Sublinear Time, Measurement-Optimal, Sparse Recovery For All

Ely Porat* and Martin J. Strauss†

Abstract

An *approximate sparse recovery* system in ℓ_1 norm makes a small number of measurements of a noisy vector with at most k large entries and recovers those *heavy hitters* approximately. Formally, it consists of parameters N, k, ϵ, an m-by-N *measurement matrix*, $\mathbf{\Phi}$, and a *decoding algorithm*, $\mathcal{D}$. Given a vector, $\mathbf{x}$, where $\mathbf{x}_k$ denotes the optimal k-term approximation to $\mathbf{x}$, the system approximates $\mathbf{x}$ by $\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{\Phi}\mathbf{x})$, which must satisfy

$$\|\widehat{\mathbf{x}} - \mathbf{x}\|_1 \leq (1+\epsilon)\|\mathbf{x} - \mathbf{x}_k\|_1 .$$

Among the goals in designing such systems are minimizing the number m of measurements and the runtime of the decoding algorithm, $\mathcal{D}$. We consider the "forall" model, in which a single matrix $\mathbf{\Phi}$, possibly "constructed" non-explicitly using the probabilistic method, is used for all signals $\mathbf{x}$.

Many previous papers have provided algorithms for this problem. But all such algorithms that use the optimal number $m = O(k\log(N/k))$ of measurements require superlinear time $\Omega(N\log(N/k))$. In this paper, we give the first algorithm for this problem that uses the optimum number of measurements (up to constant factors) and runs in sublinear time $o(N)$ when k is sufficiently less than N. Specifically, for any positive integer ℓ, our approach uses time $O(\ell^5\epsilon^{-3}k(N/k)^{1/\ell})$ and uses $m = O(\ell^8\epsilon^{-3}k\log(N/k))$ measurements, with access to a data structure requiring space and preprocessing time $O(\ell N k^{0.2}/\epsilon)$.

1 Introduction

1.1 Description of Problem Variations of the Sparse Recovery problem are well-studied in recent literature. A vector (or signal) $\mathbf{x}$ is first measured, by the matrix-vector product $\mu = \mathbf{\Phi}\mathbf{x}$, then, at a later time, a decoding algorithm $\mathcal{D}$ approximates $\mathbf{x}$ from μ. The approximation is non-vacuously useful if $\mathbf{x}$ is dominated by a small number of large magnitude entries, called "heavy hitters." Applications arise in signal and image processing and database, with further application to telecommunications and medicine [DDT+08, LDP07]. Several workshops [CA09, SPA09] have been devoted to this topic. See more at [Ric06].

In this paper, we focus on the following variation. If N is the length of the signal, k is a sparsity parameter, and ϵ is a fidelity parameter, we want $\|\widehat{\mathbf{x}} - \mathbf{x}\|_1 \leq (1+\epsilon)\|\mathbf{x}_k - \mathbf{x}\|_1$, where $\mathbf{x}_k$ is the best possible k-term representation for $\mathbf{x}$. Among the goals in designing such systems are minimizing the number m of measurements and the runtime of the decoding algorithm, $\mathcal{D}$. We consider the "forall" model, in which a single matrix $\mathbf{\Phi}$, possibly "constructed" non-explicitly using the probabilistic method in polynomial time or explicitly in exponential time, is used for all signals $\mathbf{x}$.

1.2 Advantages over Previous Work

Previous measurement-optimal algorithms are slow. Many previous papers have provided algorithms for this problem. But all such algorithms that use the constant-factor-optimal number $O(k\log(N/k))$ of measurements require superlinear time $\Omega(N\log(N/k))$. In this paper, we give the first algorithm that, for any positive integer ℓ, uses $\ell^{O(1)}\epsilon^{-3}k\log(N/k)$ measurements (*i.e.*, $O(k\log(N/k))$ measurements for constant ℓ and ϵ) and run in time $\ell^{O(1)}\epsilon^{-3}k(N/k)^{1/\ell}$. For example, with $\ell = 2$ and $\epsilon = \Omega(1)$, the runtime improves from N to $\sqrt{kN}$. In some applications, sparse recovery is the runtime bottleneck and our contribution can make some other $\Omega(N)$ computation become the new bottleneck.

The sublinear runtime of our algorithm is important not because traditional algorithms are too slow, but because the measurement-optimal algorithms that replaced them are too slow. Consider an application in which $k \ll N$. A traditional approach makes exactly N direct measurements or (in some cases) requires little more than taking a single Fast Fourier Transform of length N. Optimized code for FFTs is so fast that one cannot plausibly claim to lower the runtime, say from $N\log N$ to $\sqrt{N}$, by a complicated algorithm with heavy overhead. But, in

*Bar Ilan University. Partially supported by BSF grant 2006334 and ISF grant 1484/08. Email: porately@gmail.com

†University of Michigan. Partially supported by NSF CCF 0743372, DARPA/ONR N66001-08-1-2065, and BSF grant 2006334. Email: martinjs@eecs.umich.edu

some cases (see below), taking more measurements than necessary is a significant liability. Several papers in the literature (see Table 1) improve the number of measurements from N to $k\log(N/k)$, but only with *significant increase* in the runtime, say, from computing a Fourier Transform of length N to a more complicated problem of size N, such as solving a linear program or, more recently, performing a combinatorial algorithm on expander graphs, of a flavor similar to our approach below. When k is small compared with N, we hope that (i) the number $O(k\log(N/k))$ of measurements made by our algorithm is significantly less than N in practice, and (ii) the sublinear runtime of our algorithm is significantly faster than that of other *measurement-optimal* algorithms, all of which use time $\Omega(N\log(N/k))$, and many of which have significant overhead. We do not expect that our "sublinear time" algorithm will compete on time with naive time $O(N)$ algorithms or with a single FFT, except for in unusual circumstances and/or values of k and N.

These questions have been actively studied by several communities. See Table 1, which is based in part on a table in [IR08].

Trading runtime for fewer measurements in sublinear-time algorithms. Previous sublinear-time algorithms for this problem have used too many measurements by logarithmic factors, which we now argue is inappropriate in certain situations. In a traditional approximation algorithm, there is an objective function to be minimized, and relatively small improvements in the approximation ratio for the objective function—from $O(\log n)$ to constant-factor to $(1+o(1))$—are considered well worth a polynomial blowup in computation time. In the sparse recovery problem, there are two objective functions—the approximation ratio by which the error $\|\widehat{\mathbf{x}}-\mathbf{x}\|_1$ exceeds the optimal, and the number of measurements. In case of medical imaging, the number of measurements is proportional to the duration during which a patient must lie motionless; a measurement blowup factor of "1000 times log of something" is unacceptable. In this paper, we reduce the blowup in number of measurements to a small integer constant factor.

Previous sublinear time algorithms had runtime polynomial in $k\log(N)$, often linear in k. By contrast, our algorithm gives runtime $\epsilon^{-3}\sqrt{kN}$ or, more generally, gives runtime $\ell^{O(1)}\epsilon^{-3}k(N/k)^{1/\ell}$ using $\ell^{O(1)}\epsilon^{-3}k\log(N/k)$ measurements, which remains slightly suboptimal. But, first, a blowup in runtime from, say, $k\log^2 N$ to $k(N/k)^{1/4}$ is appropriate to reduce the approximation ratio from logarithmic to small constant in a critical objective like number of measurements in certain applications. Second, the blowup is not that big in other applications, where, say, $(N/k)^{1/4}$ is not much bigger than $\log^2(N)$. Alternatively, a parametric sweet spot for our algorithm occurs around $k=N^{1/4}$. Putting $\ell=3$, we get runtime $k(N/k)^{1/3}=\sqrt{N}=k^2$. This is about the time to multiply a vector by a dense matrix of smallest useful size, which is a tiny component in some (early) algorithms in the literature, superlinear or sublinear.

Constant factor gap in number of measurements. The best previous *superlinear* algorithms [RV06] use a number of measurements that is suboptimal by a small constant factor versus the best known lower bounds. Thus, for sublinear-time algorithms, a small constant-factor gap, rather than an approximation scheme, is currently an appropriate goal.

All signals or Each signal? The results of this paper are in the "forall" model. Recently, a sublinear-time, constant-factor-optimal measurement algorithm was given [GPLS10] in an incomparable setup. In particular, its guarantees were for the weaker "foreach" model, in which a random measurement matrix works with *each* signal, but no single matrix works simultaneously on *all* signals.[1] The stronger forall model is more appropriate in certain applications, where, for example, there is a sequence $\mathbf{x}^{(1)},\mathbf{x}^{(2)}$ of signals to be measured by the same measurement matrix, and $\mathbf{x}^{(2)}$ depends, in some subtle way, on the result of recovering $\mathbf{x}^{(1)}$. (For example, an adversary may construct $\mathbf{x}^{(2)}$ after observing an action we take in response to recovering $\mathbf{x}^{(1)}$.) In the forall model, there is no issue. In the foreach model, however, it is important that an adversary pick the signal without knowing the outcome $\mathbf{\Phi}$. If the adverary knows something about the outcome $\mathbf{\Phi}$—such as observing our reaction to recovering $\mathbf{x}^{(1)}$ from $\mathbf{\Phi}\mathbf{x}^{(1)}$—the adversary may be able to construct an $\mathbf{x}^{(2)}$ in the null space of $\mathbf{\Phi}$, which would break an algorithm in the weaker foreach model.

1.3 Overview of Results and Techniques First, following previous work [GPLS10], we show that it suffices to recover all but approximately $k/2$ of k heavy hitters at a time. The cost for this in measurements is $ck\log N/k$, for some constant c. We then repeat on the remaining $k'=k/2$ heavy hitters, with cost $ck'\log N/k'\approx\frac{1}{2}ck\log N/k$, and leaving $k/4$ heavy hitters. Continuing this way, the total cost is a geometric progression with sum $O(k\log N/k)$. In fact, we will use somewhat more than $\frac{1}{2}ck'\log N/k'$ measurements, *e.g.*, $\frac{9}{10}ck'\log N/k'$ measurements, to enforce other re-

[1] In the forall model, the guarantee is that a matrix $\mathbf{\Phi}$ generated according to a specified distribution succeeds on *all* signals in a class $\mathcal{C}$. In the foreach model, there is a distribution on matrices, such that for *each* signal $\mathbf{x}$ in a class $\mathcal{C}'$ bigger than $\mathcal{C}$, a matrix $\mathbf{\Phi}$ chosen according to the prescribed distribution succeeds on $\mathbf{x}$. The difference in models is captured in the order of quantifiers, which can be anthropomorphized into the powers of a challenger and adversary.

Paper	A/E	No. Measurements	Column sparsity/ Update time	Decode time	Approx. error	Noise
[CCFC02]	E	$k \log^c N$	$\log^c N$	$N \log^c N$	$\ell_2 \leq C\ell_2$	
[CM06]	E	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_2 \leq C\ell_2$	
[CM04]	E	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_1 \leq C\ell_1$	
[GPLS10]	E	$k \log(N/k)$	$\log^c N$	$k \log^c N$	$\ell_2 \leq C\ell_2$	Y
[Don06, CRT06]	A	$k \log(N/k)$	$k \log(N/k)$	LP	$\ell_2 \leq (C/\sqrt{k})\ell_1$	Y
[GSTV06]	A	$k \log^c N$	$\log^c N$	$k \log^c N$	$\ell_1 \leq (C \log N)\ell_1$	Y
[GSTV07]	A	$k \log^c N$	$k \log^c N$	$k^2 \log^c N$	$\ell_2 \leq (\epsilon/\sqrt{k})\ell_1$	Y
[IR08]	A	$k \log(N/k)$	$\log(N/k)$	$N \log(N/k)$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
This paper (any integer ℓ)	A	$\ell^c k \log(N/k)$	$(\ell \log N)^c$	$\ell^c k (N/k)^{1/\ell}$	$\ell_1 \leq (1+\epsilon)\ell_1$	Y
Lower bound "A"	A	$k \log(N/k)$	$\log(N/k)$	$k \log(N/k)$	$\ell_2 \leq (\epsilon/\sqrt{k})\ell_1$	Y

Table 1: Summary of the best previous results and the result obtained in this paper. Some constant factors are omitted for clarity. "LP" denotes (at least) the time to do a linear program of size at least N. The column "A/E" indicates whether the algorithm works in the forall (A) model or the foreach (E) model. The column "noise" indicates whether the algorithm tolerates noisy measurements. Measurement and decode time dependence on ϵ, where applicable, is polynomial.

quirements while still keeping the number of measurements bounded by a geometric series that converges to $O(k \log N/k)$. As in [GPLS10], we present a compound loop invariant satisfied as the number of heavy hitters drops from k to $k/2$ to $k/4$, etc.

Next, we show how to solve the transformed problem, *i.e.*, how to reduce the number of unrecovered heavy hitters from k to $k/2$, while not increasing the noise by much. As in previous results, we estimate all N coefficients of $\mathbf{x}$ by hashing the positions into $O(k)$ buckets, hoping that each heavy hitter ends up dominating its bucket, so that the bucket aggregrate is a good estimate of the heavy hitter. We repeat $O(\log(N/k))$ times, and take a median of estimates. Finally, we replace by zero all but the largest $O(k)$ estimates. If all estimates were independent, then this would give the result we need, by the Chernoff bound; below we handle the minor dependence issues. We get a simple and natural system making $O(k \log N/k)$ measurements but with runtime somewhat larger than N.

Finally, to get a sublinear time algorithm, we replace the above exhaustive search over a space of size N with constantly-many searches over spaces of size approximately $\sqrt{kN} = k(N/k)^{1/2}$. Still more generally, replace with $\ell^{O(1)}$ searches over spaces of size $\ell^{O(1)}k(N/k)^{1/\ell}$, for any positive integer value of the user-parameter ℓ. As a tradeoff, this requires the factor $\ell^{O(1)}$ times more measurements. In the case $\ell = 2$, we first hash the original signal's indices into $\sqrt{kN}$ buckets, forming a new signal $\mathbf{x}'$, indexed by buckets. As we show, heavy hitters in $\mathbf{x}$ are likely to dominate their buckets, which become heavy hitters of $\mathbf{x}'$. We then find approximately k heavy buckets exhaustively, searching a space of size $\sqrt{kN}$. Each bucket corresponds to approximately $N/\sqrt{kN} = \sqrt{N/k}$ indices in the original signal, for a total of $k\sqrt{N/k} = \sqrt{kN}$ indices, which are now searched. This naturally leads to runtime $\sqrt{kN}\log(N/k)$, or $k(N/k)^{1/\ell}\log(N/k)$ for $\ell > 2$. By absorbing $\log(N/k)$ into $\ell^{O(1)}(N/k)^{1/\ell}$, we get, for general ℓ and ϵ,

THEOREM 1.1. *For any positive integer ℓ, there is a solution to the ℓ_1 forall sparse recovery problem running in time $O(\ell^5\epsilon^{-3}k(N/k)^{1/\ell})$ and using $O(\ell^8\epsilon^{-3}k\log(N/k))$ measurements, where N is the length, k is the sparsity, and ϵ is the approximation parameter. The algorithm uses a data structure that requires space and preprocessing time $O(\ell N k^{0.2}/\epsilon)$.*

Note: For thoroughness, we count the factors of ℓ and $1/\epsilon$. The reader is warned, however, that we are aware of possible improvements, so the reader may want to abstract ℓ^8/ϵ^3 and ℓ^5/ϵ^3 to the simpler expression $(\ell/\epsilon)^{O(1)}$. Similarly, the power 0.2 of k in the preprocessing costs can be improved but with constant-factor increases to runtime or number of measurements, and we suspect that expensive preprocessing can be eliminated altogether. (The focus of this paper is just sublinear runtime and constant-factor-optimal number of measurements, while other aspects of the algorithm are reasonable but not optimal.)

1.4 Organization of this paper This paper is organized as follows. Note that we are specifying a measurement matrix and a decoding algorithm; we refer to the combination as a *system*. In Section 2, we present notation and definitions. In Section 3, we present our main result, in three subsections. In Section 3.1, we show how to get a Weak system, that recovers all but $k/2$ of k heavy hitters, while not increasing the noise by much. This is a relatively slow Weak system that illustrates several concepts, on which we build. in Section 3.2, we build on

Section 3.1 to give a sublinear time version of the Weak system. In Section 3.3, we show how to get a solution to the main problem. In Section 4, we give several open problems in connection with optimizing and generalizing our results.

2 Preliminaries and Definitions

In this section, we present notation and definitions.

Systems. We will usually present systems for measurement and decoding as algorithmic units without all the details of the measurement matrix and decoding algorithm. We will then usually argue correctness at the system level, then argue that the system can be implemented by a matrix with the claimed number of rows and a decoding algorithm with the claimed runtime.

Notation. For any vector $\mathbf{x}$, we write $\mathbf{x}_k$ for the best k-term approximation to $\mathbf{x}$ or the k'th element of $\mathbf{x}$; it will be clear from context. For any vector $\mathbf{x}$, we write $\mathrm{supp}(\mathbf{x})$ for the *support* of x, *i.e.*, $\{i : \mathbf{x}_i \neq 0\}$.

Normalization. Our overall goal is to approximate $\mathbf{x}_k$, the best k-term approximation to $\mathbf{x}$. For the analysis in this paper, it will be convenient to normalize $\mathbf{x}$ so that $\|\mathbf{x} - \mathbf{x}_k\|_1 = 1$. It is not necessary for the decoding algorithm to know the original value of $\|\mathbf{x} - \mathbf{x}_k\|_1$.

Heavy Hitters. Suppose a signal $\mathbf{x}$ can be written as $\mathbf{x} = \mathbf{y} + \mathbf{z}$, where $|\mathrm{supp}(\mathbf{y})| \leq k$ and $\|\mathbf{z}\|_1 \leq \eta$. Then we say that $\mathrm{supp}(\mathbf{y})$ are the (k, η)-*heavy-hitters* of $\mathbf{x}$. We will frequently drop the (k, η)- when clear from context. Ambiguity in the decomposition $\mathbf{x} = \mathbf{y} + \mathbf{z}$ is inherent in approximate sparse *problems* and will not cause difficulty with our *algorithm*.

Optimal number of measurements. For this paper, we only consider algorithms using the optimal number of measurements, up to constant factors.[2]

We will use the following form of the Chernoff bound.

LEMMA 2.1. (CHERNOFF) *Fix real number p, $0 < p < 1$. Let $X_1, X_2, \ldots, X_n$ be a set of independent $0/1$-valued random variables with expectation p. Let $X = \sum_i X_i$ and let $\mu = pn$ denote $E[X]$. For any $\delta > 0$, we have*

$$\Pr(X > (1+\delta)\mu) < \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^\mu \leq \left(\frac{e\mu}{a}\right)^a,$$

where $a = (1+\delta)\mu$. If $a = \Omega(n)$ and $a > (1+\Omega(1))e\mu$, then the above probability is $p^{\Omega(n)}$.

[2]The optimal number of measurements, if ϵ is considered to be a constant, is [BIPW10] $\Theta(k\log(N/k)) = \Theta\left(\log\binom{N}{k}\right)$. Our ϵ dependence is cubic (quadratic in a warmup algorithm), which is suboptimal compared with the quadratic dependence in the best algorithms.

Parameter summary. We use the parameter k for sparsity in the toplevel signal, but a different symbol, the parameter s, in subroutines, so we can say things like, "put $s = k/2^j$ in the j'th iteration." Similarly, the parameters ϵ, α, and η are related "noise" or approximation ratio parameters in the various routines, and ζ is an "omission" parameter, such that we guarantee to recover all but ζs heavy hitters in an s-sparse signal.

3 Main Result

3.1 Weak System We start with a Weak System. Intuitively, a Weak system, operating on measurements, cuts in half the number of unknown heavy hitters, while not increasing tail noise by much. We estimate all values in $\mathbf{x}$ and take the largest $O(k)$ estimates. To estimate the values, we hash all N positions into $B = O(k)$ buckets. Each position i to estimate has a $\Omega(1)$ probability of getting hashed into a bucket with no (other) items larger than $1/k$ and the sum of other items not much more than the average value, $1/B \approx 1/k$, in which case the sum of values in the bucket estimates $\mathbf{x}_i$ to within $\pm 1/k$. By a concentration of measure argument for *dependent* random variables, we conclude that $\Omega(k)$ measurements are good except with probability $p = 2^{-O(k)}$, and, if we repeat $t = O(\log(N/k))$ times, some $\Omega(k)$ items get more than $t/2$ correct estimates except with probability $p^t = (k/N)^k$. In the favorable case, the median estimate is correct. The failure probability is small enough to take a union bound over all sets of $O(k)$ positions, so we conclude that no set of $\Omega(k)$ estimates is bad, *i.e.*, there are at most, say, $k/2$ failures, as desired.

We first present an algorithm that simply estimates *all* $[N]$ as suitable candidates. This makes the runtime slightly superlinear. Below, we will show how to get a smaller set of candidates, which speeds the algorithm at the cost of a controlable increase in the number of measurements.

DEFINITION 3.1. *A Weak system consists of parameters N, s, B, η, ζ, an m-by-N* measurement matrix*, $\mathbf{\Phi}$, and a* decoding algorithm*, $\mathcal{D}$. Consider signals $\mathbf{x}$ that can be written $\mathbf{x} = \mathbf{y} + \mathbf{z}$, where $|\mathrm{supp}(\mathbf{y})| \leq s$, $\mathrm{supp}(\mathbf{y}) \subseteq I$, and $\|\mathbf{z}\|_1 \leq O(1)$.*

Given the parameters, I, a measurement matrix $\mathbf{\Phi}$, and measurements $\mathbf{\Phi x}$ for any $\mathbf{x}$ with a decomposition above, the decoding algorithm returns $\widehat{\mathbf{x}}$, such that $\mathbf{x} = \widehat{\mathbf{x}} + \widehat{\mathbf{y}} + \widehat{\mathbf{z}}$, where $|\mathrm{supp}(\widehat{\mathbf{x}})| \leq O(s)$, $|\mathrm{supp}(\widehat{\mathbf{y}})| \leq \zeta s$, and $\|\widehat{\mathbf{z}}\|_1 \leq \|\mathbf{z}\|_1 + \eta$.

Without loss of generality, we may assume that $\mathrm{supp}(\mathbf{y}) \cap \mathrm{supp}(\mathbf{z}) = \mathrm{supp}(\widehat{\mathbf{y}}) \cap \mathrm{supp}(\widehat{\mathbf{z}}) = \emptyset$, but, in general, $\mathrm{supp}(\widehat{\mathbf{x}})$ intersects both $\mathrm{supp}(\widehat{\mathbf{y}})$ and $\mathrm{supp}(\widehat{\mathbf{z}})$.

The parameter B will always be set to $2s$ in imple-

mentations. We prove correctness for general B because the generality is needed to prove Lemma 3.2 below.

LEMMA 3.1. (WEAK) *With probability* $1 - \binom{N}{s}^{-\Omega(1)}$ *over the choice of hash functions, Algorithm 1, with appropriate instantiations of constants, is a correct Weak system that uses* $O(\eta^{-2}\zeta^{-4} s \log(N/s))$ *measurements when* $B = O(s)$ *and runs in time* $O(|I|\eta^{-1}\zeta^{-2}\log(N/s))$.

Proof. The number of measurements and runtime are as claimed by construction, so we show correctness. There are several parts to this, and much of this is similar to or implicit in previous work. We show that, with probability at least 3/4 over the choice of $\mathbf{\Phi}$:

1. For any set $S = \operatorname{supp}(\mathbf{y})$ of s heavy hitters and any set $D = \operatorname{supp}(\widehat{\mathbf{x}})$ of s "decoys" that might displace S, at most $O(\zeta s)$ elements of $S \cup D$ collide, in at least $t/4$ of their buckets, with an element of $S \cup D \cup T$, where T is the set of the top $O(s/(\zeta\eta))$ elements.

2. Let A be the set of rows of $\mathbf{\Phi}$ with a one anywhere in columns $S \cup D$ and let $\mathbf{\Phi}_A$ be $\mathbf{\Phi}$ restricted to the rows of A. Let F be a set of $\Omega(s/(\zeta\eta))$ columns disjoint from $S \cup D \cup T$, and let ν be an N-vector such that $\nu = 1/|F|$ on F and zero elsewhere (ν is a "flat tail"). We have $\|\mathbf{\Phi}_A \nu\|_1 \le O(\eta\zeta t)$.

3. Let A, $\mathbf{\Phi}_A$. For any ν supported off $S \cup D$ with $\|\nu\|_1 = 1$ and $\|\nu\|_\infty \le O(1/|\operatorname{supp}(T)|) = O(\eta\zeta/s)$, we have $\|\mathbf{\Phi}_A\nu\|_1 \le O(\eta\zeta t)$.

4. There is a decomposition $\mathbf{x} = \widehat{\mathbf{x}'} + \widehat{\mathbf{y}'} + \widehat{\mathbf{z}'}$ such that:
   - $\widehat{\mathbf{x}'}$ equals $\mathbf{x}'$ on $\operatorname{supp}(\mathbf{x}_s)$ and zero elsewhere, where $\mathbf{x}'$ is as in Algorithm 1,
   - $|\operatorname{supp}(\widehat{\mathbf{y}'})| \le \zeta s$
   - $\left\|\widehat{\mathbf{z}'}\right\|_1 \le \|\mathbf{z}\|_1 + O(\eta)$.

5. (The lemma's conclusion.) There's a decomposition $\mathbf{x} = \widehat{\mathbf{x}} + \widehat{\mathbf{y}} + \widehat{\mathbf{z}}$ with $|\operatorname{supp}(\widehat{\mathbf{x}})| \le O(s)$, $|\operatorname{supp}(\widehat{\mathbf{y}})| \le \zeta s$, and $\|\widehat{\mathbf{z}}\|_1 \le \|\mathbf{z}\|_1 + \eta$.

The dependence is as follows. Item 3 for general tails follows from Item 2 for flat tails. Item 4 follows from Items 1 and 3 and shows that the *estimates* lead to an acceptable decomposition of $\mathbf{x}$, assuming that *some* choice (generally unknown to the algorithm) of support for $\widehat{\mathbf{x}}$, namely $\operatorname{supp}(\mathbf{x}_s)$, is good. Finally, Item 5 follows from Item 4 by considering the displacement of an element in the support of $\mathbf{x}_s$ by an element in the Algorithm's output, *i.e.*, the support of $\widehat{\mathbf{x}}$. Only Items 1 and 2 involve probabilistic arguments.

Item 1. Fix a decomposition $\mathbf{x} = \mathbf{y} + \mathbf{z}$ as above, let S equal $\operatorname{supp}(\mathbf{y})$, and let $D \subseteq [N]$ be any set of s positions. (We only care about the case $D = \operatorname{supp}(\widehat{\mathbf{x}})$, but, to handle stochastic dependence issues, it is necessary to prove the result for a general D of this size.) We want to show that at most $O(\zeta s)$ elements of $S \cup D$ collide with one of the top $O(s/(\zeta\eta))$ elements in at least $t/4$ of their t buckets. Let T be the set of top $O(s/(\zeta\eta))$ elements in $[N]$.

Intuitively, there are $\Omega(\eta^{-1}\zeta^{-2}B)$ hash buckets and at most $O(|T|)$ are ever occupied by an element of $S \cup D \cup T$, so each element of $S \cup D$ has at most a $O(T\eta\zeta^2/B) = O(s\zeta/B) \le O(\zeta)$ chance to collide when it is hashed. As we discuss below, this implies that the expected number of collisions (at the time of hashing *or later*) is $O(s\zeta/B)$ in each of the t repetitions. If all estimates (over all i and all repetitions) were independent, we could apply the Chernoff bound Lemma 2.1, and conclude that the number of failed element-repetition pairs exceeds $O(\zeta|S \cup D|t) = O(\zeta s t)$ only with probability $\binom{N}{|T|}^{-\Omega(1)}$, small enough to take a union bound over all (S, D, T), which is acceptably small. But it is easy to see (and also see below) that there is at least some small dependence. So instead we proceed as follows, using a form of the Method of Bounded Differences and coupling [DP09, MR95, MU05].

First hash the elements of $T \setminus (S \cup D)$. Then hash the elements of $S \cup D$, in some arbitrary order. Let X_j be the 0/1-valued random variable that takes the value 1 if the j'th element of $S \cup D$ is hashed into a bucket that is bad (occupied by an element of $S \cup D \cup T$) at the time of j's hashing. As above, each X_j has $E[X_j] \le \zeta$.

Note that even if some $i \in S \cup D$ is isolated at the time of its hashing, i may become clobbered by an element of $j \in S \cup D$ that is later hashed into its bucket. So $\sum_j X_j$ is *not* the total number of failed estimates. But observe that if some j is hashed into the same bucket as previously-hashed items, it can only clobber at most one other previously-unclobbered element i, because j is only hashed into one bucket, and that bucket has at most one previously-unclobbered item. It follows that $2\sum_j X_j$ is an upper bound on the number of colliding items in $S \cup D$, where, for some p, the X_j's are 0/1-valued random variables with the expectation of each X_j bounded by p, even conditioned on any outcomes of $X_{<j}$. This is enough to get the conclusion of the Chernoff inequality with independent trials of failure probability p, by a standard coupling argument. (See, *e.g.*, exercise 1.7 of [DP09].) In the standard proof of Chernoff, we have,

Algorithm 1 A Weak system.

Input: N, sparsity s, noise η, $\mathbf{\Phi}$, $\mathbf{\Phi x}$, hash parameter B, omission ζ, candidate set $I = [N]$
Output: $\widehat{\mathbf{x}}$
for $j = 1$ to $t = O(\eta^{-1}\zeta^{-2}\log(N/s)/\log(B/s))$ **do**
  Hash $h : [N] \to [O(\eta^{-1}\zeta^{-2}B)]$
  for $i \in I$ **do**
    $\mathbf{x}_i^{(j)} = \sum_{h(i')=h(i)} \mathbf{x}_{i'}$ // sum of signal values in i's hash bucket—an element of input $\mathbf{\Phi x}$
  end for
end for
for $i \in I$ **do**
  Let $\mathbf{x}'_i$ be the median over $j \le t$ of $\mathbf{x}_i^{(j)}$
end for
Zero out all but the largest $O(s)$ elements of $\mathbf{x}'$; get $\widehat{\mathbf{x}}$
return $\widehat{\mathbf{x}}$

for any $\lambda > 0$,

$$\begin{aligned}\Pr\left(\sum_{j=1}^{n} X_j \ge a\right) &= \Pr\left(e^{\lambda\sum X_j} \ge e^{\lambda a}\right)\\ &= \Pr\left(\prod e^{\lambda X_j} \ge e^{\lambda a}\right)\\ &\le E\left[\prod e^{\lambda X_j}\right]/e^{\lambda a}.\end{aligned}$$

At this point, if the X_j's were independent, we would get the product of expectations. Instead, we proceed as in Figure 3.1, where Y_j's are independent random variables with expectation p. Then proceed inductively, getting

$$\begin{aligned}\Pr\left(\sum X_j \ge a\right) &\le E\left[\prod e^{\lambda Y_j}\right]/e^{\lambda a}\\ &= \frac{\prod E\left[e^{\lambda Y_j}\right]}{e^{\lambda a}},\end{aligned}$$

to which the rest of the usual proof of Chernoff-type bounds applies. Thus the expected number of pairs of elements in $S \cup D$ and repetition that collide is at most $O(\zeta st)$.

Having shown that our dependent collision events behave like independent events up to constants, we now go over the arithmetic, assuming independent collisions. Each $i \in S \cup D$ fails in each repetition with probability at most $O(\zeta s/B)$ (wlog, exactly $\zeta s/B$ for now). Among the $(2st)$ pairs of $i \in S \cup D$ and repetition, we expect to get $\mu = O(\zeta s^2 t/B)$ failed pairs, and we get at least $a \ge \zeta st$ failures with probability at most $(e\mu/a)^a$, by Lemma 2.1, Chernoff. So the failure probability is

$$\begin{aligned}(e\mu/a)^a &= (s/B)^{\Omega(\zeta st)}\\ &= (s/B)^{\Omega(\eta^{-1}\zeta^{-1}s\log(N/s)/\log(B/s))}\\ &= (s/N)^{\Omega(\eta^{-1}\zeta^{-1}s)},\end{aligned}$$

which is small enough to to take a union bound over (T, S, D). In the favorable case, there is only a fraction $O(\zeta)$ of all pairs of item and repetition with a failed estimate. It follows, after adjusting constants, that less than $(1/2)\zeta s$ items get more that $t/4$ failed original estimates. The remaining $(1-\zeta/2)s$ items get good final median estimate (even if another $t/4$ original estimates fail for other reasons, as we discuss below), since a median estimate fails only if a majority of mediand estimates fail.

Item 2. Fix $S, D, T, |F|, F$, choose the $S \cup D$ columns and the T columns of $\mathbf{\Phi}$ (arbitrarily for this discussion), and thereby define A (the rows of $\mathbf{\Phi}$ with a 1 in columns $S \cup D$) and ν (equal to $1/|F|$ on F and zero elsewhere), as above. We now hash the elements of F at random, *i.e.*, choose the F columns of $\mathbf{\Phi}$. In each repetition, there are $O(\eta^{-1}\zeta^{-2}B)$ buckets, of which $O(s)$ are in A. It follows that each element in F hashes to A in each repetition with probability $O(\eta\zeta^2 s/B)$. Counting repetitions, there are a total of $t|F|$ elements that each hash into A with probability $\eta\zeta^2 s/B$. We expect $\mu = \eta\zeta^2 t|F|s/B$ element-repetition pairs of $t|F|$ total to hash into A and we get more than $a = \eta\zeta^2 t|F|$ with probability at most

$$\begin{aligned}(e\mu/a)^a &= (s/B)^{\Omega(\eta\zeta^2 t|F|)}\\ &\le (s/B)^{\Omega(|F|\log(N/s)/\log(B/s))}\\ &= (s/N)^{\Omega(|F|)},\end{aligned}$$

which is small enough to take a union bound over all $S, D, T, |F|, F$. Since elements of ν have magnitude $1/|F|$, it follows that $\|\mathbf{\Phi}_A\nu\|_1 \le a/|F| = O(\eta\zeta^2 t)$, so[3] we conclude $\|\mathbf{\Phi}_A\nu\|_1 \le O(\eta\zeta t)$.

At this point, we have that, except with probability $1/4$, at most $O(\zeta s)$ of $S \cup D$ items collide with $S \cup D$

[3] This seems loose by a factor ζ, but local fixes, like replacing ζ with $\sqrt{\zeta}$, do not seem to work. We speculate that better dependence on ζ is possible.

Figure 1: Calculation of Chernoff-like bound for dependent random variables.

$$\begin{aligned}
\Pr\left(\sum X_j \geq a\right) &\leq E\left[\prod e^{\lambda X_j}\right]/e^{\lambda a} \\
&= E\left[e^{\lambda X_n}\prod_{j<n} e^{\lambda X_j}\right]/e^{\lambda a} \\
&= \sum_{\vec{v}} \left\{\Pr\left(X_n = 0 | X_{<n} = \vec{v}\right) + \Pr\left(X_n = 1 | X_{<n} = \vec{v}\right) e^{\lambda}\right\} \\
&\qquad \cdot \Pr(X_{<n} = \vec{v}) e^{\lambda \cdot \text{weight}(\vec{v})}/e^{\lambda a} \\
&\leq \sum_{\vec{v}} (1 - p + pe^{\lambda}) \Pr(X_{<n} = \vec{v}) e^{\lambda \cdot \text{weight}(\vec{v})}/e^{\lambda a} \\
&= E\left[e^{\lambda Y_n}\prod_{j<n} e^{\lambda X_j}\right]/e^{\lambda a}.
\end{aligned}$$

or with an element of T (of magnitude at least $\eta\zeta/s$) in more than $t/4$ of their repetitions and no flat tail of support size at least $s/(\eta\zeta)$ contributes more than a constant times its expected amount, which is $O(\eta\zeta t)$ if the magnitude of ν is maximal, into the buckets A containing the top s heavy hitters. Conditioned on this holding, we proceed non-probabilistically.

Item 3. Let ν be any vector supported disjointly from $S \cup D$ with $\|\nu\|_1 = 1$ and $\|\nu\|_\infty \leq O(\zeta\eta/s)$ as above. Since $\mathbf{\Phi}$ is non-negative, we may assume that ν is non-negative, as well, by replacing ν with $|\nu|$. Next, round each non-zero element of ν up to the nearest power of 2, at most doubling ν. Write $\nu = \sum_i w_i \nu_i$, where ν_i takes on only the values 0 and 2^{-i}, and w_i is 0 or 1. Also write $\nu = \nu' + \nu''$, where ν_i contributes to ν' if the support of ν_i is at least $s/(\eta\zeta)$ and ν_i contributes to ν'', otherwise. The ν_i's contributing to ν' are multiples of flat tails of the kind handled in Item 2 and their sum, ν', which has 1-norm at most 1, is a subconvex combination of such flat tails. Since $\|\mathbf{\Phi}_A\nu\|_1$ is subadditive in ν (actually, strictly additive under our non-negativity assumption), we get $\|\mathbf{\Phi}_A\nu'\|_1 \leq O(\eta\zeta t)$.

Now consider the sum ν'' of ν_i with support less than $s/(\eta\zeta)$. In general, these *can* contribute more than their expected value, but not *much* more than the expected value, and the expected value is typically much less than for other flat tails. We will handle the sum of these at once (without using the convex combination argument), so we may assume the supports are the maximum, $s/(\eta\zeta)$, by increasing each actual support to a superset. Also, we may assume that the corresponding w_i's are as large as possible, *i.e.*, $w_i = 1$ if $2^{-i} \leq \eta\zeta/s$ and $w_i = 0$, otherwise (so that the maximum magnitude is $\eta\zeta/s$). With these assumptions, the results of Item 2 apply, so each such flat tail contributes not much more than its expected number, $O(st)$, of elements of magnitude $2^{-i} = 2^{-j}\eta\zeta/s$ for some $j \geq 0$. Thus $\|\mathbf{\Phi}_A\nu_i\|_1 = O(\eta\zeta t 2^{-j})$ for i and j as above. The *sum* (which can be greater than a convex combination of the original contribution but, it turns out, is at most a constant times a convex combination under our assumptions) contributes $\|\mathbf{\Phi}_A\nu''\|_1 \leq O(\eta\zeta t \sum_{j\geq 0} 2^{-j}) = O(\eta\zeta t)$, as desired.

Thus $\|\mathbf{\Phi}_A\nu\|_1 \leq \|\mathbf{\Phi}_A\nu'\|_1 + \|\mathbf{\Phi}_A\nu''\|_1 \leq O(\eta\zeta t)$.

Item 4. Let $\widehat{\mathbf{x}}'$ be as above. In Item 1, we showed that an acceptable number $O(\zeta s)$ elements of $S \cup D$ suffer collisions; here we we consider only the elements of $S \cup D$ that do not collide with $S \cup D \cup T$. So we can consider only the tail elements that are still relevant, *i.e.*, the elements of $[N] \setminus (S \cup D \cup T)$, which have magnitude at most $\eta\zeta/s$. These form a tail ν as described in Item 3. Consider i to be a failure if

$$|\widehat{\mathbf{x}}'_i - \mathbf{x}_i| \geq \Omega(\eta/s).$$

Then each failed i in $\widehat{\mathbf{x}}'$ requires $t/2$ failed i's in $\mathbf{x}^{(j)}$'s and, since collisions only account for $t/4$ i's in $\mathbf{x}^{(j)}$'s, each failed i in $\widehat{\mathbf{x}}'$ that does not fail due to collisions also requires $\Omega(t)$ failed i's in $\mathbf{x}^{(j)}$'s. Thus each failed but non-colliding i accounts for $\Omega(t\eta/s)$ of $\|\mathbf{\Phi}_A\nu\|$. Since $\|\mathbf{\Phi}_A\nu\| \leq O(\eta\zeta t)$, there can be at most $O(\zeta s)$ failures, as desired. The remaining at-most-s estimates of $\mathbf{x}_s$ each are good to within $O(\eta/s)$, additively, so the total 1-norm of the estimation errors is $O(\eta)$, as desired.

Item 5. To complete our analysis of correctness, we describe $\widehat{\mathbf{x}}, \widehat{\mathbf{y}}$, and $\widehat{\mathbf{z}}$ and show that they have the claimed properties. This is summarized in Table 2.

Table 2: Contributions $\mathbf{x}_i - \widehat{\mathbf{x}}_i$ to $\widehat{\mathbf{y}}$ and to $\widehat{\mathbf{z}}$ from $i \in \mathrm{supp}(\mathbf{y})$ and $i \in \mathrm{supp}(\mathbf{z})$, according to whether $i \in \widehat{\mathbf{x}}$, whether $i \in \mathrm{supp}(\mathbf{x}_i)$ has a good or bad estimate (*i.e.* whether or not the median estimate is good to within $\pm O(\eta/s)$), or, if $i \in \mathrm{supp}(\mathbf{y}) \setminus \widehat{\mathbf{x}}$, according to whether i was displaced by i' with a good or bad estimate, under an arbitrary pairing between $i \in \mathrm{supp}(\mathbf{y}) \setminus \mathrm{supp}(\widehat{\mathbf{x}})$ and $i' \in \mathrm{supp}(\widehat{\mathbf{x}}) \setminus \mathrm{supp}(\mathbf{y})$. Note that zero may be a good estimate.

	$i \in \mathrm{supp}(\mathbf{y})$		$i \in \mathrm{supp}(\mathbf{z})$	
	Good estimate	Bad estimate	Good estimate	Bad estimate
$i \in \mathrm{supp}(\widehat{\mathbf{x}})$	$\widehat{\mathbf{z}}$	$\widehat{\mathbf{y}}$	$\widehat{\mathbf{z}}$	$\widehat{\mathbf{y}}$
$i \notin \mathrm{supp}(\widehat{\mathbf{x}})$; Displaced by bad estimate	$\widehat{\mathbf{y}}$	$\widehat{\mathbf{y}}$	$\widehat{\mathbf{z}}$	$\widehat{\mathbf{z}}$
$i \notin \mathrm{supp}(\widehat{\mathbf{x}})$; Displaced by good estimate	$\widehat{\mathbf{z}}$	$\widehat{\mathbf{y}}$	$\widehat{\mathbf{z}}$	$\widehat{\mathbf{z}}$

- The pseudocode Algorithm 1 returns $\widehat{\mathbf{x}}$, which has support size $O(s)$.

- Elements $i \in \mathrm{supp}(\widehat{\mathbf{x}})$ with a good estimate (to within $\pm O(\eta/s)$) contribute $\mathbf{x}_i - \widehat{\mathbf{x}}_i$ to $\widehat{\mathbf{z}}$. There are at most $O(s)$ of these, each contributing $O(\eta/s)$, for total contribution $O(\eta)$ to $\widehat{\mathbf{z}}$.

- Elements $i \in \mathrm{supp}(\widehat{\mathbf{x}})$ with a bad estimate (not to within $\pm O(\eta/s)$) contribute $\mathbf{x}_i - \widehat{\mathbf{x}}_i$ to $\widehat{\mathbf{y}}$. There are at most $O(\zeta s)$ of these.

- Elements $i \in \mathrm{supp}(\mathbf{z}) \setminus \mathrm{supp}(\widehat{\mathbf{x}})$ contribute $\mathbf{x}_i$ to $\widehat{\mathbf{z}}$. The ℓ_1 norm of these is at most $\|\mathbf{z}\|$.

- Elements $i \in \mathrm{supp}(\mathbf{y}) \setminus \mathrm{supp}(\widehat{\mathbf{x}})$ with a good estimate that are nevertheless displaced by another element $i' \in \mathrm{supp}(\widehat{\mathbf{x}}) \setminus \mathrm{supp}(\mathbf{y})$ with a good estimate contribute to $\widehat{\mathbf{z}}$. There are at most s of these. While the value $\mathbf{x}_i$ may be large and make a large contribution to $\widehat{\mathbf{z}}$, this is offset by $\mathbf{x}_{i'}$ satisfying, for some c, $|\mathbf{x}_{i'}| \geq |\widehat{\mathbf{x}}_{i'}| - c\eta/s \geq |\widehat{\mathbf{x}}_i| - c\eta/s \geq |\mathbf{x}_i| - 2c\eta/s$, which contributes to $\mathbf{z}$ but *not* to $\widehat{\mathbf{z}}$. Thus the net contribution to $\widehat{\mathbf{z}}$ is at most $O(\eta/s)$ for each of the $O(s)$ of these i, for a total $O(\eta)$ contribution to $\widehat{\mathbf{z}}$.

  The contributions of such i and i' are summarized in the following table, whence the reader can confirm that $(\mathbf{y}+\mathbf{z})_{\{i,i'\}} = (\widehat{\mathbf{x}}+\widehat{\mathbf{y}}+\widehat{\mathbf{z}})_{\{i,i'\}}$ and $\left\|\widehat{\mathbf{z}}_{\{i,i'\}}\right\| \leq \left\|\mathbf{z}_{\{i,i'\}}\right\| + O(\eta/s)$.

	$\mathbf{y}$	$\mathbf{z}$	$\widehat{\mathbf{x}}$	$\widehat{\mathbf{y}}$	$\widehat{\mathbf{z}}$
i	$\mathbf{x}_i$				$\mathbf{x}_i$
i'		$\mathbf{x}_{i'}$	$\widehat{\mathbf{x}}_{i'}$		$\mathbf{x}_{i'} - \widehat{\mathbf{x}}_{i'}$

- Elements $i \in \mathrm{supp}(\mathbf{y}) \setminus \mathrm{supp}(\widehat{\mathbf{x}})$ that themselves have bad estimates or are displaced by elements with bad estimates contribute $\mathbf{x}_i$ to $\widehat{\mathbf{y}}$. There are at most ζs bad estimates overall, so there are at most $O(\zeta s)$ of these.

We have shown that $|\mathrm{supp}(\widehat{\mathbf{y}})| \leq O(\zeta s)$ and $\|\widehat{\mathbf{z}}\|_1 \leq \|\mathbf{z}\|_1 + O(\eta)$. By adjusting constants in the algorithm, we can arrange for the conclusion of the Lemma.

3.2 Sublinear Time In this section, we introduce a way to limit I to get a sublinear time Weak system. Since the runtime of the weak system will dominate the overall runtime, it follows that the overall algorithm will have sublinear time. We first give a basic algorithm with runtime approximately $\sqrt{kN}$, then we generalize from $\sqrt{kN} = k(N/k)^{1/2}$ to $\ell^{O(1)}k(N/k)^{1/\ell}$ for any positive integer ℓ, but with number of measurements suboptimal by the factor $\ell^{O(1)}$.

The basic idea, for $\ell = 2$ and (ignoring for now the small effects of ϵ that we set to $\Omega(1)$), is as follows. Hash $h : [N] \to [\sqrt{kN}]$, and repeat a total of two times. In each repetition, a heavy hitter avoids collisions except with probability $k/\sqrt{kN} = \sqrt{k/N}$. Also, the average amount of tail noise (sum of others in the bucket) is $1/\sqrt{kN}$, so the tail noise exceeds $1/k$ on at most the fraction $k/\sqrt{kN} = \sqrt{k/N}$ of the buckets. So a heavy hitter dominates its bucket except with probability $O(\sqrt{k/N})$. The heavy hitter dominates in at least one of the two repetitions with failure probability equal to the square of that, or $O(k/N)$, which is what we would need to apply the Chernoff bound and to conclude that, except with probability $\binom{N}{k}^{-1}$ (which is small enough to take our union bound), $\Omega(k)$ of the heavy hitters are isolated in low-noise buckets. There is some dependence here, which is handled as in Section 3.1.

Now focus on one of the two repetitions. We can form a new signal x' of length $N' = \sqrt{kN}$ and sparsity $k' = \Omega(k)$. The signal x' is indexed by hash buckets and $x'_j = \sum_{h(i)=j} x_i$, *i.e.*, we sum the values in x that are hashed to the same bucket. The original (N, k)

signal (of length N and sparsity k) and a new $(N', k') = (\sqrt{kN}, \Omega(k))$ signal form what we call a two-level *signal filtration*, of which there are two, for the two repetitions.

For each filtration, run the Weak system Algorithm 1 on the (N', k') signal $\mathbf{x}'$, getting a set H of $\Theta(k)$ heavy hitters. This uses $O(k' \log(N'/k')) = O(k \log(N/k))$ measurements and runtime led by the factor $N' = \sqrt{kN}$. Form the set $I = h^{-1}(H)$ of indices to the *original* signal. Finally, run the Weak system on the original signal, but with index set I. This also takes $O(k \log(N/k))$ measurements and runtime led by the factor $|I| = \sqrt{kN}$. Thus the overall runtime is given by the time to make two exhaustive searches over spaces of size about $\sqrt{kN}$, on each of two repetitions, *i.e.*, ℓ repetitions of ℓ exhaustive searches over spaces of size $k(N/k)^{1/\ell}$, for $\ell = 2$. For correctness, we need to argue that the filtration is faithful to the original signal in the sense that enough heavy hitters from the original signal become heavy hitters in the (k', N') signals and that we can successfully track enough of these back to the original signal.

In the general situation, ℓ may be greater than 2. We will have $\ell - 1$ intermediate signals in the levels of the filtration, which we define below. The runtime will arise from performing ℓ repetitions of ℓ cascaded exhaustive searches over spaces of size about $k(N/k)^{1/\ell}$. There is strong overlap between the set of heavy hitters in the original signal and the set of heavy hitters in the shortest signal (of length $k(N/k)^{1/\ell}$). Assuming a correspondence of heavy hitters, our task is to trace each such heavy hitter in the shortest signal through longer and longer signals, back to the original (N, k) signal. Unfortunately, each time we ascend a level, we encounter more noise, and risk losing the trail of our heavy hitter. In the case of general ℓ, we will need to control noise and other losses by setting parameters as a function of ℓ. Roughly speaking, we need to lose no more than about k/ℓ heavy hitters at each level *i.e.*, $|\operatorname{supp}(\widehat{\mathbf{y}})| \leq k/\ell$, rather than losing, say, $k/2$, and (for general ϵ) we need to increase the noise by at most $O(\epsilon/\ell)$ rather than $O(\epsilon)$, *i.e.*, $\|\widehat{\mathbf{z}}\|_1 - \|\mathbf{z}\|_1 \leq O(\epsilon/\ell)$. This is done by setting the parameter ζ to $1/\ell$ and η to $O(\epsilon/\ell)$ instead of $O(\epsilon)$. Also, the number of repetitions must increase from $O(\ell)$ to $O(\ell/\epsilon)$.

We now proceed formally, for general number ℓ of levels.

DEFINITION 3.2. *Fix integer parameters s, N, and ℓ, and real $\xi > 0$. Given a signal $\mathbf{x}$ and a hash function $h : [N] \to [O((s/\xi)(N/s)^{1/\ell})]$, an ℓ-level* signal filtration *on $\mathbf{x}$ is a collection of ℓ signals, $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(\ell)}$, defined as follows. The signal $\mathbf{x}^{(q)}$ has length $N^{(q)} = O\left((s/\xi)(N/s)^{q/\ell}\right)$. Use the hash function $h : [N] \to [N^{(1)}]$ and define $\mathbf{x}^{(1)}_j$ by $\mathbf{x}^{(1)}_j = \sum_{h(i)=j} \mathbf{x}_i$. Then, for $1 \leq q < \ell$, define $\mathbf{x}^{(q+1)}$ from $\mathbf{x}^{(q)}$ by splitting each subbucket b indexing an element of $\mathbf{x}^{(q)}$ (*i.e.*, a subset of $[N]$) into subsubbuckets, in some arbitrary, deterministic way. Denote by $\operatorname{split}(b)$ the resulting set of subsubbuckets. Then $\mathbf{x}^{(q+1)} = \bigcup_b \operatorname{split}(b)$. Each subbucket is split into exactly $(N/s)^{1/\ell}$ subsubbuckets except that buckets in $\mathbf{x}^{(\ell-1)}$, which have size only $\xi(N/s)^{1/\ell}$, are split into $\xi(N/s)^{1/\ell}$ singletons, resulting in $\mathbf{x}$. See Figure 2.*

Consider a heavy index i in the original signal. It maps to a bucket, $h(i)$. In the favorable case, i dominates $h(i)$, in the sense that $|\mathbf{x}_i|$ accounts for, say, $3/4$ of the ℓ_1 norm of $h(i)$. Because the rest of the filtration involves only splitting buckets, it follows that i will dominate its bucket at each level of the filtration. For sufficiently many such i's, we therefore find the bucket the containing i in level $q+1$ using a Weak algorithm, inductively assuming we had the correct bucket at level q. We first show that enough heavy i's dominate their buckets.

LEMMA 3.2. (FILTRATION HASHING) *Fix parameters N, s, ℓ, α and let $\xi = \Theta(\alpha)$. Let*

$$h_j : [N] \to [(s/\xi)(N/s)^{1/\ell}]$$

be $O(\ell/\alpha)$ independent hash functions. With adjustable probability $\Omega(1)$ over $\mathbf{\Phi}$, the following holds. Given signal $\mathbf{x}$, suppose $\mathbf{x} = \mathbf{y} + \mathbf{z}$, with $|\operatorname{supp}(\mathbf{y})| \leq s$ and $\|\mathbf{z}\|_1 = 1$, and suppose, without loss of generality, that $|\mathbf{x}_i| \geq \Omega(\alpha/s)$ for $i \in \operatorname{supp}(\mathbf{y})$. We have $\mathbf{x} = \widehat{\mathbf{y}} + \widehat{\mathbf{w}} + \widehat{\mathbf{z}}$, where, for all $i \in \operatorname{supp}(\widehat{\mathbf{y}})$, i dominates some $h_j(i)$ and $|x_i| \geq \alpha/s$, $|\operatorname{supp}(\widehat{\mathbf{w}})| \leq s/6$, and $\|\mathbf{z}\|_1 \leq 1 + O(\alpha)$.

Proof. This follows directly from Lemma 3.1, Item 4, letting ζ be a constant, the B of Lemma 3.1 equal $s(N/s)^{1/\ell}$, and η of Lemma 3.1 equal $\Theta(\xi)$ (which is also $\Theta(\alpha)$). Then $\mathbf{x}'$ of Lemma 3.1, Item 4 gives $\widehat{\mathbf{y}}$ of this lemma (these are the surviving heavy hitters); $\widehat{\mathbf{y}}$ of Lemma 3.1 gives $\widehat{\mathbf{w}}$ of this lemma (these are the ruined heavy hitters), and the $\widehat{\mathbf{z}}$'s in the Lemmas coincide.

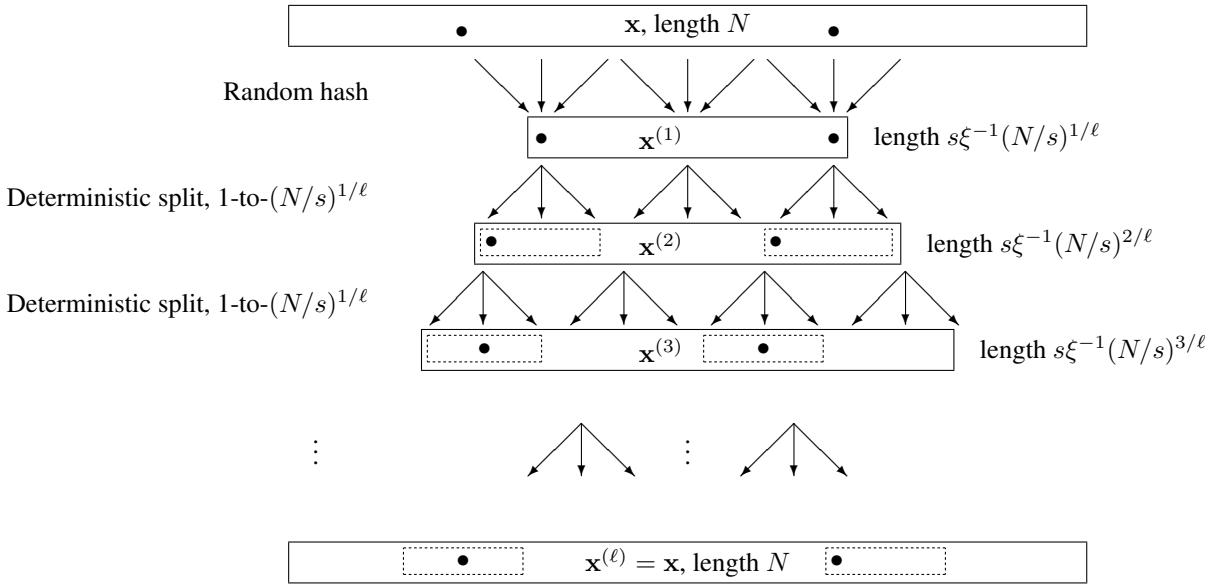
Our Sublinear Time Weak system is given in Algorithm 2.

LEMMA 3.3. *With proper instantiations of constants, and with fixed values $\zeta = 1/2$, $B = 2s$, and $I = [N]$, Algorithm 2 is a correct Weak system (Definition 3.1). The number of measurements is $O(\ell^8 \alpha^{-3} s \log(N/s))$ and the runtime is $O(\ell^5 \alpha^{-3} s (N/s)^{1/\ell})$, assuming a data structure that uses preprocessing and space $O(\ell N/\alpha)$.*

Proof. We maintain the following invariant for all q:

INVARIANT 3.1. *We have $\mathbf{x} = \mathbf{y} + \mathbf{w} + \mathbf{z}$, where $\operatorname{supp}(\mathbf{y}) \subseteq \bigcup_j I_{q,j}$, $|\operatorname{supp}(\mathbf{w})| \leq (s/6)(1 + (q-1)/\ell)$, and $\|\mathbf{z}\|_1 \leq 1 + \alpha(q-1)/\ell$. Elements of $\mathbf{y}$ dominate their buckets. The size of $\bigcup_j I_{q,j}$ is $s(N/s)^{1/\ell}$.*

Figure 2: A signal filtration. Heavy hitters, denoted by bullets, are likely isolated in low-noise buckets by the hashing, in which case they dominate their buckets at all levels of the deterministic splitting. Algorithm 1 (Weak) is used to search all of $\mathbf{x}^{(1)}$. For $q > 1$, given a set H of heavy hitters in $\mathbf{x}^{(q)}$, Algorithm 1 (Weak) is also used to find heavy hitters in $\mathbf{x}^{(q+1)}$, but we search only $(N/s)^{1/\ell}$ items of $\mathbf{x}^{(q+1)}$ in $I = \bigcup_{b \in H} \text{split}(b)$ (indicated by dashed boxes).



The invariant holds at initialization by Lemma 3.2 (Filtration). This is because the elements in $\mathbf{y}$ can be assumed to be of magnitude at least α/s and to dominate their buckets, while the filtration process preserves the noise ℓ_1 norm. The invariant is maintained as q increases by Lemma 3.1 (Weak). The failure probability can be taken small enough so that we can take a union bound over all $\ell \leq \log(N)$ levels times the number of choices in each level (addressed in the proof of Lemma 3.1 (Weak)).

At $q = \ell$, we have $\text{supp}(\mathbf{y}) \subseteq I$. Each $I_{\ell,j}$ has size $O(s)$, since it is the unsplit support of the output of Algorithm 1, so $|I| = O(s\ell/\alpha)$. Also, $|\text{supp}(\mathbf{w})| \leq s/3$ and $\|\mathbf{z}\|_1 \leq 1 + O(\alpha)$. The final call to Algorithm 1 (Weak) recovers all but another $s/6$ of the support of $\mathbf{y}$, which, when combined with $\mathbf{w}$, gives at most $s/2$ missed heavy hitters—the vector $\widehat{\mathbf{y}}$ in the definition of a Weak system. It also contributes an acceptable amount $O(\alpha)$ of additional noise that, with $\mathbf{z}$, constitute $\widehat{\mathbf{z}}$ in the definition of a Weak system.

Costs. The number of measurements and runtime is correct by construction, assuming the hash and split operations can be done in constant time. This is straightforward using a hash table with appropriate pointers for the split operation. Such a data structure needs space $O(N)$ and preprocessing $O(N)$ for each of the $O(\ell/\alpha)$ repetitions, for a total of $O(\ell N/\alpha)$. Note that the total cost, over all ℓ levels, is only $O(\ell N/\alpha)$ and not $O(\ell^2 N/\alpha)$, since the contributions from the levels form a geometric series.

In more detail, we first consider dependence on α and, below, on ℓ. The number of measurements is proportional to α^{-3}, since the number of repetitions is proportional to α^{-1} and the error parameter η is proportional to α, so each call to Algorithm 1 requires α^{-2} measurements. The bottom ($q = 1$) level takes runtime cubic in α, since there are $O(\ell/\alpha)$ repetitions of $I_{1,j}$ of size $O((s/\alpha)(N/s)^{1/\ell})$ and the error parameter η is proportional to α. Other levels take runtime just α^{-2}, since $|I_{>1,j}|$ has size $O(s(N/s)^{1/\ell})$.

The number of measurements depends on the *eighth* power of ℓ: one factor for the number of repetitions in the outer loop, one factor for the number of levels in the inner loop, ℓ^2 for the tighter approximation parameter $\eta = \alpha/\ell$ and ℓ^4 for the tighter omission parameter $\zeta = 1/\ell$, that contribute the factor $\eta^{-2}\zeta^{-4}$ to the costs. The runtime of each call to Algorithm 1 is proprotional to only the first power of $\eta\zeta^2$ times $|I|\log(N/s)$. The bottom level of the filtration involves a search over I of size $(s/\alpha)(N/s)^{1/\ell}$ for $\alpha \approx \epsilon/\ell$, while the other ℓ levels of the filtration search over $O(s(N/s)^{1/\ell})$. Thus the runtime is $O(\ell^5\alpha^{-3}s(N/s)^{1/\ell}\log(N/s))$.

Finally, note that, $(N/s)^{1/\ell}\log(N/s) \leq$

Algorithm 2 A Fast Weak system.

Input: N, sparsity s, noise α, $\mathbf{\Phi}$, $\mathbf{\Phi x}$
Global integer $\ell \geq 2$ // optimize for application
Output: $\widehat{\mathbf{x}}$
for $j \leftarrow 1$ to $t = O(\ell/\alpha)$ **do**
  Pick hash function $h : [N] \to [N^{(1)}]$, using parameters N, s, ℓ input and $\xi \leftarrow \Theta(\alpha)$.
  Implement by a hash table augmented with backpointers and threads for and enumerating preimages
  Let $\mathbf{x}^{()}$ be the filtration of $\mathbf{x}$ by h
  $I_{1,j} \leftarrow [N^{(1)}]$
  // track back through levels of the filtration
  for $q \leftarrow 1$ to $\ell - 1$ **do**
    Call Algorithm 1 (Weak) on $I_{q,j}$, $\mathbf{x}^{(q)}$, $\zeta \leftarrow 1/\ell$, noise $\eta \leftarrow \alpha/\ell$, sparsity s, and $B = 2s$, getting $\widehat{\mathbf{x}}$
    if $q < \ell$ **then**
      $I_{q+1,j} \leftarrow \bigcup_{b \in \mathrm{supp}(\widehat{\mathbf{x}})} \mathrm{split}(b)$
    end if
  end for
  $I \leftarrow \bigcup_j I_{\ell,j}$
end for
Call Algorithm 1 (Weak) on $\mathbf{x}$, I, $\zeta \leftarrow 1/6$, noise $\eta \leftarrow \Omega(\alpha)$, sparsity s, $B = 2s$, getting $\widehat{\mathbf{x}}$
return $\widehat{\mathbf{x}}$

$(N/s)^{1/(\ell-1)}$. By putting $\ell_0 = \ell - 1$, we get

$$\ell^5 (N/s)^{1/\ell} \log(N/s) \leq (\ell_0 + 1)^5 (N/s)^{1/\ell_0},$$

which is $O(\ell_0^5 (N/s)^{1/\ell_0})$, so we lose the $\log(N/s)$ factor for sufficiently large N/s.

Some remarks follow. Note that both the filtration and the measurement process of Algorithm 1 involve hashing. While the hashing of Algorithm 1 into $B = \eta^{-1}\zeta^{-2}s$ buckets results in B measurements in each of $\eta^{-1}\zeta^{-2}\log(N/s)$ repetitions, the hashing to create a filtration does not *directly* result in measurements or any recovery-time object. We never make $(N/s)^{1/\ell}$ measurements—that would be too many—and we do not instantiate the upper levels of the filtration at decode time—instantiating a signal of length $(N/s)^{1-1/\ell}$ would take too long.

3.3 Toplevel System Finally, we give a Toplevel system. The construction here closely follows [GPLS10] (where it was presented for the foreach, ℓ_2-to-ℓ_2 problem). A Toplevel system is an algorithm that solves our overall problem.

DEFINITION 3.3. *An* approximate sparse recovery system *(briefly, a* **Toplevel** *system), consists of parameters N, k, ϵ, an m-by-N* measurement matrix*, $\mathbf{\Phi}$, and a* decoding algorithm*. Fix a vector, $\mathbf{x}$, where $\mathbf{x}_k$ denotes the optimal k-term approximation to $\mathbf{x}$. Given the parameters and $\mathbf{\Phi x}$, the system approximates $\mathbf{x}$ by $\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{\Phi x})$, which must satisfy $\|\widehat{\mathbf{x}} - \mathbf{x}\|_1 \leq (1+\epsilon)\|\mathbf{x}_k - \mathbf{x}\|_1$.*

THEOREM 3.1. (TOPLEVEL) *Fix parameters N, k, ℓ. Algorithm 3 (Toplevel) returns $\widehat{\mathbf{x}}$ satisfying*

$$\|\widehat{\mathbf{x}} - \mathbf{x}\|_1 \leq (1+\epsilon)\|\mathbf{x}_k - \mathbf{x}\|_1 .$$

It uses $O(\ell^8 \epsilon^{-3} k \log(N/k))$ measurements and runs in time $O(\ell^5 \epsilon^{-3} k (N/k)^{1/\ell})$, using a data structure requiring $O(\ell N k^{0.2}/\epsilon)$ preprocessing time and storage space.

Proof. [sketch] Intuitively, the first iteration of Algorithm 3 transforms a measured but unknown k-sparse signal with noise magnitude 1 to a measured but unknown $(k/2)$-sparse signal with noise $1 + O(\epsilon)$. In subsequent iterations, the sparsity s decreases (relaxes) from k to $k/2$ to $k/4$ while the noise tolerance α decreases (tightens) from ϵ to $(9/10)\epsilon$ to $(9/10)^2\epsilon$, etc. We save a factor 2 in the number of measurements because s decreases and that more than pays for an increase in number of measurements by the factor $(10/9)^2$, that arises because η decreases. Thus measurement cost is bounded by decreasing geometric series and so is bounded by the first term, which is the measurement cost of the first iteration. Overall error is the sum of a decreasing geometric series with ratio $9/10$, so the overall error $\|\widehat{\mathbf{z}}\|_1$ remains bounded, by $1 + O(\epsilon) \leq 2$, with the given algorithm. A similar argument (with an additional wrinkle) holds for runtime.

More formally, note that the returned vector $\widehat{\mathbf{x}}$ has $O(k)$ terms. There is an invariant that $\mathbf{x} = \widehat{\mathbf{x}} + \mathbf{y} + \mathbf{z}$, where μ is the measurement vector for $\mathbf{y}+\mathbf{z}$, $|\mathrm{supp}(\mathbf{y})| \leq$

Algorithm 3 Toplevel System

Input: $\mathbf{\Phi}$, $\mathbf{\Phi x}$, N, k, ϵ
Output: $\widehat{\mathbf{x}}$
$\widehat{\mathbf{x}} \leftarrow 0$
$\mu \leftarrow \mathbf{\Phi x}$
for $j = 1$ to $\lg k$ **do**
  Run Algorithm 2 (Fast Weak) on μ with length N, sparsity $s \leftarrow k/2^j$, approx'n $\alpha \leftarrow O(\epsilon(9/10)^j)$
  Let $\mathbf{x}'$ be the result
  Let $\widehat{\mathbf{x}} = \widehat{\mathbf{x}} + \mathbf{x}'$
  Let $\mu = \mu - \mathbf{\Phi x}'$
end for
return $\widehat{\mathbf{x}}$

$k/2^j$ and

$$\|\mathbf{z}\|_1 \leq 1 + O(\epsilon)\left[\frac{9}{10} + \left(\frac{9}{10}\right)^2 + \cdots + \left(\frac{9}{10}\right)^j\right]$$

after j iterations. This is true at initialization, where $\mathbf{y} = \mathbf{x}_k$ and $\|\mathbf{z}\|_1 = \|\mathbf{x} - \mathbf{x}_k\|_1 = 1$. At termination, $\mathbf{x} = \widehat{\mathbf{x}} + \mathbf{z}$, with $\|\mathbf{z}\|_1 \leq 1 + O(\epsilon)$, since the infinite geometric series sums to 3. Maintenance of the loop invariant follows from correctness of the Weak algorithm.

Using the bound on measurements for Algorithm 2, the number of measurements used by Algorithm 3 is proportional to

$$\begin{aligned}
&\sum_j \ell^8 \epsilon^{-3}(k/2^j)\log(N2^j/k)(10/9)^{2j} \\
&\quad \leq \quad \ell^8\epsilon^{-3}k\log(N/k)\sum_j (50/81 + o(1))^j \\
&\quad = \quad O(\ell^8\epsilon^{-3}k\log(N/k)).
\end{aligned}$$

Similarly, using the runtime bound for Algorithm 1 and writing $s(N/s)^{1/\ell}$ as $s^{1-1/\ell}N^{1/\ell}$, the runtime of Algorithm 3 is proportional to

$$\begin{aligned}
&\sum_j \ell^5\epsilon^{-3}(k/2^j)^{1-1/\ell}N^{1/\ell}(10/9)^{3j} \\
&\quad \leq \quad \ell^5\epsilon^{-3}k(N/k)^{1/\ell}\sum_j \left[(10/9)^3 2^{1/\ell - 1}\right]^j \\
&\quad \leq \quad \ell^5\epsilon^{-3}k(N/k)^{1/\ell}\sum_j \left[(10/9)^3 2^{-1/2}\right]^j, \\
&\qquad\qquad \text{since } \ell \geq 2 \\
&\quad \leq \quad \ell^5\epsilon^{-3}k(N/k)^{1/\ell}\sum_j 0.97^j \\
&\quad \leq \quad O(\ell^5\epsilon^{-3}k(N/k)^{1/\ell}).
\end{aligned}$$

Finally, the storage space for hash tables in Algorithm 2 is N for each of ℓ/α repetitions. This is dominated by the smallest α, which is $\epsilon(9/10)^{\lg k} = \epsilon k^{-\lg 10/9} \geq \epsilon k^{-0.2}$, giving $N\ell k^{0.2}/\epsilon$ space and preprocessing. For any constant real-valued $c > 0$, this can be improved to $(1/c)^{O(1)}k^c$ by replacing $9/10$ with $1 - c$ and ϵ with $c\epsilon$. This will also increase the runtime and number of measurements by a constant factor that depends on c.

4 Open Problems

In this section, we present some generalizations of our algorithm that we leave as open problems.

Small space. Above we presented an algorithm that used superlinear space to store and to invert a hash function. The amount of space is partially excuseable because it can be amortized over many instances of the problem, *i.e.*, many signals. It also has the advantage over a hash function that hash operations can be performed simply in time $O(1)$. It should be possible, however, to use a standard hash function instead of a hash table to avoid the space requirement, though the runtime will likely increase. We leave as an open problem a fuller treatment of these tradeoffs.

Column Sparsity. An advantage in sparse recovery is the sparsity of the measurement matrix, $\mathbf{\Phi}$. Our matrix can easily seen to have at most $(\ell/\epsilon)^{O(1)}\log(N/k)\log(k)$ non-zeros per column, *i.e.*, there is no leading factor of k. But we have not optimized $\mathbf{\Phi}$ for column sparsity and we leave that for future work.

Post-measurement noise. Many algorithms in the literature give, as input to the decoding algorithm, not $\mathbf{\Phi x}$, but $\mathbf{\Phi x} + \nu$, where ν is an arbitrary noise vector. The algorithm's performance must degrade gracefully with $\|\nu\|$ (usually the 2-norm of ν). It can be seen that our algorithm does tolerate substantial noise, but in ℓ_1 norm. We leave to future work full analysis and possible improved algorithms.

Lower overhead in number of measurements. The approach we present produces a Toplevel system from a Weak system, using a filtration. It has a blowup factor of

ℓ^8 in the number of measurements over a weak system, where $\ell > 1$ is an *integer*. Thus the blowup factor in number of measurements for a time-$\sqrt{kN}\log(N/k)$ algorithm is at least 256, even (implausibly) ignoring all overhead and other constant factors. This should be improved.

Simplify. In [NT08], the authors take a different approach to fast algorithms. They argue that a small number of Fourier transforms of length N in a simple algorithm that takes linear time with a DFT oracle will be faster in practice than an algorithm that is complicated but asymptotically sublinear. They give an algorithm, CoSaMP, with runtime slightly greater than N, under a plausible assumption about random row-submatrices of the Fourier matrix and a bound on the "dynamic range" of the problem, *i.e.*, the ratio of $\|\mathbf{x}\|_2$ to $\|\mathbf{x} - \mathbf{x}_k\|_2$.

In the spirit of that paper, it would be good to use our speedup approach under the same assumptions as their paper, with $\ell = 2$. That is, ideally, we would want to double or triple the number of DFTs in the original CoSaMP, but reduce the length of the DFTs from N to approximately $\sqrt{kN}$. Our algorithm also suffers considerable overhead in converting a Weak algorithm into a Toplevel algorithm—a significant flaw if the goal is a simple, low-overhead algorithm—but CoSaMP has a similar iterative structure and it is conceivable that our Weak-to-Toplevel overhead can be combined subadditively with CoSaMP's iterative overhead.

Acknowledgement

We thank Anna Gilbert, Yi Li, Hung Ngo, Atri Rudra, and Mary Wootters for discussions and for reading an earlier draft.

References

[BIPW10] K. Do Ba, P. Indyk, E. Price, and D. Woodruff. Lower bounds for sparse recovery. In *ACM SODA*, page to appear, 2010.

[CA09] Lawrence Carin and Gregory Arnold. Compressive-sensing workshop. See `http://people.ee.duke.edu/~lcarin/` (retrieved Oct 1, 2011), February 2009.

[CCFC02] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. *ICALP*, 2002.

[CM04] G. Cormode and S. Muthukrishnan. Improved data stream summaries: The count-min sketch and its applications. *FSTTCS*, 2004.

[CM06] G. Cormode and S. Muthukrishnan. Combinatorial algorithms for Compressed Sensing. In *Proc. 40th Ann. Conf. Information Sciences and Systems*, Princeton, Mar. 2006.

[CRT06] 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.

[DDT+08] Marco Duarte, Mark Davenport, Dharmpal Takhar, Jason Laska, Ting Sun, Kevin Kelly, and Richard Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83–91, March 2008.

[Don06] D. L. Donoho. Compressed Sensing. *IEEE Trans. Info. Theory*, 52(4):1289–1306, Apr. 2006.

[DP09] Devdatt P. Dubhashi and Alessandro Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[GPLS10] Anna Gilbert, Ely Porat, Yi Li, and Martin Strauss. Approximate sparse recovery: Optimizing time and measurements. In *Proceedings of 42'd SToC*. ACM, June 2010.

[GSTV06] Anna C. Gilbert, Martin J. Strauss, Joel A. Tropp, and Roman Vershynin. Algorithmic linear dimension reduction in the l_1 norm for sparse vectors. *CoRR*, abs/cs/0608079, 2006.

[GSTV07] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin. One sketch for all: fast algorithms for compressed sensing. In *ACM STOC 2007*, pages 237–246, 2007.

[IR08] P. Indyk and M. Ruzic. Near-optimal sparse recovery in the l_1 norm. *FOCS*, 2008.

[LDP07] Michael Lustig, David Donoho, and John M. Pauly. Sparse MRI: The application of compressed sensing for rapid MR imaging. *Magnetic Resonance in Medicine*, 58(6):1182–1195, December 2007.

[MR95] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[NT08] D. Needell and J. A. Tropp. CoSaMP: Iterative signal recovery from incomplete and inaccurate samples. *Appl. Comp. Harmonic Anal.*, 2008.

[Ric06] Rice DSP group. `http://dsp.rice.edu/cs` (retrieved July 5, 2010), 2006.

[RV06] M. Rudelson and R. Vershynin. Sparse reconstruction by convex relaxation: Fourier and Gaussian measurements. In *CISS'06 (40th Annual Conference on Information Sciences and Systems)*, pages 207–212, 2006.

[SPA09] SPARS. Spars workshop. `http://spars09.inria.fr/ENGLISH/ENGLISH%20INDEX/welcome1.html` (retrieved July 6, 2010), April 2009.