APPROXIMATE SPARSE RECOVERY: OPTIMIZING TIME AND MEASUREMENTS

A. C. GILBERT, Y. LI, E. PORAT, AND M. J. STRAUSS

Abstract. An *approximate sparse recovery* system consists of parameters k, N, an m-by-N *measurement matrix*, $\mathbf{\Phi}$, and a decoding algorithm, $\mathcal{D}$. Given a vector, $\mathbf{x}$, the system approximates $\mathbf{x}$ by $\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{\Phi}\mathbf{x})$, which must satisfy $\|\widehat{\mathbf{x}} - \mathbf{x}\|_2 \leq C \|\mathbf{x} - \mathbf{x}_k\|_2$, where $\mathbf{x}_k$ denotes the optimal k-term approximation to $\mathbf{x}$. For each vector $\mathbf{x}$, the system must succeed with probability at least 3/4. Among the goals in designing such systems are minimizing the number m of measurements and the runtime of the decoding algorithm, $\mathcal{D}$.

In this paper, we give a system with $m = O(k \log(N/k))$ measurements—matching a lower bound, up to a constant factor—and decoding time $O(k \log^c N)$, matching a lower bound up to $\log(N)$ factors. We also consider the encode time (*i.e.*, the time to multiply $\mathbf{\Phi}$ by x), the time to update measurements (*i.e.*, the time to multiply $\mathbf{\Phi}$ by a 1-sparse x), and the robustness and stability of the algorithm (adding noise before and after the measurements). Our encode and update times are optimal up to $\log(N)$ factors. The columns of $\mathbf{\Phi}$ have at most $O(\log^2(k) \log(N/k))$ non-zeros, each of which can be found in constant time. If $\mathbf{x}$ is an exact k-sparse signal and ν_1 and ν_2 are arbitrary vectors (regarded as noise), then, setting $\widehat{\mathbf{x}} = \mathcal{D}(\Phi(\mathbf{x} + \nu_1) + \nu_2)$, we get

$$\|\widehat{\mathbf{x}} - \mathbf{x}\|_2 \leq 2 \|\nu_1\|_2 + \log(k) \frac{\|\nu_2\|_2}{\|\Phi\|_{2 \rightsquigarrow 2}},$$

where $\|\Phi\|_{2 \rightsquigarrow 2}$ is a natural scaling factor that makes our result comparable with previous results. (The $\log(k)$ factor above, improvable to $\log^{1/2+o(1)} k$, makes our result (slightly) suboptimal when $\nu_2 \neq 0$.) We also extend our recovery system to an FPRAS.

1. Introduction

Tracking heavy hitters in high-volume, high-speed data streams [4], monitoring changes in data streams [5], designing pooling schemes for biological tests [10] (e.g., high throughput sequencing, testing for genetic markers), localizing sources in sensor networks [15, 14] are all quite different technological challenges, yet they can all be expressed in the same mathematical formulation. We have a signal $\mathbf{x}$ of length N that is sparse or highly compressible; i.e., it consists of k significant entries ("heavy hitters") which we denote by $\mathbf{x}_k$ while the rest of the entries are essentially negligible. We wish to acquire a small amount information (commensurate with the sparsity) about this signal in a linear, non-adaptive fashion and then use that information to quickly recover the significant entries. In a data stream setting, our signal is the distribution of items seen, while in biological group testing, the signal is proportional to the binding affinity of each drug compound (or the expression level of a gene in a particular organism). We want to recover the identities and values only of the heavy hitters which we denote by $\mathbf{x}_k$, as the rest of the signal is not of interest. Mathematically, we have a signal $\mathbf{x}$ and an m-by-N measurement matrix $\mathbf{\Phi}$ with which we acquire measurements $\boldsymbol{y} = \mathbf{\Phi}\mathbf{x}$, and, from these measurements $\boldsymbol{y}$, we wish to recover $\widehat{\mathbf{x}}$, with $O(k)$ entries, such that

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \leq C \|\mathbf{x} - \mathbf{x}_k\|_2 .$$

Gilbert is with the Department of Mathematics, The University of Michigan at Ann Arbor. E-mail: annacg@umich.edu. Li is with the Department of Electrical Engineering and Computer Science, The University of Michigan at Ann Arbor. E-mail: leeyi@umich.edu. Porat is with the Department of Computer Science, Bar-Ilan University. E-mail: porately@cs.biu.ac.il. Strauss is with the Department of Mathematics and the Department of Electrical Engineering and Computer Science, The University of Michigan at Ann Arbor. E-mail: martinjs@umich.edu.

Paper	No. Measurements	Encode time	Column sparsity/ Update time	Decode time	Approx. error
[8, 3]	$k\log(N/k)$	$Nk\log(N/k)$	$k\log(N/k)$	$\geq N$	$\ell_2 \leq (1/\sqrt{k})\ell_1$
[4, 7]	$k\log^c N$	$N\log^c N$	$\log^c N$	$k\log^c N$	$\ell_2 \leq C\ell_2$
[6]	$k\log^c N$	$N\log^c N$	$\log^c N$	$k\log^c N$	$\ell_1 \leq C\ell_1$
This paper	$k\log(N/k)$	$N\log^c N$	$\log^c N$	$k\log^c N$	$\ell_2 \leq C\ell_2$

FIGURE 1. Summary of the best previous results and the result obtained in this paper.

Our goal, which we achieve up to constant or log factors in the various criteria, is to design the measurement matrix $\mathbf{\Phi}$ and the decoding algorithm in an optimal fashion: (i) we take as few measurements as possible $m = O(k\log(N/k))$, (ii) the decoding algorithm runs in *sublinear* time $O(k\log(N/k))$, and (iii) the encoding and update times are optimal $O(N\log(N/k))$ and $O(k\log(N/k))$, respectively. In order to achieve this, our algorithm is a randomized algorithm; i.e., we specify a distribution on the measurement matrix $\mathbf{\Phi}$ and we guarantee that, for each signal, the algorithm recovers a good approximation with high probability over the choice of matrix.

In the above applications, it is important both to take as few measurements as possible and to recover the heavy hitters extremely efficiently. Measurements correspond to physical resources (e.g., memory in data stream monitoring devices, number of screens in biological applications) and reducing the number of necessary measurements is critical these problems. In addition, these applications require efficient recovery of the heavy hitters—we test many biological compounds at once, we want to quickly identify the positions of entities in a sensor network, and we cannot afford to spend computation time proportional to the size of the distribution in a data stream application. Furthermore, Do Ba, et al. [2] give a lower bound on the number of measurements for sparse recovery $\Omega(k\log(N/k))$. There are polynomial time algorithms [13, 3, 12] meet this lower bound, both with high probability for each signal and the stronger setting, with high probability for all signals[1]. Previous sublinear time algorithms, whether in the "for each" model [4, 7] or in the "for all" model [11], however, used several additional factors of $\log(N)$ measurements. We summarize the previous sublinear algorithms in the "for each" signal model in Figure 1. The column sparsity denotes how many 1s there are per column of the measurement matrix and determines both the decoding and measurement update time and, for readability, we suppress $O(\cdot)$. The approximation error signifies the metric we use to evaluate the output; either the ℓ_2 or ℓ_1 metric. In this paper, we focus on the ℓ_2 metric.

We give a joint distribution over measurement matrices and sublinear time recovery algorithms that meet this lower bound (up to constant factors) in terms of the number of measurements and are within $\log(k)$ factors of optimal in the running time and the sparsity of the measurement matrix.

Theorem 1. *There is a joint distribution on matrices and algorithms, with suitable instantiations of anonymous constant factors, such that, given measurements $\mathbf{\Phi x} = \boldsymbol{y}$, the algorithm returns $\widehat{x}$ and approximation error*

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \leq 2\,\|\nu_1\|_2$$

with probability 3/4. *The algorithm runs in time $O(k\log^c(N))$ and $\mathbf{\Phi}$ has $O(k\log(N/k))$ rows.*

Furthermore, our algorithm is a fully polynomial randomized approximation scheme.

Theorem 2. *There is a joint distribution on matrices and algorithms, with suitable instantiations of anonymous constant factors (that may depend on ϵ, such that, given measurements $\mathbf{\Phi x} = \boldsymbol{y}$, the algorithm returns $\widehat{x}$ and approximation error*

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \leq (1+\epsilon)\,\|\nu_1\|_2$$

[1]albeit with different error guarantees and different column sparsity depending on the error metric.

with probability 3/4. *The algorithm runs in time* $O((k/\epsilon)\log^c(N))$ *and* $\mathbf{\Phi}$ *has* $O((k/\epsilon)\log(N/k))$ *rows.*

Finally, our result is robust to corruption of the measurements by an arbitrary noise vector ν_2, which is an important feature for such applications as high throughput screening and other physical measurement systems. (It is less critical for digital measurement systems that monitor data streams in which measurement corruption is less likely.) When $\nu_2 \neq 0$, our error dependence is on ν_2 is suboptimal by the factor $\log(k)$ (improvable to $\log^{1/2+o(1)} k$). Equivalently, we can use $\log(k)$ times more measurements to restore optimality.

Theorem 3. *There is a joint distribution on matrices and algorithms, with suitable instantiations of anonymous constant factors (that may depend on* ϵ*), such that, given measurements* $\mathbf{\Phi x} + \nu_2 = \boldsymbol{y} + \nu_2$*, the algorithm returns* $\widehat{x}$ *and approximation error*

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \leq (1+\epsilon)\|\nu_1\|_2 + \epsilon \log(k) \frac{\|\nu_2\|_2}{\|\Phi\|_{2\rightsquigarrow 2}}$$

with probability 3/4. *The algorithm runs in time* $O((k/\epsilon)\log^c(N))$ *and* $\mathbf{\Phi}$ *has* $O(k/\epsilon \log(N/k))$ *rows.*

Previous sublinear algorithms begin with the observation that if a signal consists of a single heavy hitter, then the trivial encoding of the positions 1 through N with $\log(N)$ bits, referred to as a bit tester, can identify the position of the heavy hitter. The second observation is that a number of hash functions drawn at random from a hash family are sufficient to isolate enough of the heavy hitters, which can then be identified by the bit tester. Depending on the type of error metric desired, the hashing matrix is pre-multiplied by random ± 1 vectors (for the ℓ_2 metric) in order to estimate the signal values. In this case, the measurements are referred to as the Count Sketch in the data stream literature [4] and, without the premultiplication, the measurements are referred to as Count Median [6, 7] and give $\ell_1 \leq C\ell_1$ error guarantees. In addition, the sublinear algorithms are typically greedy, iterative algorithms that recover portions of the heavy hitters with each iteration or that recover portions of the ℓ_2 (or ℓ_1) energy of the residual signal.

We build upon the Count Sketch design but incorporate the following algorithmic innovations to ensure an optimal number of measurements:

- With a random assignment of N signal positions to $O(k)$ measurements, we need to encode only $O(N/k)$ positions, rather than N as in the previous approaches. So, we can reduce the domain size which we encode.
- We use a good error-correcting code (rather than the trivial identity code of the bit tester).
- Our algorithm is an iterative algorithm but maintains a *compound* invariant: the number of un-discovered heavy hitters decreases at each iteration while, simultaneously, the required error tolerance and failure probability become more stringent. Because there are fewer heavy hitters to find at each stage, we can use more measurements to meet more stringent guarantees.

In Section 2 we detail the matrix algebra we use to describe the measurement matrix distribution which we cover in Section 3, along with the decoding algorithm. In Section 4, we analyze the foregoing recovery system.

2. Preliminaries

2.1. **Vectors.** Let $\mathbf{x}$ denote a vector of length N. For each $k \leq N$, let $\mathbf{x}_k$ denote either the usual k'th component of $\mathbf{x}$ or the signal of length N consisting of the j largest-magnitude terms in $\mathbf{x}$; it will be clear from context. The signal $\mathbf{x}_k$ is the best k-term representation of $\mathbf{x}$. The energy of a signal $\mathbf{x}$ is $\|\mathbf{x}\|_2^2 = \sum_{i=1}^N |\mathbf{x}_i|^2$.

operator	name	input	output dimensions and construction
$\oplus_r$	row direct sum	$\boldsymbol{A}$: $r_1 \times N$ $\boldsymbol{B}$: $r_2 \times N$	$\boldsymbol{M}$: $(r_1+r_2)\times N$ $\boldsymbol{M}_{i,j} = \begin{cases} \boldsymbol{A}_{i,j}, & 1 \le i \le r_1 \\ \boldsymbol{B}_{i-r_1,j}, & 1+r_1 \le i \le r_2 \end{cases}$
$\odot$	element-wise product	$\boldsymbol{A}$: $r \times N$ $\boldsymbol{B}$: $r \times N$	$\boldsymbol{M}$: $r \times N$ $\boldsymbol{M}_{i,j} = \boldsymbol{A}_{i,j}\boldsymbol{B}_{i,j}$
$\ltimes_{\mathrm{r}}$	semi-direct product	$\boldsymbol{A}$: $r_1 \times N$ $\boldsymbol{B}$: $r_2 \times h$	$\boldsymbol{M}$: $(r_1 r_2)\times N$ $\boldsymbol{M}_{i+(k-1)r_2,\ell} = \begin{cases} 0, & \boldsymbol{A}_{k,\ell} = 0 \\ \boldsymbol{A}_{k,\ell}\boldsymbol{B}_{i,j}, & \boldsymbol{A}_{k,\ell} = \ j\text{th nonzero in row } \ell \end{cases}$

FIGURE 2. Matrix algebra used in constructing an overall measurement matrix. The last column contains both the output dimensions of the matrix operation and its construction formula.

2.2. **Matrices.** In order to construct the overall measurement matrix, we form a number of different types of combinations of constituent matrices and to facilitate our description, we summarize our matrix operations in Table 2. The matrices that result from all of our matrix operations have N columns and, with the exception of the semi-direct product of two matrices $\ltimes_{\mathrm{r}}$, all operations are performed on matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ with N columns. A full description can be found in the Appendix.

3. Sparse recovery system

In this section, we specify the measurement matrix and detail the decoding algorithm.

3.1. **Measurement matrix.** The overall measurement matrix, $\boldsymbol{\Phi}$, is a multi-layered matrix with entries in $\{-1, 0, +1\}$. At the highest level, $\boldsymbol{\Phi}$ consists of a random permutation matrix $\boldsymbol{P}$ left-multiplying the row direct sum of $(\lg(k))$ summands, $\boldsymbol{\Phi}^{(j)}$, each of which is used in a separate iteration of the decoding algorithm. Each summand $\boldsymbol{\Phi}^{(j)}$ is the row direct sum of two separate matrices, an *identification* matrix, $\boldsymbol{D}^{(j)}$, and an *estimation* matrix, $\boldsymbol{E}^{(j)}$.

$$\boldsymbol{\Phi} = \boldsymbol{P}\begin{bmatrix} \boldsymbol{\Phi}^{(1)} \\ \hline \boldsymbol{\Phi}^{(2)} \\ \hline \vdots \\ \hline \boldsymbol{\Phi}^{(\lg(k))} \end{bmatrix} \qquad \text{where } \boldsymbol{\Phi}^{(j)} = \boldsymbol{E}^{(j)} \oplus_{\mathrm{r}} \boldsymbol{D}^{(j)}.$$

In iteration j, the identification matrix $\boldsymbol{D}^{(j)}$ consists of the row direct sum of $O(j)$ matrices, all chosen independently from the same distribution. We construct the distribution $(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{B}^{(j)}) \odot \boldsymbol{S}^{(j)}$ as follows:

- For $j = 1, 2, \ldots, \lg(k)$, the matrix $\boldsymbol{B}^{(j)}$ is a Bernoulli matrix with dimensions kc^j-by-N, where c is an appropriate constant $1/2 < c < 1$. Each entry is 1 with probability $\Theta\left(1/(kc^j)\right)$ and zero otherwise. Each row is a pairwise independent family and the set of row seeds is fully independent.
- The matrix $\boldsymbol{C}^{(j)}$ is an encoding of positions by an error-correcting code with constant rate and relative distance. That is, fix an error-correcting code and encoding/decoding algorithm that encodes messages of $\Theta(\log\log N)$ bits into longer codewords, also of length $\Theta(\log\log N)$, and can correct a constant fraction of errors. The i'th column of $\boldsymbol{C}^{(j)}$ is the direct sum of $\Theta(\log\log N)$ copies of 1 with the direct sum of $E(i_1), E(i_2), \ldots$, where $i_1, i_2, \ldots$ are blocks of $O(\log\log N)$ bits each whose concatenation is the binary expansion of i and $E(\cdot)$ is the encoding function for the error-correcting code. The number of columns in $\boldsymbol{C}^{(j)}$ matches the maximum number of non-zeros in $\boldsymbol{B}^{(j)}$, which is approximately the

expected number, $\Theta\left(c^j N/k\right)$, where $c < 1$. The number of rows in $\boldsymbol{C}^{(j)}$ is the logarithm of the number of columns, since the process of breaking the binary expansion of index i into blocks has rate 1 and encoding by $E(\cdot)$ has constant rate.

- The matrix $\boldsymbol{S}^{(j)}$ is a pseudorandom sign-flip matrix. Each row is a pairwise independent family of uniform ± 1-valued random variables. The sequence of seeds for the rows is a fully independent family. The size of $\boldsymbol{S}^{(j)}$ matches the size of $\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{B}^{(j)}$.

Note that error correcting encoding often is accomplished by a matrix-vector product, but we are *not* encoding a linear error-correcting code by the usual generator matrix process. Rather, our matrix explicitly lists all the codewords. The code may be non-linear.

The identification matrix at iteration j is of the form

$$\boldsymbol{D}^{(j)} = \begin{bmatrix} \left[(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{B}^{(j)}) \odot \boldsymbol{S}^{(j)}\right]_1 \\ \hline \cdots \\ \hline \left[(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{B}^{(j)}) \odot \boldsymbol{S}^{(j)}\right]_{O(j)} \end{bmatrix}.$$

In iteration j, the estimation matrix $\boldsymbol{E}^{(j)}$ consists of the direct sum of $O(j)$ matrices, all chosen independently from the same distribution, $\boldsymbol{B'}^{(j)} \odot \boldsymbol{S'}^{(j)}$, so that the estimation matrix at iteration j is of the form

$$\boldsymbol{E}^{(j)} = \begin{bmatrix} \left[\boldsymbol{B'}^{(j)} \odot \boldsymbol{S'}^{(j)}\right]_1 \\ \hline \cdots \\ \hline \left[\boldsymbol{B'}^{(j)} \odot \boldsymbol{S'}^{(j)}\right]_{O(j)} \end{bmatrix}.$$

The construction of the distribution is similar to that of the identification matrix, but omits the error-correcting code and uses different constant factors, etc., for the number of rows compared with the analogues in the identification matrix.

- The matrix $\boldsymbol{B'}^{(j)}$ is Bernoulli with dimensions $O(kc^j)$-by-N, for appropriate c, $1/2 < c < 1$. Each entry is 1 with probability $\Theta\left(1/(kc^j)\right)$ and zero otherwise. Each row is a pairwise independent family and the set of seeds is fully independent.
- The matrix $\boldsymbol{S'}^{(j)}$ is a pseudorandom sign-flip matrix of the same dimension as $\boldsymbol{B'}^{(j)}$. Each row of $\boldsymbol{S'}^{(j)}$ is a pairwise independent family of uniform ± 1-valued random variables. The sequence of seeds for the rows is a fully independent family.

3.2. **Measurements.** The overall form of the measurements mirrors the structure of the measurement matrices. We do not, however, use all of the measurements in the same fashion. In iteration j of the algorithm, we use the measurements $\boldsymbol{y}^{(j)} = \boldsymbol{\Phi}^{(j)}\mathbf{x}$. As the matrix $\boldsymbol{\Phi}^{(j)} = \boldsymbol{E}^{(j)} \oplus_{\mathrm{r}} \boldsymbol{D}^{(j)}$, we have a portion of the measurements $\boldsymbol{w}^{(j)} = \boldsymbol{D}^{(j)}\mathbf{x}$ that we use for identification and a portion $\boldsymbol{z}^{(j)} = \boldsymbol{E}^{(j)}\mathbf{x}$ that we use for estimation. The $\boldsymbol{w}^{(j)}$ portion is further decomposed into measurements $[\boldsymbol{v}^{(j)}, \boldsymbol{u}^{(j)}]$ corresponding to the run of $O(\log \log N)$ 1's in $\boldsymbol{C}^{(j)}$ and measurements corresponding to each of the blocks in the error-correcting code. There are $O(j)$ i.i.d. repetitions of everything at iteration j.

3.3. **Decoding.** The decoding algorithm is shown in Figure 3 in the Appendix.

4. Analysis

In this section we analyze the decoding algorithm for correctness and efficiency.

4.1. **Correctness.** Let $\mathbf{x} = \mathbf{x}_k + \nu_1$ where we assume $\mathbf{x}$ is normalized so that $\|\nu_1\|_2 = 1$ and $\mathbf{x}_k$ is the vector $\mathbf{x}$ with all but the largest-magnitude k entries zeroed out. Our goal is to guarantee an approximation $\widehat{\mathbf{x}}$ with approximation error $\|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \leq (1+\epsilon)\|\nu_1\|_2 + \epsilon\|\nu_2\|_2$. But observe that ν_2 is a different type of object from $\mathbf{x}$ or $\widehat{\mathbf{x}}$; ν_2 is added to $\mathbf{\Phi x}$. For the main theorem to make sense, therefore, we need to normalize $\mathbf{\Phi}$. We discuss this now.

Observe that the matrix $\mathbf{\Phi}$ can be scaled up by an arbitrary constant factor $c > 1$ which can be undone by the decoding algorithm: Let $\mathcal{D}'$ be a new decoding algorithm that calls the old decoding algorithm $\mathcal{D}$ as follows: $\mathcal{D}'(\boldsymbol{y}) = \mathcal{D}\left(\frac{1}{c}\boldsymbol{y}\right)$, so that $\mathcal{D}'(c\mathbf{\Phi x} + \nu_2) = \mathcal{D}\left(\mathbf{\Phi x} + \frac{1}{c}\nu_2\right)$. Thus we can *reduce* the effect of ν_2 by an arbitrary factor c and so citing performance in terms of $\|\nu_2\|$ alone is not sensible. Note also that ν_2 and $\mathbf{x}$ are different types of objects; $\mathbf{\Phi}$, as an operator, takes an object of the type of $\mathbf{x}$ and produces an object of the type of ν_2. We will stipulate that the appropriate norm of $\mathbf{\Phi}$ be bounded by 1, in order to make our results quantitatively comparable with others. Our error guarantee is in ℓ_2 norm, so we should use a 2-operator norm; i.e.,, $\max \|\mathbf{\Phi}\boldsymbol{x}\|_2$ over $\boldsymbol{x}$ with $\|\boldsymbol{x}\|_2 = 1$. But our algorithm's guarantee is in the "for each" signal model, so we need to modify the norm slightly.

Definition 4. *The $\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}$ norm of a randomly-constructed matrix $\mathbf{\Phi}$ is $\max_{\boldsymbol{x}} \mathbb{E}\left[\frac{\|\mathbf{\Phi}\boldsymbol{x}\|_2}{\boldsymbol{x}}\right]$. the smallest M such that, for all $\boldsymbol{x}$ with $\|\boldsymbol{x}\|_2 = 1$, we have $\|\mathbf{\Phi}\boldsymbol{x}\|_2 < M$ except with probability 1/4.*

Now we bound $\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}$. Each row ρ of a Bernoulli(p) matrix with sign flips, $\boldsymbol{B}\odot\boldsymbol{S}$, satisfies $\mathbb{E}[|\rho\boldsymbol{x}|^2] = p\|x\|_2^2$. So $1/p$ such rows satisfy $\|(\boldsymbol{B}\odot\boldsymbol{S})\boldsymbol{x}\|_2^2 \leq O\left(\|\boldsymbol{x}\|_2^2\right)$. Our matrix $\mathbf{\Phi}$ repeats the above j times in the j'th iteration, $j \leq \log_2(k)$, and combines it with an error-correcting code matrix of $\Theta(\log(N/k))$ dense rows. It follows that

$$\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}^2 = O(\log^2(k)\log(N/k)).$$

We are ready to state the main theorem.

Theorem 3 *Consider the matrices in Section 3.1 and the algorithms in Section 3.3 (that share randomness with the matrices). The joint distribution on those matrices and algorithms, with suitable instantiations of anonymous constant factors (that may depend on ϵ), are such that, given measurements $\mathbf{\Phi x} + \nu_2 = \boldsymbol{y} + \nu_2$, the algorithm returns $\widehat{\mathbf{x}}$ with approximation error*

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2 \leq (1+\epsilon)\|\nu_1\|_2 + \epsilon\log(k)\frac{\|\nu_2\|_2}{\|\Phi\|_{2\rightsquigarrow 2}}$$

with probability 3/4. The algorithm runs in time $k\log^c N$ and $\mathbf{\Phi}$ has $O(k\log(N/k))$ rows.

In this extended abstract, we give the proof only for $\epsilon = 1$. Our results generalize in a straightforward way for general $\epsilon > 0$ (roughly, by replacing k with k/ϵ at the appropriate places in the proof) and the number of measurements is essentially optimal in ϵ. Because our approach builds upon the Count Sketch approach in [4], we omit the proof of intermediary steps that have appeared earlier in the literature.

We maintain the following invariant. At the beginning of iteration j, the residual signal has the form

$$\text{(Loop Invariant)} \qquad \boldsymbol{r}^{(j)} = \mathbf{x}^{(j)} + \nu_1^{(j)} \quad \text{with } \left\|\mathbf{x}^{(j)}\right\|_0 \leq \frac{k}{2^j}, \text{ and } \left\|\nu_1^{(j)}\right\|_2 \leq 2 - \left(\frac{3}{4}\right)^j$$

except with probability $\frac{1}{4}(1 - (\frac{1}{2})^j)$, where $\|\cdot\|_0$ is the number of non-zero entries. The vector $\mathbf{x}^{(j)}$ consists of residual elements of $\mathbf{x}_k$. Clearly, maintaining the invariant is sufficient to prove the overall result. In order to show that the algorithm maintains the loop invariant, we demonstrate the following claim.

Claim 1. *Let $\boldsymbol{b}^{(j)}$ be the vector we recover at iteration j.*

- *The vector $\boldsymbol{b}^{(j)}$ contains all but at most $\frac{1}{4}\frac{k}{2^j}$ residual elements of $\mathbf{x}_k^{(j)}$, with "good" estimates.*
- *The vector $\boldsymbol{b}^{(j)}$ contains at most $\frac{1}{4}\frac{k}{2^j}$ residual elements of $\mathbf{x}_k$ with "bad" estimates.*
- *The total sum square error over all "good" estimates is at most*

$$\left[2-\left(\frac{3}{4}\right)^{j+1}\right]-\left[2-\left(\frac{3}{4}\right)^{j}\right]=\frac{1}{4}\left(\frac{3}{4}\right)^{j}.$$

Proof. To simplify notation, let T be the set of un-recovered elements of $\mathbf{x}_k$ at iteration j; i.e., the support of $\mathbf{x}^{(j)}$. We know that $|T| \le k/2^j$. The proof proceeds in three steps.

Step 1. Isolate heavy hitters with little noise. Consider the action of a Bernoulli sign-flip matrix $\boldsymbol{B}\odot\boldsymbol{S}$ with $O(k/2^j)$ rows. From previous work [4, 1], it follows that, if constant factors parametrizing the matrices are chosen properly,

Lemma 5. *For each row ρ of $\boldsymbol{B}$, the following holds with probability $\Omega(1)$:*

- *There is exactly one element t of T "hashed" by $\boldsymbol{B}$; i.e., there is exactly one $t \in T$ with $\rho_t = 1$.*
- *There are $O(N \cdot 2^j/k)$ total positions (out of N) hashed by $\boldsymbol{B}$.*
- *The dot product $(\rho\odot\boldsymbol{S})\boldsymbol{r}^{(j)}$ is $\boldsymbol{S}_t\boldsymbol{r}_t^{(j)} \pm O\left(\frac{2^j}{k}\left\|\nu_1^{(j)}\right\|_2\right)$.*

Proof. (Sketch.) For intuition, note that the estimator $\boldsymbol{S}_t(\rho\odot\boldsymbol{S})\boldsymbol{r}^{(j)}$ is a random variable with mean $\boldsymbol{r}_t^{(j)}$ and variance $\left\|\nu_1^{(j)}\right\|_2^2$. Then the third claim and the first two claims assert that the expected behavior happens with probability $\Omega(1)$. □

In our matrix $\boldsymbol{B}^{(j)}$, the number of rows is not $k/2^j$ but kc^j for some c, $1/2 < c < 1$. Take $c = 2/3$. We obtain a stronger conclusion to the lemma. The dot product $(\rho\odot\boldsymbol{S})\boldsymbol{r}^{(j)}$ is

$$\boldsymbol{S}_t\boldsymbol{r}_t^{(j)} \pm O\left(\frac{1}{k(2/3)^j}\left\|\nu_1^{(j)}\right\|_2\right) = \boldsymbol{S}_t\boldsymbol{r}_t^{(j)} \pm \frac{1}{8}\left((3/4)^j\frac{2^j}{k}\left\|\nu_1^{(j)}\right\|_2\right),$$

provided constants are chosen properly. Our lone hashed heavy hitter t will dominate the dot product provided

$$\left|\boldsymbol{r}_t^{(j)}\right| \ge \frac{1}{8}\left((3/4)^j\frac{2^j}{k}\left\|\nu_1^{(j)}\right\|_2\right).$$

We show in the remaining steps that we can likely recover such heavy hitters; i.e., IDENTIFY identifies them and ESTIMATE returns a good estimate of their values. There are at most $(k/2^j)$ heavy hitters of magnitude less than $\frac{1}{8}\left((3/4)^j\frac{2^j}{k}\left\|\nu_1^{(j)}\right\|_2\right)$ which we will not be able to identify nor to estimate but they contribute a total of $\frac{1}{8}\left((3/4)^j\left\|\nu_1^{(j)}\right\|_2\right)$ noise energy to the residual for the next round (which still meets our invariant).

Step 2. Identify heavy hitters with little noise. Next, we show how to identify t. Since there are $N/k^{\Theta(1)}$ positions hashed by $\boldsymbol{B}^{(j)}$, we need to learn the $O(\log(N/k))$ bits describing t in this context. Previous sublinear algorithms [7, 11] used a trivial error correcting code, in which the t'th column was simply the binary expansion of t in direct sum with a single 1. Thus, if the signal consists of $\mathbf{x}_t$ in the t'th position and zeros elsewhere, we would learn $\mathbf{x}_t$ and $\mathbf{x}_t$ times the binary expansion of t (the latter interpreted as a string of 0's and 1's as real numbers). These algorithms require strict control on the failure probability of each measurement in order to use such a trivial encoding. In our case, each measurement succeeds only with probability $\Omega(1)$ and, generally, fails with probability $\Omega(1)$. So we need to use a more powerful error correcting code and a more reliable estimate of $|x_t|$.

To get a reliable estimate of $|\mathbf{x}_t|$, we use the $b = \Theta(\log\log N)$-parallel repetition code of all 1s. That is, we get b independent measurements of $|\mathbf{x}_t|$ and we decode by taking the median. Let p denote the success probability of each individual measurement. Then we expect the fraction p to be approximately correct estimates of $|\mathbf{x}_t|$, we achieve close to the expectation, and we can arrange that $p > 1/2$. It follows that the median is approximately correct. We use this value to threshold the subsequent measurements (i.e., the bits in the encoding) to 0/1 values.

Now, let us consider these bit estimates. In a single error-correcting code block of $b = \Theta(\log\log N)$ measurements, we will get close to the expected number, bp, of successful measurements, except with probability $1/\log(N)$, using the Chernoff bound. In the favorable case, we get a number of failures less than the (properly chosen) distance of the error-correcting code and we can recover the block using standard nearest-neighbor decoding. The number of error-correcting code blocks associated with t is $O(\log(N/k)/\log\log N) \leq O(\log N)$, so we can take a union bound over all blocks and conclude that we recover t with probability $\Omega(1)$. The invariant requires that the failure probability decrease with j. Because the algorithm takes $O(j)$ parallel independent repetitions, we guarantee that the failure probability decreases with j by taking the union over the repetitions.

We summarize these discussions in the following lemma. We refer to these heavy hitters in the list Λ as the j-large heavy hitters.

Lemma 6. IDENTIFY *returns a set* Λ *of signal positions that contains at least* $3/4$ *of the heavy hitters in* T, $|T| \leq k/2^j$, *that have magnitude at least* $\frac{1}{8}\left((3/4)^j \frac{2^j}{k} \left\|\nu_1^{(j)}\right\|_2\right)$.

We also observe that our analysis is consistent with the bounds we give on the additional measurement noise ν_2. The permutation matrix $\boldsymbol{P}$ in $\boldsymbol{\Phi}$ is applied before ν_2 is added and then $\boldsymbol{P}^{-1}$ is applied after ν_2 by the decoding algorithm. It follows that we can assume ν_2 is permuted at random and, therefore, by Markov's inequality, each measurement gets at most an amount of noise energy proportional to its fair share of $\|\nu_2\|_2^2$. Thus, If there are $m = \Theta(k\log N/k)$ measurements, each measurement gets $\frac{\|\nu_2\|_2^2}{m}$ noise energy and identification succeeds anyway provided the lone heavy hitter t in that bucket has square magnitude at least $\frac{\|\nu_2\|_2^2}{m}$, so the at most k smaller heavy hitters, that we may miss, together contribute energy $\frac{k\|\nu_2\|_2^2}{m} = O\left(\frac{\|\nu_2\|_2^2}{\log(N/k)}\right)$. If we recall the definition and value of $\|\boldsymbol{\Phi}\|_{2\rightsquigarrow 2}$, we see that this error meets our bound.

Step 2. Estimate heavy hitters. Many of the details in this step are similar to those in Lemma 5 (as well as to previous work as the function ESTIMATE is essentially the same as COUNT SKETCH), so we give only a brief summary.

First, we discuss the failure probability of the ESTIMATE procedure. Each estimate is a complete failure with probability $1 - \Omega(1)$ and the total number of identified positions is $O\left(jk\,(2/3)^j\right)$. Because we perform j parallel repetitions in estimation, we can easily arrange to lower that failure probability, so we assume that the failure probability is at most $\Theta\left((3/4)^j\right)$, and that we get approximately the expected number of (nearly) correct estimates. There are $k(2/3)^j$ heavy hitters in Λ, so the expected number of failures is $(1/4)(k/2^j)$. These, along with the at most $1/4(k/2^j)$ missed j-large heavy hitters, will form $\mathbf{x}^{(j+1)}$, the at-most-$k/2^{j+1}$ residual heavy hitters at the next iteration.

In iteration j, IDENTITY returns a list Λ with $k(2/3)^j$ heavy hitter position identified. A group of $k(2/3)^j$ measurements in $\boldsymbol{E}^{(j)}$ yields estimates for the positions in Λ with aggregate ℓ_2 error $\pm O(1)$, additively. An additional $O\left((4/3)^j\right)$ times more measurements, $O(k(8/9)^j)$ in all, improves the estimation error to $(1/8)\,(3/4)^j$, additively. These errors, together with the omitted heavy hitters that are not j-large and $\nu^{(j)}$ form the new noise vector at the next iteration, $\nu^{(j+1)}$.

Finally, consider the effect of ν_2. We would like to argue that, as in the identification step, the noise vector ν_2 is permuted at random and each measurement is corrupted by $\frac{\|\nu_2\|_2^2}{m}$, where $m = \Theta(k \log(N/k))$ is the number of measurements, approximately its fair share of $\|\nu_2\|_2^2$. Unfortunately, the contributions of ν_2 to the various measurements are not independent as ν_2 is permuted, so we cannot use such a simple analysis. Nevertheless, they are negatively correlated and we can achieve the result we want using [9]. The total ℓ_2 squared error of the corruption over all $O(k)$ estimates is $\|\nu_2\|_2^2 / \log(N/k)$, which will meet our bound. That is, since $\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}^2 = O(\log^2(k) \log(N/k))$, the ν_2 contribution to the error is

$$O\left(\frac{\|\nu_2\|_2}{\sqrt{\log N/k}}\right) = O\left(\frac{\log(k)\,\|\nu_2\|_2}{\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}}\right),$$

as claimed, whence we read off the factor, $\log(k)$ (improvable to $\log^{1/2+o(1)} k$), which is directly comparable to other results that scale $\mathbf{\Phi}$ properly.

□

4.2. **Efficiency.**

4.2.1. *Number of Measurements.* The analysis of isolation and estimation matrices are similar; the number of measurements in isolation dominates.

The number of measurements in iteration j is computed as follows. There are $O(j)$ parallel repetitions in iteration j. They each consist of $k(2/3)^j$ measurements arising out of $\boldsymbol{B}^{(j)}$ for identification times $O(\log(N/k))$ measurements for the error correcting code, plus $k(2/3)^j$ times $O((4/3)^j)$ for estimation. This gives

$$\Theta\left(jk\left(\frac{2}{3}\right)^j \log(N/k) + jk\left(\frac{8}{9}\right)^j\right) = k\log(N/k)\left(\frac{8}{9} + o(1)\right)^j.$$

Thus we have a sequence bounded by a geometric sequence with ratio less than 1. The sum, over all j, is $O(k \log(N/k))$.

4.2.2. *Encoding and Update Time.* The encoding time is bounded by N times the number of non-zeros in each column of the measurement matrix. This was analyzed above in Section 4.1; there are $\log^2(k)\log(N/k)$ non-zeros per column, which is suboptimal by the factor $\log^2(k)$. By comparison, some proposed methods use dense matrices, which are suboptimal by the exponentially-larger factor k. This can be improved slightly, as follows. Recall that we used j parallel repetitions in iteration j, $j < \log(k)$, to make the failure probability at iteration be; e.g., 2^{-j}, so the sum over j is bounded. We could instead use failure probability $1/j^2$, so that the sum is still bounded, but the number of parallel repetitions will be $\log(j)$, for $j \leq \log(k)$. This results in $\log(k)\log\log(k)\log(N/k)$ non-zeros per column and ν_2 contribution to the noise equal to $\sqrt{\log(k)\log\log(k)}\frac{\|\nu_2\|_2}{\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}}$.

We can use a pseudorandom number generator such as $i \mapsto \lfloor (ai + b \bmod d)/B \rfloor$ for random a and b, where B is the number of buckets. Then we can, in time $O(1)$, determine into which bucket any i is mapped and determined the i'th element in any bucket.

Another issue is the time to find and to encode (and to decode) the error-correcting code. Observe that the length of the code is $O(\log\log N)$. We can afford time exponential in the length, *i.e.*, time $\log^{O(1)} N$, for finding and decoding the code. These tasks are straightforward in that much time.

4.2.3. *Decoding Time.* As noted above, we can quickly map positions to buckets and find the i'th element in any bucket, and we can quickly decode the error-correcting code. The rest of the claimed runtime is straightforward.

5. Conclusion

In this paper, we construct an approximate sparse recovery system that is essentially optimal: the recovery algorithm is a sublinear algorithm (with near optimal running time), the number of measurements meets a lower bound, and the update time, encode time, and column sparsity are each within log factors of the lower bounds. We conjecture that with a few modifications to the distribution on measurement matrices, we can extend this result to the $\ell_1 \leq C\ell_1$ error metric guarantee. We do not, however, think that this approach can be extended to the "for all" signal model (all current sublinear algorithms use at least one factor $O(\log N)$ additional measurements) and leave open the problem of designing a sublinear time recovery algorithm and a measurement matrix with an optimal number of rows for this setting.

References

[1] N. Alon, Y. Matias, and M. Szegedy. The Space Complexity of Approximating the Frequency Moments. *J. Comput. System Sci.*, 58(1):137–147, 1999.

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

[3] 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.

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

[5] G. Cormode and S. Muthukrishnan. What's hot and what's not: Tracking most frequent items dynamically. In *Proc. ACM Principles of Database Systems*, pages 296–306, 2003.

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

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

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

[9] Devdatt Dubhashi and Volker Priebe Desh Ranjan. Negative dependence through the fkg inequality. In *Research Report MPI-I-96-1-020, Max-Planck-Institut fur Informatik, Saarbrucken*, 1996.

[10] Yaniv Erlich, Kenneth Chang, Assaf Gordon, Roy Ronen, Oron Navon, Michelle Rooks, and Gregory J. Hannon. Dna sudoku—harnessing high-throughput sequencing for multiplexed specimen analysis. *Genome Research*, 19:1243—1253, 2009.

[11] 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.

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

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

[14] Y. H. Zheng, N. P. Pitsianis, and D. J. Brady. Nonadaptive group testing based fiber sensor deployment for multiperson tracking. *IEEE Sensors Journal*, 6(2):490–494, 2006.

[15] Y.H. Zheng, D. J. Brady, M. E. Sullivan, and B. D. Guenther. Fiber-optic localization by geometric space coding with a two-dimensional gray code. *Applied Optics*, 44(20):4306–4314, 2005.

6. Appendix

We have a full description of the matrix algebra defined in Table 2.

- **Row direct sum.** The row direct sum $\boldsymbol{A} \oplus_{\mathrm{r}} \boldsymbol{B}$ is a matrix with N columns that is the vertical concatenation of $\boldsymbol{A}$ and $\boldsymbol{B}$.
- **Element-wise product.** If $\boldsymbol{A}$ and $\boldsymbol{B}$ are both $r \times N$ matrices, then $\boldsymbol{A} \odot \boldsymbol{B}$ is also an $r \times N$ matrix whose (i, j) entry is given by the product of the (i, j) entries in $\boldsymbol{A}$ and $\boldsymbol{B}$.
- **Semi-direct product.** Suppose $\boldsymbol{A}$ is a matrix of r_1 rows (and N columns) in which each row has exactly h non-zeros and $\boldsymbol{B}$ is a matrix of r_2 rows and h columns. Then $\boldsymbol{B} \ltimes_{\mathrm{r}} \boldsymbol{A}$ is the matrix with $r_1 r_2$ rows, in which each non-zero entry a of $\boldsymbol{A}$ is replaced by a times the j'th column of $\boldsymbol{B}$, where a is the j'th non-zero in its row. This matrix construction has the following interpretation. Consider $(\boldsymbol{B} \ltimes_{\mathrm{r}} \boldsymbol{A})\mathbf{x}$ where $\boldsymbol{A}$ consists of a single row, $\boldsymbol{\rho}$, with h non-zeros and $\mathbf{x}$ is a vector of length N. Let $\boldsymbol{y} = \boldsymbol{\rho} \odot \mathbf{x}$ be the element-wise product of ρ

and $\mathbf{x}$. If $\boldsymbol{\rho}$ is 0/1-valued, $\boldsymbol{y}$ picks out a subset of $\mathbf{x}$. We then remove all the positions in $\boldsymbol{y}$ corresponding to zeros in $\boldsymbol{\rho}$, leaving a vector $\boldsymbol{y'}$ of length h. Finally, $(\boldsymbol{B} \ltimes_{\mathrm{r}} \boldsymbol{A})\mathbf{x}$ is simply the matrix-vector product $\boldsymbol{By'}$, which, in turn, can be interpreted as selecting subsets of $\boldsymbol{y}$, and summing them up. Note that we can modify this definition when $\boldsymbol{A}$ has fewer than h non-zeros per row in a straightforward fashion.

RECOVER($\boldsymbol{\Phi}, \boldsymbol{y}$)

Output: $\widehat{x}$ = approximate representation of x

$\boldsymbol{y} = \boldsymbol{P}^{-1}\boldsymbol{y}$
$\boldsymbol{a}^{(0)} = 0$
For $j = 0$ to $O(\log k)$ {
  $\boldsymbol{y} = \boldsymbol{y} - \boldsymbol{P}^{-1}\boldsymbol{\Phi}\boldsymbol{a}^{(j)}$
  split $\boldsymbol{y}^{(j)} = \boldsymbol{w}^{(j)} \oplus_{\mathrm{r}} \boldsymbol{z}^{(j)}$
  $\Lambda =$ IDENTIFY($\boldsymbol{D}^{(j)}, \boldsymbol{w}^{(j)}$)
  $\boldsymbol{b}^{(j)} =$ ESTIMATE($\boldsymbol{E}^{(j)}, \boldsymbol{z}^{(j)}, \Lambda$)
  $\boldsymbol{a}^{(j+1)} = \boldsymbol{a}^{(j)} + \boldsymbol{b}^{(j)}$
}
$\widehat{x} = \boldsymbol{a}^{(j)}$

IDENTIFY($\boldsymbol{D}^{(j)}, \boldsymbol{w}^{(j)}$)

Output: Λ = list of positions

$\Lambda = \emptyset$
Divide $\boldsymbol{w}^{(j)}$ into sections $[\boldsymbol{v}, \boldsymbol{u}]$ of size $O(\log(c^j(N/k)))$
For each section {
  $u = \mathrm{median}(|\boldsymbol{v}_\ell|)$
  For each ℓ // threshold measurements
    $\boldsymbol{u}_\ell = \Theta(\boldsymbol{u}_\ell - u/2)$ // $\Theta(u) = 1$ if $u > 0$, $\Theta(u) = 0$ otherwise
  Divide $\boldsymbol{u}$ into blocks b_i of size $O(\log \log N)$
  For each b_i
    β_i = DECODE(b_i) // using error-correcting code
  $\lambda =$INTEGER($\beta_1, \beta_2, \ldots$) // integer rep'ed by bits $\beta_1, \beta_2, \ldots$
  $\Lambda = \Lambda \cup \{\lambda\}$
}

ESTIMATE($\boldsymbol{E}^{(j)}, \boldsymbol{z}^{(j)}, \Lambda$)

Output: $\boldsymbol{b}$ = vector of positions and values

$\boldsymbol{b} = \emptyset$
For each $\lambda \in \Lambda$
  $\boldsymbol{b}_\lambda = \mathrm{median}_{\ell \text{ s.t.} \boldsymbol{B}^{(j)}_{\ell,\lambda}=1}(\boldsymbol{z}^{(j)}_\ell \boldsymbol{S}^{(j)}_{\ell,\lambda})$

FIGURE 3. Pseudocode for the overall decoding algorithm.