Near-optimal Distributions for Data Matrix Sampling

Dimitris Achlioptas*
Department of Computer Science
University of California Santa Cruz
Santa Cruz, CA 95046
optas@cs.ucsc.edu

Zohar Karnin
Yahoo! Research
Address
zkarnin@yahoo-inc.com

Edo Liberty
Yahoo! Research
Address
edo.liberty@ymail.com

Abstract

We give near-optimal distributions for the sparsification of large $m \times n$ matrices, where $m \ll n$, for example representing n observations over m attributes. Our algorithms can be applied when the non-zero entries are only available as a stream, i.e., in arbitrary order, and result in matrices which are not only sparse, but whose values are also highly compressible. In particular, algebraic operations with the resulting matrices can be implemented as (ultra-efficient) operations over indices.

1 Introduction

Given an $m \times n$ matrix A, it is often desirable to find a sparser matrix B that is a good proxy for A. Besides being an extremely natural mathematical question, such sparsification has become ubiquitous preprocessing in a number of data analysis operations. A fruitful measure for the approximation of A by B is the spectral norm of $A - B$, where for a matrix C its spectral norm $\|C\|_2 = \max_{\|x\|_2=1} \|Cx\|_2$. Randomization has been key in achieving sparsification and the problem is typically cast as follows: given a matrix A and a budget s, devise a distribution over matrices B such that the (expected) number of non-zero entries in B is s and $\|A-B\|_2$ is as small as possible.

Our work is motivated by typical big data matrices generated by some measurement process of a number of attributes (rows), each column corresponding to an observation. Thus, $m \ll n$ and, typically, the total number of non-zero entries in A exceeds the available main memory by several orders of magnitude. A key consideration in this context is the access model for A. On one end of the spectrum, A is stored in durable storage and there is no bound on the time to construct the probability distribution over matrices B. On the other end, a priori we know only some very basic features of A, e.g., it's number of rows, but the actual non-zero entries are presented to us one at a time in an arbitrary order. For each entry we must decide whether to keep it or not upon presentation.

We work with the second option above, also known as the *streaming model*. The first reason for this choice is pragmatic: for an increasing number of data analysis applications the streaming model *is* reality. The second is that even when A exists in durable storage, random access to its entries is prohibitively expensive and, in practice, one can only afford a small number of passes over A. Focusing on streaming allows us to establish that for matrices with certain features that invariantly arise in massive data collection, methods similar to the one below give *provably* near-optimal sparsification:

*Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies.

1. Retain each entry of A independently of all other entries with probability
$$P_{ij} \quad = \quad \theta_i \cdot |A_{ij}| \ , \tag{1}$$
where θ_i is a simple function of the entire i-th row.
2. Set $B_{ij} = A_{ij}/P_{ij}$ for the retained entries and $B_{ij} = 0$ for all other entries.

As we will see, the incredibly simple form (1) of the probabilities P_{ij} falls out naturally from generic optimization considerations. Besides being interesting on its own, when combined with the fact that B is an unbiased estimator of A, i.e., $B_{ij} = A_{ij}/P_{ij}$, this fact has a remarkably practical implication: every entry in the i-th row of B will belong in $\{-\theta_i, 0, +\theta_i\}$. More generally, other methods considered will give $B_{ij} = k_{ij} \cdot \theta_i$, where $k_{ij} \in \mathbb{Z}$, typically $k_{ij} \in \{-1, 0, +1\}$. Thus, we will see that the values of the non-zero entries of B can be represented using only one real number θ_i per row (attribute), and typically *a single bit* per non-zero entry (no bits if the matrix is positive).

Perhaps more importantly, the resulting matrix can easily be stored as a standard search index where each row (or column) corresponds to an indexed object, e.g., document. With such a representation, multiplying B by a vector corresponds to issuing a search query which, assuming the query is sparse, is extremely efficient. Moreover, we benefit from utilizing highly optimized compression schemes. In a simple experiment, we measured the average number of bits per sample (total size of the sketch divided by the number of samples s). The results were between 5 and 22 bits per sample depending on the matrix and s. It is important to note that the number of bits per sample is usually less than $\log(n) + \log(m)$ which is the minimal number of bit required to represent a pair (i, j). This is because only non zero index *offsets* are stored. The result is a reduction of disc space by a factor of between 2 and 5, depending on the number of samples and the matrix, relative to the compressed size of the standard row-column-value list format.

1.1 Measuring the Error

The reason for measuring the difference of A from B with respect to the L2 norm is that this is not only the most demanding, but also extremely relevant in the context of data analysis. Let us define a *linear trend* in the data of A as any tendency of the rows to align with a particular unit vector x. To examine the presence of such a trend, we need only multiply A with x: the ith coordinate of Ax is the projection of the ith row of A onto x. Thus, $\|Ax\|_2$ measures the strength of a linear trend x in A, $\|A\|_2$ measures the strongest linear trend in A, and thus, minimizing $\|A - B\|_2$ minimizes the strength of the strongest linear trend of A *not captured* by B.

In contrast, imagine that A is a 0/1 matrix, a case not uncommon in practice, and that we measured $\|A - B\|$ with respect to the Frobenius norm. In such a case, the quality of the approximation is *completely independent* of which elements of A we keep in B and only depends only on how many we keep. This is clearly bad since, assuming A does contain some structure, certain approximations are far better than others.

1.2 Our Approach

Even after committing: (i) to measuring the quality of B by $\|A - B\|_2$, (ii) to sampling the entries of A independently (enabling working with streams), and (iii) to requiring B to be an unbiased estimator of A, one is still left with the task of determining a good probability distribution P_{ij} from which to sample the entries of A in order to get B. At least two natural candidates come to mind:

- **L1-sampling:** $P_{ij} \ \propto \ |A_{ij}|$.
- **L2-sampling:** $P_{ij} \ \propto \ A_{ij}^2$.

Good arguments can and have been made in favor of both these distributions and both of them have been investigated intensively in earlier works, as we discuss in Section 3. Therein we will see that already some of the algorithmic design in the context of matrix sparsification has been guided by beautiful results in the theory of random matrices. In this work we have taken this approach to its logical conclusion. Specifically, rather than proposing a specific sampling distribution and using results from random matrix theory to demonstrate that it has good properties, we start from a cornerstone result in random matrix theory, Matrix-Bernstein's inequality (see e.g [1], Theorem 1.6) for sums of independent random matrices, and work backwards to reverse-engineer near-optimal distributions with respect to the notion of probabilistic deviations captured by the inequality.

Theorem 1.1 (Matrix Bernstein inequality). *Consider a finite sequence $\{X_i\}$ of i.i.d. random $m \times n$ matrices, where $\mathbb{E}[X_1] = 0$ and $\|X_1\| \leqslant R$. Let $\sigma^2 = \max\left\{\|\mathbb{E}[X_1 X_1^T]\|, \|\mathbb{E}[X_1^T X_1]\|\right\}$.*

For some fixed $s \geqslant 1$, let $X = (X_1 + \cdots + X_s)/s$. For all $\varepsilon \geqslant 0$,

$$\Pr[\|X\| \geqslant \varepsilon] \leqslant (m+n) \exp\left(-\frac{s\varepsilon^2}{\sigma^2 + R\varepsilon/3}\right) \quad .$$

Imagine for a moment that, having fixed the probability distribution P over the non-zero elements of A, we take B to be a random $m \times n$ matrix with exactly one non-zero element, formed by sampling an element a_{ij} of A according to P and letting $B_{ij} = a_{ij}/P_{ij}$. Since $\mathbb{E}[B_{ij}] = a_{ij}$ for every i, j, we see that B is an unbiased estimator of A, i.e., $\mathbb{E}[B] = A$. Clearly, the same is true if we repeat this s times taking i.i.d. samples $B_1, \ldots, B_s$ and let their average be our matrix B. Therefore, our goal to find a distribution P minimizing $\|E\| = \|A - (B_1 + \cdots + B_s)/s\|$. Writing $sE = (A - B_1) + \cdots + (A - B_s)$ we see that $s\|E\|$ is the operator norm of a sum of i.i.d. zero-mean random matrices $X_i = A - B_i$, i.e., exactly the setting of Theorem 1.1. The relevant parameters are

$$\sigma^2 = \max\left\{\|\mathbb{E}[(A - B_1)(A - B_1)^T]\|, \|\mathbb{E}[(A - B_1)^T(A - B_1)]\|\right\} \qquad (2)$$

$$R = \max \|A - B_1\|_2 \quad \text{over all possible realizations of } B_1 \quad . \qquad (3)$$

Equations (2) and (3) mark the starting point of our work. That is, our goal will be to find probability distributions over the elements of A for which Theorem 1.1 yields the strongest possible bound on $\|A - B\|_2$. A key conceptual contribution of our work is the discovery that these distributions *depend* on the sample budget s in a non-linear, but still highly-intuitive way, something also borne out in experiments. The fact that minimizing the deviation metric of Theorem 1.1, i.e., $\sigma^2 + R\epsilon/3$, suffices to bring out this non-linearity is testament to the theorem's sharpness.

Theorem 1.1 is stated as a bound on the probability that the norm of the error matrix is greater than some target error ε given the number of samples s. Nevertheless, in practice the target error ε is not known in advance, but rather is a minimization target given the matrix A, the number of samples s and the target confidence δ. Specifically, for any given distribution P on the elements of A, define

$$\varepsilon_1(P) = \inf\left\{\varepsilon : (m+n)\exp\left(-\frac{s\varepsilon^2}{\sigma(P)^2 + R(P)\varepsilon/3}\right) \leqslant \delta\right\} \quad . \qquad (4)$$

Our goal in the rest of the paper is to seek the distribution P^* minimizing ε_1. Our result is an easily computable distribution P which comes within a factor of 3 of $\varepsilon_1(P^*)$ and, as a result, within a factor of 9 in terms of sample complexity (in practice we expect this to be even smaller, as the factor of 3 comes from consolidating bounds for a number of different worst-case matrices). To put this in perspective note that the definition of P^* does not place *any* restriction either on the access model for A while computing P^*, or on the amount of time needed to compute P^*. In other words, we are competing against an oracle which in order to determine P^* has *all* of A in its purview at once and can spend an unbounded amount of computation to determine P^*.

The only global information we will require are the *ratios* of the L1 norms of the rows of the matrix, i.e., of $\|A_{(i)}\|_1 = \sum_j |A_{ij}|$. Clearly, the row-L1 norms (and therefore their ratios) could be trivially computed in a single pass over the matrix, yielding a 2-pass algorithm. In our setting, though, since different rows correspond to different attributes it is very reasonable to expect that good estimates of these *ratios* are available a priori, i.e., before even touching the matrix A, since they simply reflect the average absolute values of the m attributes. Moreover, as will become clear, these ratios do not need to be known exactly to apply the algorithm and even rough estimates of them will give highly competitive results. Finally, we note that these ratios can be estimated with extremely high accuracy from a relatively small number of columns, using standard concentration arguments. Due to space limitations we do not present these argument here.

2 Data Matrices and Statement of Results

Throughout $A_{(i)}$ and $A^{(j)}$ will denote the i-th row and j-th column of A, respectively. Also, we use the notation $\|A\|_1 = \sum_{i,j} |A_{i,j}|$ and $\|A\|_F^2 = \sum_{i,j} A_{i,j}^2$. Before we formally state our result we introduce a definition that expresses the class of matrices for which our results are most relevant. Out of three conditions in the definition the essential one, reflecting the data matrix setting, is

Condition 1. The other two are merely technical and hold in all non-trivial cases where Condition 1 applies.

Definition 2.1. *An $m \times n$ matrix A is a* Data *matrix if:*

1. $\min_i \|A_{(i)}\|_1 \geqslant \max_j \|A^{(j)}\|_1$.
2. $m \geqslant 30$.
3. $\|A\|_1^2 / \|A\|_2^2 \geqslant 30m$.

Regarding Condition 1 recall that we think of A as being generated by a measurement process of a fixed number of attributes (rows), each column corresponding to an observation. As a result, columns have bounded L1 norm, i.e., $\|A^{(j)}\|_1 \leqslant$ constant. While this constant may depend on the type of object and its dimensionality, it is independent of the number of objects. On the other hand, $\|A_{(i)}\|_1$ grows linearly with the number of columns. As a result, we can expect Definition 2.1 to hold for all large enough data sets. Condition 2 is trivial. Regarding Condition 3, since $\|A\|_2^2 \leqslant \|A\|_F^2$, we note that when all entries in A have the same magnitude, the ratio $\|A\|_1^2/\|A\|_2^2$ is greater than or equal to the number of non-zero entries in A. While the ratio can be smaller than the number of non-zero entries when the entry magnitudes have large variance, in all non-trivial cases it grows as $\Omega(n)$ and, thus, Condition 3 follows from $n \gg m$.

One last point is that to apply Theorem 1.1, the entries of A must be sampled *with* replacement. In the streaming model, this means that when an entry is presented we need to decide not only whether to keep it or not, but its multiplicity. A simple way to achieve this using $O(s)$ active memory was presented in [2]: at any given moment maintain s independent, not necessarily distinct, samples of A in main memory, updating them via reservoir-sampling on the $\sum P_{ij}$ for the elements seen so far. The final output are the s samples present in memory upon termination of the stream. In Section D we discuss how to implement sampling with replacement far more efficiently, using $O(\log(s))$ active memory and dramatically less computation and randomness. To simplify the exposition of our algorithm, though, we invoke the simple sampling of [2] in Step 5 below. Also for simplicity of exposition, we assume that we are given the actual$\|A_{(i)}\|_1$ for all $1 \leqslant i \leqslant m$. As we discuss immediately below the algorithm, all that is actually needed are their ratios.

Algorithm 1 Construct a proxy to a data matrix A

```
1: Input: The dimensions of an $m \times n$ matrix $A$ and $\|A_{(i)}\|_1$ for all $1 \leqslant i \leqslant m$. A desired probability of success $\delta > 0$ and a sample budget $s$.
2: Let
       $w := \log((m+n)/\delta)/s \qquad \alpha := \sqrt{w} \qquad \beta := w/3 \ . \qquad (5)$
3: For $\zeta \in \mathbb{R}^+$, let $T(\zeta) = \sum_{i=1}^{m} \rho_i(\zeta)$, where
       $\rho_i(\zeta) = \left( \frac{\alpha \|A_{(i)}\|_1}{2\zeta} + \sqrt{\left( \frac{\alpha \|A_{(i)}\|_1}{2\zeta} \right)^2 + \frac{\beta \|A_{(i)}\|_1}{\zeta}} \right)^2 \ . \qquad (6)$
4: Find $\zeta_0$ such that $T(\zeta_0) = 1$ and let
       $P_{ij} = \frac{\rho_i(\zeta_0)}{\|A_{(i)}\|_1} \cdot |A_{ij}| \ := \theta_i \cdot |A_{ij}| \ . \qquad (7)$
5: Sample $s$ non-zero elements of $A$ with replacement, each $A_{ij}$ having probability $\theta_i |A_{ij}|$.
6: For each sample $\langle i, j, A_{ij} \rangle_\ell$, let $B^{(\ell)}$ be the $m \times n$ matrix with non-zero entry $B^{(\ell)}_{ij} = A_{ij}/P_{ij}$.
7: Output: $(B^{(1)} + \cdots + B^{(s)})/s$.
```

Remark: Steps 2–4 of the algorithm simply compute row-weights θ_i. For this, first a probability distribution on the rows, $\rho_i(\zeta_0)$, is computed in Step 3. Finding ζ_0 can be done very efficiently by binary search because the function T is strictly monotone in ζ while the constraint $\sum_i \rho_i(\zeta) = 1$, implies that to find ζ_0 it suffices to know the ratios of the $\|A_{(i)}\|_1$. Similarly for computing θ_i. Conceptually, we see that the probability assigned to each element A_{ij} in Step 4 is simply the probability ρ_i of its row times its own probability, $|A_{ij}|/\|A_{(i)}\|$ under L1-sampling of elements from its row. Finally, we note that in practice one does not, of course, actually form the matrices B_i in Steps 6, 7 but outputs B by aggregating the samples into a matrix.

Theorem 2.2. *If A is a* Data *matrix and P is the probability distribution defined in* (7)*, then $\varepsilon_1(P) \leqslant 3\,\varepsilon_1(P^*)$, where P^* is the minimizer of ε_1.*

Remark: A key insight of our work is that the distribution we propose in (7) is a combination of two L1-based distributions. The first is the plain L1-distribution where $P_{i,j} \propto |A_{i,j}|$. The second is a distribution we call "Row-L1" where $P_{i,j} \propto |A_{i,j}| \cdot \|A_{(i)}\|_1$. By considering (6), it follows that when s is small, $\alpha \approx \beta$ and ρ_i is nearly linear in $\|A_{(i)}\|$, as in L1-sampling. However, as s grows, α becomes dominant compared to β and ρ_i tends towards $\|A_{(i)}\|_1^2$ as in Row-L1 sampling. This insight is also borne out in the experiments where the sampling based on (7) is consistently best across all cases, highlighting that the need to adapt the sampling distribution to the sample budget is a genuine phenomenon and not simply an artifact of having derived the distribution by optimizing a tail bound.

3 Background and Related Work

By using the spectral norm to measure error we get a natural and sophisticated target: by minimizing $\|A - B\|_2$ we seek to make $E = A - B$ a near-rotation, having only small variations in the amount by which it stretches different vectors. This idea that the error matrix E should be isotropic motivated the first work on element-wise sampling of matrices by Achlioptas and McSherry [3]. Concretely, to minimize $\|E\|_2$ it is natural both to have E be zero-mean, i.e., for B to be an unbiased estimator of A, and to sample the entries of A (and, thus, of E) independently, as independence tends to endow isotropy. Thus, in the work of [3], E is a matrix of i.i.d. zero-mean random variables and the study of the spectral characteristics of such matrices goes back all the way to Wigner's famous semi-circle law [4]. To bound $\|E\|_2$ in [3] a bound due to Alon Krivelevich and Vu [5] was used, a refinement of a bound by Juhász [6] and Füredi and Komlós [7]. The most salient feature of that bound is that it depends on the *maximum* entry-wise variance σ^2 of $A - B$, and therefore the distribution optimizing the bound is the one in which the variance of all entries in E is the same. In turn, this means keeping each entry of A independently with probability $P_{i,j} \propto A_{i,j}^2$ (up to a small wrinkle discussed below).

Several papers since have analyzed L2 sampling and variants [8, 9, 10, 11, 3]. An inherent difficulty of that strategy is the need for a special handling of small entries. This is because when each item $A_{i,j}$ is kept with probability $p_{i,j} \propto A_{i,j}^2$, the resulting entry B_{ij} in the sample matrix has magnitude $|A_{i,j}/p_{i,j}| \propto 1/|A_{i,j}|$. Thus, if an extremely small element $A_{i,j}$ is accidentally picked, the largest entry of the sample matrix "blows up". In [3] this was addressed by sampling "small" entries with probability proportional to $|A_{i,j}|$ rather than $A_{i,j}^2$. In [11], small entries are not handled separately and the bound derived depends on the ratio between the largest and the smallest non-zero magnitude.

Random matrix "technology" has witnessed dramatic progress in the last few years and [12, 13, 14, 15] provide a good overview of the results. This progress motivated Drineas and Zouzias in [10] to revisit L2 sampling but now using concentration results for *sums* of random matrices [15], as we do here. (Note that this is somewhat different from the original setting of [3] since now E is not one random matrix with independent entries, but a sum of many independent matrices since the entries are chosen with replacement.) Their work improved upon all previous L2-based sampling results and also upon the L1-sampling result of Arora, Hazan and Kale [16], discussed below, while admitting a remarkably compact proof. The issue of small entries was handled in [10] by simply deterministic discarding all "sufficiently small" entries, a strategy that gives the strongest mathematical guarantee (but see the discussion regarding deterministic truncation in the experimental section).

A completely different tack at the problem, avoiding random matrix theory, was taken by Arora et al. [16]. Their approximation keeps the largest entries in A deterministically ($A_{i,j} \geqslant \varepsilon/\sqrt{n}$ where the threshold ε needs be known a priori) and randomly rounds the remaining smaller entries to $\mathrm{sign}(A_{i,j})\varepsilon/\sqrt{n}$ or 0. They then exploit the simple fact $\|A - B\| = \sup_{\|x\|=1, \|y\|=1} x^T(A - B)y$ by noting that as a scalar quantity its concentration around its expectation can be established by standard Bernstein-Bennet type inequalities. A union bound then allows them to prove that with high probability, $x^T(A - B)y \leqslant \varepsilon$ for *every* x and y. The result of [16] admits a relatively simple proof. However, it also requires a truncation that depends on the desired approximation ε. Rather interestingly, this time the truncation amounts to keeping every entry larger than some threshold.

4 Mathematical comparison to other works

To have a definite comparison of our mathematical result with those of previous works we first state the bound implied by Theorem 2.2 on the minimal number of samples s needed by our algorithm to achieve an approximation B to the matrix A such that $\|A - B\| \leqslant \varepsilon\|A\|$ with constant probability. The proof of Theorem 4.1 is given in Appendix C.

Theorem 4.1. *Let A be any matrix meeting the conditions of Definition 2.1 and let B be the matrix returned by Algorithm 2 for $\delta = 1/10$ for a given sample budget s. For any $\varepsilon > 0$, if $s \geqslant s_0$, where s_0 is as in Table 1, $\|A - B\| \leqslant \varepsilon\|A\|$ with probability at least $9/10$.*

To compare the s_0 of Theorem 4.1 to earlier works we next introduce a few matrix metrics.

Stable rank: Denoted as sr and defined as $\|A\|_F^2/\|A\|^2$, this is a smooth analog for the algebraic rank, always bounded by it from above, and resilient to small perturbations of the matrix. For data matrices we expect it to be small, capturing the "inherent dimensionality" of the data.

Numeric density: Denoted as nd and defined as $\|A\|_1^2/\|A\|_F^2$, this is a smooth analog of the number of non-zero entries $\mathrm{nnz}(A)$. For 0-1 matrices it equals $\mathrm{nnz}(A)$, but when there is variance in the cardinality of the entries it is smaller.

Numeric row density: Denoted as nrd and defined as $\sum_i \|A_{(i)}\|_1^2/\|A\|_F^2 \leqslant n$. In practice, it is often close to the average numeric density of a single row, a quantity typically much smaller than n.

The third column of Table 1 below shows the corresponding values of s in previous works for constant success probability, in terms of the matrix metrics defined above. The fourth column presents the ratio of the samples needed by previous results divided by the samples needed by our method. (To simplify the expressions, we present the ratio between our bound and [16] only when the result of [16] gives superior bounds to [10], i.e., we always compare our bound to the stronger of the two bounds implied by these works). Holding ε and the stable rank constant we readily see that our method requires roughly $1/\sqrt{n}$ the samples needed by [16]. In the comparison with [10] we see that the key parameter is the ratio nrd/n, a quantity typically much smaller than 1 for data matrices. (As a point of reference for the assumptions, in the experimental Section 6 we provide the values of all relevant matrix metrics for all the real data matrices we worked with, wherein the ratio nrd/n is typically around 10^{-2}.)

By the discussion above, one would expect that L2-sampling should fare better than L1-sampling in experiments. As we will see, quite the opposite is true. A potential explanation for this phenomenon is the relative looseness of the bound of [16] for the performance of L1 sampling.

Citation	Method	Samples needed	Sample Ratio
[3]	L1, L2	$\mathrm{sr}\cdot(n/\varepsilon^2) + n \cdot \mathrm{polylog} n$	
[10]	L2	$\mathrm{sr}\cdot(n/\varepsilon^2)\log n$	$\dfrac{\mathrm{nrd}}{n} + \dfrac{\varepsilon}{n}\left(\dfrac{\mathrm{nd}}{\mathrm{sr}\cdot\log n}\right)^{1/2}$
[16]	L1	$(\mathrm{nd}\cdot n/\varepsilon^2)^{1/2}$	$\left(\dfrac{\mathrm{sr}\cdot\log n}{n}\right)^{1/2}$
This paper	Bernstein	$\mathrm{nrd}\cdot\mathrm{sr}/\varepsilon^2\cdot\log n + (\mathrm{sr}\cdot\mathrm{nd}/\varepsilon^2\cdot\log n)^{1/2}$	

Table 1: 'Sample Ratio' is the ratio of samples needed by the corresponding work and this paper.

5 Proof outline

In what follows we repeatedly replace our objective function with simpler and simpler functions. Each replacement will incur some (small) loss in accuracy but will bring us closer to a closed form solution. Recalling the definitions of α, β from (5) and rewriting the requirement in (4) as a quadratic form in ε gives $\varepsilon^2 - \varepsilon\beta R - (\alpha\sigma)^2 > 0$. Our first step is to observe that for any $c, d > 0$, the equation

$\varepsilon^2 - \varepsilon \cdot c - d = 0$ has one negative and one positive solution and that the latter is at least $(c + \sqrt{d})/\sqrt{2}$ and at most $c + \sqrt{d}$. Therefore, if we define[1] $\varepsilon_2 := \alpha\sigma + \beta R$ we see that $1/\sqrt{2} \leqslant \varepsilon_1/\varepsilon_2 \leqslant 1$.

Our next simplification encompasses Conditions 2, 3 of Definition 2.1. Let $\varepsilon_3 := \alpha\tilde{\sigma} + \beta\tilde{R}$ where

$$\tilde{\sigma}^2 := \max\left\{\max_i \sum_j A_{ij}^2/P_{ij}\ ,\ \max_j \sum_i A_{ij}^2/P_{ij}\right\} \quad \tilde{R} := \max_{ij} |A_{ij}|/P_{ij}\ .$$

We will prove the following, allowing us to optimize P with respect to ε_3 instead of ε_2.

Lemma 5.1. *For every matrix A satisfying Conditions 2 and 3 of Definition 2.1, for every probability distribution on the elements of A,* $|\varepsilon_2/\varepsilon_3 - 1| \leqslant 1/30$.

In minimizing ε_3 we see that there is freedom to use different rows to optimize $\tilde{\sigma}$ and $\tilde{R}$. At a cost of a factor of 2, we will couple the two minimizations by minimizing $\varepsilon_4 = \max\{\varepsilon_5, \varepsilon_6\}$ where

$$\varepsilon_5 := \max_i \left[\alpha \sqrt{\sum_j \frac{A_{ij}^2}{P_{ij}}} + \beta \max_j \frac{|A_{ij}|}{P_{ij}}\right], \qquad \varepsilon_6 := \max_j \left[\alpha \sqrt{\sum_i \frac{A_{ij}^2}{P_{ij}}} + \beta \max_i \frac{|A_{ij}|}{P_{ij}}\right] \ . \quad (8)$$

Note that the maximization of $\tilde{R}$ in ε_5 (and ε_6) is coupled with that of the $\tilde{\sigma}$-related term by constraining the optimization to consider only one row (column) at a time. Clearly, $1 \leqslant \varepsilon_3/\varepsilon_4 \leqslant 2$.

Next we focus on ε_5, the first term in the maximization of ε_4. The following lemma establishes that for all matrices satisfying Condition 1 of Lemma 2.1, minimizing ε_5 minimizes $\varepsilon_4 = \max\{\varepsilon_5, \varepsilon_6\}$.

Lemma 5.2. *For every matrix satisfying Condition 1 of Definition 2.1,* $\operatorname{argmin}_P \varepsilon_5 \subseteq \operatorname{argmin}_P \varepsilon_4$.

Finally, we will derive in closed form the probability distribution P minimizing ε_5.

Lemma 5.3. *The function ε_5 is minimized by $P_{ij} = \frac{\rho_i(\zeta_0)}{\|A_{(i)}\|_1} \cdot |A_{ij}|$, where ρ_i is as in* (6) *and $\zeta_0 > 0$ is the unique $\zeta > 0$ such that $\sum \rho_i(\zeta_0) = 1$.*

To prove Theorem 2.2 we see that Lemmas 5.2 and 5.3 combined imply that there is an efficient algorithm for minimizing ε_4 for every matrix A satisfying Condition 1 of Definition 2.1. If A also satisfies Conditions 2 and 3 of Definition 2.1, then the facts $1/\sqrt{2} \leqslant \varepsilon_1/\varepsilon_2 \leqslant 1$, $|\varepsilon_2/\varepsilon_3 - 1| \leqslant 1/30$, and $1 \leqslant \varepsilon_3/\varepsilon_4 \leqslant 2$, imply $\frac{1}{2\sqrt{2}(29/30)} \leqslant \varepsilon_1/\varepsilon_4 \leqslant 31/30$. In general, if $c \leqslant \varepsilon_4/\varepsilon_1 \leqslant C$ we can conclude that $\varepsilon_1(\arg\min(\varepsilon_4)) \leqslant (C/c)\min(\varepsilon_1)$. Thus, the fact $62\sqrt{2}/29 < 3$ yields our claim.

6 Experiments

We experimented with 4 matrices with different characteristics, the contents and properties of which are as follows. The sizes and many other statistics of the matrices are summarized in Table 2.

Enron: Subject lines of emails in the Enron email corpus [17]. Columns correspond to subject lines, rows to words, and entries to tf-idf values. This matrix is extremely sparse to begin with.
Wikipedia: Document-term matrix of a fragment of the English Wikipedia. Entries are tf-idf values.
Images: A collection of images of buildings from Oxford [18]. Each column represents the wavelet transform a single 128×128 pixel grayscale image.
Synthetic: This synthetic matrix simulates a collaborative filtering matrix. Each row corresponds to an item and each column to a user. Each user and each item was first assigned a random latent vector (i.i.d. Gaussian). Each value in the matrix is the dot product of the corresponding latent vectors plus additional Gaussian noise. We simulated the fact that some items are more popular than others by retaining each entry of each item i with probability $1 - i/m$ where $i = 0, \ldots, m - 1$.

6.1 Sampling techniques and quality measure

The experiments report the accuracy of sampling according to four distributions. In Figure 6.1, **"Bernstein"** denotes the distribution of this paper, i.e., equation (7). The **Row-L1** distribution is a

[1] Here and in the following, to lighten notation, we will omit all arguments, i.e., $P, \sigma(P), R(P)$, from the objective functions ε_i we seeks to optimize, as they are readily understood from context.

simplified version of the Bernstein distribution, where $P_{ij} \propto |A_{ij}| \cdot |A_{(i)}|_1$ Finally, **L1** and **L2** refer to $P_{ij} \propto |A_{ij}|$ and $P_{ij} \propto |A_{ij}|^2$, respectively, as defined earlier in the paper.

Although to derive our sampling probability distributions we targeted minimizing $\|A - B\|_2$, in experiments it turns out to be more fruitful to consider a more sensitive measure of quality of approximation. The reason is that, due to scaling, for a number of values of s one has $\|A - B\|_2 > \|A\|_2$ which would suggest that the all-0s matrix is a better sketch for A than the sampled matrix, something which, as we will see, is far from the case. As a trivial example, consider the possibility $B \approx 10A$. Clearly, B is very informative of A although $\|A - B\| \geqslant 9\|A\|$. To avoid this pitfall, we computed instead $\|P_k^B A\|_F / \|A_k\|_F$, where P_k^B is the projection on the top k left singular vectors of B, while $A_k = P_k^A A$ is the optimal rank k approximation of A. Intuitively, this measures how well the k left singular space of B capture A, relative to A's own, and thus optimal, top-k left singular vectors. We also computed $\|AQ_k^B\|_F / \|A_k\|_F$ where Q_k^B is the projection on the top k right singular vectors of A. Note that, for a given k, approximating the row-space is harder than approximating the column-space since it is of dimension n which is significantly larger than m, a fact also borne out in the experiments. In the experiments we made sure to choose a sufficiently wide range of sample sizes so that at least the best method for each matrix goes from poor to near-perfect both in approximating the row and the column space.

In all cases we report $k = 20$ which is close to the upper end of what could be efficiently computed on a single machine for matrices of this size. (The results for all smaller values of k are qualitatively indistinguishable to $k = 20$.) Finally, we note that in carrying out L1 and L2 sampling we did not implement truncation. This is because in both cases, for most values of the sample budget s considered, i.e., values for which our method manages to deliver very good approximations, the truncations required by the accompanying theoretical results would result in simply returning either the entire matrix, in the case of L1 sampling, or the empty matrix, in the case of L2 sampling.

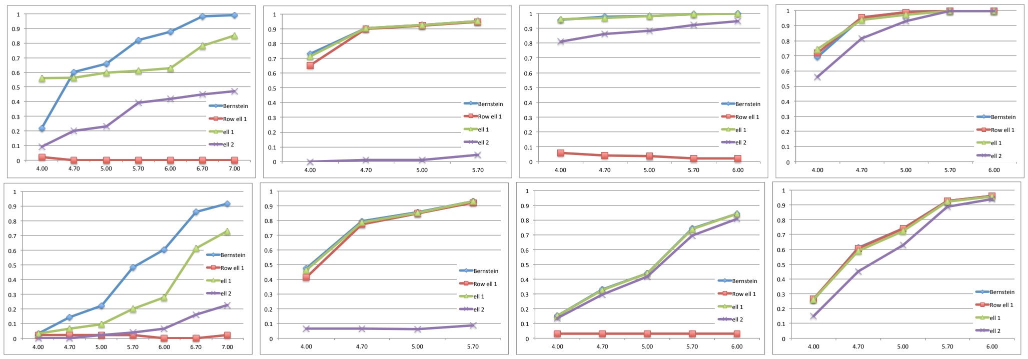

Figure 1: Each vertical pair of plots corresponds to one matrix. Left to right: Wikipedia, Images, Enron, and Synthetic. Each top plot shows the quality of approximation ratio, $\|P_B^k A\|_F / \|A_k\|$ (bottom plots show $\|AQ_B^k\|_F / \|A_k\|$). The number of samples s is on the x-axis in log scale $x = \log_{10}(s)$.

Measure	Synthetic	Enron	Images	Wikipedia
m	1.00e+02	1.26e+04	5.14e+03	4.39e+05
n	1.00e+04	1.76e+05	4.92e+05	3.43e+06
$\mathrm{nnz}(A)$	5.00e+05	7.24e+05	2.52e+08	5.34e+08
$\|A\|_1$	1.76e+07	4.01e+09	6.54e+09	5.30e+09
$\|A\|_F$	3.17e+04	5.76e+06	1.97e+06	7.47e+05
$\|A\|_2$	8.653+03	1.02e+06	1.75e+06	1.63e+05
sr	1.34e+01	3.21e+01	1.27e+00	2.10e+01
nd	3.09e+05	4.85e+05	1.10e+07	5.04e+07
nrd	3.18e+03	1.53e+03	2.32e+03	1.91e+04

Table 2: Values of the metrics defined in Section 4 for the four matrices in our experiments.

6.2 Insights

The experiments demonstrate two main insights. First and most important, Bernstein-sampling is never worse than any of the other techniques and is often strictly better. A dramatic example of this is the Wikipedia matrix for which it is far superior to all other methods.

The second insight is that L1-sampling, i.e., simply taking $P_{ij} = |A_{ij}|/\|A\|_1$, performs rather well in many cases. Hence, if it is impossible to perform more than one pass over the matrix and one can not obtain any estimate of the ratios of the L1-weights of the rows, L1-sampling seems to be a highly viable option.

References

[1] Joel A Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

[2] Petros Drineas, Ravi Kannan, and Michael W. Mahoney. Fast monte carlo algorithms for matrices; approximating matrix multiplication. *SIAM J. Comput.*, 36(1):132–157, July 2006.

[3] Dimitris Achlioptas and Frank Mcsherry. Fast computation of low-rank matrix approximations. *J. ACM*, 54(2), 2007.

[4] Eugene P. Wigner. On the distribution of the roots of certain symmetric matrices. *Annals of Mathematics*, 67(2):pp. 325–327, 1958.

[5] Noga Alon, Michael Krivelevich, and VanH. Vu. On the concentration of eigenvalues of random symmetric matrices. *Israel Journal of Mathematics*, 131:259–267, 2002.

[6] F. Juhász. On the spectrum of a random graph. In *Algebraic methods in graph theory, Vol. I, II (Szeged, 1978)*, volume 25 of *Colloq. Math. Soc. János Bolyai*, pages 313–316. North-Holland, Amsterdam, 1981.

[7] Z. Füredi and J. Komlós. The eigenvalues of random symmetric matrices. *Combinatorica*, 1(3):233–241, 1981.

[8] NH Nguyen, Petros Drineas, and TD Tran. Matrix sparsification via the khintchine inequality, 2009.

[9] Nam H Nguyen, Petros Drineas, and Trac D Tran. Tensor sparsification via a bound on the spectral norm of random tensors. *arXiv preprint arXiv:1005.4732*, 2010.

[10] Petros Drineas and Anastasios Zouzias. A note on element-wise matrix sparsification via a matrix-valued bernstein inequality. *Inf. Process. Lett.*, 111(8):385–389, 2011.

[11] Alex Gittens and Joel A Tropp. Error bounds for random matrix approximation schemes. *arXiv preprint arXiv:0911.4108*, 2009.

[12] Rudolf Ahlswede and Andreas Winter. Strong converse for identification via quantum channels. *IEEE Transactions on Information Theory*, 48(3):569–579, 2002.

[13] Mark Rudelson and Roman Vershynin. Sampling from large matrices: An approach through geometric functional analysis. *J. ACM*, 54(4), July 2007.

[14] Joel A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389–434, 2012.

[15] Benjamin Recht. A simpler approach to matrix completion. *J. Mach. Learn. Res.*, 12:3413–3430, December 2011.

[16] Sanjeev Arora, Elad Hazan, and Satyen Kale. A fast random sampling algorithm for sparsifying matrices. In *Proceedings of the 9th international conference on Approximation Algorithms for Combinatorial Optimization Problems, and 10th international conference on Randomization and Computation*, APPROX'06/RANDOM'06, pages 272–279, Berlin, Heidelberg, 2006. Springer-Verlag.

[17] Will Styler. The enronsent corpus. In *Technical Report 01-2011, University of Colorado at Boulder Institute of Cognitive Science, Boulder, CO.*, 2011.

[18] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

Appendix: Near-optimal Distributions for Data Matrix Sampling

A Optimizations on the L1 ball

Lemma A.1. *For any $x, p \in \mathbb{R}^n$, if $p_i \geqslant 0$ and $\|p\|_1 = 1$, then $\max_k |x_k|/p_k \geqslant \|x\|_1$ and $\sum_k x_k^2/p_k \geqslant \|x\|_1^2$, with equality holding in both cases if and only if $p_k = |x_k|/\|x\|_1$.*

Proof. To prove $\max_k |x_k|/p_k \geqslant \|x\|_1$ we note that if $|x_i|/p_i \neq |x_j|/p_j$, then changing p_i, p_j to p'_i, p'_j such that $p'_i + p'_j = p_i + p_j$ and $|x_i|/p'_i = |x_j|/p'_j$ can only reduce the maximum. In order for all $|x_k|/p_k$ to be equal it must be that $p_k = |x_k|/\|x\|_1$ for all j, in which case $\max_k |x_k|/p_k = \|x\|_1$.

The second claim follows from applying Jensen's inequality to the convex function $x \mapsto x^2$. □

To prove Lemma 5.1 we first establish the following.

Lemma A.2. *For any matrix A and any probability distribution P on the elements of A, we have $|\sigma^2/\tilde{\sigma}^2 - 1| \leqslant \frac{\|A\|_2^2}{\sum_i \|A_{(i)}\|_1^2}$ and $|R/\tilde{R} - 1| \leqslant \frac{\|A\|_2}{\|A\|_1}$.*

Proof. Recall that B_1 contains one non-zero element A_{ij}/P_{ij}, while all its other entries are 0. Therefore, $\mathbb{E}[B_1 B_1^T]$ and $\mathbb{E}[B_1^T B_1]$ are both diagonal matrices where

$$\mathbb{E}[(B_1 B_1^T)_{i,i}] = \sum_j A_{i,j}^2/P_{i,j} \qquad \text{and} \qquad \mathbb{E}[(B_1^T B_1)_{j,j}] = \sum_i A_{i,j}^2/P_{i,j} \ .$$

Since the operator norm of a diagonal matrix equals its largest entry we see that

$$\tilde{\sigma}^2 := \max\left\{ \max_i \sum_j A_{ij}^2/P_{ij} \ , \ \max_j \sum_i A_{ij}^2/P_{ij} \right\} = \max\{\|\mathbb{E}[B_1 B_1^T]\|, \|\mathbb{E}[B_1^T B_1]\|\} \ .$$

We will also need to bound $\tilde{\sigma}^2$ from below. Trivially, $\tilde{\sigma}^2 \geqslant \|\mathbb{E}[B_1 B_1^T]\| = \max_i \sum_j A_{ij}^2/P_{ij}$. Let $\rho_i = \sum_j P_{ij}$. Applying Lemma A.1 to the i-th row of A by taking $x_k = A_{ik}$ and $p_k = P_{ik}/\rho_i$ yields the second inequality below. Applying it with $x_k = \|A_{(k)}\|_1$ and $p_k = \rho_k$ yields the third

$$\tilde{\sigma}^2 \geqslant \max_i \sum_j \frac{A_{ij}^2}{P_{ij}} = \max_i \rho_i^{-1} \sum_k \frac{A_{ik}^2}{p_k} \geqslant \max_i \rho_i^{-1} \|A_{(i)}\|_1^2 \geqslant \sum_i \|A_{(i)}\|_1^2 \ . \tag{9}$$

On the other hand, $\sigma^2 = \max\{\|\mathbb{E}[Z_1 Z_1^T]\|, \|\mathbb{E}[Z_1^T Z_1]\|\}$, where $Z_1 = B_1 - A$. Since $\mathbb{E}[B_1] = A$,

$$\|\mathbb{E}[Z_1 Z_1^T]\| = \|\mathbb{E}[B_1 B_1^T - A B_1^T - B_1 A^T + A A^T]\| = \|\mathbb{E}[B_1 B_1^T] - A A^T\|$$

and, analogously, $\|\mathbb{E}[Z_1^T Z_1]\| = \|\mathbb{E}[B_1^T B_1] - A^T A\|$. Therefore, by the triangle inequality, $|\sigma^2 - \tilde{\sigma}^2| \leqslant \|A\|^2$ and the claim now follows from (9).

Recall that B_1 contains one non-zero entry A_{ij}/P_{ij} and that R is the maximum of $\|B_1 - A\|$ over all possible realizations of P, i.e., choices of i, j. Thus by the triangle inequality,

$$R = \max \|B_1 - A\| \leqslant \max \|B_1\| + \|A\| \quad \text{and} \quad R \geqslant \max \|B_1\| - \|A\| \ .$$

Since B_1 has one non-zero entry, we see that $\max \|B_1\|_2 = \max_{ij} |A_{ij}|/P_{ij} = \tilde{R}$ and, thus, $|R/\tilde{R} - 1| \leqslant \|A\|_2/\tilde{R}$. Applying Lemma A.1 to $A \in \mathbb{R}^{m \times n}$ with distribution P yields $\tilde{R} \geqslant \|A\|_1$. □

Proof of Lemma 5.1. It suffices to prove that both $|\sigma^2/\tilde{\sigma}^2 - 1|$ and $|R/\tilde{R} - 1|$ are bounded by $1/30$.

Lemma A.2 yields the first inequality below and Condition 3 of Definition 2.1 the second. The third inequality holds for every matrix A, with equality occurring when all rows have the same L1 norm.

$$|\sigma^2/\tilde{\sigma}^2 - 1| \leqslant \frac{\|A\|_2^2}{\sum_i \|A_{(i)}\|_1^2} \leqslant \frac{\|A\|_1^2}{30m \sum_i \|A_{(i)}\|_1^2} \leqslant \frac{1}{30} \ .$$

Lemma A.2 yields the first inequality below. The second inequality follows from rearranging the factors in the second inequality above. Condition 2 of Definition 2.1, i.e., $m \geqslant 30$, implies the third.

$$|R/\tilde{R} - 1| \leqslant \frac{\|A\|_2}{\|A\|_1} \leqslant \frac{1}{\sqrt{30m}} \leqslant \frac{1}{30} \ .$$

□

B Global minimization over the distribution

To find the probability distribution P that minimizes ε_5 we start by writing $P = \rho_i q_{ij}$, without loss of generality. That is, we decompose P to a distribution $\rho_i \geqslant 0$ over the rows of the matrix, i.e., $\sum_i \rho_i = 1$, and a distribution $q_{ij} \geqslant 0$ within each row i, i.e., $\sum_j q_{ij} = 1$, for all i. We first prove that (surprisingly) the optimal q has a closed form solution while the optimal ρ is efficiently computable.

For any ρ, writing ε_5 in terms of ρ_i, q_{ij} we see that ε_5 is the maximum, over rows $1 \leqslant i \leqslant m$, of

$$\frac{\alpha}{\sqrt{\rho_i}} \sqrt{\sum_j \frac{A_{ij}^2}{q_{ij}}} + \frac{\beta}{\rho_i} \max_j \frac{|A_{ij}|}{q_{ij}} \ . \tag{10}$$

Observe that since ρ is fixed, the only variables in the above expression for each row i are the q_{ij}. Lemma A.1 implies that setting $q_{ij} = |A_{ij}|/\|A_{(i)}\|_1$ simultaneously minimizes both terms in (10). This means that for *every* fixed probability distribution ρ, the minimizer of ε_5 satisfies $q_{ij} = \frac{|A_{ij}|}{\|A_{(i)}\|_1}$. Thus, we are left to determine

$$\Phi(\rho) = \max_i \left[\frac{\alpha \|A_{(i)}\|_1}{\sqrt{\rho_i}} + \frac{\beta \|A_{(i)}\|_1}{\rho_i} \right] \ .$$

Unlike the intrarow optimization, the two summands in Φ achieve their respective minima at different distributions ρ. To get some insight into the tradeoff, let us first consider the two extreme cases. When $\beta = 0$, minimizing the maximum over i requires equating all $\|A_{(i)}\|_1/\sqrt{\rho_i}$, i.e., $\rho_i \propto \|A_{(i)}\|_1^2$, leading to the distribution we call "row-L_1", i.e., $P_{ij} \propto |A_{ij}| \cdot \|A_{(i)}\|_1$. When $\alpha = 0$, equating the $\|A_{(i)}\|_1/\rho_i$ requires $\rho_i \propto \|A_{(i)}\|_1$, leading to the "plain-$L_1$" distribution $P_{ij} \propto |A_{ij}|$.

Nevertheless, since we wish to minimize the maximum over several functions, we can seek P under which all functions are equal, i.e., such that there exists $\zeta > 0$ such that for all i,

$$\frac{\alpha \|A_{(i)}\|_1}{\sqrt{\rho_i}} + \frac{\beta \|A_{(i)}\|_1}{\rho_i} = \zeta > 0 \ .$$

Solving the resulting quadratic equation and selecting for the positive root yields equation (6), i.e.,

$$\rho_i(\zeta) = \left(\frac{\alpha \|A_{(i)}\|_1}{2\zeta} + \sqrt{\left(\frac{\alpha \|A_{(i)}\|_1}{2\zeta} \right)^2 + \frac{\beta \|A_{(i)}\|_1}{\zeta}} \right)^2 \ . \tag{11}$$

Since the quantities under the square root in (6) are all positive we see that it is always possible to find $\zeta > 0$ such that all equalities hold, and thus (6) does minimize ε_5 for every matrix A. Moreover, since the right hand side of (6) is strictly decreasing in ζ, binary search finds the unique value of ζ such that $\sum \rho_i = 1$.

Finally, recall that our overall goal is to determine the minimizer of $\varepsilon_4 = \max\{\varepsilon_5, \varepsilon_6\}$. We already have determined the minimizer of ε_5. We will now show that for matrices satisfying Condition 1 of Lemma 2.1 the minimizer of ε_5 is also the minimizer of ε_4. We first prove that

Lemma B.1. *For any two functions f, g, if $x_0 = \arg\min_x f(x)$ and $g(x_0) \leqslant f(x_0)$, then* $\min_x \max\{f(x), g(x)\} = f(x_0)$.

Proof.

$$\min_x \max\{f(x), g(x)\} \geqslant \min_x f(x) = f(x_0) = \max\{f(x_0), g(x_0)\} \geqslant \min_x \max\{f(x), g(x)\}$$

□

Thus, it suffices to evaluate ε_6 at the distribution P minimizing ε_5 and check that $\varepsilon_6(P) \leqslant \varepsilon_5(P)$.

We know that P is of the form $P_{i,j} = \rho_i |A_{i,j}|/\|A_{(i)}\|_1$ for some distribution ρ. Substituting this form of P into ε_6 gives (12). Condition 1 of Lemma 2.1, i.e., $\max_j \|A^{(j)}\|_1 \leqslant \min_i \|A_{(i)}\|_1$, allows us to pass from (13) to (14). Finally, to pass from (14) to (15) we note that the two maximizations over i in (14) involve the same expression, thus externalizing the maximization has no effect.

$$\begin{aligned}
\varepsilon_6(P) &= \max_j \left[\alpha \left(\sum_i \frac{\|A_{(i)}\|_1 \cdot |A_{ij}|}{\rho_i} \right)^{1/2} + \beta \max_i \frac{\|A_{(i)}\|_1}{\rho_i} \right] && (12) \\
&\leqslant \max_j \left[\alpha \left(\max_i \frac{\|A_{(i)}\|_1}{\rho_i} \cdot \sum_i |A_{ij}| \right)^{1/2} + \beta \max_i \frac{\|A_{(i)}\|_1}{\rho_i} \right] \\
&= \max_j \left[\alpha \left(\max_i \frac{\|A_{(i)}\|_1}{\rho_i} \cdot \|A^{(j)}\|_1 \right)^{1/2} + \beta \max_i \frac{\|A_{(i)}\|_1}{\rho_i} \right] \\
&\leqslant \alpha \left(\max_i \frac{\|A_{(i)}\|_1}{\rho_i} \cdot \max_j \|A^{(j)}\|_1 \right)^{1/2} + \beta \max_i \frac{\|A_{(i)}\|_1}{\rho_i} && (13) \\
&\leqslant \alpha \left(\max_i \frac{\|A_{(i)}\|_1}{\rho_i} \cdot \min_i \|A_{(i)}\|_1 \right)^{1/2} + \beta \max_i \frac{\|A_{(i)}\|_1}{\rho_i} && (14) \\
&\leqslant \max_i \left[\alpha \left(\frac{\|A_{(i)}\|_1}{\rho_i} \cdot \min_i \|A_{(i)}\|_1 \right)^{1/2} + \beta \frac{\|A_{(i)}\|_1}{\rho_i} \right] && (15) \\
&\leqslant \max_i \left[\alpha \frac{\|A_{(i)}\|_1}{\sqrt{\rho_i}} + \beta \max_i \frac{\|A_{(i)}\|_1}{\rho_i} \right] \\
&= \varepsilon_5(P) \ .
\end{aligned}$$

C Proof of Theorem 4.1

Proof of Theorem 4.1. We start by computing the value of ε_1 as a function of s, δ, for the probability distribution P_0 minimizing ε_5. Recall that in deriving (11) we established that $\varepsilon_5(P_0) = \zeta_0$, where ζ_0 is such that $\sum_{i=1}^m \rho_i(\zeta_0) = 1$, i.e.,

$$1 = \sum_{i=1}^m \left(\frac{\alpha\|A_{(i)}\|_1}{2\zeta_0} + \sqrt{\left(\frac{\alpha\|A_{(i)}\|_1}{2\zeta_0}\right)^2 + \frac{\beta\|A_{(i)}\|_1}{\zeta_0}} \right)^2 \leqslant \sum_{i=1}^m \frac{\alpha^2\|A_{(i)}\|_1^2}{\zeta_0^2} + \frac{2\beta\|A_{(i)}\|_1}{\zeta_0} \ . \tag{16}$$

This yields the following quadratic equation in ζ_0

$$\zeta_0^2 - \zeta_0 \cdot 2\beta\|A\|_1 - \alpha^2 \sum_i \|A_{(i)}\|_1^2 \leqslant 1 \tag{17}$$

Treating (17) as an equality and bounding the larger root of the resulting quadratic equation we get

$$\zeta_0 = O\left(\beta\|A\|_1 + \alpha\sqrt{\sum_i \|A_{(i)}\|_1^2} \right) = O\left(\frac{\log\left(\frac{m+n}{\delta}\right)\|A\|_1}{s} + \sqrt{\frac{\log\left(\frac{m+n}{\delta}\right)\sum_i \|A_{(i)}\|_1^2}{s}} \right) \tag{18}$$

The second equality is obtain by replacing α, β with their corresponding expression given in (2): $\alpha = \sqrt{\log((m+n)/\delta)/s}$ and $\beta = \log((m+n)/\delta)/(3s)$. Recall that to prove Theorem 2.2 we proved that if A meets the conditions of Definition 2.1, then

$$\min_P \varepsilon_1(P) = \Theta(\zeta_0) \ .$$

It follows that for $\varepsilon^* = \min_P \varepsilon_1(P)$,

$$s = O\left(\frac{\log((m+n)/\delta)\sum_i \|A_{(i)}\|_1}{\varepsilon^*} + \frac{\log((m+n)/\delta)\sum_i \|A_{(i)}\|_1^2}{(\varepsilon^*)^2} \right)$$

The theorem now follows by taking $\varepsilon^* = \varepsilon\|A\|$. □

D Efficient Parallel Reservoir Sampling

Assume we are to receive an unknown-length stream of items, each item having weight w_i and that we want to sample a single item from the stream so that the probability each item has to be chosen is $p_i = w_i/W$, where $W = \sum_i w_i$. Reservoir sampling is the classic solution to this problem: select the very first item in the stream as the "current" sample and from then on have each successive item i replace the current sample with probability w_i/Z_i, where $Z_i = \sum_{j \leqslant i} w_j$.

Assume now that, instead, we wanted to take $s > 1$ items from the stream, but as if the stream was a set and we could sample it *with* replacement. One way to do this is to execute s independent reservoir samplers as above in parallel, as was pointed out in [10]. Implementing this solution though in a straightforward manner requires $O(s)$ active memory, i.e., RAM, since each sampler must be able to replace its sample with the current item at the pace of the stream and, moreover, $O(s)$ randomized operations *per item in the stream*, i.e., $\Omega(Ns)$ operations where N is the length of the stream.

When s is large this can be problematic and becomes impossible when $s = \mathrm{nnz}(B)$ is such that B is too large to fit in main memory. Below we describe an algorithm that requires $O(\log s)$ *active* memory and $O(1)$ operations per item. The idea is that since the samplers are independent, we can simulate the process above by determining for each item i the (random) number of samplers, r_i, that would have replaced their current sample with item i when item i appeared. This random variable is Bernouli distributed and can be sampled efficiently. If this number is greater than 0, we write item i along with r_i to durable storage (disk) and process the next item in the stream. This processing generates a sketch of the stream on disk, the length of which can be shown to be bounded by $O(s \log(bN))$, where $b := \max_i |w_i| / \min_{i:\, w_i \neq 0} |w_i|$. When the stream finishes, we process the sketch we generated from *end to beginning* as follows: for each pair $\langle \text{item } j, r_j \rangle$ we encounter in the sketch we process the r_j update operations as the throwing of r_j balls into s bins uniformly at random (this is because in the naive setting whether item j will replace the current sample, X, of a particular sampler is independent of X). Notice that since we are going over the sketch backwards, the very first ball we place in a bin corresponds to the very last update of the sampler in the original execution. Thus, for each bin, we ignore all but the first first ball placement and we stop as soon as each bin has received a ball (thus we also avoid simulating the "irrelevant" part of the naive computation). Performing this simulation only requires a bit-vector of length s in active memory.

Finally, we can avoid even the cost of the bit-vector, as follows. Imagine that we have encountered an item i with $r_i = 100$. A naive simulation would choose 100 bins and put item i in those bins that happen to be empty. All we really need, though, is to know how many of the r_i balls would be assigned to empty bins, a random variable having the distribution $\text{mixed-bag}(s, m, k)$ detailed below. To sample this random variable we do not need to know *which* bins are empty but only *how many* are empty, meaning we need only to maintain a counter, i.e., $O(\log s)$ bits of active memory.

```
Input: $A$, $s$
$W = 0$
$T$ = empty stack
$S$ = empty list
for $(i, w_i) \in$ stream do
    $W \leftarrow W + w_i$
    $p = w_i/W$
    $k = \mathrm{binomial}(s, p)$
    if $k > 0$ then
        Push $(i, k)$ into $T$
while $\mathrm{length}(T) > 0$ and $\mathrm{length}(S) < s$ do
    $(i, k) = \mathrm{pop}(T)$
    Pick $t$ from a $\text{mixed-bag}(s, s - \mathrm{length}(S), k)$ distribution.
    Add $t$ instances of $i$ to $S$.
```

The distribution $\text{mixed-bag}(s, m, k)$ assigns each integer t probability $\binom{m}{t}\binom{s-m}{k-t}/\binom{s}{k}$. In words, assume we have a bag containing s pieces of fruit only m of which are raisins. If we pick k fruit randomly from the bag, then the number of raisins picked will be distributed as $\text{mixed-bag}(s, m, k)$.