SIAM J. Comput.
Vol. 41, No. 2, pp. 436–453

APPROXIMATE SPARSE RECOVERY: OPTIMIZING TIME AND MEASUREMENTS[*]

ANNA C. GILBERT[†], YI LI[‡], ELY PORAT[§], AND MARTIN J. STRAUSS[¶]

Abstract. A Euclidean *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 $\widehat{\mathbf{x}} = \mathcal{D}(\mathbf{\Phi x})$, 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 $\mathbf{x}$. (The output $\widehat{\mathbf{x}}$ may have more than k terms.) 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 $k \log^{O(1)} N$, matching a lower bound up to a polylog(N) factor. 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 (resilience to noise before and after the measurements). Our encode and update times are optimal up to $\log(k)$ factors. The columns of $\mathbf{\Phi}$ have at most $O(\log^2(k)\log(N/k))$ nonzeros, each of which can be found in constant time. Our full result, a fully polynomial randomized approximation scheme, is as follows. If $\mathbf{x} = \mathbf{x}_k + \nu_1$, where ν_1 and ν_2 (below) are arbitrary vectors (regarded as noise), then setting $\widehat{\mathbf{x}} = \mathcal{D}(\Phi\mathbf{x} + \nu_2)$, and for properly normalized $\mathbf{\Phi}$, we get $|\mathbf{x} - \widehat{\mathbf{x}}|_2^2 \leq (1+\epsilon)\,|\nu_1|_2^2 + \epsilon\,|\nu_2|_2^2$ using $O((k/\epsilon)\log(N/k))$ measurements and $(k/\epsilon)\log^{O(1)}(N)$ time for decoding.

Key words. approximation, embedding, sketching, sparse approximation, sublinear algorithms

AMS subject classifications. 94A12, 68W25, 68W20, 68P30

DOI. 10.1137/100816705

1. Introduction.

Tracking heavy hitters in high-volume, high-speed data streams [5], monitoring changes in data streams [8], designing pooling schemes for biological tests [13] (e.g., high throughput sequencing, testing for genetic markers), localizing sources in sensor networks [19, 20], and combinatorial pattern matching [6] 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 of information (commensurate with the sparsity) about this signal in a linear, nonadaptive 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

[*] Received by the editors December 2, 2010; accepted for publication (in revised form) January 15, 2012; published electronically April 24, 2012. A preliminary version of this paper appeared in *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, 2010, pp. 475–484.
http://www.siam.org/journals/sicomp/41-2/81670.html

[†] Department of Mathematics, University of Michigan, Ann Arbor, MI 48104 (annacg@umich.edu). This author's work was supported in part by DARPA/ONR N66001-08-1-2065.

[‡] Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 (leeyi@umich.edu). This author's work was supported in part by NSF CCF 0743372.

[§] Department of Computer Science, Bar-Ilan University, Ramat Gan 52900, Israel (porately@cs.biu.ac.il).

[¶] Department of Mathematics and the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48104 (martinjs@umich.edu). This author's work was supported in part by NSF CCF 0743372 and DARPA/ONR N66001-08-1-2065.

want to recover the identities and values of only 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 $\mathbf{y} = \mathbf{\Phi x}$, and, from these measurements $\mathbf{y}$, we wish to recover $\widehat{\mathbf{x}}$, with $O(k)$ entries, such that

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

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 minimize the number $m = O(k \log N/k)$ of measurements, (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 $O(\log N/k)$, respectively. In order to achieve this, our algorithm is randomized; 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), or, in medical imaging, to the radiation that a patient receives in a CT scan; thus 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. In several of the applications, such as high throughput screening and other physical measurement systems, it is also important that the result be robust to the corruption of the measurements by an arbitrary noise vector ν_2. (It is less critical for digital measurement systems that monitor data streams in which measurement corruption is less likely.)

1.1. Related work. Do Ba et al. [2] give a lower bound of $\Omega(k \log(N/k))$ for the number of measurements for sparse recovery in a model that is related to ours but different in some important respects. There are polynomial time algorithms [16, 4, 15] meeting 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 [5, 10] or in the "for all" model [14], however, used several additional factors of $\log(N)$ measurements. We summarize some previous algorithms in Table 1.1. The column sparsity denotes how many ones 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 noise column denotes whether the algorithm tolerates postmeasurement noise ν_2. The approximation error signifies the metric we use to evaluate the output; $\ell_p \leq C\ell_q(+\ell_r)$ is shorthand for $\|\mathbf{x} - \widehat{\mathbf{x}}\|_p \leq C \|\mathbf{x} - \mathbf{x}_k\|_q (+C \|\nu_2\|_r)$. (Some previous results that did not directly claim stability with respect to ν_2 can be modified easily to accommodate nonzero ν_2.) It has been shown [7] that to achieve $\ell_2 \leq C\ell_2$ in the "for all" model requires $\Omega(n)$ measurements.

1.2. Our result. We give a sublinear time recovery algorithm and a distribution over normalized measurement matrices that meet the lower bound (up to constant

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

TABLE 1.1

Summary of the best previous results and the result obtained in this paper. The sketch type A refers to the "for all" model: for certain probability, a random measurement matrix works for all signals. Type E refers to "for each" model: for each signal, a random measurement matrix works for certain probability. Some decoding times depend on a parameter $T = \log(\|\mathbf{x}\|_2/\|\mathbf{x}-\mathbf{x}_k\|_2)$. LP denotes the time complexity of solving a linear program. The constants c in different rows can be different.

Paper	For all/ For each	No. measurements	Column sparsity/ update time	Decode time	Approx. error	Noise
[11, 4]	A	$k\log(N/k)$	$k\log(N/k)$	LP	$\ell_2 \leq (1/\sqrt{k})\ell_1 + \ell_2$	Y
[5, 10]	E	$k\log^c N$	$\log^c N$	$k\log^c N$	$\ell_2 \leq C\ell_2$	
[9]	E	$k\log^c N$	$\log^c N$	$k\log^c N$	$\ell_1 \leq C\ell_1$	
[14]	A	$k\log^c N$	$\log^c N$	$k^2\log^c N$	$\ell_2 \leq (1/\sqrt{k})\ell_1$	
[3]	A	$k\log(N/k)$	$\log(N/k)$	LP	$\ell_2 \leq (C/\sqrt{k})\ell_1 + \ell_2$	Y
[15]	A	$k\log(N/k)$	$\log(N/k)$	$k\log(N/k)$	$\ell_1 \leq C\ell_1 + \ell_1$	Y
[16]	A	$k\log(N/k)$	$\log(N/k)$	$Tnk\log(N/k)$	$\ell_2 \leq (C/\sqrt{k})\ell_1 + \ell_2$	Y
This paper	E	$k\log(N/k)$	$\log^c N$	$k\log^c N$	$\ell_2 \leq C\ell_2 + \ell_2$	Y

factors) in terms of the number of measurements and are within $\log^{O(1)} N$ factors of optimal in the running time and $\log^2 k$ in the sparsity of the measurement matrix.

THEOREM 1.1. *There is an algorithm and distribution on matrices $\mathbf{\Phi}$ satisfying $\max_{\mathbf{x}} \mathbb{E}[\|\mathbf{\Phi x}\|_2 / \|\mathbf{x}\|_2] = 1$ such that given $\mathbf{\Phi x} + \nu_2$, the parameters, and a concise description of $\mathbf{\Phi}$, the algorithm returns $\widehat{\mathbf{x}}$ with approximation error $\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2 \leq (1 + \epsilon)\|\nu_1\|_2^2 + \epsilon\|\nu_2\|_2^2$ with probability $3/4$. The algorithm runs in time $k/\epsilon \log^{O(1)} N$ and $\mathbf{\Phi}$ has $O\big(k/\epsilon\log(N/k)\big)$ rows. In expectation, there are $O(\log^2(k)\log(N/k))$ nonzeros in each column of $\mathbf{\Phi}$.*

The approximation $\widehat{\mathbf{x}}$ may have more than k terms. From previous work, e.g., [14], it is known that if

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2 \leq (1 + \epsilon^2)\|\mathbf{x} - \mathbf{x}_k\|_2^2 + \epsilon^2\|\nu_2\|_2^2,$$

then the truncation $\widehat{\mathbf{x}}_k$ of $\widehat{\mathbf{x}}$ to k terms satisfies

$$\|\mathbf{x} - \widehat{\mathbf{x}}_k\|_2^2 \leq (1 + \Theta(\epsilon))\|\mathbf{x} - \mathbf{x}_k\|_2^2 + \epsilon\|\nu_2\|_2^2.$$

So an approximation with exactly k terms is possible, but with cost $1/\epsilon^2$ versus $1/\epsilon$ for the general case.

1.3. Our technical contributions. 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, as in the following:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 7 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 7 \\ 0 \\ 7 \\ 0 \end{pmatrix}.$$

The second observation in previous work is that a number of hash or Bernoulli 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 premultiplied 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 [5] and, without the premultiplication, the measurements are referred to as Count Median [9, 10] 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, where the energy of ℓ_2 norm, where ℓ_p energy of $\mathbf{x}$ is defined as $\|\mathbf{x}\|_p^p$.

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 $O(k)$ subsignals, we need to encode only $O(N/k)$ positions, rather than N as in the previous approaches. Thus 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: in our algorithm, the number of undiscovered 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.

We believe we are the first to consider a "for each" algorithm with postmeasurement noise, ν_2. As we discuss below, we need to give a new definition of the appropriate metric under which to normalize $\mathbf{\Phi}$.

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 kth component of $\mathbf{x}$ or the signal of length N consisting of the k 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 $\|\mathbf{x}\|_2^2 = \sum_{i=1}^N |x_i|^2$.

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 2.1. The matrices that result from all our matrix operations have N columns and, with the exception of the semidirect product of two matrices $\ltimes_{\mathrm{r}}$, all operations are performed on matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ with N columns. The full description of the matrix algebra defined in Table 2.1 is as follows:

- 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 $\boldsymbol{B}$.
- Elementwise 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 $\boldsymbol{B}$.

Table 2.1

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.

Operator	Name	Input	Output dimensions and construction
$\oplus_r$	row direct sum	$\boldsymbol{A}\colon r_1 \times N$ $\boldsymbol{B}\colon r_2 \times N$	$\boldsymbol{M}\colon (r_1 + r_2) \times N$ $\boldsymbol{M}_{i,j} = \begin{cases} \boldsymbol{A}_{i,j}, & 1 \le i \le r_1 \\ \boldsymbol{B}_{i-r_1,j}, & 1 + r_1 \le i \le r_2 \end{cases}$
$\odot$	elementwise product	$\boldsymbol{A}\colon r \times N$ $\boldsymbol{B}\colon r \times N$	$\boldsymbol{M}\colon r \times N$ $\boldsymbol{M}_{i,j} = \boldsymbol{A}_{i,j}\boldsymbol{B}_{i,j}$
$\ltimes_{\mathrm{r}}$	semidirect product	$\boldsymbol{A}\colon r_1 \times N$ $\boldsymbol{B}\colon r_2 \times h$	$\boldsymbol{M}\colon (r_1 r_2) \times N$ $\boldsymbol{M}_{i+(k-1)r_2,\ell} = \begin{cases} 0, & \boldsymbol{A}_{k,\ell} = 0 \\ \boldsymbol{A}_{k,\ell}\boldsymbol{B}_{i,j}, & \boldsymbol{A}_{k,\ell} = j\text{th nonzero in row } \ell \end{cases}$

- Semidirect product. Suppose $\boldsymbol{A}$ is a matrix of r_1 rows (and N columns) in which each row has exactly h nonzeros 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 nonzero entry a of $\boldsymbol{A}$ is replaced by a times the jth column of $\boldsymbol{B}$, where a is the jth nonzero in its row.

  This definition can be modified for our purposes in a straightforward fashion when $\boldsymbol{A}$ has fewer than h nonzeros per row.

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 multilayered. At the highest level, $\boldsymbol{\Phi}$ consists of a random permutation matrix $\boldsymbol{P}$ left-multiplying the row direct sum of $O(\log(k))$ 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, $\boldsymbol{E}^{(j)}$:

$$\boldsymbol{\Phi} = \boldsymbol{P} \left[\begin{array}{c} \boldsymbol{\Phi}^{(1)} \\ \hline \boldsymbol{\Phi}^{(2)} \\ \hline \vdots \\ \hline \boldsymbol{\Phi}^{(\log(k))} \end{array}\right], \qquad \text{where } \boldsymbol{\Phi}^{(j)} = \boldsymbol{E}^{(j)} \oplus_{\mathrm{r}} \boldsymbol{D}^{(j)}.$$

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 that distribution,

$$\frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}} (\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}) \odot \boldsymbol{S}^{(j)},$$

as follows:

- For $j = 1, 2, \ldots, \log k$, the matrix $\boldsymbol{H}^{(j)}$ is a hashing matrix with dimensions $kc^j \times N$, where c in the range $1/2 < c < 1$ will be specified later. Each column has exactly one nonzero, a one, in a uniformly random row. The columns are pairwise independent.

- The matrix $\boldsymbol{C}^{(j)}$ is an encoding of positions by an error-correcting code with constant rate and relative distance, together with several ones. That is, fix an error-correcting code and encoding and decoding algorithms that encode 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. Let $E(\cdot)$ be its encoding function. The ith column of $\boldsymbol{C}^{(j)}$ is the direct sum of $\Theta(\log\log N)$ copies of one with the direct sum of $E(i_1)$, $E(i_2)$, $\dots$, where $i_1, i_2, \dots$ are blocks of $O(\log\log N)$ bits each, whose concatenation is the binary expansion of i. The number of columns in $\boldsymbol{C}^{(j)}$ is the same as the maximum number of nonzeros in $\boldsymbol{H}^{(j)}$, 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 existence of such an error-correcting code can be shown by a simple counting argument. Given a codeword of length $c_2 n$ and a fraction $r < 1/4$, there is a ball of radius $2rc_2n$ about it with volume $\binom{c_2n}{2rc_2n}2^{2rc_2n}$. If no other codeword is in that ball, nearest-neighbor decoding will recover the correct codeword. Assuming that q codewords have disjoint balls about them, the size of their union is at most $q\binom{c_2n}{2rc_2n}2^{2rc_2n}$. As long as this volume is less than the total number of strings of length c_2n (i.e., 2^{c_2n}), there are more potential codewords we can use. If there are 2^{c_1n} messages (each of length c_1n), each of which needs a codeword, it is possible to find enough decodable codewords as long as

$$2^{c_1n}\binom{c_2n}{2rc_2n}2^{2rc_2n} \le 2^{c_2n}.$$

  This relationship holds for appropriately chosen c_1, c_2, and large n. 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 nonlinear.
- 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 $\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}$.

Below, to achieve our claimed runtime, we will construct $\boldsymbol{C}^{(j)}$ and $\boldsymbol{H}^{(j)}$ together. See Figure 3.1 and section 4.2.2.

The identification matrix at iteration j is of the form

$$\boldsymbol{D}^{(j)} = \frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}}\left[\begin{array}{c} \left[(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}) \odot \boldsymbol{S}^{(j)}\right]_1 \\ \hline \vdots \\ \hline \left[(\boldsymbol{C}^{(j)} \ltimes_{\mathrm{r}} \boldsymbol{H}^{(j)}) \odot \boldsymbol{S}^{(j)}\right]_{O(j)} \end{array}\right].$$

In iteration j, the estimation matrix $\boldsymbol{E}^{(j)}$ consists of the direct sum of $O(j + \log(1/\epsilon))$ matrices, all chosen independently from the same distribution, $\frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}}$

$$\boldsymbol{H} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad \boldsymbol{C} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix}$$

The matrix $\boldsymbol{H}$ is formed from hash function h which maps $\langle 8, 0, 3, 6\rangle$ to $\langle 0, 1, 2, 3\rangle$. If ρ is the top row of $\boldsymbol{H}$ and $\boldsymbol{S}$ arbitrary, then

$$(\boldsymbol{C} \ltimes_{\mathrm{r}} \rho) \odot \boldsymbol{S} = \begin{pmatrix} -1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

FIG. 3.1. *Example measurement matrix for identification. Here* $N = 11$, $k = 3$, *and, in the hashing* $h : i \mapsto a + bi \bmod N$, *we have* $a = 1$ *and* $b = 4$, *so that the sequence* $i = \langle 0, 1, 2, 3,\ 4, 5, 6, 7,\ 8, 9, 10\rangle$ *is mapped to* $\langle 1, 5, 9, 2,\ 6, 10, 3, 7,\ 0, 4, 8\rangle$. *The three buckets are* $\{i : 0 \le a + bi < 4\}, \{i : 4 \le a + bi < 8\}$, *and* $\{i : 8 \le a + bi < 11\}$. *We number starting from* 0, *so* $0 \le i < 11$. *In this example, we use two rows of ones and the double repetition code instead of a good code.*

$\boldsymbol{H'}^{(j)} \odot \boldsymbol{S'}^{(j)}$, so that the estimation matrix at iteration j is of the form

$$\boldsymbol{E}^{(j)} = \frac{2^{-\Theta(j)}}{\sqrt{\log(N/k)}} \left[\begin{array}{c} \left[\boldsymbol{H'}^{(j)} \odot \boldsymbol{S'}^{(j)}\right]_1 \\ \hline \vdots \\ \hline \left[\boldsymbol{H'}^{(j)} \odot \boldsymbol{S'}^{(j)}\right]_{O(j+\log(1/\epsilon))} \end{array} \right].$$

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

- The matrix $\boldsymbol{H'}^{(j)}$ is a hashing matrix with dimensions $O(kc^j) \times N$ for appropriate c, $1/2 < c < 1$. Each column has exactly one nonzero, a one, in a uniformly random row. The columns are pairwise independent.
- The matrix $\boldsymbol{S'}^{(j)}$ is a pseudorandom sign-flip matrix of the same dimension as $\boldsymbol{H'}^{(j)}$. 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 fully independent.

3.2. Measurements. The overall form of the measurements mirrors the structure of the measurement matrices. We do not, however, use all the measurements in the same fashion. Upon receiving $\boldsymbol{\Phi}\mathbf{x} + \nu_2$, the algorithm first applies the permutation $\boldsymbol{P}^{-1}$. In iteration j of the algorithm, we use the measurements $\mathbf{y}^{(j)} = \boldsymbol{\Phi}^{(j)}\mathbf{x} + (\boldsymbol{P}^{-1}\nu_2)^{(j)}$. As the matrix $\boldsymbol{\Phi}^{(j)} = \boldsymbol{E}^{(j)} \oplus_{\mathrm{r}} \boldsymbol{D}^{(j)}$, we have a portion of the measurements $\mathbf{w}^{(j)} = \boldsymbol{D}^{(j)}\mathbf{x} + (\boldsymbol{P}^{-1}\nu_2)^{D(j)}$ that we use for identification and a portion $\mathbf{z}^{(j)} = \boldsymbol{E}^{(j)}\mathbf{x} + (\boldsymbol{P}^{-1}\nu_2)^{E(j)}$ that we use for estimation. The $\mathbf{w}^{(j)}$ portion is further decomposed into measurements $[\mathbf{v}^{(j)}, \mathbf{u}^{(j)}]$ corresponding to the run of $O(\log\log N)$ ones in $\boldsymbol{C}^{(j)}$ and measurements corresponding to each of the blocks in the error-correcting

```
                              $\textsc{Recover}(\mathbf{\Phi}, \mathbf{y})$

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

$\mathbf{y} = \boldsymbol{P}^{-1}\mathbf{y}$
$\boldsymbol{a}^{(0)} = 0$
For $j = 0$ to $O(\log k)$ {
        $\mathbf{y} = \mathbf{y} - \boldsymbol{P}^{-1}\mathbf{\Phi}\boldsymbol{a}^{(j)}$
        split $\mathbf{y}^{(j)} = \mathbf{w}^{(j)} \oplus_{\mathrm{r}} \mathbf{z}^{(j)}$              // Recall that $\mathbf{y} = (\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(O(\log k))})$
        $\Lambda = \textsc{Identify}(\boldsymbol{D}^{(j)}, \mathbf{w}^{(j)})$
        $\boldsymbol{b}^{(j)} = \textsc{Estimate}(\boldsymbol{E}^{(j)}, \mathbf{z}^{(j)}, \Lambda)$
        $\boldsymbol{a}^{(j+1)} = \boldsymbol{a}^{(j)} + \boldsymbol{b}^{(j)}$
}
$\widehat{x} = \boldsymbol{a}^{(j)}$
```

```
                              $\textsc{Identify}(\boldsymbol{D}^{(j)}, \mathbf{w}^{(j)})$

Output:  $\Lambda =$ list of positions

$\Lambda = \emptyset$
Divide $\mathbf{w}^{(j)}$ into sections $[\mathbf{v}, \mathbf{u}]$ of size $O(\log(c^j(N/k)))$
For each section {
        $u = \mathrm{median}(|\mathbf{v}_\ell|)$
        For each $\ell$                                           // threshold measurements
                $\mathbf{u}_\ell = H(|\mathbf{u}_\ell| - u/2)$           // $H(u) = 1$ if $u > 0$, $H(u) = 0$ otherwise
        Divide $\mathbf{u}$ into blocks $b_i$ of size $O(\log\log N)$
        For each $b_i$
                $\beta_i$ = $\textsc{Decode}(b_i)$                           // using error-correcting code
        $\lambda = \textsc{Integer}(\beta_1, \beta_2, \ldots)$             // integer represented by bits $\beta_1, \beta_2, \ldots$
        $\lambda = \textsc{Convert}(\lambda)$                      // convert bucket index to signal index
        $\Lambda = \Lambda \cup \{\lambda\}$
}
```

```
                              $\textsc{Estimate}(\boldsymbol{E}^{(j)}, \mathbf{z}^{(j)}, \Lambda)$

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

$\boldsymbol{b} = \emptyset$
For each $\lambda \in \Lambda$
        $\boldsymbol{b}_\lambda = \mathrm{median}_{\ell \text{ s.t.} \boldsymbol{H}^{(j)}_{\ell,\lambda} = 1}(\mathbf{z}^{(j)}_\ell \boldsymbol{S}^{(j)}_{\ell,\lambda})$
For each $\lambda \in \Lambda$
        If $|\boldsymbol{b}_\lambda|$ is not among top $\Theta(k/2^j)$
                $\boldsymbol{b}_\lambda = 0$
```

FIG. 3.2. *Pseudocode for the overall decoding algorithm.*

code. There are $O(j)$ independently and identically distributed (i.i.d.) repetitions in the identification part and $O(j + \log(1/\epsilon))$ repetitions in the estimation part.

3.3. Decoding. The decoding algorithm is shown in Figure 3.2.

4. Analysis.

The overall structure of our algorithm is greedy, similar to other algorithms in the literature. At each iteration, the algorithm recovers some of the signal but introduces errors both through many coefficient estimates that are approximately but not perfectly correct and through a small number of terms that can be arbitrarily bad. The result is called a *residual signal*.

The measurement and runtime costs of the first iteration dominate the combined cost of all the others. In it, we reduce a bound on the number of heavy hitters to recover from k to $k/2$ while increasing the noise energy from 1 to $1 + \epsilon/4$ using $O((k/\epsilon)\log(N/k))$ measurements. In subsequent iterations, the number of heavy hitters is reduced to $k/2^j$, which reduces the leading cost factor from k/ϵ to $2^{-j}k/\epsilon$. This gives the algorithm 2^j times more resources. In particular, the algorithm can tighten the approximation constant from $1 + \epsilon/4$ to $1 + (\epsilon/4)c^j$ for appropriate c in the range $1/2 < c < 1$ at cost factor $(1/c)^j < 2^j$, which is more than paid for by the $2^{-j} < 1$ savings in the leading factor. Similarly, the algorithm can simultaneously afford to have a smaller failure probability at iteration j. With the tightened approximation constant, the algorithm can tolerate additional ν_2 noise in later iterations, which, as we show below, saves resources.

To prove our result formally, we state a loop invariant maintained by our algorithm and prove that this invariant holds in the Loop Invariant Maintenance (LIM) Lemma. We demonstrate how it characterizes a single iteration of the algorithm: (i) how many measurements are used, (ii) how many nonzeros there are in each column of the measurement matrix, (iii) the runtime, and (iv) the properties of the residual. To prove the LIM lemma, we proceed as follows:

- In Claim 2, we explain, structurally, how the conclusions of the lemma are met—what are the sources of errors, etc.
- We then examine the three subroutines in the algorithm: (i) isolating heavy hitters, (ii) identifying them, and (iii) estimating coefficients.
- Finally, we show that the number of measurements used, the sparsity of the measurement matrix, the running time, and the effect of postmeasurement noise are all as claimed in the lemma.

Finally, we discuss normalization of $\mathbf{\Phi}$ and show that it is, indeed, normalized. We conclude by analyzing the correctness and efficiency of the overall algorithm using our results about each iteration.

4.1. Correctness.

Without loss of generality, assume $\|\nu_1\|_2 = \|\nu_2\|_2 = 1$, since our analysis can scale the signal (the algorithm does not need to know the scaling) and, if ν_1 and ν_2 have different energies, we can increase the weaker of the two. Formally, we maintain the following invariant.

CLAIM 1 (loop invariant). *At the beginning of iteration j, the residual signal has the form $\boldsymbol{r}^{(j)} = \sigma^{(j)} + \nu_1^{(j)}$ with*

$$\left\|\sigma^{(j)}\right\|_0 \leq \frac{k}{2^j} \ \text{and} \ \left\|\nu_1^{(j)}\right\|_2^2 \leq 1 + \epsilon\left(1 - \left(\frac{3}{4}\right)^j\right)$$

except with probability $\frac{1}{4}(1 - (\frac{1}{2})^j)$, where $\|\cdot\|_0$ is the number of nonzero entries. Furthermore, the algorithm has computed (the sparse partial representation) $\widehat{\mathbf{x}}^{(j)} = \mathbf{x} - \boldsymbol{r}^{(j)}$.

Clearly, the invariant holds at the start and 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 lemma, which, after proper instantiation of the lemma's variables, can be used to show the invariant is maintained.

4.1.1. Loop invariant maintenance.

LEMMA 4.1 (loop invariant maintenance). *Fix numerical parameters N, ℓ, δ, and η with $\delta > 0$ and $\eta > 1/N$. Let $\boldsymbol{a}$ be a vector of length N that can be written as $\boldsymbol{a} = \sigma + \nu_1$ with $\|\sigma\|_0 \leq \ell$. Let $\mathbf{\Phi}$ be of the form of $O(\log(1/\delta))$ repetitions of*

$(\boldsymbol{C}\ltimes_{\mathrm{r}}\boldsymbol{H})\odot\boldsymbol{S}$ *in row direct sum with* $O(\log 1/(\delta\eta))$ *repetitions of* $\boldsymbol{H'}\odot\boldsymbol{S'}$ *described in section* 3.1, *where* $\boldsymbol{H}$ *and* $\boldsymbol{H'}$ *have* $O(\ell/\eta)$ *rows. Let* ν_2 *be a noise vector, where each component has magnitude at most* $\frac{4}{\sqrt{m}}\|\nu_2\|_2$, *where* m *is the length of* ν_2.

Then, except with probability δ, *given* $\boldsymbol{\Phi}$, $\mathbf{y} = \boldsymbol{\Phi}\boldsymbol{a} + \nu_2$, *and appropriate parameters, the inner loop of the* Recover *algorithm in Figure* 3.2 *recovers* $\boldsymbol{b}$, *which can be written as* $\boldsymbol{b} = \sigma' + \nu_1'$ *with* $\|\sigma'\|_0 \le \ell/2$ *and* $\|\nu_1'\|_2^2 \le (1+\eta)\,\|\nu_1\|_2^2 + \frac{16\eta}{\gamma}\,\|\nu_2\|_2^2$, *where* γ *is the common expected number of nonzeros in each column of* $\boldsymbol{\Phi}$. *Furthermore,*

- *the number of rows in* $\boldsymbol{\Phi}$ *is* $O(\ell/\eta)\log(N/\ell)\log(1/\delta)$,
- *the computation time is* $(\ell/\eta)\log^{O(1)}(N/(\ell\delta\eta))$,
- *the expected number* γ *of non-zeros in each column of* $\boldsymbol{\Phi}$ *is* $O((\log N/\ell)\log(1/\delta))$.

Proof. Much of the algorithm and the analysis are similar to previous work (e.g., [5]), so we sketch the proof, focusing on changes versus previous work. We first address the case $\nu_2 = 0$.

Recall that $\boldsymbol{\Phi}$ works by giving each element of the signal a random sign flip, hashing each item pairwise independently at random to each measurement, and encoding each index by an error-correcting code. We have the next claim.

Claim 2. *Except with probability* $\delta/3$,

- *the vector* $\boldsymbol{b}$ *contains all but at most* $\ell/4$ *terms of* σ, *with "good" estimates;*
- *the vector* $\boldsymbol{b}$ *contains at most* $\ell/4$ *terms with "bad" estimates, i.e., with square error greater than proportional to* $\eta/\ell \cdot \|\nu_1\|_2^2$;
- *the total sum square error over all "good" estimates is at most* $\eta/4$.

Proof. To simplify notation, let T be the set of terms of $\boldsymbol{a}$ that are both among the top ℓ and have energy at least $\frac{\eta}{8\ell}\|\nu_1\|_2^2$. We know that $|T| \le O(\ell)$. We call the elements in T heavy hitters. The proof proceeds in three steps.

Step 1. *Isolate heavy hitters with little noise.* Consider the action of a hashing and sign-flip matrix $\boldsymbol{H}\odot\boldsymbol{S}$ with $O(\ell/\eta)$ rows. From previous work [5, 1], it follows that if constant factors parametrizing the matrices are chosen properly, the next lemma follows.

Lemma 4.2. *For each* $t \in T$, *the following holds with probability* $1 - O(\delta\eta)$:

(a) *The term* t *is hashed by at least one row* ρ *in* $\boldsymbol{H}$.

(b) *There are* $O(\eta N/\ell)$ *total positions (out of* N*) hashed by* ρ.

(c) *The dot product* $(\rho\odot\boldsymbol{s})\boldsymbol{a}$ *is* $\boldsymbol{s}_t\boldsymbol{a}_t \pm O(\sqrt{\frac{\eta}{\ell}}\,\|\nu_1\|_2)$, *where* $\boldsymbol{s}$ *is a sign-flip vector.*

(d) *Every* $t' \in T \setminus \{t\}$ *is not hashed by* ρ.

Proof (sketch). For intuition, note that the estimator $\boldsymbol{s}_t(\rho\odot\boldsymbol{s})\boldsymbol{a}$ is a random variable with mean $\boldsymbol{a}_t$ and variance $\|\nu_1\|_2^2$. Then the claims in the lemma assert that the expected behavior happens, up to constant factors, with probability $\Omega(1)$. The $O(\log 1/(\delta\eta))$ repetitions of $\boldsymbol{H}\odot\boldsymbol{S}$ bring the failure probability down to $O(\delta\eta)$.

In the favorable case, into each row of $\boldsymbol{H}$ is hashed exactly one term of T that dominates the other $\eta N/\ell$ terms hashed into that row. □

Call a row ρ that satisfies the conditions in Lemma 4.2 a good row.

Step 2. *Identify heavy hitters with little noise.* Next, we show how to identify the heavy hitter t in a good row. Since there are $\eta N/\ell$ different positions hashed by $\boldsymbol{H}$, we need to learn the $O(\log(\eta N/\ell))$ bits describing t in this context. Previous sublinear algorithms [10, 14] used a trivial error-correcting code, in which the tth column was simply the binary expansion of t in direct sum with a single 1 for the matrix $\boldsymbol{C}$ in semidirect product with $\boldsymbol{H}$. Thus, if the signal $\mathbf{x}$ consists of $\mathbf{x}_t$ in the tth position and zeros elsewhere, the vector $(\boldsymbol{C}\ltimes_{\mathrm{r}}\boldsymbol{H})\mathbf{x}$ would include $\mathbf{x}_t$ and $\mathbf{x}_t$ times the binary expansion of t (the latter interpreted as a string of zeroes and ones 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 $|\mathbf{x}_t|$.

Recall that we have a portion $\mathbf{w}$ of the measurements that are used for identification and that these are further decomposed into the pieces $[\mathbf{v}, \mathbf{u}]$ that correspond to the parallel repetition of $\Theta(\log\log N)$ ones and to the error-correcting code blocks, respectively. We use the block $\mathbf{v}$ of $b = \Theta(\log\log N)$ independent measurements of $|\mathbf{x}_t|$ to obtain an estimate u of $|\mathbf{x}_t|$ that we use to threshold the subsequent measurements $\mathbf{u}$ to $0/1$ values that correspond to the bits in the encoding of t. Let p denote the success probability of each individual measurement in $\mathbf{v}$. We can arrange that $p > 1 - r$. (Recall that r is the relative distance of the error-correcting code.) Then, we expect the fraction p to be approximately correct estimates of $|\mathbf{x}_t|$, we achieve close to this expected fraction, and the median u over the $\Theta(\log\log N)$ estimates is approximately correct with high probability.

Next, we use the median u to threshold the remaining measurements $\mathbf{u}$ to $0/1$ values. 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 $O(\log(\eta N/\ell)/\log\log N) \leq O(\log N)$, so we can take a union bound over all blocks and conclude that we recover t with probability $\Omega(1)$. Because the algorithm takes $O(\log(1/\delta))$ parallel independent repetitions, we guarantee that the failure probability is δ for each $t \in T$ and we expect $\delta|T| = O(\delta\ell)$ failures, overall. The probability of getting more than $\ell/4$ failures is at most $O(\delta)$.

Step 3. *Estimate heavy hitters.* Many of the details in this step are similar to those in Lemma 4.2 (as well as to previous work as the function Estimate is essentially the same as Count Sketch), so we give only a brief summary.

The error-correcting code is not necessary for estimating the coefficient values and we use a separate set of measurements $\mathbf{z}$ that do not include the coding overhead. As above, random sign flips and hashing into $O(\ell/\eta)$ buckets suffices to isolate a term t so that the remaining terms hashed to t's bucket have expected energy $O(\eta/\ell)$ and realize energy $O(\eta/\ell)$ with constant probability. Another factor $\log 1/(\delta\eta)$ repetitions suffices to make the failure probability $\delta\eta$, so that except with probability δ, we have $O(\eta|\Lambda|) = O(\ell)$ failures overall among the $|\Lambda| = \Theta(\ell/\eta)$ candidates whose coefficients we estimate.

This concludes the proof of the claim. □

Number of measurements. We now consider the number of measurements in the matrix. The hashing matrix $\boldsymbol{H}$ contributes $O(\ell/\eta)$ rows. The constant-rate error-correcting code matrix $\boldsymbol{C}$ contributes an additional factor of $O(\log \eta N/\ell)$ to identify one index out of $\eta N/\ell$. The $O(\log 1/\delta)$ repetitions contribute that additional factor to drive down the overall failure probability of identification from $1 - \Omega(1)$ to δ. The $\boldsymbol{S}$ matrix does not contribute to the number of rows. This gives a product of

$$O((\ell/\eta)(\log(\eta N/\ell)\log 1/\delta))$$

for identification.

Similarly, for estimation, we have $O(\ell/\eta)$ rows for hashing. Since we are estimating coefficients for $O(\ell/\eta)$ candidates and can only afford $O(\ell)$ errors except with probability δ, the Markov inequality requires that each estimate fail with probability at most $\eta\delta$, which contributes the factor $O(\log 1/(\eta\delta))$. Thus we get $O((\ell/\eta)\log 1/(\eta\delta))$ for estimation and

$$O((\ell/\eta)(\log(\eta N/\ell)\log 1/\delta + \log 1/(\eta\delta)),$$

overall. Note that we may assume $\eta > \ell/N$, since otherwise we may use $\ell/\eta > N$ measurements to recover trivially. Thus the overall number of measurements is

$$O((\ell/\eta)(\log(N/\ell)\log 1/\delta)).$$

Number of nonzeros. The expected number of nonzeros in each column of the identification part of $\mathbf{\Phi}$ is $O(1)$ from hashing, times the factor $O(\log \eta N/\ell)$ from the general (dense) error-correcting code, times $O(\log 1/\delta)$ for repetition. Analysis of the estimation part is similar. We get $O(\log(N/\ell)\log 1/\delta)$ nonzeros altogether.

Postmeasurement noise. Finally, consider the effect of ν_2. Suppose there are m rows in $\mathbf{\Phi}$. A careful inspection of the above proof indicates that ν_1 enters only through the expected energy in each bucket, which is $\Theta((\eta/\ell)\left\|\nu_1\right\|_2^2)$; the error-correcting code and parallel repetitions lead to energy $\Theta((\eta/\ell)\left\|\nu_1\right\|_2^2) = \Theta((\gamma/m)\left\|\nu_1\right\|_2^2)$ in each component of $\mathbf{\Phi}\boldsymbol{a}$. The error $(1+\eta)\left\|\nu_1\right\|_2^2$ represents the "inevitable" error $\left\|\nu_1\right\|_2^2$ due to terms outside the top $O(\ell)$ that are not recovered by the algorithm, plus "excess" error $\eta\left\|\nu_1\right\|_2^2$, which is introduced through many small coefficient approximation errors. Since ν_2 does not affect the inevitable error, we can replace $(\gamma/m)\left\|\nu_1\right\|_2^2$ with $(\gamma/m)\left\|\nu_1\right\|_2^2 + (16/m)\left\|\nu_2\right\|_2^2$ when figuring the excess error, giving the claimed result. (Below we will see that γ can be viewed as a normalization factor for $\mathbf{\Phi}$, which makes $\mathbf{\Phi}\nu_1$ and ν_2 comparable.)

The computation time is straightforward. This concludes the proof of Lemma 4.1, the LIM Lemma. □

4.1.2. Normalization of the measurement matrix. Next we consider the normalization of the overall matrix $\mathbf{\Phi}$ from section 3.1. As has been observed [2], $\mathbf{\Phi}$ should be normalized in the setting of $\nu_2 \neq 0$. Otherwise, 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 $\mathcal{D}'(\boldsymbol{y}) = \mathcal{D}(\frac{1}{c}\boldsymbol{y})$, so that $\mathcal{D}'(c\mathbf{\Phi}\mathbf{x} + \nu_2) = \mathcal{D}(\mathbf{\Phi}\mathbf{x} + \frac{1}{c}\nu_2)$. Thus we would be able to *reduce* the effect of ν_2 by an arbitrary factor $c > 1$. In our "for each, $\ell_2 \leq C\ell_2$" model, an appropriate way to normalize $\mathbf{\Phi}$ is as follows.

DEFINITION 4.3. *The* $\left\|\mathbf{\Phi}\right\|_{2\rightsquigarrow 2}$ *norm of a randomly constructed matrix* $\mathbf{\Phi}$ *is*

$$\max_{\boldsymbol{x}\neq 0}\mathbb{E}\left[\frac{\left\|\mathbf{\Phi}\mathbf{x}\right\|_2}{\left\|\mathbf{x}\right\|_2}\right].$$

Note that the usual ℓ^2-operator norm,

$$\left\|\mathbf{\Phi}\right\|_2 = \max_{\mathbf{x}\neq 0}\left[\frac{\left\|\mathbf{\Phi}\mathbf{x}\right\|_2}{\left\|\mathbf{x}\right\|_2}\right],$$

is typically much larger than $\left\|\mathbf{\Phi}\right\|_{2\rightsquigarrow 2}$, which would lead to a much weaker result. But it corresponds to an adversary choosing $\mathbf{x}$ and ν_2 knowing the outcome $\mathbf{\Phi}$, which is counter to the spirit of the "for each" model in previous work. Here we assume the adversary knows the distribution on $\mathbf{\Phi}$, but not the outcome, when choosing $\mathbf{x}$ and ν_2.

Now we bound $\|\mathbf{\Phi}\|_{2\rightsquigarrow 2}$ for our $\mathbf{\Phi}$. It is straightforward to see that this is the maximum expected column ℓ_2 norm. In the jth iteration, there are at most $j\log N/k$ nonzero entries, each of magnitude $\sqrt{\frac{c^j}{\log N/k}}$ for some c in the range $1/2 < c < 1$. It follows that

$$\|\mathbf{\Phi}\|^2_{2\rightsquigarrow 2} \le \sum_j jc^j = O(1)$$

if constants are chosen properly.

4.1.3. Invariant. Now we show that the invariant is satisfied, using the LIM lemma. That is all that remains to prove our main theorem, Theorem 1.1.

THEOREM 1.1. *There is an algorithm and distribution on matrices* $\mathbf{\Phi}$ *satisfying*

$$\max_{\mathbf{x}} \mathbb{E}[\|\mathbf{\Phi x}\|_2 \,/\, \|\mathbf{x}\|_2] = 1$$

such that given $\mathbf{\Phi x}$*, the parameters, and a concise description of* $\mathbf{\Phi}$*, the algorithm returns* $\widehat{x}$ *with approximation error*

$$\|\mathbf{x} - \widehat{\mathbf{x}}\|_2^2 \le (1+\epsilon)\,\|\nu_1\|_2^2 + \epsilon\,\|\nu_2\|_2^2$$

with probability $3/4$. *The algorithm runs in time* $O((k/\epsilon)\log^{O(1)} N)$ *and* $\mathbf{\Phi}$ *has* $O((k/\epsilon)\log(N/k))$ *rows. In expectation, there are* $O(\log^2(k)\log(N/k))$ *nonzeros in each column of* $\mathbf{\Phi}$.

Proof. Note that the matrices described in section 3.1 have two additional features compared with the matrices in the LIM lemma. First, there is a single random permutation matrix $\boldsymbol{P}$ that multiplies all the error-correcting code, hashing, and sign-flip matrices, and second, the matrices in iteration j are multiplied by $c^{j/2}$, where c is an appropriate constant in the range $1/2 < c < 1$. Also note that ν_2 is not a priori guaranteed to be symmetric, as stipulated by the LIM lemma.

Consider the effect of ν_2. We would like to argue that the noise vector ν_2 is "distributed at random" by the permutation and each measurement is corrupted by $\frac{\|\nu_2\|_2^2}{m}$, approximately its fair share of $\|\nu_2\|_2^2$, where m is the number of measurements. 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 thus the Chernoff bound still applies [12].

For a more complete analysis, set $I = \{i : (\boldsymbol{P}^{-1}\nu_2)_i \ge \frac{4}{\sqrt{m}}\|\nu_2\|_2\}$, so $|I| \le \frac{m}{16}$. We say row i in the measurement matrix is *heavily corrupted* if $i \in I$. The measurement matrix is decomposed into B blocks (in the sense of error-correcting codes) of rows, and these blocks are used to identify a heavy hitter or to estimate a signal position value. For identification, we have a block size of $O(\log\log N)$ and an explicit encoding/decoding procedure, while for estimation we have a block size of $O(\log N)$ and a trivial encoding/decoding procedure. If some of the blocks are corrupted by the measurement noise, we may still be able to decode accurately. In order to ascertain how many blocks are heavily corrupted and what influence this has on the decoding procedure, we must analyze how the random permutation disperses I over the blocks.

Let $X_i = \mathbb{1}_{\{i\in I\}}$ and $\Lambda_1, \ldots, \Lambda_B$ be the set of indices of the blocks. Define $Y_k = \sum_{i\in\Lambda_k} X_i$ $(1 \le k \le B)$ to be the number of corrupted measurements in block k. The most desirable situation is that as in LIM Lemma, $Y_k \le \frac{|\Lambda_k|}{16}$ for all k, which is, however, extremely unlikely to happen. We could only expect something weaker. It follows from the Chernoff bound that

$$\Pr\left(|Y_k| \geq \frac{|\Lambda_k|}{6}\right) \leq e^{-0.05\cdot|\Lambda_k|}.$$

Since $|\Lambda_k| = \Omega(\log\log N)$ in the encoding portion of the identification matrix $\boldsymbol{D}$, the probability above is $\frac{1}{\log^{\Omega(1)} N}$. Furthermore, there are $O(\log N)$ rows in a block of the hashing portion of $\boldsymbol{D}$; thus the union bound gives $o(1)$ failure probability of $|Y_k| \leq \frac{|\Lambda_k|}{6}$ for all k corresponding to a specific row in the hashing matrix.

Suppose there are g good hashing rows, $g = \sum g_t$, where $g_t = \Omega(j)$ (recall that the identification matrix $\boldsymbol{D}$ has $\Theta(j)$ layers) is the number of good rows containing heavy hitter t. From the negative association, the probability that $\frac{4}{5}g_t$ good rows are heavily corrupted is at most $o(1)^{g_t} = O(c^{-j})$ for some constant c, in which case we say the heavy hitter t is ruined. By the Markov inequality, only a small fraction of heavy hitters are ruined except with small probability $O(c^{-j})$, which is sufficient for recovery in the jth iteration. Similar arguments work for the estimation matrix, where heavy hitters and non-heavy hitters are discussed separately. Summing the failure probability over j, we conclude that except with probability $o(1)$, the post-measurement noise ν_2 will be dispersed favorably, i.e., the blocks corresponding to most heavy hitters have at most $1/6$ of the measurements being heavily corrupted.

Next, we claim that with the measurement noise dispersed favorably, we only need an increase of a constant factor in the number of measurements to accommodate the noise. Let $\{X_i\}_{i=1}^m$ be i.i.d. Bernoulli random variables with parameter p that denote the failure of measurement i (in which case $X_i = 1$). Let $\lambda > p$ be the thresholding constant. The Chernoff bound tells us that the failure probability of a fraction λm of all the measurements is

$$\Pr\left(\sum_{i=1}^m X_i \geq m\lambda\right) \leq e^{-C(\delta)mp},$$

where $\delta = \frac{\lambda}{p} - 1$ and $C(\delta)$ is a constant depending on δ. With postmeasurement noise, a fraction θ of X_i's are corrupted and not usable, where θ is sufficiently small such that $\theta + p < \lambda$. (For instance, following the above constants, we have that $\theta = \frac{1}{6}$ and we can adjust p and λ in the arguments of the case $\nu_2 = 0$ such that $\theta + p < \lambda$.) The threshold becomes $m(\lambda - \theta)$ instead of $m\lambda$, and thus

$$\Pr\left(\sum_{i=1}^m X_i \geq m(\lambda - \theta)\right) \leq e^{-C(\zeta)mp},$$

where $\zeta = \frac{\lambda-\theta}{p} - 1$. It is now clear that m needs to increase by only a constant factor, namely, $\frac{C(\delta)}{C(\zeta)}$, to keep the probability bound unchanged. Henceforth, we may assume ν_2 corrupts each measurement in $\boldsymbol{\Phi}\mathbf{x}$ by at most $16\|\nu_2\|^2/m$.

We turn now to the complete proof of the invariant (Claim 1) with postmeasurement noise. With assumed normalization $\|\nu_1\|_2 = \|\nu_2\|_2 = 1$, we have that $\|\nu_1^{(0)}\|_2 = 1$.

In iteration j, we make ℓ of the LIM lemma equal to $k/2^j$, η of the LIM lemma is $\Theta(\epsilon\beta^j)$, and $\delta = 2^{-j}$, for $\beta < 1$ to be specified below. It is straightforward to confirm that $\|\sigma^{(j+1)}\|_0 \leq k/2^{j+1}$, provided the invariant held at the previous iteration. We now turn to $\|\nu_1^{(j+1)}\|_2$.

At the beginning of the jth iteration,

$$\left\|\nu_1^{(j)}\right\|_2^2 \leq 1+\epsilon\left(1-\left(\frac{3}{4}\right)^j\right).$$

This means that $1 \leq \|\nu_1^{(j)}\|_2^2 \leq 2$ remains unchanged up to factor 2. By the LIM lemma and the discussion at the beginning of section 4.1.2, each repetition gives with high probability an estimate with

$$\left\|\nu_1^{(j+1)}\right\|_2^2 - \left\|\nu_1^{(j)}\right\|_2^2 \quad \text{at most} \quad \epsilon\beta^j\left(\left\|\nu_1^{(j)}\right\|_2^2 + 16c^{-j}\left\|\nu_2\right\|_2^2\right).$$

It follows that the median over repetitions of this estimate satisfies the same bound with high probability. Since $1 \ll c^{-j}$, it follows that $1+c^{-j} \approx c^{-j}$. If we put $\beta \approx 5/8$ and $c \approx 5/6$, the invariant is satisfied.

We have proved that the algorithm returns $\widehat{\mathbf{x}}$ with approximation error

$$\|\mathbf{x}-\widehat{\mathbf{x}}\|_2^2 \leq (1+\epsilon)\left\|\nu_1\right\|_2^2 + 16\epsilon\left\|\nu_2\right\|_2^2.$$

Now, replace 16ϵ by ϵ to achieve the desired form of error bound while introducing only a constant to the time cost. □

4.2. Efficiency.

4.2.1. Number of measurements. The number of measurements in iteration j is computed as follows. There are $\log(1/\delta) = O(j)$ parallel repetitions in iteration j. They each consist of $O(k/(\epsilon\beta^j 2^j)\log(N/k))$ measurements, where $\beta = 5/8$. That is, the number of measurements is

$$\Theta\left(\frac{jk}{\epsilon}\left(\frac{4}{5}\right)^j \log(N/k)\right) = \frac{k}{\epsilon}\log(N/k)\left(\frac{4}{5}+o(1)\right)^j.$$

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

Note that the dimension of the random permutation matrix $\boldsymbol{P}$ matches the number of rows, $O((k/\epsilon)\log(N/k))$.

4.2.2. Encoding, decoding, and update time. The encoding time is bounded by N times the number of nonzeros in each column of the measurement matrix. This was analyzed above in section 4.1; there are $\log(j)\log(N/k)$ nonzeros per column in iteration j for $j \leq O(\log(k))$, so the total is $\log^2(k)\log(N/k)$ nonzeros per column. This is suboptimal by the factor $\log^2(k)$. By comparison, however, some proposed methods use dense matrices, which are suboptimal by the exponentially larger factor k.

When constructing the matrix for measuring the original signal or some intermediate representation, our algorithm will need to find, quickly, the bucket to which an index i is hashed and a codeword for i, where i is in the range $1 \leq i \leq N$. Note that it is crucial that we use $O(\log(N/B))$ bits for the codeword to meet the sketch length lower bound $O(k\log(N/k))$ (instead of $O(k\log N)$), where B is the number of buckets, and not $\log(N)$ bits. This means we need to find codewords for just the i's hashed to a particular bucket. Upon decoding, we need to be able to find i from its codeword, quickly.

We can use a pseudorandom number generator that hashes i to a bucket j if $jN/B \leq ai+b \bmod N < (j+1)N/B$ for random a and b. Then we encode i by

$E(ai + b - jN/B)$, assuming quick encoding for numbers in the contiguous range 0 to $N/B - 1$. To decode, knowing j, we first recover $ai + b - jN/B$, whence we easily recover $ai + b$ and subsequently i. Define hash function $f(i) = ai + b$ on $\mathbb{Z}_N$. The nonzeroes at positions $i_1, \ldots, i_h$ in a row of the hashing matrix are ordered such that $f(i_1) < f(i_2) < \cdots < f(i_h)$. An example is given in Figure 3.1.

Another issue is the time to find and to encode and 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, encoding, and decoding the code. These tasks are straightforward in that much time. Alternatively, we can use a look-up table of size polylog(N) to decode a code in $O(1)$ time.

5. Conclusion and open problems.

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 leave the following problems open and make conjectures:

- We do not think that the current approach can be extended to the "for all" signal model in $\ell_1 \leq C\ell_1$ and $\ell_2 \leq (C/\sqrt{k})\ell_1$ error metric guarantees (all current sublinear algorithms use at least one factor $O(\log N)$ additional measurements), and we leave open the problem of designing a sublinear time recovery algorithm and a measurement matrix with an optimal number of rows for this setting, under any suitable error metric.
- Our current algorithm outputs a desirable approximation with constant probability. It remains to improve the success probability to high probability, i.e., $\geq 1 - 1/N$, which is the case of several previous algorithms listed in Table 1.1.
- It remains to improve the number of nonzeros in the columns of our measurement matrix from
$$O(\log^2(k)\log(N/k)) \quad \text{to} \quad O(\log(N/k)).$$
- To obtain the index of a heavy hitter, we divide a message of length $O(\log(N/k))$ into pieces of length $O(\log\log N)$ and encode/decode each piece individually. It is possible to use a better error-correcting code with codewords of length $O(\log(N/k))$ with polynomial recovery time, such as low-density parity check codes (LDPC). Adopting such a scheme may also simplify the analysis. However, nearest-neighbor decoding is easier and can take only constant time by using a look-up table of size polylog N.
- A straightforward way to encode a signal takes time proportional to the signal support size times the number of nonzeros per column of the measurement matrix. This is the case with our algorithm and the relevant related work. If the measurement matrix is properly structured, however—for example, if it consists of rows of a Fourier matrix in arithmetic progression—it is possible to multiply the measurement matrix by a vector faster than the trivial algorithm. Thus it would be interesting to improve the encode time of our algorithm (assuming a dense signal), even if one cannot improve the number of nonzeros per column.
- As discussed earlier, Do Ba et al. [2] give a lower bound of $\Omega(k\log(N/k))$ for the number of measurements for constant ϵ, and Wainwright [18] gives a lower bound of $\Omega((k/\epsilon)\log(N/k))$, under certain restrictions, for problems and in models that are not exactly the same as ours. These lower bounds can

be regarded as characterizing obstacles in our setting. It remains to give a lower bound for the dependence on ϵ and to meet it.

Note that the number of measurements used by our algorithm is

$$O((k/\epsilon)(\log(\epsilon N/k) + \log(1/\epsilon))).$$

Above we quoted the larger expression

$$O((k/\epsilon)\log(N/k)).$$

Note that we may assume that $\epsilon > k/N$ since otherwise we may use $k/\epsilon > N$ measurements and recover trivially. Also, if $\epsilon^{1.1} > k/N$, then

$$\log(1/\epsilon) \leq O(\log(\epsilon N/k)).$$

Thus the number of measurements used by our algorithm is $O((k/\epsilon)\log(\epsilon N/k))$ except for the (rare?) cases of $k/N < \epsilon < (k/N)^{.9}$. Nevertheless, we conjecture that the lower bound is $\Omega((k/\epsilon)\log(N/k))$, which quantitatively matches the result in [18], and that this holds even if one of ν_1 and ν_2 is zero.

The dependence on ϵ seems to increase from $1/\epsilon$ to $1/\epsilon^2$ for the recovery of an exactly k-sparse representation. It would be good to have a precise characterization of the lower bound here as well.[2]

Acknowlegement. The authors would like to thank the anonymous referees for their helpful comments and suggestions.

REFERENCES

[1] N. Alon, Y. Matias, and M. Szegedy, *The space complexity of approximating the frequency moments*, J. Comput. System Sci., 58 (1999), pp. 137–147.

[2] K. Do Ba, P. Indyk, E. Price, and D. Woodruff, *Lower bounds for sparse recovery*, in Proceedings of the ACM-SIAM Symposium on Discrete Algorithms, 2010.

[3] R. Berinde, A. Gilbert, P. Indyk, H. Karloff, and M. Strauss, *Combining Geometry and Combinatorics: A Unified Approach to Sparse Signal Recovery*, Allerton Press, Allerton, IA, 2008.

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

[5] M. Charikar, K. Chen, and M. Farach-Colton, *Finding frequent items in data streams*, in Proceedings of the International Colloquium on Automata, Languages, and Programming, 2002.

[6] R. Clifford, K. Efremenko, E. Porat, and A. Rothschild, *k-mismatch with don't cares*, in Proceedings of the European Symposium on Algorithms, 2007, pp. 151–162.

[7] A. Cohen, W. Dahmen, and R. DeVore, *Compressed sensing and best k-term approximation*, J. AMS, 22 (2009), pp. 211–231.

[8] G. Cormode and S. Muthukrishnan, *What's hot and what's not: Tracking most frequent items dynamically*, in Proceedings of the ACM Symposium on Principles of Database Systems, 2003, pp. 296–306.

[9] G. Cormode and S. Muthukrishnan, *Improved data stream summaries: The count-min sketch and its applications*, in Proceedings of the Annual Conference on Foundations of Software Technology and Theoretical Computer Science, 2004.

[10] G. Cormode and S. Muthukrishnan, *Combinatorial algorithms for compressed sensing*, in Proceedings of the 40th Annual Conference Information Sciences and Systems, Princeton, NJ, 2006.

[2] Very recently, Price and Woodruff [17] have proved a lower bound of $\Omega((k/\epsilon)\log(N/k))$ measurements for nonsparse output and $\Omega(k/\epsilon^2)$ measurements for sparse outputs, resolving much of this last open problem.

[11] D. L. Donoho, *Compressed Sensing*, IEEE Trans. Inform. Theory, 52 (2006), pp. 1289–1306.

[12] D. Dubhashi and V. Priebe Desh Ranjan, *Negative Dependence Through the FKG Inequality*, Research report MPI-I-96-1-020, Max-Planck-Institut für Informatik, Saarbrucken, Germany, 1996.

[13] Y. Erlich, K. Chang, A. Gordon, R. Ronen, O. Navon, M. Rooks, and G. J. Hannon, *DNA sudoku—harnessing high-throughput sequencing for multiplexed specimen analysis*, Genome Research, 19 (2009), pp. 1243—1253.

[14] A. C. Gilbert, M. J. Strauss, J. A. Tropp, and R. Vershynin, *One sketch for all: Fast algorithms for compressed sensing*, in Proceedings of the ACM Symposium on Theory of Computing, 2007, pp. 237–246.

[15] P. Indyk and M. Ruzic, *Near-optimal sparse recovery in the L_1 norm*, in Proceedings of the Foundations of Computer Science, 2008, pp. 199–207.

[16] D. Needell and J. A. Tropp, *CoSaMP: Iterative signal recovery from incomplete and inaccurate samples*, Appl. Comput. Harmonic Anal., 26 (2009), pp. 301–321.

[17] E. Price and D. Woodruff, $(1+\epsilon)$*-approximate sparse recovery*, in Proceedings of the Foundations of Computer Science, 2011, pp. 295–304.

[18] M. Wainwright, *Information-theoretic bounds on sparsity recovery in the high-dimensional and noisy setting*, IEEE Trans. Inform. Theory, 55 (2009), pp. 5728–5741.

[19] 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*, Appl. Optics, 44 (2005), pp. 4306–4314.

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