Provable Deterministic Leverage Score Sampling

Dimitris Papailiopoulos
Electrical and Computer Engineering
UT Texas
dimitris@utexas.edu

Anastasios Kyrillidis
School of Computer and Communication Sciences
EPFL
anastasios.kyrillidis@epfl.ch

Christos Boutsidis
Yahoo! Labs
New York, NY
boutsidis@yahoo-inc.com

ABSTRACT

We explain *theoretically* a curious empirical phenomenon: "Approximating a matrix by *deterministically* selecting a subset of its columns with the corresponding largest leverage scores results in a good low-rank matrix surrogate". In this work, we provide a novel theoretical analysis of *deterministic leverage score sampling*. We show that such sampling can be provably as accurate as its randomized counterparts, if the leverage scores follow a moderately steep power-law decay. We support this power-law assumption by providing empirical evidence that such decay laws are abundant in real-world data sets. We then demonstrate empirically the performance of deterministic leverage score sampling, which many times matches or outperforms the state-of-the-art techniques.

Categories and Subject Descriptors

G.1.3 [**Mathematics of Computing**]: Numerical Analysis-Numerical Linear Algebra; E.m [**Data**]: Miscellaneous

Keywords

Subset selection; low-rank matrix approximation; leverage scores; deterministic sampling; power law distributions

1. INTRODUCTION

Recently, there has been a lot of interest on selecting the "best" or "more representative" columns from a data matrix [13, 26]. Qualitatively, these columns reveal the most important information hidden in the underlying matrix structure. This is similar to what principal components carry, as extracted via Principal Components Analysis (PCA) [23]. In sharp contrast to PCA, using actual columns of the data matrix to form a low-rank surrogate offers interpretability, making it more attractive to practitioners and data analysts [33, 5, 34, 26].

To make the discussion precise and to rigorously characterize the "best" columns of a matrix, let us introduce the following *Column Subset Selection Problem* (CSSP).

KDD'14, August 24–27, 2014, New York, NY, USA.

ACM 978-1-4503-2956-9/14/08 ...$15.00.
http://dx.doi.org/10.1145/2623330.2623698.

Column Subset Selection Problem. *Let $\mathbf{A} \in \mathbb{R}^{m\times n}$ and let $c < n$ be a sampling parameter. Find c columns of $\mathbf{A}$ – denoted as $\mathbf{C} \in \mathbb{R}^{m\times c}$ – that minimize*

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\mathrm{F} \quad or \quad \|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_2,$$

where $\mathbf{C}^\dagger$ denotes the Moore-Penrose pseudo-inverse.

State of the art algorithms for the CSSP utilize both deterministic and randomized techniques; we discuss related work in Section 5. Here, we describe two algorithms from prior literature that suffice to highlight our contributions.

A central part of our discussion will involve the *leverage scores* of a matrix $\mathbf{A}$, which we define below.

Definition 1. *[Leverage scores] Let $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ contain the top k right singular vectors of a $m \times n$ matrix $\mathbf{A}$ with rank $\rho = rank(\mathbf{A}) \geq k$. Then, the (rank-$k$) leverage score of the i-th column of $\mathbf{A}$ is defined as*

$$\ell_i^{(k)} = \|[\mathbf{V}_k]_{i,:}\|_2^2, \quad i = 1, 2, \ldots, n.$$

Here, $[\mathbf{V}_k]_{i,:}$ denotes the i-th row of $\mathbf{V}_k$.

One of the first algorithms for column subset selection dates back to 1972: in [21], Joliffe proposes a deterministic sampling of the columns of $\mathbf{A}$ that correspond to the largest leverage scores $\ell_i^{(k)}$, for some $k < \mathrm{rank}(\mathbf{A})$. Although this simple approach has been extremely successful in practice [21, 22, 29, 8], to the best of our knowledge, there has been no theoretical explanation why the approximation errors $\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\mathrm{F}$ and $\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_2$ should be small.

One way to circumvent the lack of a theoretical analysis for the above deterministic algorithm is by utilizing randomization. Drineas et al. [13] proposed the following approach: for a target rank $k < \mathrm{rank}(\mathbf{A})$, define a probability distribution over the columns of $\mathbf{A}$, *i.e.,* the ith column is associated with a probability $p_i = \ell_i^{(k)}/k$; observe that $\sum_i p_i = 1$, since $\sum_i \ell_i^{(k)} = \|\mathbf{V}_k\|_\mathrm{F}^2 = k$. Then, in c independent and identically distributed passes, sample with replacement c columns from $\mathbf{A}$, with probabilities given by p_i. Drineas et al. [13], using results in [30], show that this random subset of columns $\mathbf{C} \in \mathbb{R}^{m\times c}$ approximates $\mathbf{A}$, with constant probability, within relative error: $\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\mathrm{F} \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_\mathrm{F}$, when the number of sampled columns is $c = O(k \log k/\varepsilon^2)$, for some $0 < \varepsilon < 1$. Here, $\mathbf{A}_k \in \mathbb{R}^{m\times n}$ is the best rank-k matrix obtained via the SVD.

There are two important remarks that need to be made: *(i)* the randomized algorithm in [13] yields a matrix estimate that is "near optimal", i.e., has error close to that of the best rank-k approximation; and *(ii)* the above random

sampling algorithm is a straightforward randomized version of the deterministic algorithm of Joliffe [21].

From a practical perspective, the deterministic algorithm of Joliffe [21] is extremely simple to implement, and is computationally efficient. Unfortunately, as of now, it did not admit provable performance guarantees. An important open question [13, 29, 8] is: *Can one simply keep the columns having the largest leverage scores, as suggested in [21], and still have a provably tight approximation?*

1.1 Contributions

In this work, we establish a new theoretical analysis for the deterministic leverage score sampling algorithm of Joliffe [21]. We show that if the leverage scores $\ell_i^{(k)}$ follow a sufficiently steep power-law decay, then this deterministic algorithm has provably similar or better performance to its randomized counterparts (see Theorems 2 and 3 in Section 2). This means that under the power-law decay assumption, deterministic leverage score sampling provably obtains near optimal low-rank approximations and it can be as accurate as the "best" algorithms in the literature [4, 18].

From an applications point of view, we support the power law decay assumption of our theoretical analysis by demonstrating that several real-world data-sets have leverage scores following such decays. We further run several experiments on synthetic and real data sets, and compare deterministic leverage score sampling with the state of the art algorithms for the CSSP. In most experiments, the deterministic algorithm obtains tight low-rank approximations, and is shown to perform similar, if not better, than the state of the art.

1.2 Notation

We use $\mathbf{A}, \mathbf{B}, \ldots$ to denote matrices and $\mathbf{a}, \mathbf{b}, \ldots$ to denote column vectors. $\mathbf{I}_n$ is the $n \times n$ identity matrix; $\mathbf{0}_{m\times n}$ is the $m \times n$ matrix of zeros; $\mathbf{e}_i$ belongs to the standard basis (whose dimensionality will be clear from the context). Let $\mathbf{C} = [\mathbf{a}_{i_1}, \ldots, \mathbf{a}_{i_c}] \in \mathbb{R}^{m\times c}$ contain c columns of $\mathbf{A}$. We can equivalently write $\mathbf{C} = \mathbf{AS}$, where the *sampling matrix* is $\mathbf{S} = [\mathbf{e}_{i_1}, \ldots, \mathbf{e}_{i_c}] \in \mathbb{R}^{n\times c}$. We define the Frobenius and the spectral norm of a matrix as $\|\mathbf{A}\|_{\mathrm{F}}^2 = \sum_{i,j} \mathbf{A}_{ij}^2$ and $\|\mathbf{A}\|_2 = \max_{\mathbf{x}:\|\mathbf{x}\|_2=1} \|\mathbf{Ax}\|_2$, respectively.

2. DETERMINISTIC COLUMN SAMPLING

In this section, we describe the details of the deterministic leverage score sampling algorithm. In Section 3, we state our approximation guarantees. In the remaining of the text, given a matrix $\mathbf{A}$ of rank ρ, we assume that the "target rank" is $k < \rho$. This means that we wish to approximate $\mathbf{A}$ using a subset of $c \geq k$ of its columns, such that the resulting matrix has an error close to that of the best rank-k approximation.

The deterministic leverage score sampling algorithm can be summarized in the following three steps:

Step 1: Obtain $\mathbf{V}_k$, the top-k right singular vectors of $\mathbf{A}$. This can be carried by simply computing the singular value decomposition (SVD) of $\mathbf{A}$ in $O(\min\{m, n\}mn)$ time.

Step 2: Calculate the leverage scores $\ell_i^{(k)}$. For simplicity, we assume that $\ell_i^{(k)}$ are sorted in descending order, hence the columns of $\mathbf{A}$ have the same sorting as well.[1]

[1]Otherwise, one needs to sort them in $O(n \log n)$ time-cost.

Step 3: Output the c columns of $\mathbf{A}$ that correspond to the largest c leverage scores $\ell_i^{(k)}$ such that their sum $\sum_{i=1}^{c} \ell_i^{(k)}$ is more than θ. This ensures that the selected columns have accumulated "energy" at least θ. In this step, we have to carefully pick θ, our *stopping threshold*. This parameter essentially controls the *quality of the approximation*.

In Section 7, we provide some guidance on how the stopping parameter θ should be chosen. Note that, if θ is such that $c < k$, we force $c = k$. This is a necessary step that prevents the error in the approximation from "blowing up" (see Section 7). The exact steps are given in Algorithm 1.

Algorithm 1 `LeverageScoresSampler`$(\mathbf{A}, k, \theta)$

Input: $\mathbf{A} \in \mathbb{R}^{m\times n}$, k, θ

1: Compute $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ (top k right sing. vectors of $\mathbf{A}$)

for $i = 1, 2, \ldots, n$

2: $\quad \ell_i^{(k)} = \|[\mathbf{V}_k]_{i,:}\|_2^2$

end for

without loss of generality, let $\ell_i^{(k)}$'s be sorted:

$$\ell_1^{(k)} \geq \cdots \geq \ell_i^{(k)} \geq \ell_{i+1}^{(k)} \geq \cdots \geq \ell_n^{(k)}.$$

3: Find index $c \in \{1, \ldots, n\}$ such that:

$$c = \operatorname*{argmin}_c \left(\sum_{i=1}^{c} \ell_i^{(k)} > \theta \right).$$

4: If $c < k$, set $c = k$.

Output: $\mathbf{S} \in \mathbb{R}^{n\times c}$ s.t. $\mathbf{AS}$ has the top c columns of $\mathbf{A}$.

Algorithm 1 requires $O(\min\{m, n\}mn)$ arithmetic operations. In the full version of this paper [28], we discuss modifications that improve the running time.

3. APPROXIMATION GUARANTEES

Our main technical innovation is a bound on the approximation error of Algorithm 1 in regard to the CSSP; the proof of the following theorem can be found in Section 6.

Theorem 2. *Let $\theta = k - \varepsilon$, for some $\varepsilon \in (0, 1)$, and let $\mathbf{S}$ be the $n \times c$ output sampling matrix of Algorithm 1. Then, for $\mathbf{C} = \mathbf{AS}$ and $\xi = \{2, F\}$, we have $\|\mathbf{A} - \mathbf{CC}^\dagger\mathbf{A}\|_\xi^2 < (1-\varepsilon)^{-1} \cdot \|\mathbf{A} - \mathbf{A}_k\|_\xi^2$.*

Choosing $\varepsilon \in (0, 1/2)$ implies $(1-\varepsilon)^{-1} \leq 1 + 2\varepsilon$ and, hence, we have a relative-error approximation:

$$\|\mathbf{A} - \mathbf{CC}^\dagger\mathbf{A}\|_\xi^2 < (1 + 2\varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_\xi^2.$$

3.1 Bounding the number of sampled columns

Algorithm 1 extracts at least $c \geq k$ columns of $\mathbf{A}$. However, an upper bound on the number of output columns c is not immediate. We study such upper bounds below.

From Theorem 2, it is clear that the stopping parameter $\theta = k - \varepsilon$ directly controls the number of output columns c. This number, extracted for a specific error requirement ε, depends on the decay of the leverage scores. For example, if the leverage scores decay fast, then we intuitively expect $\sum_{i=1}^{c} \ell_i^{(k)} = k - \varepsilon$ to be achieved for a "small" c.

Let us for example consider a case where the leverage scores follow an extremely fast decay:

$$\ell_1^{(k)} = k - 2k \cdot \varepsilon,$$
$$\ell_2^{(k)} = \ldots = \ell_{2k}^{(k)} = \varepsilon,$$
$$\ell_{2k+1}^{(k)} = \ldots = \ell_n^{(k)} = \frac{\varepsilon}{n-2k}.$$

Then, in this case $\sum_{i=1}^{2k} \ell_i^{(k)} = k - \varepsilon$, and Algorithm 1 outputs the $c = 2k$ columns of $\mathbf{A}$ that correspond to the $2k$ largest leverage scores. Due to Theorem 2, this subset of columns $\mathbf{C} \in \mathbb{R}^{n \times 2k}$ comes with the following guarantee:

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\|_\xi^2 < \frac{1}{1-\varepsilon} \cdot \|\mathbf{A} - \mathbf{A}_k\|_\xi^2.$$

Hence, from the above example, we expect that, when the leverage scores decay fast, a small number of columns of $\mathbf{A}$ will offer a good approximation of the form $\mathbf{C}\mathbf{C}^\dagger\mathbf{A}$.

However, in the worst case Algorithm 1 can output a number of columns c that can be as large as $\Omega(n)$. To highlight this subtle point, consider the case where the leverage scores are uniform $\ell_i^{(k)} = \frac{k}{n}$. Then, one can easily observe that if we want to achieve an error of ε according to Theorem 2, we have to set $\theta = k - \varepsilon$. This directly implies that we need to sample $c > (n/k)\theta$ columns. Hence, if $\varepsilon = o(1)$, then,

$$c \geq (n/k)\theta = (1 - \varepsilon/k)n = \Omega\,(n).$$

Hence, for $\varepsilon \to 0$ we have $c \to n$, which makes the result of Theorem 2 trivial.

We argued above that when the leverage scores decay is "fast" then a good approximation is to be expected with a "small" c. We make this intuition precise below [2]. The next theorem considers the case where the leverage scores follow a power-law decay; the proof can be found in Section 6.

Theorem 3. *Let the leverage scores follow a power-law decay with exponent $\alpha_k = 1 + \eta$, for $\eta > 0$:*

$$\ell_i^{(k)} = \frac{\ell_1^{(k)}}{i^{\alpha_k}}.$$

Then, if we set the stopping parameter to $\theta = k - \varepsilon$, for some ε with $0 < \varepsilon < 1$, the number of sampled columns in $\mathbf{C} = \mathbf{AS}$ that Algorithm 1 outputs is

$$c = \max\left\{ \left(\frac{2k}{\varepsilon}\right)^{\frac{1}{1+\eta}} - 1, \quad \left(\frac{2k}{\eta \cdot \varepsilon}\right)^{\frac{1}{\eta}} - 1, \quad k \right\},$$

and $\mathbf{C}$ achieves the following approximation error

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\|_\xi^2 < \frac{1}{1-\varepsilon} \cdot \|\mathbf{A} - \mathbf{A}_k\|_\xi^2, \quad \textit{for } \xi = \{2, \mathrm{F}\}.$$

3.2 Theoretical comparison to state of the art

We compare the number of chosen columns c in Algorithm 1 to the number of columns chosen in the randomized leverage scores sampling case [13]. The algorithm of [13] requires

$$c = O(k \log k/\varepsilon^2)$$

columns for a relative-error bound with respect to the Frobenius error in the CSSP:

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\|_\mathrm{F}^2 \leq (1+\varepsilon)\|\mathbf{A} - \mathbf{A}_k\|_\mathrm{F}^2.$$

Assuming leverage scores follow a power-law decay, Algorithm 1 requires fewer columns for the same ε when:

$$\max\left\{ \left(\frac{2k}{\varepsilon}\right)^{\frac{1}{1+\eta}}, \quad \left(\frac{2k}{\eta \cdot \varepsilon}\right)^{\frac{1}{\eta}} \right\} < C \cdot \frac{k \log k}{\varepsilon^2},$$

where C is an absolute constant. Hence, under the power law decay, Algorithm 1 offers provably a matrix approximation similar or better than [13].

Let us now compare the performance of Algorithm 1 with [4], which are the current state of the art for the CSSP. Theorem 1.5 in [4] provides a randomized algorithm which selects

$$c = \frac{2k}{\varepsilon}(1 + o(1))$$

columns in $\mathbf{C}$ such that

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\|_\mathrm{F}^2 < (1+\varepsilon) \cdot \|\mathbf{A} - \mathbf{A}_k\|_\mathrm{F}^2$$

holds in expectation. This result is in fact optimal, up to a constant 2, since there is a lower bound indicating that such a relative error approximation is not possible unless $c = k/\varepsilon$ (see Section 9.2 in [4]). The approximation bound of Algorithm 1 is indeed better than the upper/lower bounds in [4] for any $\eta > 1$. We should note here that the lower bound in [4] is for general matrices; however, the upper bound of Theorem 3 is applied to a specific class of matrices whose leverage scores follow a power law decay.

Next, we compare the spectral norm bound of Theorem 3 to the spectral norm bound of Theorem 1.1 in [4], which indicates that there exists a deterministic algorithm selecting $c > k$ columns with error

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger \mathbf{A}\|_2^2 < O\,(n/c) \cdot \|\mathbf{A} - \mathbf{A}_k\|_2^2.$$

This upper bound is also tight, up to constants, since [4] provides a matching lower bound. Notice that a relative error upper bound requires $c = \Omega\,(n/(1+\varepsilon))$ in the general case. However, under the power law assumption in Theorem 3, we provide such a relative error bound with asymptotically fewer columns. To our best knowledge, fixing η to a constant, this is the *first* relative-error bound for the spectral norm version of the CSSP with $c = \mathrm{poly}(k, 1/\varepsilon)$ columns.

4. EXPERIMENTS

In this section, we first provide evidence that power law decays are prevalent in real-world data sets. Then, we investigate the empirical performance of Algorithm 1 on real and synthetic data sets.

Our experiments are not meant to be exhaustive; however, they provide clear evidence that: (*i*) the leverage scores of real world matrices indeed follow "sharp" power law decays; and (*ii*) deterministic leverage score sampling in such matrices is particularly effective.

4.1 Power-law decays in real data sets

We demonstrate the leverage score decay behavior of many real-world data sets. These range from social networks and product co-purchasing matrices to document-term bag-of-words data sets, citation networks, and medical imaging

[2]We chose to analyze in detail the case where the leverage scores follow a power law decay; other models for the leverage scores, example, exponential decay, are also interesting, and will be the subject of the full version of this work.

Dataset	$m \times n$	Description	Dataset	$m \times n$	Description
Amazon	262111 × 262111	Purchase netw. [25]	Citeseer	723131 × 723131	Citation netw. [24]
4square	106218 × 106218	Social netw. [36]	Github	56519 × 120867	Soft. netw. [24]
Gnutella	62586 × 62586	P2P netw. [25]	Google	875713 × 875713	Web conn. [24]
Gowalla	875713 × 875713	Social netw. [24]	LJournal	4847571 × 4847571	Social netw. [25]
Slashdot	82168 × 82168	Social netw. [25]	NIPS	12419 × 1500	Word/Docs [2]
Skitter	1696415 × 1696415	System netw. [24]	CT slices	386 × 53500	CT images [2]
Cora	23166 × 23166	Citation netw. [24]	Writer	81067 × 42714	Writers/Works [24]
Youtube	1134890 × 1134890	Video netw. [25]	YT groups	94238 × 30087	Users/Groups [24]

Table 1: Summary of datasets used in the experiments of Subsection 4.1

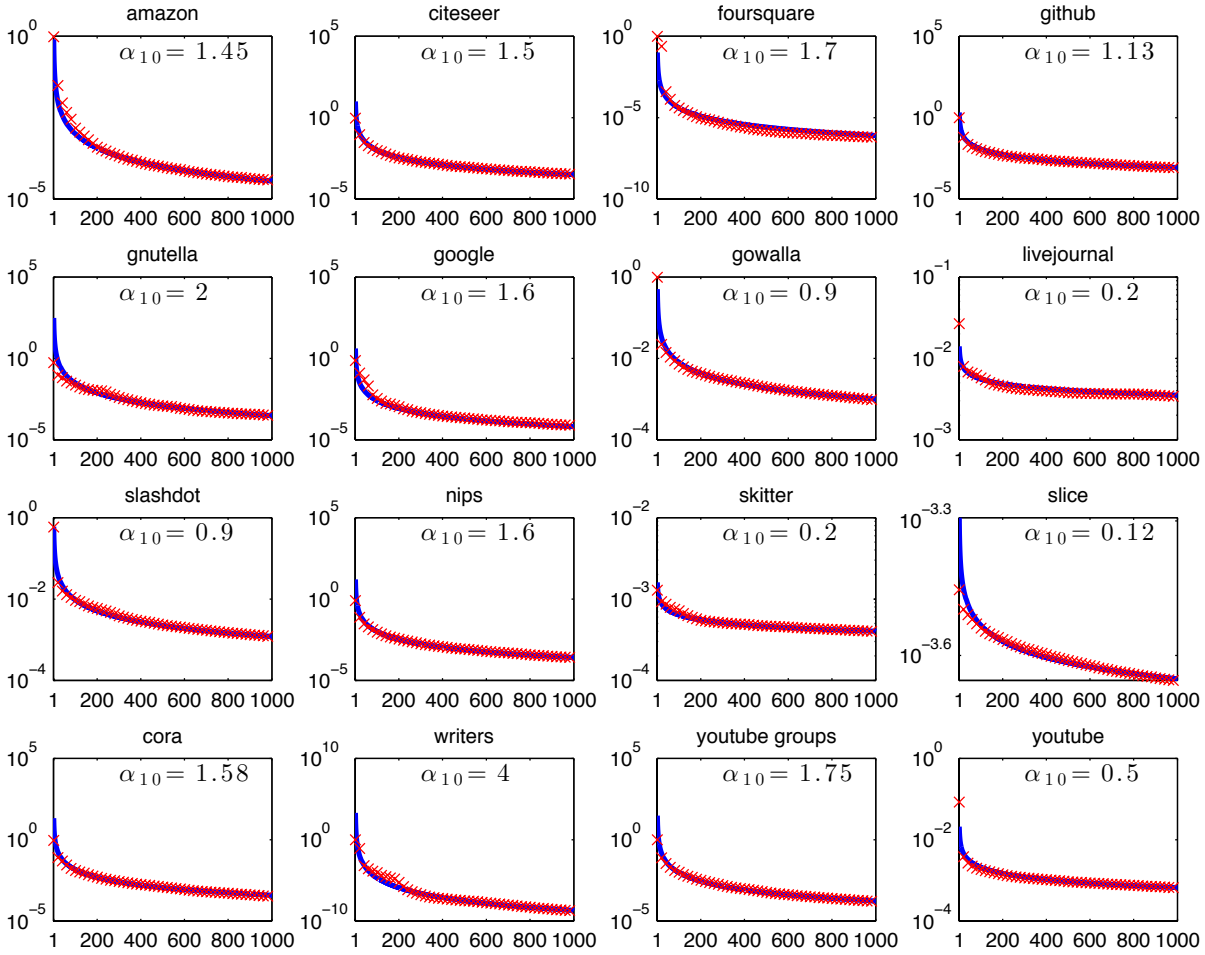

Figure 1: We plot the top $1,000$ leverage scores for 16 different data sets, obtained through $\mathbf{V}_k$ for $k = 10$. The plots are in logarithmic scale. For each data-set, we plot a fitting power-law curve $\beta \cdot x^{-\alpha_k}$. The exponent is listed on each figure as α_{10}. The leverage scores are plotted with a red $\times$ marker, and the fitted curves are denoted with a solid blue line. We observe that the power law fit offers a good approximation of the true leverage scores. We further observe that many data sets exhibit sharp decays ($\alpha_k > 1$), while only a few have leverage scores that decay slowly ($\alpha_k < 1$).

samples. Their dimensions vary from thousands to millions of variables. The data-set description is given in Table 1.

In Figure 1, we plot the top $1,000$ leverage scores extracted from the matrix of the right top-k singular vectors $\mathbf{V}_k$. In all cases we set $k = 10$.[3] For each dataset, we plot a fitting power-law curve of the form $\beta \cdot x^{-\alpha_k}$, where α_k is the exponent of interest.

We can see from the plots that a power law indeed seems to closely match the behavior of the top leverage scores. What is more interesting is that for many of our data sets we observe a decay exponent of $\alpha_k > 1$: this is the regime where deterministic sampling is expected to perform well. It

[3]We performed various experiments for larger k, *e.g.*, $k = 30$ or $k = 100$ (not shown due to space limitations). We found that as we move towards higher k, we observe a "smoothing" of the speed of decay. This is to be expected, since for the case of $k = \text{rank}(\mathbf{A})$ all leverage scores are equal.

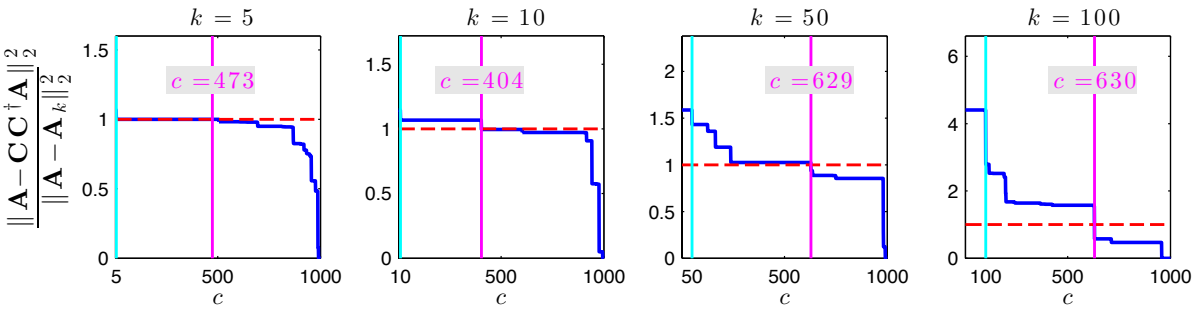

Figure 2: Nearly-uniform leverage scores case: Here, we plot as a blue curve the relative error ratio $\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_2^2/\|\mathbf{A}-\mathbf{A}_k\|_2^2$ achieved by Algorithm 1 as a function of the output columns c. The leftmost vertical cyan line corresponds to the point where $k = c$. When $c < k$ the output error can be large; this justifies why we enforce the algorithm to output $c \geq k$ columns. The rightmost vertical magenta line indicates the point where the c sampled columns offer as good an approximation as that of the best rank-k matrix $\mathbf{A}_k$.

seems that these sharp decays are naturally present in many real-world data sets.

We would like to note that as we move to smaller scores (i.e., after the 10,000-th score), we empirically observe that the leverage scores tail usually decays much faster than a power law. This only helps the bound of Theorem 2.

4.2 Synthetic Experiments

In this subsection, we are interested in understanding the performance of Algorithm 1 on matrices with *(i)* uniform and *(ii)* power-law decaying leverage scores.

To generate matrices with a prescribed leverage score decay, we use the implementation of [20]. Let $\mathbf{V}_k \in \mathbb{R}^{n\times k}$ denote the matrix we want to construct, for some $k < n$. Then, [20] provides algorithms to generate tall-and-skinny orthonormal matrices with specified row norms (i.e., leverage scores). Given the $\mathbf{V}_k$ that is the output of the matrix generation algorithm in [20], we run a basis completion algorithm to find the perpendicular matrix $\mathbf{V}_k^\perp \in \mathbb{R}^{n\times(n-k)}$ such that $\mathbf{V}_k^{\mathrm{T}}\mathbf{V}_k^\perp = \mathbf{0}_{k\times(n-k)}$. Then, we create an $n\times n$ matrix $\mathbf{V} = [\mathbf{V}_k\mathbf{V}_k^\perp]$ where the first k columns of $\mathbf{V}$ are the columns of $\mathbf{V}_k$ and the rest $n-k$ columns are the columns of $\mathbf{V}_k^\perp$; hence, $\mathbf{V}$ is a full orthonormal basis. Finally we generate $\mathbf{A} \in \mathbb{R}^{m\times n}$ as $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$; where $\mathbf{U} \in \mathbb{R}^{m\times m}$ is any orthonormal matrix, and $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$ any diagonal matrix with $\min\{m,n\}$ positive entries along the main diagonal. Therefore, $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$ is the full SVD of $\mathbf{A}$ with leverage scores equal to the squared ℓ_2-norm of the rows of $\mathbf{V}_k$. In our experiments, we pick $\mathbf{U}$ as an orthonormal basis for an $m\times m$ matrix where each entry is chosen i.i.d. from the Gaussian distribution. Also, $\mathbf{\Sigma}$ contains $\min\{m,n\}$ positive entries (sorted) along its main diagonal, where each entry was chosen i.i.d. from the Gaussian distribution.

4.2.1 *Nearly-uniform scores*

We set the number of rows to $m = 200$ and the number of columns to $n = 1000$ and construct $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}} \in \mathbb{R}^{m\times n}$ as described above. The row norms of $\mathbf{V}_k$ are chosen as follows: First, all row norms are chosen equal to k/n, for some fixed k. Then, we introduce a small perturbation to avoid singularities: for every other pair of rows we add $\beta \in \mathcal{N}(0, 1/100)$ to a row norm and subtract the same β from the other row norm – hence the sum of $\ell_i^{(k)}$ equals to k.

We set k to take the values $\{5, 10, 50, 100\}$ and for each k we choose: $c = \{1, 2, \ldots, 1000\}$. We present our findings in Figure 2, where we plot the relative error achieved $\frac{\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_2^2}{\|\mathbf{A}-\mathbf{A}_k\|_2^2}$, where the $n\times c$ matrix $\mathbf{C}$ contains the first c columns of $\mathbf{A}$ that correspond to the k largest leverage scores of $\mathbf{V}_k$, as sampled by Algorithm 1. Then, the leftmost vertical cyan line corresponds to the point where $k = c$, and the rightmost vertical magenta line indicates the point where the c sampled columns achieve an error of $\|\mathbf{A}-\mathbf{A}_k\|_2^2$, where $\mathbf{A}_k$ is the best rank-k approximation.

In the plots of Figure 2, we see that as we move to larger values of k, if we wish to achieve an error of $\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_2^2 \approx \|\mathbf{A}-\mathbf{A}_k\|_2^2$, then we need to keep in $\mathbf{C}$, approximately almost half the columns of $\mathbf{A}$. This agrees with the uniform scores example that we showed earlier in Subsection 3.1. However, we observe that Algorithm 1 can obtain a moderately small relative error, with significantly smaller c. See for example the case where $k = 100$; then, $c \approx 200$ sampled columns suffice for a relative error approximately equal to 2, *i.e.,* $\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_2^2 \approx 2\cdot\|\mathbf{A}-\mathbf{A}_k\|_2^2$. This indicates that our analysis could be loose in the general case.

4.2.2 *Power-law decay*

In this case, our synthetic eigenvector matrices $\mathbf{V}_k$ have leverage scores that follow a power law decay. We choose two power-law exponents: $\alpha_k = 0.5$ and $\alpha_k = 1.5$. Observe that the latter complies with Theorem 3, that predicts the near optimality of leverage score sampling under such decay.

In the first row of Figure 3, we plot the relative error vs. the number of output columns c of Algorithm 1 for $\alpha_k = 0.5$. Then, in the second row of Figure 3, we plot the relative error vs. the number of output columns c of Algorithm 1 for $\alpha_k = 1.5$. The blue line represents the relative error in terms of spectral norm. We can see that the performance of Algorithm 1 in the case of the fast decay is surprising: $c \approx 1.5\cdot k$ suffices for an approximation as good as of that of the best rank-k approximation. This confirms the approximation performance in Theorem 3.

4.3 Comparison with other techniques

We will now compare the proposed algorithm to state of the art approaches for the CSSP, both for $\xi = 2$ and $\xi = \mathrm{F}$. We report results for the errors $\|\mathbf{A}-\mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\xi^2/\|\mathbf{A}-\mathbf{A}_k\|_\xi^2$. A comparison of the running time complexity of those algorithms is out of the scope of our experiments.

Table 2 contains a brief description of the datasets used in our experiments. We employ the datasets used in [15], which

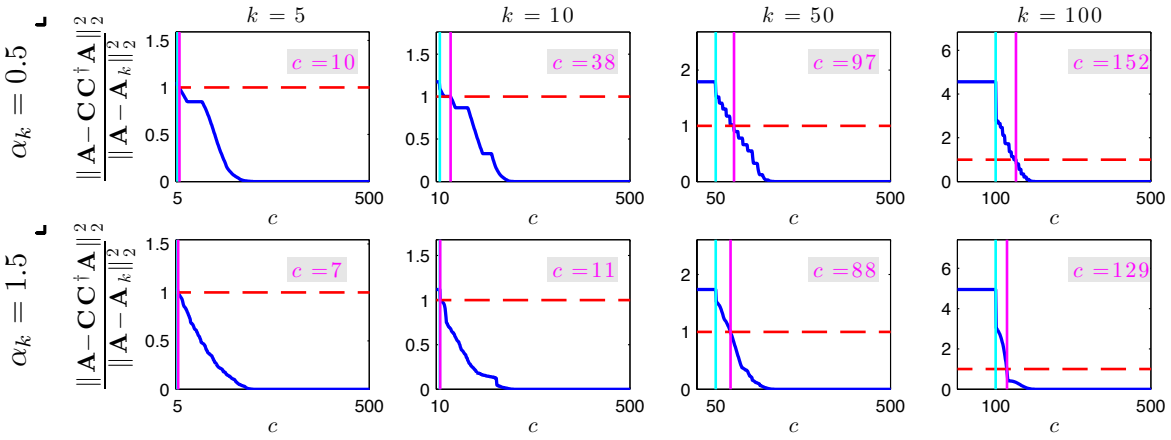

Figure 3: Power-law decaying leverage scores case: We choose two power-law exponents: $\alpha_k = 0.5$ and $\alpha_k = 1.5$. In the first row we plot the relative error of Algorithm 1 vs. c for the first decay profile, and the second row is the error performance of Algorithm 1 for the second, sharpest decay profile. The vertical cyan line corresponds to the point where $k = c$, and the vertical magenta line indicates the point where the c sampled columns offer a better approximation compared to the best rank-k matrix $\mathbf{A}_k$.

Dataset	m	n	rank($\mathbf{A}$)	Description
`Protein`	357	6621	356	Saccharomyces cerevisiae dataset
`SNPS`	46	5523	46	Single Nucleotide - polymorphism dataset
`Enron`	3000	3000	2569	A subgraph of the Enron email graph

Table 2: Summary of datasets used in the experiments of Subsection 4.3 [15]

presents exhaustive experiments for matrix approximations obtained through randomized leverage scores sampling.

4.3.1 *List of comparison algorithms*

We compare Algorithm 1 against three methods for the CSSP. First, the authors in [4] present a near-optimal deterministic algorithm, as described in Theorem 1.2 in [4]. Given $\mathbf{A}$, $k < \text{rank}(\mathbf{A})$ and $c > k$, the proposed algorithm selects $\tilde{c} \leq c$ columns of $\mathbf{A}$ in $\mathbf{C} \in \mathbb{R}^{m\times\tilde{c}}$ with

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\mathrm{F} \leq \left(1 + \left(1 - \sqrt{k/c}\right)^{-1}\right) \|\mathbf{A} - \mathbf{A}_k\|_\mathrm{F}.$$

Second, in [16], the authors present a deterministic pivoted QR algorithm such that:

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_2 \leq \left(1 + \sqrt{n-k}\cdot 2^k\right) \|\mathbf{A} - \mathbf{A}_k\|_2.$$

This bound was proved in [17]. In our tests, we use the `qr`$(\cdot)$ built-in Matlab function, where one can select $c = k$ columns of $\mathbf{A}$ as:

$$[\mathbf{Q},\ \mathbf{R}, \boldsymbol{\pi}] = \texttt{qr}(\mathbf{A}, 0); \quad \mathbf{C} = \mathbf{A}_{:,\boldsymbol{\pi}_{1:c}},$$

where $\mathbf{A} = \mathbf{Q}\mathbf{R}$, $\mathbf{Q} \in \mathbb{R}^{m\times n}$ contains orthonormal columns, $\mathbf{R} \in \mathbb{R}^{n\times n}$ is upper triangular, and $\boldsymbol{\pi}$ is a permutation information vector such that $\mathbf{A}_{:,\boldsymbol{\pi}} = \mathbf{Q}\mathbf{R}$.

Third, we also consider the randomized leverage-scores sampling method with replacement, presented in [13]. According to this work and given $\mathbf{A}, k < \text{rank}(\mathbf{A})$, and $c = \Omega(k\log k)$, the bound provided by the algorithm is

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\mathrm{F} \leq \left(1 + O\left(\sqrt{k\log k/c}\right)\right) \|\mathbf{A} - \mathbf{A}_k\|_\mathrm{F},$$

which holds only with constant probability. In our experiments, we use the software tool developed in [20] for the randomized sampling step.

We use our own Matlab implementation for each of these approaches. For [13], we execute 10 repetitions and report the one that minimizes the approximation error.

4.3.2 *Performance Results*

Table 3 contains a subset of our results; a complete set is reserved for an extended version of this work. We observe that the performance of Algorithm 1 is particularly appealing: It is almost as good as randomized leverage scores sampling in almost all cases - when randomized sampling is better, the difference is on the first or second decimal digit.

Figure 4 shows the leverage scores for the three matrices used in our experiments. Although the decay for the first data sets does not fit a "sharp" power law as it is required in Theorem 3, the performance of the algorithm is still competitive in practice. Interestingly, we do observe good performance for the third data set (Enron). For this case, the power law decay fits the decay profile needed to establish the near optimality of Algorithm 1.

5. RELATED WORK

One of the first deterministic results regarding the CSSP goes back to the seminal work of Gene Golub on pivoted QR factorizations [16]. Similar algorithms have been developed in [16, 19, 10, 11, 17, 35, 32, 3, 27]; see also [6] for a recent survey. The best of these algorithms is the so-called *Strong Rank-revealing QR* (Strong RRQR) algorithm in [17]:

Model			$\frac{\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_2}{\|\mathbf{A}-\mathbf{A}_k\|_2}$			$\frac{\|\mathbf{A}-\mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_F}{\|\mathbf{A}-\mathbf{A}_k\|_F}$			
	k	c	[13]	[16]	This work	[4]	[13]	[16]	This work
Protein	50	51	**1.7334**	2.2621	2.6809	1.1068	1.0888	1.1017	**1.1000**
		118	**1.3228**	1.4274	2.2536	0.9344	**0.9233**	0.9258	0.9259
		186	1.0846	**1.0755**	1.7357	0.7939	0.7377	0.7423	**0.7346**
		253	**0.9274**	0.9281	1.3858	0.6938	0.5461	0.5326	**0.5264**
		320	0.7899	**0.7528**	0.8176	0.5943	0.2831	0.2303	**0.2231**
	100	101	1.8568	**1.8220**	2.5666	1.1789	**1.1506**	1.1558	1.1606
		156	**1.3741**	1.3987	2.4227	0.9928	**0.9783**	0.9835	0.9820
		211	1.3041	**1.1926**	2.3122	0.8182	0.8100	0.7958	**0.7886**
		265	**1.0270**	1.0459	2.0509	0.6241	0.6004	0.5820	**0.5768**
		320	0.9174	**0.8704**	1.8562	0.3752	0.3648	**0.2742**	0.2874
SNPS	5	6	**1.4765**	1.5030	1.5613	1.1831	**1.0915**	1.1030	1.1056
		12	1.2601	**1.2402**	1.2799	1.0524	0.9649	0.9519	**0.9469**
		18	1.0537	**1.0236**	1.1252	1.0183	0.8283	**0.8187**	0.8281
		24	**0.8679**	0.9063	0.9302	0.9537	0.6943	**0.6898**	0.6975
		30	**0.7441**	0.7549	0.8742	0.9558	0.5827	**0.5413**	0.5789
	10	11	1.6459	**1.5206**	1.6329	1.2324	1.1708	1.1500	**1.1413**
		16	**1.3020**	1.4265	1.5939	1.1272	1.0386	**1.0199**	1.0420
		21	1.2789	**1.1511**	1.1676	1.0225	0.9170	**0.8842**	0.9011
		25	1.1022	**1.0729**	1.0935	0.9838	0.8091	**0.7876**	0.8057
		30	0.9968	**0.9256**	1.0020	0.8088	0.6636	**0.6375**	0.6707
Enron	10	11	2.2337	1.8320	**1.7217**	1.1096	1.0992	1.0768	**1.0704**
		83	**1.0717**	1.0821	1.1464	1.0123	0.9381	**0.9094**	0.9196
		156	0.8419	**0.8172**	0.8412	1.0044	0.8692	**0.8091**	0.8247
		228	**0.6739**	0.6882	0.6993	0.9984	0.8096	**0.7311**	0.7519
		300	0.6061	**0.6041**	0.6057	1.0000	0.7628	**0.6640**	0.6837
	20	21	2.1726	**1.9741**	2.1669	1.1344	1.1094	**1.0889**	1.0931
		91	1.3502	**1.3305**	1.3344	1.0194	0.9814	**0.9414**	0.9421
		161	1.0242	1.0504	**1.0239**	0.9999	0.9004	**0.8434**	0.8484
		230	0.9099	0.9025	**0.9006**	0.9730	0.8505	**0.7655**	0.7740
		300	0.8211	0.7941	**0.7936**	0.9671	0.8037	**0.6971**	0.7087
	50	51	2.6520	2.2788	**2.2520**	1.1547	1.1436	**1.1053**	1.1076
		113	1.7454	**1.6850**	1.8122	1.0350	1.0425	**0.9902**	0.9929
		176	**1.3524**	1.4199	1.4673	0.9835	0.9718	**0.8999**	0.9011
		238	1.2588	**1.2303**	1.2450	0.9607	0.9187	**0.8251**	0.8282
		300	1.2209	**1.1014**	1.1239	0.9384	0.8806	**0.7593**	0.7651
	100	101	2.2502	**2.2145**	2.2721	1.1938	1.1805	**1.1223**	1.1238
		151	2.2399	**1.8677**	1.8979	1.0891	1.1122	**1.0357**	1.0393
		201	1.7945	1.6350	**1.6332**	1.0236	1.0631	**0.9646**	0.9664
		250	1.6721	**1.5001**	1.5017	0.9885	1.0026	**0.9025**	0.9037
		300	1.3946	**1.3711**	1.3847	0.9485	0.9672	**0.8444**	0.8467

Table 3: We present the performance of Algorithm 1 as compared to the state of the art in CSSP. We run experiments on 3 data sets described in the above table, for various values of k and c. The performance of Algorithm 1, especially in terms of the Frobenius norm error, is very close to optimal, while at the same time similar, if not better, to the performance of the more sophisticated algorithms of the comparison.

Given $\mathbf{A}$, $c = k$, and constant $f \geq 1$, Strong RRQR requires $O(mnk \log_f n)$ arithmetic operations to find k columns of $\mathbf{A}$ in $\mathbf{C} \in \mathbb{R}^{m \times k}$ that satisfy

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_2 \leq \left(1 + f^2\sqrt{k(n-k)+1}\right) \cdot \|\mathbf{A} - \mathbf{A}_k\|_2.$$

As discussed in Section 1, [21] suggests column sampling with the largest corresponding leverage scores. A related result in [35] suggests column sampling through selection over $\mathbf{V}_k^{\mathrm{T}}$ with Strong RRQR. Notice that the leverage scores sampling approach is similar, but the column selection is based on the largest Euclidean norms of the columns of $\mathbf{V}_k^{\mathrm{T}}$.

From a probabilistic point of view, much work has followed the seminal work of [14] for the CSSP. [14] introduced the idea of randomly sampling columns based on specific probability distributions. [14] use a simple probability distribution where each column of $\mathbf{A}$ is sampled with probability proportional to its Euclidean norm. The approximation bound achieved, which holds only in expectation, is

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\mathrm{F}}^2 \leq \|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 + (k/c)\|\mathbf{A}\|_{\mathrm{F}}^2.$$

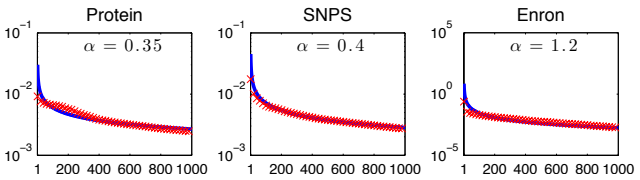

Figure 4: The plots are for $k = 10$ and are in logarithmic scale. The exponent is listed on each figure as α. The leverage scores are plotted with a red $\times$ marker, and the fitted curves are denoted with a solid blue line.

[13] improved upon the accuracy of this result by using a distribution over the columns of $\mathbf{A}$ where each column is sampled with probability proportional to its leverage score. From a different perspective, [12, 18] presented some optimal algorithms using volume sampling. [4] obtained faster optimal algorithms while [7] proposed optimal algorithms that run in input sparsity time.

Another line of research includes row-sampling algorithms for tall-and-skinny orthonormal matrices, which is relevant to our results: we essentially apply this kind of sampling to the rows of the matrix $\mathbf{V}_k$ from the SVD of $\mathbf{A}$. See Lemma 5 in the Section 6 for a precise statement of our result. Similar results exist in [1]. We should also mention the work in [37], which corresponds to a derandomization of the randomized sampling algorithm in [13].

6. PROOFS

Before we proceed, we setup some notation and definitions. For any two matrices $\mathbf{A}$ and $\mathbf{B}$ with appropriate dimensions, $\|\mathbf{A}\|_2 \leq \|\mathbf{A}\|_{\mathrm{F}} \leq \sqrt{\mathrm{rank}(\mathbf{A})}\|\mathbf{A}\|_2$, $\|\mathbf{AB}\|_{\mathrm{F}} \leq \|\mathbf{A}\|_{\mathrm{F}}\|\mathbf{B}\|_2$, and $\|\mathbf{AB}\|_{\mathrm{F}} \leq \|\mathbf{A}\|_2\|\mathbf{B}\|_{\mathrm{F}}$. $\|\mathbf{A}\|_\xi$ indicates that an expression holds for both $\xi = 2, \mathrm{F}$. The thin (compact) Singular Value Decomposition (SVD) of a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $\mathrm{rank}(\mathbf{A}) = \rho$ is:

$$\mathbf{A} = \underbrace{\begin{pmatrix} \mathbf{U}_k & \mathbf{U}_{\rho-k} \end{pmatrix}}_{\mathbf{U_A}\in\mathbb{R}^{m\times\rho}} \underbrace{\begin{pmatrix} \mathbf{\Sigma}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{\Sigma}_{\rho-k} \end{pmatrix}}_{\mathbf{\Sigma_A}\in\mathbb{R}^{\rho\times\rho}} \underbrace{\begin{pmatrix} \mathbf{V}_k^{\mathrm{T}} \\ \mathbf{V}_{\rho-k}^{\mathrm{T}} \end{pmatrix}}_{\mathbf{V_A^T}\in\mathbb{R}^{\rho\times n}},$$

with singular values $\sigma_1(\mathbf{A}) \geq \ldots \sigma_k(\mathbf{A}) \geq \sigma_{k+1}(\mathbf{A}) \geq \ldots \geq \sigma_\rho(\mathbf{A}) > 0$. The matrices $\mathbf{U_A}$ and $\mathbf{V_A}$ contain the left and right singular vectors, respectively. It is well-known that $\mathbf{A}_k = \mathbf{U}_k\mathbf{\Sigma}_k\mathbf{V}_k^{\mathrm{T}} = \mathbf{U}_k\mathbf{U}_k^{\mathrm{T}}\mathbf{A} = \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} \in \mathbb{R}^{m\times n}$ minimizes $\|\mathbf{A} - \mathbf{X}\|_\xi$ over all matrices $\mathbf{X} \in \mathbb{R}^{m\times n}$ of rank at most $k \leq \mathrm{rank}(\mathbf{A})$. The best rank-$k$ approximation to $\mathbf{A}$ satisfies $\|\mathbf{A} - \mathbf{A}_k\|_2 = \sigma_{k+1}(\mathbf{A})$ and $\|\mathbf{A} - \mathbf{A}_k\|_{\mathrm{F}}^2 = \sum_{i=k+1}^{\rho}\sigma_i^2(\mathbf{A})$. $\mathbf{A}^\dagger$ denotes the Moore-Penrose pseudo-inverse of $\mathbf{A}$. Let $\mathbf{B} \in \mathbb{R}^{m\times n}$ $(m \leq n)$ and $\mathbf{A} = \mathbf{B}\mathbf{B}^{\mathrm{T}} \in \mathbb{R}^{m\times m}$; then, for all $i = 1, ..., m$, $\lambda_i(\mathbf{A}) = \sigma_i^2(\mathbf{B})$ is the i-th eigenvalue of $\mathbf{A}$.

6.1 Proof of Theorem 2

To prove Theorem 2, we will use the following result.

LEMMA 4. *[Eqn. 3.2, Lemma 3.1 in [4]] Consider $\mathbf{A} = \mathbf{A}\mathbf{Z}\mathbf{Z}^{\mathrm{T}} + \mathbf{E} \in \mathbb{R}^{m\times n}$ as a low-rank matrix factorization of $\mathbf{A}$, with $\mathbf{Z} \in \mathbb{R}^{n\times k}$, and $\mathbf{Z}^{\mathrm{T}}\mathbf{Z} = \mathbf{I}_k$. Let $\mathbf{S} \in \mathbb{R}^{n\times c}$ $(c \geq k)$ be any matrix such that*

$$\mathit{rank}(\mathbf{Z}^{\mathrm{T}}\mathbf{S}) = k.$$

Let $\mathbf{C} = \mathbf{AS} \in \mathbb{R}^{m\times c}$. Then, for $\xi = 2, \mathrm{F}$:

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\xi^2 \leq \|\mathbf{E}\|_\xi^2 \cdot \|\mathbf{S}(\mathbf{Z}^{\mathrm{T}}\mathbf{S})^\dagger\|_2^2.$$

We will also use the following novel lower bound on the smallest singular value of the matrix $\mathbf{V}_k$, after deterministic selection of its rows based on the largest leverage scores.

LEMMA 5. *Repeat the conditions of Theorem 2. Then,*

$$\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{S}) > 1 - \varepsilon.$$

PROOF. We use the following perturbation result on the sum of eigenvalues of symmetric matrices.

LEMMA 6. *[Theorem 2.8.1; part (i) in [9]] Let $\mathbf{X}$ and $\mathbf{Y}$ be symmetric matrices of order k and, let $1 \leq i, j \leq n$ with $i + j \leq k + 1$. Then,*

$$\lambda_i(\mathbf{X}) \geq \lambda_{i+j-1}(\mathbf{X} + \mathbf{Y}) - \lambda_j(\mathbf{Y}).$$

Let $\mathbf{S} \in \mathbb{R}^{n\times c}$ sample c columns from $\mathbf{A}$ with $c \geq k$. Similarly, let $\hat{\mathbf{S}} \in \mathbb{R}^{n\times(n-c)}$ sample the rest $n - c$ columns from $\mathbf{A}$. Hence,

$$\mathbf{I}_k = \mathbf{V}_k^{\mathrm{T}}\mathbf{V}_k = \mathbf{V}_k^{\mathrm{T}}\mathbf{S}\mathbf{S}^{\mathrm{T}}\mathbf{V}_k + \mathbf{V}_k^{\mathrm{T}}\hat{\mathbf{S}}\hat{\mathbf{S}}^{\mathrm{T}}\mathbf{V}_k.$$

Let

$$\mathbf{X} = \mathbf{V}_k^{\mathrm{T}}\mathbf{S}\mathbf{S}^{\mathrm{T}}\mathbf{V}_k, \mathbf{Y} = \mathbf{V}_k^{\mathrm{T}}\hat{\mathbf{S}}\hat{\mathbf{S}}^{\mathrm{T}}\mathbf{V}_k, \; i = k, \quad \text{and} \quad j = 1,$$

in Lemma 6. Notice that $i + j \leq k + 1$, and $\lambda_k(\mathbf{X} + \mathbf{Y}) = 1$; hence:

$$\begin{aligned} \lambda_k(\mathbf{V}_k^{\mathrm{T}}\mathbf{S}\mathbf{S}^{\mathrm{T}}\mathbf{V}_k) &\geq 1 - \lambda_1(\mathbf{V}_k^{\mathrm{T}}\hat{\mathbf{S}}\hat{\mathbf{S}}^{\mathrm{T}}\mathbf{V}_k) \\ &= 1 - \|\mathbf{V}_k^{\mathrm{T}}\hat{\mathbf{S}}\|_2^2 \\ &\geq 1 - \|\mathbf{V}_k^{\mathrm{T}}\hat{\mathbf{S}}\|_{\mathrm{F}}^2 \\ &> 1 - (k - \theta) \end{aligned}$$

Replacing $\theta = k - \varepsilon$ and $\lambda_k(\mathbf{V}_k^{\mathrm{T}}\mathbf{S}\mathbf{S}^{\mathrm{T}}\mathbf{V}_k) = \sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{S})$ concludes the proof. ■

The proof of Theorem 2 is a straightforward combination of the Lemmas 4 and 5. First, by picking $\mathbf{Z} = \mathbf{V}_k$ in Lemma 4 we obtain:

$$\begin{aligned} \|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\xi^2 &\leq \|\mathbf{A} - \mathbf{A}_k\|_\xi^2 \cdot \|\mathbf{S}(\mathbf{V}_k^{\mathrm{T}}\mathbf{S})^\dagger\|_2^2 \\ &\leq \|\mathbf{A} - \mathbf{A}_k\|_\xi^2 \cdot \|\mathbf{S}\|_2^2 \cdot \|(\mathbf{V}_k^{\mathrm{T}}\mathbf{S})^\dagger\|_2^2 \\ &= \|\mathbf{A} - \mathbf{A}_k\|_\xi^2 \cdot \|(\mathbf{V}_k^{\mathrm{T}}\mathbf{S})^\dagger\|_2^2 \\ &= \|\mathbf{A} - \mathbf{A}_k\|_\xi^2/\sigma_k^2(\mathbf{V}_k^{\mathrm{T}}\mathbf{S}) \end{aligned}$$

In the above, we used the facts that

$$\mathbf{E} = \mathbf{A} - \mathbf{A}\mathbf{V}_k\mathbf{V}_k^{\mathrm{T}} = \mathbf{A} - \mathbf{A}_k,$$

and the spectral norm of the sampling matrix $\mathbf{S}$ equals one. Also, we used that $\mathrm{rank}(\mathbf{V}_k^{\mathrm{T}}\mathbf{S}) = k$, which is implied from Lemma 5. Next, via the bound in Lemma 5:

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^\dagger\mathbf{A}\|_\xi^2 < \|\mathbf{A} - \mathbf{A}_k\|_\xi^2/(1 - \varepsilon).$$

6.2 Proof of Theorem 3

Let $\alpha_k = 1 + \eta$ for some $\eta > 0$. We assume that the leverage scores follow a power law decay such that: $\ell_i^{(k)} = \ell_1^{(k)}/i^{1+\eta}$. According to the proposed algorithm, we select c columns such that $\sum_{i=1}^{c}\ell_i^{(k)} > \theta$. Here, we bound the number of columns c required to achieve an $\varepsilon := k - \theta$ approximation in Theorem 2. To this end, we use the extreme case $\sum_{i=1}^{c}\ell_i^{(k)} = \theta$ which guarantees an $(1 + \varepsilon)$-approximation.

For our analysis, we use the following well-known result.

PROPOSITION 7. *[Integral test for convergence] Let $f(\cdot) \geq 0$ be a function defined over the set of positive reals. Furthermore, assume that $f(\cdot)$ is monotone decreasing. Then,*

$$\int_j^{J+1} f(i)di \;\leq\; \sum_{i=j}^{J} f(i) \;\leq\; f(j) + \int_j^J f(x)dx,$$

over the interval $[j, \ldots, J]$ for j, J positive integers.

In our case, consider $f(i) = \frac{1}{i^{1+\eta}}$. By definition of the leverage scores, we have:

$$k = \sum_{i=1}^{n} \ell_i^{(k)} = \ell_1^{(k)} \sum_{i=1}^{n} \frac{1}{i^{1+\eta}} \Longrightarrow \ell_1^{(k)} = \frac{k}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}}.$$

By construction, we collect c leverage scores such that $k - \theta = \varepsilon$. This leads to:

$$\begin{aligned} k - \varepsilon = \ell_1^{(k)} \cdot \sum_{i=1}^{c} \frac{1}{i^{1+\eta}} &= \frac{k}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} \cdot \sum_{i=1}^{c} \frac{1}{i^{1+\eta}} \\ &= k \left(\frac{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}} - \sum_{i=c+1}^{n} \frac{1}{i^{1+\eta}}}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} \right) \\ &= k \left(1 - \frac{\sum_{i=c+1}^{n} \frac{1}{i^{1+\eta}}}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} \right) \Longrightarrow \\ \varepsilon = k \cdot \frac{\sum_{i=c+1}^{n} \frac{1}{i^{1+\eta}}}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}}. \end{aligned}$$

To bound the quantity on the right hand side, we observe

$$\begin{aligned} \frac{\sum_{i=c+1}^{n} \frac{1}{i^{1+\eta}}}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} &\leq \frac{\frac{1}{(c+1)^{1+\eta}} + \int_{i=c+1}^{n} \frac{1}{x^{1+\eta}} dx}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} \\ &= \frac{\frac{1}{(c+1)^{1+\eta}} + \left[-\frac{1}{x^{1+\eta}} \right]_{i=c+1}^{n}}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} \\ &= \frac{\frac{1}{(c+1)^{1+\eta}} + \frac{1}{\eta} \left(\frac{1}{(c+1)^{\eta}} - \frac{1}{n^{\eta}} \right)}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} \\ &\leq \frac{\frac{1}{(c+1)^{1+\eta}} + \frac{1}{\eta(c+1)^{\eta}}}{\sum_{i=1}^{n} \frac{1}{i^{1+\eta}}} \\ &= \frac{\frac{1}{(c+1)^{1+\eta}} + \frac{1}{\eta(c+1)^{\eta}}}{1 + \sum_{i=2}^{n} \frac{1}{i^{1+\eta}}} \\ &< \frac{1}{(c+1) \cdot (c+1)^{\eta}} + \frac{1}{\eta(c+1)^{\eta}} \\ &\leq \max \left\{ \frac{2}{(c+1)^{1+\eta}}, \quad \frac{2}{\eta \cdot (c+1)^{\eta}} \right\} \end{aligned}$$

where the first inequality is due to the right hand side of the integral test and the third inequality is due to $1 + \sum_{i=2}^{n} \frac{1}{i^{1+\eta}} > 1$. Hence, we may conclude:

$$\varepsilon < k \cdot \max \left\{ \frac{2}{(c+1)^{1+\eta}}, \quad \frac{2}{\eta \cdot (c+1)^{\eta}} \right\}.$$

The above lead to the following two cases: if

$$\varepsilon < \frac{2k}{(c+1)^{1+\eta}},$$

we have:

$$c < \left(\frac{2 \cdot k}{\varepsilon} \right)^{\frac{1}{1+\eta}} - 1,$$

whereas in the case where

$$\varepsilon < \frac{2 \cdot k}{\eta \cdot (c+1)^{\eta}},$$

we get

$$c < \left(\frac{2 \cdot k}{\eta \cdot \varepsilon} \right)^{\frac{1}{\eta}} - 1.$$

7. THE KEY ROLE OF θ

In the proof of Theorem 2, we require that

$$\sigma_k^2(\mathbf{V}_k^{\mathrm{T}} \mathbf{S}) > 1 - (k - \theta) := 1 - \varepsilon.$$

For this condition to hold, the sampling matrix $\mathbf{S}$ should preserve the rank of $\mathbf{V}_k^{\mathrm{T}}$ in $\mathbf{V}_k^{\mathrm{T}} \mathbf{S}$, i.e., choose θ such that $\mathrm{rank}(\mathbf{V}_k^{\mathrm{T}} \mathbf{S}) = k$.

Failing to preserve the rank of $\mathbf{V}_k^{\mathrm{T}}$ has immediate implications for the CSSP. To highlight this, let $\mathbf{A} \in \mathbb{R}^{m \times n}$ of rank $k < \min\{m, n\}$ with SVD $\mathbf{A} = \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}}$. Further, assume that the kth singular value of $\mathbf{A}$ is arbitrary large, i.e., $\sigma_k(\mathbf{A}) \to \infty$. Also, let $\mathrm{rank}(\mathbf{V}^{\mathrm{T}} \mathbf{S}) = \gamma < k$ and $\mathbf{C} = \mathbf{A}\mathbf{S}$. Then,

$$\begin{aligned} &\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\xi} \\ &= \|\mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} - \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S} (\mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S})^{\dagger} \mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}}\|_{\xi} \\ &= \|\mathbf{\Sigma}_k - \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S} (\mathbf{U}_k \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S})^{\dagger} \mathbf{U}_k \mathbf{\Sigma}_k\|_{\xi} \\ &= \|\mathbf{\Sigma}_k - \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S} (\mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S})^{\dagger} (\mathbf{U}_k)^{\dagger} \mathbf{U}_k \mathbf{\Sigma}_k\|_{\xi} \\ &= \|\mathbf{\Sigma}_k - \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S} (\mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S})^{\dagger} \mathbf{\Sigma}_k\|_{\xi} \\ &= \|\mathbf{\Sigma}_k - \mathbf{U}_{\mathbf{X}} \mathbf{U}_{\mathbf{X}}^{\mathrm{T}} \mathbf{\Sigma}_k\|_{\xi} \\ &\geq \sigma_k(\mathbf{A}) \end{aligned}$$

The second equality is due to the fact that both spectral and Frobenius norms are invariant to unitary transformations. In the third equality, we used the fact that $(\mathbf{W}\mathbf{Z})^{\dagger} = \mathbf{Z}^{\dagger}\mathbf{W}^{\dagger}$ if $\mathbf{W}^{\mathrm{T}}\mathbf{W}$ is the identity matrix. Then, set $\mathbf{X} = \mathbf{\Sigma}_k \mathbf{V}_k^{\mathrm{T}} \mathbf{S} \in \mathbb{R}^{k \times c}$ where $\mathrm{rank}(\mathbf{X}) = \gamma$. Using this notation, let $\mathbf{U}_{\mathbf{X}} \in \mathbb{R}^{m \times \gamma}$ be any orthonormal basis for $span(\mathbf{X})$. Observe $\mathbf{U}_{\mathbf{X}} \mathbf{U}_{\mathbf{X}}^{\mathrm{T}} = \mathbf{X}\mathbf{X}^{\dagger}$. The last inequality is due to $\mathbf{U}_{\mathbf{X}} \mathbf{U}_{\mathbf{X}}^{\mathrm{T}}$ being an $m \times m$ diagonal matrix with γ ones along its main diagonal and the rest zeros. Thus, we may conclude that for this $\mathbf{A}$:

$$\|\mathbf{A} - \mathbf{C}\mathbf{C}^{\dagger}\mathbf{A}\|_{\xi} \geq \sigma_k(\mathbf{A}) \to \infty.$$

8. CONCLUDING REMARKS

We provided a rigorous theoretical analysis of an old and popular *deterministic* feature selection algorithm from the statistics literature [21]. Although randomized algorithms are often easier to analyze, we believe that deterministic algorithms are simpler to implement and explain, hence more attractive to practitioners and data analysts.

One interesting path for future research is understanding the connection of this work with the so-called "spectral graph sparsification problem" [31]. In that case, edge selection in a graph is implemented via randomized leverage scores sampling from an appropriate matrix (see Theorem 1 in [31]). Note that in the context of graph sparsification, leverage scores correspond to the so-called "effective resistances" of the graph. Can deterministic effective resistances sampling be rigorously analyzed? What graphs have effective resistances following a power law distribution?

References

[1] H. Avron and C. Boutsidis. Faster subset selection for matrices and applications. *SIAM Journal on Matrix Analysis and Applications (SIMAX)*, 2013.

[2] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[3] C. H. Bischof and G. Quintana-Ortí. Computing rank-revealing QR factorizations of dense matrices. *ACM Trans. Math. Softw*, 24(2):226–253, 1998.

[4] C. Boutsidis, P. Drineas, and M. Magdon-Ismail. Near optimal column based matrix reconstruction. *SIAM Journal on Computing (SICOMP)*, 2013.

[5] C. Boutsidis, M. W. Mahoney, and P. Drineas. Unsupervised feature selection for principal components analysis. In *KDD*, pages 61–69, 2008.

[6] C. Boutsidis, M. W. Mahoney, and P. Drineas. An improved approximation algorithm for the column subset selection problem. In *SODA*, pages 968–977, 2009.

[7] C. Boutsidis and D. Woodruff. Optimal cur matrix decompositions. *In STOC*, 2014.

[8] M. E. Broadbent, M. Brown, K. Penner, I. Ipsen, and R. Rehman. Subset selection algorithms: Randomized vs. deterministic. *SIAM Undergraduate Research Online*, 3:50–71, 2010.

[9] A. A. E. Brouwer and W. H. Haemers. *Spectra of graphs*. Springer, 2012.

[10] T. F. Chan and P. C. Hansen. Low-rank revealing QR factorizations. *Numerical Linear Algebra with Applications*, 1:33–44, 1994.

[11] S. Chandrasekaran and I. C. F. Ipsen. On rank-revealing factorizations. *SIAM J. Matrix Anal. Appl.*, 15:592–622, 1994.

[12] A. Deshpande and L. Rademacher. Efficient volume sampling for row/column subset selection. In *Proceedings of the 42th Annual ACM Symposium on Theory of Computing (STOC)*, 2010.

[13] P. Drineas, M. W. Mahoney, and S. Muthukrishnan. Relative-error cur matrix decompositions. *SIAM Journal Matrix Analysis and Applications*, 30(2):844–881, 2008.

[14] A. Frieze, R. Kannan, and S. Vempala. Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM*, 51(6):1025–1041, 2004.

[15] A. Gittens and M. W. Mahoney. Revisiting the nystrom method for improved large-scale machine learning. In *ICML (3)*, pages 567–575, 2013.

[16] G. H. Golub. Numerical methods for solving linear least squares problems. *Numer. Math.*, 7:206–216, 1965.

[17] M. Gu and S. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM Journal on Scientific Computing*, 17:848–869, 1996.

[18] V. Guruswami and A. K. Sinop. Optimal column-based low-rank matrix reconstruction. In *SODA*. SIAM, 2012.

[19] Y. P. Hong and C. T. Pan. Rank-revealing QR factorizations and the singular value decomposition. *Mathematics of Computation*, 58:213–232, 1992.

[20] I. C. Ipsen and T. Wentworth. The effect of coherence on sampling from matrices with orthonormal columns, and preconditioned least squares problems. *arXiv preprint arXiv:1203.4809*, 2012.

[21] I. Jolliffe. Discarding variables in a principal component analysis. i: Artificial data. *Applied Statistics*, 21(2):160–173, 1972.

[22] I. Jolliffe. Discarding variables in a principal component analysis. ii: Real data. *Applied Statistics*, 22(1):21–31, 1973.

[23] I. Jolliffe. *Principal Component Analysis*. Springer; 2nd edition, 2002.

[24] J. Kunegis. Konect: the koblenz network collection. In *WWW*. International World Wide Web Conferences Steering Committee, 2013.

[25] J. Leskovec. Snap stanford network analysis project. 2009.

[26] M. W. Mahoney and P. Drineas. Cur matrix decompositions for improved data analysis. *Proceedings of the National Academy of Sciences*, 106(3):697–702, 2009.

[27] C. T. Pan. On the existence and computation of rank-revealing LU factorizations. *Linear Algebra and its Applications*, 316:199–222, 2000.

[28] D. Papapailiopoulos, A. Kyrillidis, and C. Boutsidis. Provable Deterministic Leverage Score Sampling. *arXiv preprint arXiv:1404.1530,* 2014

[29] P. Paschou, E. Ziv, E. G. Burchard, S. Choudhry, W. Rodriguez-Cintron, M. W. Mahoney, and P. Drineas. Pca-correlated snps for structure identification in worldwide human populations. *PLoS genetics*, 3(9):e160, 2007.

[30] M. Rudelson and R. Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *JACM: Journal of the ACM*, 54, 2007.

[31] N. Srivastava and D. Spielman. Graph sparsifications by effective resistances. In *STOC*, 2008.

[32] G. Stewart. Four algorithms for the efficient computation of truncated QR approximations to a sparse matrix. *Numerische Mathematik*, 83:313–323, 1999.

[33] J. Sun, Y. Xie, H. Zhang, and C. Faloutsos. Less is more: Compact matrix decomposition for large sparse graphs. In *SDM*. SIAM, 2007.

[34] H. Tong, S. Papadimitriou, J. Sun, P. S. Yu, and C. Faloutsos. Colibri: fast mining of large static and dynamic graphs. In *KDD*. ACM, 2008.

[35] E. Tyrtyshnikov. Mosaic-skeleton approximations. *Calcolo*, 33(1):47–57, 1996.

[36] R. Zafarani and H. Liu. Social computing data repository at ASU, 2009.

[37] A. Zouzias. A matrix hyperbolic cosine algorithm and applications. In *ICALP*. Springer, 2012.