Pass Efficient Algorithms for Approximating Large Matrices

Petros Drineas[*]
CS Department, Yale University
drineas@cs.yale.edu

Ravi Kannan [†]
CS Department, Yale University
kannan@cs.yale.edu

1 Summary

In many applications, an $m \times n$ matrix A is stored on disk and is too large to be read into RAM. Our main result is a succinct, easily computed, approximation A' to A, which is also an $m \times n$ matrix; A' has the following properties (s is a positive integer under our choice, usually constant):

(i) $A' = CUR$, where C is an $m \times s$ matrix consisting of s (randomly picked) columns of A; R is an $s \times n$ matrix consisting of s (randomly picked) rows of A and U is an $s \times s$ matrix computed from C, R.

(ii) C, U, R can be constructed after making two passes through the whole matrix A from disk,

(iii) using RAM space and additional time (in addition to the two full passes) $O(m + n)$ and

(iv) satisfies

$$\max_{x:|x|=1} |(A-A')x|^2 = |A-A'|_2^2 \leq \epsilon \sum_{i,j} A_{ij}^2 = \epsilon \|A\|_F^2$$

(v) satisfies an upper bound on $\|A - A'\|_F$ (to be described later). This upper bound is much smaller than $\|A\|_F$ when A has a good low-rank approximation, as is the case in many practical applications.

We will also present simple information theoretic arguments to show that this is in essence the best we can do. The above approximation can be used for "similarity query" problems, widely used in Information Retrieval and other areas: after A has been preprocessed, we get "query" vectors x and must find the similarity of x to each row of A. Here, the similarity of two vectors is defined to be their dot product or their normalized dot product; our technique can handle both.

Using an alternative method (essentially the technique of Achlioptas and McSherry in [1]), we also show another approximation A'' which can be computed in just one pass, but has the disadvantage that it does not satisfy (v); $\|A - A''\|_F$ might be as large as $\|A\|_F$.

Our algorithm uses adaptive sampling where we take a small sample of the data, but with non-uniform probabilities which reflect the relative sizes of the entries; **two passes** through the data are needed. Uniform sampling (**one pass** through the data) has been recently shown to be useful in approximately solving several problems, like the maximum cut problem on dense graphs using only $O(1)$ space. We formulate a model of "out-of-core" computation, which emphasizes **the number of passes** through the data from disk. This model has some similarities to the "streaming model" as well as older models studied in the context of sorting. In this model, we show some extensions of results approximating e.g. the maximum cut problem on dense graphs to certain non-dense graphs using adaptive sampling and **two passes**.

2 Introduction

We consider the problem of deriving a succinct approximation A' to an $m \times n$ matrix A stored on disk. It is easy to see (by information theory arguments) that if we require $\|A - A'\|_F^2 \leq \epsilon \|A\|_F^2$, then we will in general need Ω(number of non-zero entries in A) space. But, in many applications, we only need to compute Ax for "query vectors" x; a more natural measure of the approximation in these cases in the 2-norm (denoted $|\cdot|_2$) of $A - A'$, namely $\max_{x:|x|=1} |(A - A')x|$. We emphasize that the measure $|A - A'|_2$ is a worst case measure; this is more useful in many contexts than an average case measure, since the relevant query x often comes from a small dimensional subspace and *is not* random.

As stated in the summary, we prove $|A - A'|_2^2 \leq \epsilon \|A\|_F^2$; obviously, such a bound is only useful for matrices A for which $\|A\|_2^2 \gg \epsilon \|A\|_F^2$. This is indeed the case for matrices occurring in many contexts (e.g., matrices for which so-called Principal Component Analysis is used). But, we also prove a good upper bound on $\|A - A'\|_F$ for the more restricted class of matrices A for which there exists a good approximation of low rank.

Two quick examples are in order – one is the

[*]Supported by NSF Grant CCR-9820850.
[†]Supported in part by NSF Grant CCR-9820850.

"document-term" matrix, where we have a collection of m documents and n terms (used in the documents) and A_{ij} represents the number of occurrences of term j in document i or a function of this number. A second example pertains to a collection of images: the rows of the matrix represent images, the columns represent pixels and A_{ij} gives the intensity of pixel j in image i; in general, our technique can be used in the broad area of Principal Component Analysis where it may substitute for Singular Value Decomposition.

Our approximation A' is of the form CUR, where R is an $s \times n$ matrix consisting of $s = \theta(1/\epsilon^2)$ rows of A picked independently at random and C is an $m \times s$ matrix consisting of $s = \theta(1/\epsilon^2)$ columns of A picked independently at random. U is a $s \times s$ matrix which can be computed from C, R; assuming $\epsilon = \Omega(1)$, the picture looks like

$$\left(\qquad A \qquad \right) \approx \left(\quad C \quad \right) \cdot \left(\; U \; \right) \cdot \left(\qquad R \qquad \right)$$

Note that the length of our succinct approximation is $O((m+n)/\epsilon^2)$. In case the matrix A is sparse, with at most m' entries in any column and at most n' entries in any row, then we develop an approximation of length (of representation) at most $O((m' + n')/\epsilon^2)$; we will sketch a simple information theoretic argument proving a lower bound of $\Omega(m+n)$. The random sampling to get C, R will be according to a carefully chosen probability distribution, not necessarily the uniform. Also, as a by-product of the CUR approximation, we can estimate the singular values of A.

Our approximation may be viewed as a "dimension reduction" technique. The two main known techniques for dimension reduction which have been widely used are Random Projections (see [30], [28], [34]) and Singular Value Decomposition (SVD) (see [24], [17], [29], [11]). These methods do not share (i) and (ii) and thus are not suited for very large problems. Our algorithm achieves (i) and (ii) at the cost of some accuracy – the $\epsilon\|A\|_F^2$ error. Another aspect of the approximation is that it can be viewed as a way to reconstruct an approximation to the whole matrix A given a randomly chosen subset of columns and a randomly chosen subset of rows. But we caution that, as our theorem stands, it needs to know the probabilities that each sampled row and column was picked with (up to a scale factor); we emphasize that these probabilities should be known *only* for the randomly picked rows and columns, not for all rows and columns. Our approximation has already been used for recommendation systems applications to reconstruct a large matrix (see [13]).

We also show that the approximation only requires the input matrix to be presented in a particular general form, which we call the *unordered sparse* representation; the non-zero entries of A are presented as triples (i, j, A_{ij}) in any order. This is suited to applications, where multiple agents may write in parts of the matrix to a central database and we cannot make assumptions about the rules for write-conflict resolution; an example of this may be the "load" matrix, where each of many routers writes into a central database a log of the messages it routed during a day in the form of triples (source, destination, number of bytes).

Our algorithm draws a random sample of the entries of A and analyzes the sample. However, the difference between usual sampling algorithms and ours is that we do not blindly draw a random sample; what we do may be called adaptive sampling, where the sampling probabilities depend on the entries. In the first pass, we pick out a sample of rows and columns; the second pass is used to pick up the required sub matrices which form the whole sample. Then, we perform some computations in RAM with the sample. We remark that recently, there have been a number of results demonstrating the power of just blind sampling (see [5, 10, 16, 23, 3]); i.e., sampling where the probabilities are *not* based on the data. The advantage of blind sampling is that we may pick the sample before seeing the data, and, in just one pass through the data, extract the sample and analyze it. So, these results are one-pass algorithms and fit the well-studied "streaming" model (which we discuss later.) For example, one central problem the above papers solve by blind sampling is the following: given the adjacency matrix of a graph G, find the value of the maximum cut in G within **additive error** ϵn^2; this is useful only for dense graphs.

We will show here (section 4 and 4.1) that in our model, with two passes, we can tackle some interesting classes of non-dense graphs. In a sense, the algorithm we give can be viewed as exploiting the ability to sample a random edge in the graph, rather than a random vertex. What we show is that with two passes, and $O(\log n)$ extra RAM space and time, we can find in any graph $G(V, E)$ the MAX-CUT up to additive error $\epsilon(|V| - \ell)^2$, where the ℓ lowest vertex degrees add up to $\epsilon|E|/2$. Thus, if all but $r = o(n)$ vertices in the graph have "low" degrees, then the error is ϵr^2, instead of ϵn^2. We will tackle – using the CUR approximation – a wider class of problems of which MAX-CUT is an example.

3 The CUR approximation

The main technical result of this paper is stated and proved in this section. For any $m \times n$ matrix A, suppose C is an $m \times c$ matrix formed by a random subset of c columns of A, picked in c independent identical trials;

in each trial one of the n columns of A is picked with the probability of picking column j being q_j. The $\{q_j\}_{j=1}^n$ are nonnegative reals adding to 1, to be specified later; c is a positive integer. Our upper bounds on the errors will depend on c and will decrease as c increases. Similarly, suppose R is an $r \times n$ matrix formed by a random subset of r rows of A, picked in r independent identical trials; in each trial one of the m rows of A is picked with the probability of picking row i being p_i. Again, $\{p_i\}_{i=1}^m$ are nonnegative reals adding up to 1. In the following, $A_{(i)}$ will denote the i th row of matrix A (as a row vector) and $A^{(j)}$ will denote the j th column (as a column vector).

The main theorem will say that from C, R, we can compute a $c \times r$ matrix U such that $C \cdot U \cdot R$ is a good approximation to A provided the $\{p_i\}$ and the $\{q_j\}$ satisfy certain conditions; intuitively, heavier rows and columns have higher probabilities of being picked. To put our theorem in context, it will be useful to contrast it with Singular Value Decomposition (SVD). We remind the reader that the SVD of A expresses A as (here ρ is the rank of A)

$$A = \sum_{t=1}^{\rho} \sigma_t(A) u^{(t)} {v^{(t)}}^T$$

where $\{\sigma_t(A)\}_{t=1}^{\rho}$ are the singular values of A and $u^{(t)}$, $v^{(t)}$ the corresponding left/right singular vectors of A respectively. It is well-known that the first k terms of this expansion give us the "optimal" rank k approximation to A with respect to both the 2-norm and the Frobenius norm.

Note that we can write $\sum_{t=1}^{k} \sigma_t(A) u^{(t)} {v^{(t)}}^T$ as $U_k \Sigma_k V_k^T$, where U_k is an $m \times k$ matrix, Σ_k is a $k \times k$ diagonal matrix and V_k is a $k \times n$ matrix. So, computing the SVD gives us good succinct approximations, since it only takes space $O(k(m+n))$ to write down U_k, Σ_k, V_k. But the computational problem of finding the SVD is difficult and cannot be carried out by any means in $O(1)$ passes. Instead our theorem will say that weaker bounds (which are however similar in spirit) may be achieved by CUR (which is similar to $U\Sigma V$). Indeed, we will show that if the rows and columns are picked with probabilities proportional to their length squared, then the bounds given by the following corollary on the errors hold.

Corollary 3.1. *If $p_i = |A_{(i)}|^2/\|A\|_F^2$ and $q_j = |A^{(j)}|^2/\|A\|_F^2$ for $i = 1, \ldots, m$ and $j = 1, \ldots, n$, then*

$$\begin{aligned}\mathbf{E}\left(|A - CUR|_2^2\right) &\leq |A - A_{\sqrt{s}+1}|_2^2 + \epsilon_1 \|A\|_F^2 \\ \mathbf{E}\left(\|A - CUR\|_F^2\right) &\leq \|A - A_{\sqrt{s}+1}\|_F^2 + \epsilon_2 \|A\|_F^2\end{aligned}$$

where $\epsilon_1 = 3/\sqrt{s}$ and $\epsilon_2 = 2/s^{1/4} + 1/\sqrt{s}$.

An advantage of our method is that if A is sparse and each row and column of A has a small number of non-zero entries, then the CUR representation also has a small number of entries; this should be obvious since C, R are just small parts of A and U is a $c \times r$ matrix. This advantage is not enjoyed by the SVD which in general destroys sparsity.

For simplicity we only present results for the expectation of the error (tight concentration can be shown through martingale arguments). The corollary above will follow from a more general theorem which we presently describe; we relax the condition that the probabilities be exactly proportional to the row (or column) length squared. Instead, we require $(\alpha, \beta \leq 1)$:

(3.1) $\quad p_i \geq 0, \quad \sum_i p_i = 1, \quad p_i \geq \alpha |A_{(i)}|^2/\|A\|_F^2$

(3.2) $\quad q_j \geq 0, \quad \sum_j q_j = 1, \quad q_j \geq \beta |A^{(j)}|^2/\|A\|_F^2$

We also let C have c columns and R have r rows (we need not have $r = c$). Finally, there will be another parameter k under our control, which we will specify later; we describe the computation of C, U, R in the following algorithm:

The CUR algorithm

Input: $m \times n$ matrix A, positive integers $r \leq m$, $c \leq n$ and $k \leq \min(r, c)$, $\{p_i\}_{i=1}^m$, $\{q_j\}_{j=1}^n$.
Output: C ($m \times c$ matrix), U ($c \times r$ matrix) and R ($r \times n$ matrix).

1. **for** $t = 1$ **to** c **independently** pick $j_t \in \{1 \ldots n\}$ in i.i.d. trials with $\mathbf{Pr}(j_t = j) = q_j$. Let C be the $m \times c$ matrix whose t-th column is $A^{(j_t)}$ for $t = 1, \ldots, c$.

2. Let D_1 be the $c \times c$ diagonal matrix with $1/\sqrt{c q_{j_t}}$ in the (t,t)-th position for $t = 1, \ldots, c$.

   Note: we need to know q_j only for the sampled columns j.

3. Let $C' = C D_1$ (C' is C with columns suitably scaled). Compute ${C'}^T C'$, a $c \times c$ matrix and its SVD; suppose it is
$$ {C'}^T C' = \sum_t \sigma_t^2(C') y^{(t)} {y^{(t)}}^T $$
   Choose a k such that $\sigma_k(C') > 0$.

4. **for** $t = 1$ **to** r **independently** pick $i_t \in \{1 \ldots m\}$ with $\mathbf{Pr}(i_t = i) = p_i$.

   Let R be the $r \times n$ matrix whose t-th row is $A_{(i_t)}$ for $t = 1, \ldots, r$.

5. Let D_2 be the $r \times r$ diagonal matrix with $1/\sqrt{rp_{i_t}}$ as the (t,t) th entry for $t = 1, \ldots, r$.

   Note: we need to know p_i only for the sampled rows i.

6. Let W be the $r \times m$ matrix such that $W_{1i_1} = 1, W_{2i_2} = 1, \ldots, W_{ri_r} = 1$ (the remaining elements of W are zeros). Then WC' is the $r \times c$ matrix whose rows are $C'_{(i_t)}$, $t = 1, \ldots, r$. Define

$$U = D_1 \left(\sum_{t=1}^{k} \frac{1}{\sigma_t^2(C')} y^{(t)} {y^{(t)}}^T \right) {C'}^T W^T D_2^2$$

Theorem 3.1. *If $\{p_i\}_{i=1}^m$ satisfy (3.1) and $\{q_j\}_{i=1}^n$ satisfy (3.2),*

$$\begin{aligned} \text{(3.3)}\ \mathbf{E}\left(\|A - CUR\|_F^2\right) &\le \|A - A_k\|_F^2 + \epsilon_1 \|A\|_F^2 \\ \text{(3.4)}\ \ \mathbf{E}\left(|A - CUR|_2^2\right) &\le |A - A_k|_2^2 + \epsilon_2 \|A\|_F^2 \\ &\le \left(\frac{1}{k+1} + \epsilon_2 \right) \|A\|_F^2 \end{aligned}$$

where $\epsilon_1 = 2\sqrt{k/\beta c} + k/\alpha r$ and $\epsilon_2 = 2/\sqrt{\beta c} + k/\alpha r$.

Remarks: Corollary 3.1 will follow from the theorem by taking $r = c = s$ and $k = \sqrt{s}$. If $\epsilon > 0$ is an error parameter, choosing $k = 2/\epsilon$, $c = 64/(\beta\epsilon^2)$ and $r = 8/(\alpha\epsilon^2)$ makes $|A - CUR|_2 \le \epsilon \|A\|_F^2$. Thus in essence the theorem says that sampling $\Omega(1/\epsilon^2)$ rows and $\Omega(1/\epsilon^2)$ columns is sufficient for an approximation within 2-norm error at most $\epsilon\|A\|_F^2$.

Proof: We would like to give some intuition on why CUR is close to A. Let, $h^{(t)} = C'y^{(t)}/\sigma_t(C')$; $h^{(t)}$ are the left singular vectors of C'. Also, let $H_{m \times k} = \left(h^{(1)}\ h^{(2)}\ \ldots h^{(k)}\right)$; the projection $\tilde{A}$ of A to the subspace spanned by the top k $h^{(t)}$'s

$$\tilde{A} = \sum_{t=1}^{k} h^{(t)} {h^{(t)}}^T A = HH^T A \tag{3.5}$$

can be shown to "capture" *almost* as much of the Frobenius norm of A as A's projection into the space spanned by its own top k left singular vectors. Indeed it was shown in [11] that $\|A - \tilde{A}\|_F^2$ is small (see Theorem 3.2 below). Thus, $\tilde{A}$ would have been a fine approximation to A, but for the fact that it is hard to compute – multiplying each ${h^{(t)}}^T$ by A requires one pass through A for a total of k passes! We emphasize here that although equation 3.6 of Theorem 3.2 was known from [11], equation 3.7 of Theorem 3.2 which proves a similar bound with respect to the 2-norm is new. Note that the singular value decomposition of C' is $C' = \sum_t \sigma_t(C') h^{(t)} {y^{(t)}}^T$. Thus, we may write

$$\begin{aligned} CUR &= C' \left(\sum_{t=1}^{k} \frac{1}{\sigma_t^2(C')} y^{(t)} {y^{(t)}}^T \right) {C'}^T W^T D_2^2 R \\ &= \left(\sum_{t_1} \sigma_{t_1}(C') h^{(t_1)} {y^{(t_1)}}^T \right) \left(\sum_{t_2=1}^{k} \frac{1}{\sigma_{t_2}^2(C')} y^{(t_2)} {y^{(t_2)}}^T \right) \cdot \\ &\quad \cdot \left(\sum_{t_3} \sigma_{t_3}(C') y^{(t_3)} {h^{(t_3)}}^T \right) W^T D_2^2 R \\ &= \left(\sum_{t=1}^{k} h^{(t)} {h^{(t)}}^T \right) W^T D_2^2 R = HH^T W^T D_2^2 R \end{aligned}$$

We remind that W^T is an $m \times r$ matrix, denoting which rows of A are included in R. Moving back to equation 3.5, instead of explicitly computing the product $H^T A$ we will approximate it using a technique of [12]; this technique says that if we pick a random subset of columns of H^T and the corresponding set of rows of A, scale them appropriately and multiply the resulting matrices, we get an accurate approximation to the product $H^T A$, *if* the probabilities used for the random choices satisfy certain inequalities (see lemma 3.1). Indeed, the reader may verify that $(H^T W^T D_2)(D_2 R)$ is obtained precisely by choosing columns $i_1, i_2, \ldots, i_r$ of H^T and the corresponding rows of A, while the multiplication by D_2 scales the rows/columns appropriately! This will lead us to the result that $H^T A \approx (H^T W^T D_2)(D_2 R)$ in both the 2-norm and the Frobenius norm sense, giving us the bounds of the theorem.

We start with the following lemma (very similar to a lemma of [12]), whose proof may be found in the Appendix.

Lemma 3.1. *Suppose A, B are $m \times n$ and $n \times p$ matrices respectively; let $\{p_i\}_{i=1}^n$ be nonnegative reals summing to 1 such that, for all i,* **either** *$p_i \ge \alpha |A^{(i)}|^2 / \|A\|_F^2$* **or** *$p_i \ge \alpha |B_{(i)}|^2 / \|B\|_F^2$ for some $\alpha \le 1$. Suppose $\{i_t\}_{t=1}^s$ are picked by i.i.d. trials in each of which an element from $\{1, 2, \ldots, n\}$ is picked according to the probabilities p_i. Let S be the $m \times s$ matrix with columns $A^{(i_t)}/\sqrt{sp_{i_t}}$ and R be the $s \times p$ matrix with rows $B_{(i_t)}/\sqrt{sp_{i_t}}$. Then,*

$$\mathbf{E}\left(\|AB - SR\|_F^2\right) \le \frac{1}{\alpha s} \|A\|_F^2 \|B\|_F^2$$

Lemma 3.2. *Using the above notation,*

$$\begin{aligned} \mathbf{E}\left(\|H(H^T A - H^T W^T D_2^2 R)\|_F^2\right) &\le \frac{1}{\alpha r} \|H^T\|_F^2 \|A\|_F^2 \\ &= \frac{k}{\alpha r} \|A\|_F^2 \end{aligned}$$

Proof: Since H is an orthogonal matrix, it follows that $\|H(H^TA - H^TW^TD_2^2R)\|_F^2 = \|H^TA - H^TW^TD_2^2R\|_F^2$. Then, the first inequality follows from lemma 3.1, by observing that D_2R consists of a few rows of A suitably scaled and picked with probabilities proportional to the square of their lengths; $H^TW^TD_2$ consists of the corresponding columns of A. Finally, we observe that the columns of H are unit vectors and $\|H^T\|_F^2 = k$.

$\diamond$

The Frobenius norm bound in the following theorem essentially comes from [11]. The 2-norm bound is new; it nicely complements the Frobenius norm bound by removing its dependency on k. Its proof may be found in the Appendix.

THEOREM 3.2. *If H is defined as in equation 3.5, then, for any $c \leq n$,*

$$\mathbf{E}\left(\|A - HH^TA\|_F^2\right) \leq \|A - A_k\|_F^2 + 2\sqrt{\frac{k}{\beta c}}\|A\|_F^2 \tag{3.6}$$

$$\mathbf{E}\left(|A - HH^TA|_2^2\right) \leq |A - A_k|_2^2 + \frac{2}{\sqrt{\beta c}}\|A\|_F^2 \tag{3.7}$$

To prove equation 3.3 of Theorem 3.1, observe that (from linearity of expectation)

$$\begin{aligned}
& \mathbf{E}\left(\|A - CUR\|_F^2\right) \leq \mathbf{E}\left(\|A - HH^TA\|_F^2\right) + \\
+ \quad & \mathbf{E}\left(\|HH^TA - HH^TW^TD_2^2R\|_F^2\right) \\
\leq \quad & \|A - A_k\|_F^2 + 2\sqrt{\frac{k}{\beta c}}\|A\|_F^2 \\
+ \quad & \mathbf{E}\left(\|H^TA - H^TW^TD_2^2R\|_F^2\right) \\
\leq \quad & \|A - A_k\|_F^2 + \left(\sqrt{\frac{4k}{\beta c}} + \frac{k}{\alpha r}\right)\|A\|_F^2
\end{aligned}$$

For the above derivation we used Theorem 3.2 and lemma 3.2. Similarly, since $|\cdot|_2 \leq \|\cdot\|_F$,

$$\begin{aligned}
\mathbf{E}\left(|A - CUR|_2^2\right) \quad \leq \quad & \mathbf{E}\left(|A - HH^TA|_2^2\right) \\
+ \quad & \mathbf{E}\left(\|H^TA - H^TW^TD_2^2R\|_F^2\right) \\
\leq \quad & |A - A_k|_2^2 + \left(\frac{2}{\sqrt{\beta c}} + \frac{k}{\alpha r}\right)\|A\|_F^2
\end{aligned}$$

and equation 3.4 of Theorem 3.1 is proven as well (since $|A - A_k|_2^2 = \sigma_{k+1}^2(A) \leq \|A\|_F^2/(k+1)$).

$\diamond$

3.1 Sampling We prove that with the matrix presented in sparse unordered representation, all the sampling necessary to compute C, R can be done in two passes through the matrix. The following two lemmas are simple technical claims.

LEMMA 3.3. *Suppose $a_1, a_2, \ldots a_n$ are n non-negative reals which are read once in this order (streaming). Then with $O(s)$ additional storage, we can pick i.i.d. samples $i_1, i_2, \ldots i_s \in \{1, 2, \ldots n\}$ such that*

$$\mathbf{Pr}(i_t = i) = \frac{a_i}{\sum_{j=1}^n a_j}.$$

Proof: We argue that we can pick i_1. The others can be done by running s independent copies of this process. To pick i_1, suppose we have read $a_1, a_2, \ldots a_l$ so far and have a sample i_1 such that $\mathbf{Pr}(i_1 = i) = a_i / \sum_{j=1}^l a_j$ and also we keep the running sum $\sum_{j=1}^l a_j$. On reading a_{l+1}, we just replace the current i_1 with $l+1$ with probability $a_{l+1} / \sum_{j=1}^{l+1} a_j$. It is easy to see by induction that this works.

$\diamond$

LEMMA 3.4. *In one pass, plus $O(m+n)$ additional storage, we can pick i.i.d. samples $j_1, j_2, \ldots j_c$ drawn according to probabilities for the columns $\{q_j\}$ satisfying (3.2) and also pick i.i.d. samples $i_1, i_2, \ldots i_r$ drawn according to probabilities for the rows satisfying (3.1).*

Proof: To pick i_1 just pick (using the previous claim) an entry (i, j) with probabilities proportional to their squares and just take $i_1 = i$. The other i_t and the j_t are also picked by running $c + r$ independent experiments simultaneously.

$\diamond$

In the second pass, we pick out the entries of the matrices C and R; note that we know the scaling factors since we know the probabilities with which we pick each row and column. Since we have $O(m+n)$ storage, these can be explicitly computed as well as C^TC. The running time of the CUR algorithm, excluding the two passes through A, is $O(m)$.

3.2 Lower Bound The following lemma provides a lower bound for matrix approximation:

LEMMA 3.5. *For any m, n positive integers, and $1 > \epsilon > \frac{8}{\sqrt{n}} + \frac{8}{\sqrt{m}}$, there exists a set of $\Omega(\epsilon^{-n-m})$ $m \times n$ matrices, such that*

- *Each matrix in the set has Frobenius norm ≤ 4.*
- *Each entry of each matrix in the set is an integer multiple of $\epsilon/64\sqrt{mn}$.*
- *For two distinct matrices A, A' in the set, $|A - A'|_2 \geq \epsilon/80$.*

The idea of the simple, but quite technical, proof is to pick a "large" set of vectors $u \in \mathbf{R}^n$ such that each u is of length almost exactly 1 and we have

$|u - u'|, |u + u'| \geq \epsilon$ for each pair of distinct u, u'. Similarly one picks a set of $v \in \mathbf{R}^m$ satisfying similar conditions. Then we form all rank 1 matrices of the form uv^T and show that this class has the claimed properties. The lemma supplies a lower bound, since any algorithm which approximates these matrices must output a different approximation to each one, requiring it to output at least $O((n + m)\log(1/\epsilon))$ bits; this is only a lower bound on the number of output bits.

Remark: random projections will not do; by the lower bound, with $o(m + n)$ description length, we can only achieve an error bound of $\Omega(\epsilon\|A\|_F)$. Assume that we do a random projection of each row to an s-dimensional space to get a matrix B; say that for a particular unit length vector x its projection is x'. Then, we only get that, with high probability, $|A_{(i)}x - B_{(i)}x'| \leq \epsilon|A_{(i)}|$. Squaring and adding over all the rows, $|Ax - Bx'| \leq \epsilon\|A\|_F$ holds with high probability for each x. So, for *most* x ' s the inequality holds. But this is useless, since, for most x ' s, we may have $|Ax| \leq \epsilon\|A\|_F$. The only x ' s that are important are of small measure.

3.3 An alternative approach The results of this section come from [1] (see also [6], [31]); our sole contribution is the introduction of adaptive sampling to improve their error bounds. Their idea is very simple and appealing: given an $m \times n$ matrix A sample elements of A; create an $m \times n$ matrix $\tilde{A}$ by keeping elements of A that are included in the sample (after dividing them by the probability of being sampled). The remaining elements of $\tilde{A}$ are zeroed out; essentially, $\tilde{A}$ is a sparse version of A. Using an elegant result of Furedi and Komlos (see [18]) one may prove that $A - \tilde{A}$ is small with respect to the 2-norm. Thus, one could use *a low rank approximation* to $\tilde{A}$ as a succinct approximation to A. More specifically, if $\tilde{A} = \tilde{U}\tilde{\Sigma}\tilde{V}^T$, the succinct approximation to A is $A'' = \tilde{U}_k\tilde{\Sigma}_k\tilde{V}_k^T$ for some positive integer k; we can store A'' in $O(k(m+n))$ space. We note that $\tilde{A}$ is an $m \times n$ matrix, thus in order to compute its SVD efficiently we employ the Lanczos/Arnoldi techniques. The proof of the following theorem is straight-forward using the results of [1] and a bound on $|A - \tilde{A}|_2$ if *adaptive sampling* is used (instead of the uniform sampling of [1]).

THEOREM 3.3. *If $\tilde{A}_k$ is constructed as described above (by including $sm + sn$ elements of A in $\tilde{A}$), then, with probability at least* $1 - (m + n)^{-1}$,

$$\begin{aligned} |A - A''|_2^2 &\leq \left(\frac{1}{k+1} + \frac{296}{s}\right)\|A\|_F^2 \\ \|A - A''\|_F^2 &\leq \|A - A_k\|_F^2 + 8k\sigma_{k+1}^2(A) + \frac{392k}{s}\|A\|_F^2 \end{aligned}$$

Remark: we assume that $sm+sn$ elements of A are included in $\tilde{A}$ in order to easily compare $A' = CUR$ and A''. For the above theorem to hold certain assumptions must be satisfied; for now we ignore them.

We now compare A' and A'' asymptotically, ignoring the constants and the probabilities with which they hold: the error bound of the element-wise sampling with respect to the 2-norm is asymptotically better; more specifically, for the CUR algorithm to achieve an error of $\epsilon\|A\|_F^2$, one must set $k = O(1/\epsilon)$ and sample $O(1/\epsilon^2)$ rows and $O(1/\epsilon^2)$ columns; element-wise sampling achieves the same error by setting $k = O(1/\epsilon)$ and picking $O(1/\epsilon)$ rows and columns. On the other hand, the error bound of the CUR algorithm with respect to the Frobenius norm is generally better, because of the $k \cdot \sigma_{k+1}^2(A)$ factor that appears in the element-wise sampling error bound. Even worse, for matrices such that $\sigma_{k+1}^2(A) \approx \|A\|_F^2/(k+1)$ (its maximum value) we can easily see that the error of the element-wise approach is almost $\|A\|_F^2$. Thus, the element-wise error sampling lacks a general, useful error bound with respect to the Frobenius norm; one must assume that the singular values of A are dropping fast and, specifically, that $\sigma_{k+1}^2(A) \ll \|A\|_F^2/(k+1)$.

We also note that element-wise sampling (even weighted) may be implemented in one pass using lemma 3.3, while CUR necessitates two passes through A. On the other hand, to approximate the singular vectors/values of $\tilde{A}$ one needs to run Lanczos/Arnoldi algorithms. In the analysis above, we assumed that these algorithms returned exact results *and* converged fast; still, their running time and accuracy depends on various factors such as the choice of an initial random vector and the spectral structure of the matrix.

4 The "pass-efficient" model

In this section we formulate a model of "out-of-core" computation, which emphasizes the number of passes through the data from disk (the data are *not* necessarily a matrix); the algorithms of section 3 conform to this model. The only access we assume to the data is via a pass which is a sequential read of the entire input from disk. The motivation behind our model is simple: in modern computers, the amount of disk storage (sequential access memory) has increased enormously while RAM and computing speeds have increased, but at a substantially slower pace. Thus, we have the ability to store very large amounts of data, but not in RAM. Also, we do not have the ability to process this data with algorithms which may take low polynomial time, or even linear time with large constants. To model this reality, we propose a model of computation in which we allow a small number (for example 2) of passes (sequential reads) through the entire data plus sub-linear computation time and Random Access Memory space; algorithms that conform to the above model are called *pass-efficient*.

This model assumes that the input data may only be accessed by making passes through it; a pass consists of one sequential read of the entire data plus additional computation time of at most q units after each r bits of data are read. The number r reflects the fact that usually one reads whole blocks of data, not just one bit. The quantity q is there for the following reason: input/output operations (from out of core) take a lot of cycles, so, if q is sufficiently small – compared to the number of cycles needed to read in a block – then, with a small increase to the running time, we can process the block just read. We are not particularly interested in the values of q and r, but the spirit is that a pass is just one read of the data with a small amount of processing. Note that since additional space is only sub-linear, we really have to allow some processing during a pass, at least to figure out which parts of the input to retain in memory. Otherwise, since we cannot by far store all the data read in a pass, the pass would just be wasted.

The algorithm is also allowed to use some computation time and Random Access Memory space besides the passes. These are required to be sub-linear – sometimes substantially so; as an example, since a matrix is an $m = n^2$ stream, our CUR algorithm spends $O(\sqrt{m}) = O(n)$ time. We will measure three parameters of the algorithm: the number of passes, additional time and space. A model where the number of passes was measured as a basic parameter was first used by Munro and Paterson [32] in the special case of sorting and selection algorithms; their definition of a pass though allowed additional computation time of essentially linear time, which, as argued here, is not practical for our problems.

The "Streaming Model" (see [27], [4], [14]), on which there is substantial work, allows only one pass through the data and restricts the RAM usage to poly-logarithmic amount of space. But the model formalized in [14] allows poly-logarithmic time for processing after reading each bit; so a pass for them is even more generous than the model in [32]. The primary concern of both these models is the additional space (RAM) usage, rather than time. The restriction to one pass in the streaming model allows processing vast amounts of data supplied from other sources which we do not have space to store at all, but which we can "leisurely" look at from a read only 1-way tape taking super-linear time ($O(n)$ times poly-logarithmic) for the pass.

4.1 A 2-pass algorithm for approximating MAX-CUT In this section, we sketch how in two passes plus $O(\log n)$ additional time and RAM space, we can find the value of the maximum cut in a graph $G(V, E)$ to additive error $\epsilon(|V| - \ell)^2$, where ℓ is chosen so that the sum of the lowest ℓ degrees is $\epsilon|E|/2$. This algorithm will only use elementary ideas; essentially, we pick vertices in the first pass with probabilities proportional to the degrees (by picking a random edge); we will then discard certain vertices of low degree and then do some rejection sampling after which, we have a uniform random sample from the high degree vertices. Here are a few more details.

Let $G(\{1, 2, \dots n\}, E)$ be a graph. Let d_i be the degree of the vertex i. We assume that each $d_i \le \epsilon|E|/4$. This is a mild assumption that no vertex has more than an $\epsilon/4$ fraction of all edges incident to it, which is obviously true if for example, we have a super-linear number of edges. For any $f \in (0\ ,\ 1)$, define $v(f)$ to be the least positive integer such that $\sum_{i: d_i \le v(f)} d_i \ge f|E|$ and let $V(f) = \{i : d_i \le v(f)\}$.

We pick $i_1, i_2, \dots i_{3s}$ all i.i.d. samples, where s will be $\Omega(\log n/\text{poly}(\epsilon))$ with $\mathbf{Pr}(i_t = i) = d_i/(2|E|)$ [by picking a random edge] in the first pass; we also find $|E|$ exactly in the first pass. In the second pass, we collect the induced graph on these $3s$ vertices and also compute the degree of each of these $3s$ vertices in the whole graph.

Define L, M to be the minimum positive integers such that

$$|\{t : 1 \le t \le s, d_{i_t} \le L\}| \ge 2\epsilon s$$
$$|\{t : 1 \le t \le s, d_{i_t} \le M\}| \ge 5\epsilon s$$

We have that the probability of a single i_t falling in $V(\epsilon)$ is equal to $\sum_{i \in V(\epsilon)} d_i/|E|$ which is between ϵ and $(5/4)\epsilon$. So using Hoeffding on Bernouli trials, we have that with high probability:

$$|\{i_1, i_2, \dots i_s\} \cap V(\epsilon)| \le 1.5\epsilon s \quad \Rightarrow \quad L \ge v(\epsilon)$$
$$|\{i_1, i_2, \dots i_s\} \cap V(3\epsilon)| \ge 2.5\epsilon s \quad \Rightarrow \quad L \le v(3\epsilon).$$

Similarly, with high probability, $v(4\epsilon) \le M \le v(6\epsilon)$. Let $W = \left\{ i : v(\epsilon) \le d_i \le v(6\epsilon) : d_i \le \frac{\epsilon^2|E|}{2\log n |\{j : d_j \ge d_i\}|} \right\}$.

$$\begin{aligned} \sum_{i \in W} d_i &\le \epsilon^2 |E| \frac{1}{2\log n} \sum_{i \in V(6\epsilon) \setminus V(\epsilon)} (1/|\{j : d_j \ge d_i\}|) \\ &\le \frac{\epsilon^2}{2\log n} |E| \left(\sum_{i=1}^{n} \frac{1}{i} \right) \le \epsilon^2|E|/2 \end{aligned}$$

Thus the probability that a particular i_t belongs to W is at most $\epsilon^2/2$. Now, we use the second set of s samples.

Let $P = \{i_t : s+1 \le t \le 2s, d_{i_t} \in [L\ ,\ M]\}$. Pick uniformly at random an element q of P. By the above, with high probability, we have that $q \notin W$.

Now, we use the third batch of s samples. For each $i_t, 2s+1 \le t \le 3s$, we independently do the following: if $d_{i_t} < d_q$, then we set all entries in row i_t and column i_t of our sampled sub-matrix to be zero. Otherwise, we accept sample i_t with probability d_q/d_{i_t}. Now we have that the acceptance probability of a sample is $\sum_{i:d_i \ge d_q}(d_q d_i/d_i|E|) \ge \epsilon^2/2\log n$. Thus, with $s = \Omega(\log n/\text{poly}(\epsilon))$, we will have the required number of poly$(1/\epsilon)$ samples surviving the rejection process for us to appeal to the earlier results. Also a simple calculation shows that these are samples drawn uniformly from vertices of degree at least d_q, so appealing to results of say [23], we get the claimed algorithm.

5 Conclusions and Future Directions

We presented an algorithm that computes a succinct approximation A' to any $m \times n$ matrix A, s.t. $A - A'$ is small with respect to *both* the 2-norm and the Frobenius norm. The running time of the algorithm is $O(m)$ *after* two passes through A; essentially, each pass is a sequential read of the elements of A. We should note here that the $O(m)$ time is spent in the computation of U and, in particular, in the computation of $C'^T C'$ (step 3 of the algorithm, see section 3); we can improve on that by *approximating* $C'^T C'$ by sampling a few columns of C'^T (using the technique of [12], see lemma 3.1). One can prove that the loss in accuracy is not significant, thus we get a *constant* time algorithm; we defer its proof to the full version.

In section 4.1 we presented an algorithm to approximate the maximum cut using **two passes** through the graph. Our algorithm returns meaningful error bounds for a larger class of graphs than previous results; its power stems from the *adaptive sampling* of vertices in the second pass. We presented this algorithm in order to illustrate the power of "pass-efficient" algorithms; we are currently investigating how CUR approximation (our main two pass algorithm) and its constant time variant may be used to design efficient approximation algorithms for all MAX-2-CSP problems. Our goal is to efficiently approximate *sparser* instances of MAX-2-CSP problems in constant time and space; we remind the reader that there are efficient approximation algorithms for dense instances for all MAX-2-CSP problems, but almost nothing on sparse instances.

Finally, there are many interesting problems to investigate in the context of the model proposed in section 4; essentially problems investigated in the context of the "streaming" model. We mention some that have received significant attention in the recent years: approximating frequency moments in data streams and lower bounds (see [4, 14, 28]); estimating the symmetric difference of data streams with respect to p-norms, $p \in (0, 2]$ (see [14, 28, 33]); finding frequent or duplicate items (see [4, 9, 8]); nearest and near-neighbor problems (see [22, 26]); clustering data streams and histogram computation (see [25, 20, 7]); probabilistic counting in a data stream – i.e. number of inversions, number of distinct elements, etc. (see [2, 15, 19]; Fourier transforms (see [21]). One might hope that with more than one passes through the data more efficient and meaningful samples of small size could be constructed and thus the polylogarithmic time spent in the "streaming" model could be avoided.

References

[1] D. Achlioptas and F. McSherry. Fast computation of low rank approximations. *Proceedings of the 33rd Annual Symposium on Theory of Computing*, 2001.

[2] M. Ajtai, T. Jayram, R. Kumar, and D. Sivakumar. Approximate counting of inversions in a data stream. *Proceedings of the 34th Annual Symposium on Theory of Computing*, 2002.

[3] N. Alon, W.F. DeLaVega, R. Kannan, and M. Karpinski. Random sub-problems of Max-SNP problems. *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, 2002.

[4] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *Proceedings of the 28th Annual ACM Symposium on Theory of Computing*, 1996.

[5] S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. *Proceedings of the 27th Annual ACM Symposium on Theory of Computing*, 1995.

[6] Y. Azar, A. Fiat, A. Karlin, F. McSherry, and J. Saia. Spectral analysis of data. *Proc. of the 33rd ACM Symposium on Theory of Computing*, 2001.

[7] M. Badoiu, S. Har-Peled, and P. Indyk. Approximate clustering via core sets. *Proceedings of the 34th Annual Symposium on Theory of Computing*, 2002.

[8] A. Broder, M. Charikar, A. Frieze, and M. Mitzenmacher. Min-wise independent permutations. *Proceedings of the 30th Annual ACM Symposium on Theory of Computing*, 1998.

[9] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. *International Colloquium on Automata, Languages and Programming*, 2002.

[10] W. Fernandez de-la Vega. Max-cut has a randomized approximation scheme in dense graphs. *Random Structures and Algorithms*, 8:187–199, 1996.

[11] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay. Clustering in large graphs and matrices. *Proceedings of the 10th ACM-SIAM Symposium on Discrete Algorithms*, 1999.

[12] P. Drineas and R. Kannan. Fast monte-carlo algorithms for approximate matrix multiplication. *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, 2001.

[13] P. Drineas, I. Kerenidis, and P. Raghavan. Competitive recommendation systems. *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 668–677, 2002.

[14] J. Feigenbaum, S. Kannan, M. Strauss, and M. Vishwanathan. An approximate l^1-differnce algorithm for massive data sets. *Proceedings of the 40th Annual IEEE Symposium on the Foundations of Computer Science*, 1999.

[15] P. Flajolet and G. Martin. Probabilistic counting. *Proc. 24th IEEE Symp. on Foundations of Computer Science*, 1983.

[16] A. Frieze and R. Kannan. The regularity lemma and approximation schemes for dense problems. *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computing*, 1996.

[17] A. Frieze, R. Kannan, and S. Vempala. Fast monte-carlo algorithms for finding low rank approximations. *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, 1998.

[18] Z. Furedi and J. Komlos. The eigenvalues of random symmetric matrices. *Combinatorica*, 1:233–241, 1981.

[19] P. Gibbons and Y. Matias. Synopsis data structures for massive data sets. *DIMACS Series in Discrite Mathematics and Theoretical Computer Science: Special Issue on external memory algorithms and visualization, American Mathematical Society*, pages 39–70, 1999.

[20] A. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. Strauss. Fast, small-space algorithms for approximate histogram maintanance. *Proceedings of the 34th Annual Symposium on Theory of Computing*, 2002.

[21] A. Gilbert, S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss. Near optimal sparse fourier representations via sampling. *Proceedings of the 34th Annual Symposium on Theory of Computing*, 2002.

[22] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. *Proceedings of the 25th VLDB Conference*, 1999.

[23] O. Goldreich, S. Goldwasser, and D. Ron. Property testing and its connection to learning and approximation. *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computing*, 1996.

[24] G. Golub and C. Van Loan. *Matrix Computations.* Johns Hopkins University Press, 1989.

[25] S. Guha, N. Koudas, and K. Shim. Data streams and histograms. *Proceedings of the 33rd Annual Symposium on Theory of Computing*, 2001.

[26] D. Gunopulos, G. Kollios, V. Tsotras, and C. Domeniconi. Approximate multi-dimensional aggregate range queries over real attributes. *Proceedings of ACM SIGMOD conference*, 2000.

[27] M. Henzinger, P. Raghavan, and S. Rajagopalan. Computing on data streams. Technical note 1998-011, Digital Systems Research Center, Palo Alto, CA, May 1998.

[28] P. Indyk. Stable distributions, pseudo-random generators, embeddings, and data stream computation. *Proceedings of the 41st Annual IEEE Symposium on Foundations of Computing*, 2000.

[29] R. Kanth, D. Agrawal, and Amr El Abbadi. Dimensionality reduction for similarity searching in dynamic databases. *ACM SIGMOD*, 1998.

[30] J. Kleinberg. Two algorithms for nearest neighbor search in high dimensions. *Proceedings of the 29th Symposium on Theory of Computing*, 1997.

[31] F. McSherry. Spectral partitioning of random graphs. *Proceedings of the 42nd Annual Symposium on Foundations of Computer Science*, 2001.

[32] J.I. Munro and M.S. Paterson. Selection and sorting with limited storage. *Proceedings of the 19th Annual IEEE Symposium on Foundations of Computing*, 1978.

[33] M. Saks and X. Sun. Space lower bounds for distance approximation in the data stream model. *Proceedings of the 34th Annual Symposium on Theory of Computing*, 2002.

[34] S. Vempala. Random Projection: A new approach to VLSI layout. *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computing*, 1998.

Appendix

Proof of lemma 3.1: Following the lines of [17] and [12], we seek to bound $\mathbf{E}\left(\sum_{i,j=1}^{m,p}\left((AB)_{ij}-(SR)_{ij}\right)^2\right)$.

Fix attention on one particular i,j. For $t=1\ldots s$ define the random variable $w_t = \left(\frac{A^{(i_t)}B_{(i_t)}}{sp_{i_t}}\right)_{ij} = A_{ii_t}B_{i_tj}/sp_{i_t}$. So, the w_t's are independent random variables. Also, $(SR)_{ij}=\sum_{t=1}^{s} w_t$. Thus, its expectation is equal to the sum of the expectations of the w_t's. But, $\mathbf{E}(w_t)=\sum_{k=1}^{n}\frac{A_{ik}B_{kj}}{sp_k}p_k = \frac{1}{s}(AB)_{ij}$. So, $\mathbf{E}((SR)_{ij}) = \sum_{t=1}^{s}\mathbf{E}(w_t) = (AB)_{ij}$. Since $(SR)_{ij}$ is the sum of s independent random variables, the variance of $(SR)_{ij}$ is the sum of the variances of these variables. But, using $\mathbf{Var}(w_t) = \mathbf{E}\left(w_t^2\right) - \mathbf{E}^2(w_t)$ we see that $\mathbf{Var}(w_t) = \sum_{k=1}^{n}\frac{A_{ik}^2B_{kj}^2}{s^2p_k} - \frac{1}{s^2}(AB)_{ij}^2 \leq \sum_{k=1}^{n}\frac{A_{ik}^2B_{kj}^2}{s^2p_k}$. Thus, $\mathbf{Var}(SR)_{ij} \leq s\sum_{k=1}^{n}\frac{A_{ik}^2B_{kj}^2}{s^2p_k}$. Using $\mathbf{E}((AB-SR)_{ij})=0$ and the lower bound for p_k,

$$\begin{aligned}
\mathbf{E}\left(\|AB-SR\|_F^2\right) &= \sum_{i=1,j=1}^{m,p}\mathbf{E}\left((AB-SR)_{ij}^2\right)\\
=\sum_{i=1,j=1}^{m,p}\mathbf{Var}((SR)_{ij}) &= \frac{1}{s}\sum_{k=1}^{n}\frac{1}{p_k}(\sum_i A_{ik}^2)(\sum_j B_{kj}^2)\\
=\frac{1}{s}\sum_{k=1}^{n}\frac{1}{p_k}|A^{(k)}|^2|B_{(k)}|^2 &\leq \frac{\|A\|_F^2}{\alpha s}\sum_{k=1}^{n}\left(|B^{(k)}|\right)^2\\
&= \frac{1}{\alpha s}\|A\|_F^2\|B\|_F^2
\end{aligned}$$

Proof of Theorem 3.2: We remind the reader that $h^{(t)}$, $t = 1, \ldots, k$ denote the top k left singular vectors of C and $\sigma_t(C)$ the corresponding singular values; since the $h^{(t)}$ are orthogonal, $H^T \cdot H = I$, thus

$$\begin{aligned} \|A - HH^T A\|_F^2 &= \mathbf{Tr}\left((A^T - A^T HH^T)(A - HH^T A)\right) \\ &= \mathbf{Tr}(A^T A) - \mathbf{Tr}(A^T HH^T A) \\ &= \|A\|_F^2 - \sum_{t=1}^{k} |A^T h^{(t)}|^2 \qquad (5.8) \end{aligned}$$

Writing AA^T and CC^T both in a coordinate system with $h^{(1)}, \ldots, h^{(k)}$ as the top k coordinate vectors, we see that $h^{(t)^T}(AA^T - CC^T)h^{(t)}$ is the (t,t) entry of $AA^T - CC^T$. So we have

$$\sum_{t=1}^{k} \left(h^{(t)^T}(AA^T - CC^T)h^{(t)}\right)^2 \le \|AA^T - CC^T\|_F^2$$

or, equivalently (since $C^T h^{(t)} = \sigma_t(C) h^{(t)}$)

$$\sum_{t=1}^{k} \left(|A^T h^{(t)}|^2 - \sigma_t^2(C)\right)^2 \le \|AA^T - CC^T\|_F^2$$

and, using the Cauchy-Schwartz inequality,

$$(5.9) \quad \sum_{t=1}^{k} \left(|A^T h^{(t)}|^2 - \sigma_t^2(C)\right) \ge -\sqrt{k}\|AA^T - CC^T\|_F$$

Applying the Hoffman-Wielandt inequality (see [24]) on the symmetric matrices AA^T and CC^T we see that

$$\begin{aligned} \sum_{t=1}^{k} (\sigma_t(CC^T) - \sigma_t(AA^T))^2 &= \sum_{t=1}^{k} (\sigma_t^2(C) - \sigma_t^2(A))^2 \\ &\le \|AA^T - CC^T\|_F^2 \end{aligned}$$

and, using the Cauchy-Schwartz inequality,

$$(5.10) \quad \sum_{t=1}^{k} \left(\sigma_t^2(C) - \sigma_t^2(A)\right) \ge -\sqrt{k}\|AA^T - CC^T\|_F$$

Using lemma 3.2, we see that

$$(5.11) \quad \mathbf{E}\left(\|AA^T - CC^T\|_F\right) \le (1/\sqrt{\beta c})\|A\|_F^2$$

Adding (5.9) and (5.10), we get

$$\sum_{t=1}^{k} \left(|A^T h^{(t)}|^2 - \sigma_t^2(A)\right) \ge -2\sqrt{k}\|AA^T - CC^T\|_F$$

Thus, using (5.11),

$$\mathbf{E}\left(\sum_{t=1}^{k} |A^T h^{(t)}|^2\right) \ge \sum_{t=1}^{k} \sigma_t^2(A) - 2\sqrt{\frac{k}{\beta c}}\|A\|_F^2$$

The Frobenius norm result of the first statement of the theorem follows by substituting this result to (5.8), since $\|A - A_k\|_F^2 = \|A\|_F^2 - \sum_{t=1}^{k} \sigma_t^2(A)$.

In order to prove (3.7), let $\mathcal{H}_k = \text{range}(H) = \mathbf{span}\left(h^{(1)}, h^{(2)}, \ldots, h^{(k)}\right)$. Let $\mathcal{H}_{m-k}$ be the orthogonal complement of $\mathcal{H}_k$ in $\mathcal{R}^m$:

$$\|A - HH^T A\|_2 = \max_{x \in \mathcal{R}^m, |x|=1} |x^T(A - HH^T A)|$$

But, x can be expressed as $a_1 \cdot y + a_2 \cdot z$, such that $y \in \mathcal{H}_k$, $z \in \mathcal{H}_{m-k}$, $a_1, a_2 \in \mathcal{R}$ and $a_1^2 + a_2^2 = 1$. Thus,

$$\begin{aligned} &\max_{x \in \mathcal{R}^m : |x|=1} (x^T(A - HH^T A) \le \\ \le\ &\max_{y \in \mathcal{H}_k : |y|=1} (|a_1 y^T(A - HH^T A)| \\ +\ &\max_{z \in \mathcal{H}_{m-k} : |z|=1} (|a_2 z^T(A - HH^T A)| \\ \le\ &\max_{y \in \mathcal{H}_k : |y|=1} (|y^T(A - HH^T A)| \\ +\ &\max_{z \in \mathcal{H}_{m-k} : |z|=1} (|z^T(A - HH^T A)| \end{aligned}$$

But, for any $y \in \mathcal{H}_k$, $y^T HH^T$ is equal to y. Thus, $\forall y$, $|y^T(A - HH^T A)| = |y^T A - y^T A| = 0$. Similarly, for any $z \in \mathcal{H}_{m-k}$, $z^T HH^T = 0$. Thus, we are only seeking a bound for $\max_{z \in \mathcal{H}_{m-k}} |z^T A|$. To that effect,

$$\begin{aligned} |z^T A|^2 &= z^T AA^T z = z^T(AA^T - CC^T)z + z^T CC^T z \\ &\le \|AA^T - CC^T\|_F + \sigma_{k+1}^2(C) \end{aligned}$$

The maximum $|z^T C|$ over all $z \in \mathcal{H}_{m-k}$ appears when z is equal to the $k+1$ left singular vector of C. Thus,

$$|A - HH^T A|_2^2 \le \sigma_{k+1}^2(C) + \|AA^T - CC^T\|_F^2$$

Now, AA^T, CC^T are symmetric matrices and a result of perturbation theory (see [24]) states that $|\sigma_{k+1}(AA^T) - \sigma_{k+1}(CC^T)| \le |AA^T - CC^T|_2$. From lemma 3.1, $\mathbf{E}\left(\|AA^T - CC^T\|_F\right) \le (1/\sqrt{\beta c})\|A\|_F^2$. Thus,

$$\begin{aligned} |\sigma_{k+1}(AA^T) - \sigma_{k+1}(CC^T)| &= |\sigma_{k+1}^2(A) - \sigma_{k+1}^2(C)| \\ &\le \frac{1}{\sqrt{\beta c}}\|A\|_F^2 \end{aligned}$$

and the second statement of the theorem follows.