Refinement of Low Rank Approximation of a Matrix at Sub-linear Cost

Victor Y. Pan[1,2],[a] and Qi Luan[2],[b]

[1] Department of Computer Science
Lehman College of the City University of New York
Bronx, NY 10468 USA
[2] Ph.D. Programs in Computer Science and Mathematics
The Graduate Center of the City University of New York
New York, NY 10036 USA
[a] victor.pan@lehman.cuny.edu
http://comet.lehman.cuny.edu/vpan/
[b] qi_luan@yahoo.com

Abstract

Low rank approximation (LRA) of a matrix is a hot subject of modern computations. In application to Big Data mining and analysis the input matrices are so immense that one must apply sub-linear cost algorithms, which only access and process a tiny fraction of the input entries and which use much fewer flops[1] and memory cells than an input matrix has entries. Under this restriction the algorithms cannot compute accurate LRA of the worst case input matrix and even of the matrices of some specific small families in our Appendix, but we recently prove that such algorithms do this with a high probability (whp) for a random input, and this result is in good accordance with our tests and with more than a decade of worldwide computation of LRA at sub-linear cost by means of Cross–Approximation algorithms. A natural challenge is to complement such computation of LRA by its refinement at sub-linear cost, and we take that challenge and propose two algorithms for this task together with some recipes for a posteriori estimation of the errors of LRA at sub-linear cost.

Key Words: Low-rank approximation, Sub-linear cost, A posteriori error estimation, Iterative refinement

2000 Math. Subject Classification: 65Y20, 65F30, 68Q25, 68W20

1 Introduction

(a) Accurate LRA at sub-linear cost: the problem and our recent progress. Numerical Linear and Multilinear Algebra and Data Mining and Analysis, with applications ranging from machine learning theory and neural networks to term document data and DNA SNP data (see surveys [HMT11], [M11], and [KS16]). Matrices representing Big Data are usually so immense that

---

[1]Flop is a floating point arithmetic operation.

realistically one can only access and process a tiny fraction of their entries, but quite typically these matrices admit LRA, that is, are close to low rank matrices,[2] with which one can operate by using *sub-linear arithmetic time and memory space*, that is, much fewer flops and memory cells than the matrix has entries.

Such an LRA algorithm (running at sub-linear cost) fails on a worst case input and even on the small families of matrices of our Appendix A. In worldwide computational practice and in our extensive tests, however, some sub-linear cost algorithms consistently compute reasonably accurate LRA of matrices admitting LRA, and the papers [PLSZ16], [PLSZ17], [PLSZa], [PLSZb], and [LPSa] provide some formal support for this empirical observation.

In various applications of LRA it is highly important or sometimes imperative to obtain as close LRA as possible, and then again we are challenged to achieve this at sub-linear cost. In this paper we take the challenge and propose two techniques for iterative refinement at sub-linear cost given a crude but reasonably close initial LRA.

Our first approach naturally extends the known methods for iterative refinement as a popular tool for the solution and least squares solution of a linear system of equations (see [S98, Sections 3.3.4 and 4.2.5], [H02, Chapter 12 and Section 20.5], [DHK06], [GL13, Sections 3.5.3 and 5.3.8], [B15, Sections 1.4.6, 2.1.4, and 2.3.7]), also known for other matrix computations, including nonlinear ones (see [S98, page 223 and the references on page 225]), but to the best of our knowledge, it has not been applied to LRA so far, possibly because computation of LRA at sub-linear cost has not been studied as much as it deserves.

Our second approach is dramatically different. It relies on recursive application of randomized LRA algorithms of [DMM08] by means of random sampling directed by the sampling probabilities, called *leverage scores*. They are computed from the top right singular space of an input matrix, but the computation allows some leverage, and so one can expect that these scores computed for a given LRA could define an even closer LRA for an input matrix.

All known algorithms for the computation of LRA by means of subspace sampling directed by leverage scores sample a large number of rows and columns of an input matrix, and so does our second algorithm for iterative refinement. This may restrict practical value of these algorithms (see [TYUC17, Section 1.7.3]), although our Algorithm 3.1 provides partial remedy at sub-linear cost.

In order to develop our refinement algorithms we supply some recipes for a posteriori error estimation of LRA at sub-linear cost; this can be of some independent value.

Our numerical tests are in good accordance with our formal study.

Our work seems to be the first step into highly important but long-ignored research area, and we hope to motivate further efforts into this direction.

Organization of our paper. In the next section we recall some background material. In Section 3 we transform any LRA into its SVD at sub-linear cost. We propose some algorithms for iterative refinement of an LRA at sub-linear cost in Sections 4 and 6 and study the problem of error estimation for LRA at sub-linear cost in Section 5. We devote Section 7, the contribution of the second author, to numerical tests. In Appendix A we specify some small families of matrices on which every LRA algorithm fails if it runs at sub-linear cost, even though every matrix of these families admits LRA. In Appendix B we recall the known estimates for the output errors of LRA at sub-linear cost in a very special but important case where an input matrix is filled with i.i.d. values of a single random variable.

---

[2]Here and throughout we use such concepts as "low", "small", "nearby", etc. defined in context.

2 Some background for LRA

Definition 2.1. *(i) An $m \times n$ matrix $\tilde{M}$ has rank at most ρ, $\mathrm{rank}(\tilde{M}) \leq \rho$, if*

$$\tilde{M} = AB, \ A \in \mathbb{C}^{m \times \rho}, \ \text{and} \ B \in \mathbb{C}^{\rho \times n}. \tag{2.1}$$

(ii) Such a matrix M has ξ-rank at most ρ for a fixed tolerance ξ if M admits its approximation within an error norm ξ by a matrix $\tilde{M}$ of rank at most ρ or equivalently if there exist three matrices A, B and E such that

$$M = \tilde{M} + E \ \text{where} \ |E| \leq \xi, \ \tilde{M} = AB, \ A \in \mathbb{C}^{m \times \rho}, \ \text{and} \ B \in \mathbb{C}^{\rho \times n}. \tag{2.2}$$

Here and hereafter $|\cdot|$ unifies definitions of the spectral norm $||\cdot||$ and the Frobenius norm $||\cdot||_F$.

The 0-rank is the rank; the ξ-rank of a matrix M for a small tolerance ξ is said to be its *numerical rank*, hereafter denoted $\mathrm{nrank}(M)$. A matrix admits its close approximation by a matrix of rank at most ρ if and only if it has numerical rank at most ρ.

A 2-factor LRA AB of M of (2.2) can be generalized to a 3-factor LRA

$$M = \tilde{M} + E, \ |E| \leq \xi, \ \tilde{M} = XTY, \ X \in \mathbb{C}^{m \times k}, \ T \in \mathbb{C}^{k \times l}, \ Y \in \mathbb{C}^{l \times n}, \tag{2.3}$$

$$\rho \leq k \leq m, \ \rho \leq l \leq n \tag{2.4}$$

for $\rho = \mathrm{rank}(\tilde{M})$.

Remark 2.1. *The pairs of the maps $XT \to A$ and $Y \to B$ as well as $X \to A$ and $TY \to B$ turn a 3-factor LRA XTY of (2.3) into a 2-factor LRA AB of (2.2).*

An important 3-factor LRA is the ρ-*top SVD* $M_\rho = U_\rho \Sigma_\rho V_\rho^*$ where Σ_ρ is the diagonal matrix of the ρ top (largest) singular values of M and U_ρ and V_ρ are the unitary (orthogonal) matrices of the associated top singular vectors of M. M_ρ is said to be the ρ-*truncation* of M, obtained from M by setting to zero all its singular values but the ρ largest ones.

The ρ-top SVD defines an optimal 3-factor LRA under both spectral and Frobenius norms.

Theorem 2.1. *[GL13, Theorem 2.4.8]. Write $\tau_{\rho+1}(M) := |M_\rho - M| = \min_{N: \ \mathrm{rank}(N)=\rho} |M - N|$. Then $\tau_{\rho+1}(M) = \sigma_{\rho+1}(M)$ under the spectral norm $|\cdot| = ||\cdot||$ and $\tau_{\rho+1}(M) = \sigma_{F,\rho+1}(M) := \sum_{j \geq \rho} \sigma_j^2(M)$ under the Frobenius norm $|\cdot| = ||\cdot||_F$.*

Lemma 2.1. *[GL13, Corollary 8.6.2]. For $m \geq n$ and a pair of $m \times n$ matrices M and $M + E$ it holds that*

$$|\sigma_j(M + E) - \sigma_j(M)| \leq ||E|| \ \text{for} \ j = 1, \dots, n.$$

Lemma 2.2. *[The norm of the pseudo inverse of a matrix product.] Suppose that $A \in \mathbb{R}^{k \times r}$, $B \in \mathbb{R}^{r \times l}$ and the matrices A and B have full rank $r \leq \min\{k, l\}$. Then $|(AB)^+| \leq |A^+| \, |B^+|$.*

3 Transform of LRA into its SVD at sub-linear cost

Algorithm 3.1. [Transform of any LRA into its SVD.]

INPUT: *Three matrices $A \in \mathbb{R}^{m \times k}$, $B \in \mathbb{R}^{k \times n}$, and $M' = AB \in \mathbb{R}^{m \times n}$.*

OUTPUT: *$\rho = \mathrm{rank}(M')$ and three matrices $U' \in \mathbb{R}^{m \times \rho}$ (orthogonal), $\Sigma' \in \mathbb{R}^{\rho \times \rho}$ (diagonal), and $V'^* \in \mathbb{R}^{\rho \times n}$ (orthogonal) such that $M'_\rho = U' \Sigma' V'^*$.*

COMPUTATIONS: 1. *Compute the matrices $Q_A \in \mathbb{C}^{m \times k}$, $Q_B \in \mathbb{C}^{n \times k}$, $R_A \in \mathbb{C}^{k \times k}$, and $R_B \in$ $\mathbb{C}^{k \times k}$ being Q and R factors in thin QR factorization $A = Q_A R_A$ and $B = Q_B R_B$ or (in order to strengthen numerical stability) being Q and $R\Pi$ factors in rank-revealing $QR\Pi$ factorization $A = Q_A R_A \Pi_A$ and $B = Q_B R_B \Pi_B$.*

2. *Compute the $k \times k$ matrix $W = R_A R_B^*$.*

3. *Compute its full SVD $W = U_W \Sigma_W V_W^*$ where $\Sigma = \Sigma_W = \mathrm{diag}(\Sigma, O)$ and Σ is a nonsingular $\rho \times \rho$ diagonal matrix..*

4. *Compute the matrices U' and V' by keeping just first ρ columns of the matrices $Q_A U_W$, $Q_B V_W$, respectively, and deleting the other columns. Output $\rho = \mathrm{rank}(\Sigma)$ and the matrices U', $\Sigma' = \Sigma_\rho$, and V'^*.*

The algorithm runs at sub-linear cost if $k^2 \ll \min\{m, n\}$, and its correctness is readily verified.

The transition $AB \to (AB)_\rho$ decreases the rank of LRA to at most ρ at the price of only a mild increase of the error norm bound. Indeed triangle inequality implies that

$$|M - M'_\rho| \le |M - M'| + |M' - M'_\rho| \text{ and } |M' - M'_\rho| = \tau_\rho(M') \le |M - M'| + \tau_\rho(M) \qquad (3.1)$$

for any matrix M (cf. [TYUC17, Proposition 6.1]).

By using Remark 2.1 we can readily transform any 3-factor LRA $M' = XTY$ into a 2-factor LRA; then by applying Algorithm 3.1 we can transform it further to its SVD.

We can a little simplify the transformation of a 3-factor LRA into a 2-factor LRA as follows:

(i) compute compact SVD $T = U_T \Sigma_T V_T^*$ where Σ_T is a nonsingular $\rho' \times \rho'$ diagonal matrix for $\rho' = \mathrm{rank}(T)$,

(ii) obtain two matrices X' and Y' by keeping the first ρ' columns of the matrices $X U_T'$ and $Y^* V_T'$, respectively, and deleting the other columns, and

(iii) finally apply the recipes of Remark 2.1 to the 3-factor LRA $M' = X' \Sigma_T Y'^*$.

4   Deterministic iterative refinement of an LRA at sub-linear cost

Next we propose an algorithm for iterative refinement of a sufficiently close LRA of M (see Appendix 6 for an alternative algorithm.) At the ith step we try to improve the current LRA $M^{(i)}$ by applying a fixed LRA algorithm to the current error matrix $E^{(i)} = M^{(i)} - M$ or to the matrix of its modification $E^{(i)} - E^{(i-1)}$ (see Remark 4.3). At every iteration the rank of the new tentative LRA is at least doubled, but we periodically cut it back to the value $\mathrm{nrank}(M)$ (see Remark 4.2).

Algorithm 4.1. (Iterative refinement of a CUR LRA at sub-linear cost. See Remarks 4.2 and 4.3.)

INPUT: *Three integers m, n, and ρ, $\rho \le \min\{m, n\}$, an $m \times n$ matrix M of numerical rank ρ, a Sub-algorithm APPROX(r), which for a fixed positive integer r computes a rank-r approximation of its input matrix at sub-linear cost, and a Stopping Criterion, which signals when current candidate LRA is expected to be satisfactory (see the next section).*

INITIALIZATION: $M^{(0)} = O^{m \times n}$.

COMPUTATIONS: *Recursively for $i = 1, 2, \ldots$ do:*

1. *Apply Sub-algorithms APPROX(r_{i-1}) to the matrix $E^{(i-1)} = M - M^{(i-1)}$ for $r_{i-1} = 2^{i-1}\rho$.*

2. *Let $\Delta^{(i-1)}$ denote its output matrix of rank at most r_{i-1}. Compute a new approximation $M^{(i)} = M^{(i-1)} + \Delta^{(i-1)}$ of M and the matrix $E^{(i)} = M - M^{(i)}$, of numerical rank at most $2^i \rho$.*

3. *Replace i by $i+1$ and repeat stages 1 and 2 until either i exceeds the allowed limitation, and then stop and output FAILURE, or Stopping Criterion is satisfied for some integer $i = t - 1$, and then stop and output the matrix $\tilde{M} = M^{(t)}$.*

Remark 4.1. Progress in refinement. *Write $e_i = |E^{(i)}|$ for all i and observe that $E^{(i)} = E^{(i-1)} - \Delta^{(i-1)}$, and so $e_i < e_{i-1}$ if $\Delta^{(i-1)}$ approximates $E^{(i-1)}$ closer than the matrix O filled with 0s.*

Remark 4.2. Management of the rank growth. *The bound on the rank of the matrices $E^{(i)}$ is doubled at every iteration of Algorithm 4.1; by allowing its further increase we can obtain more accurate LRA. Fast growth of the rank, however, implies fast growth of the complexity of an iteration, and so we should periodically (possibly for every i) compress the computed LRA $M^{(i)}$ into its ρ-truncation $M_\rho^{(i)}$ by applying Algorithm 3.1. The error of LRA would grow in compression within bound (3.1), that is, less significantly if an LRA $M^{(i)}$ is close to an input matrix M.*

Remark 4.3. Management of the precision of computing. *As usual in iterative refinement we should apply the mixed precision technique, that is, perform the subtraction stage 2 with a higher precision than stage 1. Furthermore we can replace the summation $M^{(i)} = M^{(i-1)} + \Delta^{(i-1)}$ by the summation of more than two terms, by representing the matrix $M^{(i-1)} == M^{(i-2)} + \Delta^{(i-2)}$ by a pair of matrices $M^{(i-2)}$ and $\Delta^{(i-2)}$ and possibly similarly representing the matrix $M^{(i-2)}$. The basic observation is that we need lower precision in order to represent each term than their sums.*

5 Heuristic a posteriori error estimation for LRA at sub-linear cost

Accurate a posteriori error estimation is impossible at sub-linear cost even for the small input families of Appendix A, but next we discuss some heuristic recipes for this task.

(i) Clearly the value $|e_{i,j}|$ for every entry $e_{i,j}$ of the LRA error matrix $E = (e_{i,j})_{i,j} = \tilde{M} - M$ is a lower bound on the norm $|E|$ of this matrix. One can compute s entries of E by using at most $2\rho s$ arithmetic operations; this cost is sub-linear if $2\rho s \ll mn$. This deterministic lower bound on the LRA error norm $|E|$ also implies its a posteriori randomized upper bounds if, say, the error matrix E is filled with i.i.d. values of a single random variable and has sufficiently many entries, e.g., 100 entries or more (see Appendix B).

(ii) By generalizing this technique we obtain deterministic lower bounds on the error norm $|E|$ from the norms $|FE|$, $|EH|$ or $|FEH|$ for any pair of matrices $F \in \mathbb{R}^{k \times m}$ and $H \in \mathbb{R}^{n \times l}$. The computation also defines randomized upper bounds on the error norm $|FEH|$ for random matrices F and/or H and sufficiently large integers k and/or l (see [F79]) and runs at sub-linear cost if $k \ll m$, $l \ll n$ and if the matrices F and H are sufficiently sparse, e.g., are sub-permutation matrices.

(iii) Suppose that we have computed a set of LRA $\tilde{M}_1, \ldots, \tilde{M}_s$ for a matrix M, by applying to it various LRA algorithms running at sub-linear cost or the same algorithm with distinct parameters, e.g., the algorithm of [TYUC17] with sparse multipliers F and H. Furthermore suppose that we have computed a median $\tilde{M} = \tilde{M}_j$ for some j in the range $1 \le j \le s$. Then we can consider this median a heuristic LRA and the norms $|\tilde{M}_i - \tilde{M}|$ for $i = 1, \ldots, s$ heuristic error bounds.

For heuristic computation of a median at sub-linear cost one may select the median index $j = j(\mathbf{f}, \mathbf{h})$ for the product $\mathbf{f}^* \tilde{M}_j \mathbf{h}$ over all subscripts j in the range $1 \le j \le s$ for a pair of fixed

common vectors **f** and **h**. For more dependable selection one may first compute such medians for a number of pairs of vectors **f** and **h** and then select a median of these medians.

(iv) In Appendix B we recall the well-known techniques for a posteriori error estimation at sub-linear cost for LRA of a matrix filled with i.i.d. values of a single variable.

All these recipes for a posteriori error estimation at sub-linear cost apply to any LRA. In [PLSZa] and [PLSZb] we deduce such estimates for LRA output by some specific sub-linear cost algorithms. We refer the reader to [B00], [BR03], and [BG06] for such recipe in the case of Adaptive Cross-Approximation iterations for LRA.

6 Randomized iterative refinement of LRA at sub-linear cost by means of refinement of leverage scores

Given a crude but reasonably close LRA $\tilde{M} = ATB$ such that $||\tilde{M} - M|| \ll \sigma_\rho(M)$ we can try to refine it at sub-linear cost by extending the random subspace sampling algorithms of [DMM08] by Drineas et al. and of various subsequent papers (see [KS16], [BW17], and the references therein). These algorithms compute CUR LRA of a matrix M at sub-linear cost if its top SVD is available. Namely, given the top SVD the algorithm computes at sub-linear cost the so called *leverage scores*, which serve a sampling probabilities that define random subspace sampling at sub-linear cost.

Given ρ-top SVD $M_\rho = U_\rho \Sigma_\rho V_\rho^*$, with $V_\rho = (\mathbf{v}_j^{(\rho)})_{j=1}^n$ one can fix any real β, $0 < \beta \leq 1$, and then define the *SVD-based leverage scores* $p_1, \ldots, p_n$ satisfying

$$p_j \geq (\beta/r)||\mathbf{v}_j^{(r)}||^2 \text{ and } \sum_{j=1}^n p_j = 1 \text{ for } j = 1, \ldots, n. \tag{6.1}$$

(6.1) turns into $p_j = ||\mathbf{v}_j^{(r)}||^2/r$ for $j = 1, \ldots, n$ for $\beta = 1$, but for $\beta < 1$ we obtain leverage scores $p_1, \ldots, p_n$ that we can *reuse in a small neighborhood of an input matrix*.

Remark 6.1. *The leverage scores are defined by the top right singular space but do not depend on the choice of its orthogonal basis; they stay invariant if the matrix V_ρ^* is pre-multiplied by an orthogonal matrix. By virtue of Theorem 6.1 below, they change little in a small perturbation of that space.*

We next recall a bound on the perturbation of singular spaces of the ρ-truncation M_ρ of a matrix M into those of a matrix $M + E$. The bound is stated in terms of the *minimal perturbation of unitary or orthogonal bases* for the transition from the singular spaces of M to those of $M + E$. Compare similar results by Davis–Kahan 1970 and Wedin 1972 and 1973 involving angles between singular spaces.

Theorem 6.1. (The impact of a perturbation of a matrix on its top singular spaces. [GL13, Theorem 8.6.5].)[3] *Suppose that*

$$g =: \sigma_\rho(M) - \sigma_{\rho+1}(M) > 0 \text{ and } ||E||_F \leq 0.2g.$$

---

[3][GL13, Theorem 8.6.5] cites [S73, Theorem 6.4], which claims a slightly more optimistic bound

$$\max\{||B_{\rho,\text{left}}(M + E) - B_{\rho,\text{left}}(M)||_F, ||B_{\rho,\text{right}}(M + E) - B_{\rho,\text{right}}(M)||_F\} \leq 2\frac{||E||_F}{g} \text{ if } ||E||_F \leq \delta/2.$$

Then there exist unitary or orthogonal matrix bases[4] $B_{\rho,\text{left}}(M)$, $B_{\rho,\text{right}}(M)$, $B_{\rho,\text{left}}(M+E)$, *and* $B_{\rho,\text{right}}(M+E)$ *for the left and right top singular spaces (associated with the r largest singular values) of the matrices M and $M+E$ such that*

$$\max\{||B_{\rho,\text{left}}(M+E) - B_{\rho,\text{left}}(M)||_F, ||B_{\rho,\text{right}}(M+E) - B_{\rho,\text{right}}(M)||_F\} \leq 4\frac{||E||_F}{g}.$$

For example, if $\sigma_\rho(M) \geq 2\sigma_{\rho+1}(M)$, which implies that $g \geq 0.5 \ \sigma_\rho(M)$, and if $||E||_F \leq 0.1 \ \sigma_\rho(M)$, then the upper bound on the right-hand side is approximately $8||E||_F/\sigma_\rho(M)$.

The above observations and estimates combined lead us to a heuristic recipe for iterative refinement of LRA by means of updating the leverage scores of an input matrix.

Namely suppose that an LRA $\tilde{M}$ is sufficiently close to the matrix M. Then by virtue of Theorem 6.1 the top right singular spaces of the matrix M and of its LRA $\tilde{M}$ are close to one another, and so the sets of their leverage scores are close to one another whp [DMM08]. At this point we propose to fix $\beta < 1$, to compute the leverage scores of the LRA at sub-linear cost, and then to apply the algorithm of [DMM08] to the matrix M by using these leverage scores. For a proper choice $\beta < 1$ the new LRA should improve $\tilde{M}$ whp if the norm $|E| = |M - \tilde{M}|$ is small enough. Recursively we arrive at a heuristic recipe for randomized iterative refinement of a crude but sufficiently close LRA.

We leave as research challenges (i) the relevant estimates for an upper bound on the norm $|E|$ that would imply its decrease, say, by twice with a probability at least $1/2$ and (ii) numerical tests that would support these estimates.

It is not clear how large should be samples of columns and rows of M in the tests. Large overhead constants are involved in the known upper estimates for the numbers of row and column samples in the subspace sampling algorithms of [DMM08] and all subsequent papers using leverage scores, but the tests in [DMM08] have succeeded with much fewer samples.

Remark 6.2. *The randomized algorithms of [DMM08] compute nearly optimal LRA at nearly optimal cost whp provided that they sample fairly many rows and columns of an input matrix according to leverage scores of (6.1). [TYUC17, Section 1.73] discusses deficiency of LRA based on sampling many rows and columns; our Algorithm 3.1 provides some partial remedy.*

Remark 6.3. *One can refine a crude basis for ρ-top right singular subspace of an $m \times n$ matrix M' by applying orthogonal iterations (cf. [GL13, Sections 8.2.3 and 8.3.7]), but each such an iteration involves $(m + \rho)n$ memory cells and $O(mn\rho)$ flops, so it does not run at sub-linear cost.*

7 Numerical Experiments for Algorithm 4.1

In this subsection we present the test results for Algorithm 4.1 on inputs of four types made up of synthetic and real-world data with various spectra. Our figures display the relative error ratio

$$r = \frac{||M - \tilde{M}||_F}{||M - M_\rho||_F} \tag{7.1}$$

where M denotes the input matrix, $\tilde{M}$ denotes its approximation output by Algorithm 4.1, M_ρ denotes the ρ-top SVD of M, and ρ is a rank value pre-determined for each input matrix. Unless the output of the algorithm was corrupted by rounding errors, the ratio r was not supposed to be noticeably exceeded by 1, and it was not in our experiments.

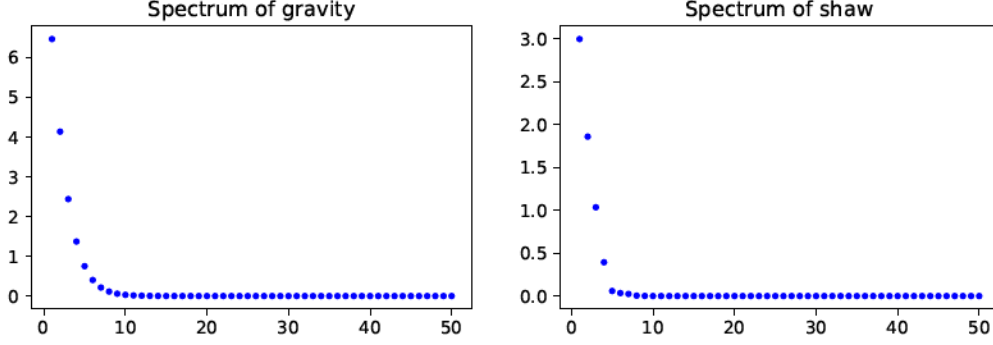[4]Recall that such bases are not unique even where all singular values are distinct.

Figure 1: Spectrums of real world input matrices

The algorithm was implemented in Python, and we run all experiments on a 64bit MacOS Sierra 10.12.6 machine with 1.6GHz CPU and 4GB Memory. We called scipy.linalg version 0.4.9 for numerical linear algebra routines such as QR factorization with pivoting, Moore-Penrose matrix inversion, and linear least squares regression.

Synthetic Input: We used random synthetic 1024×1024 input matrices of two kinds – with rapidly and slowly decaying spectra. In both types we generated these matrices as products $U\Sigma V^T$, where U and V were the matrices of the left and right singular vectors of a random Gaussian matrix. In the case of rapidly decaying spectrum we let $\Sigma = \text{diag}(v)$, where $v_i = 1$ for $i = 1, 2, 3, \ldots 40$, $v_i = \frac{1}{2}^i$ for $i = 41, \ldots, 100$, and $v_i = 0$ for $i > 100$. In the case of slowly decaying spectrum we let $\Sigma = \text{diag}(u)$, where $u_i = 1$ for $i = 1, 2, 3, \ldots 40$, and $u_i = \frac{1}{1+i}$ for $i > 40$.

Real-wold Input: The input matrices of this category were 1000×1000 dense matrices with real values having low numerical rank. They represent discretization of Integral Equations, provided in the built-in problems of the Regularization Tools[5]. We used the two test matrices. One of them, called **gravity**, came from a one-dimensional gravity surveying model problem and another one, called **shaw**, came from a one-dimensional image restoration model problem. Their distribution of singular values is displayed in figure 1; we padded these matrices with 0s in order to increase their size to 1024×1024.

Sub-algorithm: Our sub-algorithm modifies the sketching approach of [TYUC17]. Namely, in the ith iteration step we drew two multipliers F and H, then approximated the residual $R_i = M - \tilde{M}_{i-1}$ by $\tilde{R}_i = R_i H (F R_i H)^+ F R_i$, and finally computed the ith approximation $\tilde{M}_i = \tilde{M}_{i-1} + \tilde{R}_i$. In our tests, we used the *abridged* Hadamard multipliers defined in [PLSZ16] and [PLSZ17] and having size 5×1024 and recursion depth 3. We also included results of similar tests with Gaussian multipliers for comparison (cf. [TYUC17]).

For each input matrix, we performed 100 times the iterative refinement algorithm 4.1 and recorded the mean relative error ratio for every iteration step in figure 2.

We notice that the abridged SRHT multipliers performed similarly to Gaussian multipliers in our tests, and are only slightly worse in few places. This minor deterioration of the output accuracy was a reasonable price for using abridged (very sparse) Hadamard multipliers, with which we only access a small fraction of the input matrix at each iteration step.

It is worth noticing that even in the test with some random input matrices that have slowly decaying spectrum the relative error ratio did not decrease below 1, which could be caused by the

---

[5]See http://www.math.sjsu.edu/singular/matrices and http://www2.imm.dtu.dk/~pch/Regutools
For more details see Ch4 of http://www.imm.dtu.dk/~pcha/Regutools/RTv4manual.pdf
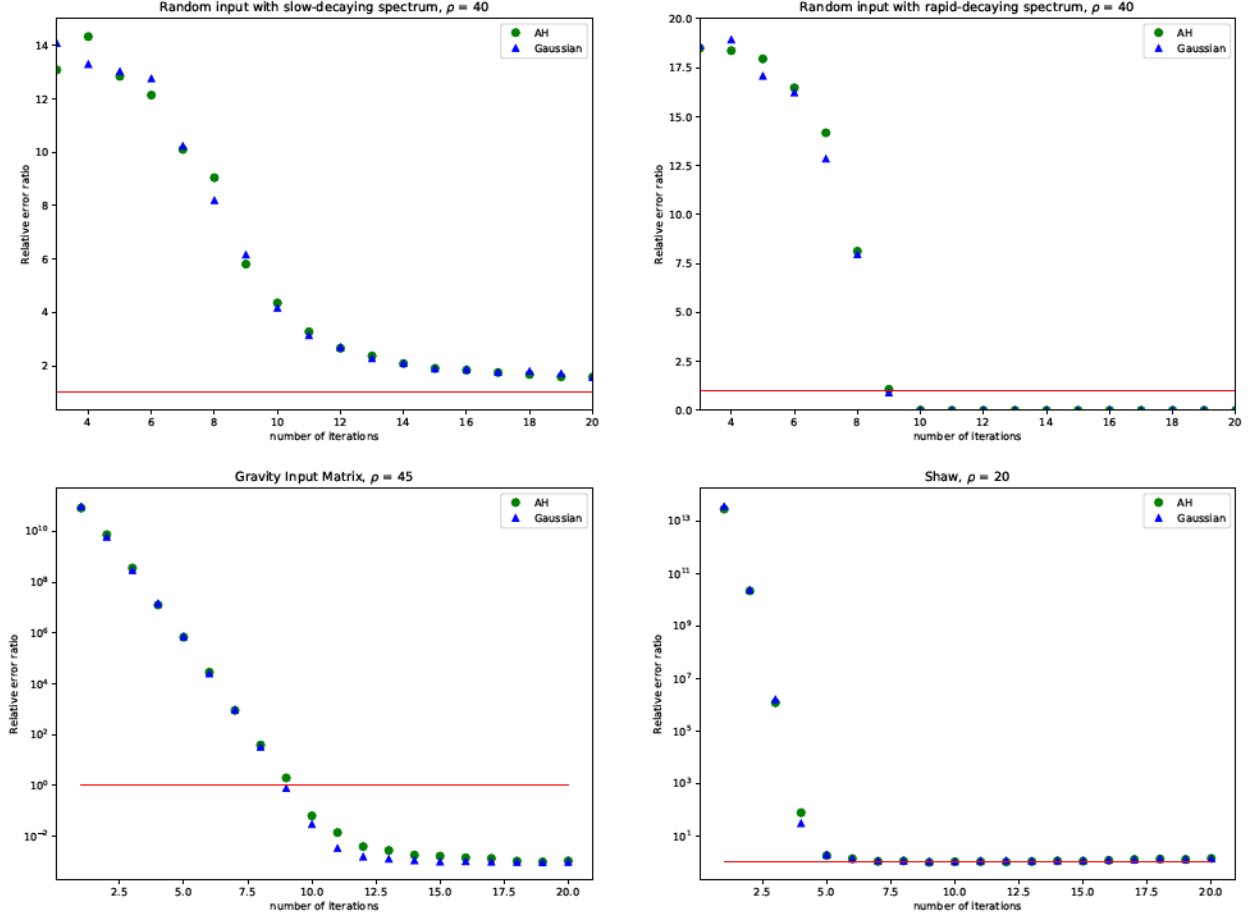
8

Figure 2: Test result for Algorithm 4.1

"heavy" tail in the spectrum. In our tests with some other inputs which were not "well-mixed" in some sense, it was necessary to increase the recursion depth of the abridged Hadamard multipliers in order to bring the relative error ratio close to 1.

Appendix

A   Small families of hard inputs for LRA at sub-linear cost

Any sub-linear cost LRA algorithm fails on the following small families of LRA inputs.

Example A.1. *Define the following family of $m \times n$ matrices of rank 1 (we call them δ-matrices):* $\{\Delta_{i,j}, \ i = 1, \ldots, m; \ j = 1, \ldots, n\}$. *Also include the $m \times n$ null matrix $O_{m,n}$ into this family. Now fix any LRA algorithm running sub-linear cost; it does not access the (i,j)th entry of its input matrices for some pair of i and j. Therefore it outputs the same approximation of the matrices $\Delta_{i,j}$ and $O_{m,n}$, with an undetected error at least $1/2$. Apply the same argument to the set of $mn + 1$ small-norm perturbations of the matrices of the above family and to the $mn + 1$ sums of the latter matrices with any fixed $m \times n$ matrix of low rank. Finally, the same argument shows that a posteriori output errors of an LRA algorithm applied to the same input families cannot be estimated at sub-linear cost.*

9

B A posteriori error estimation sub-linear cost for LRA of a matrix filled with i.i.d. values of a single variable

In our randomized a posteriori error estimation sub-linear cost below 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-ρ 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 $qs = O((m+n)kl)$ and $qs \ll mn$; 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 μ and a variance σ^2 and that we have computed the observed average value and variance

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

respectively. Then, for a fixed reasonably large K, both

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

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

References

[B00]   M. Bebendorf, Approximation of Boundary Element Matrices, *Numer. Math.*, **86, 4**, 565–589, 2000.

[B15]   A. Björk, *Numerical Methods in Matrix Computations*, Springer, New York, 2015.

[BG06]   M. Bebendorf, R. Grzhibovskis, Accelerating Galerkin BEM for linear elasticity using adaptive cross approximation, *Math. Methods Appl. Sci.*, **29**, 1721-1747, 2006.

[BR03]   M. Bebendorf, S. Rjasanow, Adaptive Low-Rank Approximation of Collocation Matrices, *Computing*, **70, 1**, 1–24, 2003.

[BW17]   C. Boutsidis, D. Woodruff, Optimal CUR Matrix Decompositions, *SIAM Journal on Computing*, **46, 2**, 543–589, 2017, DOI:10.1137/140977898.

[C46]   Harald Cramér, *Mathematical Methods of Statistics*, Princeton University Press, 575 pages, 1999 (first edition 1946).

[DHK06]   J. Demmel, Y. Hida, W. Kahan, X. S. Li, S. Mukherjee, E. J. Riedy, Error bounds from extra-precise iterative refinement, *ACM Trans. Math. Softw.*, **32, 2,** 325–351, 2006.

[DMM08]   P. Drineas, M.W. Mahoney, S. Muthukrishnan, Relative-error CUR Matrix Decompositions, *SIAM Journal on Matrix Analysis and Applications*, **30, 2**, 844–881, 2008.

[EW07]   A. C. Elliott, W. A. Woodward, *Statistical Analysis Quick Reference Guidebook: With SPSS Examples*, Sage, 2007.

[F79]   R. Freivalds, Fast probabilistic algorithms, *Mathematical Foundations of Computer Science*, 57–69, *Lecture Notes in Comp. Sci.*, **74**, Springer, Berlin, Heidelberg, 1979.

[GL13]   G. H. Golub, C. F. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, Maryland, 2013 (fourth edition).

[H02]   N. J. Higham, *Accuracy and Stability in Numerical Analysis*, SIAM, Philadelphia, 2002 (second edition).

[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.

[KS16]   N. Kishore Kumar, J. Schneider, Literature Survey on Low Rank Approximation of Matrices, arXiv:1606.06511v1 [math.NA] 21 June 2016.

[LPSa]   Q. Luan, V. Y. Pan, J. Svadlenka, Low Rank Approximation Directed by Leverage Scores and Computed at Sub-linear Cost, preprint, 2019.

[M11]   M. W. Mahoney, Randomized Algorithms for Matrices and Data, *Foundations and Trends in Machine Learning*, NOW Publishers, **3, 2**, 2011.

[PLSZ16]   V. Y. Pan, Q. Luan, J. Svadlenka, L.Zhao, Primitive and Cynical Low Rank Approximation, Preprocessing and Extensions, arXiv 1611.01391 (November 2016).

[PLSZ17]   V. Y. Pan, Q. Luan, J. Svadlenka, L. Zhao, Superfast Accurate Low Rank Approximation, preprint, arXiv:1710.07946 (October 2017).

[PLSZa]   V. Y. Pan, Q. Luan, J. Svadlenka, L. Zhao, CUR Low Rank Approximation at Sublinear Cost, preprint, 2019.

[PLSZb]   V. Y. Pan, Q. Luan, J. Svadlenka, L. Zhao, Low Rank Approximation at Sub-linear Cost by Means of Subspace Sampling, preprint, 2019.

[S73]   G. W. Stewart, Error and Perturbation Bounds for Subspaces Associated with Certain Eigenvalue Problems, *SIAM Review*, **15, 4**, 727–764, 1973.

[S98]   G. W. Stewart, *Matrix Algorithms, Vol. **I**: Basic Decompositions*, SIAM, 1998.

[TYUC17]   J. A. Tropp, A. Yurtsever, M. Udell, V. Cevher, Practical Sketching Algorithms for Low-rank Matrix Approximation, *SIAM J. Matrix Anal.*, **38, 4**, 1454–1485, 2017.