$ for all $i$ and $\sum_{i=1}^n p_i = 1$.

Next assume that some sampling probabilities $p_1, \ldots, p_n$ are given to us and recall the Exactly($l$) and the Expected($l$) algorithms, which are [DMM08, Algorithms 4 and 5], respectively (cf. [BW17, Definition 3.6]).

For a fixed positive integer $l$ they sample either exactly $l$ columns of an input matrix $W$ (the $i$-th column with probability $p_i$) or at most $l$ its columns in expectation (the $i$-th column with probability $\min\{1, lp_i\}$), respectively.

**Algorithm 115.** The Exactly($l$) sampling and rescaling.

**INPUT:** Two integers $l$ and $n$ such that $1 \leq l \leq n$ and $n$ nonnegative scalars $p_1, \ldots, p_n$ such that $\sum_{i=1}^n p_i = 1$.

**INITIALIZATION:** Write $S := O_{n,l}$ and $D := O_{l,l}$.

**COMPUTATIONS:**

1. For $t = 1, \ldots, l$ do
   - Pick $i_t \in \{1, \ldots, n\}$ such that $\text{Probability}(i_t = i) = p_i$;
   - $s_{i_t,t} := 1$;
   - $d_{t,t} = 1 / \sqrt{lp_{i_t}}$;
   - end

2. Write $s_{i,t} = 0$ for all pairs of $i$ and $t$ unless $i = i_t$.  

68
OUTPUT: $n \times t$ sampling matrix $S = (s_{i,t})_{i,t=1}^{n,t}$ and $t \times t$ rescaling matrix $D = \text{diag}(d_{t,t})_{t=1}^{l}$. 

Algorithm 116. The Expected($l$) sampling and rescaling. 

Input, Output and Initialization are as in Algorithm 115. 

Computations: Write $t := 1$; 
for $j = 1, \ldots, n$ do 
Pick $j$ with probability $\min\{1, lp_j\}$; 
if $j$ is picked, then 
$s_{j,t} := 1$; 
$d_{t,t} := 1/\min\{1, \sqrt{lp_j}\}$; 
$t := t + 1$; 
end 
end 

Computations of Algorithm 116 involve $ln$ memory cells and $O((r+1)n + l\log(l))$ flops. 

Next we recall [BW17, Lemmas 3.7 and 3.8] (cf. [RV07]). 

Theorem 117. Suppose that $n > r$, $V \in \mathbb{C}^{n \times r}$ and $V^TV = I_r$. Let $0 < \delta \leq 1$ and $4k\ln(2k/\delta) < l$. Define sampling probabilities by equations (62) and then compute the sampling and rescaling matrices $S$ and $D$ by applying Algorithm 116. Then, for all $i = 1, \ldots, r$ and with a probability at least $1 - \delta$ it holds that 

$1 - \sqrt{4r\ln(2r/\delta)/l} \leq \sigma_i^2(V^TSD) \leq 1 + \sqrt{4r\ln(2r/\delta)/l}$. 

(63) 

Notice that $||D^{-1}||_F \leq \sqrt{r}$. 

Theorem 118. Define the sampling and scaling matrices $S$ and $D$ as in Theorem 117. Then for an $m \times n$ matrix $W$ it holds with a probability at least 0.9 that $||WSD||_F^2 \leq ||W||_F^2$. 

D LUP- and QRP-based CUR LRA 

D.1 Computation of CUR LRA by means of rank-revealing factorization 

The algorithms of [GE96] and [P00] output submatrices $W_{I,J}$ of an input matrix $W$ that have locally maximal volumes. This leads towards the second CUR criterion, but also to the following bounds on the $r$th singular values of these submatrices. 

Theorem 119. (See [CI94], [GZT95], [GTZ97], [GTZ97a], [GE96], [P00].) 

Suppose that $\min\{h, h'\} \geq 1$, $W_{I,J} \in \mathbb{C}^{k \times l}$ is a submatrix of a matrix $W \in \mathbb{C}^{m \times n}$, and $t^2_{p,r,g} = (p - r)rg^2 + 1$ (cf. (57)). Then 

$t_{n,r,h} \sigma_r(W_{I,J}) \geq \sigma_r(W_{I,:})$ 

if $k = r \leq l$ and if the volume $v_2(W_{I,J})$ is locally column-wise $h$-maximal and 

$t_{m,r,h'} \sigma_r(W_{I,J}) \geq \sigma_r(W_{:,J})$ 

if $k \geq l = r$ and if this volume is locally row-wise $h'$-maximal.
Notice that \( v_2(W_{I,J}) = v_{2,r}(W_{I,J}) \) for the above matrices \( W_{I,J} \) of sizes \( r \times l \) and \( k \times r \).

**Proof.** The theorem turns into [P00, Lemma 3.5] for \( k = l = r \) and is extended to the case where \( r = \min\{k,l\} \) because no singular value of a matrix increases in the transition to its submatrix. \( \square \)

**Corollary 120.** The estimates of Theorem 119 hold for any \( h \geq 1 \) and \( k \times l \) matrix \( W_{I,J} \) output by [GE96, Algorithm 4 for \( f = h \)] applied to a matrix \( W \) of size \( m \times l \) or \( k \times n \).

**Proof.** The corollary holds if \( k = l = r \) because the output matrix \( W_{I,J} \) has column-wise or row-wise locally maximal volume in this case. By applying Algorithm 53 we extend this result to any \( k \) and \( l \) not exceeded by \( r \). \( \square \)

**Remark 121.** Suppose that an \( m \times l \) input matrix \( W \) for \( m \geq l \) has numerical rank \( r \), that is, the ratio \( ||W||/\sigma_r(W) \) is not large, while \( \sigma_r(W) \gg \sigma_{r+1}(W) \). Then Corollary 120 defines a CUR generator \( W_{k,l} = W_{I,J} \) such that \( ||U|| \leq t_{m,k,h}/||\tilde{\sigma}_{r+1}|| \) for \( \tilde{\sigma}_{r+1} \). Its computation involves \( ml \) memory cells and \( O(ml^2) \) flops. By virtue of Claim (ii) of Corollary 21 such a CUR generator defines a close canonical CUR LRA of the matrix \( W \). If \( W \) is a tall-skinny matrix, that is, \( m \gg l \) and if \( k \ll m \), then by applying the randomized algorithms of Theorems 117 and 118 we can compute its \( q \times n \) submatrix \( \tilde{W} \) for \( q = 4fk\ln(2k/\delta) \) that for sufficiently large factor \( f \) with a high probability has its singular values within a factor \( 1 + \phi \) from those of the matrix \( W \) for \( \phi \to 0 \) as \( q \to \infty \). Then we can apply the algorithms of Corollary 120 to the matrix \( \tilde{W} \) rather than \( W \) and decrease the upper bound on the norm \( ||U|| \) of the nucleus \( U \) by a factor of \( t_{m,k,h}/t_{q,k,h} \) and also use by a factor of \( l \) fewer flops.

**Remark 122.** [GE96, Algorithm 4 for \( f = h \)] relies on computing strong rank-revealing QR factorization of the matrix \( W \). Alternative [P00, Algorithm 3 for \( \mu = h \)] computes strong rank-revealing LU factorization. It also uses \( O(mn \min\{m,n\}) \) flops and ensures the upper bounds of Theorem 119 for \( t_{n,r,h} \) replaced by \( t_{n,r,h}^2 \) and for \( t_{m,r,h'} \) replaced by \( t_{m,r,h'}^2 \), but it is simpler for implementation and has produced adequate results in our experiments.

### D.2 An iterative LUP-based C-A algorithm

Similarly to algorithms of [GE96] and [P00] the iterative C-A algorithm of [GOSTZ10] seeks a \( r \times r \) submatrix of locally maximal volume in a \( r \times n \) matrix. We first revisit and then modify this algorithm.

**Definition 123.** Let \( C_g \) for \( g > 1 \) denote a \( r \times g \) matrix and let \( C_{g,j} \) denote its \( r \times (g - 1) \) submatrix obtained by removing the \( j \)th column of \( C_g \). If \( v_2(C_{g,j'}) = \max_{j=1}^{g} v_2(C_{g,j}) \), then write \( C_{g-1} =: C_{g,j'} \) and call the map \( C_g \to C_{g-1} \) a greedy local contraction or just a greedy contraction of the matrix \( C_g \).

**Definition 124.** Let \( C_g \) be a \( r \times g \) submatrix of a \( r \times n \) matrix \( W \), append on its right the \( j \)th column of the matrix \( W \), and let \( C_{g,j,+} \) denote the resulting \( r \times (g + 1) \) matrix. If \( v_2(C_{g,j,+}) = \max_{j=1}^{g} v_2(C_{g,j,+}) \), then write \( C_{g+1} =: C_{g,j,+} \) and call the map \( C_g \to C_{g+1} \) a greedy local expansion or just a greedy expansion of the matrix \( C_g \).

\(^{17}\) [GE96, Algorithm 4 for \( f = h \)] and similarly [P00, Algorithm 3 for \( \mu = h \)] use parameters \( f \) and \( \mu \) a little exceeding 1 in order to control the impact of rounding errors.
The algorithm of [GOSTZ10] seeks a submatrix having locally maximal volume among the $r \times r$ submatrices of a $r \times n$ matrix. The algorithm performs a greedy expansion followed by a greedy contraction of a fixed matrix, repeats these two steps recursively and stops wherever one verifies that the volume of a current matrix is locally maximal.

Each recursive step involves just $(n - r)r$ flops and strictly increases the volume of the submatrix. Thus the algorithm can encounter no submatrix twice and has not more iterations than the exhaustive search. For the worst case input we have no better upper bound, but empirically much fewer iterations are usually sufficient.

The paper [GOSTZ10] analyzes the algorithm by using basic concept of a dominant submatrix.

We specified the choice of the initial matrix $\bar{B}$ and place it in the leftmost position by reordering the columns of the matrix $A$.

The algorithm uses $i(r) = O(r^3)$ flops for inverting an $r \times r$ matrix at the initialization stage and then uses $(n - r)r$ flops per iteration (by exploiting a very special form of the matrix $\bar{B}$), that is, $(n - r)r\alpha + i(r)$ in $\alpha$ iterations.

We specified the choice of the initial matrix $\bar{B}$ in the otherwise equivalent algorithm below. It expresses the algorithm as a recursive LU factorization with column pivoting.

**Algorithm 125.** Computing a dominant submatrix.

**INPUT:** A $r \times n$ matrix $A$ of full rank $r$.

**OUTPUT:** A $r \times r$ dominant submatrix $B$.

**INITIALIZATION:** Fix a $r \times r$ nonsingular submatrix of $\bar{B}$ and place it in the leftmost position by reordering the columns of the matrix $A$.

**COMPUTATIONS:**

1. Compute the matrix $C = \bar{B}^{-1}A$ and find its absolutely maximal entry $c_{ij}$. If $|c_{ij}| \leq 1$, then $B$ is dominant; in this case output $B =: \bar{B}$ and stop.

2. Otherwise swap columns $i$ and $j$ of $C$. (This strictly increases the volumes of the leftmost $r \times r$ submatrices of $C$ and $A$.) Denote by $\bar{B}$ the leftmost $r \times r$ submatrix of $A$ and go to stage 1.

The algorithm uses $i(r) = O(r^3)$ flops for inverting an $r \times r$ matrix at the initialization stage and then uses $(n - r)r$ flops per iteration (by exploiting a very special form of the matrix $\bar{B}$), that is, $(n - r)r\alpha + i(r)$ in $\alpha$ iterations.

We specified the choice of the initial matrix $\bar{B}$ in the otherwise equivalent algorithm below. It expresses the algorithm as a recursive LU factorization with column pivoting.

**Algorithm 126.** An iterative C-A algorithm by means of recursive LUP factorization.

**INPUT and OUTPUT** as in Algorithm 125.

**COMPUTATIONS:**

1. Compute LUP factorization of the matrix $A$, where $L$ is a $r \times r$ lower triangular matrix, $U = (U_0 \mid U_1)$ is a $r \times n$ upper triangular matrix, $U_0$ is a nonsingular $r \times r$ matrix, and $P$ is an $n \times n$ permutation matrix.

2. Compute the matrices $\bar{B} = LU_0$ and $C = \bar{B}^{-1}AP^T = (I_r \mid C')$ for $C' \in \mathbb{C}^{r \times (n-r)}$, $(\det(C_{\cdot J'}) = \det(\bar{B}^{-1}) \det((AP^T)_{\cdot J'})$ for any $r$-tuple of indices $J'$, and so the map $AP^T \to A'$ keeps pairwise order of the volumes of $r \times r$ submatrices.)
3. If \( ||A'||_C \leq 1 \), output the submatrix \( A_{-\mathcal{J}} \) of the matrix \( A \) where the set \( \mathcal{J} \) is made up of the \( r \) indices of nonzero columns of the permutation matrix \( P \). (In this case \( I_r \) is a dominant \( r \times r \) submatrix of \( A' \).) Otherwise write \( C \to A \) and go to stage 1.

**D.3 Computation of a contracted generator: basic techniques and results**

Osinsky’s techniques of [O16] support extension of the algorithm of [GOSTZ10] towards the maximization of the volume or \( r \)-projective volume of a \( k \times l \) CUR generator for \( r < \min \{k, l\} \). Next we explore and extend some recipes of [O16] towards this goal by recursively applying QRP factorization. The arithmetic cost of their iterative step a little exceeds that of [GOSTZ10], but we decrease the output error bound, based on Theorem 39.

We begin with some results supporting greedy expansion/contraction.

Hereafter a matrix is said to be a *matrix basis* for its range, that is, for its column span.

The following theorems provide sufficient criterion for local maximality of the volume of a fixed submatrix.

**Theorem 127.** Let \( B = (A \mid b) \) denote a \( r \times (q + 1) \) matrix for \( r > q \) and let \( U \) be a unitary matrix basis for the null space of \( A^* \). Then

\[
v_2(B) = v_2(A) ||U^*b||.
\]  

**Proof.** Recall that \( v_2^2(B) = \det(B^*B) \) and notice that

\[
B^*B = \begin{pmatrix} A^*A & A^*b \\ b^*A & b^*b \end{pmatrix} = \begin{pmatrix} I_r & 0 \\ x^* & 1 \end{pmatrix} \begin{pmatrix} A^*A & 0 \\ 0^* & b^*Sb \end{pmatrix} \begin{pmatrix} I_r & x \\ 0^* & 1 \end{pmatrix}
\]

for \( x := (A^*A)^{-1}A^*b \) and \( S := I_r - A(A^*A)^{-1}A^* \). Hence \( v_2^2(B) = \det(B^*B) = \det(A^*A) \det(b^*(I_r - A(A^*A)^{-1}A^*)b) \). Recall that \( S = I_r - A(A^*A)^{-1}A^* = UU^* \) where \( U \) is a unitary matrix basis for the null space of the matrix \( A \) (see [S98, Section 1.4.2]). Substitute \( \det(A^*A) = v_2^2(A) \) and \( I_r - A(A^*A)^{-1}A = UU^* \) and deduce the theorem.

**Corollary 128.** A \( r \times q \) submatrix \( A \) of a \( r \times n \) matrix \( W := (A \mid V) \) for \( q < r \leq n \) has locally \( h \)-maximal volume for \( h \) equal to the maximal norm of the column of the matrix \( U^*V \) where \( U \) is a unitary matrix basis for the null space of the matrix \( A \).

**Proof.** Suppose that \( v_2(A) \leq v_2(\bar{A} \mid b) \) where \( \bar{A} \) is a \( r \times (q - 1) \) submatrix of \( A \) and \( b \) is a column of \( V \). Write \( B = (A \mid b) \) as in Theorem 127, and deduce from this theorem that \( v_2(B) = v_2(A)||U^*b|| \), and so \( v_2(B) \leq h v_2(A) \) for \( h = \max ||U^*b||. \) Also recall that \( v_2(B) \geq v_2(\bar{A} \mid b) \) because \( (\bar{A} \mid b) \) is a submatrix of \( B \). Thus \( hv_2(A) \geq v_2(B) \geq v_2((A \mid b)) \).

**Corollary 129.** Remove the \( j' \)th column \( b_{j'} \), \( 1 \leq j' \leq p \), from a \( p \times p \) upper triangular matrix \( B = (b_{ij})_{i,j=1}^p \), denote the resulting \( p \times (p - 1) \) submatrix \( A = A_{j'} \) and denote \( b_{j'} \) the \( j' \)th coordinate of the vector \( b_{j'} \). Then

\[
v_2(B) = v_2(A) ||b_{j'}||.
\]  

**Proof.** Apply Theorem 127 and observe that in this case \( U \) is the \( j' \)th coordinate vector, \((0, \ldots, 0, 1, 0, \ldots, 0)^*\), filled with zeros except for the \( j' \)th coordinate 1.
Theorem 130. Let \( B := (A \mid b) \) denote a \( p \times (q + 1) \) matrix for \( p \leq q \). Then
\[
v_2^2(A) = v_2^2(B)(1 - b^*(BB^*)^{-1}b) = v_2^2(B)(1 - \|B^+b\|^2) \tag{66}
\]
and
\[
v_2^2(B) = v_2^2(A)(1 + b^*(AA^*)^{-1}b) = v_2^2(A)(1 + \|A^+b\|^2). \tag{67}
\]
In particular if the matrix \( B \) is unitary, then
\[
v_2^2(A) = v_2^2(B)(1 - \|b\|^2). \tag{68}
\]
and if the matrix \( A \) is unitary, then
\[
v_2^2(B) = v_2^2(A)(1 + \|b\|^2). \tag{69}
\]
Proof. \(^{18}\) Observe that
\[
AA^* = BB^* - bb^*,
\]
and so
\[
det(AA^*) = det(BB^* - bb^*) = det(BB^*) det(I_p - (BB^*)^{-1}bb^*).
\]
Substitute \( det(BB^*) = v_2^2(B) \), \( det(I_p - (BB^*)^{-1}bb^*) = det(I_1 - b^*(BB^*)^{-1}b) \), and \( I_1 = 1 \) and obtain equation (66). One can similarly prove equation (67).

Similarly to Corollary 128 deduce the following result (use equation (67) instead of Theorem 127).

Corollary 131. Given a \( r \times q \) submatrix \( A \) of a \( r \times n \) matrix \( (A \mid V) \) for \( r \leq q \leq n \), let \( b \) denote the maximal spectral norm of the columns of the matrix \( A^+V \). Then \( v_2(A) \) is locally \( (1 + b^2) \)-maximal.

[O16, Lemma 4] extends (67) to a bound on the \( r \)-projective volume as follows:

Theorem 132. Suppose that again \( B := (A \mid b) \) denote a \( p \times (q + 1) \) matrix and let \( v_{2,r}(A) \) be the maximal \( r \)-projective volume among all \( p \times q \) submatrices of the matrix \( B \). Then
\[
v_{2,r}^2(B) \geq v_{2,r}^2(A)(1 + b^*(A(rA(r)^*)^{-1}b) = v_{2,r}^2(A)(1 + \|A(r)^+b\|^2). \tag{70}
\]

The theorem does not imply extension of Corollary 131, but one can maximize \( r \)-projective volume via maximization of volume by applying Algorithm 53.

D.4 QRP-based greedy iterative search for locally maximal volume

Corollaries 128 and 131 motivate greedy expansion and contraction based on Theorems 127 and 130. Suppose that we seek a \( r \times l \) submatrix of a \( r \times n \) matrix \( W \), for \( \max\{r, l\} \leq n \). We can fix its column having the largest norm, denote it \( C_1 \), recursively apply greedy expansion based on Theorems 127 and 130, until we arrive at a \( r \times l \) submatrix \( C_l \) of \( W \). Then we can check whether the sufficient criteria of Corollaries 128 and 131 for local maximization of the volume \( v_2(C_l) \) are satisfied, and if not, apply a sequence of expansions/contractions to the matrix \( C_l \) similarly to Algorithm 125. Next we describe two implementations of this outline.

\(^{18}\) We slightly simplify the proof of a similar result hidden in the proof of [O16, claim 4 of Lemma 1].
Algorithm 133. Greedy expansions from a vector up to a square matrix.

**INPUT:** A $r \times n$ matrix $W$ and an integer $q$ such that $q \leq r \leq n$.

**OUTPUT:** An $n \times n$ permutation matrix $P$ such that every leftmost $r \times g$ submatrices $C_g$ of $WP$ is a greedy expansion of its $r \times (g-1)$ leftmost predecessor $C_{g-1}$ for $g = 2, \ldots, q$.

**INITIALIZATION:** Write $W_0 = V_0 := W$, $U_0 := I_r$, $P_0 := I_n$ and let $C_0$ denote the empty $r \times 0$ matrix.

**COMPUTATIONS:** For $g = 0, 1, \ldots, q - 1$, recursively proceed as follows:

1. Given a $r \times n$ matrix $W_g := U_gWP_g = (C_g \| V_g)$, for a unitary matrix $U_g$, a permutation matrix $P_g$, a $r \times g$ matrix $C_g := \begin{pmatrix} R_g \\ O_{g,r-g} \end{pmatrix}$, and an $g \times g$ upper triangular matrix $R_g$ having all its diagonal entries 1, fix a column vector $b_g$ whose subvector made up of its last $r - g$ coordinates has maximal spectral norm among all such subvectors of the column vectors of the matrix $V_g$. Form the matrix $B_g := (C_g \| b_g)$.

2. Move the vector $b_g$ from its position in the matrix $W_g$ into the $(g+1)$st column of the new matrix, $W_g' := W_gP_{g+1} = (C_g' \| V_g')$ where $C_g' := (C_g \| b_g)$.

3. Define a Householder reflection matrix $H'_g$ such that every vector $v$ shares its first $g$ coordinates with the vector $H_gv$ for $H_g = \text{diag}(I_g,H'_g)$ and the vector $b_{g+1} := H_gb_g$ shares its remaining $r - g$ coordinates with the coordinate vector $(1,0,\ldots,0)^T$ of dimension $r - g$ (see [GL13, Section 5.1]).

4. Compute the matrix $W_{g+1} := H_gW'_g$ and let $C_{g+1}$ denote its $r \times (g+1)$ leftmost submatrix, such that $C_{g+1} := H_gC_g' = \begin{pmatrix} R_{g+1} \\ O_{g+1,r-g-1} \end{pmatrix}$ and $R_{g+1} := (R_g \| b_{g+1})$ is a $r \times (g+1)$ upper triangular matrix with diagonal entries 1.

5. Compute and output the permutation matrix $P := P_0P_1 \cdots P_{q-1}$.

**Correctness** of the algorithm follows because the matrices $U_g = \text{diag}(O_{g,g},I_{r-g})$ are unitary matrix bases for the null spaces of the matrices $C_g$ for $g = 0, \ldots, q - 1$.

**Computational cost.** The algorithm computes the squares of the $l_2$-norms of $n-g$ vectors of dimensions $r-g$ for $g = 0, \ldots, q - 1$ by using $\sum_{g=0}^{q-1}(n-g)(2r-2g-1) < 2qrn$ flops. It multiplies $(r-g) \times (r-g)$ Householder reflection matrices by $(r-g) \times (n-g)$ matrices by using at most $\sum_{g=0}^{q-1}6(n-g)(r-g)) < 6qrn$ flops, hence less than $8qrn$ flops overall.

Algorithm 134. Greedy expansions from a $r \times r$ submatrix to a $r \times q$ submatrix for $q > r$.

**INPUT:** An $r \times n$ matrix $W$, with a $r \times r$ leftmost submatrix $C_r$, and an integer $q$, $r < q \leq n$.

**OUTPUT:** An $n \times n$ permutation matrix $P$ such that every $r \times (g+1)$ leftmost submatrix $C_{g+1}$ of the matrix $WP$ for $g = r, \ldots, q - 1$ has been computed by means of greedy expansion of its preceding $r \times g$ leftmost submatrix $C_g$.

**INITIALIZATION:** Write $W = W_r := (C_r \| V_r)$.

**COMPUTATIONS:** Recursively, for $g = r,r+1,\ldots,q-1$, proceed as follows:
1. Pre-multiply the matrix $W_g$ by a $r \times r$ matrix $R_g$ that orthogonalizes the submatrix $C_g$.

2. Among the columns of the matrix $R_g V_g$, select a column vector $b_{g+1}$ having the maximal spectral norm.

3. Move this vector into the $(g + 1)$st position in the matrix $R_g W_g$, thus turning $R_g W_g$ into the matrix $W_{g+1} := R_g W_g P_g$ for a permutation matrix $P_g$. If $g = q - 1$, stop and output the permutation matrix $P := \prod_{g=r}^{q-1} P_g$.

Equation (69) implies that the $g$th loop of invocation of stages 1–3, for $g = r, \ldots, q - 1$, appends to the matrix $C_g$ a column that maximally increases its volume; then correctness of the algorithm follows.

The computational cost of performing the algorithm amounts to the cost of $q - r - 1$ orthogonalizations after $q - r - 1$ movements of columns; this involves $O(rn)$ flops after each movement (see [GL13, Section 6.5.2]), that is, $O((q - r - 1)rn)$ flops overall.

Now suppose that for a fixed $h \geq 1$ we seek a submatrix having a locally $h$-maximal volume among $r \times l$ submatrices of a $r \times n$ matrix $W$. Then, by applying Algorithms 133 and 134, we compute a greedy sequence of $l$ submatrices $C_g$, $g = 1, \ldots, l$, and check whether Corollary 128 or 131 together imply local $h$-optimality of the submatrix $C_l$. If not so, we can extend the greedy sequence by a fixed number $p$ matrices by applying Algorithms 133 and 134 and then contract it back to the length $l$ by reversing these algorithms. We can proceed for $p = 1$, if this does not work, then for $p = 2$, and so on.

We can reverse Algorithm 133 by applying (65) and reverse Algorithm 134 by applying equations (69) or (67) instead of equation (66). In the latter case, equations (65), (68) and (69) imply that at each cycle of expansion and contraction the volume of the input matrix $A$ cannot exceed that of the output matrix $A$.

Acknowledgements: Our research has been supported by NSF Grants CCF–1116736 and CCF–1563942 and PSC CUNY Award 68862–00 46. Victor Pan is also grateful to A. Osinsky and N. L. Zamarashkin for a preprint of the paper [O16] and ample verbal introduction to it, to S. A. Goreinov for reprints of his papers, to I. V. Oseledets and A. V. Knyazev for helpful pointers to the bibliography, to E. E. Tyrtyshnikov for both such pointers and the challenge of formally supporting fast empirical convergence of C-A algorithms and to V. Mehrmann for his advice for in depth study of the subject after the submission to LAA in the summer of 2016 of the first draft of this paper, containing our extension of superfast CUR LRA to dramatic acceleration of FMM and CG algorithms.

References

[A94] O. Axelsson, *Iterative Solution Methods*, Cambridge Univ. Press, Cambridge, England, 1994.

[ABBB99] Anderson, E.; Bai, Z.; Bischof, C.; Blackford, S.; Demmel, J.; Dongarra, J.; Du Croz, J.; Greenbaum, A.; Hammarling, S.; McKenney, A.; Sorensen, D., *LAPACK Users’ Guide* (Third ed.), Society for Industrial and Applied Mathematics, Philadelphia, PA. 1999, ISBN 0-89871-447-8

[ADM15] D. G. Anderson, S. S. Du, M. W. Mahoney, C. Melgaard, K. Wu, M. Gu, Spectral Gap Error Bounds for Improving CUR Matrix Decomposition and the Nyström
Method, Proc. of the 18th International Conference on Artificial Intelligence and Statistics, 1927, 2015.

[B41] A. C. Berry, The accuracy of the Gaussian approximation to the sum of independent variates, Transactions of the American Mathematical Society, 49, 1, 122–136, 1941.

[B00] M. Bebendorf, Approximation of boundary element matrices, Numer. Math., 86, 4, 565–589, 2000.

[B02] M. Benzi, Preconditioning Techniques for Large Linear Systems: a Survey, J. of Computational Physics, 182, 418–477, 2002.

[B10] S. Börm, Efficient Numerical Methods for Non-local Operators: $H^2$-Matrix Compression, Algorithms and Analysis, European Math. Society, 2010.

[B11] M. Bebendorf, Approximation of boundary element matrices, Numerische Mathematik, 86, 4, 565–589, 2000.

[B15] A. Björk, Numerical Methods in Matrix Computations, Springer, New York, 2015.

[BBC93] R. Barrett, M. W. Berry, T. F. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, H. van der Vorst, Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods, SIAM, Philadelphia, 1993.

[BDK15] H. Bouwmeester, A. Dougherty, A. Knyazev, Nonsymmetric preconditioning for conjugate gradient and steepest descent methods, International Conference On Computational Science (ICCS 2015), Procedia Computer Science, 51, 276–285, 2015. Also arXiv:1212.6680. Submitted on 30 Dec 2012, last revised 21 Jun 2013.

[BDM-I 14] C. Boutsidis, P. Drineas, M. Magdon-Ismail, Near-Optimal Column-Based Matrix Reconstruction, SIAM Journal on Computing, 43, 2, 687–717, 2014.

[BG65] P.A. Businger, G. H. Golub, Linear least squares solutions by Householder transformations, Numer. Math., 7, 269–276, 1965.

[BGH03] S. Börn, L. Grasedyck, W. Hackbusch, Introduction to Hierarchical Matrices with Applications, Engineering Analysis with Boundary Elements, 27, 5, 405–422, 2003.

[BGP05] A. Bini, L. Gemignani, V. Y. Pan, Fast and Stable QR Eigenvalue Algorithms for Generalized Semiseparable Matrices and Secular Equation, Numerische Mathematik, 100, 3, 373–408, 2005.

[B-I92] A. Ben-Israel, A volume associated with $m \times n$ matrices, Linear Algebra and its Applications, 167, 87–111, 1992.

[BKV15] M. Bebendorf, C. Kuske, R. Venn, Wideband nested cross approximation for Helmholtz problems, Numerische Mathematik, 130, 1, 1–34, 2015.

[BR03] M Bebendorf, S Rjasanow, Adaptive low-rank approximation of collocation matrices, Computing, 70 1, 1–24, 2003.

[BV88] W. Bruns, U. Vetter, Determinantal Rings, Lecture Notes in Math., 1327, Springer, Heidelberg, 1988.
[BW14] C. Boutsidis, D. Woodruff, Optimal CUR Matrix Decompositions, *Proceedings of ACM Symposium on Theory of Computing (STOC 2014)*, 353–362, ACM Press, New York, 2014.

[BW17] C. Boutsidis, D. Woodruff, Optimal CUR Matrix Decompositions, *SIAM Journal on Computing*, 46, 2, 543–589, 2017, DOI:10.1137/140977898. Also in arXiv:1405.7910

[BY13] L. A. Barba, R. Yokota, How Will the Fast Multipole Method Fare in Exascale Era? *SIAM News*, 46, 6, 1–3, July/August 2013.

[C46] Harald Cramér, *Mathematical Methods of Statistics*, Princeton University Press, 575 pages, 1999 (first edition 1946), ISBN: 9780691005478.

[C84] T.F. Chan, On the existence and computation of LU-factorizations with small pivots, *Math. of Computation*, 42, 535–547, 1984.

[C87] T.F. Chan, Rank revealing QR-Factorizations, *Lin. Algebra and Its Appls.*, 88/89, 67–82, 1987.

[C00] B. A. Cipra, The Best of the 20th Century: Editors Name Top 10 Algorithms, *SIAM News*, 33 4, 2, May 16, 2000.

[CBSW14] Y. Chen, S. Bhojanapalli, S. Sanghavi, R. Ward, Coherent Matrix Completion, *Proceedings of the 31st International Conference on Machine Learning*, PMLR 32, 1, 674–682, 2014.

[CC70] J. D. Carroll, J. J. Chang, Analysis of individual differences in multidimensional scaling via an N-way generalization of Eckart-Young decomposition, *Psychometrika*, 35, 283–319, 1970.

[CD05] Z. Chen, J. J. Dongarra, Condition Numbers of Gaussian Random Matrices, *SIAM. J. on Matrix Analysis and Applications*, 27, 603–620, 2005.

[CGR88] J. Carrier, L. Greengard, V. Rokhlin, A Fast Adaptive Algorithm for Particle Simulation, *SIAM Journal on Scientific Computing*, 9, 669–686, 1988.

[CGS07] S. Chandrasekaran, M. Gu, X. Sun, J. Xia, J. Zhu, A Superfast Algorithm for Toeplitz Systems of Linear Equations, *SIAM. J. on Matrix Analysis and Applications*, 29, 4, 1247–1266, 2007.

[CH90] T.F. Chan, P.C. Hansen, Computing truncated singular value decomposition least squares solutions by rank revealing QR-factorizations, *SIAM Journal on Scientific and Statistical Computing*, 11, 519 – 530, 1990.

[CH92] T.F. Chan, P.C. Hansen, Some applications of the rank revealing QR factorization, *SIAM Journal on Scientific and Statistical Computing*, 13, 727 – 741, 1992.

[CI94] S. Chandrasekaran, I. Ipsen, On rank revealing QR factorizations, *SIAM J. Matrix Anal. Appl.*, 15, 592–622, 1994.
A. Cichocki, D. Mandic, L. D. Lathauwer, G. Zhou, Q. Zhao, C. Caiafa, H. A. Phan, Tensor decompositions for signal processing applications: From two-way to multiway component analysis, *IEEE Signal Processing Magazine*, **32**, 2, 145–163, March 2015.

J. Demmel, The Probability That a Numerical Analysis Problem Is Difficult, *Math. of Computation*, **50**, 449–480, 1988.

*Dillon Engineering*, http://www.dilloneng.com/fft_ip/parallel-fft.

P. Drineas, R. Kannan, Pass efficient algorithms for approximating large matrices, *Proceedings of the 14th Annual ACM–SIAM Symposium on Discrete Algorithms*, pages 223–232, 2003.

P. Drineas, R. Kannan, M.W. Mahoney, Fast Monte Carlo algorithms for matrices II: Computing a low-rank approximation to a matrix, *SIAM Journal on Computing*, **36**, 158–183, 2006.

P. Drineas, M.W. Mahoney, S. Muthukrishnan, Relative-error CUR matrix decompositions, *SIAM Journal on Matrix Analysis and Applications*, **30**, 2, 844–881, 2008.

R. A. Demillo, R. J. Lipton, A Probabilistic Remark on Algebraic Program Testing, *Information Processing Letters*, **7**, 4, 193–195, 1978.

K. R. Davidson, S. J. Szarek, Local Operator Theory, Random Matrices, and Banach Spaces, in *Handbook on the Geometry of Banach Spaces* (W. B. Johnson and J. Lindenstrauss editors), pages 317–368, North Holland, Amsterdam, 2001.

A. Edelman, Eigenvalues and Condition Numbers of Random Matrices, *SIAM J. on Matrix Analysis and Applications*, **9**, 4, 543–560, 1988.

A. Edelman, Eigenvalues and condition numbers of random matrices, Ph.D. thesis, Massachusetts Institute of Technology, 1989.

Y. Eidelman, I. Gohberg, I. Haimovici, *Separable Type Representations of Matrices and Fast Algorithms*, volumes 1 and 2, Birkhäuser, 2013.

A. Edelman, B. D. Sutton, Tails of Condition Number Distributions, *SIAM J. on Matrix Analysis and Applications*, **27**, 2, 547–560, 2005.

A. C. Elliott, W. A. Woodward, *Statistical analysis quick reference guidebook: With SPSS examples*, Sage, 2007.

A. Frieze, R. Kannan, S. Vempala, Fast Monte-Carlo algorithms for finding low-rank approximations. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS 98)*, 370–378, IEEE Computer Soc. Press, 1998.

A. Frieze, R. Kannan, S. Vempala, Fast Monte-Carlo algorithms for finding low-rank approximations, *Journal of the ACM*, **51**, 6, 1025–1041, 2004.

F. R. Gantmacher, *Theory of matrices*, Chelsea, New York, 1959.
[G65] G. H. Golub, Numerical methods for solving linear least squares problem, Numerische Math., 7, 206–216, 1965.

[G97] A. Greenbaum, Iterative Methods for Solving Linear Systems, SIAM, Philadelphia, 1997.

[GE96] M. Gu, S.C. Eisenstat, An efficient algorithm for computing a strong rank revealing QR factorization, SIAM J. Sci. Comput., 17, 848–869, 1996.

[GH03] L. Grasedyck, W. Hackbusch, Construction and Arithmetics of H-Matrices, Computing, 70, 4, 295–334, 2003.

[GL13] G. H. Golub, C. F. Van Loan, Matrix Computations, The Johns Hopkins University Press, Baltimore, Maryland, 2013 (fourth edition).

[GOSTZ10] S. Goreinov, I. Oseledets, D. Savostyanov, E. Tyrtyshnikov, and N. Zamarashkin, How to find a good submatrix, in Matrix Methods: Theory, Algorithms, Applications, (dedicated to the Memory of Gene Golub, edited by V. Olshevsky and E. Tyrtyshnikov), pages 247–256, World Scientific Publishing, New Jersey, ISBN-13 978-981-283-601-4, ISBN-10-981-283-601-2, 2010.

[GR87] L. Greengard, V. Rokhlin, A Fast Algorithm for Particle Simulation, Journal of Computational Physics, 73, 325–348, 1987.

[GS12] V. Guruswami, A. K. Sinop, Optimal Column-Based Low-Rank Matrix Reconstruction Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2012), SIAM, Philadelphia, 12071214, 2012. Also arXiv:1104.1732, submitted 9 Apr 2011, last revised 4 Jan 2012.

[GT01] S. A. Goreinov, E. E. Tyrtyshnikov, The maximal-volume concept in approximation by low rank matrices, Contemporary Mathematics, 208, 47–51, 2001.

[GT11] S. A. Goreinov, E. E. Tyrtyshnikov, Quasioptimality of skeleton approximation of a matrix on the Chebyshev norm, Russian Academy of Sciences: Doklady, 83, 3, 1–2, 2011.

[GTZ97] S. A. Goreinov, E. E. Tyrtyshnikov, N. L. Zamarashkin, A Theory of Pseudo-skeleton Approximations, Linear Algebra and Its Applications, 261, 1–21, 1997.

[GTZ97a] S. A. Goreinov, E. E. Tyrtyshnikov, N. L. Zamarashkin, Pseudo-skeleton Approximations by Matrices of Maximal Volume, Mathematical Notes, 62, 4, 515–519, 1997.

[GZT95] S. A. Goreinov, N. L. Zamarashkin, E. E. Tyrtyshnikov, Pseudo-skeleton Approximations of Matrices, Russian Academy of Sciences: Doklady, 343, 2, 151–152, 1995.

[H70] R. A. Harshman, Foundations of the PARAFAC procedure: Models and conditions for an ”explanatory” multi-modal factor analysis, UCLA Working Papers in Phonetics, 16, 1–84, 1970.

Available at http://publish.uwo.ca/~harshman/wpppfac0.pdf.
[H94] P.C. Hansen, Regularization tools: A Matlab package for analysis and solution of discrete ill-posed problems. *Numerical algorithms*, 6.1, 1–35, 1994.

[HLY92] T.-S. Hwang, W.-W. Lin, E.K. Yang, Rank revealing LU factorizations, *Linear Algebra and Its Applications*, 175, 115–141, 1992.

[HMT11] N. Halko, P. G. Martinsson, J. A. Tropp, Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions, *SIAM Review*, 53, 2, 217–288, 2011.

[HP92] Y.P. Hong, C.-T. Pan, Rank-revealing QR factorizations and the singular value decomposition, *Math. of Computation*, 58, 213–232, 1992.

[K85] D. E. Knuth, Semi-optimal bases for linear dependencies, *Linear and Multilinear Algebra*, 17, 1, 1–4, 1985.

[KB09] T.G. Kolda, B.W. Bader, Tensor decompositions and applications, *SIAM Review*, 51, 455–500, 2009.

[KL07] A. V. Knyazev, I. Lashuk, Steepest descent and conjugate gradient methods with variable preconditioning, *SIAM J. Matrix Anal. Appl.*, 29, 4, 1267–1280, 2007.

[KPW16] M. Kapralov, V. Pothuru, D. Woodruff, How to fake multiply by a Gaussian matrix, Proceedings of the 33rd International Conference on Machine Learning, PMLR 48, 2101–2110, 2016, and https://arxiv.org/pdf/1606.05732.pdf, 2016.

[KS16] N. Kishore Kumar, J. Schneider, Literature survey on low rank approximation of matrices, arXiv:1606.06511v1 [math.NA] 21 June 2016.

[KV16] B. N. Khoromskij, A. Veit, Efficient computation of highly oscillatory integrals by using QTT tensor approximation, *Comp. Meth. Appl. Math.* 16, 1, 145–159, 2016. Also available at arXiv:1408.5224, 2014.

[LK00] B. Laurent, P. Massart, Adaptive Estimation of a Quadratic Functional by Model Selection, *The Annals of Statistics* 28, 5, 1302–1338, 2000. Also see http://www.jstor.org/stable/2674095

[LPSZZa] V. Y. Pan, Q. Luan, J. Svadlenka, V. Zaderman, L. Zhao, Superfast tensor decomposition: algorithms and complexity, to appear.

[M11] M. W. Mahoney, Randomized Algorithms for Matrices and Data, *Foundations and Trends in Machine Learning*, NOW Publishers, 3, 2, 2011. (Abridged version in: Advances in Machine Learning and Data Mining for Astronomy, edited by M. J. Way, et al., pp. 647-672, 2012.)

[M14] A. Y. Mikhailov, Method for the low rank Block Approximation of a Matrix from Its Entries, PhD Thesis, 105 pages, 2014.

[MD09] M. W. Mahoney, P. Drineas, CUR Matrix Decompositions for Improved Data Analysis, *Proc. Natl. Acad. Sci. USA*, 106, 697–702, 2009.

[MMD08] M.W. Mahoney, M. Maggioni, P. Drineas, Tensor-CUR decompositions for tensor-based data, *SIAM Journal on Matrix Analysis and Applications*, 30, 3, 957–987, 2008.
[MO15] A.Y. Mikhalev, I.V. Oseledets. Iterative representing set selection for nested cross-approximation, Numer. Linear Algebra Appl., 23, 2, 230–248, 2016. Also available at arxiv: 1309.1773.

[MRT05] P. G. Martinsson, V. Rokhlin, M. Tygert, A Fast Algorithm for the Inversion of General Toeplitz Matrices, Comput. Math. Appl., 50, 741–752, 2005.

[MRT06] P.-G. Martinsson, V. Rokhlin, M. Tygert. A randomized algorithm for the approximation of matrices, Applied and Computational Harmonic Analysis, 30, 1, 47–68, 2011. Also Technical Report YALEU/DCS/TR-1361, Yale University Department of Computer Science, New Haven, CT, June 2006.

[MZ08] A. Magen, A. Zousias, Near Optimal Dimensionality Reduction That Preserves Volumes, APPROX. and RANDOM(A. Goel et al., editors), LNCS, 5171, 623–534, 2008.

[O16] A.I. Osinsky, Error Estimates for Pseudo-skeleton Approximations of Matrices, preprint, 2016.

[OT10] I.V. Oseledets, E.E. Tyrtyshnikov, TT-cross approximation for multi-dimensional arrays, Linear Algebra and its Applications, 432, 70–88, 2010.

[P00] C.-T. Pan, On the existence and computation of rank-revealing LU factorizations, Linear Algebra and its Applications, 316, 199–222, 2000.

[P01] V. Y. Pan, Structured Matrices and Polynomials: Unified Superfast Algorithms, Birkhäuser/Springer, Boston/New York, 2001.

[P15] V. Y. Pan, Transformations of Matrix Structures Work Again, Linear Algebra and Its Applications, 465, 1–32, 2015.

[P17] V. Y. Pan, Fast Matrix Multiplication and Its Algebraic Neighborhood, Mathematical Sbornik, 2017.

[PLSZ16] V. Y. Pan, Q. Luan, J. Svadlenka, L. Zhao, Low Rank Approximation: New Insights, Accurate Superfast Algorithms, Pre-processing and Extensions, arXiv 1611.01391 (Submitted on 3 Nov, 2016).

[PW08] V. Y. Pan, X. Wang, Degeneration of Integer Matrices Modulo an Integer, Linear Algebra and Its Applications, 429, 2113–2130, 2008.

[PZ16] V. Y. Pan, L. Zhao, Low rank Approximation of a Matrix: Novel Insights, New Progress, and Extensions, Proc. of the Eleventh International Computer Science Symposium in Russia (CSR’2016), (Alexander Kulikov and Gerhard Woeginger, editors), St. Petersburg, Russia, June 2016, Lecture Notes in Computer Science (LNCS), 9691, 352–366, Springer International Publishing, Switzerland (2016). Also arXiv:1510.06142 [math.NA], submitted on 21 Oct 2015, revised June 2016.

[PZ17] V. Y. Pan, L. Zhao, New Studies of Randomized Augmentation and Additive Pre-processing, Linear Algebra Appl., 512, 256–305, 2017. Available at http://dx.doi.org/10.1016/j.laa.2016.09.035 and arXiv 1412.5864.
[PZ17a] V. Y. Pan, L. Zhao, Numerically Safe Gaussian Elimination with No Pivoting, *Linear Algebra and Its Applications*, **527**, 349–383, 2017, Available at http://dx.doi.org/10.1016/j.laa.2017.04.007 and arxiv 1501.05385

[RO15] M. V. Rakhuba, I. V. Oseledets, Fast multidimensional convolution in low-rank tensor formats via cross approximation, *SIAM J. Sci. Computing*, **37**, 2, A565–A582, 2015. Also available at arXiv:1402.5649.

[RV07] M. Rudelson, R. Vershynin, Sampling from large matrices: An approach through geometric functional analysis, *Journal of the ACM*, **54**, 4, Article number 1255449, 2007.

[RV09] M. Rudelson, R. Vershynin, The smallest singular value of a rectangular random matrix, *Comm. Pure Appl. Math.*, **62**, 12, 1707–1739, 2009.

[S80] J. T. Schwartz, Fast Probabilistic Algorithms for Verification of Polynomial Identities, *Journal of ACM*, **27**, 4, 701–717, 1980.

[S98] G. W. Stewart, *Matrix Algorithms, Vol I: Basic Decompositions*, SIAM, Philadelphia, 1998.

[S03] Y. Saad, *Iterative Methods for Sparse Linear Systems*, SIAM, Philadelphia, 2003.

[S06] T. Sarlos, Improved approximation algorithms for large matrices via random projections, *47th Annual IEEE Symposium Foundations of Computer Science (FOCS’06)*, 143–152, 2006.

[SST06] A. Sankar, D. Spielman, S.-H. Teng, Smoothed Analysis of the Condition Numbers and Growth Factors of Matrices, *SIAM J. on Matrix Analysis and Applics.*, **28**, 2, 446–476, 2006.

[SWZ17] Z. Song, D. P. Woodruff, P. Zhong, Low rank approximation with entrywise $l_1$-norm error, *Proc. STOC 2017*. Available at arXiv 1611.00898, November 2016.

[T96] E.E. Tyrtyshnikov, Mosaic-skeleton approximations, *Calcolo*, **33**, 1, 47–57, 1996.

[T00] E. Tyrtyshnikov, Incomplete Cross-Approximation in the Mosaic-Skeleton Method, *Computing*, **64**, 367–380, 2000.

[T11] J. A. Tropp, Improved analysis of the subsampled randomized Hadamard transform, *Adv. Adapt. Data Anal.*, **3**, 1–2, Special Issue ”Sparse Representation of Data and Images,” 115-126, 2011.

[TV10] Terence Tao, Van Vu, Random matrices: The distribution of the smallest singular values, *Geometric and Functional Analysis*, **20**, 1, 260–297, 2010.

[VL16] C. F. Van Loan, Structured Matrix Problems from Tensors, pages 1–63 in “Exploiting Hidden Structure in Matrix Computations: Algorithms and Applications”, M. Benzi and V. Simoncini (Eds.), C.I.M.E. Foundation Subseries, Springer, 2016.

[VVGM05] R. Vandebril, M. Van Barel, G. Golub, N. Mastronardi, A Bibliography on Semiseparable Matrices, *Calcolo*, **42**, 3–4, 249–270, 2005.
[VVM07/08] R. Vandebril, M. Van Barel, N. Mastronardi, *Matrix Computations and Semiseparable Matrices* (Volumes 1 and 2), The Johns Hopkins University Press, Baltimore, Maryland, 2007.

[VVVF10] M. Van Barel, R. Vandebril, P. Van Dooren, K. Frederix, Implicit Double Shift QR-algorithm for Companion Matrices, *Numerische Mathematik*, 116, 177–212, 2010.

[W14] D.P. Woodruff, Sketching As a Tool for Numerical Linear Algebra, *Foundations and Trends in Theoretical Computer Science*, 10, 1–2, 1–157, 2014.

[WZ13] S. Wang, Z. Zhang, Improving CUR matrix decomposition and the Nyström approximation via adaptive sampling, *Journal of Machine Learning Research*, 14, 1, 2729–2769, 2013.

[X12] J. Xia, On the Complexity of Some Hierarchical Structured Matrix Algorithms, *SIAM J. Matrix Anal. Appl.*, 33, 388–410, 2012.

[X13] J. Xia, Randomized Sparse Direct Solvers, *SIAM J. Matrix Anal. Appl.*, 34, 197–227, 2013.

[XXCB14] J. Xia, Y. Xi, S. Cauley, V. Balakrishnan, Superfast and Stable Structured Solvers for Toeplitz Least Squares via Randomized Sampling, *SIAM J. Matrix Anal. Appl.*, 35, 44–72, 2014.

[XXG12] J. Xia, Y. Xi, M. Gu, A Superfast Structured Solver for Toeplitz Linear Systems via Randomized Sampling, *SIAM J. Matrix Anal. Appl.*, 33, 837–858, 2012.

[Z79] R. E. Zippel, Probabilistic Algorithms for Sparse Polynomials, *Proceedings of EUROSAM’79, Lecture Notes in Computer Science*, 72, 216–226, Springer, Berlin, 1979.

[ZBD15] Z. Zhang, K. Batselier, H. Liu, L. Daniel, N. Wong, Tensor computation: A new framework for high-dimensional problems in EDA, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36, 4, 521–536, April 2017.