Algorithms and Lower Bounds for Sparse Recovery

by

Eric Price

Submitted to the Department of Electrical Engineering and Computer Science

in partial fulfillment of the requirements for the degree of

Master of Engineering in Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

February 2010
[September 2010]

Author .....
Department of Electrical Engineering and Computer Science
February 2, 2010

Certified by .....
Piotr Indyk
Associate Professor
Thesis Supervisor

Accepted by .....
Christopher J. Terman
Chairman, Department Committee on Graduate Theses

Algorithms and Lower Bounds for Sparse Recovery

by

Eric Price

Submitted to the Department of Electrical Engineering and Computer Science
on February 2, 2010, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Computer Science

Abstract

We consider the following k-sparse recovery problem: design a distribution of $m \times n$ matrix A, such that for any signal x, given Ax with high probability we can efficiently recover $\hat{x}$ satisfying $\|x - \hat{x}\|_1 \leq C \min_{k\text{-sparse } x'} \|x - x'\|_1$. It is known that there exist such distributions with $m = O(k \log(n/k))$ rows; in this thesis, we show that this bound is tight.

We also introduce the *set query algorithm*, a primitive useful for solving special cases of sparse recovery using less than $\Theta(k \log(n/k))$ rows. The set query algorithm estimates the values of a vector $x \in \mathbb{R}^n$ over a support S of size k from a randomized sparse binary linear sketch Ax of size $O(k)$. Given Ax and S, we can recover x' with $\|x' - x_S\|_2 \leq \epsilon \|x - x_S\|_2$ with probability at least $1 - k^{-\Omega(1)}$. The recovery takes $O(k)$ time.

While interesting in its own right, this primitive also has a number of applications. For example, we can:

- Improve the sparse recovery of Zipfian distributions $O(k \log n)$ measurements from a $1 + \epsilon$ approximation to a $1 + o(1)$ approximation, giving the first such approximation when $k \leq O(n^{1-\epsilon})$.
- Recover block-sparse vectors with $O(k)$ space and a $1 + \epsilon$ approximation. Previous algorithms required either $\omega(k)$ space or $\omega(1)$ approximation.

Thesis Supervisor: Piotr Indyk
Title: Associate Professor

Acknowledgments

Foremost I thank my advisor Piotr Indyk for much helpful advice.

Some of this work has appeared as a publication with coauthors. I would like to thank my coauthors Khanh Do Ba, Piotr Indyk, and David Woodruff for their contributions.

Contents

List of Figures

Chapter 1

Introduction

In recent years, a new "linear" approach for obtaining a succinct approximate representation of n-dimensional vectors (or signals) has been discovered. For any signal x, the representation is equal to Ax, where A is an $m \times n$ matrix, or possibly a random variable chosen from some distribution over such matrices. The vector Ax is often referred to as the *measurement vector* or *linear sketch* of x. Although m is typically much smaller than n, the sketch Ax often contains plenty of useful information about the signal x.

A particularly useful and well-studied problem is that of *stable sparse recovery*. The problem is typically defined as follows: for some norm parameters p and q and an approximation factor $C > 0$, given Ax, recover a vector x' such that

$$\|x' - x\|_p \leq C \cdot \mathrm{Err}_k^q(x), \text{ where } \mathrm{Err}_k^q(x) = \min_{k\text{-sparse } \hat{x}} \|\hat{x} - x\|_q \tag{1.1}$$

where we say that $\hat{x}$ is k-sparse if it has at most k non-zero coordinates. Sparse recovery has applications to numerous areas such as data stream computing [40, 29] and compressed sensing [6, 17], notably for constructing imaging systems that acquire images directly in compressed form (e.g., [18, 41]). The problem has been a subject of extensive study over the last several years, with the goal of designing schemes that enjoy good "compression rate" (i.e., low values of m) as well as good algorithmic properties (i.e., low encoding and recovery times). It is known that there

exist distributions of matrices A and associated recovery algorithms that for any x with high probability produce approximations x' satisfying Equation (1.1) with $\ell_p = \ell_q = \ell_1$, constant approximation factor $C = 1 + \epsilon$, and sketch length $m = O(k \log(n/k))$[1]. Similar results for other combinations of ℓ_p/ℓ_q norms are known as well. In comparison, using a *non-linear* approach, one could obtain a shorter sketch of length $O(k)$: it suffices to store the k coefficients with the largest absolute values, together with their indices.

This thesis has two main parts. The first part is a tight lower bound on the number of measurements for general sparse recovery, and the second part introduces a useful primitive for circumventing the lower bound in special cases of sparse recovery.

1.1 A lower bound

Surprisingly, it was not known if the $O(k \log(n/k))$ bound on measurements for linear sketching could be improved upon[2], and $O(k)$ sketch length was known to suffice if the signal vectors x are required to be *exactly* k-sparse [1]. This raised hope that the $O(k)$ bound might be achievable even for general vectors x. Such a scheme would have been of major practical interest, since the sketch length determines the compression ratio, and for large n any extra $\log n$ factor worsens that ratio tenfold.

In the first part of this thesis we show that, unfortunately, such an improvement is not possible. We address two types of recovery scheme:

- A *deterministic* one, which involves a fixed matrix A and a recovery algorithm which work for all signals x. The aforementioned results of [6] and others are examples of such schemes.

- A *randomized* one, where the matrix A is chosen at random from some distribution, and for each signal x the recovery procedure is correct with constant

[1] In particular, a random Gaussian matrix [10] or a random sparse binary matrix ([28], building on [9, 13]) has this property with overwhelming probability. See [27] for an overview.

[2] The lower bound of $\Omega(k \log(n/k))$ was known to hold for specific recovery algorithms, specific matrix types, or other recovery scenarios. See Section 2.2 for an overview.

probability. Some of the early schemes proposed in the data stream literature (e.g., [9, 13]) belong to this category.

Our main result in this section is that, even in the randomized case, the sketch length m must be at least $\Omega(k \log(n/k))$. By the aforementioned result of [6] this bound is tight. Thus, our results show that the linear compression is inherently more costly than the simple non-linear approach. Chapters 3 and 4 contain this result.

1.2 Set query algorithm

Because it has proved impossible to improve on the sketch size in the general sparse recovery problem, recently there has been a body of work on more restricted problems that are amenable to more efficient solutions. This includes *model-based compressive sensing* [4], which imposing additional constraints (or *models*) on x beyond near-sparsity. Examples of models include *block sparsity*, where the large coefficients tend to cluster together in blocks [4, 21]; *tree sparsity*, where the large coefficients form a rooted, connected tree structure [4, 37]; and being *Zipfian*, where we require that the histogram of coefficient size follow a *Zipfian* (or *power law*) distribution.

A sparse recovery algorithm needs to perform two tasks: locating the large coefficients of x and estimating their value. Existing algorithms perform both tasks at the same time. In contrast, we propose decoupling these tasks. In models of interest, including Zipfian signals and block-sparse signals, existing techniques can locate the large coefficients more efficiently or accurately than they can estimate them. Prior to this work, however, estimating the large coefficients after finding them had no better solution than the general sparse recovery problem. We fill this gap by giving an optimal method for estimating the values of the large coefficients after locating them. We refer to this task as the *Set Query Problem*[3].

Main result. (Set Query Algorithm.) We give a randomized distribution over $O(k) \times n$ binary matrices A such that, for any vector $x \in \mathbb{R}^n$ and set $S \subseteq \{1, \ldots, n\}$

[3]The term "set query" is in contrast to "point query," used in e.g. [13] for estimation of a single coordinate.

with $|S| = k$, we can recover an x' from $Ax + \nu$ and S with

$$\|x' - x_S\|_2 \leq \epsilon(\|x - x_S\|_2 + \|\nu\|_2)$$

where $x_S \in \mathbb{R}^n$ equals x over S and zero elsewhere. The matrix A has $O(1)$ non-zero entries per column, recovery succeeds with probability $1 - k^{-\Omega(1)}$, and recovery takes $O(k)$ time. This can be achieved for arbitrarily small $\epsilon > 0$, using $O(k/\epsilon^2)$ rows. We achieve a similar result in the ℓ_1 norm.

The set query problem is useful in scenarios when, given the sketch of x, we have some alternative methods for discovering a "good" support of an approximation to x. This is the case, e.g., in block-sparse recovery, where (as we show in this paper) it is possible to identify "heavy" blocks using other methods. It is also a natural problem in itself. In particular, it generalizes the well-studied *point query problem* [13], which considers the case that S is a singleton. We note that although the set query problem for sets of size k can be reduced to k instances of the point query problem, this reduction is less space-efficient than the algorithm we propose, as elaborated in Chapters 5 and 6.

Chapter 2

Overview of Lower Bound

This chapter gives intuition and related work for our lower bounds. Proofs and further detail lie in Chapters 3 and 4.

2.1 Our techniques

On a high level, our approach is simple and natural, and utilizes the packing approach: we show that any two "sufficiently" different vectors x and x' are mapped to images Ax and Ax' that are "sufficiently" different themselves, which requires that the image space is "sufficiently" high-dimensional. However, the actual arguments are somewhat subtle.

Consider first the (simpler) deterministic case. We focus on signals $x = y + z$, where y can be thought of as the "head" of the signal and z as the "tail". The "head" vectors y come from a set Y that is a binary error-correcting code, with a minimum distance $\Omega(k)$, where each codeword has weight k. On the other hand, the "tail" vectors z come from an ℓ_1 ball (say B) with a radius that is a small fraction of k. It can be seen that for any two elements $y, y' \in Y$, the balls $y + B$ and $y' + B$, as well as their images, must be disjoint. At the same time, since all vectors x live in a "large" ℓ_1 ball B' of radius $O(k)$, all images Ax must live in a set AB'. The key observation is that the set AB' is a scaled version of $A(y + B)$ and therefore the ratios of their volumes can be bounded by the scaling factor to the power of the dimension m. Since

the number of elements of Y is large, this gives a lower bound on m.

Unfortunately, the aforementioned approach does not seem to extend to the randomized case. A natural approach would be to use Yao's principle, and focus on showing a lower bound for a scenario where the matrix A is fixed while the vectors $x = y + z$ are "random". However, this approach fails, in a very strong sense. Specifically, we are able to show that there is a distribution over matrices A with *only* $O(k)$ rows so that for a fixed $y \in Y$ and z chosen uniformly at random from the small ball B, we can recover y from $A(y + z)$ with high probability. In a nutshell, the reason is that a random vector from B has an ℓ_2 norm that is much smaller than the ℓ_2 norm of elements of Y (even though the ℓ_1 norms are comparable). This means that the vector x is "almost" k-sparse (in the ℓ_2 norm), which enables us to achieve the $O(k)$ measurement bound.

Instead, we resort to an altogether different approach, via *communication complexity* [36]. We start by considering a "discrete" scenario where both the matrix A and the vectors x have entries restricted to the polynomial range $\{-n^c \dots n^c\}$ for some $c = O(1)$. In other words, we assume that the matrix and vector entries can be represented using $O(\log n)$ bits. In this setting we show the following: there is a method for encoding a sequence of $d = O(k \log(n/k) \log n)$ bits into a vector x, so that any sparse recovery algorithm can recover that sequence given Ax. Since each entry of Ax conveys only $O(\log n)$ bits, it follows that the number m of rows of A must be $\Omega(k \log(n/k))$.

The encoding is performed by taking

$$x = \sum_{j=1}^{\log n} D^j x_j,$$

where $D = O(1)$ and the x_j are chosen from the error-correcting code Y defined as in the deterministic case. The intuition behind this approach is that a good ℓ_1/ℓ_1 approximation to x reveals most of the bits of $x_{\log n}$. This enables us to identify $x_{\log n}$ exactly using error correction. We could then compute $Ax - Ax_{\log n} = A(\sum_{j=1}^{\log n - 1} D^j x_j)$, and identify $x_{\log n - 1} \dots x_1$ in a recursive manner. The only obstacle to completing this

argument is that we would need the recovery algorithm to work for *all* x_i, which would require lower probability of algorithm failure (roughly $1/\log n$). To overcome this problem, we replace the encoding argument by a reduction from a related communication complexity problem called Augmented Indexing. This problem has been used in the data stream literature [11, 32] to prove lower bounds for linear algebra and norm estimation problems. Since the problem has communication complexity of $\Omega(d)$, the conclusion follows.

We apply the argument to arbitrary matrices A by representing them as a sum $A' + A''$, where A' has $O(\log n)$ bits of precision and A'' has "small" entries. We then show that $A'x = A(x + s)$ for some s with $\|s\|_1 < n^{-\Omega(1)} \|x\|_1$. In the communication game, this means we can transmit $A'x$ and recover $x_{\log n}$ from $A'(\sum_{j=1}^{\log n} D^j x_j) = A(\sum_{j=1}^{\log n} D^j x_j + s)$. This means that the Augmented Indexing reduction applies to arbitrary matrices as well.

2.2 Related Work

There have been a number of earlier works that have, directly or indirectly, shown lower bounds for various models of sparse recovery and certain classes of matrices and algorithms. Specifically, one of the most well-known recovery algorithms used in compressed sensing is ℓ_1-minimization, where a signal $x \in \mathbb{R}^n$ measured by matrix A is reconstructed as

$$x' := \arg\min_{\hat{x}:\, A\hat{x}=Ax} \|\hat{x}\|_1.$$

Kashin and Temlyakov [34] gave a characterization of matrices A for which the above recovery algorithm yields the ℓ_2/ℓ_1 guarantee, i.e.,

$$\|x - x'\|_2 \leq Ck^{-1/2} \min_{k\text{-sparse } \hat{x}} \|x - \hat{x}\|_1$$

for some constant C, from which it can be shown that such an A must have $m = \Omega(k \log(n/k))$ rows.

Note that the ℓ_2/ℓ_1 guarantee is somewhat stronger than the ℓ_1/ℓ_1 guarantee in-

vestigated in this paper. Specifically, it is easy to observe that if the approximation x' itself is required to be $O(k)$-sparse, then the ℓ_2/ℓ_1 guarantee implies the ℓ_1/ℓ_1 guarantee (with a somewhat higher approximation constant). For the sake of simplicity, in this paper we focus mostly on the ℓ_1/ℓ_1 guarantee. However, our lower bounds apply to the ℓ_2/ℓ_1 guarantee as well: see footnote on page 31.

On the other hand, instead of assuming a specific recovery algorithm, Wainwright [44] assumes a specific (randomized) measurement matrix. More specifically, the author assumes a k-sparse binary signal $x \in \{0, \alpha\}^n$, for some $\alpha > 0$, to which is added i.i.d. standard Gaussian noise in each component. The author then shows that with a random Gaussian matrix A, with each entry also drawn i.i.d. from the standard Gaussian, we cannot hope to recover x from Ax with any sub-constant probability of error unless A has $m = \Omega(\frac{1}{\alpha^2} \log \frac{n}{k})$ rows. The author also shows that for $\alpha = \sqrt{1/k}$, this is tight, i.e., that $m = \Theta(k \log(n/k))$ is both necessary and sufficient. Although this is only a lower bound for a specific (random) matrix, it is a fairly powerful one and provides evidence that the often observed upper bound of $O(k \log(n/k))$ is likely tight.

More recently, Dai and Milenkovic [15], extending on [24] and [26], showed an upper bound on superimposed codes that translates to a lower bound on the number of rows in a compressed sensing matrix that deals only with k-sparse signals but can tolerate measurement noise. Specifically, if we assume a k-sparse signal $x \in ([-t, t] \cap \mathbb{Z})^n$, and that arbitrary noise $\mu \in \mathbb{R}^n$ with $\|\mu\|_1 < d$ is added to the measurement vector Ax, then if exact recovery is still possible, A must have had $m \geq Ck \log n / \log k$ rows, for some constant $C = C(t, d)$ and sufficiently large n and k.[1]

Concurrently with our work, Foucart et al. [25] have done an analysis of Gelfand widths of ℓ_p-balls that implies a lower bound on sparse recovery. Their work essentially matches our deterministic lower bound, and does not extend to the randomized case.

[1]Here A is assumed to have its columns normalized to have ℓ_1-norm 1. This is natural since otherwise we could simply scale A up to make the image points Ax arbitrarily far apart, effectively nullifying the noise.

2.3 Preliminaries

2.3.1 Notation

For $n \in \mathbb{Z}^+$, we denote $\{1, \ldots, n\}$ by $[n]$. Suppose $x \in \mathbb{R}^n$. Then for $i \in [n]$, $x_i \in \mathbb{R}$ denotes the value of the i-th coordinate in x. As an exception, $e_i \in \mathbb{R}^n$ denotes the elementary unit vector with a one at position i. For $S \subseteq [n]$, x_S denotes the vector $x' \in R^n$ given by $x'_i = x_i$ if $i \in S$, and $x'_i = 0$ otherwise. We use $\text{supp}(x)$ to denote the support of x. We use upper case letters to denote sets, matrices, and random distributions. We use lower case letters for scalars and vectors.

We use $B_p^n(r)$ to denote the ℓ_p ball of radius r in $\mathbb{R}^n$; we skip the superscript n if it is clear from the context. For any vector x, we use $\|x\|_0$ to denote the "ℓ_0 norm of x", i.e., the number of non-zero entries in x.

2.3.2 Sparse recovery

In this paper we focus on recovering sparse approximations x' that satisfy the following C-approximate ℓ_1/ℓ_1 guarantee with sparsity parameter k:

$$\|x - x'\|_1 \leq C \min_{k\text{-sparse } \hat{x}} \|x - \hat{x}\|_1 . \tag{2.1}$$

We define a C-approximate *deterministic* ℓ_1/ℓ_1 recovery algorithm to be a pair $(A, \mathscr{A})$ where A is an $m \times n$ observation matrix and $\mathscr{A}$ is an algorithm that, for any x, maps Ax (called the *sketch* of x) to some x' that satisfies Equation (2.1).

We define a C-approximate *randomized* ℓ_1/ℓ_1 recovery algorithm to be a pair $(A, \mathscr{A})$ where A is a *random variable* chosen from some distribution over $m \times n$ measurement matrices, and $\mathscr{A}$ is an algorithm which, for any x, maps a pair (A, Ax) to some x' that satisfies Equation (2.1) with probability at least 3/4.

Chapter 3

Deterministic Lower Bound

3.1 Proof

We will prove a lower bound on m for any C-approximate deterministic recovery algorithm. First we use a discrete volume bound (Lemma 1) to find a large set Y of points that are at least k apart from each other. Then we use another volume bound (Lemma 2) on the images of small ℓ_1 balls around each point in Y. If m is too small, some two images collide. But the recovery algorithm, applied to a point in the collision, must yield an answer close to two points in Y. This is impossible, so m must be large.

Lemma 1. *(Gilbert-Varshamov) For any $q, k \in \mathbb{Z}^+, \epsilon \in \mathbb{R}^+$ with $\epsilon < 1 - 1/q$, there exists a set $Y \subset \{0,1\}^{qk}$ of binary vectors with exactly k ones, such that Y has minimum Hamming distance $2\epsilon k$ and*

$$\log |Y| > (1 - H_q(\epsilon))k \log q$$

where H_q is the q-ary entropy function $H_q(x) = -x \log_q \frac{x}{q-1} - (1-x)\log_q(1-x)$.

See appendix for proof.

Lemma 2. *Take an $m \times n$ real matrix A, positive reals ϵ, p, λ, and $Y \subset B_p^n(\lambda)$. If $|Y| > (1 + 1/\epsilon)^m$, then there exist $z, \overline{z} \in B_p^n(\epsilon\lambda)$ and $y, \overline{y} \in Y$ with $y \neq \overline{y}$ and*

$A(y+z) = A(\overline{y}+\overline{z})$.

Proof. If the statement is false, then the images of all $|Y|$ balls $\{y + B_p^n(\epsilon\lambda) \mid y \in Y\}$ are disjoint. However, those balls all lie within $B_p^n((1+\epsilon)\lambda)$, by the bound on the norm of Y. A volume argument gives the result, as follows.

Let $S = AB_p^n(1)$ be the image of the n-dimensional ball of radius 1 in m-dimensional space. This is a polytope with some volume V. The image of $B_p^n(\epsilon\lambda)$ is a linearly scaled S with volume $(\epsilon\lambda)^m V$, and the volume of the image of $B_p^n((1+\epsilon)\lambda)$ is similar with volume $((1+\epsilon)\lambda)^m V$. If the images of the former are all disjoint and lie inside the latter, we have $|Y|\,(\epsilon\lambda)^m V \leq ((1+\epsilon)\lambda)^m V$, or $|Y| \leq (1+1/\epsilon)^m$. If Y has more elements than this, the images of some two balls $y + B_p^n(\epsilon\lambda)$ and $\overline{y} + B_p^n(\epsilon\lambda)$ must intersect, implying the lemma. □

Theorem 3. *Any C-approximate deterministic recovery algorithm must have*

$$m \geq \frac{1 - H_{\lfloor n/k \rfloor}(1/2)}{\log(4+2C)} k \log \left\lfloor \frac{n}{k} \right\rfloor .$$

Proof. Let Y be a maximal set of k-sparse n-dimensional binary vectors with minimum Hamming distance k, and let $\gamma = \frac{1}{3+2C}$. By Lemma 1 with $q = \lfloor n/k \rfloor$ we have $\log |Y| > (1 - H_{\lfloor n/k \rfloor}(1/2)) k \log \lfloor n/k \rfloor$.

Suppose that the theorem is not true; then $m < \log|Y| \,/\, \log(4+2C) = \log|Y| \,/\, \log(1+1/\gamma)$, or $|Y| > (1+\frac{1}{\gamma})^m$. Hence Lemma 2 gives us some $y, \overline{y} \in Y$ and $z, \overline{z} \in B_1(\gamma k)$ with $A(y+z) = A(\overline{y}+\overline{z})$.

Let w be the result of running the recovery algorithm on $A(y+z)$. By the definition of a deterministic recovery algorithm, we have

$$\begin{aligned}
\|y+z-w\|_1 &\leq C \min_{k\text{-sparse } \hat{y}} \|y+z-\hat{y}\|_1 \\
\|y-w\|_1 - \|z\|_1 &\leq C\,\|z\|_1 \\
\|y-w\|_1 &\leq (1+C)\,\|z\|_1 \leq (1+C)\gamma k = \tfrac{1+C}{3+2C}\,k,
\end{aligned}$$

and similarly $\|\overline{y} - w\|_1 \leq \frac{1+C}{3+2C} k$, so

$$\|y - \overline{y}\|_1 \leq \|y - w\|_1 + \|\overline{y} - w\|_1 = \frac{2+2C}{3+2C} k < k.$$

But this contradicts the definition of Y, so m must be large enough for the guarantee to hold. □

Corollary 4. *If C is a constant bounded away from zero, then $m = \Omega(k \log(n/k))$.*

This suffices to lower bound the number of measurements for deterministic matrices. The next section gives evidence that this proof is hard to generalize to randomized matrices; hence in the next chapter we will take a different approach.

3.2 Randomized upper bound for uniform noise

The standard way to prove a randomized lower bound is to find a distribution of hard inputs, and to show that any deterministic algorithm is likely to fail on that distribution. In our context, we would like to define a "head" random variable y from a distribution Y and a "tail" random variable z from a distribution Z, such that any algorithm given the sketch of $y + z$ must recover an incorrect y with non-negligible probability.

Using our deterministic bound as inspiration, we could take Y to be uniform over a set of k-sparse binary vectors of minimum Hamming distance k and Z to be uniform over the ball $B_1(\gamma k)$ for some constant $\gamma > 0$. Unfortunately, as the following theorem shows, one can actually perform a recovery of such vectors using only $O(k)$ measurements; this is because $\|z\|_2$ is very small (namely, $\tilde{O}(k/\sqrt{n})$) with high probability.

Theorem 5. *Let $Y \subset \mathbb{R}^n$ be a set of signals with the property that for every distinct $y_1, y_2 \in Y$, $\|y_1 - y_2\|_2 \geq r$, for some parameter $r > 0$. Consider "noisy signals" $x = y + z$, where $y \in Y$ and z is a "noise vector" chosen uniformly at random from $B_1(s)$, for another parameter $s > 0$. Then using an $m \times n$ Gaussian measurement*

matrix $A = (1/\sqrt{m})(g_{ij})$, *where* g_{ij}*'s are i.i.d. standard Gaussians, we can recover* $y \in Y$ *from* $A(y+z)$ *with probability* $1 - 1/n$ *(where the probability is over both* A *and* z*), as long as*

$$s \leq O\left(\frac{rm^{1/2}n^{1/2-1/m}}{|Y|^{1/m}\log^{3/2} n}\right).$$

Because of this theorem, our geometric lower bound for deterministic matrices is hard to generalize to randomized ones. Hence in Chapter 4 we use a different technique to extend our results.

To prove the theorem we will need the following two lemmas.

Lemma 6. *For any* $\delta > 0$, $y_1, y_2 \in Y$, $y_1 \neq y_2$, *and* $z \in \mathbb{R}^n$, *each of the following holds with probability at least* $1 - \delta$*:*

- $\|A(y_1 - y_2)\|_2 \geq \frac{\delta^{1/m}}{3}\|y_1 - y_2\|_2$, *and*
- $\|Az\|_2 \leq (\sqrt{(8/m)\log(1/\delta)} + 1)\|z\|_2$.

Proof. By standard arguments (see, e.g., [30]), for any $D > 0$ we have

$$\Pr\left[\|A(y_1 - y_2)\|_2 \leq \frac{\|y_1 - y_2\|_2}{D}\right] \leq \left(\frac{3}{D}\right)^m$$

and

$$\Pr[\|Az\|_2 \geq D\|z\|_2] \leq e^{-m(D-1)^2/8}.$$

Setting both right-hand sides to δ yields the lemma. □

Lemma 7. *A random vector* z *chosen uniformly from* $B_1(s)$ *satisfies*

$$\Pr[\|z\|_2 > \alpha s \log n/\sqrt{n}] < 1/n^{\alpha-1}.$$

Proof. Consider the distribution of a single coordinate of z, say, z_1. The probability density of $|z_1|$ taking value $t \in [0, s]$ is proportional to the $(n-1)$-dimensional volume of $B_1^{(n-1)}(s-t)$, which in turn is proportional to $(s-t)^{n-1}$. Normalizing to ensure the probability integrates to 1, we derive this probability as

$$p(|z_1| = t) = \frac{n}{s^n}(s-t)^{n-1}.$$

It follows that, for any $D \in [0, s]$,

$$\Pr[|z_1| > D] = \int_D^s \frac{n}{s^n}(s-t)^{n-1}\, dt = (1 - D/s)^n.$$

In particular, for any $\alpha > 1$,

$$\begin{aligned}\Pr[|z_1| &> \alpha s \log n/n] = (1 - \alpha \log n/n)^n < e^{-\alpha \log n} \\ &= 1/n^\alpha.\end{aligned}$$

Now, by symmetry this holds for every other coordinate z_i of z as well, so by the union bound

$$\Pr[\|z\|_\infty > \alpha s \log n/n] < 1/n^{\alpha-1},$$

and since $\|z\|_2 \leq \sqrt{n} \cdot \|z\|_\infty$ for any vector z, the lemma follows. □

Proof of theorem. In words, Lemma 6 says that A cannot bring faraway signal points too close together, and cannot blow up a small noise vector too much. Now, we already assumed the signals to be far apart, and Lemma 7 tells us that the noise is indeed small (in ℓ_2 distance). The result is that in the image space, the noise is not enough to confuse different signals. Quantitatively, applying the second part of Lemma 6 with $\delta = 1/n^2$, and Lemma 7 with $\alpha = 3$, gives us

$$\|Az\|_2 \leq O\left(\frac{\log^{1/2} n}{m^{1/2}}\right)\|z\|_2 \leq O\left(\frac{s \log^{3/2} n}{(mn)^{1/2}}\right) \tag{3.1}$$

with probability $\geq 1 - 2/n^2$. On the other hand, given signal $y_1 \in Y$, we know that every other signal $y_2 \in Y$ satisfies $\|y_1 - y_2\|_2 \geq r$, so by the first part of Lemma 6 with $\delta = 1/(2n|Y|)$, together with a union bound over every $y_2 \in Y$,

$$\|A(y_1 - y_2)\|_2 \geq \frac{\|y_1 - y_2\|_2}{3(2n|Y|)^{1/m}} \geq \frac{r}{3(2n|Y|)^{1/m}} \tag{3.2}$$

holds for every $y_2 \in Y$, $y_2 \neq y_1$, simultaneously with probability $1 - 1/(2n)$.

Finally, observe that as long as $\|Az\|_2 < \|A(y_1 - y_2)\|_2/2$ for every competing

signal $y_2 \in Y$, we are guaranteed that

$$\begin{aligned} \|A(y_1 + z) - Ay_1\|_2 &= \|Az\|_2 \\ &< \|A(y_1 - y_2)\|_2 - \|Az\|_2 \\ &\leq \|A(y_1 + z) - Ay_2\|_2 \end{aligned}$$

for every $y_2 \neq y_1$, so we can recover y_1 by simply returning the signal whose image is closest to our measurement point $A(y_1 + z)$ in ℓ_2 distance. To achieve this, we can chain Equations (3.1) and (3.2) together (with a factor of 2), to see that

$$s \leq O\left(\frac{rm^{1/2}n^{1/2-1/m}}{|Y|^{1/m}\log^{3/2} n}\right)$$

suffices. Our total probability of failure is at most $2/n^2 + 1/(2n) < 1/n$.

The main consequence of this theorem is that for the setup we used in Chapter 3 to prove a deterministic lower bound of $\Omega(k\log(n/k))$, if we simply draw the noise uniformly randomly from the same ℓ_1 ball (in fact, even one with a much larger radius, namely, polynomial in n), this "hard distribution" can be defeated with just $O(k)$ measurements:

Corollary 8. *If Y is a set of binary k-sparse vectors, as in Chapter 3, and noise z is drawn uniformly at random from $B_1(s)$, then for any constant $\epsilon > 0$, $m = O(k/\epsilon)$ measurements suffice to recover any signal in Y with probability $1 - 1/n$, as long as*

$$s \leq O\left(\frac{k^{3/2+\epsilon}n^{1/2-\epsilon}}{\log^{3/2} n}\right).$$

Proof. The parameters in this case are $r = k$ and $|Y| \leq \binom{n}{k} \leq (ne/k)^k$, so by Theorem 5, it suffices to have

$$s \leq O\left(\frac{k^{3/2+k/m}n^{1/2-(k+1)/m}}{\log^{3/2} n}\right).$$

Choosing $m = (k+1)/\epsilon$ yields the corollary. □

Chapter 4

Randomized Lower Bound

In the previous chapter, we gave a simple geometric proof of the lower bound for deterministic matrices. We also showed that this proof does not easily generalize to randomized matrices. In this chapter, we use a different approach to show a lower bound for randomized as well as deterministic matrices. We will use a reduction from a communication game with a known lower bound on communication complexity.

The communication game will show that a message Ax must have a large number of bits. To show that this implies a lower bound on the number of rows of A, we will need A to be discrete. But if A is poorly conditioned, straightforward rounding could dramatically change its recovery characteristics. Therefore in Section 4.1 we show that it is sufficient to consider orthonormal matrices A. In Section 4.2 we define our communication game and show a lower bound on its communication complexity. In Section 4.3 we prove the lower bound.

4.1 Reducing to orthonormal matrices

Before we discretize by rounding, we need to ensure that the matrix is well conditioned. We show that without loss of generality, the rows of A are orthonormal.

We can multiply A on the left by any invertible matrix to get another measurement matrix with the same recovery characteristics. If we consider the singular value decomposition $A = U\Sigma V^*$, where U and V are orthonormal and Σ is 0 off the

diagonal, this means that we can eliminate U and make the entries of Σ be either 0 or 1. The result is a matrix consisting of m orthonormal rows. For such matrices, we prove the following:

Lemma 9. *Consider any $m \times n$ matrix A with orthonormal rows. Let A' be the result of rounding A to b bits per entry. Then for any $v \in \mathbb{R}^n$ there exists an $s \in \mathbb{R}^n$ with $A'v = A(v - s)$ and $\|s\|_1 < n^2 2^{-b} \|v\|_1$.*

Proof. Let $A'' = A - A'$ be the roundoff error when discretizing A to b bits, so each entry of A'' is less than 2^{-b}. Then for any v and $s = A^T A''v$, we have $As = A''v$ and

$$\begin{aligned} \|s\|_1 &= \left\|A^T A''v\right\|_1 \leq \sqrt{n} \left\|A''v\right\|_1 \\ &\leq m\sqrt{n}2^{-b} \|v\|_1 \leq n^2 2^{-b} \|v\|_1 . \end{aligned}$$

□

4.2 Communication complexity

We use a few definitions and results from two-party communication complexity. For further background see the book by Kushilevitz and Nisan [36]. Consider the following communication game. There are two parties, Alice and Bob. Alice is given a string $y \in \{0,1\}^d$. Bob is given an index $i \in [d]$, together with $y_{i+1}, y_{i+2}, \ldots, y_d$. The parties also share an arbitrarily long common random string r. Alice sends a single message $M(y,r)$ to Bob, who must output y_i with probability at least $3/4$, where the probability is taken over r. We refer to this problem as Augmented Indexing. The communication cost of Augmented Indexing is the minimum, over all correct protocols, of the length of the message $M(y,r)$ on the worst-case choice of r and y.

The next theorem is well-known and follows from Lemma 13 of [38] (see also Lemma 2 of [3]).

Theorem 10. *The communication cost of* Augmented Indexing *is $\Omega(d)$.*

Proof. First, consider the private-coin version of the problem, in which both parties can toss coins, but do not share a random string r (i.e., there is no public coin). Consider any correct protocol for this problem. We can assume the probability of error of the protocol is an arbitrarily small positive constant by increasing the length of Alice's message by a constant factor (e.g., by independent repetition and a majority vote). Applying Lemma 13 of [38] (with, in their notation, $t = 1$ and $a = c' \cdot d$ for a sufficiently small constant $c' > 0$), the communication cost of such a protocol must be $\Omega(d)$. Indeed, otherwise there would be a protocol in which Bob could output y_i with probability greater than $1/2$ without any interaction with Alice, contradicting that $\Pr[y_i = 1/2]$ and that Bob has no information about y_i. Our theorem now follows from Newman's theorem (see, e.g., Theorem 2.4 of [35]), which shows that the communication cost of the best public coin protocol is at least that of the private coin protocol minus $O(\log d)$ (which also holds for one-round protocols). □

4.3 Randomized lower bound theorem

Theorem 11. *For any randomized ℓ_1/ℓ_1 recovery algorithm $(A, \mathscr{A})$, with approximation factor $C = O(1)$, A must have $m = \Omega(k \log(n/k))$ rows.*

Proof. We shall assume, without loss of generality, that n and k are powers of 2, that k divides n, and that the rows of A are orthonormal. The proof for the general case follows with minor modifications.

Let $(A, \mathscr{A})$ be such a recovery algorithm. We will show how to solve the Augmented Indexing problem on instances of size $d = \Omega(k \log(n/k) \log n)$ with communication cost $O(m \log n)$. The theorem will then follow by Theorem 10.

Let X be the maximal set of k-sparse n-dimensional binary vectors with minimum Hamming distance k. From Lemma 1 we have $\log |X| = \Omega(k \log(n/k))$. Let $d = \lfloor \log |X| \rfloor \log n$, and define $D = 2C + 3$.

Alice is given a string $y \in \{0,1\}^d$, and Bob is given $i \in [d]$ together with $y_{i+1}, y_{i+2}, \ldots, y_d$, as in the setup for Augmented Indexing.

Alice splits her string y into $\log n$ contiguous chunks $y^1, y^2, \ldots, y^{\log n}$, each containing $\lfloor \log |X| \rfloor$ bits. She uses y^j as an index into X to choose x_j. Alice defines

$$x = D^1 x_1 + D^2 x_2 + \cdots + D^{\log n} x_{\log n}.$$

Alice and Bob use the common randomness r to agree upon a random matrix A with orthonormal rows. Both Alice and Bob round A to form A' with $b = \lceil 2(1 + \log D) \log n \rceil = O(\log n)$ bits per entry. Alice computes $A'x$ and transmits it to Bob.

From Bob's input i, he can compute the value $j = j(i)$ for which the bit y_i occurs in y^j. Bob's input also contains $y_{i+1}, \ldots, y_n$, from which he can reconstruct $x_{j+1}, \ldots, x_{\log n}$, and in particular can compute

$$z = D^{j+1} x_{j+1} + D^{j+2} x_{j+2} + \cdots + D^{\log n} x_{\log n}.$$

Bob then computes $A'z$, and using $A'x$ and linearity, $A'(x - z)$. Then

$$\|x - z\|_1 \leq \sum_{i=1}^{j} kD^i < k \frac{D^{1+\log n}}{D-1} < kD^{2\log n}.$$

So from Lemma 9, there exists some s with $A'(x - z) = A(x - z - s)$ and

$$\|s\|_1 < n^2 2^{-2\log n - 2\log D \log n} \|x - z\|_1 < k.$$

Set $w = x - z - s$. Bob then runs the estimation algorithm $\mathscr{A}$ on A and Aw, obtaining w' with the property that with probability at least $3/4$,

$$\|w - w'\|_1 \leq C \min_{k\text{-sparse } \hat{w}} \|w - \hat{w}\|_1 .$$

Now,

$$\begin{aligned}\min_{k\text{-sparse } \hat{w}} \|w - \hat{w}\|_1 &\leq \left\|w - D^j x_j\right\|_1 \\ &\leq \|s\|_1 + \sum_{i=1}^{j-1} \left\|D^i x_i\right\|_1 \\ &< k(1 + D + D^2 + \cdots + D^{j-1}) \\ &< k \cdot \frac{D^j}{D-1}.\end{aligned}$$

Hence

$$\begin{aligned}\left\|D^j x_j - w'\right\|_1 &\leq \left\|D^j x_j - w\right\|_1 + \|w - w'\|_1 \\ &\leq (1 + C) \left\|D^j x_j - w\right\|_1 \\ &< \frac{kD^j}{2}.\end{aligned}$$

And since the minimum Hamming distance in X is k, this means $\|D^j x_j - w'\|_1 < \|D^j x' - w'\|_1$ for all $x' \in X, x' \neq x_j$[1]. So Bob can correctly identify x_j with probability at least 3/4. From x_j he can recover y^j, and hence the bit y_i that occurs in y^j.

Hence, Bob solves Augmented Indexing with probability at least 3/4 given the message $A'x$. The entries in A' and x are polynomially bounded integers (up to scaling of A'), and so each entry of $A'x$ takes $O(\log n)$ bits to describe. Hence, the communication cost of this protocol is $O(m \log n)$. By Theorem 10, $m \log n = \Omega(k \log(n/k) \log n)$, or $m = \Omega(k \log(n/k))$. □

[1]Note that these bounds would still hold with minor modification if we replaced the ℓ_1/ℓ_1 guarantee with the ℓ_2/ℓ_1 guarantee, so the same result holds in that case.

Chapter 5

Overview of Set Query Algorithm

The previous chapters show that the general sparse recovery problem cannot be solved with fewer than $\Theta(k \log(n/k))$ linear measurements. However, many problems that can be cast as sparse recovery problems are special cases of the class; one can still hope to solve the special cases more efficiently. The rest of this thesis introduces the *set query problem*, a useful primitive for this kind of problem. We show a particularly efficient algorithm for this problem and two applications where our algorithm allows us to improve upon the best known results. Recall from the introduction that we achieve the following:

We give a randomized distribution over $O(k) \times n$ binary matrices A such that, for any vector $x \in \mathbb{R}^n$ and set $S \subseteq \{1, \ldots, n\}$ with $|S| = k$, we can recover an x' from $Ax + \nu$ and S with

$$\|x' - x_S\|_2 \leq \epsilon(\|x - x_S\|_2 + \|\nu\|_2)$$

where $x_S \in \mathbb{R}^n$ equals x over S and zero elsewhere. The matrix A has $O(1)$ non-zero entries per column, recovery succeeds with probability $1 - k^{-\Omega(1)}$, and recovery takes $O(k)$ time. This can be achieved for arbitrarily small $\epsilon > 0$, using $O(k/\epsilon^2)$ rows. We achieve a similar result in the ℓ_1 norm.

5.1 Our techniques

Our method is related to existing sparse recovery algorithms, including Count-Sketch [9] and Count-Min [13]. In fact, our sketch matrix A is almost identical to the one used in Count-Sketch—each column of A has d random locations out of $O(kd)$ each independently set to ± 1, and the columns are independently generated. We can view such a matrix as "hashing" each coordinate to d "buckets" out of $O(kd)$. The difference is that the previous algorithms require $O(k \log k)$ measurements to achieve our error bound (and $d = O(\log k)$), while we only need $O(k)$ measurements and $d = O(1)$.

We overcome two obstacles to bring d down to $O(1)$ and still achieve the error bound with high probability[1]. First, in order to estimate the coordinates x_i, we need a more elaborate method than, say, taking the median of the buckets that i was hashed into. This is because, with constant probability, all such buckets might contain some other elements from S (be "heavy") and therefore using *any* of them as an estimator for y_i would result in too much error. Since, for super-constant values of $|S|$, it is highly likely that such an event will occur for at least one $i \in S$, it follows that this type of estimation results in large error.

We solve this issue by using our knowledge of S. We know when a bucket is "corrupted" (that is, contains more than one element of S), so we only estimate coordinates that lie in a large number of uncorrupted buckets. Once we estimate a coordinate, we subtract our estimation of its value from the buckets it is contained in. This potentially decreases the number of corrupted buckets, allowing us to estimate more coordinates. We show that, with high probability, this procedure can continue until it estimates every coordinate in S.

The other issue with the previous algorithms is that their analysis of their probability of success does not depend on k. This means that, even if the "head" did not interfere, their chance of success would be a constant (like $1 - 2^{-\Omega(d)}$) rather than high probability in k (meaning $1 - k^{-\Omega(d)}$). We show that the errors in our estimates of coordinates have low covariance, which allows us to apply Chebyshev's inequality

[1]In this paper, "high probability" means probability at least $1 - 1/k^c$ for some constant $c > 0$.

to get that the total error is concentrated around the mean with high probability.

5.2 Related work

A similar recovery algorithm (with $d = 2$) has been analyzed and applied in a streaming context in [23]. However, in that paper the authors only consider the case where the vector y is k-sparse. In that case, the termination property alone suffices, since there is no error to bound. Furthermore, because $d = 2$ they only achieve a constant probability of success. In this paper we consider general vectors y so we need to make sure the error remains bounded, and we achieve a high probability of success.

5.3 Applications

Our efficient solution to the set query problem can be combined with existing techniques to achieve sparse recovery under several models.

We say that a vector x follows a *Zipfian* or *power law* distribution with parameter α if $|x_{r(i)}| = \Theta(|x_{r(1)}| \, i^{-\alpha})$ where $r(i)$ is the location of the ith largest coefficient in x. When $\alpha > 1/2$, x is well approximated in the ℓ_2 norm by its sparse approximation. Because a wide variety of real world signals follow power law distributions ([39, 5]), this notion (related to "compressibility"[2]) is often considered to be much of the reason why sparse recovery is interesting ([4, 7]). Prior to this work, sparse recovery of power law distributions has only been solved via general sparse recovery methods: $(1 + \epsilon)\,\mathrm{Err}_k^2(x)$ error in $O(k \log(n/k))$ measurements.

However, locating the large coefficients in a power law distribution has long been easier than in a general distribution. Using $O(k \log n)$ measurements, the Count-Sketch algorithm [9] can produce a candidate set $S \subseteq \{1, \ldots, b\}$ with $|S| = O(k)$ that includes all of the top k positions in a power law distribution with high probability (if $\alpha > 1/2$). We can then apply our set query algorithm to recover an approximation x' to x_S. Because we already are using $O(k \log n)$ measurements on Count-Sketch,

[2] A signal is "compressible" when $|x_{r(i)}| = O(|x_{r(1)}| \, i^{-\alpha})$ rather than $\Theta(|x_{r(1)}| \, i^{-\alpha})$ [4]. This allows it to decay very quickly then stop decaying for a while; we require that the decay be continuous.

we use $O(k \log n)$ rather than $O(k)$ measurements in the set query algorithm to get an $\epsilon/\sqrt{\log n}$ rather than ϵ approximation. This lets us recover an x' with $O(k \log n)$ measurements with

$$\|x' - x\|_2 \leq \left(1 + \frac{\epsilon}{\sqrt{\log n}}\right) \mathrm{Err}_k^2(x).$$

This is especially interesting in the common regime where $k < n^{1-c}$ for some constant $c > 0$. Then no previous algorithms achieve better than a $(1+\epsilon)$ approximation with $O(k \log n)$ measurements, and the lower bound in [16] shows that any $O(1)$ approximation requires $\Omega(k \log n)$ measurements[3]. This means at $\Theta(k \log n)$ measurements, the best approximation changes from $\omega(1)$ to $1 + o(1)$.

Another application is that of finding *block-sparse* approximations. In this case, the coordinate set $\{1 \ldots n\}$ is partitioned into n/b blocks, each of length b. We define a (k, b)-block-sparse vector to be a vector where all non-zero elements are contained in at most k/b blocks. An example of block-sparse data is time series data from n/b locations over b time steps, where only k/b locations are "active". We can define

$$\mathrm{Err}_{(k,b)}(x) = \min_{(k,b)-\text{block-sparse } \hat{x}} \|x - \hat{x}\|_2 .$$

The block-sparse recovery problem can be now formulated analogously to Equation 1.1. Since the formulation imposes restrictions on the sparsity patterns, it is natural to expect that one can perform sparse recovery from fewer than $O(k \log(n/k))$ measurements needed in the general case. Because of that reason and its practical relevance, the problem of stable recovery of variants of block-sparse approximations has been recently a subject of extensive research (e.g., see [22, 42, 4, 8]). The state of the art algorithm has been given in [4], who gave a probabilistic construction of a single $m \times n$ matrix A, with $m = O(k + \frac{k}{b} \log n)$, and an $n \log^{O(1)} n$-time algorithm for performing the block-sparse recovery in the ℓ_1 norm (as well as other variants). If the blocks have size $\Omega(\log n)$, the algorithm uses only $O(k)$ measurements, which is a substantial improvement over the general bound. However, the approximation factor

[3]The lower bound only applies to geometric distributions, not Zipfian ones. However, our algorithm applies to more general *sub-Zipfian* distributions (defined in Section 7.1), which includes both.

C guaranteed by that algorithm was super-constant.

In this paper, we provide a distribution over matrices A, with $m = O(k + \frac{k}{b} \log n)$, which enables solving this problem with a *constant* approximation factor and in the ℓ_2 norm, with high probability. As with Zipfian distributions, first one algorithm tells us where to find the heavy hitters and then the set query algorithm estimates their values. In this case, we modify the algorithm of [2] to find *block heavy hitters*, which enables us to find the support of the $\frac{k}{b}$ "most significant blocks" using $O(\frac{k}{b} \log n)$ measurements. The essence is to perform dimensionality reduction of each block from b to $O(\log n)$ dimensions, then estimate the result with a linear hash table. For each block, most of the projections are estimated pretty well, so the median is a good estimator of the block's norm. Once the support is identified, we can recover the coefficients using the set query algorithm.

5.4 Preliminaries

5.4.1 Notation

For $n \in \mathbb{Z}^+$, we denote $\{1, \ldots, n\}$ by $[n]$. Suppose $x \in \mathbb{R}^n$. Then for $i \in [n]$, $x_i \in \mathbb{R}$ denotes the value of the i-th coordinate in x. As an exception, $e_i \in \mathbb{R}^n$ denotes the elementary unit vector with a one at position i. For $S \subseteq [n]$, x_S denotes the vector $x' \in R^n$ given by $x'_i = x_i$ if $i \in S$, and $x'_i = 0$ otherwise. We use $\text{supp}(x)$ to denote the support of x. We use upper case letters to denote sets, matrices, and random distributions. We use lower case letters for scalars and vectors.

5.4.2 Negative association

This paper would like to make a claim of the form "We have k observations each of whose error has small expectation and variance. Therefore the average error is small with high probability in k." If the errors were independent this would be immediate from Chebyshev's inequality, but our errors depend on each other. Fortunately, our errors have some tendency to behave even better than if they were independent:

the more noise that appears in one coordinate, the less remains to land in other coordinates. We use *negative dependence* to refer to this general class of behavior. The specific forms of negative dependence we use are *negative association* and *approximate negative correlation*; see Appendix A.2 for details on these notions.

Chapter 6

Set-Query Algorithm

Theorem 12. *We give a randomized sparse binary sketch matrix A and recovery algorithm $\mathscr{A}$, such that for any $x \in \mathbb{R}^n$, $S \subseteq [n]$ with $|S| = k$, $x' = \mathscr{A}(Ax+\nu, S) \in \mathbb{R}^n$ has* $\text{supp}(x') \subseteq S$ *and*

$$\|x' - x_S\|_2 \leq \epsilon(\|x - x_S\|_2 + \|\nu\|_2)$$

with probability at least $1 - 1/k^c$. Our A has $O(\frac{c}{\epsilon^2}k)$ rows and $O(c)$ non-zero entries per column, and $\mathscr{A}$ runs in $O(ck)$ time.

We can achieve $\|x' - x_S\|_1 \leq \epsilon(\|x - x_S\|_1 + \|\nu\|_1)$ under the same conditions, but with only $O(\frac{c}{\epsilon}k)$ rows.

We will first show Theorem 12 for a constant $c = 1/3$ rather than for general c. Parallel repetition gives the theorem for general c, as described in Subsection 6.7. We will also only show it with entries of A being in $\{0, 1, -1\}$. By splitting each row in two, one for the positive and one for the negative entries, we get a binary matrix with the same properties. The paper focuses on the more difficult ℓ_2 result; see Appendix B for details on the ℓ_1 result.

6.1 Intuition

We call x_S the "head" and $x - x_S$ the "tail." The head probably contains the heavy hitters, with much more mass than the tail of the distribution. We would like to estimate x_S with zero error from the head and small error from the tail with high probability.

Our algorithm is related to the standard Count-Sketch [9] and Count-Min [13] algorithms. In order to point out the differences, let us examine how they perform on this task. These algorithms show that hashing into a single $w = O(k)$ sized hash table is good in the sense that each point x_i has:

1. Zero error from the head with constant probability (namely $1 - \frac{k}{w}$).
2. A small amount of error from the tail in expectation (and hence with constant probability).

They then iterate this procedure d times and take the median, so that each estimate has small error with probability $1 - 2^{-\Omega(d)}$. With $d = O(\log k)$, we get that all k estimates in S are good with $O(k \log k)$ measurements with high probability in k. With fewer measurements, however, some x_i will probably have error from the head. If the head is much larger than the tail (such as when the tail is zero), this is a major problem. Furthermore, with $O(k)$ measurements the error from the tail would be small only in expectation, not with high probability.

We make three observations that allow us to use only $O(k)$ measurements to estimate x_S with error relative to the tail with high probability in k.

1. The total error from the tail over a support of size k is concentrated more strongly than the error at a single point: the error probability drops as $k^{-\Omega(d)}$ rather than $2^{-\Omega(d)}$.
2. The error from the head can be avoided if one knows where the head is, by modifying the recovery algorithm.
3. The error from the tail remains concentrated after modifying the recovery algorithm.

For simplicity this paper does not directly show (1), only (2) and (3). The modification to the algorithm to achieve (2) is quite natural, and described in detail in Section 6.2. Rather than estimate every coordinate in S immediately, we only estimate those coordinates which mostly do not overlap with other coordinates in S. In particular, we only estimate x_i as the median of at least $d-2$ positions that are not in the image of $S \setminus \{i\}$. Once we learn x_i, we can subtract Ax_ie_i from the observed Ax and repeat on $A(x - x_ie_i)$ and $S \setminus \{i\}$. Because we only look at positions that are in the image of only one remaining element of S, this avoids any error from the head. We show in Section 6.3 that this algorithm never gets stuck; we can always find some position that mostly doesn't overlap with the image of the rest of the remaining support.

We then show that the error from the tail has low expectation, and that it is strongly concentrated. We think of the tail as noise located in each "cell" (coordinate in the image space). We decompose the error of our result into two parts: the "point error" and the "propagation". The point error is error introduced in our estimate of some x_i based on noise in the cells that we estimate x_i from, and equals the median of the noise in those cells. The "propagation" is the error that comes from point error in estimating other coordinates in the same connected component; these errors propagate through the component as we subtract off incorrect estimates of each x_i.

Section 6.4 shows how to decompose the total error in terms of point errors and the component sizes. The two following sections bound the expectation and variance of these two quantities and show that they obey some notions of negative dependence[1]. We combine these errors in Section 6.7 to get Theorem 12 with a specific c (namely $c = 1/3$). We then use parallel repetition to achieve Theorem 12 for arbitrary c.

6.2 Algorithm

We describe the sketch matrix A and recovery procedure in Algorithm 6.2.1. Unlike Count-Sketch [9] or Count-Min [13], our A is not split into d hash tables of size

[1]See Section 5.4.2 and Appendix A.2 for discussions of negative dependence.

$O(k)$. Instead, it has a single $w = O(d^2k/\epsilon^2)$ sized hash table where each coordinate is hashed into d unique positions. We can think of A as a random d-uniform hypergraph, where the non-zero entries in each column correspond to the terminals of a hyperedge. We say that A is drawn from $\mathbb{G}^d(w, n)$ with random signs associated with each (hyperedge, terminal) pair. We do this so we will be able to apply existing theorems on random hypergraphs.

Figure 6-1 shows an example Ax for a given x, and Figure 6-2 demonstrates running the recovery procedure on this instance.

Definition of sketch matrix A**.** For a constant d, let A be a $w \times n = O(\frac{d^2}{\epsilon^2}k) \times n$ matrix where each column is chosen independently uniformly at random over all exactly d-sparse columns with entries in $\{-1, 0, 1\}$. We can think of A as the incidence matrix of a random d-uniform hypergraph with random signs.

Recovery procedure.

procedure SETQUERY(A, S, b) ▷ Recover approximation x' to x_S from $b = Ax + \nu$
  $T \leftarrow S$
  while $|T| > 0$ **do**
    Define $P(q) = \{j \mid A_{qj} \neq 0, j \in T\}$ as the set of hyperedges in T that contain q.
    Define $L_j = \{q \mid A_{qj} \neq 0, |P(q)| = 1\}$ as the set of isolated vertices in hyperedge j.
    Choose a random $j \in T$ such that $|L_j| \geq d - 1$. If this is not possible, find a random $j \in T$ such that $|L_j| \geq d - 2$. If neither is possible, abort.
    $x'_j \leftarrow \text{median}_{q \in L_j} A_{qj} b_q$
    $b \leftarrow b - x'_j A e_j$
    $T \leftarrow T \setminus \{j\}$
  end while
  return x'
end procedure

Algorithm 6.2.1: Recovering a signal given its support.

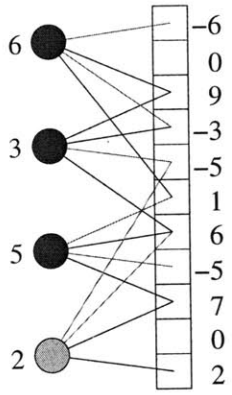

Figure 6-1: An instance of the set query problem. There are n vertices on the left, corresponding to x, and the table on the right represents Ax. Each vertex i on the left maps to d cells on the right, randomly increasing or decreasing the value in each cell by x_i. We represent addition by black lines, and subtraction by red lines. We are told the locations of the heavy hitters, which we represent by blue circles; the rest is represented by yellow circles.

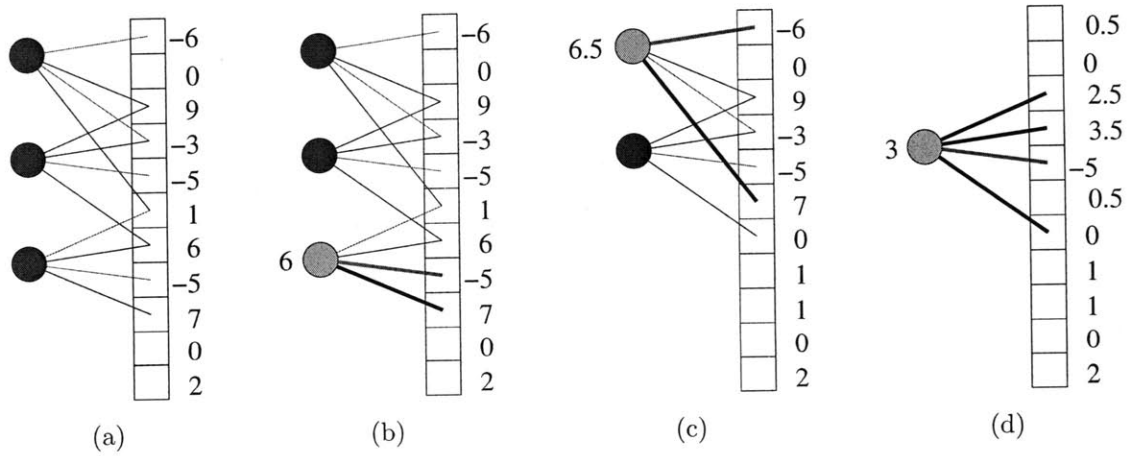

Figure 6-2: Example run of the algorithm. Part (a) shows the state as considered by the algorithm: Ax and the graph structure corresponding to the given support. In part (b), the algorithm chooses a hyperedge with at least $d-2$ isolated vertices and estimates the value as the median of those isolated vertices multiplied by the sign of the corresponding edge. In part (c), the image of the first vertex has been removed from Ax and we repeat on the smaller graph. We continue until the entire support has been estimated, as in part (d).

Lemma 13. *Algorithm 6.2.1 runs in time $O(dk)$.*

Proof. A has d entries per column. For each of the at most dk rows q in the image of S, we can store the preimages $P(q)$. We also keep track of the sets of possible next hyperedges, $J_i = \{j \mid |L_j| \geq d - i\}$ for $i \in \{1, 2\}$. We can compute these in an initial pass in $O(dk)$. Then in each iteration, we remove an element $j \in J_1$ or J_2 and update x'_j, b, and T in $O(d)$ time. We then look at the two or fewer non-isolated vertices q in hyperedge j, and remove j from the associated $P(q)$. If this makes $|P(q)| = 1$, we check whether to insert the element in $P(q)$ into the J_i. Hence the inner loop takes $O(d)$ time, for $O(dk)$ total. □

6.3 Exact recovery

The random hypergraph $\mathbb{G}^d(w, k)$ of k random d-uniform hyperedges on w vertices is well studied in [33].

We define a connected hypergraph H with r vertices on s hyperedges to be a *hypertree* if $r = s(d-1) + 1$ and to be *unicyclic* if $r = s(d-1)$. Then Theorem 4 of [33] shows that, if the graph is sufficiently sparse, $\mathbb{G}^d(w, k)$ is probably composed entirely of hypertrees and unicyclic components. The precise statement is as follows[2]:

Lemma 14 (Theorem 4 of [33]). *Let $m = w/d(d-1) - k$. Then with probability $1 - O(d^5 w^2/m^3)$, $\mathbb{G}^d(w, k)$ is composed entirely of hypertrees and unicyclic components.*

We use a simple consequence:

Corollary 15. *If $d = O(1)$ and $w \geq 2d(d-1)k$, then with probability $1 - O(1/k)$, $\mathbb{G}^d(w, k)$ is composed entirely of hypertrees and unicyclic*

We now prove some basic facts about hypertrees and unicyclic components:

Lemma 16. *Every hypertree has a hyperedge incident on at least $d - 1$ isolated vertices. Every unicyclic component either has a hyperedge incident on $d - 1$ isolated*

[2]Their statement of the theorem is slightly different. This is the last equation in their proof of the theorem.

vertices or has a hyperedge incident on $d-2$ isolated vertices, the removal of which turns the unicyclic component into a hypertree.

Proof. Let H be a connected component of s hyperedges and r vertices.

If H is a hypertree, $r = (d-1)s + 1$. Because H has only ds total (hyperedge, incident vertex) pairs, at most $2(s-1)$ of these pairs can involve vertices that appear in two or more hyperedges. Thus at least one of the s edges is incident on at most one vertex that is not isolated, so some edge has $d-1$ isolated vertices.

If H is unicyclic, $r = (d-1)s$ and so at most $2s$ of the (hyperedge, incident vertex) pairs involve non-isolated vertices. Therefore on average, each edge has $d-2$ isolated vertices. If no edge is incident on at least $d-1$ isolated vertices, every edge must be incident on exactly $d-2$ isolated vertices. In that case, each edge is incident on exactly two non-isolated vertices and each non-isolated vertex is in exactly two edges. Hence we can perform an Eulerian tour of all the edges, so removing any edge does not disconnect the graph. After removing the edge, the graph has $s' = s-1$ edges and $r' = r-d+2$ vertices; therefore $r' = (d-1)s'+1$ so the graph is a hypertree. □

These two lemmas show that the algorithm terminates without aborting:

Lemma 17. *With probability at least $1-O(1/k)$, Algorithm 6.2.1 terminates without aborting. Furthermore, in each component at most one hyperedge is chosen with only $d-2$ isolated vertices.*

6.4 Total error in terms of point error and component size

Define $C_{i,j}$ to be the event that hyperedges i and j are in the same component, and $D_i = \sum_j C_{i,j}$ to be the number of hyperedges in the same component as i. Define L_i to be the cells that are used to estimate i; so $L_i = \{q \mid A_{qj} \neq 0, |P(q)| = 1\}$ at the round of the algorithm when i is estimated. Define $Y_i = \text{median}_{q \in L_i} A_{qi}(b - Ax_S)_q$ to be the "point error" for hyperedge i, and x' to be the output of the algorithm. Then

the deviation of the output at any coordinate i is at most twice the sum of the point errors in the component containing i:

Lemma 18.

$$|(x' - x_S)_i| \leq 2 \sum_{j \in S} |Y_j| \, C_{i,j}.$$

Proof. Let $T_i = (x' - x_S)_i$, and define $R_i = \{j \mid j \neq i, \exists q \in L_i \text{ s.t. } A_{qj} \neq 0\}$ to be the set of hyperedges that overlap with the cells used to estimate i. Then from the description of the algorithm, it follows that

$$T_i = \operatorname*{median}_{q \in L_i} A_{qi}((b - Ax_S)_q - \sum_j A_{qj} T_j)$$

$$|T_i| \leq |Y_i| + \sum_{j \in R_i} |T_j| \, .$$

We can think of the R_i as a directed acyclic graph (DAG), where there is an edge from j to i if $j \in R_i$. Then if $p(i,j)$ is the number of paths from i to j,

$$|T_i| \leq \sum_j p(j,i) \, |Y_i| \, .$$

Let $r(i) = |\{j \mid i \in R_j\}|$ be the outdegree of the DAG. Because the L_i are disjoint, $r(i) \leq d - |L_i|$. From Lemma 17, $r(i) \leq 1$ for all but one hyperedge in the component, and $r(i) \leq 2$ for that one. Hence $p(i,j) \leq 2$ for any i and j, giving the result. □

We use the following corollary:

Corollary 19.

$$\|x' - x_S\|_2^2 \leq 4 \sum_{i \in S} D_i^2 Y_i^2$$

Proof.

$$\|x' - x_S\|_2^2 = \sum_{i \in S} (x' - x_S)_i^2 \leq 4 \sum_{i \in S} (\sum_{\substack{j \in S \\ C_{i,j}=1}} |Y_j|)^2 \leq 4 \sum_{i \in S} D_i \sum_{\substack{j \in S \\ C_{i,j}=1}} |Y_j|^2 = 4 \sum_{i \in S} D_i^2 Y_i^2$$

where the second inequality is the power means inequality. □

The D_j and Y_j are independent from each other, since one depends only on A over S and one only on A over $[n] \setminus S$. Therefore we can analyze them separately; the next two sections show bounds and negative dependence results for Y_j and D_j, respectively.

6.5 Bound on point error

Recall from Section 6.4 that based entirely on the set S and the columns of A corresponding to S, we can identify the positions L_i used to estimate x_i. We then defined the "point error"

$$Y_i = \underset{q \in L_i}{\text{median}} A_{qi}(b - Ax_S)_q = \underset{q \in L_i}{\text{median}} A_{qi}(A(x - x_S) + \nu)_q$$

and showed how to relate the total error to the point error. Here we would like to show that the Y_i have bounded moments and are negatively dependent. Unfortunately, it turns out that the Y_i are not negatively associated so it is unclear how to show negative dependence directly. Instead, we will define some other variables Z_i that are always larger than the corresponding Y_i. We will then show that the Z_i have bounded moments and negative association.

We use the term "NA" throughout the proof to denote negative association. For the definition of negative association and relevant properties, see Appendix A.2.

Lemma 20. *Suppose $d \geq 7$ and define $\mu = O(\frac{\epsilon^2}{k}(\|x - x_S\|_2^2 + \|\nu\|_2^2))$. There exist random variables Z_i such that the variables Y_i^2 are stochastically dominated by Z_i, the Z_i are negatively associated, $\mathrm{E}[Z_i] = \mu$, and $\mathrm{E}[Z_i^2] = O(\mu^2)$.*

Proof. The choice of the L_i depends only on the values of A over S; hence conditioned on knowing L_i we still have $A(x - x_S)$ distributed randomly over the space. Furthermore the distribution of A and the reconstruction algorithm are invariant under permutation, so we can pretend that ν is permuted randomly before being added to Ax. Define $B_{i,q}$ to be the event that $q \in \text{supp}(Ae_i)$, and define $D_{i,q} \in \{-1, 1\}$

independently at random. Then define the random variable

$$V_q = (b - Ax_S)_q = \nu_q + \sum_{i \in [n] \setminus S} x_i B_{i,q} D_{i,q}.$$

Because we want to show concentration of measure, we would like to show negative association (NA) of the $Y_i = \text{median}_{q \in L_i} A_{qi} V_q$. We know ν is a permutation distribution, so it is NA [31]. The $B_{i,q}$ for each i as a function of q are chosen from a Fermi-Dirac model, so they are NA [20]. The $B_{i,q}$ for different i are independent, so all the $B_{i,q}$ variables are NA. Unfortunately, the $D_{i,q}$ can be negative, which means the V_q are not necessarily NA. Instead we will find some NA variables that dominate the V_q. We do this by considering V_q as a distribution over D.

Let $W_q = \mathrm{E}_D[V_q^2] = \nu_q^2 + \sum_{i \in [n] \setminus S} x_i^2 B_{i,q}$. As increasing functions of NA variables, the W_q are NA. By Markov's inequality $\Pr_D[V_q^2 \geq cW_q] \leq \frac{1}{c}$, so after choosing the $B_{i,q}$ and as a distribution over D, V_q^2 is dominated by the random variable $U_q = W_q F_q$ where F_q is, independently for each q, given by the p.d.f. $f(c) = 1/c^2$ for $c \geq 1$ and $f(c) = 0$ otherwise. Because the distribution of V_q over D is independent for each q, the U_q jointly dominate the V_q^2.

The U_q are the componentwise product of the W_q with independent positive random variables, so they too are NA. Then define

$$Z_i = \operatorname*{median}_{q \in L_i} U_q.$$

As an increasing function of disjoint subsets of NA variables, the Z_i are NA. We also have that

$$Y_i^2 = (\operatorname*{median}_{q \in L_i} A_{qi} V_q)^2 \leq (\operatorname*{median}_{q \in L_i} |V_q|)^2 = \operatorname*{median}_{q \in L_i} V_q^2 \leq \operatorname*{median}_{q \in L_i} U_q = Z_i$$

so the Z_i stochastically dominate Y_i^2. We now will bound $\mathrm{E}[Z_i^2]$. Define

$$\mu = E[W_q] = \mathrm{E}[\nu_q^2] + \sum_{i \in [n] \setminus S} x_i^2 E[B_{i,q}] = \frac{d}{w} \|x - x_S\|_2^2 + \frac{1}{w} \|\nu\|_2^2 \leq \frac{\epsilon^2}{k} (\|x - x_S\|_2^2 + \|\nu\|_2^2).$$

Then we have

$$\Pr[W_q \geq c\mu] \leq \frac{1}{c}$$
$$\Pr[U_q \geq c\mu] = \int_0^\infty f(x) \Pr[W_q \geq c\mu/x] dx$$
$$\leq \int_1^c \frac{1}{x^2} \frac{x}{c} dx + \int_c^\infty \frac{1}{x^2} dx = \frac{1 + \ln c}{c}$$

Because the U_q are NA, they satisfy marginal probability bounds [20]:

$$\Pr[U_q \geq t_q, q \in [w]] \leq \prod_{i \in [n]} \Pr[U_q \geq t_q]$$

for any t_q. Therefore

$$\Pr[Z_i \geq c\mu] \leq \sum_{\substack{T \subset L_i \\ |T| = |L_i|/2}} \prod_{q \in T} Pr[U_q \geq c\mu] \leq 2^{|L_i|} \left(\frac{1 + \ln c}{c}\right)^{|L_i|/2} \leq \left(4\frac{1 + \ln c}{c}\right)^{d/2-1} \tag{6.1}$$

If $d \geq 7$, this makes $\mathrm{E}[Z_i] = O(\mu)$ and $\mathrm{E}[Z_i^2] = O(\mu^2)$. □

6.6 Bound on component size

Lemma 21. *Let D_i be the number of hyperedges in the same component as hyperedge i. Then for any $i \neq j$,*

$$Cov(D_i^2, D_j^2) = \mathrm{E}[D_i^2 D_j^2] - \mathrm{E}[D_i^2]^2 \leq O(\frac{\log^6 k}{\sqrt{k}}).$$

Furthermore, $\mathrm{E}[D_i^2] = O(1)$ and $\mathrm{E}[D_i^4] = O(1)$.

Proof. The intuition is that if one component gets larger, other components tend to get smaller. There is a small probability that i and j are connected, in which case D_i and D_j are positively correlated, but otherwise D_i and D_j should be negatively correlated. However analyzing this directly is rather difficult, because as one com-

ponent gets larger, the remaining components have a lower average size but higher variance. Our analysis instead takes a detour through the hypergraph where each hyperedge is picked independently with a probability that gives the same expected number of hyperedges. This distribution is easier to analyze, and only differs in a relatively small $\tilde{O}(\sqrt{k})$ hyperedges from our actual distribution. This allows us to move between the regimes with only a loss of $\tilde{O}(\frac{1}{\sqrt{k}})$, giving our result.

Suppose instead of choosing our hypergraph from $\mathbb{G}^d(w,k)$ we chose it from $\mathbb{G}^d(w, \frac{k}{\binom{w}{d}})$; that is, each hyperedge appeared independently with the appropriate probability to get k hyperedges in expectation. This model is somewhat simpler, and yields a very similar hypergraph $\overline{G}$. One can then modify $\overline{G}$ by adding or removing an appropriate number of random hyperedges I to get exactly k hyperedges, forming a uniform $G \in \mathbb{G}^d(w,k)$. By the Chernoff bound, $|I| \leq O(\sqrt{k}\log k)$ with probability $1 - \frac{1}{k^{\Omega(1)}}$.

Let $\overline{D}_i$ be the size of the component containing i in $\overline{G}$, and $H_i = D_i^2 - \overline{D}_i^2$. Let E denote the event that any of the D_i or $\overline{D}_i$ is more than $C\log k$, or that more than $C\sqrt{k}\log k$ hyperedges lie in I, for some constant C. Then E happens with probability less than $\frac{1}{k^5}$ for some C, so it has negligible influence on $\mathrm{E}[D_i^2 D_j^2]$. Hence the rest of this proof will assume E does not happen.

Therefore $H_i = 0$ if none of the $O(\sqrt{k}\log k)$ random hyperedges in I touch the $O(\log k)$ hyperedges in the components containing i in $\overline{G}$, so $H_i = 0$ with probability at least $1 - O(\frac{\log^2 k}{\sqrt{k}})$. Even if $H_i \neq 0$, we still have $|H_i| \leq (D_i^2 + D_j^2) \leq O(\log^2 k)$.

Also, we show that the $\overline{D}_i^2$ are negatively correlated, when conditioned on being in separate components. Let $\overline{D}(n,p)$ denote the distribution of the component size of a random hyperedge on $\mathbb{G}^d(n,p)$, where p is the probability an hyperedge appears. Then $\overline{D}(n,p)$ dominates $\overline{D}(n',p)$ whenever $n > n'$ — the latter hypergraph is contained within the former. If $\overline{C}_{i,j}$ is the event that i and j are connected in $\overline{G}$, this means

$$\mathrm{E}[\overline{D}_i^2 \mid \overline{D}_j = t, \overline{C}_{i,j} = 0]$$

is a decreasing function in t, so we have negative correlation:

$$\mathrm{E}[\overline{D}_i^2\overline{D}_j^2 \mid \overline{C}_{i,j}=0] \le \mathrm{E}[\overline{D}_i^2 \mid \overline{C}_{i,j}=0]\,\mathrm{E}[\overline{D}_j^2 \mid \overline{C}_{i,j}=0] \le \mathrm{E}[\overline{D}_i^2]\,\mathrm{E}[\overline{D}_j^2].$$

Furthermore for $i \neq j$, $\Pr[\overline{C}_{i,j}=1] = \mathrm{E}[\overline{C}_{i,j}] = \frac{1}{k-1}\sum_{l\neq i}\mathrm{E}[\overline{C}_{i,l}] = \frac{\mathrm{E}[\overline{D}_i]-1}{k-1} = O(1/k)$. Hence

$$\begin{aligned}\mathrm{E}[\overline{D}_i^2\overline{D}_j^2] &= \mathrm{E}[\overline{D}_i^2\overline{D}_j^2 \mid \overline{C}_{i,j}=0]\Pr[\overline{C}_{i,j}=0] + \mathrm{E}[\overline{D}_i^2\overline{D}_j^2 \mid \overline{C}_{i,j}=1]\Pr[\overline{C}_{i,j}=1] \\ &\le \mathrm{E}[\overline{D}_i^2]\,\mathrm{E}[\overline{D}_j^2] + O(\frac{\log^4 k}{k}).\end{aligned}$$

Therefore

$$\begin{aligned}\mathrm{E}[D_i^2D_j^2] &= \mathrm{E}[(\overline{D}_i^2+H_i)(\overline{D}_j^2+H_j)] \\ &= \mathrm{E}[\overline{D}_i^2\overline{D}_j^2] + 2\,\mathrm{E}[H_i\overline{D}_j^2] + \mathrm{E}[H_iH_j] \\ &\le \mathrm{E}[\overline{D}_i^2]\,\mathrm{E}[\overline{D}_j^2] + O(2\frac{\log^2 k}{\sqrt{k}}\log^4 k + \frac{\log^2 k}{\sqrt{k}}\log^2 k) \\ &= \mathrm{E}[D_i^2 - H_i]^2 + O(\frac{\log^6 k}{\sqrt{k}}) \\ &= \mathrm{E}[D_i^2]^2 - 2\,\mathrm{E}[H_i]\,\mathrm{E}[D_i^2] + \mathrm{E}[H_i]^2 + O(\frac{\log^6 k}{\sqrt{k}}) \\ &\le \mathrm{E}[D_i^2]^2 + O(\frac{\log^6 k}{\sqrt{k}})\end{aligned}$$

Now to bound $\mathrm{E}[D_i^4]$ in expectation. Because our hypergraph is exceedingly sparse, the size of a component can be bounded by a branching process that dies out with constant probability at each step. Using this method, Equations 71 and 72 of [12] state that $\Pr[\overline{D} \ge k] \le e^{-\Omega(k)}$. Hence $\mathrm{E}[\overline{D}_i^2] = O(1)$ and $\mathrm{E}[\overline{D}_i^4] = O(1)$. Because H_i is 0 with high probability and $O(\log^2 k)$ otherwise, this immediately gives $\mathrm{E}[D_i^2] = O(1)$ and $\mathrm{E}[D_i^4] = O(1)$. □

6.7 Wrapping it up

Recall from Corollary 19 that our total error

$$\|x' - x_S\|_2^2 \leq 4 \sum_i Y_i^2 D_i^2 \leq 4 \sum_i Z_i D_i^2.$$

The previous sections show that Z_i and D_i^2 each have small expectation and covariance. This allows us to apply Chebyshev's inequality to concentrate $4 \sum_i Z_i D_i^2$ about its expectation, bounding $\|x' - x_S\|_2$ with high probability.

Lemma 22. *We can recover x' from $Ax + \nu$ and S with*

$$\|x' - x_S\|_2 \leq \epsilon(\|x - x_S\|_2 + \|\nu\|_2)$$

with probability at least $1 - \frac{1}{c^2 k^{1/3}}$ in $O(k)$ recovery time. Our A has $O(\frac{c}{\epsilon^2} k)$ rows and sparsity $O(1)$ per column.

Proof. Recall from Corollary 19 that our total error

$$\|x' - x_S\|_2^2 \leq 4 \sum_i Y_i^2 D_i^2 \leq 4 \sum_i Z_i D_i^2.$$

Then by Lemma 20 and Lemma 21,

$$\mathrm{E}[4 \sum_i Z_i D_i^2] = 4 \sum_i \mathrm{E}[Z_i] \mathrm{E}[D_i^2] = k\mu$$

where $\mu = O(\frac{\epsilon^2}{k}(\|x - x_S\|_2^2 + \|\nu\|_2^2))$. Furthermore,

$$\begin{aligned}
\mathrm{E}[(\sum_i Z_i D_i^2)^2] &= \sum_i \mathrm{E}[Z_i^2 D_i^4] + \sum_{i \neq j} \mathrm{E}[Z_i Z_j D_i^2 D_j^2] \\
&= \sum_i \mathrm{E}[Z_i^2]\,\mathrm{E}[D_i^4] + \sum_{i \neq j} \mathrm{E}[Z_i Z_j]\,\mathrm{E}[D_i^2 D_j^2] \\
&\leq \sum_i O(\mu^2) + \sum_{i \neq j} \mathrm{E}[Z_i]\,\mathrm{E}[Z_j](\mathrm{E}[D_i^2]^2 + O(\frac{\log^6 k}{\sqrt{k}})) \\
&= O(\mu^2 k\sqrt{k}\log^6 k) + k(k-1)\,\mathrm{E}[Z_i D_i^2]^2 \\
\mathrm{Var}(\sum_i Z_i D_i^2) &= \mathrm{E}[(\sum_i Z_i D_i^2)^2] - k^2\,\mathrm{E}[Z_i D_i^2]^2 \\
&\leq O(\mu^2 k\sqrt{k}\log^6 k)
\end{aligned}$$

By Chebyshev's inequality, this means

$$\begin{aligned}
\Pr[4\sum_i Z_i D_i^2 \geq (1+c)\mu k] &\leq O(\frac{\log^6 k}{c^2\sqrt{k}}) \\
\Pr[\|x' - x_S\|_2^2 \geq (1+c)C\epsilon^2(\|x - x_S\|_2^2 + \|\nu\|_2^2)] &\leq O(\frac{1}{c^2 k^{1/3}})
\end{aligned}$$

for some constant C. Rescaling ϵ down by $\sqrt{C(1+c)}$, we can get

$$\|x' - x_S\|_2 \leq \epsilon(\|x - x_S\|_2 + \|\nu\|_2)$$

with probability at least $1 - \frac{1}{c^2 k^{1/3}}$:

□

Now we shall go from $k^{-1/3}$ probability of error to k^{-c} error for arbitrary c, with $O(c)$ multiplicative cost in time and space. We simply perform Lemma 22 $O(c)$ times in parallel, and output the pointwise median of the results. By a standard parallel repetition argument, this gives our main result:

Theorem 12. *We can recover x' from $Ax + \nu$ and S with*

$$\|x' - x_S\|_2 \leq \epsilon(\|x - x_S\|_2 + \|\nu\|_2)$$

with probability at least $1 - \frac{1}{k^c}$ *in* $O(ck)$ *recovery time. Our* A *has* $O(\frac{c}{\epsilon^2}k)$ *rows and sparsity* $O(c)$ *per column.*

Proof. Suppose our algorithm (as in Lemma 22) gives an x' such that $\|x' - x_S\|_2 \leq \mu$ with probability at least $1 - p$, and that we run this algorithm m times independently in parallel to get output vectors $x^1, \ldots, x^m$. We output y given by $y_i = \text{median}_{j \in [m]}(x^j)_i$, and claim that with high probability $\|y - x_S\|_2 \leq \mu\sqrt{3}$.

Let $J = \{j \in [m] \mid \|x^j - x_S\|_2 \leq \mu\}$. Each $j \in [m]$ lies in J with probability at least $1 - p$, so the chance that $|J| \leq 3m/4$ is less than $\binom{m}{m/4} p^{m/4} \leq (4ep)^{m/4}$. Suppose that $|J| \geq 3m/4$. Then for all $i \in S$, $|\{j \in J \mid (x^j)_i \leq y_i\}| \geq |J| - \frac{m}{2} \geq |J|/3$ and similarly $|\{j \in J \mid (x^j)_i \geq y_i\}| \geq |J|/3$. Hence for all $i \in S$, $|y_i - x_i|$ is smaller than at least $|J|/3$ of the $|(x^j)_i - x_i|$ for $j \in J$. Hence

$$|J|\,\mu^2 \geq \sum_{i \in S} \sum_{j \in J} ((x^j)_i - x_i)^2 \geq \sum_{i \in S} \frac{|J|}{3} (y_i - x_i)^2 = \frac{|J|}{3} \|y - x\|_2^2$$

or

$$\|y - x\|_2 \leq \sqrt{3}\mu$$

with probability at least $1 - (4ep)^{m/4}$.

Using Lemma 22 to get $p = \frac{1}{16k^{1/3}}$ and $\mu = \epsilon(\|x - x_S\|_2 + \|\nu\|_2)$, with $m = 12c$ repetitions we get our result. □

Chapter 7

Applications of Set Query Algorithm

We give two applications where the set query algorithm is a useful primitive.

7.1 Heavy hitters of sub-Zipfian distributions

For a vector x, let r_i be the index of the ith largest element, so $|x_{r_i}|$ is non-increasing in i. We say that x is *Zipfian with parameter* α if $|x_{r_i}| = \Theta(|x_{r_0}|\, i^{-\alpha})$. We say that x is *sub-Zipfian with parameter* α if $|x_{r_i}| = \Theta(|x_{r_0}|\, i^{-\alpha} f(i))$ for some non-increasing function f.

Zipfian distributions are common in real-world data sets, and finding heavy hitters is one of the most important problems in data streams. Therefore this is a very natural problem to try to improve; indeed, the original paper on Count-Sketch discussed it [9]. They show something complementary to our work, that one can find the support efficiently:

Lemma 23 (Section 4.1 of [9]). *If x is sub-Zipfian with parameter $\alpha > 1/2$, one can recover a candidate support set S with $|S| = O(k)$ from Ax such that $\{r_1, \ldots, r_k\} \subseteq S$. A has $O(k \log n)$ rows and recovery succeeds with high probability in n.*

Proof sketch. Let $S_k = \{r_1, \ldots, r_k\}$. With $O(\frac{1}{\epsilon^2} k \log n)$ measurements, Count-Sketch

identifies each x_i to within $\frac{\epsilon}{k}\|x - x_{S_k}\|_2$ with high probability. If $\alpha > 1/2$, this is less than $|x_{r_k}|/3$ for appropriate ϵ. But $|x_{r_{9k}}| \leq |x_{r_k}|/3$. Hence only the largest $9k$ elements of x could be estimated as larger than anything in x_{S_k}, so the locations of the largest $9k$ estimated values must contain S_k. □

They observe in [9] that a two-pass algorithm could identify the heavy hitters exactly. However, for one-pass algorithms, nothing better has been known for Zipfian distributions than for arbitrary distributions; in fact, the lower bound [16] on linear sparse recovery uses a geometric (and hence sub-Zipfian) distribution. The set query algorithm lets us improve from a $1+\epsilon$ approximation to a $1+o(1)$ approximation:

Theorem 24. *Suppose x comes from a sub-Zipfian distribution with parameter $\alpha > 1/2$. Then we can recover x' from Ax with*

$$\|x' - x\|_2 \leq \frac{\epsilon}{\sqrt{\log n}} \mathrm{Err}_k^2(x)$$

with $O(\frac{c}{\epsilon^2} k \log n)$ rows and $O(n \log n)$ recovery time, with probability at least $1 - \frac{1}{k^c}$.

Proof. By Lemma 23 we can identify a set S of size $O(k)$ that contains all the heavy hitters. We then run the set query algorithm of Theorem 12 with $\frac{\epsilon}{\sqrt{\log n}}$ substituted for ϵ. □

7.2 Block-sparse vectors

In this section we consider the problem of finding *block-sparse* approximations. In this case, the coordinate set $\{1 \ldots n\}$ is partitioned into n/b blocks, each of length b. We define a (k, b)-block-sparse vector to be a vector where all non-zero elements are contained in at most k/b blocks. That is, we partition $\{1, \ldots, n\}$ into $T_i = \{(i-1)b+1, \ldots, ib\}$. A vector x is (k, b)-block-sparse if there exist $S_1, \ldots, S_{k/b} \in \{T_1, \ldots, T_{n/b}\}$ with $\mathrm{supp}(x) \subseteq \bigcup S_i$. Define

$$\mathrm{Err}_{(k,b)}(x) = \min_{(k,b)-\text{block-sparse } \hat{x}} \|x - \hat{x}\|_2 .$$

Finding the support of block-sparse vectors is closely related to finding block heavy hitters, which is studied for the ℓ_1 norm in [2]. We show how to do it in Appendix C.

Lemma 25. *For any b and k, there exists a family of matrices A with $O(\frac{k}{\epsilon^5 b}\log n)$ rows and column sparsity $O(\frac{1}{\epsilon^2}\log n)$ such that we can recover a support S from Ax in $O(\frac{n}{\epsilon^2 b}\log n)$ time with*

$$\|x - x_S\|_2 \leq (1+\epsilon)\,\mathrm{Err}_{(k,b)}(x)$$

with probability at least $1 - n^{-\Omega(1)}$.

Once we know a good support S, we can run Algorithm 6.2.1 to estimate x_S.

Theorem 26. *For any b and k, there exists a family of binary matrices A with $O(\frac{1}{\epsilon^2}k + \frac{k}{\epsilon^5 b}\log n)$ rows such that we can recover a (k,b)-block-sparse x' in $O(k + \frac{n}{\epsilon^2 b}\log n)$ time with*

$$\|x' - x\|_2 \leq (1+\epsilon)\,\mathrm{Err}_{(k,b)}(x)$$

with probability at least $1 - \frac{1}{k^{\Omega(1)}}$.

If the block size b is at least $\log n$ and ϵ is constant, this gives an optimal bound of $O(k)$ rows.

Appendix A

Standard Mathematical Lemmas

A.1 Proof of Lemma 1

Proof. We will construct a codebook T of block length k, alphabet q, and minimum Hamming distance ϵk. Replacing each character i with the q-long standard basis vector e_i will create a binary qk-dimensional codebook S with minimum Hamming distance $2\epsilon k$ of the same size as T, where each element of S has exactly k ones.

The Gilbert-Varshamov bound, based on volumes of Hamming balls, states that a codebook of size L exists for some

$$L \geq \frac{q^k}{\sum_{i=0}^{\epsilon k - 1} \binom{k}{i}(q-1)^i}.$$

Using the claim (analogous to [43], p. 21, proven below) that for $\epsilon < 1 - 1/q$

$$\sum_{i=0}^{\epsilon k} \binom{k}{i}(q-1)^i < q^{H_q(\epsilon)k},$$

we have that $\log L > (1 - H_q(\epsilon))k \log q$, as desired. □

Claim 27. *For* $0 < \epsilon < 1 - 1/q$,

$$\sum_{i=0}^{\epsilon k} \binom{k}{i}(q-1)^i < q^{H_q(\epsilon)k}.$$

Proof. Note that

$$q^{-H_q(\epsilon)} = \left(\frac{\epsilon}{(q-1)(1-\epsilon)}\right)^{\epsilon}(1-\epsilon) < (1-\epsilon).$$

Then

$$\begin{aligned}
1 &= (\epsilon + (1-\epsilon))^k \\
&> \sum_{i=0}^{\epsilon k} \binom{k}{i} \epsilon^i (1-\epsilon)^{k-i} \\
&= \sum_{i=0}^{\epsilon k} \binom{k}{i} (q-1)^i \left(\frac{\epsilon}{(q-1)(1-\epsilon)}\right)^i (1-\epsilon)^k \\
&> \sum_{i=0}^{\epsilon k} \binom{k}{i} (q-1)^i \left(\frac{\epsilon}{(q-1)(1-\epsilon)}\right)^{\epsilon k} (1-\epsilon)^k \\
&= q^{-H_q(\epsilon)k} \sum_{i=0}^{\epsilon k} \binom{k}{i} (q-1)^i
\end{aligned}$$

□

A.2 Negative dependence

Negative dependence is a fairly common property in balls-and-bins types of problems, and can often cleanly be analyzed using the framework of *negative association* ([20, 19, 31]).

Definition 1 (Negative Association). *Let $(X_1, \ldots, X_n)$ be a vector of random variables. Then $(X_1, \ldots, X_n)$ are* negatively associated *if for every two disjoint index sets, $I, J \subseteq [n]$,*

$$\mathrm{E}[f(X_i, i \in I)g(X_j, j \in J)] \leq \mathrm{E}[f(X_i, i \in I)]E[g(X_j, j \in J)]$$

for all functions $f\colon \mathbb{R}^{|I|} \to \mathbb{R}$ and $g\colon \mathbb{R}^{|J|} \to \mathbb{R}$ that are both non-decreasing or both non-increasing.

If random variables are negatively associated then one can apply most standard concentration of measure arguments, such as Chebyshev's inequality and the Chernoff bound. This means it is a fairly strong property, which makes it hard to prove directly. What makes it so useful is that it remains true under two composition rules:

Lemma 28 ([20], Proposition 7).

1. *If* $(X_1, \ldots, X_n)$ *and* $(Y_1, \ldots, Y_m)$ *are each negatively associated and mutually independent, then* $(X_1, \ldots, X_n, Y_1, \ldots, Y_m)$ *is negatively associated.*

2. *Suppose* $(X_1, \ldots, X_n)$ *is negatively associated. Let* $I_1, \ldots, I_k \subseteq [n]$ *be disjoint index sets, for some positive integer* k. *For* $j \in [k]$, *let* $h_j \colon \mathbb{R}^{|I_j|} \to \mathbb{R}$ *be functions that are all non-decreasing or all non-increasing, and define* $Y_j = h_j(X_i, i \in I_j)$. *Then* $(Y_1, \ldots, Y_k)$ *is also negatively associated.*

In terms of our set query algorithm, Lemma 28 allows us to relatively easily show that one component of our error (the point error) is negatively associated without performing any computation. Unfortunately, the other component of our error (the component size) is not easily built up by repeated applications of Lemma 28[1]. Therefore we show something much weaker for this error, namely *approximate negative correlation*:

$$\mathrm{E}[X_i X_j] - \mathrm{E}[X_i] E[X_j] \leq \frac{1}{k^{\Omega(1)}} \mathrm{E}[X_i] \, \mathrm{E}[X_j]$$

for all $i \neq j$. This is still strong enough to use Chebyshev's inequality.

[1]This manuscript considers the component size of each hyperedge, which clearly is not negatively associated: if one hyperedge is in a component of size k than so is every other hyperedge. But one can consider variants that just consider the distribution of component sizes, which seems plausibly negatively associated. However, this is hard to prove.

Appendix B

Bounding the Set Query Algorithm in the ℓ_1 Norm

This section works through all the changes to prove the set query algorithm works in the ℓ_1 norm with $w = O(\frac{1}{\epsilon}k)$ measurements.

We use Lemma 18 to get an ℓ_1 analog of Corollary 19:

$$\|x' - x_S\|_1 = \sum_{i \in S} |(x' - x_S)_i| \leq \sum_{i \in S} 2 \sum_{j \in S} C_{i,j} |Y_j| = 2 \sum_{i \in S} D_i |Y_i| . \tag{B.1}$$

Then we bound the expectation, variance, and covariance of D_i and $|Y_i|$. The bound on D_i works the same as in Section 6.6: $\mathrm{E}[D_i] = O(1)$, $\mathrm{E}[D_i^2] = O(1)$, $\mathrm{E}[D_i D_j] - \mathrm{E}[D_i]^2 \leq O(\log^4 k/\sqrt{k})$.

The bound on $|Y_i|$ is slightly different. We define

$$U'_q = |\nu_q| + \sum_{i \in [n] \setminus S} |x_i| B_{i,q}$$

and observe that $U'_q \geq |V_q|$, and U'_q is NA. Hence

$$Z'_i = \operatorname*{median}_{q \in L_i} U'_q$$

is NA, and $|Y_i| \leq Z'_i$. Define

$$\mu = \mathrm{E}[U'_q] = \frac{d}{w} \|x - x_S\|_1 + \frac{1}{w} \|\nu\|_1 \leq \frac{\epsilon}{k}(\|x - x_S\|_1 + \|\nu\|_1)$$

then

$$\Pr[Z'_i \geq c\mu] \leq 2^{|L_i|}(\frac{1}{c})^{|L_i|/2} \leq \left(\frac{4}{c}\right)^{d-2}$$

so $\mathrm{E}[Z'_i] = O(\mu)$ and $\mathrm{E}[Z'^2_i] = O(\mu^2)$.

Now we will show the analog of Section 6.7. We know

$$\|x' - x_S\|_2 \leq 2 \sum_i D_i Z'_i$$

and

$$\mathrm{E}[2 \sum_i D_i Z'_i] = 2 \sum_i \mathrm{E}[D_i] \mathrm{E}[Z'_i] = k\mu'$$

for some $\mu' = O(\frac{\epsilon}{k}(\|x - x_S\|_1 + \|\nu\|_1))$. Then

$$\begin{aligned}
\mathrm{E}[(\sum D_i Z'_i)^2] &= \sum_i \mathrm{E}[D_i^2] \mathrm{E}[Z'^2_i] + \sum_{i \neq j} \mathrm{E}[D_i D_j] \mathrm{E}[Z'_i Z'_j] \\
&\leq \sum_i O(\mu'^2) + \sum_{i \neq j} (\mathrm{E}[D_i]^2 + O(\log^4 k / \sqrt{k})) \mathrm{E}[Z'_i]^2 \\
&= O(\mu'^2 k \sqrt{k} \log^4 k) + k(k-1) \mathrm{E}[D_i Z'_i]^2 \\
\mathrm{Var}(2 \sum_i Z'_i D_i) &\leq O(\mu'^2 k \sqrt{k} \log^4 k).
\end{aligned}$$

By Chebyshev's inequality, we get

$$\Pr[\|x' - x_S\|_1 \geq (1 + \alpha) k\mu'] \leq O(\frac{\log^4 k}{\alpha^2 \sqrt{k}})$$

and the main theorem (for constant $c = 1/3$) follows. The parallel repetition method of Section 6.7 works the same as in the ℓ_2 case to support arbitrary c.

Appendix C

Locating Block Heavy Hitters

Lemma 25. *For any b and k, there exists a family of matrices A with $O(\frac{k}{\epsilon^5 b}\log n)$ rows and column sparsity $O(\frac{1}{\epsilon^2}\log n)$ such that we can recover a support S from Ax in $O(\frac{n}{\epsilon^2 b}\log n)$ time with*

$$\|x - x_S\|_2 \leq (1+\epsilon)\,\mathrm{Err}_{(k,b)}(x)$$

with probability at least $1 - n^{-\Omega(1)}$.

Proof. This proof follows the method of [2], but applies to the ℓ_2 norm and is in the (slightly stronger) sparse recovery framework rather than the heavy hitters framework. The idea is to perform dimensionality reduction, then use an argument similar to those for Count-Sketch (first in [14], but we follow more closely the description in [27]).

Define $s = k/b$ and $t = n/b$, and decompose $[n]$ into equal sized blocks $T_1, \ldots, T_t$. Let $x_{(T_i)} \in \mathbb{R}^b$ denote the restriction of x_{T_i} to the coordinates T_i. Let $U \subseteq [t]$ have $|U| = s$ and contain the s largest blocks in x, so $\mathrm{Err}_{(k,b)}(x) = \left\|\sum_{i\notin U} x_{T_i}\right\|_2$.

Choose an i.i.d. standard Gaussian matrix $\rho \in \mathbb{R}^{m\times b}$ for $m = O(\frac{1}{\epsilon^2}\log n)$. Define $y_{q,i} = (\rho x_{(T_q)})_i$, so as a distribution over ρ, $y_{q,i}$ is a Gaussian with variance $\left\|x_{(T_q)}\right\|_2^2$.

Let $h_1, \ldots, h_m \colon [t] \to [l]$ be pairwise independent hash functions for some $l = O(\frac{1}{\epsilon^3}s)$, and $g_1, \ldots, g_m \colon [t] \to \{-1, 1\}$ also be pairwise independent. Then we make m hash tables $H^{(1)}, \ldots, H^{(m)}$ of size l each, and say that the value of the jth cell in

the ith hash table $H^{(i)}$ is given by

$$H_j^{(i)} = \sum_{q:h_i(q)=j} g_i(q) y_{q,i}$$

Then the $H_j^{(i)}$ form a linear sketch of $ml = O(\frac{k}{\epsilon^5 b} \log n)$ cells. We use this sketch to estimate the mass of each block, and output the blocks that we estimate to have the highest mass. Our estimator for $\|x_{T_i}\|_2$ is

$$z_i' = \alpha \operatorname*{median}_{j \in [m]} \left| H_{h_j(i)}^{(j)} \right|$$

for some constant scaling factor $\alpha \approx 1.48$. Since we only care which blocks have the largest magnitude, we don't actually need to use α.

We first claim that for each i and j with probability $1 - O(\epsilon)$, $(H_{h_j(i)}^{(j)} - y_{i,j})^2 \leq O(\frac{\epsilon^2}{s}(\mathrm{Err}_{(k,b)}(x))^2)$. To prove it, note that the probability any $q \in U$ with $q \neq i$ having $h_j(q) = h_j(i)$ is at most $\frac{s}{l} \leq \epsilon^3$. If such a collision with a heavy hitter does not happen, then

$$\begin{aligned}
\mathrm{E}[(H_{h_j(i)}^{(j)} - y_{i,j})^2] &= \mathrm{E}[\sum_{p \neq i, h_j(p) = h_j(i)} y_{p,j}^2] \\
&\leq \sum_{p \notin U} \frac{1}{l} \mathrm{E}[y_{p,j}^2] \\
&= \frac{1}{l} \sum_{p \notin U} \left\| x_{T_p} \right\|_2^2 \\
&= \frac{1}{l} (\mathrm{Err}_{(k,b)}(x))^2
\end{aligned}$$

By Markov's inequality and the union bound, we have

$$\Pr[(H_{h_j(i)}^{(j)} - y_{i,j})^2 \geq \frac{\epsilon^2}{s} (\mathrm{Err}_{(k,b)}(x))^2] \leq \epsilon + \epsilon^3 = O(\epsilon)$$

Let $B_{i,j}$ be the event that $(H_{h_j(i)}^{(j)} - y_{i,j})^2 > O(\frac{\epsilon^2}{s}(\mathrm{Err}_{(k,b)}(x))^2)$, so $\Pr[B_{i,j}] = O(\epsilon)$. This is independent for each j, so by the Chernoff bound $\sum_{j=1}^{m} B_{i,j} \leq O(\epsilon m)$ with high probability in n.

Now, $|y_{i,j}|$ is distributed according to the positive half of a Gaussian, so there is some constant $\alpha \approx 1.48$ such that $\alpha\,|y_{i,j}|$ is an unbiased estimator for $\|x_{T_i}\|_2$. For any $C \geq 1$ and some $\delta = O(C\epsilon)$, we expect less than $\frac{1-C\epsilon}{2}m$ of the $\alpha\,|y_{i,j}|$ to be below $(1-\delta)\,\|x_{T_i}\|_2$, less than $\frac{1-C\epsilon}{2}m$ to be above $(1+\delta)\,\|x_{T_i}\|_2$, and more than $C\epsilon m$ to be in between. Because $m \geq \Omega(\frac{1}{\epsilon^2}\log n)$, the Chernoff bound shows that with high probability the actual number of $\alpha\,|y_{i,j}|$ in each interval is within $\frac{\epsilon}{2}m = O(\frac{1}{\epsilon}\log n)$ of its expectation. Hence

$$\left|\|x_{T_i}\|_2 - \alpha \operatorname*{median}_{j\in[m]} |y_{i,j}|\right| \leq \delta\,\|x_{T_i}\|_2 = O(C\epsilon)\,\|x_{T_i}\|_2\,.$$

even if $\frac{(C-1)\epsilon}{2}m$ of the $y_{i,j}$ were adversarially modified. We can think of the events $B_{i,j}$ as being such adversarial modifications. We find that

$$|\|x_{T_i}\|_2 - z_i| = \left|\|x_{T_i}\|_2 - \alpha \operatorname*{median}_{j\in[m]} \left|H^{(j)}_{h_j(i)}\right|\right| \leq O(\epsilon)\,\|x_{T_i}\|_2 + O(\frac{\epsilon}{\sqrt{s}}\,\mathrm{Err}_{(k,b)}(x)).$$

$$(\|x_{T_i}\|_2 - z_i)^2 \leq O(\epsilon^2\,\|x_{T_i}\|_2^2 + \frac{\epsilon^2}{s}(\mathrm{Err}_{(k,b)}(x))^2)$$

Define $w_i = \|x_{T_i}\|_2$, $\mu = \mathrm{Err}_{(k,b)}(x)$, and $\hat{U} \subseteq [t]$ to contain the s largest coordinates in z. Since z is computed from the sketch, the recovery algorithm can compute $\hat{U}$. The output of our algorithm will be the blocks corresponding to $\hat{U}$.

We know $\mu^2 = \sum_{i\notin U} w_i^2 = \left\|w_{[t]\setminus U}\right\|_2^2$ and $|w_i - z_i| \leq O(\epsilon w_i + \frac{\epsilon}{\sqrt{s}}\mu)$ for all i. We will show that

$$\left\|w_{[t]\setminus\hat{U}}\right\|_2^2 \leq (1+O(\epsilon))\mu^2.$$

This is analogous to the proof of Count-Sketch. Note that

$$\left\|w_{[t]\setminus\hat{U}}\right\|_2^2 = \left\|w_{U\setminus\hat{U}}\right\|_2^2 + \left\|w_{[t]\setminus(U\cup\hat{U})}\right\|_2^2$$

For any $i \in U \setminus \hat{U}$ and $j \in \hat{U} \setminus U$, we have $z_j > z_i$, so

$$w_i - w_j \leq O(\frac{\epsilon}{\sqrt{s}}\mu + \epsilon w_i)$$

Let $a = \max_{i \in U \setminus \hat{U}} w_i$ and $b = \min_{j \in \hat{U} \setminus U} w_j$. Then $a \leq b + O(\frac{\epsilon}{\sqrt{s}}\mu + \epsilon a)$, and dividing by $(1 - O(\epsilon))$ we get $a \leq b(1 + O(\epsilon)) + O(\frac{\epsilon}{\sqrt{s}}\mu)$. Furthermore $\left\|w_{\hat{U} \setminus U}\right\|_2^2 \geq b^2 \left|\hat{U} \setminus U\right|$, so

$$\begin{aligned}
\left\|w_{U \setminus \hat{U}}\right\|_2^2 &\leq \left(\left\|w_{\hat{U} \setminus U}\right\|_2 \frac{1 + O(\epsilon)}{\sqrt{\left|\hat{U} \setminus U\right|}} + O(\frac{\epsilon}{\sqrt{s}}\mu) \right)^2 \left|\hat{U} \setminus U\right| \\
&\leq \left(\left\|w_{\hat{U} \setminus U}\right\|_2 (1 + O(\epsilon)) + O(\epsilon\mu) \right)^2 \\
&= \left\|w_{\hat{U} \setminus U}\right\|_2^2 (1 + O(\epsilon)) + (2 + O(\epsilon)) \left\|w_{\hat{U} \setminus U}\right\|_2 O(\epsilon\mu) + O(\epsilon^2\mu^2) \\
&\leq \left\|w_{\hat{U} \setminus U}\right\|_2^2 + O(\epsilon\mu^2)
\end{aligned}$$

because $\left\|w_{\hat{U} \setminus U}\right\|_2 \leq \mu$. Thus

$$\begin{aligned}
\|w - w_{\hat{U}}\|_2 = \left\|w_{[t] \setminus \hat{U}}\right\|_2^2 &\leq O(\epsilon\mu^2) + \left\|w_{\hat{U} \setminus U}\right\|_2^2 + \left\|w_{[t] \setminus (U \cup \hat{U})}\right\|_2^2 \\
&= O(\epsilon\mu^2) + \mu^2 = (1 + O(\epsilon))\mu^2.
\end{aligned}$$

This is exactly what we want. If $S = \bigcup_{i \in \hat{U}} T_i$ contains the blocks corresponding to $\hat{U}$, then

$$\|x - x_S\|_2 = \|w - w_{\hat{U}}\|_2 \leq (1 + O(\epsilon))\mu = (1 + O(\epsilon)) \operatorname{Err}_{(k,b)}(x)$$

Rescale ϵ to change $1 + O(\epsilon)$ into $1 + \epsilon$ and we're done. □

Bibliography

[1] M. Akcakaya and V. Tarokh. A frame construction and a universal distortion bound for sparse representations. *Signal Processing, IEEE Transactions on*, 56(6):2443 –2450, jun. 2008.

[2] A. Andoni, K. Do Ba, and P. Indyk. Block heavy hitters. *MIT Technical Report TR-2008-024*, 2008.

[3] Z. Bar-Yossef, T. S. Jayram, R. Krauthgamer, and R. Kumar. The sketching complexity of pattern matching. *RANDOM*, 2004.

[4] R. G. Baraniuk, V. Cevher, M. F. Duarte, and C. Hegde. Model-based compressive sensing. *IEEE Transactions on Information Theory*, 56, No. 4:1982–2001, 2010.

[5] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web. *Comput. Netw.*, 33(1-6):309–320, 2000.

[6] E. J. Candès, J. Romberg, and T. Tao. Stable signal recovery from incomplete and inaccurate measurements. *Comm. Pure Appl. Math.*, 59(8):1208–1223, 2006.

[7] V. Cevher. Learning with compressible priors. In *NIPS*, Vancouver, B.C., Canada, 7–12 December 2008.

[8] V. Cevher, P. Indyk, C. Hegde, and R. G. Baraniuk. Recovery of clustered sparse signals from compressive measurements. *SAMPTA*, 2009.

[9] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. *Theor. Comput. Sci.*, 312(1):3–15, 2004.

[10] Z. Chen and J. J. Dongarra. Condition numbers of gaussian random matrices. *SIAM Journal on Matrix Analysis and Applications*, 27:603–620, 2004.

[11] K. L. Clarkson and D. P. Woodruff. Numerical linear algebra in the streaming model. In *STOC*, pages 205–214, 2009.

[12] A. Coja-Oghlan, C. Moore, and V. Sanwalani. Counting connected graphs and hypergraphs via the probabilistic method. *Random Struct. Algorithms*, 31(3):288–329, 2007.

[13] G. Cormode and S. Muthukrishnan. Improved data stream summaries: The count-min sketch and its applications. *Latin*, 2004.

[14] G. Cormode and S. Muthukrishnan. Combinatorial algorithms for compressed sensing. *Sirocco*, 2006.

[15] W. Dai and O. Milenkovic. Weighted superimposed codes and constrained integer compressed sensing. *Preprint*, 2008.

[16] K. Do Ba, P. Indyk, E. Price, and D. Woodruff. Lower bounds for sparse recovery. *SODA*, 2010.

[17] D. L. Donoho. Compressed Sensing. *IEEE Trans. Info. Theory*, 52(4):1289–1306, Apr. 2006.

[18] M. Duarte, M. Davenport, D. Takhar, J. Laska, T. Sun, K. Kelly, and R. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 2008.

[19] D. Dubhashi, V. Priebe, and D. Ranjan. Negative dependence through the FKG inequality. In *Research Report MPI-I-96-1-020, Max-Planck-Institut fur Informatik, Saarbrucken*, 1996.

[20] D. Dubhashi and D. Ranjan. Balls and bins: A study in negative dependence. *Random Structures & Algorithms*, 13:99–124, 1996.

[21] Y. C. Eldar, P. Kuppinger, and H. Bölcskei. Compressed sensing of block-sparse signals: Uncertainty relations and efficient recovery. *CoRR*, abs/0906.3173, 2009.

[22] Y.C. Eldar and H. Bolcskei. Block-sparsity: Coherence and efficient recovery. *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, 2009.

[23] D. Eppstein and M. T. Goodrich. Space-efficient straggler identification in round-trip data streams via Newton's identitities and invertible Bloom filters. *WADS*, 2007.

[24] T. Ericson and L. Györfi. Superimposed codes in $\mathbb{R}^n$. *IEEE Trans. on Information Theory*, 34(4):877–880, 1988.

[25] S. Foucart, A. Pajor, H. Rauhut, and T. Ullrich. The Gelfand widths of lp-balls for $0 < p \leq 1$. *preprint*, 2010.

[26] Z. Füredi and M. Ruszinkó. An improved upper bound of the rate of euclidean superimposed codes. *IEEE Trans. on Information Theory*, 45(2):799–802, 1999.

[27] A. Gilbert and P. Indyk. Sparse recovery using sparse matrices. *Proceedings of IEEE*, 2010.

[28] A. C. Gilbert, Y. Li, E. Porat, and M. J. Strauss. Approximate sparse recovery: Optimizing time and measurements. *CoRR*, abs/0912.0229, 2009.

[29] P. Indyk. Sketching, streaming and sublinear-space algorithms. *Graduate course notes, available at* `http://stellar.mit.edu/S/course/6/fa07/6.895/`, 2007.

[30] P. Indyk and A. Naor. Nearest neighbor preserving embeddings. *ACM Trans. on Algorithms*, 3(3), Aug. 2007.

[31] K. Joag-Dev and F. Proschan. Negative association of random variables with applications. *The Annals of Statistics*, 11(1):286–295, 1983.

[32] D. Kane, J. Nelson, and D. Woodruff. On the exact space complexity of sketching and streaming small norms. In *SODA*, 2010.

[33] M. Karoński and T. Łuczak. The phase transition in a random hypergraph. *J. Comput. Appl. Math.*, 142(1):125–135, 2002.

[34] B. S. Kashin and V. N. Temlyakov. A remark on compressed sensing. *Preprint*, 2007.

[35] I. Kremer, N. Nisan, and D. Ron. On randomized one-round communication complexity. *Computational Complexity*, 8(1):21–49, 1999.

[36] E. Kushilevitz and N. Nisan. *Communication Complexity*. Cambridge University Press, 1997.

[37] C. La and M. N. Do. Signal reconstruction using sparse tree representation. In *in Proc. Wavelets XI at SPIE Optics and Photonics*, 2005.

[38] P. B. Miltersen, N. Nisan, S. Safra, and A. Wigderson. On data structures and asymmetric communication complexity. *J. Comput. Syst. Sci.*, 57(1):37–49, 1998.

[39] M. Mitzenmacher. A brief history of generative models for power law and lognormal distributions. *Internet Mathematics*, 1:226–251, 2004.

[40] S. Muthukrishnan. Data streams: Algorithms and applications (invited talk at soda'03). *Available at http://athos.rutgers.edu/∼muthu/stream-1-1.ps*, 2003.

[41] J. Romberg. Compressive sampling by random convolution. *SIAM Journal on Imaging Science*, 2009.

[42] M. Stojnic, F. Parvaresh, and B. Hassibi. On the reconstruction of block-sparse signals with an optimal number of measurements. *IEEE Trans. Signal Processing*, 2009.

[43] J.H. van Lint. *Introduction to coding theory*. Springer, 1998.

[44] M. Wainwright. Information-theoretic bounds on sparsity recovery in the high-dimensional and noisy setting. *IEEE Int'l Symp. on Information Theory*, 2007.